From the early classification of time series to machine learning-based early decision-making

Youssef Achenchabe

THESE DE DOCTORAT

NNT : 2022UPASB070

université PARIS-SACLAY

From the early classification of time series to machine learning-based early decision-making

De la classification précoce des séries temporelles à la prise de décision précoce basée sur l'apprentissage machine

Thèse de doctorat de l'université Paris-Saclay

École doctorale n°581 : agriculture, alimentation, biologie, environnement, santé (ABIES)
Spécialité de doctorat : mathématiques appliquées
Graduate School : Biosphera. Référent : AgroParisTech

Thèse préparée dans l'UMR **MIA-Paris-Saclay** (Université Paris-Saclay, AgroParisTech, INRAE), sous la direction d'**Antoine CORNUÉJOLS**, Professeur et le co-encadrement d'**Alexis BONDU**, Chercheur

Thèse soutenue à Paris-Saclay, le 25 novembre 2022, par

Youssef ACHENCHABE

Composition du Jury

Membres du jury avec voix délibérative

Romain TAVENARD Professeur, Université de Rennes 2	Président & Rapporteur
Tony BAGNALL Professeur, University of East Anglia (Royaume-Uni)	Rapporteur & Examinateur
Albert BIFET Professeur, Télécom Paris	Examinateur
Eric GAUSSIER Professeur, Université Grenoble Alpes	Examinateur
Usue MORI Maîtresse de Conférences, University of the Basque Country (Espagne)	Examinatrice

université PARIS-SACLAY

ÉCOLE DOCTORALE
Agriculture, alimentation, biologie, environnement, santé (ABIES)

Titre : De la classification précoce des séries temporelles à la prise de décision précoce basée sur l'apprentissage machine

Mots clés : séries temporelles, décisions précoces, classification, apprentissage machine, décisions sensibles aux coûts.

Résumé : Dans de nombreuses situations réelles, nous devons prendre des décisions précoces sans avoir une connaissance complète du problème. Le problème auquel sont confrontés les décideurs est que, généralement, plus la décision est retardée, plus le résultat probable est clair, mais aussi plus le coût sera élevé, car des décisions précoces permettent de mieux se préparer. Ce compromis précocité-précision est principalement présent dans le problème de la classification précoce des séries temporelles (ECTS). Un framework générique sensible aux coûts a été présenté pour résoudre ce problème, et une nouvelle implémentation a été proposée. Cependant, ce problème souffre de multiples limitations identifiées lors de cette thèse. Deux limites ont été abordées. La première est l'irrévocabilité des décisions. Un nouvel algorithme à régime révocable a été proposé pour modifier la décision prise en cas de réception de nouvelles mesures de la série remettant en cause l'ancienne décision. La deuxième limite est que l'ECTS est limité à des séries chronologiques de longueur finie et une seule étiquette associée à la série chronologique complète. Le nouvel algorithme proposé est capable de traiter des séries chronologiques sans limites temporelles et où différents événements surviennent, éventuellement de longueurs différentes, chacun avec son étiquette de classe. Enfin, un problème général sous le nom de ML-EDM (prise de décision précoce basée sur l'apprentissage machine) a été formalisé et dix défis ont été proposés à la communauté scientifique pour des recherches plus approfondies.

Title : From the early classification of time series to machine learning-based early decision-making

Keywords : time series, early decisions, classification, machine learning, cost-sensitive decisions

Abstract : In numerous real-world situations, we have to make early decisions without complete knowledge of the problem. The issue facing the decision makers is that, usually, the longer the decision is delayed, the clearer the likely outcome, but also the higher the cost that will be incurred if only because earlier decisions allow one to be better prepared. This earliness-accuracy trade-off is mainly involved in the Early Classification of Time Series (ECTS) problem. A generic cost-sensitive framework has been presented to solve this problem, and a novel implementation has been proposed. However, the ECTS problem suffers from multiple limitations identified in this thesis. Two limitations have been tackled. The first one is the irrevocability of decisions. A novel revocable regime algorithm has been proposed to change the decision taken in case of receipt of new measurements of the series that question the old decision. The second limitation is that ECTS is limited to time series with finite length and a single label associated with the complete time series. The novel proposed algorithm is capable of dealing with time series with no time bounds and where different events arise, possibly of different lengths, each with its class label. Finally, a generic problem under the name ML-EDM (machine learning-based early decision-making) with the rest of the ECTS limitations have been suggested to the scientific community for further research.

Résumé de la thèse

Dans de nombreuses situations réelles, nous devons prendre des décisions précoces sans avoir une connaissance complète du problème. Le problème auquel sont confrontés les décideurs est que, généralement, plus la décision est retardée, plus le résultat probable est clair, mais aussi plus le coût sera élevé, car des décisions précoces permettent de mieux se préparer. Ce compromis précocité-précision est principalement présent dans le problème de la classification précoce des séries temporelles (ECTS). Nous présentons un critère d'optimisation qui prend en compte à la fois le coût de mauvaise classification et le coût du retardement de la décision. Sur la base de ce critère d'optimisation, nous avons dérivé une famille d'algorithmes non myopes qui tentent d'anticiper le gain d'information futur attendu en équilibre avec le coût de l'attente, C'est-à-dire que les décisions prises sont basées sur ce qui peut se passer dans le futur. Dans une classe d'algorithmes, non supervisés, les attentes utilisent le regroupement de séries temporelles, tandis que dans une deuxième classe, basée sur la supervision, les séries temporelles sont regroupées en fonction Des expérimentaions approfondies sur des ensembles de données réelles en utilisant une large gamme de fonctions de coût d'attente montrent que les algorithmes présentés sont capables de résoudre le compromis entre la précocité et la précision, avec la partition supervisée. Les approches supervisées basées sur des partitions s'en sortent mieux que celles non supervisées basées sur des partitions. En outre, toutes ces méthodes sont plus performantes dans une grande variété de conditions qu'une méthode de l'état de l'art basée sur une stratégie stratégie myopique qui est reconnue comme étant très compétitive. De plus, nos expériences montrent que la caractéristique non-myopique des approches proposées explique en grande partie les performances obtenues. Cependant, Le problème de la classification précoce des séries temporelles souffre de multiples limitations identifiées lors de cette thèse.

Deux limites ont été abordées. La première est l'irrévocabilité des décisions. Dans le problème de l'ECTS une fois la décision prise, le processus de collecte de nouvelles mesures est terminé, et une étiquette de classe est prédite. Dans le régime révocable nous continuons à recevoir de nouvelles mesures même si une étiquette de classe a été prédite. Dans de nombreuses situations, cependant, il est possible de prendre une décision puis de décider de la modifier après avoir obtenu de nouvelles informations. Le changement peut être coûteux mais néanmoins justifié car il semble susceptible de conduire à un résultat bien meilleur. C'est le cas, par exemple, lorsqu'un événement en plein air est annulé en raison d'un changement radical des prévisions météorologiques, ou lorsqu'un médecin révise ce qui semble maintenant être un mauvais diagnostic. Le problème consiste maintenant à identifier les séquences de décision optimales compte tenu d'une série de mesures entrantes et des différents coûts existants. L'impact d'une telle stratégie intelligente révocable pourrait avoir sur l'entretien des prévisions, les unités de soins intensifs, les voitures autonomes et bien d'autres domaines d'application où des décisions doivent être prises en optimisant les coûts des décisions erronées et des retards . Nous avons proposé un nouvel algorithme à régime révocable pour modifier la décision prise en cas de réception de nouvelles mesures de la série remettant en cause l'ancienne décision. Dans cet algorithme, nous introduisons le coût de changement de décision ou de révocation. L'objectif étant de modéliser le coût de révocation aux instants futurs. L'intuition est si une décision est susceptible de changer dans le futur, il vaut mieux la changer le plus tôt possible.

La deuxième limite est que l'ECTS est limité à des séries chronologiques de longueur finie et une seule étiquette associée à la série chronologique complète. Dans le nouveau setting ECOTS que nous présentons, pendant un temps indéfini, de nouvelles mesures sont reçues à chaque pas de temps, et la série temporelle entrante consiste en de multiples morceaux ayant des étiquettes de classe différentes. Lors de la phase de test, l'utilisateur ne sait pas quand une partie spécifique avec une classe différente commence et quand elle se termine, contrairement au problème classique de l'ECTS où l'utilisateur sait que la série temporelle entrante se termine à un

certain moment et où une seule étiquette de classe est associée à la série temporelle complète. Nous définissons correctement le problème ECOTS et, ensuite, nous présentons une méthodologie pour adapter toute approche ECTS au problème ECOTS. En conséquence, nous montrons i) comment le rôle des classificateurs doit être repensé et transformé ; ii) comment le compromis entre la précision et l'exactitude se traduit dans le nouveau scénario et ce que devient le système de déclenchement des décisions. Parmi la grande variété d'applications que cette nouvelle approche ouvre, nous développons un cas d'utilisation de maintenance prédictive qui optimise les temps de déclenchement des alarmes, démontrant ainsi la puissance de cette nouvelle approche.

Finalement, nous proposons des directions de recherche pour étendre l'ECTS à un problème plus générique que nous appelons Prise de décision précoce basée sur l'apprentissage automatique (ML-EDM). Nous avons proposé dix défis proposés afin de développer des approches ML-EDM pour un large éventail de problèmes. Certains d'entre eux ont été abordés dans cette thèse, le reste est laissé à la communauté scientifique pour de futures recherches. Ces défis sont regroupés en différentes catégories, notamment les tâches d'apprentissage, les types de données, la prise de décision précoce en ligne et les décisions révocables. Ensuite, quelques exemples d'applications des techniques ML-EDM sont fournis. Enfin, nous concluons avec les perspectives de développement du domaine ML-EDM dans les années à venir.

Acknowledgements

First and foremost, I want to thank my Ph.D. thesis advisor, Dr. Alexis Bondu and my thesis director Pr. Antoine Cornuéjols. I would like to express my deep feelings of gratitude to them. It would have been quite impossible to carry on the research work and make it into the final shape of a thesis without their guidance, kindness and sympathetic encouragement. I appreciate all their contributions of time, brilliant ideas, long brain storming sessions, and jokes to make my Ph.D. experience productive, stimulating, and joyful.

I also thank my thesis reviewers, Pr. Tony Bagnall and Pr. Romain Tavenard, for taking the time to read my thesis and providing me with insightful remarks. In addition, my thanks go to the whole jury members.

My special thanks go to Vincent Lemaire for accepting to be an invited researcher for my thesis defense, for his interesting comments on the research work done during this thesis and for proofreading this thesis at various stages, including the final draft. I would also like to thank my thesis committee members, including Jesse Read, Pierre-françois Marteau, and Wolfram Liebermeister for their valuable advice. I sincerely thank Albert Bifet, Fabrice Clérot, João Gama, Georges Hébrail, and Pierre-françois Marteau for their contributions to this thesis through the position paper.

The BRAIN team members at Orange Group and Ekinoks (ex LINK) at INRAe have contributed immensely to my personal and professional time. I would like to thank everyone for their kindness, presence, and exciting conversations.

Further, my sincere thanks go to my brother Yassine and my uncle Khalid for their immense help and advice. Many thanks to all my friends from *"Cité Internationale Universitaire de Paris"*, and close friends, including Abdelkoddous, Abdelmoughit, Anas, Evelyn and Naoufal, with whom I spent quality time and who helped me refresh my mind and ideas.

Finally, my heartiest thanks go to my parents who raised me with a love of knowledge and supported me in all my pursuits. I want to thank them for their love, encouragement, and everyday prayers, which made it possible to complete this thesis.

"Prediction is very difficult, especially about the future."

Niels Bohr

PARIS-SACLAY UNIVERSITY

Abstract

Doctor of Philosophy

From the early classification of time series to machine learning-based early decision-making

by Youssef ACHENCHABE

In numerous real-world situations, we have to make early decisions without complete knowledge of the problem. The issue facing the decision makers is that, usually, the longer the decision is delayed, the clearer the likely outcome, but also the higher the cost that will be incurred if only because earlier decisions allow one to be better prepared. This earliness-accuracy trade-off is mainly involved in the Early Classification of Time Series (ECTS) problem. A generic cost-sensitive framework has been presented to solve this problem, and a novel implementation has been proposed. However, the ECTS problem suffers from multiple limitations identified in this thesis. Two limitations have been tackled. The first one is the irrevocability of decisions. A novel revocable regime algorithm has been proposed to change the decision taken in case of receipt of new measurements of the series that question the old decision. The second limitation is that ECTS is limited to time series with finite length and a single label associated with the complete time series. The novel proposed algorithm is capable of dealing with time series with no time bounds and where different events arise, possibly of different lengths, each with its class label. Finally, a generic problem under the name ML-EDM (machine learning-based early decision-making) with the rest of the ECTS limitations have been suggested to the scientific community for further research.

Contents

List of Figures

List of Tables

List of Abbreviations

ECTS	**E**arly **C**lassification of **T**time **S**eries
ECOTS	**E**arly **C**lassification of **O**pen **T**time **S**eries
ECONOMY	**E**arly **C**lassification for **O**ptimized and **NO**n **MY**opic online decision making
LUPI	**L**earning **U**nder **P**riviliged **I**nformation
WSL	**W**eakly **S**upervised **L**earning

List of Notations

$x_t \in \mathbf{R}^p$	Time series measurement at timestamp t of dimension p.
$\mathbf{x}_T = \langle x_1, \ldots, x_T \rangle$	Complete time series.
$\mathbf{x}_t = \langle x_1, \ldots, x_t \rangle$	Incomplete time series of length t $\leq$ T.
$y \in \mathcal{Y}$	True class label.
$\hat{y} \in \mathcal{Y}$	Predicted class label.
$\mathrm{C}_m(\hat{y}\|y)$	Misclassification cost when predicting $\hat{y}$ while true class label is y.
$\mathrm{C}_d(t)$	Delay cost at timestamp t.
$\mathrm{C}_{cd}(y''\|y')$	The cost of changing the prediction from y′ to y″.
$\mathcal{D}_{train} = \{(\mathbf{x}_T^i, y^i)\}_{i=1}^m$	Training set for a classical ECTS approach.
$\mathcal{D}_{val} = \{(\mathbf{x}_T^i, y^i)\}_{i=1}^v$	Validation set for a classical ECTS approach.
$\mathcal{D}_{train}^t = \{(\mathbf{x}_t^i, y^i)\}_{i=1}^m$	Training set of time series truncated to their first t measurements.
$\mathcal{H}$	Hypothesis space.
$h_t \in \mathcal{H}$	Classifier that takes $\mathbf{x}_t$ as input and predicts $\hat{y} \in \mathcal{Y}$ associated with $\mathbf{x}_T$.
$\{h_t(\cdot)\}_{t \in [1,T]}$	Collection of classifiers for each time step.
I_t^i	The i-th interval at timestamp t.
M_t^{t+1}	Transition matrix from timestamp t to timestamp t+1.
$f_\tau(\mathbf{x}_t)$	Estimated cost of decision at timestamp t+τ.
K	Number of groups in the ECONOMY framework.
$\hat{y}_t \in \mathcal{Y}$	Predicted class label at time stamp t.
$\mathbb{D}_T$	The set of all possible sequences of class labels o f maximum length T.
$\mathcal{D}_{train}^{\text{open}} = \{(\mathbf{x}^i, y^i)\}_{i=1}^m$	Training set of open time series, labeled for each timestamp.
$\eta \in \mathbb{Z}$	Prediction horizon.
w	Window size.
$\mathbf{x}_{(t-w,t)} = \langle x_{t-w}, \ldots, x_t \rangle$	Subsequence of time series $\mathbf{x}$ between $t - w$ and t.
η_m	Minimal prediction horizon.
η_M	Maximal prediction horizon.
$\mathcal{G}$	Set of groups of time series.
$\mathfrak{g}_k \in \mathcal{G}$	Group of time series.
$\mathfrak{g}_k^t \in \mathcal{G}$	Group of time series at timestamp t.
$\mathcal{D}_\ell$	Sequence of ℓ predictions.

Chapter 1

Introduction

In numerous real situations, we have to make *early* decisions in the absence of *complete knowledge* of the problem at hand. For example, such decisions are necessary for medicine (Mathukia et al., 2015) when a physician must make a diagnosis, possibly leading to an urgent surgical operation, before having the results of all scheduled medical tests. Another example is Eisenhower, in June 1944, having to decide when to launch the landing on the French coast (Eisenhower, 1944). He had an imperfect knowledge of the weather conditions. The longer he waited, the more precise they became, allowing for a more informed decision: to launch the landing today or wait for another day, but the more difficult it became to ensure that all arrangements would be met and that the enemy remained unaware of the danger.

In such situations, the issue facing the decision-makers is that most of the time, the longer the decision is delayed, the clearer is the likely outcome (e.g., the critical or not critical state of the patient) but also the higher the cost that will be incurred if only because decisions taken earlier allow one to be better prepared. In everyday life, we thus seek to decide at the moment that seems to be the best compromise between the *earliness* and the *accuracy* of our decision.

This **earliness *vs.* accuracy** dilemma is a key part of many decision-making scenarios and is especially involved in the problem of Early Classification of Time Series (ECTS). However, as we will see, it takes place in a larger perspective.

1.1 Early Classification of Time Series: a particular case

This section defines the problem of early classification of time series, and a set of identified limitations are presented.

1.1.1 Problem setting

The ECTS problem consists in finding the *optimal time* to trigger the class prediction of an input finite-length time series observed over time. As successive measurements provide more and more information about the incoming time series, ECTS algorithms aim to optimize online the trade-off between the *earliness* and the *accuracy* of their decisions.

More formally, the individuals considered are time series of *finite length* T. At testing time, the measurements of the incoming time series are received over time, and the history of measurements available at time t is denoted by $\mathbf{x}_t = \langle x_1, \ldots, x_t \rangle$. It is assumed that each time series can be ascribed to some class $y \in \mathcal{Y}$, and the task

is to predict the class of each incoming time series as early as possible because a time-increasing cost must be paid when the decision is triggered. In the ECTS problem, a single decision is triggered for each incoming time series, which is irrevocable and final. An ECTS approach is generally made of two main components (see Figure 1.1):

- (*i*) an *hypothesis*[1] $h \in \mathcal{H}$ capable of predicting the class $y \in \mathcal{Y}$ of the incoming series at any time, such that $h(\mathbf{x}_t) = \hat{y}, \forall t \in [1, T]$.
- (*ii*) a *triggering strategy* capable of making decisions at the right moments, denoted by *Trigger*.

Both the hypothesis and the triggering strategy are learned in batch mode (i.e., offline), using a training set made of complete time series with their associated labels.

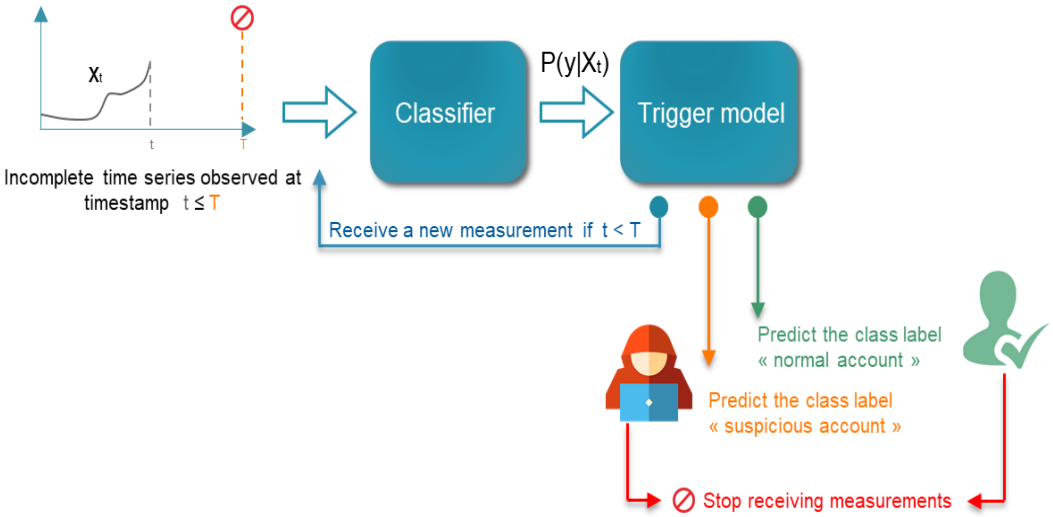


FIGURE 1.1: General schema of ECTS approaches, with a special case illustrating the early classification of normal vs suspicious account from time series network data

1.1.2 Limitations of the ECTS problem

While ECTS covers a wide range of applications, it does not exhaust all cases where a *Machine Learning* model can be applied on data acquired over time, and where the trade-off between the *earliness* and the *accuracy* of decisions must be optimized. Indeed, ECTS, as defined above, is limited to:

- a classification problem.
- an available training set which contains completely and properly labeled time series.
- a decision deadline T that is finite, fixed and known.
- only one decision for each incoming time series.
- decisions that once made can never be reconsidered.
- fixed decision costs which do not depend on the triggering time and the decisions made.

[1] An hypothesis is a candidate model which approximates the concept $P(y|\mathbf{x}_t)$.

1.2 Thesis story

In summary, the primary purpose of this thesis is to question the assumptions of the ECTS problem listed in 1.1.2, identify the main challenges of the field, propose solutions to some of them and finally formalize an extension of ECTS towards a more generic problem, that we call *Machine Learning based Early Decision-Making* (ML-EDM), for which an official Github page has been created: `http://www.github.com/ML-EDM/`

In this thesis, the main focus is put on the *triggering strategy* described in Section 1.1.1. Developping novel time series classification algorithms is out of the scope of this thesis, we hypothesize that the time series classifier is supposed to be trained independently and given by the user.

In the following, we highlight in a chronological order an overview of the crucial questions of this thesis, provided with some response elements that will be detailed later in the next chapters.

Could the state of the art of ECTS be improved? How to evaluate ECTS approaches?

The approaches developed in the literature can be divided into cost-based ones and the ones that make implicit assumptions about the cost of delay. In this thesis, we claim that costs are essential for the decision-making process, especially the cost of misclassification (cost of a false positive and a false negative in the case of binary classification) and the cost of delaying the decision. These costs depend on the application domain, and the user should provide them as prior information. The ECONOMY -K method presented in (Dachraoui, Bondu, and Cornuéjols, 2015) has the merit of taking into account explicitly these costs in the decision process. In this thesis, we show that this method could be extended to a general framework for ECTS, and we propose a novel approach that implements this framework. Extensive experiments on a benchmark of 45 datasets from different domain areas show the superiority of the proposed method ECONOMY-γ over the state of the art of the field. Researchers in the literature used to evaluate ECTS approaches, by computing earliness and accuracy independently. We propose the average cost incurred while using an ECTS approach as the evaluation criterion. More details are presented in Chapter 3.

In the following, the questions concern the limitations of the ECTS problem that have been addressed in this thesis.

If decisions could be revocable, what would be, if needed, the optimal moment to revoke a decision?

Until now, the ECTS problem has been dealt with by considering only irrevocable decisions. This new setting is slightly different and addresses this limitation; the decision can be revoked at any time when new measurements arrive and question the validity of the last decision taken. It is a challenging problem since the algorithm should deal with the stability-reactivity dilemma and the accuracy-earliness one. It should be able to revoke decisions only if it improves the performance. In this thesis, in order to formalize and tackle this problem, we propose a new cost-based framework and derive two new approaches from it. The first approach does not

explicitly consider the cost of changing decisions, while the second does. Extensive experiments are conducted to evaluate these approaches on a large benchmark of real datasets. The empirical results obtained convincingly show (*i*) that the ability to revoke decisions significantly improves performance over the irrevocable regime and (*ii*) that taking into account the cost of changing decisions brings even better results in general. More details are presented in Chapter 4.

If time series were open (indefinite length), and labeled in portions, how to optimize online the accuracy-earliness tradeoff?

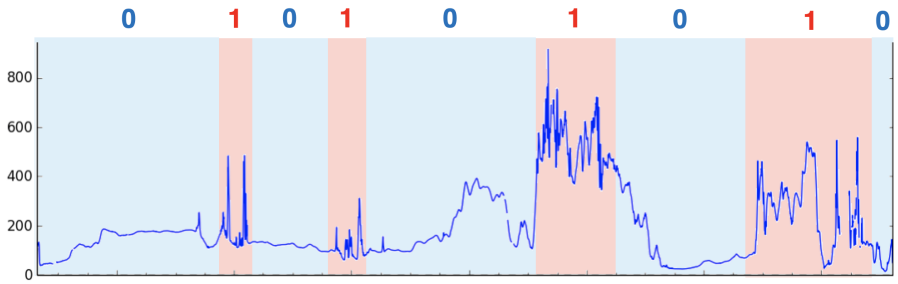


FIGURE 1.2: Example of a part of an open time series where events of possibly different lengths are here labeled with '0s' and '1s'.

This scenario addresses two limitations of ECTS, which are i) time series are of finite length; ii) one unique label is associated with the complete time series. In this thesis, for the first time, we investigate such a trade-off when events of different classes occur in a streaming fashion, with no predefined end. In the *Early Classification in Open Time Series* problem (ECOTS), the task is to predict events, i.e., their class and time interval, at the moment that optimizes the accuracy vs. earliness trade-off. Interestingly, we find that ECTS algorithms can sensibly be adapted in a principled way to this new problem. We illustrate our methodology by transforming two state-of-the-art ECTS algorithms for the ECOTS scenario. More details are presented in Chapter 5.

Can we define a more generic problem than ECTS?

In this thesis, we claim that the earliness-accuracy trade-off is involved in a larger perspective than the one defined in ECTS. It takes place whenever a machine learning model is used on data acquired over time, and a decision should be taken as soon as possible. We propose a new problem that we call *Machine Learning based Early Decision-Making* (ML-EDM) that defines rigorously the learning task of a trigger system in a generic setting as well as its positioning to classical decision-making problems in the literature. More details are presented in Chapter 6.

What are the promising challenges that the scientific community needs to tackle in order to develop ML-EDM approaches?

At the end of this thesis, a particular effort has been made to identify the main challenges that the community needs to overcome in order to open the path to more applications. In the following, a list of the proposed challenges is given in Table 1.1. Some of the challenges were addressed in this thesis, and the other ones are left to

the scientific community for future work. The challenges are detailed in Chapter 6 with their associated practical implications.

ML-EDM challenges	Addressed
#1: Extending non-myopia to unsupervised approaches	
#2: Addressing other supervised learning tasks	
#3: Early weakly supervised learning (WSL)	
#4: Data type agnostic ML-EDM	Chapter 3
#5: Online predictions to be located in time	Chapter 5
#6: Online accuracy vs. earliness trade-off	Chapter 5
#7: Management of non-stationarity in ML-EDM	
#8: Reactivity vs. stability dilemma for revocable decisions	Chapter 4
#9: Non-myopia to revocation risk	Chapter 4
#10: Scheduling strategy and time-dependent decision costs	

TABLE 1.1: Overview of the challenges identified

#1: We have shown in this thesis that non-myopic approaches have enormous potential in early classification of time series, the next step would be to extend them to the unsupervised setting. This would open the path to more industrial applications since supervised data is always difficult and costly to acquire.

#2 and #3: The idea is to extend existing approaches that consider the cost of delaying a decision for classification towards approaches in new settings, namely regression, time series forecasting, or weakly supervised learning.

#4: We argue that triggering models should be data type agnostic in order to deal with any type of data that is enriched over time.

#5: Develop algorithms that can trigger decisions located in the future or the past. For example, predict the state of the machine as soon as possible during the following week.

#6: The compromise between accuracy and earliness should be optimized in more generic scenarios, and not only in the setting of ECTS.

#7: Streaming environments evolve over time; thus, managing stationarity would be of a great benefit to deploy ML-EDM models into production.

#8: On the one hand, the algorithm needs to be reactive by changing its decision promptly when necessary. On the other hand, it is required to provide stable decisions over time by avoiding excessively frequent and undue changes.

#9: A critical challenge is to estimate the future information gain by considering the risk of revocation. Specifically, a decision that will probably be revoked afterward should be delayed due to this risk. Conversely, a decision which promises to be sustainable should be anticipated.

#10: We might think of a scheduling strategy which depends on the decision made and the decision time. Such a scheduling strategy is helpful in applications where the actions to be performed after a decision can be adapted to a time budget available to perform them.

1.3 List of publications

- Early classification of time series. **Y Achenchabe**, A Bondu, A Cornuéjols, A Dachraoui - **Machine Learning**, 2021. (Chapter 3)
- Early Classification of Time Series: Cost-based multiclass Algorithms. PE Zafar, **Y Achenchabe**, A Bondu, A Cornuéjols, V Lemaire - IEEE 8th International Conference on Data Science and Advanced Analytics **(DSAA)**, 2021. (Chapter 3)
- Early and Revocable Time Series Classification. **Y Achenchabe**, A Bondu, A Cornuéjols, V Lemaire - International Joint Conference on Neural Networks **(IJCNN)**, 2022. (Chapter 4)
- When to Classify Events in Open Time Series. **Y Achenchabe**, A Bondu, A Cornuéjols, V Lemaire - The 14th Asian Conference on Machine Learning **ACML**, 2022. (Chapter 5)
- Open challenges for Machine Learning based Early Decision-Making research. A Bondu, **Y Achenchabe**, A Bifet, F Clérot, A Cornuéjols, J Gama, G Hébrail, V Lemaire, PF Marteau - **SIGKDD** explorations journal, 2022. (Chapter 6)

The rest of the thesis is organized as follows, Chapter 2 presents some background material and an overview of the state of the art of classification and early classification of time series methods. In Chapter 3 the problem of ECTS is studied in-depth, and a novel method is proposed and performs better than the best method of state of the art on a benchmark of 45 datasets. Chapter 4 deals with the irrevocability limitation of ECTS, a novel algorithm that models the risk of revocation is proposed and assessed on a benchmark of datasets. In Chapter 5, two other limitations of ECTS are dealt with. In fact, ECTS deals only with finite length time series, and only one single label is assigned to the entire time series. A novel algorithm which solves the trade-off between earliness and accuracy is proposed under this new setting. In Chapter 6, a more generic problem is formalized and called Machine Learning based Early decision-making (ML-EDM). It extends the ECTS problem to more challenging decision-making problems. A list of 10 challenges is suggested to the scientific community for further research. Chapter 7 summarizes the work done in this thesis and gives research directions for future work.

Chapter 2

Background and State of the art

Abstract

This chapter defines the necessary background to understand the rest of the thesis. Moreover, an overview of time series classification methods is given. In addition a detailed review of state of the art approaches of the problem of early classification of time series (ECTS) is presented.

Some parts of the state of the art have been published in a journal paper:

- Early classification of time series. Y Achenchabe, A Bondu, A Cornuéjols, A Dachraoui - **Machine Learning**, 2021

Introduction

This chapter briefly presents Machine learning which is the core learning framework used in this thesis. Then, it is applied on time series, in the form of a supervised learning problem. Some basic notions about time series are defined for the reader and can be skipped for experts in this research area. Afterwards, an overview of time series classification methods is presented, and finally a detailed review of the state of the art of early classification of time series approaches is discussed.

Machine learning framework

Machine learning, as defined in (Barber, 2012) is the study of data-driven methods capable of mimicking, understanding, and aiding human and biological information processing tasks. In this pursuit, many related issues arise, such as how to compress, interpret and process data. Often these methods are not necessarily directed to mimicking human processing directly but rather to enhance it, such as predicting the stock market or retrieving information rapidly. Probability theory is essential since our limited data and understanding of the problem force us to address uncertainty. In the broadest sense, Machine Learning and related fields aim to learn valuable information about the environment within which the agent operates. Machine Learning is also closely allied with Artificial Intelligence, with Machine Learning emphasizing using data to drive and adapt the model.

Most problems can be formalized as following: i) learning a predictive model; ii) we observe an input x; iii) we make a prediction $\hat{y}$; iv) we evaluate the prediction with respect to the true outcome.

Then we have two spaces: $\mathcal{X}$ the input space, and $\mathcal{Y}$ the outcome space.

More formally, let us start with some definitions:

Definition 2.0.1. ***Hypothesis function***
A hypothesis or prediction function gets input $x \in \mathcal{X}$ and produces a label $\hat{y} \in \mathcal{Y}$.

$$h\colon \begin{cases} \mathcal{X} \to \mathcal{Y} \\ x \mapsto h(x) = \hat{y} \end{cases}$$

Definition 2.0.2. ***Loss function***
A loss function evaluates a prediction with respect to the true label.

$$l\colon \begin{cases} \mathcal{Y} \times \mathcal{Y} \to \mathbb{R} \\ (\hat{y}, y) \mapsto l(\hat{y}, y) \end{cases}$$

We assume that there is a data generating distribution $P_{\mathcal{X}\times\mathcal{Y}}$, from which samples are drawn independently from the same distribution (*i.i.d*). The *hypothesis function* is meant to do well on average, which means that the loss function associated with it is small. It is called the *risk*.

Definition 2.0.3. ***Risk***
The risk of a hypothesis function h is the expected loss of h on a new sample (x,y) drawn from $P_{\mathcal{X}\times\mathcal{Y}}$.

$$l\colon \begin{cases} \mathcal{H} \to \mathbb{R} \\ h \mapsto R(h) = \mathbb{E}[l(h(x), y)] \end{cases}$$

Remark. *This expectation can not be computed because $P_{\mathcal{X}\times\mathcal{Y}}$ is unknown.*

Definition 2.0.4. ***Bayes hypothesis function***
A Bayes hypothesis function $h^* : \mathcal{X} \to \mathcal{Y}$ is the function that achieves the minimal risk among all possible functions $h^* = Argmin_{h\in\mathcal{H}} R(h)$.

Remark. *This function is the target function since it is the best hypothesis function we can produce.*

However, we can not compute the risk, since we do not know $P_{\mathcal{X}\times\mathcal{Y}}$. But we can estimate it if we have access to a *training dataset* $\{(x^1, y^1), \ldots, (x^m, y^m)\}$ of samples that are drawn i.i.d from $P_{\mathcal{X}\times\mathcal{Y}}$.

Definition 2.0.5. ***Empirical risk***
The empirical risk of h with respect to $\{(x^1, y^1), \ldots, (x^m, y^m)\}$ of samples that are drawn i.i.d from $P_{\mathcal{X}\times\mathcal{Y}}$ is:

$$\hat{R}_m(h) = \frac{1}{m}\sum_{i=1}^{m} l(h(x^i), y^i)$$

and by the strong law of large numbers $\lim_{m\to\infty} \hat{R}_m(h) = R(h)$ almost surely.

Definition 2.0.6. ***Empirical risk minimizer***
A hypothesis $\hat{h}$ is an empirical risk minimizer if:

$$\hat{h} = Argmin_{h\in\mathcal{H}} \hat{R}_m(h)$$

Then, the goal of machine learning approaches is to find the empirical risk minimizer. They only differ in how they define the hypothesis space $\mathcal{H}$.

Time series classification (TSC)

Time series data are ubiquitous (Eamonn, 2006). In domains as diverse as finance, entertainment, transportation, and health care, we observe a fundamental shift from parsimonious, infrequent measurement to nearly continuous monitoring and recording. Rapid advances in diverse sensing technologies, ranging from remote sensors to wearables and social sensing, are generating rapid growth in the size and complexity of time series archives, one of the most famous being the UCR archive (Chen et al., 2015).

Definition 2.0.7. ***Time series***
A time series $\mathbf{x}_t = \langle x_1, \ldots, x_t \rangle$ is a collection of t observations made sequentially over time, where $x_{i(1\leq i\leq t)}$ belong to some input domain.

Dealing with time series data is challenging for many reasons. First, databases are extensive. Consequently, the need for efficient algorithms that deal with large databases is essential. Second, the similarity is subjective, which means that its definition depends on the user and the application domain. Third, dealing with real-time series data involves handling different data formats, sampling rates, and missing values.

All the essential tasks that the time series mining community is dealing with require similarity matching, namely clustering (Liao, 2005), classification (Abanda,

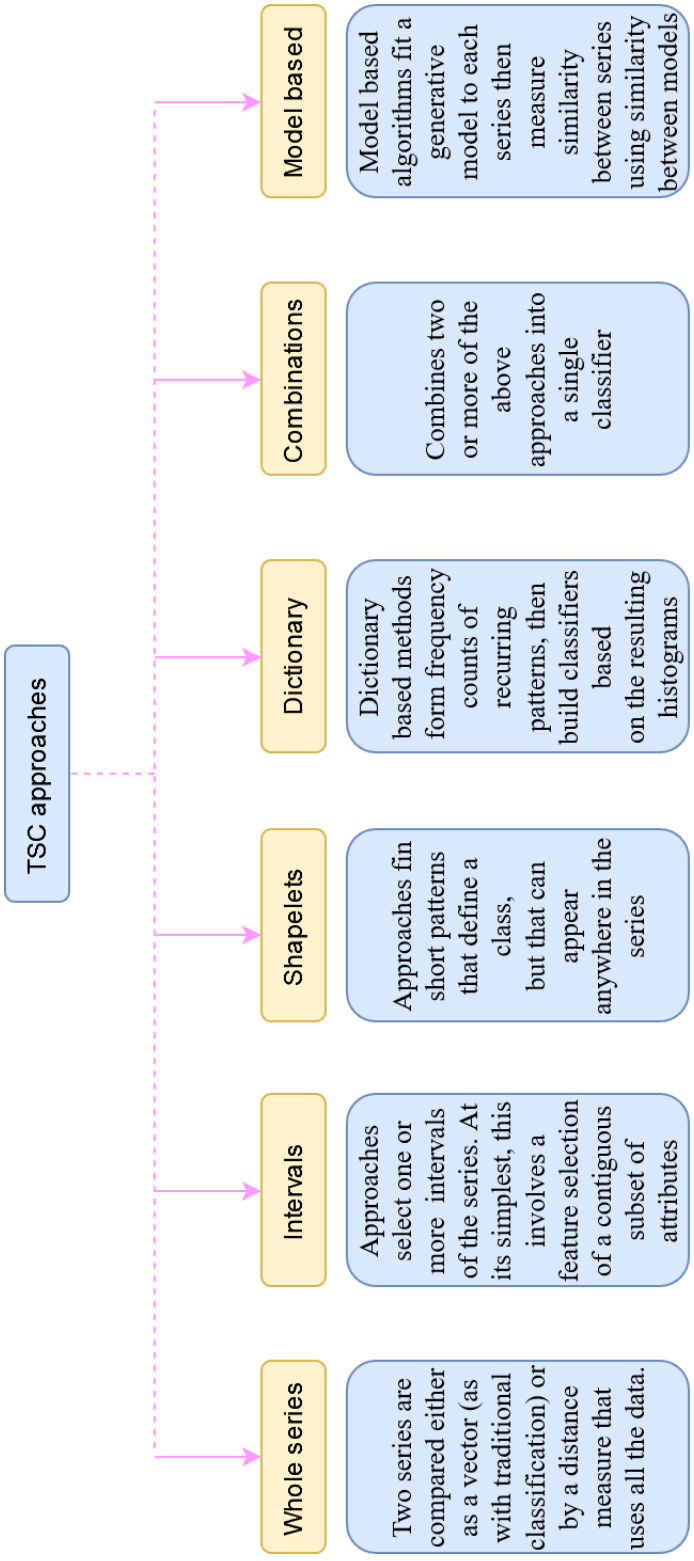


FIGURE 2.1: Grouping of time series classification approaches by discriminatory features

Mori, and Lozano, 2019), motif discovery (Torkamani and Lohweg, 2017) and rule discovery (Das et al., 1998).

In a very influential paper (Anthony Bagnall et al., 2017), the authors proposed a nice way to group time series classification approaches. It is done by the type of discriminatory features the algorithm is attempting to find, as shown in Figure 2.1. First, approaches that compare time series as vectors or using a distance measure, namely euclidean distance or DTW already explained above, or weighted dynamic time warping proposed in (Y.-S. Jeong, M. K. Jeong, and Omitaomu, 2011) or time warp edit (Marteau, 2008) (TWE), move-split-merge (MSM) (Stefan, Athitsos, and Das, 2012), complexity invariant distance (CID) (Batista et al., 2014).

Then, other approaches use time series intervals instead of the whole time series, for example, in a problem that involves classifying an electric device as either a kettle, microwave, or toaster based on a daily usage profile. Toasters are used more frequently in the morning and microwaves in the evening. Consequently, there are large areas of redundant information in each instance, as shown in Figure 2.2. Some interval approaches include *time series forest* (TSF) (Deng et al., 2013), *time series bag of features* (TSBF) (Baydogan, Runger, and Tuv, 2013), *learned pattern similarity* (LPS) (Baydogan and Runger, 2016).

Shapelet based approaches consider that one or more patterns within a series define a class, but their location is irrelevant. Shapelets are subsequences that are highly discriminative for a certain class. This idea has been presented in (Ye and Eamonn Keogh, 2011) and finds shapelets through enumerating all possible candidates, then uses the best shapelet as the splitting criterion at each node of a decision tree. This work has been improved later in (Rakthanmanon and Keogh, 2011), (Hills et al., 2014) and (Grabocka et al., 2014).

While *shapelet* based approaches find a single phase independent pattern that differentiates classes, *Dictionary* based approaches, however, look for the relative frequency of patterns that distinguishes the classes. In this type of problems, the shapelet approach will fail because it looks for the closest single match in each series, not the number of repetitions. This idea has been developed in (J. Lin, Khade, and Y. Li, 2012), (Senin and Malinchik, 2013) and (Schäfer, 2015).

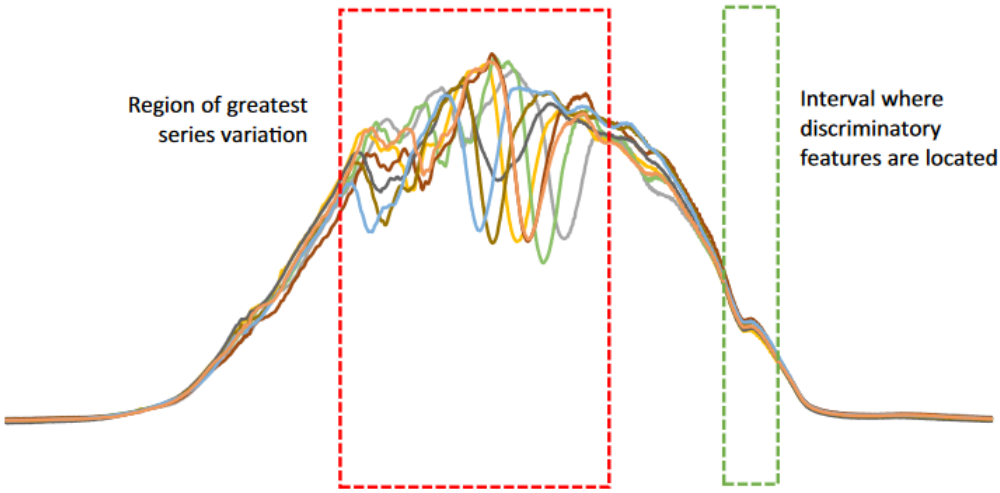


FIGURE 2.2: Time series where interval methods should do better than whole series methods (Anthony Bagnall et al., 2017)

Early classification of time series (ECTS)

Let us start with a classical business problem, *customer churn*, which is the rate at which customers stop doing business with an entity and is expressed as the percentage of service subscribers who discontinue their subscriptions within a given time period. Classically, this problem is solved without taking into account the temporal aspect of the data. In other words, the customers are ranked according to the probability that they are going to churn, and people at the top of the list are contacted during a campaign retention. However, the earlier the actions, the more efficient they are likely to be, detecting churners early enough would be beneficial in order to convince them to stay before they take their final decision. Thus, churn detection can be approached as a timely decision making problem, where customer data is acquired over time, and a decision to contact the customer can be taken at any time according to his behavior. For instance, (Óskarsdóttir et al., 2018) exploit customer's social circle to early detect potential churners using telecommunication data.
This problem is not restricted to numerical continuous time series but also covers image and text data-based problems, (Mocaër, Anquetil, and Kulpa, 2021) worked on early recognition of handwritten gestures since it is of great importance for the coexistence of direct manipulation (zooming, scrolling) and real gestures like symbols in interactive context. (Ebihara et al., 2020) proposed NMNIST dataset which contains videos with 20 frames of MNIST (LeCun et al., 1998) handwritten digits buried with noise at the first frame, gradually denoised towards the last frame, and the main goal is to classify the digit number as soon as possible. Timely decisions are also crucial in the agricultural domain; it plays a significant role in providing food security, preventing famine, and determining policies for sustainable agriculture. Namely, (Rußwurm, Tavenard, et al., 2019) used satellite images for early detection of crop types.

In natural language processing, the problem of early rumor detection (K. Zhou et al., 2019), (Yahui Liu, Jin, and Shen, 2019) has attracted attention because a successfully-detected malicious rumor can still cause significant damage if it is not detected in a timely manner. (R. Singh et al., 2019) enable online platforms to actively look for signs of antisocial behavior and intervene before it gets out of control, namely, preventing situations that can lead to someone committing suicide. Applications were also imagined in Astronomy, where (Muthukrishna et al., 2019) uses early classification on astronomical time series in order to identify transients from within a day of the initial alert, to the whole lifetime of a light curve. In Engineering (Nath et al., 2020) claim that structural rotor fault is the root cause of most rotating machinery issues and still the least addressed faults in rotor faults diagnosis. They propose an early classification method to detect this type of fault as early as possible and improve diagnosis. In the medical domain, (Zhao et al., 2019) proposed a method that predicts whether a patient should be transferred into intensive care units. The timing of this prediction is critical since it is a matter of life and death since the survival rate for patients is improved if they get treated carefully and adequately in time.

Formally, in the problem of *early classification of time series*, we suppose that measurements are made available over time in a time series which, at time t, is $\mathbf{x}_t = \langle x_1, \ldots, x_t \rangle$ where x_t is the current measurement and the $x_{i(1 \leq i \leq t)}$ belong to some input domain (e.g. temperature and blood pressure of a patient). We suppose furthermore that each time series can be ascribed to some class $y \in \mathcal{Y}$ (e.g., patient who needs a surgical operation or not). The task is to predict the class of an incoming time

series as early as possible because a cost is incurred at the time of the decision, where the cost function increases with time.

If the measurements in a time series are supposed independently and identically distributed (i.i.d.) according to a distribution of unknown "parameter" θ, then the relevant framework is the one of *sequential decision making* and *optimal statistical decisions* (Berger, 1985; DeGroot, 2005). In this setting, the problem is to determine as soon as possible whether the measurements have been generated by a distribution of parameter θ_0 (hypothesis H_0) or of parameter θ_1 (hypothesis H_1) with $\theta_0 \neq \theta_1$. One technique especially has gained a wide exposition: Sequential Probability Ratio Test *(SPRT)* invented by (Wald and Wolfowitz, 1948). The log-likelihood ratio $R_t = \log \frac{P(\langle x_1^i,\ldots,x_t^i\rangle \mid y=1)}{P(\langle x_1^i,\ldots,x_t^i\rangle \mid y=0)}$ is computed and compared with two thresholds that are set according to the required error of the first kind (*false positive error*) and error of the second kind (*false negative error*) as shown in Figure 2.3. It has been proved that when measurements are acquired from independently and identically distributed data, SPRT minimizes the required measurements to achieve the required false positive rate and false negative rate. Extensions to non-stationary distributions have been put forward in (Yu Liu and X. R. Li, 2013; Novikov, 2008).

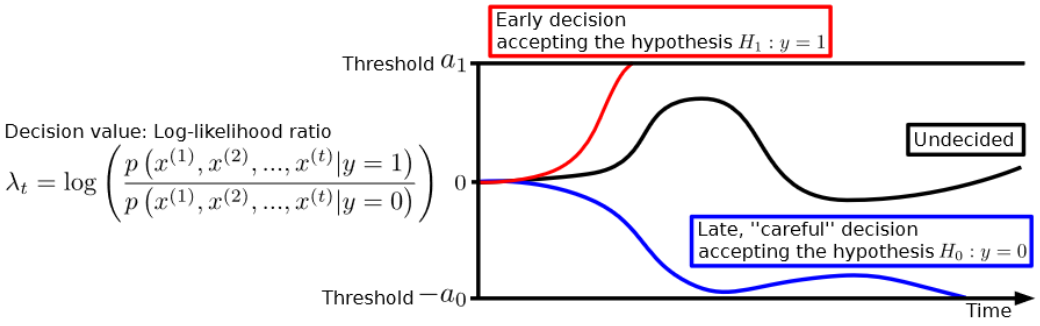


FIGURE 2.3: SPRT algorithm: the log-likelihood ratio is computed and compared to two predefined thresholds (Ebihara et al., 2020).

However, in the early classification of time series problem, the i.i.d assumption does not hold because sequential measurements are usually highly correlated. To compensate for this weaker assumption, it is assumed that a labeled training set exists made of time series of finite length T: $\mathbf{x}_T^i = \langle {x_1}^i, \ldots, {x_T}^i \rangle$ together with their corresponding labels, $\mathcal{D}_{train} = \{(\mathbf{x}_T^i, y_i)\}_{1 \leq i \leq m}$. Each measurement x_j^i can be multivariate.

In the test phase, the scenario goes as follows. At each time step $t < T$, a new measurement x_t is collected, and a decision has to be made as whether to make a prediction now or to defer the decision to some future time step. When $t = T$, a decision is forced.

The problem of deciding online whether a prediction and the attendant actions, should be made or if it should be delayed, can be cast in the LUPI framework (Vapnik and Vashist, 2009). In the following, we examine previous works on the early classification problem in this light.

To the best of our knowledge, (Alonso González and Diez, 2004) is the earliest paper explicitly mentioning "classification when only part of the series are presented to the classifier", and the main thrust of it is to show how the boosting method can be employed to the classification of incomplete time series.

For many researchers, the question to solve is *can we classify an incomplete times series while ensuring some minimum probability threshold that the same decision would be made on the complete input?* To answer this question, several approaches have been put forward.

One is to assume that the time series are generated i.i.d. according to some probability distribution, and to estimate the parameters of the class distributions from the training set. Once $p(\mathbf{x}_T|\mathbf{x}_t)$ the conditional probability of the complete time series $\mathbf{x}_T$ given an incomplete realization $\mathbf{x}_t$ is estimated, it becomes possible to derive guarantees of the form:

$$p\big(h_T(\mathbf{x}_T) = y|\mathbf{x}_t\big) \;=\; \int_{\mathbf{x}_T \text{ s.t. } h_T(\mathbf{x}_T)=y} p(\mathbf{x}_T|\mathbf{x}_t)\, d\mathbf{x}_T \;\geq\; \epsilon$$

where $\mathbf{x}_T$ is a random variable associated with the complete times series, ϵ is a confidence threshold, and $h_T(\cdot)$ is a classifier learnt over the training set $\mathcal{D}_{train}$ of complete times series. At each time step t, $p(h_T(\mathbf{x}_T) = y|\mathbf{x}_t)$ is evaluated and the prediction is triggered if this term becomes greater than some predefined threshold. (Anderson et al., 2012; Parrish et al., 2013) present this method and propose ways to make the required estimations, in particular the mean and the covariance of the complete training data, when the time series are generated by Gaussian processes. It so far applies only with linear and quadratic classifiers.

A set of approaches do not make assumptions about the form of the underlying distributions on the time series. For instance, (Ebihara et al., 2020) proposed SPRT-TANDEM inspired from the famous Wald's SPRT algorithm discussed at the beginning of this section. The main goal is to overcome the i.i.d assumption of the SPRT algorithm that prevents its application to the early classification of time series. The authors propose to approximate the log-likelihood ratio (LLR) as an N-th order Markov process, the log-likelihood ration is expressed as follows using the Bayes theorem:

$$\begin{aligned} log\frac{P(\langle x_1^i,\ldots,x_t^i\rangle \mid y=-1)}{P(\langle x_1^i,\ldots,x_t^i\rangle \mid y=+1)} = &\sum_{s=N+1}^{t} log\frac{P(\langle y=-1 \mid x_{s-N}^i,\ldots,x_s^i\rangle)}{P(\langle y=+1 \mid x_{s-N}^i,\ldots,x_s^i\rangle)} \\ &- \sum_{s=N+2}^{t} log\frac{P(\langle y=-1 \mid x_{s-N}^i,\ldots,x_{s-1}^i\rangle)}{P(\langle y=+1 \mid x_{s-N}^i,\ldots,x_{s-1}^i\rangle)} \\ &+ log\frac{p(y=-1)}{p(y=+1)} \end{aligned}$$

A novel deep neural network architecture was developed in order to compute the first two terms of the equation above, while the last term can be seen as a bias term, which is chosen by the user according to the distribution of the dataset at hand, or it could be left as a flat prior. The parameter N can be optimized using hyper-parameter tuning algorithms. During the testing phase, this algorithm works exactly like the classical SPRT algorithm. Once the approximated LLR is computed, it is compared to user predefined thresholds in order to trigger a decision to classify the time series as $y = +1$ or $y = -1$. Extension to multi-class problems has been presented in (Miyagawa and Ebihara, 2021).

A set of approaches are based on the concept of *minimum prediction length* (MPL) presented for the first time in (Xing, Pei, and Philip, 2009). The key idea is to learn a timestamp from which the prediction about the class of the incoming time series

becomes stable, which corresponds to a prefix of time series, this type of approaches is also called *prefix based approaches* in the literature.
First, let us introduce some notations. For a time series $\mathbf{x}$ and a training set $\mathcal{D}_{train}$, let us consider $NN^l(\mathbf{x}) = ArgMin_{\mathbf{x}' \in \mathcal{D}_{train}}\{dist(\mathbf{x}_l, \mathbf{x}'_l))\}$ to be the set of the nearest neighbors of $\mathbf{x}$ in $\mathcal{D}_{train}$, and $RNN^l(\mathbf{x}) = \{\mathbf{x}' \in \mathcal{D}_{train} | \mathbf{x} \in NN^l(\mathbf{x}')\}$ is the set of reverse nearest neighbors, consequently if there exists a timestamp $l_0 \leq T$ such that $RNN^{l_0}(\mathbf{x}) = RNN^{l_0+1}(\mathbf{x}) = \cdots = RNN^T(\mathbf{x})$ then l_0 could be used to make prediction early at l_0 without loss in prediction accuracy because any time series $\mathbf{x}' \in \mathcal{D}$ which means that a time series that have $\mathbf{x}$ as nearest neighbor at T has also $\mathbf{x}$ as nearest neighbor at l_0, thus early prediction can be made at l_0 without loss of expected accuracy. The formal definition given in the original paper (Xing, Pei, and Philip, 2009) is the following:

Definition 2.0.8. ***Minimum prediction length (MPL) of a time series***
For a time series $\mathbf{x}$*, and a training dataset* $\mathcal{D}_{train}$ *the minimum prediction length (MPL) of* $\mathbf{x}$ *MPL(*$\mathbf{x}$*) = k if*

1. $\forall l$ *such that* $k \leq l \leq T$, $RNN^l(\mathbf{x}) = RNN^T(\mathbf{x}) \neq \emptyset$.
2. $RNN^{k-1}(\mathbf{x}) \neq RNN^T(\mathbf{x})$.

Specifically if $RNN^T(\mathbf{x}) = \emptyset$ *then MPL(*$\mathbf{x}$*) = T.*

The first simple method to perform early classification of time series starts during the training phase by computing the MPL for each time series in the training dataset. In the testing phase, a decision to classify the incoming time series is triggered at timestamp t if a time series in $NN^t(\mathbf{x})$ has an MPL at most t, and the dominating class in this set is returned. Otherwise, the decision is delayed. The main drawback of this 1NN early classification method is that it may overfit the training dataset because the MPL is computed using the RNN, which may consider a few time series. The method called *ECTS* presented in (Xing, Pei, and Philip, 2009) addresses this issue by finding the MPL for a cluster of similar time series instead of a single time series. Before giving the formal definition of the MPL of a cluster, the authors introduce a couple of notions. A *discriminative cluster* is a cluster where time series carry the same label, and it could be used for classification. The reverse nearest neighbor of a cluster S is the union of all clusters except S. A cluster S is called *1NN consistent* if for each time series $\mathbf{x}$ in S, the 1NN of $\mathbf{x}$ also belongs to S. The formal definition as proposed by the authors is the following:

Definition 2.0.9. ***Minimum prediction length (MPL) of a cluster***
For a discriminative cluster S*, and a training dataset* $\mathcal{D}_{train}$ *the minimum prediction length (MPL) of S MPL(S) = k if*

1. $\forall l$ *such that* $k \leq l \leq T$, $RNN^l(S) = RNN^T(S)$.
2. S *is 1NN consistent in space* $\mathbf{R}^l$.
3. *for l = k-1 (1) and (2) are not satisfied.*

During training the nearest neighbors for each time series in T in all prefix spaces are computed, they are used later while performing agglomerative hierarchical clustering method to cluster the training data set in full length space. For each discriminative subcluster S, using the pre-computed 1NN information in the prefix spaces, the 1NN consistency of S is checked and the stability of $RNN^l(S)$ for $l < T$ in the value l descending order until MPL(S) S is determined.

The *ECDIRE* method has been proposed later in (Mori, Alexander Mendiburu, Eamonn Keogh, et al., 2017) in order to learn the MPL using the classifier's posterior probabilities. During the training phase, a collection of classifiers is learned for each time stamp by truncating the time series, in order to assign a class label to the complete but still unknown time series while having an incomplete time series as input. Then for each class label c, a timestamp from which it is safe to predict c is learned as shown in Figure 2.4. A prediction is considered safe when it reaches the percentage of accuracy of the classifiers that has the complete knowledge of the time series, and this ratio is given by the user. The second step consists of learning thresholds for each class label and time stamp. The others proposed to take the minimum of the distance between the first and second largest probabilities evaluated only on correctly classified time series. During the prediction phase, only safe class labels will be considered. In other words, if the classifier assigns a class label before the time stamp from which it is safe to consider this class label, it is not taken into account, and the decision would be to gather more data. Otherwise, if the predicted class is safe, then the second step is to check if it is reliable enough, which means that the difference between the first and the second largest probabilities is higher than the learned threshold.

The MPL-based approaches presented above make an implicit assumption about the cost of delay. In fact, the cost of delay is implicitly considered low, as in (Mori, Alexander Mendiburu, Eamonn Keogh, et al., 2017) the algorithm waits until a safe time stamp is reached to predict a specific class. Moreover, we argue that the eariness-accuracy tradeoff is a key part of decision-making, and it is not modeled explicitly in these approaches. In addition, they are *myopic*. In other words, they only decide if the current timestamp is the right moment to trigger a decision to classify the input time series. Myopic algorithms give no information about when the alarm will be triggered in the future.

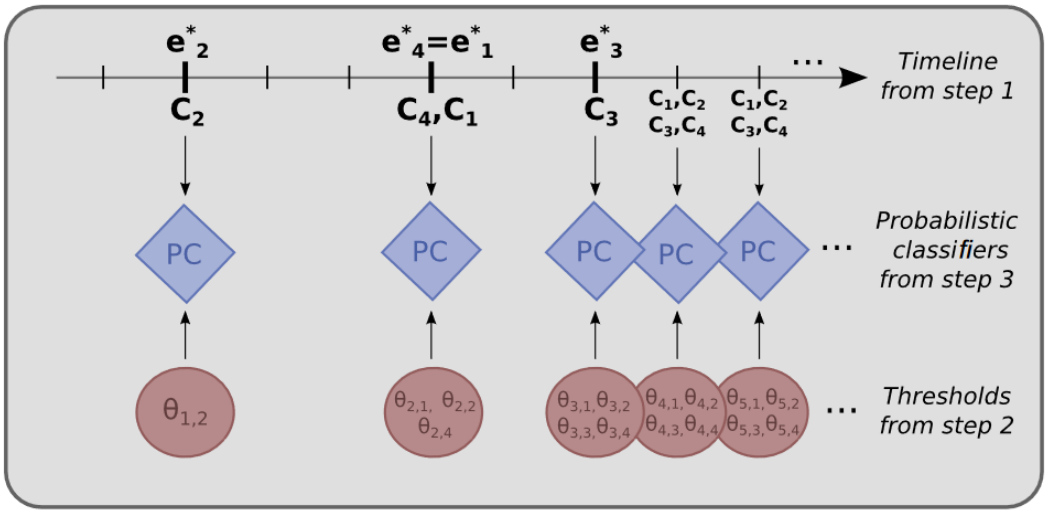


FIGURE 2.4: MPL for each class and reliability thresholds (Mori, Alexander Mendiburu, Eamonn Keogh, et al., 2017).

Another line of research is concerned with finding good descriptors of the time series, especially on their starting subsequences, so that early predictions can be reliable because they would be based on relevant similarities on the time series.

We start by giving an overview of the concept of *Shapelets* introduced in (Ye and Eamonn Keogh, 2009, 2011).

Definition 2.0.10. ***Shapelet***
Shapelets are time series subsequences which are in some sense maximally representative of a class. A shapelet is denoted as $f = (s, \delta, c)$, where s is the time series subsequence, δ is the threshold distance that should be exceeded to predict class label c.

In other words, it is an extracted contiguous part of a time series. This part is highly discriminative regarding a class. This concept has many advantages. First, it improves drastically the space and time complexity of the k-nearest neighbors algorithm, because instead of having to store the entire time series dataset in order to look for the nearest neighbor at the test phase, only shapelets will be stored. Second, shapelets are interpretable, which enriches understanding of the data to be analyzed. Third, shapelets are local features, which may be more efficient than global features methods in some applications. The question now is how to choose the shapelet among the candidates? The answer was given in (Ye and Eamonn Keogh, 2009), shortly, the shapelet is the subsequence that maximizes the *information gain*.

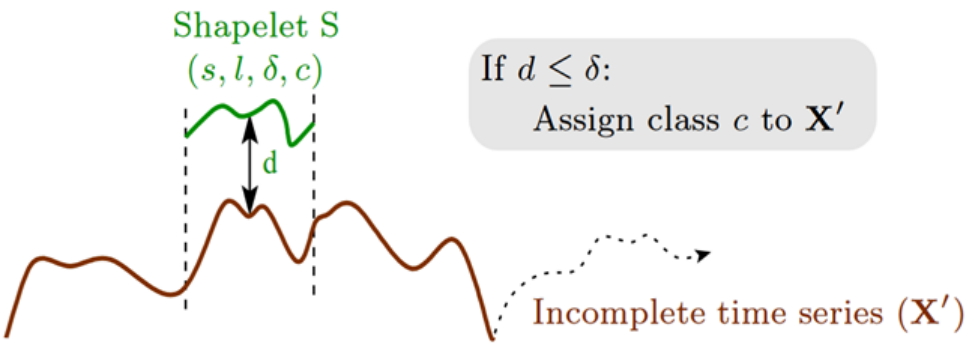


FIGURE 2.5: Shapelet based methods .

More formally, let us present some definitions that would be useful for the rest of the chapter.

Definition 2.0.11. ***Entropy***
let us consider $\mathcal{D}_{train}$ as a time series dataset, which contains two classes c_1 and c_2 with proportions $p(c_1)$ and $p(c_2)$ respectively, then the entropy of the dataset $\mathcal{D}$ is defined as:

$$E(\mathcal{D}) = -p(c_1)log(c_1) - p(c_2)log(c_2)$$

Definition 2.0.12. ***Information gain***
let us consider $\mathcal{D}$ as a time series dataset. After splitting $\mathcal{D}$ into $\mathcal{D}_1$ and $\mathcal{D}_2$ such that $\mathcal{D} = \mathcal{D}_1 \cup \mathcal{D}_2$, the information gain is defined as

$$IG(\mathcal{D}_1, \mathcal{D}_2) = E(\mathcal{D}) - r(\mathcal{D}_1)E(\mathcal{D}_1) - r(\mathcal{D}_2)E(\mathcal{D}_2)$$

where $r(\mathcal{D}_1)$ is the ratio of the time series of $\mathcal{D}_1$ in $\mathcal{D}$.

Definition 2.0.13. ***Best matching distance (BMD) (Xing, Pei, P. S. Yu, et al., 2011)***
For a local shapelet f = (s,?,c) and a time series $\mathbf{x}$ such that the length of s is less than the length of t, the BMD between f and $\mathbf{x}$ is:

$$BMD(f, \mathbf{x}) = min_{s' \subset \mathbf{x}, len(s') = len(s)}\, dist(s, s')$$

The distance between the candidate subsequence and every time series in $\mathcal{D}$ will be computed. It corresponds to the best matching location in the time series for the candidate subsequence then these distances are sorted, and a distance threshold is selected in order to maximize information gain. Finally, the candidate subsequence with the maximum gain is chosen as the *shapelet* for $\mathcal{D}$.

Various methods in the ECTS literature used *shapelets* in order to solve the problem of early classification of time series. The key idea is to extract shapelets during the training phase as done in time series classification. Then, during the testing phase, measurements are gathered sequentially. The objective would be to detect shapelets as soon as possible in the incoming incomplete time series.

To the best of our knowledge, (Xing, Pei, P. S. Yu, et al., 2011) is the first to deal with extracting interpretable shapelets for the early classification on time series. The motivation behind this paper is to provide shapelets as interpretable features as they are essential for critical applications, namely in the medical domain. The authors propose a method called *EDSC* that consists of two main steps, feature extraction and feature selection.

For the feature extraction part, the algorithm extracts all subsequences of a given minimum and maximum lengths, and for each subsequence, a robust distance threshold is learned. The authors proposed multiple methods to learn it, *first*, the most simple one is to set a precision threshold, for example, 95% precision, and choose a distance threshold that can achieve a precision above the precision threshold. If multiple distance thresholds meet the requirement, the largest one can be picked since this enables the local shapelet to cover more training data. The *second* way of learning the distance threshold is based on kernel density estimation (KDE) (Marzio and Taylor, 2005). It is used on the list of the computed BMDs to estimate the probability density functions of the target class and the non-target classes, and then set the distance threshold so that for all time series that fit in this distance range, at every the probability density of belonging to the target class passes a probability threshold.

For the *feature selection* phase, the authors proposed to rank candidate shapelets according to a utility function defined in their paper as following:

Definition 2.0.14. ***Utility***

let f = (s,δ,c) a shapelet, the utility of f is defined as:

$$Utility(f) = \frac{2 \times Precision(f) \times WRecall(f)}{Precision(f) + WRecall(f)}$$

where the WRecall is based on earliest match length that takes into account earliness

Top shapelets that cover the entire datasets are selected, finally time series are early classified based on the closest matching shapelet.

In (G. He, Duan, G. Zhou, et al., 2014), authors propose a similar method called *MCFEC*, the novelty consists of proposing a generalized extended F-measure (GEFM for short) to evaluate the quality of shapelets in terms of the recall, precision and earliness as following:

Definition 2.0.15. ***GEFM***

let f = (s,δ,c) a shapelet, the GEFM of f is defined as:

$$GEFM(f) = \frac{1}{\frac{w_0}{Earliness(f)} + \frac{w_1}{Precision(f)} + \frac{w_2}{Recall(f)}}$$

(Ghalwash and Obradovic, 2012) extends the concept of shapelets to the multivariate case, and a vector of thresholds for each dimension is learned. (Y.-F. Lin et al., 2015) takes this work a step further by proposing a framework called *REACT* that incorporates categorical attributes in multivariate time series. Unlike previously described approaches, (Wang et al., 2016) presents an approach called *EA-ConvNets* that uses a convolutional neural network to learn shapelets from a training data set instead of extracting them using hand-crafted rules. It takes as input an incomplete time series and outputs the class label of the time series. This framework consists of 3 main stages: First, early awareness, which consists of stochastically choosing a truncation point for each time series in the training set. Technically it is done using a geometric distribution because of its exponential decay property. Second, a deep convolutional feature extractor is used to identify shapelet-based features using 1D convolution operation, in addition to average and max pooling operations. Third, a final classifier followed by a softmax operation which minimizes the cross-entropy loss. The whole architecture is shown in Figure 2.6. However, this framework does not solve the problem of early classification completely. In fact, it does not determine the time stamp at which it is possible to classify an incoming time series.

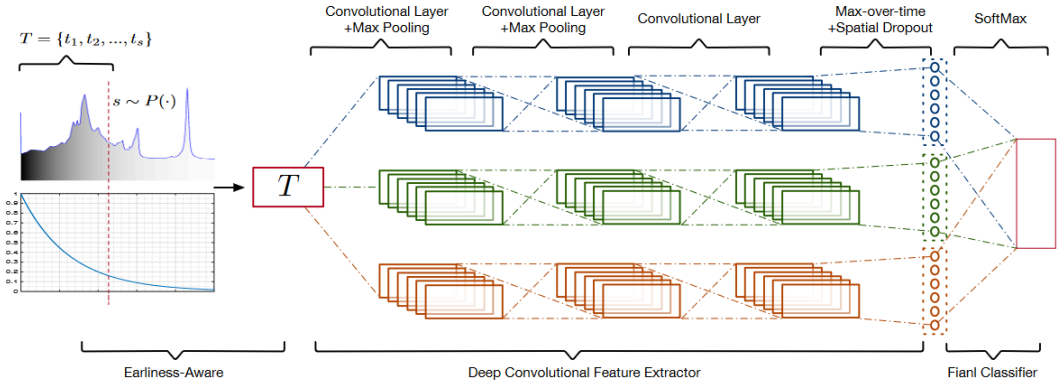


FIGURE 2.6: EA-ConvNets architecture (Wang et al., 2016)

(Ghalwash, Radosavljevic, and Obradovic, 2014) introduce *MEDSC-U* which is an improved version of *EDSC* method by providing uncertainty quantification. The idea is to modify the selection and classification steps in *EDSC* in order to obtain more discriminative shapelets and capture a robust uncertainty estimate. For the pruning phase, a list of equal-performance shapelets is extracted, and it is longer than the one extracted in the classical method because, with a longer list, the uncertainty would be better estimated. For the classification phase, at each time step, uncertainty is computed for each class based on the matched shapelets with the incomplete time series at hand. The decision to classify the time series is triggered when a predefined level of uncertainty is reached, and the class that has minimum uncertainty is chosen.

The Shapelet-based approaches presented above have the same limitations as MPL-based ones. Algorithms are also designed in order to obtain a minimal accuracy

defined by the user. Some metrics (e.g. GEFM) that incorporate earliness and accuracy at the same time are proposed, and optimized during training, which is a way to deal with the ealiness-accuracy tradeoff, however these approaches do not model this compromise explicitly in the decision criterion. Moreover, all these approaches are myopic.

Another set of approaches considered the trade-off between earliness and accuracy. (Dachraoui, Bondu, and Cornuéjols, 2015), for the first time, the problem of early classification of time series is cast as the optimization of a loss function which combines the expected cost of misclassification at the time of decision plus the cost of having delayed the decision thus far. Besides the fact that this optimization criterion is well-founded, it also permits applying the LUPI framework because the expected costs for an incoming subsequence $\mathbf{x}_t$ can be estimated for future time steps and thus a non-myopic decision procedure can be used. This approach will be explained in details in the next chapter.
The merit of (Mori, Alexander Mendiburu, Dasgupta, et al., 2017) is to recognize that the earliness vs. accuracy trade-off depends on each domain and dataset and that it must therefore be expressed as a single optimization criterion. The authors expressed this criterion as a cost function that depends on the chosen stopping rule of the following form:

$$Trigger^{\gamma}\left(h_t(\mathbf{x}_t)\right) = \begin{cases} 0 & \text{if } \gamma_1 p_1 + \gamma_2 p_2 + \gamma_3 \frac{t}{T} \leq 0 \\ 1 & \text{otherwise} \end{cases} \tag{2.1}$$

where p_1 is the largest posterior probability estimated by the classifier h_t: $p_1 = \text{ArgMax}_{y \in \mathcal{Y}}(\hat{p}(y|\mathbf{x}_t))$, p_2 is the difference between the two largest posterior probabilities, defined as $|\hat{p}(y=1|\mathbf{x}_t) - \hat{p}(y=0|\mathbf{x}_t)|$ in the case of binary classification problems, and where the last term $\frac{t}{T}$ represents the proportion of the incoming time series that is visible at time t, $\gamma_1, \gamma_2, \gamma_3$ are hyperparameters in $[-1, 1]$ to be optimized according to a cost function of the form:

$$CF(\mathbf{x}, Trigger^{\gamma}) = \sum_{(\mathbf{x}_T, y) \in \mathcal{D}} \left(\alpha\, C_m(Trigger^{\gamma}|y) + (1-\alpha)\, C_d(Trigger^{\gamma})\right) \tag{2.2}$$

With α, the parameter which controls the trade-off between earliness and accuracy, C_m is the cost of misclassification, and C_d is the cost of delay. This contrasts with approaches whereby the decision is made solely on the basis of a given confidence threshold that should be attained. However, the optimization criterion put forward is heuristic, supposes that the cost of delaying a decision is linear in time, and involves a complex setup. Most importantly, again, it is a *myopic* procedure which does not consider the foreseeable future. The same authors formulated the problem in (Mori, Alexander Mendiburu, Miranda, et al., 2019) as a multi-objective optimization problem. The key idea is instead of finding one optimal trigger function $Trigger^{\gamma^*}$, the algorithm will find a set $\{Trigger^{\gamma^1}, Trigger^{\gamma^2}, \ldots, Trigger^{\gamma^n}\}$, which will provide non-dominated results in accuracy and earliness. For this, the cost of misclassification, as well as the cost of delaying the decision, will be considered in the following optimization problem:

$$min_{\gamma}(C_m(Trigger^{\gamma}), C_d(Trigger^{\gamma})) \tag{2.3}$$

This algorithm provides multiple solutions to the user instead of manually choosing the parameter α that controls the trade-off between earliness and accuracy. For all these apparent shortcomings, this method has been found to be quite effective, beating most competing methods in extensive experiments. This is why it is used as a reference method for comparison in this thesis, as is also done in (Rußwurm, Lefevre, et al., 2019) which compares several techniques for early classification of time series.

The trade-off between earliness and accuracy has also been considered in (Shekhar et al., 2021), this approach is called *Benefitter* and the key idea is to consider at each time step benefits instead of costs already considered in approaches described before. This approach considers a target variable called benefit, which is the overall value of outputting a certain label at a certain time t given as:

$$benefit(t) = S(t) - M \tag{2.4}$$

with $S(t) = (L - t) \times$ s is the savings made at time t, M is the misclassification cost, L is the length of the monitoring period, and s is the saving per unit of time. The authors distinguish two problems: Outcome classification and Type classification. On one hand, in outcome classification, labels encode the outcome observed at the end of the monitoring period (e.g., survival or death). In such cases, predicting the normal outcome does not change anything and is considered as a default state. For example, predicting the survival of a monitored patient in a hospital or that a machine is working normally in a factory does not lead to any real savings or costs. On the other hand, in type classification, labels encode the generation process of time series, for example, predicting the type of a bird from audio recordings. In such cases, the prediction of any label at a given time has an associated cost of misclassification and cost of delay (or savings for earliness). In order to solve the classification problem, authors have access to a training dataset $\mathcal{D}_{train} = \{(\mathbf{x}^i, y^i)\}_{i=1...m}$ where $\mathbf{x}$ is a time series with a length L_i and y^i is the corresponding label. The main idea of this

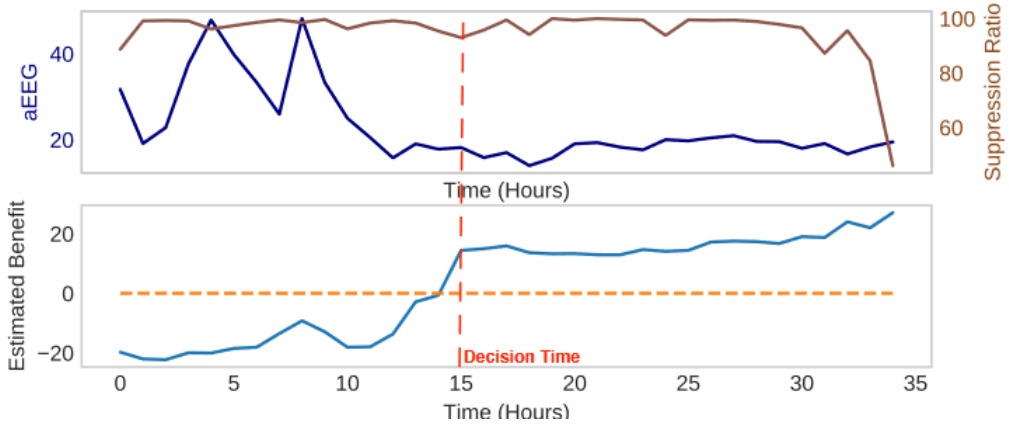


FIGURE 2.7: Benefitter in real-time (Shekhar et al., 2021)

approach is to transform this classification problem into a supervised regression problem, where the target variable *benefit* is predicted at each time step using an LSTM regressor. This model is trained on a new dataset constructed from $\mathcal{D}_{train}$. For each time series $\mathbf{x}^i$ in $\mathcal{D}$, we extract a set of subsequences $\{(\mathbf{x}^i_t, b_{it})\}_{t=1...L_i}$ of time series of increasing length with the corresponding benefit b_{it} computed as described in Equation 2.4. In real-time, as shown in Figure 2.7 measurements of time series become available over time. At each time step, we gather a new measurement, and the regressor predicts the benefit at the current time step using the incomplete time

series at hand, if the benefit is positive then an early decision is taken. Otherwise, the decision is delayed. This approach is considered as a myopic approach, because it only looks at the current time step without considering if the future holds a better benefit.

(Rußwurm, Lefevre, et al., 2019) proposed a trainable deep learning framework for early classification of time series that can be fine-tuned end-to-end using standard gradient back-propagation. First, it learns a fixed length hidden representation h_t using an LSTM or a 1D-convolutional shapelet model. The authors assumed that this hidden representation is likely to encode information about whether to stop at the current prediction or wait for more observations. Second, from this assumption, they proposed to use this information to produce a stopping decision probability δ_t so that at classification time, for each timestamp t, the decision about performing classification or waiting for more measurements is drawn from a Bernoulli distribution of parameter δ_t. In order to optimize the accuracy-earliness trade-off they proposed the following overall cost function :

$$L(x_t, y; \alpha) = \sum_{t=0}^{T} P(t)(\alpha L_c(x_t, y) + (1 - \alpha)L_e(t))$$

which contains L_c the cost of misclassification, L_e the cost of delaying the decision and $P(t)$ the probability of making the decision at timestamp t. This gives larger gradients for time stamps when the decision is more likely to happen. The authors used a two-phase training procedure. They first pre-train the model purely for accuracy with a logistic regression loss and then fine-tune it end-to-end for accuracy and earliness.

The methods described above model the earliness-accuracy tradeoff, in addition, the cost of misclassification as well as the cost of delay are taken into account explicitly, which makes these methods cost sensitive. However, the only method that is non-myopic is the ECONOMY approach.

Another set of approaches focused on applying deep learning and reinforcement learning to the problem of early classification of time series.

Reinforcement learning (RL) aims at learning to solve a decision-making problem through trial and error. A transition occur from state $s_t \in \mathcal{S}$ to state $s_{t+1} \in \mathcal{S}$ under an action $a \in \mathcal{A}$ chosen by a policy $\pi : \mathcal{S} \to \mathcal{A}$, consequently rewards $r_t = r(s_t, a) \in \mathbb{R}$ are associated with those transitions. At each timestamp the goal of the agent is to maximize its long term reward which is $\sum_{k=0}^{\infty} \gamma^k r_{t+k}$ with $\gamma \in [0, 1]$ is a discount factor valuing more immediate rewards. In the RL literature, two functions are defined, the *State value* function, which is the expected reward that the agent can hope to get while starting at state $s \in \mathcal{S}$ and following its policy π: $V_\pi(s_t) = \mathbb{E}_\pi[\sum_{k=0}^{\infty} \gamma^k r_{t+k} | s_t]$ and the *Action value* function which is the expected reward that the agent can hope to get while starting at state $s \in \mathcal{S}$, picking action a_t and then following its policy $Q_\pi(s_t, a_t) = \mathbb{E}_\pi[\sum_{k=0}^{\infty} \gamma^k r_{t+k} | s_t, a_t] = \mathbb{E}_\pi[r_t + \gamma Q_\pi(s_{t+1}, a_{t+1}) | s_t, a_t]$. The optimal policy can be obtained from the action value function, the optimal one is defined as $Q^*_\pi(s, a) = max_\pi Q_\pi(s, a)$ then the optimal policy is $\pi^*(s) = argmax_{a \in \mathcal{A}} Q^*_\pi(s, a)$.

(Martinez, Perrin, et al., 2018) approached the problem of early classification as a sequential decision-making problem in the framework of reinforcement learning. The state s_t at time step t is the visible part of the incomplete time series at hand, the set of actions $\mathcal{A} = \{wait, \cup_{y \in \mathcal{Y}} predict\, y\}$, an episode ends when the partial sequence is fully completed, or when the agent predicts a class label, and finally the rewards

are chosen in order to learn a policy that solves the accuracy-earliness trade-off. The authors proposed a reward of +1 when the agent predicts the right label, -1 if it predicts the wrong label, and an increasing waiting cost if it decides to wait for more measurements. A deep neural network was proposed to estimate the state-value function, which allows selecting the optimal policy, the one that maximizes the state-value function. To take advantage of the recent success in representation learning and reinforcement learning, (Hartvigsen et al., 2019) proposed a method called EARLIEST (Early and Adaptive Recurrent Label ESTimator). It is composed of a deep network consisting of three sub-networks: First, a recurrent neural network generating low dimensional vector representation of the sequence in input, whatever its length is. Second a discriminator network learns from the hidden representation to predict class labels of input time series. Third, a controller network is a reinforcement learning agent that decides whether to predict class label or delay the decision at each time step. This approach was tested on a benchmark of 7 datasets and versus 3 baselines. However, unfortunately, it was not tested against state-of-the-art approaches. The authors extended this framework in (Hartvigsen et al., 2020) to the multi-label early classification problem.

The reinforcement learning framework is very general. It uses immediate and delayed rewards. As shown in this section, there is in principle no obstacle to apply reinforcement learning to the learning of a good triggering strategy. However, if used directly, the generality of RL is paid for by a need for a large number of "experiments". In addition, the state space is continuous in the case of the ECTS problem, thus an interpolating functions must be used in order to represent the values such as $v_\pi(s)$ and this entails the choice of a family of functions and setting their associated parameters.

Another approach, the one favored in the current literature for ECTS (Achenchabe, Bondu, and Cornuéjols, 2021), is to choose functions for representing the expected values of decision times, and thus providing a ground for the triggering strategy. This has the merit of incorporating prior knowledge of the trade-off between earliness and accuracy, at the cost of making modeling choices that may bias the method of estimating the expected future cost.

The respective performances, merits and limits of both approaches should be studied empirically by a comparison of RL based ECTS approaches, such as (Martinez, Ramasso, et al., 2020), with approaches that explicitly exploit the form of the optimization criterion designed for ECTS as in (Achenchabe, Bondu, and Cornuéjols, 2021).

In the rest of this thesis, the ECONOMY approach will be studied in depth for its interesting properties discussed while comparing the state of the art methods. It will be used as a core component for the proposed algorithms and adapted for new online scenarios.

Chapter 3

Early classification of time series

Abstract

An increasing number of applications require to recognize the class of an incoming time series as quickly as possible without unduly compromising the accuracy of the prediction. In this chapter, we put forward a new optimization criterion which takes into account both the cost of misclassification and the cost of delaying the decision. Based on this optimization criterion, we derived a family of non-myopic algorithms which try to anticipate the expected future gain in information in balance with the cost of waiting. In one class of algorithms, unsupervised-based, the expectations use the clustering of time series, while in a second class, supervised-based, time series are grouped according to the confidence level of the classifier used to label them. Extensive experiments carried out on real datasets using a large range of delay cost functions show that the presented algorithms are able to solve the earliness vs. accuracy trade-off, with the supervised partition based approaches faring better than the unsupervised partition based ones. In addition, all these methods perform better in a wide variety of conditions than a state of the art method based on a myopic strategy which is recognized as being very competitive. Furthermore, our experiments show that the non-myopic feature of the proposed approaches explains in large part the obtained performances.

The content of this chapter has been published as a journal paper as well as a conference paper:

- *Early classification of time series*. Y Achenchabe, A Bondu, A Cornuéjols, A Dachraoui - **Machine Learning**, 2021
- *Early Classification of Time Series: Cost-based multiclass Algorithms* PE Zafar, Y Achenchabe, A Bondu, A Cornuéjols, V Lemaire - **IEEE 8th International Conference on Data Science and Advanced Analytics (DSAA)**, 2021

3.1 Introduction

In emergency wards of hospitals (Mathukia et al., 2015), in control rooms of national or international electrical power grids (Dachraoui, Bondu, and Cornuejols, 2013), in government councils assessing emergency situations, in all kinds of contexts, it is essential to make timely decisions in absence of complete knowledge of the true outcome (e.g. should the patient undergo a risky surgical operation?). The issue facing the decision makers is that, usually, the longer the decision is delayed, the clearer is the likely outcome (e.g. the critical or not critical state of the patient) but, also, the higher the cost that will be incurred if only because earlier decisions allow one to be better prepared. How to optimize online the tradeoff between the earliness and the accuracy of the decision is the object of the early classification of time series problem and this is what is addressed in this chapter.

In the previous chapter, we have presented the state-of-the-art of early classification of time series approaches. In this chapter, we will dive deep into the ECONOMY approach for the following reasons:

- It formalizes explicitly the tradeoff between earliness and accuracy.
- It is well founded and takes into account the cost of misclassification and the cost of delay.
- It is the only non-myopic approach in the literature, in other words, the trigger system may even predict what will be the best time in the future to perform the classification of the incoming time series.

Overall, the early classification of time series (ECTS) problem can be seen as involving two components (see Figure 3.1):

- A classifier[1] that takes as an input a given incomplete time series, and produces a probabilistic estimation of associated class as an output.
- A *trigger model* which decides whether to make the prediction at the current time t or to wait for at least one more measurement x_{t+1}.

The example illustrated in Figure 3.1 is the early classification of a potential suspicious account in a given platform. The objective is to trigger as soon as possible a potential suspicious activity associated with an account, this could be for example, a potential hacker, or a fake account in social media platforms.

The work presented in this chapter takes the earliness versus accuracy trade-off at face value in the spirit of (Dachraoui, Bondu, and Cornuéjols, 2015) and formalizes it in a generic way. We extend this previous work as a generic framework for the early classification of time series. Furthermore, this previous work described a method based on the clustering algorithm K-means. However, the use of an unsupervised approach to capture the relevant groups of time series leaves important information aside. This is why, we propose here to resort to supervised techniques in order to get better prediction performance. Interestingly, we claim that the problem of deciding online whether a prediction, and the attendant actions, should be made, or if

[1] Developping novel time series classification algorithms is out of the scope of this thesis

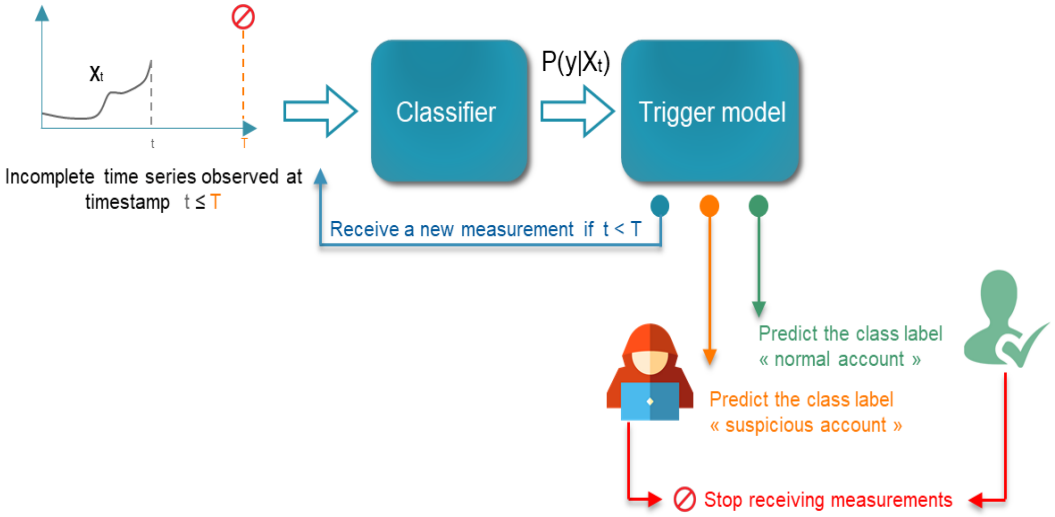


FIGURE 3.1: General schema of ECTS approaches

it should be delayed, can be cast in the LUPI (Learning Under Privileged Information) framework (Vapnik and Vashist, 2009). During the learning phase, the learner has access to the full knowledge about the training time series in addition to their class, while at testing time, only the incoming, and incomplete, time series is known. And decisions as whether to wait for additional measurements or not have to be made from this incomplete knowledge. The resulting optimization criterion can be used in a *non-myopic* procedure where estimates of future costs can be compared to the cost incurred if the decision was made at the current time.

The second objective of this chapter is to design efficient optimization algorithms which implement the presented framework. We define a set of design choices that allow the categorization of possible *non-myopic* approaches (see Table 3.1). This enables us to define three novel algorithms by varying these choices. They are then carefully evaluated and compared in experiments in order to identify the best approach. In addition, the three proposed *non-myopic* approaches outperform the best known approach in the literature (Mori, Alexander Mendiburu, Dasgupta, et al., 2017) evaluated on the same collection of datasets proposed by the authors.

The rest of the chapter is organized as follows. The following section reformulates in a generic way the cost-based formalism leading to an optimization criterion. Then, Section 3.3 presents the new methods that allow finer estimations of this criterion. These methods vary depending on how they take advantage of the training set of complete time series to estimate future costs of decision. This gives rise to a set of questions as what are the characteristics that most drive the performance up. Section 3.4 highlights some key ideas to extend ECONOMY -γ to deal with multi-class classification problems. Then Section 3.5 presents experiments and results for binary classification problems, while Section 3.6 is reserved for multi-class classification problems. Finally, Section 3.7 concludes by underlying the main findings of this research and by discussing directions for future works.

3.2 ECONOMY framework

Classifying time series with as little measurements as possible implies optimizing a trade-off. The less data is available for classification, the lower is the attainable accuracy in general. But waiting for more data implies incurring higher delay costs. There is therefore an optimization problem to be solved involving both the classification accuracy, which translates into a misclassification cost, and the cost associated with gathering measurements. In the framework of online classification, this optimization problem becomes an online sequential decision making problem, where at each new time step, and a corresponding new piece of data, the system must decide whether to output a label for the incoming time series or to wait for more measurements. If the decision is made solely on the basis of the currently available information, the approach is myopic. On the other hand, if the decision involves some sort of prediction about the expected value of the optimization criterion in the future, the approach is non-myopic. This is what is allowed by the LUPI framework. Such a perspective was presented in (Dachraoui, Bondu, and Cornuéjols, 2015). We generalize it in this section and lay the ground for three novel cost-based optimization criteria that are the object of this chapter.

This approach assumes that the user provides two cost functions and a collection of classifiers $\{h_t(\mathbf{x}_t)\}_{t\in[1,T]}$ capable of predicting at any moment $\hat{y}$ the class label of the complete time series $\mathbf{x}_T$ from the incomplete time series $\mathbf{x}_t$ given as input:

- $\mathrm{C}_m(\hat{y}|y) : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}$ is the *misclassification cost function* that defines the cost of predicting $\hat{y}$ when the true class is y.
- $\mathrm{C}_d(t) : \mathbb{R} \to \mathbb{R}$ is the *delay cost function* which is non decreasing over time.

Both of these costs are expressed in the same unit (e.g. in dollars) and convey the characteristics of the application domain as known by experts.

The expected cost of a decision (i.e. triggering the classification of the time series at hand) at time step t, when $\mathbf{x}_t$ is the incoming time series, can be expressed as:

$$\begin{aligned} f(\mathbf{x}_t) &= \mathbb{E}^{\,t}\left[\mathrm{C}_m|\mathbf{x}_t\right] + \mathrm{C}_d(t) \\ &= \sum_{(y,\hat{y})\in\mathcal{Y}^2} P_t(\hat{y}, y|\mathbf{x}_t)\mathrm{C}_m(\hat{y}|y) + \mathrm{C}_d(t) \\ &= \sum_{y\in\mathcal{Y}} P_t(y|\mathbf{x}_t) \sum_{\hat{y}\in\mathcal{Y}} P_t(\hat{y}|y, \mathbf{x}_t)\, \mathrm{C}_m(\hat{y}|y) + \mathrm{C}_d(t) \end{aligned} \tag{3.1}$$

The expectation comes both from the misclassification probability $P_t(\hat{y}|y, \mathbf{x}_t)$ which can be estimated by the confusion matrix of the classifier $h_t(\cdot)$ applied at time t, and the posterior probability of each class given the input incomplete time series estimate $P_t(y|\mathbf{x}_t)$.

If the input time series was fully observed, this cost could be computed for all time steps $t \in \{1, \ldots, T\}$, and the optimal time t^* for triggering the classifier's prediction would be:

$$t^* = \underset{t\in\{1,\ldots,T\}}{\mathrm{ArgMin}}\, f(\mathbf{x}_t) \tag{3.2}$$

But of course, this would defeat the whole purpose of early classification, as one would have to observe the entire time series before knowing what would have been the optimal decision time. Then, instead of waiting until the entire time series is known, at each time t, one could "look into the future" and guess what will be the

best decision time. And if the estimated best decision time matches the current time step t, then the decision must be made. For the incoming time series $\mathbf{x}_t$, the expected cost at τ time steps in the future is:

$$f_\tau(\mathbf{x}_t) = \sum_{y \in \mathcal{Y}} P_{t+\tau}(y|\mathbf{x}_t) \sum_{\hat{y} \in \mathcal{Y}} P_{t+\tau}(\hat{y}|y, \mathbf{x}_t)\, \mathrm{C}_m(\hat{y}|y) + \mathrm{C}_d(t+\tau) \tag{3.3}$$

$$\tau^* = \underset{\tau \in \{0,\dots,T-t\}}{\mathrm{ArgMin}} \; f_\tau(\mathbf{x}_t) \tag{3.4}$$

and if $\tau^* = 0$ the decision is instantly requested, and $\widehat{t^*} = t$ denotes the trigger time. The problem now is *how to estimate* $P_{t+\tau}(\hat{y}|y, \mathbf{x}_t)$ *and* $P_{t+\tau}(y|\mathbf{x}_t)$ *from the knowledge of* $\mathbf{x}_t$. Can the LUPI framework help? Yes it can. The cost-based formalism proposed (Dachraoui, Bondu, and Cornuéjols, 2015) which consists in learning typical clusters of time series from the training set, and then in predicting the likely continuations of $\mathbf{x}_t$ with regard to these groups (see Figure 3.2).

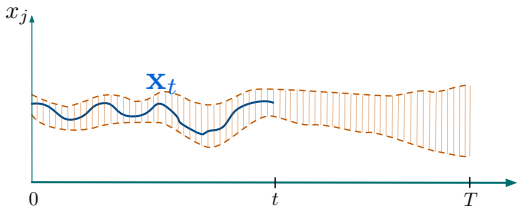


FIGURE 3.2: The incoming time series $\mathbf{x}_t$ is viewed as a member of or close to some group(s) of times series, and this is used to guess the "envelope" of its foreseeable futures.

Let us note $\mathfrak{g}_k$ the k-th typical groups of time series, Equation (3.1) then can be re-expressed as:

$$f(\mathbf{x}_t) \approx \sum_{\mathfrak{g}_k \in \mathcal{G}} P_t(\mathfrak{g}_k|\mathbf{x}_t) \sum_{y \in \mathcal{Y}} P_t(y|\mathfrak{g}_k) \sum_{\hat{y} \in \mathcal{Y}} P_t(\hat{y}|y, \mathfrak{g}_k) \mathrm{C}_m(\hat{y}|y) + \mathrm{C}_d(t) \tag{3.5}$$

And similarly, for Equation (3.3):

$$f_\tau(\mathbf{x}_t) \approx \sum_{\mathfrak{g}_k \in \mathcal{G}} P_{t+\tau}(\mathfrak{g}_k|\mathbf{x}_t) \sum_{y \in \mathcal{Y}} P_{t+\tau}(y|\mathfrak{g}_k) \sum_{\hat{y} \in \mathcal{Y}} P_{t+\tau}(\hat{y}|y, \mathfrak{g}_k)\, \mathrm{C}_m(\hat{y}|y) + \mathrm{C}_d(t+\tau) \tag{3.6}$$

Equation (3.6) can be easily interpreted by splitting it into two parts. The first term $P_{t+\tau}(\mathfrak{g}_k|\mathbf{x}_t)$ estimates the posterior probabilities of each group given $\mathbf{x}_t$. The next term expresses the expectations of the cost of misclassification over future possible continuations of $\mathbf{x}_t$. Namely, the second term $P_{t+\tau}(y|\mathfrak{g}_k)$ corresponds to the prior probabilities of class values within each group assumed to be constant $\forall t \in [1, T]$. And the third term $P_{t+\tau}(\hat{y}|y, \mathfrak{g}_k)$ estimates the probabilities of misclassification within each group, at time step $t + \tau$. The terms $\mathrm{C}_m(\hat{y}|y)$ and $\mathrm{C}_d(t+\tau)$ are the cost functions expressing properties of the domain of application.

In this general framework, several choices can be made to implement this optimization criteria. Foremost is the determination of relevant groups $\mathfrak{g}_k$ of time series from the complete training set $\mathcal{D}_{train}$. In what follows, we propose four different alternatives to anticipate the expected future misclassification costs.

3.3 Implementation

This section presents three novel non-myopic approaches to solve cost-based optimization problem. These are three ways of Learning Using Privileged Information (LUPI) and therefore being able to foresee likeliest future values for the optimization criterion. The characteristics of these approaches are summarized in Table 3.1.

Approaches	Partition type	Partitions number	Anticipation type
ECONOMY-K (Dachraoui, Bondu, and Cornuéjols, 2015)	**Unsupervised** *(K-means)*	**1** partition of full-length time series.	**Weak model**: the continuations of time series are used in the groups.
ECONOMY-MULTI-K	**Unsupervised** *(K-means)*	**T** partitions corresponding to the different time steps.	**Weak model**: the continuations of time series are used in the groups.
ECONOMY-γ-LITE *(binary classification)*	**Supervised**: quantiles of the classifiers confidence levels are used	**T** partitions corresponding to the different time steps.	**Weak model**: the continuations of time series are used in the groups.
ECONOMY-γ *(binary classification)*	**Supervised**: quantiles of the classifiers' confidence levels are used	**T** partitions corresponding to the different time steps.	**Markov Chain** technique is used to anticipate missing measurements.

TABLE 3.1: Overview of the design choices of the different approaches: each approach differing from the previous one by only one design choice.

The three proposed approaches seek to better extract information about the foreseeable future of incoming time series so as to offer a better basis for deciding whether to label the time series at the current time step or to wait for more measurements. As can be seen from Table 3.1, each proposed approach can be viewed as an incremental modification of a previous method, from ECONOMY-K presented in (Dachraoui, Bondu, and Cornuéjols, 2015) to ECONOMY-γ.

Where ECONOMY-K is computing partitions of the time series based on the complete training time series, through a clustering process, ECONOMY-MULTI-K partitions incomplete times series for each time step in order to increase adaptiveness to the incoming and incomplete time series (see Section 3.3.2). Both methods do not use the labels of the time series in order to compute the partitions.

By contrast, both ECONOMY-γ-LITE and ECONOMY-γ use the labels in order to predict the likely future of the incoming time series. ECONOMY-γ-LITE uses the level of confidence of the classifier at time t on the incoming $\mathbf{x}_t$ in order to define groups of time series (see Section 3.3.3), whereas ECONOMY-γ uses Markov chains in order to anticipate the likely future measurements on the time series (see Section 3.3.4). The following implementations of ECONOMY-γ-LITE and of ECONOMY-γ only accommodate binary classification tasks, but extensions to multi-class problems are presented later in Section 3.4.

The number of groups K is a hyper-parameter shared by all of these approaches. In practice, it can be tuned using cross validation as detailed in Section 3.5.4. An open-source code is available for full reproducibility of the experiments presented in this chapter: `https://github.com/YoussefAch/Economy`. The following sub-sections provide the main operating principles and key ideas for each ECONOMY approach.

3.3.1 ECONOMY-K

ECONOMY-K has been introduced in (Dachraoui, Bondu, and Cornuéjols, 2015). The idea is to first identify groups $\mathfrak{g}_k$ of times series using a clustering algorithm on

the training set, here K-means with Euclidean distance. Then, given an incoming time series $\mathbf{x}_t$, the memberships $P(\mathfrak{g}_k|\mathbf{x}_t)$ are estimated using a logistic function of a distance between $\mathbf{x}_t$ and the centers of the clusters $\mathfrak{g}_k$. In order to estimate the terms $P_t(\hat{y}|y, \mathfrak{g}_k)$ of the confusion matrix for each time step $t \in [1, T]$, a collection of classifiers $\{h_t\}_{t\in\{1,\dots,T\}}$ is learned using training sets $\mathcal{D}^t_{train}$ of time series truncated to their first t measurements.

In the end, implementing the ECONOMY-K approach relies on the following choices:

- A distance function for K-means.
- A distance between an incomplete time series and cluster's center.

As explained in Section 3.2, Equation (3.6) is used to estimate the cost of deciding for future time steps $t + \tau$ ($0 \leq \tau \leq T - t$), and if τ^* given by Equation (5.4) is equal to zero or $t = T$, then a decision is triggered, otherwise a new measurement x_{t+1} is made, and the decision mechanism is called again.

Algorithm 1 Learn ECONOMY-K

Input: K: number of groups
1: build the groups $\mathfrak{g}_k$ by using K-means on a set of full-length time series $\mathcal{D}_{train}$
2: **for all** $t \in 1, \dots, T$ **do**
3: learn a classifier $h_t(\cdot)$ from a set of truncated time series $\mathcal{D}^t_{train}$
4: **for all** $g_k \in \mathcal{G}$ **do**
5: estimate the confusion matrix of $h_t(\cdot)$ in the group g_k
6: **end for**
7: **end for**

Algorithm 1 provides the pseudo-code which summarizes the learning stage of the ECONOMY-K approach. The next sections describe the algorithms 2, 3 and 4 emphasizing for each of them the single difference with the previous algorithm presented (see the italicized bold line in each algorithm).

Algorithm 2 Learn ECONOMY-MULTI-K

Input: K: number of groups
1: **for all** $t \in 1, \dots, T$ **do**
2: ***build the groups*** $\mathfrak{g}^t_k$ ***by using K-means on a set of truncated time series*** $\mathcal{D}^t_{train}$
3: learn a classifier $h_t(\cdot)$ from a set of truncated time series $\mathcal{D}^t_{train}$
4: **for all** $g_k \in \mathcal{G}$ **do**
5: estimate the confusion matrix of $h_t(\cdot)$ in the group g_k
6: **end for**
7: **end for**

3.3.2 ECONOMY-MULTI-K

Instead of grouping time series using their full-length descriptions, an alternative consists in computing the clusters $\mathfrak{g}^t_k$ for each time step t using training sets $\mathcal{D}^t_{train}$ of truncated time series from the training set $\mathcal{D}_{train}$ (see line 2 of Algorithm 2). Indeed, clustering time series on the fly, at each time step, may allow for a increased adaptiveness to the specifics of the the beginning of the time series. The term $P(\mathfrak{g}_k|\mathbf{x}_t)$ in Equation 3.6 then becomes $P(\mathfrak{g}^t_k|\mathbf{x}_t)$. The cost of potential future decisions is now estimated based on the terms $P_{t+\tau}(\hat{y}|y, \mathfrak{g}^t_k)$.

3.3.3 ECONOMY-γ-LITE

In the previous approaches, the confusion matrix with the term $P_{t+\tau}(\hat{y}|y, \mathfrak{g}_k)$ in Equation (3.6), is computed using time series in $\mathfrak{g}_k$ and potentially aggregates all confidence levels of $h_{t+\tau}$, corresponding to all possible values of the conditional probability $p(y = 1|\mathbf{x}_{t+\tau})$. If this confusion matrix was instead computed over time series that share approximately the same confidence level in their classification, the estimation of future decision costs could be much more precise. This is the motivation behind the algorithms ECONOMY-γ and ECONOMY-γ-LITE.

In these methods, the groups $\mathfrak{g}_k^t$ are obtained by stratifying the time series by confidence levels[2] of h_t (see line 3 of Algorithm 3). At each time step t, the confidence level $p(h_t(\mathbf{x}_t) = 1)$ of the classifier can take a value in $[0, 1]$. Examining the confidence levels for all time series in the validation set $\mathcal{D}_{val}^t$ truncated to the first t observations, we can discretize the interval $[0, 1]$ into K equal frequency intervals, denoted $\{I_t^1, \ldots, I_t^K\}$. For instance, if $K = 5$, and $|\mathcal{D}_{val}^t| = 1000$, the intervals $I_t^1 = [0, 0.30[$, $I_t^2 = [0.30, 0.45[$, $I_t^3 = [0.45, 0.58[$, $I_t^4 = [0.58, 0.83[$, $I_t^5 = [0.83, 1]$ could each correspond to 200 training time series.

Given an incoming time series $\mathbf{x}_t$, the classifier h_t is used to get an estimate of $p(y = 1|\mathbf{x}_t)$ and determine the group $\mathfrak{g}_k^t$ to which $\mathbf{x}_t$ belongs. The algorithm is the same as ECONOMY-MULTI-K, only with the groups $\mathfrak{g}_k^t$ obtained in a supervised way by leveraging the information about the membership to the classes.

One can notice that, in addition to the expected gain in performance due to a more informed grouping of time series than in the clustering-based approaches, this method as well as ECONOMY-γ, does not require (i) the choice of a distance function for K-means, nor (ii) the determination of another distance between an incomplete time series $\mathbf{x}_t$ and a cluster of full-length time series, and finally (iii) neither the choice of a membership function in order to estimate $P(\mathfrak{g}_k|\mathbf{x}_t)$. The approach is therefore much simpler to implement.

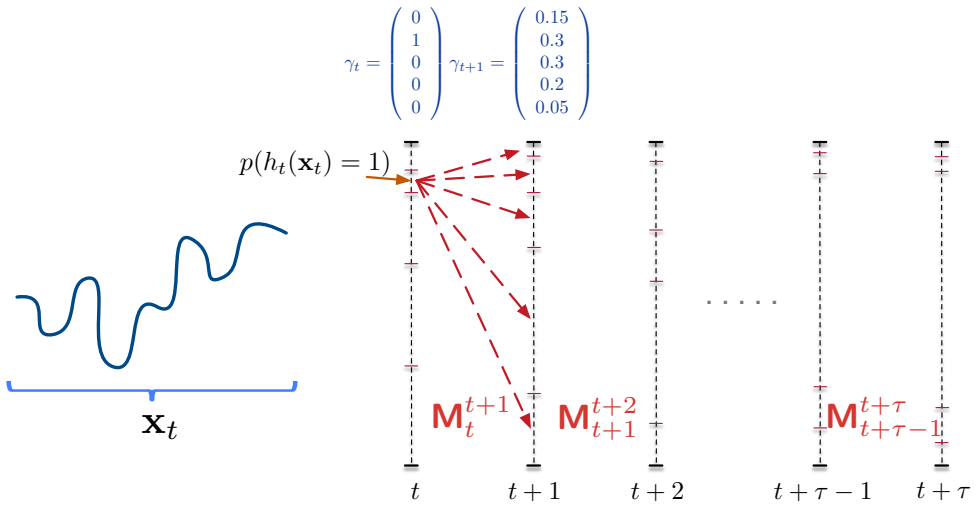


FIGURE 3.3: ECONOMY-γ, computing the probability distribution $p(\gamma_{t+\tau}|\gamma_t)$. Here $h_t(\mathbf{x}_t)$ falls in the second confidence level interval. Given a supposed learned transition matrix M_t^{t+1}, the next vector of confidence levels will be $(0.15, 0.3, 0.3, 0.2, 0.05)^\top$.

[2] This restricts these methods to binary classification problems.

Algorithm 3 Learn ECONOMY-γ-LITE

Input: K: number of groups
1: **for all** $t \in 1, \ldots, T$ **do**
2: learn a classifier $h_t(\cdot)$ from a set of truncated time series $\mathcal{D}^t_{train}$
3: ***build the groups*** $\mathfrak{g}^t_k$ ***by discretizing the confidence level of*** $h_t(\cdot)$ ***into*** K ***intervals***
4: **for all** $g_k \in \mathcal{G}$ **do**
5: estimate the confusion matrix of $h_t(\cdot)$ in the group g_k
6: **end for**
7: **end for**

3.3.4 ECONOMY-γ

ECONOMY-γ uses the ECONOMY-γ-LITE principle to assign an incoming time series $\mathbf{x}_t$ to a given group $\mathfrak{g}^t_k$, but it tries to get better estimates of the future terms $P_{t+\tau}(\hat{y}|y, \mathfrak{g}^t_k)$ of the confusion matrices by replacing $\mathfrak{g}^t_k$ by a projection $\mathfrak{g}^{t+\tau}_k$ into the future as a probability distribution over the confidence intervals of $h_{t+\tau}$.

Let us call $\gamma_t = (\gamma^1_t, \ldots, \gamma^K_t)^\top$ the real-value vector of K components γ^i_t, where each of the components is the probability $p(h_t(\mathbf{x}_t) \in I^i_t)$. For instance, in Figure 3.3, $\gamma_t = (0, 1, 0, 0, 0)^\top$, all components are zero except $\gamma^2_t = 1$.

We would like to compute the vectors $\gamma_{t+\tau}$ $(0 < \tau \leq T - t)$ consisting of the components:

$$\gamma^j_{t+\tau} \;=\; p(h_{t+\tau}(\mathbf{x}_{t+\tau}) \in I^j_{t+\tau}) \tag{3.7}$$

In ECONOMY-γ, we propose to estimate $\gamma^j_{t+\tau}$ by using the $K \times K$ transitions matrices $\{M^{t'+1}_{t'}\}_{t' \in \{1,\ldots,T-1\}}$ from $\gamma_{t'}$ to $\gamma_{t'+1}$ (see line 4 of Algorithm 4) where each component of the matrix is estimated by:

$$m_{i,j} \;=\; p(\, h_{t'+1}(\mathbf{x}_{t'+1}) \in I^j_{t'+1} \mid h_{t'}(\mathbf{x}_{t'}) \in I^i_{t'}\,) \tag{3.8}$$

given a validation set of time series. At time step t, and from γ_t it then becomes possible to compute $\gamma_{t+\tau}$ by:

$$\gamma_{t+\tau} \;=\; {\gamma_t}^\top \prod_{s=0}^{\tau-1} M^{t+s+1}_{t+s} \tag{3.9}$$

Like in Equation (3.6), the future expected costs of decision are estimated through:

$$f_\tau(\mathbf{x}_t) \;=\; \underbrace{\sum_{j=1}^{K} \gamma^j_{t+\tau}}_{(1)} \sum_{y \in \mathcal{Y}} P(y|I^j_{t+\tau}) \underbrace{\sum_{\hat{y} \in \mathcal{Y}} P_{t+\tau}(\hat{y}|y, I^j_{t+\tau})\; \mathrm{C}_m(\hat{y}|y)}_{(2)} + \mathrm{C}_d(t+\tau) \tag{3.10}$$

(1): for all confidence intervals $I^j_{t+\tau}$ of $h_{t+\tau}$

(2): probability of misclassification when $h_{t+\tau}(\mathbf{x}_{t+\tau}) \in I^j_{t+\tau}$

Again, a decision is triggered at time $\widehat{t^*} = t$, if $\tau^* \;=\; \mathrm{ArgMin}_{\tau \in \{0,\ldots,T-t\}} f_\tau(\mathbf{x}_t)$ is found to be 0.

3.3.5 Complexity analysis

We provide here an analysis of the computational complexities of the proposed algorithms, first in relation to the learning stage, and then with regard to the inference

Algorithm 4 Learn ECONOMY-γ

Input: K: number of groups

1: **for all** $t \in 1, \ldots, T$ **do**
2: learn a classifier $h_t(\cdot)$ from a set of truncated time series $\mathcal{D}^t_{train}$
3: build the groups $\mathfrak{g}^t_k$ by discretizing the confidence level of $h_t(\cdot)$ into K intervals
4: ***estimate the transition matrix*** M^{t+1}_t
5: **end for**

phase.

These complexities are expressed in a generic way, i.e. regardless of the classifier (of complexity denoted *Learn*) and clustering (*Clustering*) algorithms used. The single difference between two successive algorithms is underlined using bold letters in the following expressions that relate to the **learning phase**.

$$\text{ECO-K: } \mathcal{O}(T.Learn + Clustering + T.|\mathcal{D}_{train}|.Predict + |Y|.K.|\mathcal{D}_{train}|)$$

$$\text{ECO-MULTI-K: } \mathcal{O}(T.Learn + \mathbf{T}.Clustering + T.|\mathcal{D}_{train}|.Predict + |Y|.K.|\mathcal{D}_{train}|)$$

$$\text{ECO-}\gamma\text{-LITE: } \mathcal{O}(T.Learn + T.|\mathcal{D}_{\mathbf{train}}|.\mathbf{log}(|\mathcal{D}_{\mathbf{train}}|)) + T.|\mathcal{D}_{train}|.Predict + |Y|.K.|\mathcal{D}_{train}|)$$

$$\text{ECO-}\gamma\text{: } \mathcal{O}(T.Learn + T.|\mathcal{D}_{train}|.log(|\mathcal{D}_{train}|)) + T.|\mathcal{D}_{train}|.Predict + |Y|.K.|\mathcal{D}_{train}| + |\mathcal{D}_{train}|^{\mathbf{2}}.\mathbf{K^2})$$

ECONOMY-K learns a collection of T classifiers and build a partition of full-length time series using a *Clustering* algorithm with a $\mathcal{O}(T.Learn + Clustering)$ complexity. Then, confusion matrices are computed with a $\mathcal{O}(T.|\mathcal{D}_{train}|.Predict)$ complexity and the prior probability for each class in each group is computed with a $\mathcal{O}(|Y|.K.|\mathcal{D}_{train}|)$ complexity.

ECONOMY-MULTI-K has almost the same complexity and only differs by computing **T** different partitions.

ECONOMY-γ-LITE discretizes the outputs of the classifiers by sorting time series according to their confidence level with a $\mathcal{O}(T.|\mathcal{D}_{\mathbf{train}}|.\mathbf{log}(|\mathcal{D}_{\mathbf{train}}|))$ complexity.

Finally, ECONOMY-γ adds the estimation of the transition matrices computed with a $\mathcal{O}(|\mathcal{D}_{train}|^{\mathbf{2}}.\mathbf{K^2})$ complexity.

During the **inference phase**, a new measurement is received at each time step and the future costs must be estimated. All the approaches carry out this estimate in a $\mathcal{O}(T.|Y|^2.K)$ complexity, except ECONOMY-γ that uses a matrix-vector product for transition matrix to estimate future costs with a $\mathcal{O}(T.|Y|^2.K + K^2)$ time complexity.

3.4 Extending ECONOMY-γ to multi-class problems

While ECONOMY$-\gamma$ algorithm summarized an incoming time series $\mathbf{x}_t$ as a sequence of scalars $\langle p(h_1(\mathbf{x}_1) = 1, \ldots, p(h_t(\mathbf{x}_t) = 1\rangle$ in order to estimate the likely future of $\mathbf{x}_t$ as used in Equation 3.10, the extension to the multi-class problem requires using a more complex summary. Now, instead of having one scalar by time stamp, we must do with a vector of $|\mathcal{Y}|$ real values: $< p(h_t(\mathbf{x}_t) = 1), ..., p(h_t(\mathbf{x}_t) = |\mathcal{Y}|) >$ *for each time stamp*, where $\mathcal{Y}$ is the set of classes.

In this section, we propose two leads to adapt ECONOMY-γ to multi-class problems:

- Using a confidence score that aggregates the $|\mathcal{Y}|$ probabilities estimated by the classifier into a scalar value.

- Using a clustering algorithm in the vector space formed by the $|\mathcal{Y}|$ probabilities as in (Lemaire, Clérot, and Creff, 2015; Lemaire, Ismaili, et al., 2020; Zhdanov, 2019).

3.4.1 Confidence scores aggregating probabilities

A first way to adapt ECONOMY-γ is to use a confidence score which aggregates the output vector of probabilities of the classifiers into a single scalar value: *Confidence*: $\mathbb{R}^K \rightarrow \mathbb{R}$. The algorithm 4 is then slightly modified, replacing only line 3 by a new step, i.e. discretizing the output range of the *Confidence*() function into K equal frequency intervals. To do this, this function is applied to the time series of the validation set $\mathcal{D}^t_{val}$. The obtained intervals are used as the states of the Markov Chain, and the rest of Algorithm 4 remains unchanged. This section presents the four approaches we propose, which use different confidence scores.

i) The **ECO-γ-entropy** approach uses the Shannon's entropy function to compute a confidence score with $Confidence(p_1, ..., p_{|\mathcal{Y}|}) = -\sum_{i=1}^{|\mathcal{Y}|} p_i \log(p_i)$, where $p_i = p(h_t(\mathbf{x}_t) = i)$. For each validation example, the entropy value is estimated based on the conditional probabilities of the classes. A high entropy indicates scattered probability values, which corresponds to an uncertain prediction. By contrast, low probabilities on all classes except one which is dominant lead to a low entropy value which corresponds to a highly confident prediction.

ii) The **ECO-γ-gini** approach exploits the gini impurity index in a very similar way as the entropy approach. Gini index is defined as $Confidence(p_1, ..., p_{|\mathcal{Y}|}) = 1 - \sum_{i=0}^{K-1} p_i^2$. This score behaves the same way as the entropy function for the different case scenarios, and it is computationally less expensive than the entropy score.

iii) The **ECO-γ-margins** approach uses the function $Confidence(p_1, ..., p_{|\mathcal{Y}|}) = p_i - p_j$ where p_i is the maximum conditional probability and p_j the second largest, with $p_i \geq p_j$. This margin score is commonly used as a confidence score in active learning strategies (Balcan, Broder, and T. Zhang, 2007; Desreumaux and Lemaire, 2020). A large margin corresponds to a high confidence level in the prediction. Conversely, if the two highest probabilities are close, the margin is low and this corresponds to an uncertain prediction.

iv) The **ECO-γ-max** approach focuses only on the maximum probability estimated by the classifier by taking $Confidence(p_1, ..., p_{|\mathcal{Y}|}) = \max_{1 \leq i \leq |\mathcal{Y}|} p_i$. Since $\sum_{i=1}^{|\mathcal{Y}|} p_i = 1$, a high maximum probability value implies low values for the other probabilities. Conversely, an important value of the maximum probability correspond to a confident prediction. This confidence score is less sophisticated than the margin one, since it cannot differentiate the cases where the two largest probabilities are close or not. It is thus interesting to see how it behaves nonetheless.

These four approaches differ only in the confidence scores they use. One objective of the second part of experiments (Section 3.6) is to compare them and to identify the confidence score that leads to the best performance.

3.4.2 Clustering

Another way to extend the ECONOMY-γ approach to multiclass problems is to use clustering methods on the outputs of the classifiers such as to form a set $\mathcal{G}$ of groups. At time t, for the classifier h_t, the output vector $< p(h_t(\mathbf{x}_t) = 1), \ldots, p(h_t(\mathbf{x}_t) = |\mathcal{Y}|) >$ belongs to the vector space $\mathbb{R}^{|\mathcal{Y}|}$, and each validation example of $\mathcal{D}_{val}$ can

be represented by a point in this vector space whose coordinates correspond to the conditional probabilities estimated by the classifier at time t. Moreover, these points belong to a sub-variety of $\mathbb{R}^{|\mathcal{Y}|}$ which is a hyperplane defined by the equation $\sum_{i=1}^{|\mathcal{Y}|} p(h_t(\mathbf{x}_t) = i) = 1$, i.e. the estimated probabilities over all class values must sum to 1. The use of a clustering algorithm can identify dense groups belonging to this hyperplane, and which differ by their confidence level on (all or part of) the classes. This section presents two approaches using a clustering algorithm.

i) The **ECO-γ-Kmeans** approach directly applies the K-means algorithm[3] on the validation examples which are defined as probability vectors in $\mathbb{R}^{|\mathcal{Y}|}$. Line 3 of Algorithm 4 is replaced by: cluster the time series of $\mathcal{D}_{val}$ into K clusters (which correspond to the groups $\mathcal{G}$).

ii) The **ECO-γ-Kmeans-cal** approach is a variant that includes a pre-processing step consisting in calibrating the probabilities provided by the classifiers before using the K-means algorithm. In this approach, the probabilities $p(h_t(\mathbf{x}_t) = i)$ estimated by the classifier are replaced by their normalized rank, in a similar way to isotonic calibration (Flach, 2016). Intuitively, this calibration seems to be required to prevent the K-means algorithm from misidentifying dense groups due to biases in the calibration of the classifiers h_t ($1 \leq t \leq T$).

3.5 Experiments on binary classification problems

3.5.1 Goal of the experiments

The approaches presented all rely on the estimation of the best decision time according to a cost-based criterion which expresses the expected misclassification cost for future time steps plus a delay cost. This is the basis of these non-myopic strategies.

A first set of questions regards the importance and impact of the various design choices that distinguish the four ECONOMY algorithms (see first part of the experiments in Section 3.5.5):

1. Time series are partitioned in an unsupervised way for ECONOMY-K and ECONOMY-MULTI-K and in a supervised mode for ECONOMY-γ-LITE and ECONOMY-γ. Is one approach better than the other?
2. On a finer grain, is it better to cluster series using their full-length descriptions as in ECONOMY-K or on their truncated description at each time step t as in ECONOMY-multi-K?
3. Is it useful to try to have a more precise anticipation of the future of the incoming time series as is done in ECONOMY-γ compared with the simpler albeit coarser approach of ECONOMY-γ-LITE?
4. How distant the cost incurred is from the ideal optimal cost that one would have paid if one had known the whole series and therefore the best decision time? This is akin to a *regret* for not having a perfect a posteriori knowledge.

A second set of questions is whether developing non-myopic approaches brings performance gains compared to state of the art approaches that are myopic?

[3] We used the K-means algorithm provided with the $\mathcal{L}_2$ norm, and with 10 random initializations using Kmeans++ (Arthur and Vassilvitskii, 2007).

The second part of experiments (Section 3.5.5), therefore, compares the best ECONOMY approach with (Mori, Alexander Mendiburu, Dasgupta, et al., 2017) which has state of the art performance, as confirmed by a recent paper (Rußwurm, Lefevre, et al., 2019).

Our experiments are designed to answer these two sets of questions. This is why in Section 3.5.3, *two different collections of datasets* are used in the experiments dedicated to each set of questions.

3.5.2 Evaluation criterion

In order to compare the methods, it is important to consider a criterion which expresses its worth for the final user. We define a new evaluation criterion used in our experiments both to optimize K on a validation set and to evaluate the early classification approaches on a test set. In actual use, ultimately, the value of employing an early classification method corresponds to the average cost that is incurred using it. For a given dataset $\mathcal{D}_{val}$, it is defined as follows:

$$AvgCost(\mathcal{D}_{val}) = \frac{1}{|\mathcal{D}_{val}|} \sum_{(\mathbf{x}_T, y) \in \mathcal{D}_{val}} \left(C_m \left(h_{\widehat{t^*}}(\mathbf{x}_{\widehat{t^*}}) | y \right) + C_d(\widehat{t^*}) \right) \tag{3.11}$$

where $\widehat{t^*}$ is the decision time chosen by the method as the one optimizing the trade-off between earliness and accuracy. In our experiments, *AvgCost* is evaluated for each dataset and for each early classification method. Statistical tests allow us to detect significant difference in performance.

3.5.3 Datasets

Two distinct collections of datasets are used in our experiments.

The datasets for the comparison of the ECONOMY approaches:
With respect to the first set of questions presented in Section 3.5.1, about the role of the various design choices, one cannot easily measure differences of performance if the datasets only include time series that are easy to classify very early or that are hard to classify even when the whole series are known. Indeed, if this happens, all methods either decide early to output a label or wait until the end, and their performances are almost indistinguishable. In order, then, to be able to measure differences between the various online decision methods, we excluded datasets with these characteristics.

All the selected datasets come from the UEA & UCR Time Series Classification Repository[4]. First, we removed potentially correlated datasets since it is important to select only independent datasets for the use of statistical tests. We identified almost identical dataset names and sizes. For instance, the datasets "Ford A" and "Ford B" contain the same number of time series with the same length. In this case, we keep only one dataset chosen at random. Then, we learned a collection of classifiers for the remaining datasets and manually removed those for which the successive classifiers did not improve their quality over time. More specifically, we plotted the Cohen's *kappa* score (Cohen, 1960) for each possible lengths of the input time series. We removed the datasets with low and almost constant quality of classifiers over time. Please note that the removed datasets are overwhelmingly the smallest, as the low number of training examples generally leads to poor quality

[4] Available at : http://www.timeseriesclassification.com

classifiers. The 34 selected datasets and their description are available at https://tinyurl.com/ycmbxurq. Additional experiment results on a 45 datasets benchmark used in (Mori, Alexander Mendiburu, Miranda, et al., 2019) are reported in Appendix B and shows similar results.

The datasets for comparisons with the state of the art methods:
In order to be able to make direct comparison with the method described in (Mori, Alexander Mendiburu, Dasgupta, et al., 2017) we use the same datasets as they did. This benchmark consists of 45 datasets of variable sizes that come from a variety of application areas. This collection of datasets has also been used in (Schäfer and Leser, 2020) and (Mori, Alexander Mendiburu, Miranda, et al., 2019), making our experiences easily comparable to previous works.

Dataset preparation: First, the training and test sets were merged for each dataset to overcome the possibly unbalanced or biased split of the original data files. The remaining datasets were then transformed into binary classes since ECONOMY-γ and ECONOMY-γ-LITE are limited to binary classification. This was done by retaining the majority class and merging all the others. In order to reduce the computation time of the experiments and to compare datasets with time series of different lengths, we trained a classifier every 5% of the total length of the time series, instead of one classifier per time step, as done in (Mori, Alexander Mendiburu, Dasgupta, et al., 2017). Furthermore, for each dataset and for each possible length (i.e *5%*, *10%*, *15%* ... of the total length), we extracted 60 features[5] from the corresponding truncated time series in order to train the associated classifiers. To do this, we used the Time Series Feature Extraction Library (Barandas et al., 2020), which automatically extracts features on the statistical, temporal and spectral domains.

3.5.4 Experimental protocol

The datasets were divided by uniformly selecting 70% of the examples for the training set and using the remaining 30% for the test set. Furthermore, the training sets were divided into three disjoint subsets as follows: (subset a) 40% for training the various classifiers $\{h_t\}_{t\in\{1,\dots,T\}}$; (subset b) 40% for learning the meta parameters; (subset c) 20% to optimize the number of groups in $\mathcal{G}$.

(subset a) *Learning the collection of classifiers*: for each dataset, the classifiers corresponding to the possible lengths of the input time series (i.e. every 5% of the total length) were learned. The XGboost Python library [6] was used, keeping the default values for the hyper-parameters.

(subset b) *Learning the meta-parameters*: they were learned for each ECONOMY approach, except the parameter K which is optimized using (subset c). For instance, a meta-model learned by the ECONOMY-γ approach consists of: (i) the discretization into K intervals of the confidence level for each classifier (one for each possible length), and (ii) the transition matrices between a time step to the next one (i.e. every 5% of the time series length).

(subset c) *Optimizing the number of groups K*: the ECONOMY algorithms were trained by varying the number of groups between 1 to 20 and evaluated by the $AvgCost(.)$

[5] More details are available in: https://docs.google.com/spreadsheets/d/13u7L_5IX3XxFuq_SnbOZF1dXQfcBB0wR3PXhvevhPYA/

[6] XGBoost is available in: https://xgboost.readthedocs.io

criterion which represents the average cost actually paid by the user (see Equation (5.5)). The value of K which minimizes the $AvgCost(.)$ criterion has been kept.

Costs setting: the misclassification cost was set in the same way for all datasets: $C_m(\hat{y}|y) = 1$ if $\hat{y} \neq y$, and $= 0$ otherwise. The delay cost $C_d(t)$ is provided by the domain experts in actual use case. In the absence of this knowledge, we define it as a linear function of time, with coefficient, or slope, α:

$$C_d(t) = \alpha \times \frac{t}{T} \tag{3.12}$$

The larger the α coefficient, the more costly it is to wait for more measurements in the incoming time series. The delay cost $C_d(t)$ is obviously of paramount importance to control the best decision time. If α is very low, it does not hurt to wait for the whole time series to be known and $t^* = T$. If, on the contrary, α is very high, the gain in misclassification cost obtained thanks to more observations cannot compensate for the increase of the delay cost, and it is better to make a decision at the beginning of the observations. Our experiments were run over a three ranges of values of α: low time cost with $\alpha \in$ [1e-04, 2e-04, 4e-04, 8e-04, 1e-03, 3e-03, 5e-03, 8e-03]; medium time cost with $\alpha \in$ [0.01, 0.02, 0.03, 0.04, 0.05, 0.06, 0.07, 0.08, 0.09]; high time cost with $\alpha \in$ [0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0].

3.5.5 Results and analysis

This section presents the results of the experiments aimed at identifying the best design choices for the ECONOMY approaches (see Section 3.5.3).

Comparison of the ECONOMY approaches with a non adaptive baseline

As a first sanity test, it is interesting to see if the ECONOMY algorithms do indeed adapt the decision time to the incoming time series, or if they treat them all the same. In order to perform this test, each of the ECONOMY approaches is run once in its adaptive mode, and once made unable to adapt by forcing the number of groups $K = 1$ (there is thus no difference made between the series).

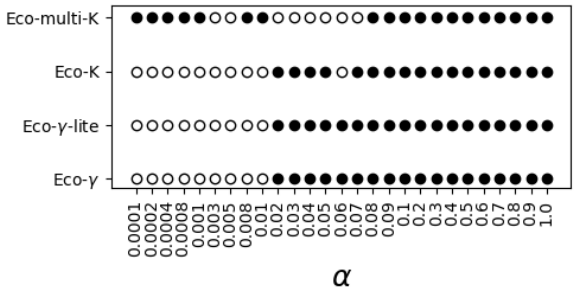


FIGURE 3.4: Success of adapting the trigger times - Wilcoxon signed-rank test results for different values of α: black dotes indicate success and circles failures.

The ECONOMY approaches are trained on the 34 selected datasets by varying the value of α, and then, evaluated on the test sets using the $AvgCost$ criterion. The Wilcoxon signed-rank test is used to assess whether the observed performance gap is significant. Figure 3.4 presents the results of the Wilcoxon signed-rank test for each ECONOMY approach, applied over the 34 datasets by varying the values of α.

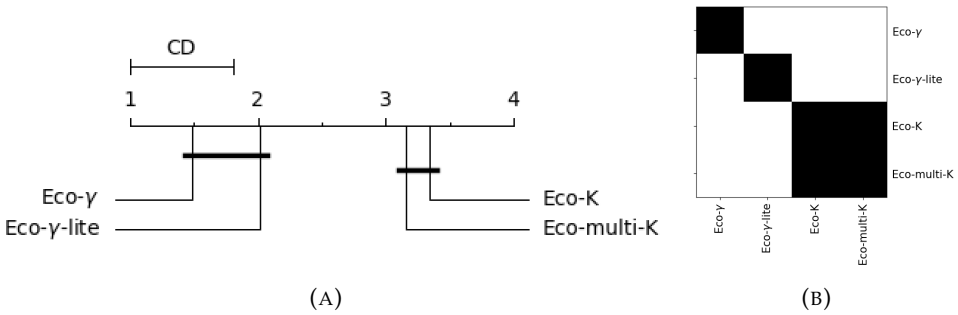
In the range $\alpha \in [0.0001, 0.01]$, ECONOMY-MULTI-K is the only approach that succeed in adapting its trigger times. By contrast, in the range $\alpha \in [0.02, 0.1]$, it appears that the ECONOMY approaches actually succeed in adapting their trigger times, with the exception of ECONOMY-MULTI-K which fails this test one-third of the time and behaves rather erratically when α varies.

At the end, these approaches succeed in improving performance by adapting their trigger times, differing in their range of success.

Comparison of the ECONOMY approaches

(a) Comparison with respect to the average decision cost

The *AvgCost* criterion was evaluated on the 34 test sets, and α was adjusted for each dataset in order to reveal the greatest differences in performance between the best and worst approach (see Table 3.2 for more details). The Nemenyi test (Nemenyi, 1962) was used to rank the different ECONOMY approaches in terms of average decision cost. The Nemenyi test consists of two successive steps. First, the Friedman test is applied to the average decision cost of competing approaches to determine whether their overall performance is similar. If not, the post-hoc test is applied to determine groups of approaches whose overall performance is significantly different from that of the other groups.



(A) (B)

FIGURE 3.5: Evaluation based on *AvgCost*: (a) Nemenyi test applied to the 34 datasets; (b) pairwise comparison using the Wilcoxon signed-rank test, with black squares identifying non-significant comparisons.

Figure 3.5 (a) reports the results of the Nemenyi test, shows two groups of methods of which the performances are significantly different. Specifically, the ECONOMY-γ and ECONOMY-γ-LITE methods exhibit much better average decision costs than ECONOMY-K and ECONOMY-MULTI-K.

Figure 3.5 (b) shows pairwise comparison using the Wilcoxon signed-rank test between the approaches. The small black squares identify pairs of approaches that do not differ significantly in performance. It is thus apparent that ECONOMY-γ performs significantly better than ECONOMY-γ-LITE.

Figure 3.7 shows the mean of the AvgCost computed over 34 datasets for ECONOMY-γ and ECONOMY-K. The superiority of ECONOMY-γ is confirmed over a whole range of α values. In addition, the perfect cost is the cost paid by the user at the first time stamp where the label is predicted correctly. No other method can do better than the perfect one. Furthermore, Figure 3.7 shows that there is still a large margin to improve those methods towards the perfect one.

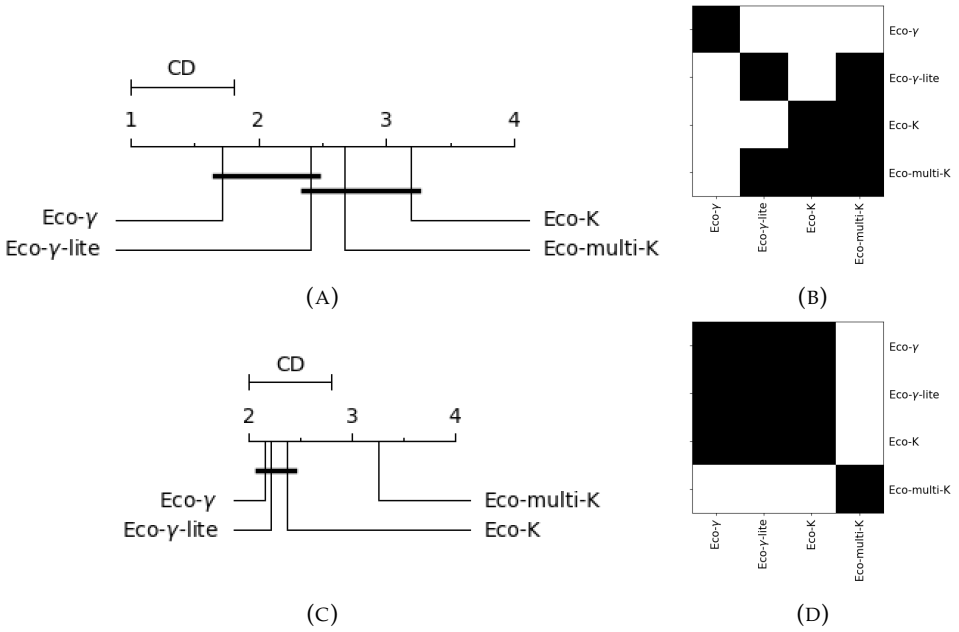


FIGURE 3.6: Earliness (a, b) and predictive performance (c, d) comparison of the ECONOMY approaches.

(b) Comparison with respect to the earliness of the decision time
In the following, the *earliness* of early classification approaches is evaluated using the median of the trigger times normalized by the length of the series, defined by $earliness = med\{\widehat{t^*}\}/T$ (see Table 3.2). Figure 3.6a shows that ECONOMY-γ, on average, triggers its decision earlier than the competing methods, followed by ECONOMY-γ-LITE. Furthermore, according to the Wilcoxon signed-rank test, this difference is significant compared to the other ECONOMY approaches (Figure 3.6b).

Figure 3.8 shows a clear decreasing tendency of the average decision moment for both methods when α gets large, which is expected since methods must decide earlier when delay cost gets large. ECONOMY-γ decides earlier on average than ECONOMY-K with a large margin, until $\alpha = 0.3$ where the two methods are close to each other with slightly earlier predictions for ECONOMY-K. The standard deviation decreases as α gets high since decisions become more straightforward. When the delay cost is very high, methods try to decide as early as possible without caring too much about adapting to the measurements.

(c) Comparison with respect to the predictive performance of the algorithms
The *predictive* performance is evaluated using the Cohen's *kappa* score (Cohen, 1960) computed at $\widehat{t^*}$, since this criterion properly manages unbalanced datasets (see Table 3.2). Again, the ECONOMY-γ and ECONOMY-γ-LITE dominate in terms of predictive performance, but here the difference is not statistically significant.

(d) Pareto curves when varying the α coefficient controlling the delay cost
In Figure 3.9, the coordinates of each point are given by the average *Kappa* score and the average *earliness* obtained over the 34 datasets when the delay cost α is chosen in the range $[10^{-4}, 1]$, and the Pareto curve is drawn for each of the approaches. The result is strikingly clear. For each value of α, ECONOMY-γ dominates all others approaches, even if ECONOMY-γ-LITE is not far behind. The ECONOMY-K and

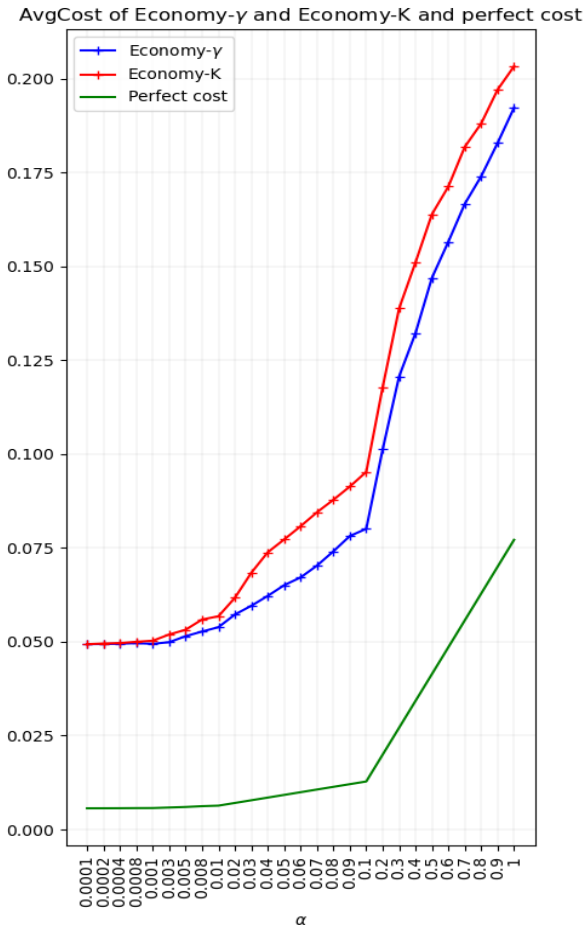


FIGURE 3.7: The mean of the AvgCost computed over 34 datasets for ECONOMY-γ and ECONOMY-K and the perfect cost which is the best an approach can do

ECONOMY-MULTI-K approaches yield much weaker results and are indistinguishable from each other.

(e) Comparing decision moments distribution

Figure 3.10 shows that ECONOMY-γ makes earlier decisions compared to ECONOMY-K even when the α parameter (delay cost) is small. For high delay costs, ECONOMY-K tends to be very early, with a distribution that seems to be less spread than ECONOMY-γ. In other words, ECONOMY-γ is more capable of adapting its decision moment for high delay costs than ECONOMY-K. These two points could explain the performance superiority of ECONOMY-γ. Similar Figures to Figure 3.10 for other values of α are given in Appendix B.

(f) Sensitivity of the number of groups chosen Figure 3.11 shows that the average of the number of groups is overall stable with respect to the α parameter, with a slight increasing tendency for high values of α for ECONOMY-γ, this might be interpreted

as the method trying to adapt its decision moment more when delay cost is high, this behavior has also been noticed in Figure 3.10 in the form of a more spread distribution of decision moments for high delay costs.

(h) Comparison with the best possible performance

The approaches presented are able to adapt their decision time $\widehat{t^*}$ to the characteristics of the time series and to perform well in terms of average decision costs $AvgCost$, but

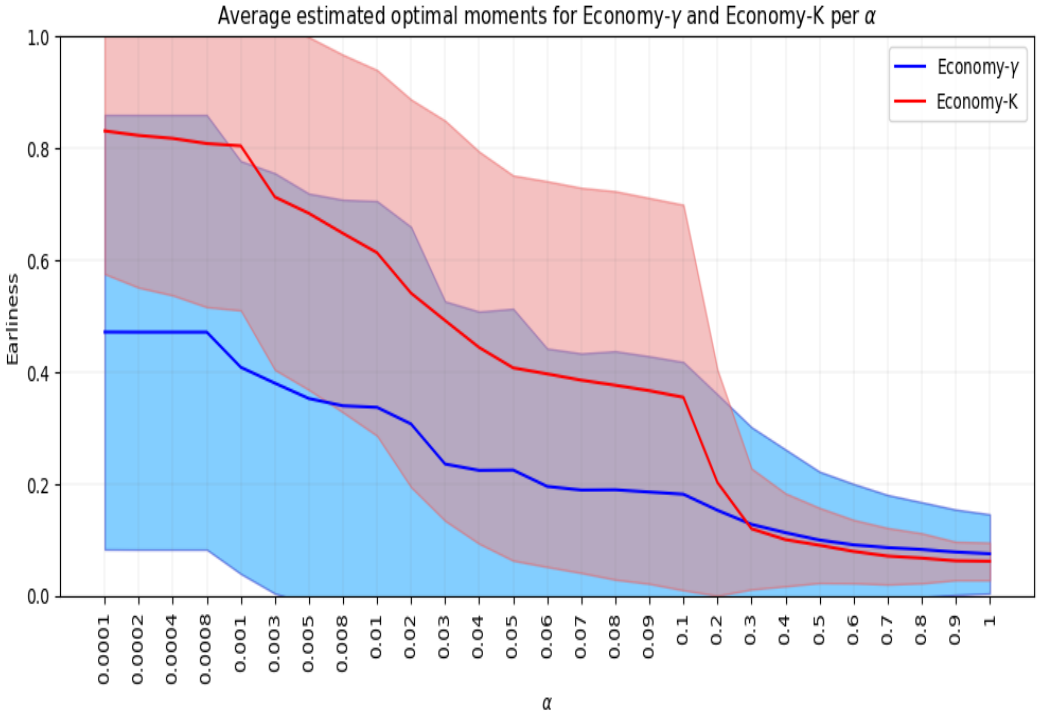


FIGURE 3.8: The average optimal decision moment chosen by ECONOMY-γ and ECONOMY-K with one standard deviation.

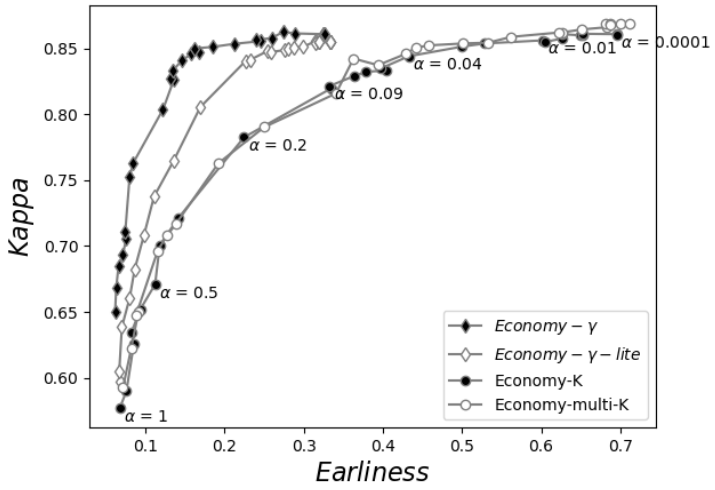
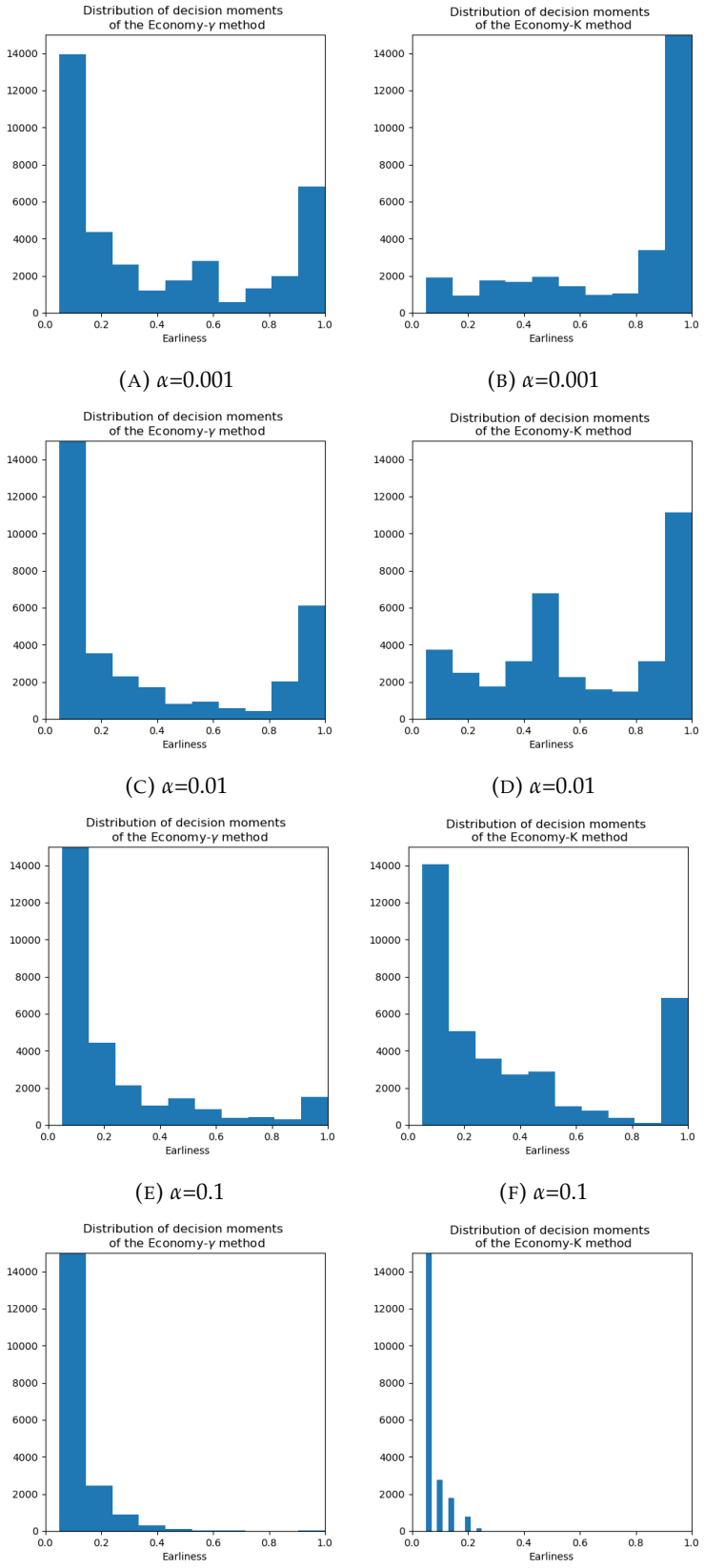


FIGURE 3.9: Average *Earliness* vs. Average *Kappa* score obtain over the 34 datasets by varying the slope of the time cost, such as $\alpha \in [10^{-4}, 1]$.



(A) α=0.001 (B) α=0.001

(C) α=0.01 (D) α=0.01

(E) α=0.1 (F) α=0.1

(G) α=1 (H) α=1

FIGURE 3.10: The distribution of decision moments for ECONOMY-γ (left column) and ECONOMY-K (right column)

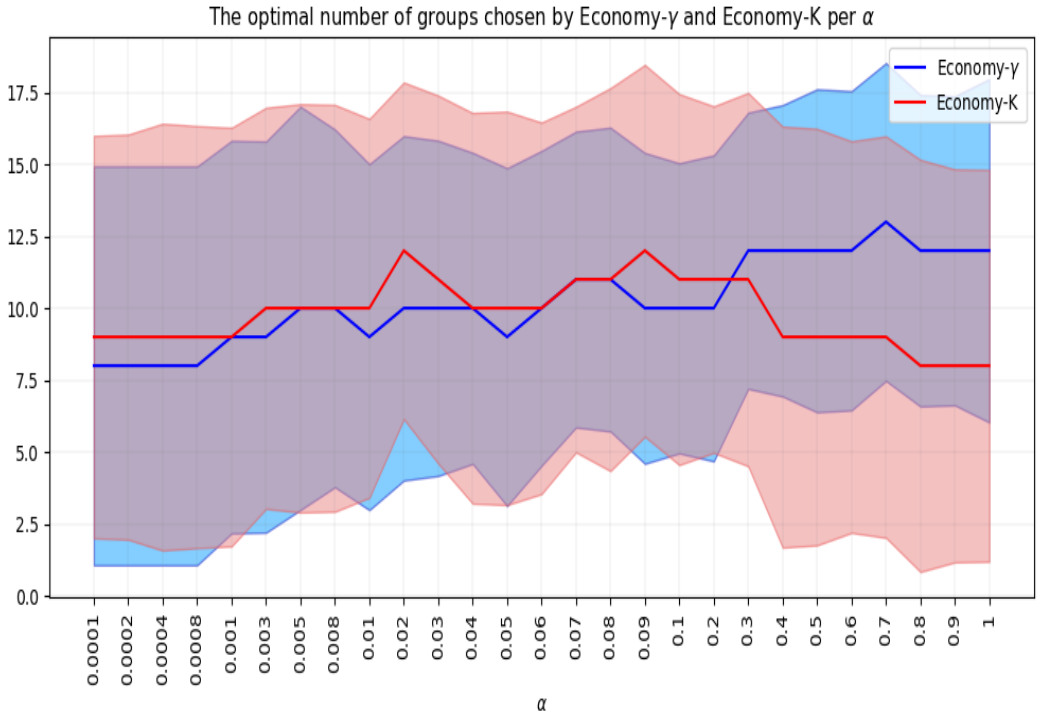
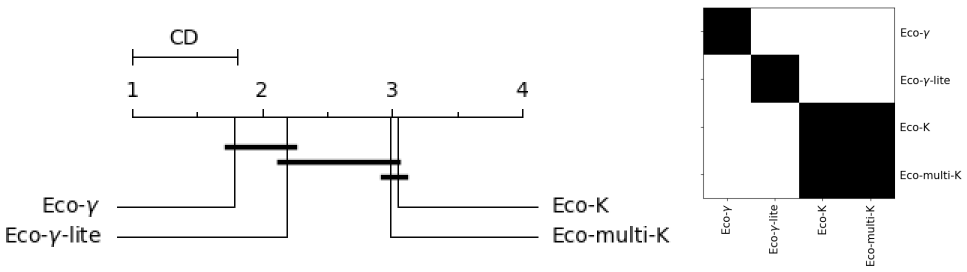


FIGURE 3.11: The average optimal number of groups chosen by ECONOMY-γ and ECONOMY-K with one standard deviation.

to what extent these results differ from the optimal ones $AvgCost^*$ computable *after* the entire time series is known? For each dataset, $\Delta_{cost} = |AvgCost - AvgCost^*|$ was computed. Figure B.15 shows that ECONOMY-γ provides the best online decisions compared to the optimal ones, on average, followed by ECONOMY-γ-LITE. According to the Wilcoxon signed-rank test, this difference is significant compared to the other ECONOMY approaches (Figure B.15b).



(A) (B)

FIGURE 3.12: Evaluation of the quality of online decisions based on Δ_{cost}.

From all these results, several conclusions can be drawn.

1. The ECONOMY approaches that partition the time series using the learned classifiers (supervised-based methods) perform significantly better than those which exploit the K-means algorithm (unsupervised-based methods).

2. For the unsupervised approaches, partitioning the time series on full-length time series (ECONOMY-K), or on truncated ones (ECONOMY-MULTI-K) does not significantly affect the performances.

3. Regarding the supervised methods, using a more sophisticated anticipation mechanism of the incoming time series as done by ECONOMY-γ is profitable and allows it to beat the less sophisticated ECONOMY-γ-LITE method

Comparing ECONOMY-γ and the state of the art

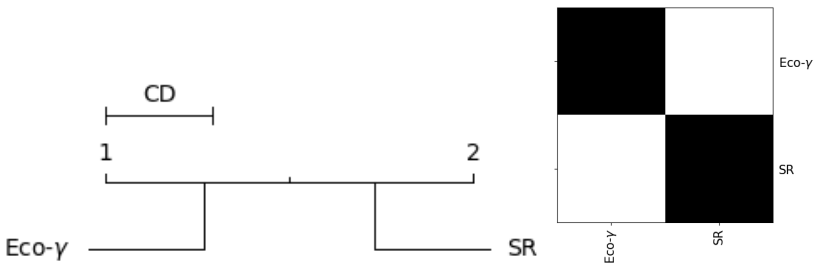
This section compares ECONOMY-γ to the state of the art approach (Mori, Alexander Mendiburu, Dasgupta, et al., 2017) and investigates the effects of the non-myopic property on its performance.

Comparing the performances An important question is whether it is worth considering explicitly, in a single optimization criterion, earliness and accuracy, as in the ECONOMY approaches, and furthermore to adopt a non-myopic strategy with the modeling and computational costs involved. To assess this, we compared the ECONOMY methods with a competing algorithm, called SR presented in (Mori, Alexander Mendiburu, Dasgupta, et al., 2017) which is claimed to dominate all other algorithms over 45 benchmark datasets. The SR algorithm uses a trigger function to decide if the current prediction is reliable (output 1) or if it is preferable to wait for more data (output 0). Among several triggered functions, all of a heuristic nature, the most effective is:

$$Trigger\left(h_t(\mathbf{x}_t)\right) = \begin{cases} 0 & \text{if } \gamma_1 p_1 + \gamma_2 p_2 + \gamma_3 \frac{t}{T} \leq 0 \\ 1 & \text{otherwise} \end{cases} \tag{3.13}$$

where p_1 is the largest posterior probability estimated by the classifier h_t: $p_1 = \text{ArgMax}_{y \in \mathcal{Y}}(\hat{p}(y|\mathbf{x}_t))$, p_2 is the difference between the two largest posterior probabilities, defined as $|\hat{p}(y=1|\mathbf{x}_t) - \hat{p}(y=0|\mathbf{x}_t)|$ in the case of binary classification problems, and where the last term $\frac{t}{T}$ represents the proportion of the incoming time series that is visible at time t.

The parameters $\gamma_1, \gamma_2, \gamma_3$ are real values in $[-1, 1]$ to be optimized. In our experiments, these parameters were tuned using a grid-search over the set of values $[-1, -0.95, -0.90, ..., 0, 0.05, ..., 0.90, 0.95, 1]$ in order to minimize the criterion *AvgCost*. The optimization was carried out for all possible time cost functions with a slope $\alpha \in [10^{-4}, 1]$.



(A) (B)

FIGURE 3.13: SR *vs.* ECONOMY-γ: evaluation based on *AvgCost*.

After training, the *AvgCost* criterion was evaluated on the 45 test sets, and α was adjusted for each dataset in order to find the most favorable setting for the SR algorithm, namely one maximizing $AvgCost_{SR} - AvgCost_{Eco-\gamma}$.

Figure 3.13a reports the results of the Nemenyi test and demonstrates that even in these situations favoring the SR algorithm, ECONOMY-γ reaches significantly better performances. The Wilcoxon signed-rank test presented in Figure 3.13b reinforces this conclusion.

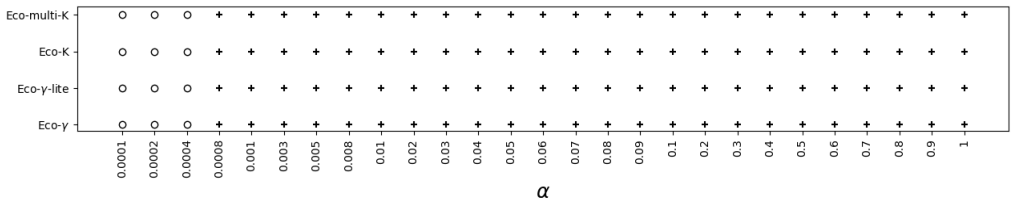


FIGURE 3.14: MORI *vs.* ECONOMY approaches : evaluation based on *AvgCost* using the Wilcoxon signed-rank test, for different values of α: "+" indicate success of ECONOMY approaches and "○" insignificant difference in performance.

We also carried out the Wilcoxon signed-rank test to compare the SR approach with the four ECONOMY approaches, for each value of $\alpha \in [10^{-4}, 1]$. The results (see Figure 3.16) shows forcibly that the ECONOMY approaches perform significantly better than the SR approach, regardless of the value of α; except for $\alpha \in \{10^{-4}, 2.10^{-4}, 4.10^{-4}\}$ where this difference is not significant.

Measuring the effect of the non-myopic property of ECONOMY One important feature that differentiates the ECONOMY approaches from the state of the art methods is being non-myopic. Where standard methods decide whether to make a prediction at the current time based only on currently available information, the Economy algorithms look at future instants in order to predict the best decision time.

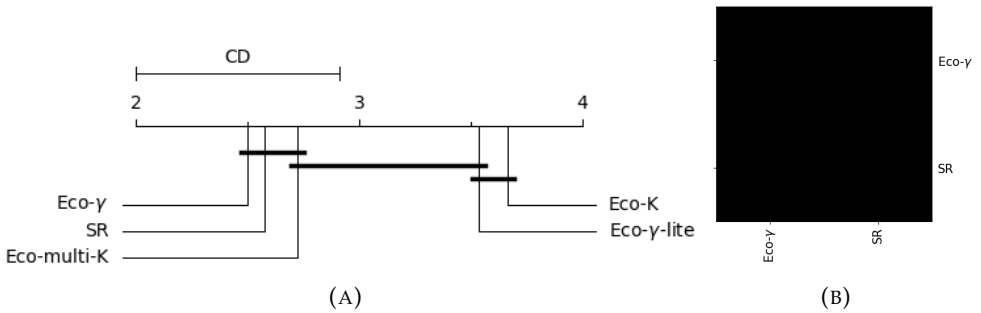
This section presents experiments aimed at answering the following question: Is it better to be non-myopic for an online decision system?

To answer this question, four myopic versions of the proposed ECONOMY approaches were implemented by limiting the horizon to only one measurement in the future, instead of looking at all the future time steps. The experiments performed with these myopic approaches are similar to those described in Section 3.5.5. The results are reported in Figure 3.15.

Figure 3.15a reports the results of the Nemenyi test between the myopic ECONOMY approaches and the SR approach of (Mori, Alexander Mendiburu, Dasgupta, et al., 2017). They are cast in the same group. Moreover, the Wilcoxon signed-rank test presented in Figure 3.15b shows that there is no statistical significant difference in performance. Consequently, it appears that the non-myopic feature is a **key property** required to obtain better results.

Datasets	α	Nb groups: K					*AvgCost*				*Median* $\widehat{t}^*$				*Kappa*			
		E-γ	E-γ-l	E-K	E-m-K		Eco-γ	Eco-γ-l	Eco-K	Eco-m-K	Eco-γ	Eco-γ-l	Eco-K	Eco-m-K	Eco-γ	Eco-γ-l	Eco-K	Eco-m-K
CBF	0.8	11	16	7	4		0.167	**0.163**	0.231	0.222	**0.09**	0.14	0.23	0.23	0.83	**0.89**	0.87	**0.89**
ChlorineConcentration	0.4	14	11	20	1		**0.252**	0.302	0.303	0.322	**0.10**	0.14	0.05	0.14	**0.59**	0.50	0.46	0.46
CinCECGTorso	0.1	20	15	1	1		0.014	**0.012**	0.021	0.021	0.05	0.05	0.05	0.05	**0.98**	**0.98**	0.96	0.96
Crop	0.06	10	19	19	18		**0.028**	0.031	0.031	0.042	0.04	0.04	0.04	0.04	0.70	0.68	0.65	**0.71**
ECG5000	0.5	18	15	1	19		**0.046**	0.051	0.065	0.064	0.05	0.05	0.05	0.05	**0.97**	0.96	0.92	0.92
ECGFiveDays	0.3	5	7	14	7		0.115	**0.097**	0.163	0.125	**0.09**	0.13	0.49	0.18	0.83	0.92	**0.95**	0.92
ElectricDevices	0.1	18	4	13	10		0.114	**0.109**	0.150	0.146	0.08	**0.04**	0.83	0.46	0.75	0.80	**0.82**	0.75
FaceAll	0.01	16	16	19	16		**0.002**	**0.002**	0.006	0.004	**0.05**	**0.05**	**0.05**	0.32	**0.99**	**0.99**	0.98	**0.99**
FacesUCR	0.5	11	2	13	18		**0.093**	0.111	0.123	0.133	**0.05**	**0.05**	0.14	0.14	**0.77**	0.67	0.69	0.72
FiftyWords	0.5	15	2	13	7		**0.104**	0.119	0.148	0.160	**0.05**	0.10	**0.05**	**0.05**	**0.68**	0.66	0.29	0.55
FordA	0.3	14	7	7	2		**0.114**	0.118	0.155	0.164	0.15	0.15	0.15	0.15	**0.88**	**0.88**	0.81	0.79
FreezerRegularTrain	0.2	17	4	7	17		**0.016**	0.021	0.024	0.025	0.05	0.05	0.05	0.05	**0.99**	0.98	0.98	0.98
HandOutlines	0.01	17	4	4	10		**0.109**	0.113	0.124	0.160	**0.25**	0.40	0.50	1.00	**0.76**	**0.76**	0.73	0.66
InsectWingbeatSound	1	19	1	1	1		0.151	**0.130**	**0.130**	**0.130**	0.05	0.05	0.05	0.05	0.25	**0.29**	**0.29**	**0.29**
ItalyPowerDemand	0.5	9	14	1	1		**0.265**	0.302	0.349	0.349	**0.04**	0.08	0.33	0.33	0.61	0.54	**0.64**	**0.64**
Mallat	0.08	9	12	10	17		**0.016**	0.019	0.034	0.028	**0.05**	**0.05**	0.30	0.15	0.97	0.96	0.93	**0.99**
Medical1mages	0.07	12	11	19	20		**0.210**	0.238	0.240	0.279	**0.08**	0.16	0.12	0.69	**0.61**	0.57	0.55	0.54
MelbournePedestrian	0.8	12	3	1	11		0.126	**0.111**	0.134	0.122	0.04	0.04	0.04	0.04	0.71	0.85	0.70	**0.86**
MixedShapesRegularTrain	0.1	20	5	3	17		0.030	**0.020**	0.047	0.044	**0.05**	0.10	0.15	0.10	0.94	**0.97**	0.91	0.92
MoteStrain	0.4	7	18	11	18		**0.128**	0.142	0.156	0.178	**0.10**	0.14	0.14	0.29	**0.86**	0.84	0.80	0.82
NonInvasiveFatalECGT2	0.04	5	19	9	7		**0.011**	**0.011**	0.014	0.012	0.05	0.05	0.05	0.05	0.81	0.80	0.71	**0.83**
PhalangesOutlinesCorrect	0.2	5	16	4	1		**0.297**	0.348	0.320	0.316	**0.15**	0.25	**0.15**	**0.15**	**0.4**	0.29	0.36	0.35
ProximalPhalanxOutlineCo	0.5	6	4	7	17		**0.267**	0.300	0.300	0.306	**0.05**	0.10	**0.05**	0.20	**0.43**	0.39	0.36	0.55
SemgHandGenderCh2	0.3	4	8	1	7		0.176	0.176	**0.174**	0.265	**0.15**	0.20	0.20	0.40	0.74	**0.77**	0.76	0.64
SonyAIBORobotSurface2	0.8	7	10	17	15		**0.176**	0.210	0.228	0.208	**0.09**	0.18	0.18	0.18	0.84	**0.85**	0.82	0.84
StarLightCurves	0.3	17	12	18	3		**0.062**	0.067	0.089	0.095	**0.05**	**0.05**	0.15	0.20	**0.93**	0.92	0.89	0.91
Strawberry	0.6	2	9	17	8		0.195	**0.193**	0.251	0.201	0.09	0.14	**0.05**	0.19	0.78	0.77	0.59	**0.83**
Symbols	0.2	15	15	15	4		0.034	**0.031**	0.057	0.042	0.05	0.05	0.05	0.05	0.97	0.98	0.89	**0.99**
TwoLeadECG	0.9	15	16	19	19		**0.160**	0.168	0.197	0.185	**0.10**	**0.10**	0.15	0.15	0.88	**0.90**	0.86	0.86
TwoPatterns	0.08	13	11	10	19		0.079	**0.065**	0.104	0.104	**0.56**	0.66	0.89	0.89	0.90	**0.95**	0.91	0.91
UWaveGestureLibraryX	0.5	12	1	18	17		**0.130**	0.148	0.153	0.126	0.05	0.05	0.05	0.05	0.42	0.14	0.22	**0.62**
Wafer	0.2	18	18	15	6		**0.010**	**0.010**	0.011	**0.010**	0.05	0.05	0.05	0.05	**1.00**	**1.00**	0.99	**1.00**
WordSynonyms	0.6	19	20	20	1		**0.177**	0.218	0.209	0.307	**0.10**	**0.10**	**0.10**	0.14	**0.63**	0.53	0.54	0.36
Yoga	0.03	14	8	8	20		**0.168**	0.246	0.183	0.193	**0.10**	0.49	0.49	0.49	**0.67**	0.54	0.66	0.65

TABLE 3.2: Details of experimental results for each dataset.



(A) (B)

FIGURE 3.15: Evaluation of the myopic version of *Economy* approaches and SR approach based on *AvgCost*: (a) Nemenyi test applied to the 45 datasets; (b) pairwise comparison using the Wilcoxon signed-rank test, with black squares identifying non-significant comparisons.

3.6 Experiments on multi-class classification problems

All the methods presented extend the ECONOMY-γ technique to multiclass problems, except ECO-K which is natively adapted to multiclass problems.

The *first question* that our experiments aim at answering is whether the proposed multi-class approaches bring significant performance gains compared to the state of the art approaches especially in case of multi-class classification problems[7].

The *second question* concerns the different ways to extend ECONOMY-γ to multi-class problems:

1. Is it a good idea to aggregate the probabilities estimated by the classifier into a scalar value? Or is it better to form the groups without aggregating these probabilities and using a clustering algorithm over the outputs of the classifier?
2. For the approaches which aggregate the estimated probabilities, which univariate confidence score leads to the best performances ?

Section 3.6.3 presents the obtained results.

3.6.1 Datasets

In order to be able to make direct comparisons with (Mori, Alexander Mendiburu, Dasgupta, et al., 2017) we use the same datasets as they did. This benchmark consists of 45 datasets of variable sizes that come from a variety of application areas. This collection of datasets has also been used in (Schäfer and Leser, 2020) and (Mori, Alexander Mendiburu, Miranda, et al., 2019), making our experiences easily comparable to previous works. We keep the 33 datasets for which the number of classes is greater than two, which is appropriate for multiclass problems.

In order to reduce the computation time of the experiments and to compare datasets with time series of different lengths, we trained a classifier every 5% of the total length of the time series, instead of one classifier per time step, as done in (Mori, Alexander Mendiburu, Dasgupta, et al., 2017). Furthermore, for each dataset and for each possible length (i.e *5%*, *10%*, ... of the total length), we extracted 60 features[8] from the corresponding truncated time series in order to train the associated

[7] Note: This was already demonstrated in the case of binary classification in Section 3.5 by comparing the ECONOMY family of algorithms with (Mori, Alexander Mendiburu, Dasgupta, et al., 2017) which is currently the best performing myopic approach, as confirmed by a recent paper (Rußwurm, Lefevre, et al., 2019)

[8] More details are available in: `https://cutt.ly/jvaKejI`

classifiers. To do this, we used the Time Series Feature Extraction Library (Barandas et al., 2020), which automatically extracts features on the statistical, temporal and spectral domains.

3.6.2 Experimental protocol

The datasets were divided by uniformly selecting 70% of the examples for the training set and using the remaining 30% for the test set. Then, the training sets were divided into three disjoint subsets as follows:

- 40% for training the collection of classifiers $\{h_t\}_{t\in\{1,\dots,T\}}$ using the Python XGboost library[9] with the default values of the hyper-parameters;
- 40% for learning the meta-parameters of the proposed approaches, which consists of: (i) the discretization of the confidence score into K intervals for each classifier, and (ii) the transition matrices between a time step to the next one (i.e. every 5% of the time series length);
- 20% to optimize the number of groups K: all the approaches were trained by varying the number of groups between 1 to 10, and evaluated by $AvgCost(.)$ (see Equation 5.5). In order to manage datasets with a large number of classes, the values $K \in \{|\mathcal{Y}|, 2|\mathcal{Y}|\}$ are also evaluated. The value which minimizes the $AvgCost(.)$ criterion has been kept.

Costs setting: the misclassification cost was set in the same way for all datasets: $C_m(\hat{y}|y) = 1$ if $\hat{y} \neq y$, and $= 0$ otherwise. The delay cost $C_d(t)$ is provided by the domain experts in actual use cases. In the absence of this knowledge, we define it as a linear function of time, with coefficient, or slope, α:

$$C_d(t) = \alpha \times \frac{t}{T} \tag{3.14}$$

The range of values used for α is $\{0.001, 0.01, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1\}$.

3.6.3 Results and analysis

First, our experiments compare the performance of the proposed approaches with the state of the art.

The **SR approach** is a very strong competitor. It was demonstrated in (Mori, Alexander Mendiburu, Dasgupta, et al., 2017) to dominate all other algorithms in the literature over a benchmark with numerous datasets. In this algorithm, a trigger function is used to decide if the current prediction is reliable (output 1) or if it is better to wait for other measures (output 0):

$$Trigger\left(h_t(\mathbf{x}_t)\right) = \begin{cases} 0 & \text{if } \gamma_1 p_1 + \gamma_2 p_2 + \gamma_3 \frac{t}{T} \leq 0 \\ 1 & \text{otherwise} \end{cases} \tag{3.15}$$

where p_1 is the largest conditional probability estimated by the classifier h_t, p_2 is the difference between the two largest probabilities and $\frac{t}{T}$ represents the proportion of the incoming time series that is visible at time t.

[9] XGBoost is available in: `https://xgboost.readthedocs.io`

The parameters $\gamma_1, \gamma_2, \gamma_3$ are real values in $[-1, 1]$ to be optimized. In our experiments, these parameters were tuned for each value of $\alpha \in [10^{-3}, 1]$ by minimizing the value of *AvgCost* thanks to a grid-search on the values $[-1, -0.90, ..., 0, 0.1, ..., 0.90, 1]$.

After training, the *AvgCost* criterion was evaluated on the 33 test sets for all values of α, both for the SR algorithm and for the proposed approaches. Then, Wilcoxon signed-rank tests were carried out to compare the SR approach with the six proposed variants of the ECONOMY approach, for each value of $\alpha \in [10^{-3}, 1]$. The results are presented in Figure 3.16, which shows that all the ECONOMY approaches perform significantly better than the SR approach, whatever the value of α.

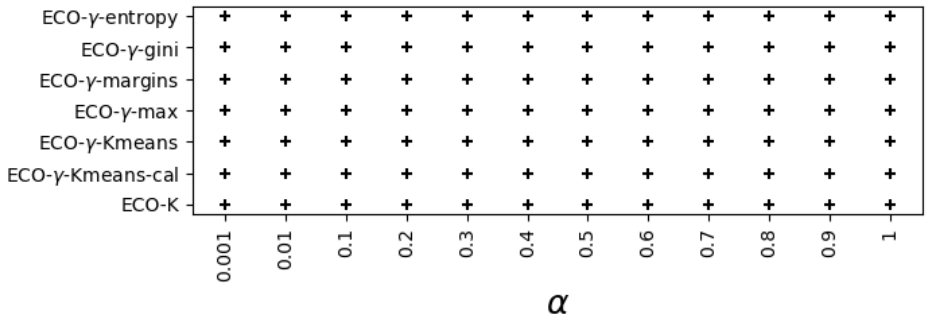


FIGURE 3.16: SR *vs.* ECONOMY approaches: the evaluation is based on *AvgCost* using the Wilcoxon signed-rank test, for different values of α. The symbol "+" indicates that all ECONOMY approaches win over the SR method. It is remarkable that the table only contains "'+".

A similar result was obtained in Section 3.5 in the case of binary classification problems. Figure 1 shows that the dominance of the Economy approaches is still verified for multiclass problems and confirms that the design choices of the proposed approaches are reasonable.

At this point, it remains to identify the best approach among those proposed, i.e., identify the best way to extend ECONOMY-γ to multiclass problems.

For this purpose, we compare each ECONOMY approach to all others and for all values of α, using the Wilcoxon signed-rank test (as in Figure 3.16). This comparison is reported in Table 3.3, where the second column counts the number of significant wins of each approach against all others; the third column counts the number of significant defeats;the fourth column reports the number of non-significant differences in performance; and, finally, the fifth column corresponds to the difference between the number of wins and the number of defeats.

Table 3.3 shows that the best performances are achieved by the ECO-γ-max and ECO-γ-gini approaches, when considering all the values of α. Actually, these two approaches have no significant defeats and have a large number of wins.

Surprisingly, the performance gap between the ECO-γ-gini and ECO-γ-entropy approaches is important. Even if these two confidence scores are mathematically very close, they do not produce exactly the same ranking of the examples and therefore the groups resulting from the discretization of these confidence scores are different.

At the other end of the spectrum, the ECO-γ-Kmeans and ECO-γ-Kmeans-cal approaches are the worst performing ones, which shows that the most promising lead to adapt ECONOMY-γ to the multiclass problems is to aggregate the classifier outputs into a confidence score.

TABLE 3.3: ECONOMY approaches comparison using Wilcoxon signed-rank test: significant wins / defeats of each approach (against all the other) counted for all α, based onthe *AvgCost* criterion.

Algorithm	wins	defeats	ties	balance
ECO-γ-max	16	0	56	+16
ECO-γ-gini	16	0	56	+16
ECO-γ-entropy	9	6	57	+3
ECO-K	8	4	60	4
ECO-γ-margins	1	9	62	-8
ECO-γ-Kmeans-cal	0	15	57	-15
ECO-γ-Kmeans	0	16	56	-16

Table 3.3 provides the ranking of the different approaches by their performance level, but this result is aggregated for all α values. The rest of the results presented in this section study the impact of the delay cost $C_d(t)$ on the ranking of these approaches.

In our experiments, the proposed ECONOMY approaches are not distinguishable for the large majority of the cases where $\alpha > 0.4$ (see Appendix B). Thus, we choose here to show detailed results for three representative cases, which correspond to $\alpha \in \{0.01, 0.1, 0.3\}$. The same results are available in Appendix B for the other α values.

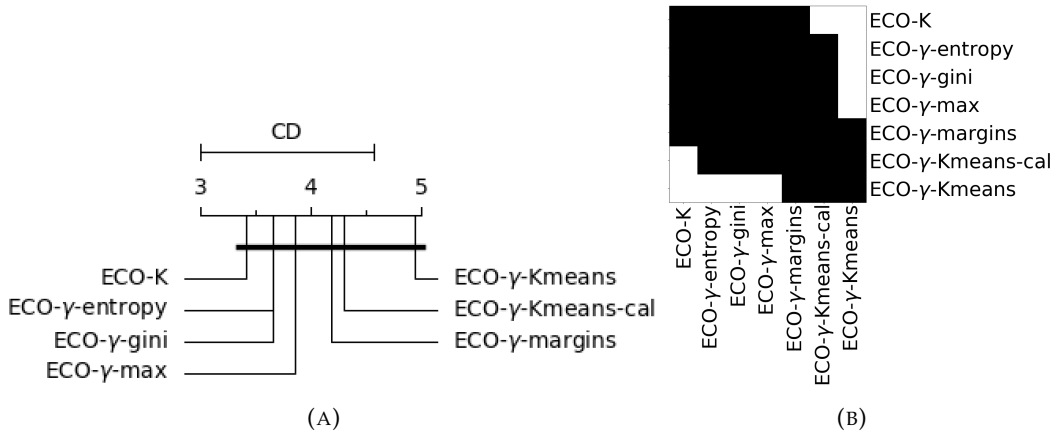


FIGURE 3.17: Comparison of ECONOMY approaches for $\alpha = 0.01$ using (a) Nemenyi and (b) Wilcoxon signed-rank tests.

Figure 3.17 (a) shows the Nemenyi test (Nemenyi, 1962) applied for $\alpha = 0.01$. This test consists of two successive steps. First, the Friedman test is applied to the *AvgCost* obtained by the competing approaches to determine whether their overall performance is similar. If not, the post-hoc test is applied to determine groups of approaches whose overall performance is significantly different from that of the other groups. In this case, the Nemenyi test is not able to show a significant difference, since all approaches belong to the same group.

Figure 3.17 (b) shows pairwise comparison using the Wilcoxon signed-rank test between the approaches. The small black squares identify pairs of approaches that do not differ significantly in performance. It appears that: (i) ECO-γ-Kmeans is dominated by ECO-K, ECO-γ-entropy, ECO-γ-gini and ECO-γ-max; (ii) the ECO-γ-Kmeans-cal is dominated only by ECO-K. These results confirm the bad ranking of

clustering based approaches observed in Table 3.3.

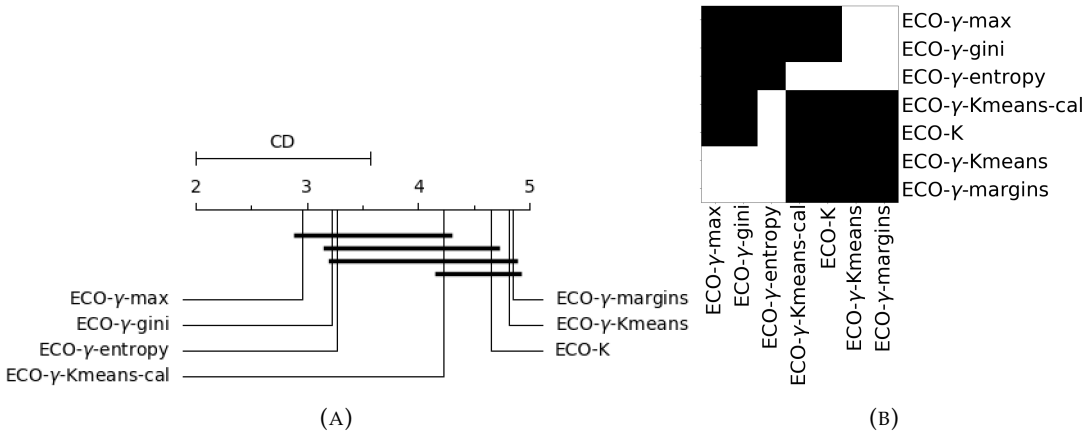


FIGURE 3.18: Comparison of ECONOMY approaches for $\alpha = 0.1$ using (a) Nemenyi and (b) Wilcoxon signed-rank tests

Figure 3.18 (a) plots the Nemenyi test for $\alpha = 0.1$, and shows that: (i) ECO-γ-max is significantly better than ECO-K, ECO-γ-Kmeans and ECO-γ-margins; (ii) ECO-γ-gini and ECO-γ-entropy are significantly better than ECO-γ-Kmeans and ECO-γ-margins. The Wilcoxon tests in Figure 3.18 (b) confirm these results, except for ECO-γ-max that is not significantly better than ECO-K.

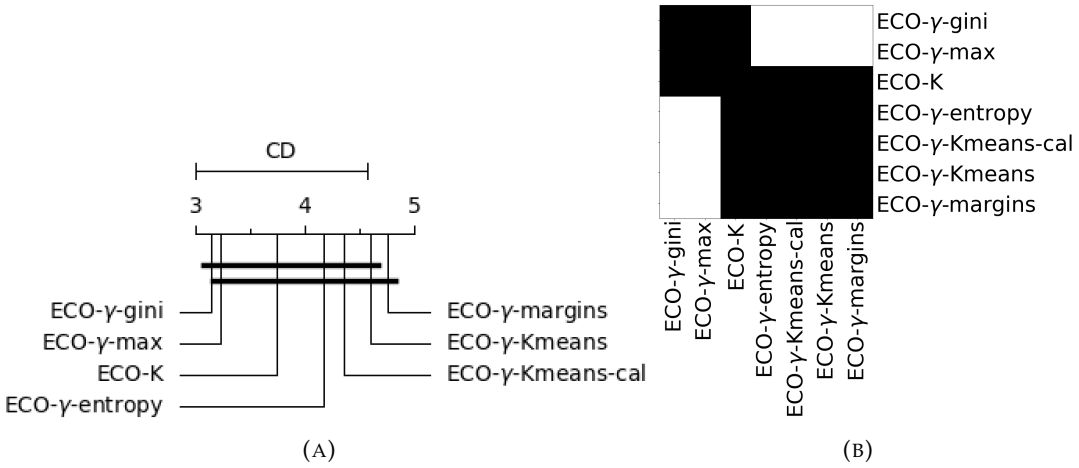


FIGURE 3.19: Comparison of ECONOMY approaches for $\alpha = 0.3$ using (a) Nemenyi and (b) Wilcoxon signed-rank tests

Figure 3.19 shows the same plots for a higher delay cost set by $\alpha = 0.3$. In this case, the approaches ECO-γ-gini and ECO-γ-max remain at the top of the ranking, and these two methods are significantly better than all the other ones except ECO-K, considering the Wilcoxon signed-rank tests.

Finally, these results based on statistical tests are in line with the results of Table 3.3, and show that the two approaches and are consistently in the top group. Henceforth, the following results compare the competing approaches by varying α in a more fine-grained way, and by evaluating both their: *(i) earliness; and (ii) predictive performance*.

For a given dataset and a given value of $\alpha \in [10^{-3}, 1]$, the *earliness* is evaluated using the median of the trigger times $\hat{t^*}$ normalized by the length of the series, defined as: $Earliness = med\{\hat{t^*}\}/T$. On the other hand, the *predictive performance* is evaluated

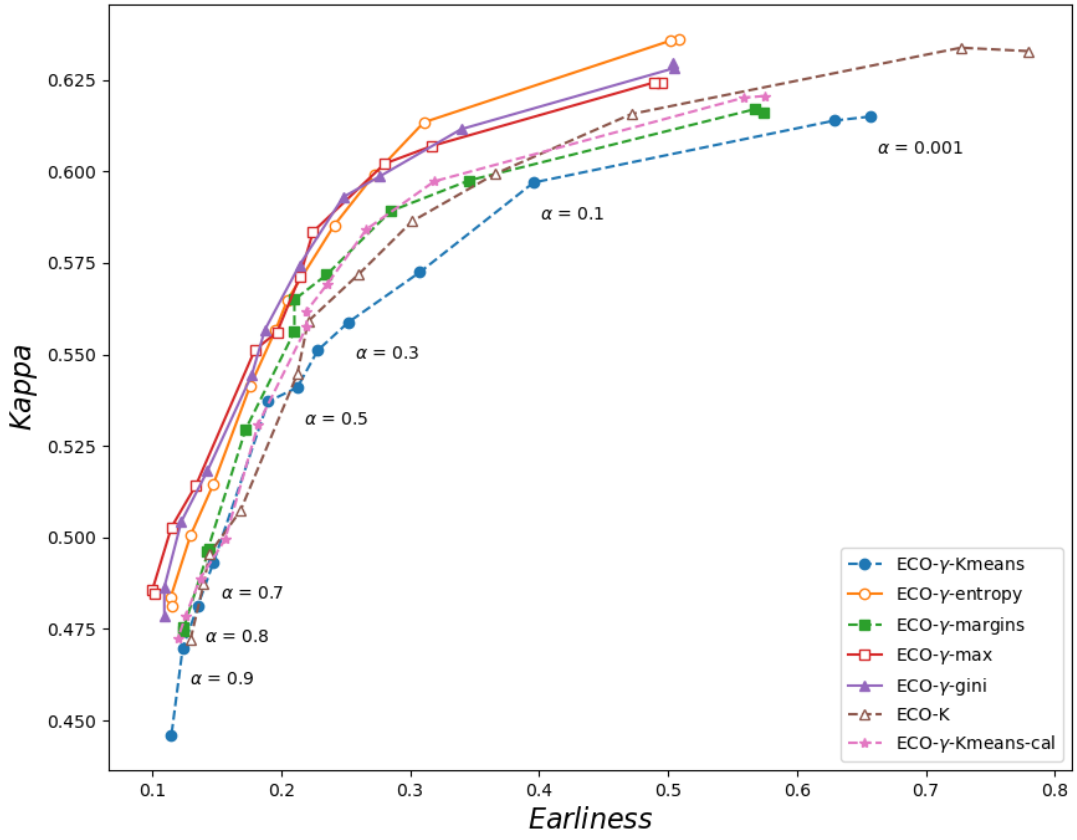


FIGURE 3.20: Average Earliness vs. Average Kappa score obtain over the 33 datasets by varying the slope of the time cost, such as $\alpha \in [10^{-3}, 1]$.

using the Cohen's Kappa score (Cohen, 1960) computed at the time of decision $\hat{t}^*$, since this criterion properly manages imbalanced datasets.

In Figure 3.20, the coordinates of each point are given by the average *Earliness* and the average Kappa score obtained over the 33 used datasets when the delay cost α is chosen in the range $[10^{-3}, 1]$. The Pareto curve is then drawn for each of the competing approaches. Two distinct groups of approaches can be identified in this figure: (i) the *top group* consists of the , , approaches; (ii) the *second group* includes the other approaches, , ECO-γ-Kmeans-cal, and . The *top group* dominates the *second group* on both *Earliness* and Kappa criteria, i.e. two curves belonging to the different groups do not intersect. In contrast, the approaches within each group can not be clearly distinguished, since the curves in the same group cross each other. Furthermore, it can be noticed that for $\alpha \geq 0.4$ the curves of the two groups are very close to each other (see the lower left part of Figure 3.20), which is consistent with previous Wilcoxon signed-rank tests that failed to significantly distinguish the performance of the competing approaches based on the *AvgCost* criterion.

3.7 Perspectives and future work

An increasing number of applications require the ability to recognize the class of an incoming time series as quickly as possible without unduly compromising the accuracy of the prediction. In this chapter, we reformulated in a generic way an optimization criterion put forward in (Dachraoui, Bondu, and Cornuéjols, 2015) which takes into account both the cost of misclassification and the cost of delaying the decision.

This generic framework has been technically declined, leading to the design of three new "non-myopic" algorithms - i.e. able to anticipate the expected future gain in information in balance with the cost of waiting. In one class of algorithms, unsupervised-based, the expectations use the clustering of time series, while in a second class, supervised-based, time series are grouped according to the confidence level of the classifier used to label them.

We have defined a new evaluation criterion that represents the average cost incurred when the method is applied over a set of labelled time series. This criterion makes it possible to evaluate both earliness and predictive performance as a single objective, with respect to the ground truth. It offers a well-grounded framework widely applicable for the comparison of methods.

Extensive experiments carried out on real datasets using a large range of delay cost functions show that the presented algorithms are able to satisfactorily solve the earliness vs. accuracy trade-off, with the supervised partition based approaches generally faring better than the unsupervised partition based ones. In addition, all these methods perform better in a wide variety of conditions than the state of the art competitive method of (Mori, Alexander Mendiburu, Dasgupta, et al., 2017). The non-myopic feature of the ECONOMY approaches is required for this good achievement. We have shown that the non-myopic property of the ECONOMY approaches plays a key role for these good performances.

Given the merit of the novel approach ECONOMY-γ, we proposed two leads to extend it to multi-class problems: (i) by using a *confidence score* that aggregates the probabilities estimated by the classifier into a scalar value; (ii) by using a *clustering* algorithm in the vector space formed by the estimated probabilities. The first lead has resulted in several competing approaches that used entropy, Gini index, margins, and maximum probability as confidence scores. In addition, we proposed two approaches

derived from the second lead which used the K-means algorithm on the probabilities estimated by the classifier, with an optional calibration step. Extensive experiments on 33 datasets of multiclass classification problems allowed us to compare the performance of the six proposed approaches to the state-of-the-art method (Mori, Alexander Mendiburu, Dasgupta, et al., 2017). Our experiments show that: (i) all proposed methods perform significantly better than the state of the art method; and (ii) the best way to extend ECONOMY-γ to multi-class problems is to use a confidence score, either the Gini index or the maximum probability.

Non-myopic approaches have shown a great potential in dealing with the early classification of time series as a supervised learning problem. The next step is to extend them to the unsupervised setting. This will open the path to more applications where it is very costly to label samples. Other techniques than Markov chains could be used to estimate the groups for future time steps for ECONOMY-γ, for example modeling it as a supervised learning problem.

While ECTS covers a wide range of applications, it has some limitations. Namely, decisions are irrevocable. Once a decision is taken, it is final, and measurements are not acquired anymore. In the next chapter, this ECTS problem will be extended to a new problem that we call *Early and Revocable time series classification*. The motivation behind this extension is to make possible the use of ECTS techniques in applications where the decision could be changed if the new measurements confirm a wrong decision taken in the past.

Chapter 4

Early and Revocable time series classification

Abstract

While the early classification of time series in the *irrevocable* regime has been addressed in several papers in the last few years, we do not know of similar works for the *revocable* regime. Nevertheless, our work on a large set of datasets and a broad spectrum of misclassification, delay, and revision costs, shows that intelligently identifying revocation instants can yield significant gains. Indeed, even with state-of-the-art early classification methods, there exist situations where additional knowledge of the unfolding time series warrants to change decisions, even in the face of increasing delay cost and additional cost of decision changes. We thus found that, depending on these relative costs, between 3% to 8% of the times series would benefit if changes of decision are made. Furthermore, a revocable strategy that considers the cost of changing decisions almost always beats a naive yet *non-myopic* revocable strategy that changes decisions without considering the decision change cost.

The content of this chapter has been published as a conference paper:

- *Early and Revocable Time Series Classification*, Y Achenchabe, A Bondu, A Cornuéjols, V Lemaire - 2022 **International Joint Conference on Neural Networks (IJCNN)**

4.1 Introduction

In the previous chapter, we presented the problem of *early classification of time series*, with a particular focus on ECONOMY approach. We put forward a general framework for the ECTS problem, and implemented it using a novel approach based on the classifier's confidence. Extensive experiments were conducted to show the significance of the new approach.

In this chapter, we focus only on one limitation of ECTS that has been mentioned in the introduction chapter, which is that decisions in ECTS are irrevocable. Once a decision is taken, the process of gathering new measurements is terminated, and a class label is predicted. Figure 4.1 illustrates the changes in the procedure, which consist of receiving new measurements even if a class label has been predicted. In many situations, however, one can make a decision and then decide to *change* it after some new pieces of information become available. The change may be costly but still warranted because it seems likely to lead to a much better outcome. This can be the case for instance when an outdoor event is canceled due to a dramatic change in the weather forecast, or when a doctor revises what now seems a misdiagnosis.

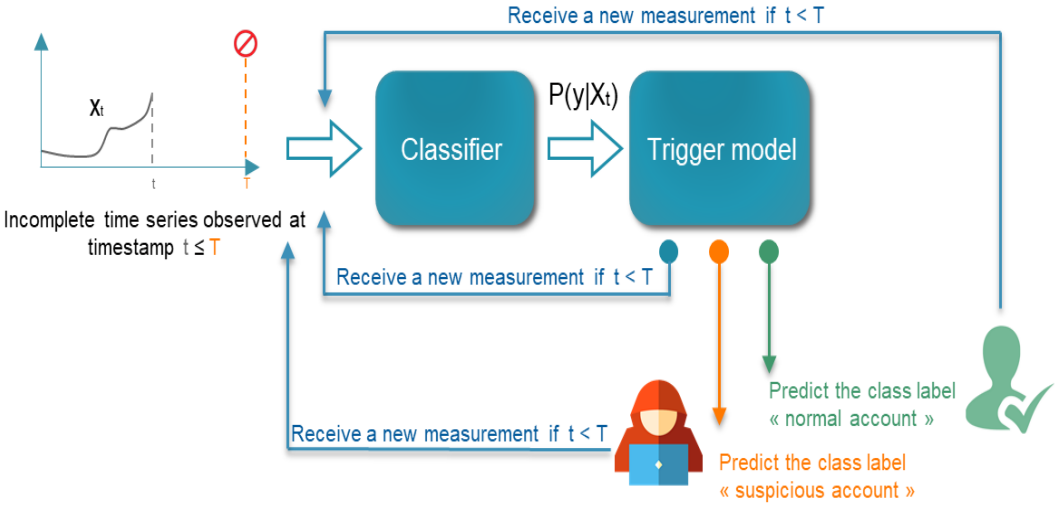


FIGURE 4.1: General schema of Early and Revocable time series classification approaches

The problem now is identifying the optimal decision sequences given an incoming series of measurements and the various existing costs.

The impact of such an intelligent revocable strategy could have on prediction maintenance, intensive care units, autonomous cars, and many more application domains where decisions have to be made optimizing costs of mistaken decisions and delay costs is quite significant.

The contribution of this chapter is threefold. *First*, it formalizes the optimization problem associated with the revocable regime for the early classification problem. *Second*, it proposes two approaches to tackle this problem, and we introduced an extended notion of *non-myopia*. Both approaches are *non-myopic* in that, to make their decisions, they take into account expectancies of the cost likely to incur in the foreseeable future:

1. The first approach is *conventionally* non-myopic, in the sense that it is only aware of the delay and misclassification costs: it is ready to revoke a decision as soon as this seems reasonable, without considering the cost of changing the decision.

2. The second approach is non-myopic of *second order*, as it estimates the future expected cost of a decision by taking into account the risk of revocation itself, which is not trivial. Specifically, a decision that will probably be revoked afterward should be delayed due to this risk. Conversely, a decision that promises to be sustainable should be anticipated.

Third, extensive experiments are presented and show that it is better to be able to revise decisions than to implement an irrevocable decision strategy. In addition, it is worth considering the non-myopic of *second order* approach.

More formally, we assume that there exists a data set $\mathcal{D}_{train} = \{(\mathbf{x}_T^i, y_i)\}_{1 \leq i \leq m}$ of *complete* time series $\mathbf{x}_T = \langle x_1, \ldots, x_T \rangle$ each of which is associated with a label $y \in \mathcal{Y}$ (e.g. patient who needs a surgical operation or patient who does not). The measurements x_i $(1 \leq i \leq T)$ belong to some input space $\mathcal{X}$ and can be univariate as well as multivariate. At each time step t, the decision-maker gets to know the time series measured so far: $\mathbf{x}_t = \langle x_1, \ldots, x_t \rangle$ and must decide either to make a prediction $\hat{y}_t$ about the class of the incoming time series or to postpone the decision.

In the *irrevocable regime* (Chapter 3), once a decision has been taken, it cannot be changed and the decision-maker endures a cost which is the sum of the misclassification cost $\mathrm{C}_m(\hat{y}_t|y)$ plus the cost of having delayed the decision until time t: $\mathrm{C}_d(t)$. Whereas, in the *revocable regime*, the decision-maker can change its prediction several times before the time limit T. Let us call $\mathcal{D}_\ell$, the sequence of the ℓ successive predictions $\langle \hat{y}_{t_1}, \ldots, \hat{y}_{t_\ell} \rangle$ made at times $t_1, \ldots, t_\ell$ in the time interval $[1, T]$. It is assumed that each decision change from $\hat{y}_{t_i}$ to $\hat{y}_{t_{i+1}}$ entails a cost $\mathrm{C}_{cd}(\hat{y}_{t_{i+1}}|\hat{y}_{t_i})$ that is greater or equal to 0.

This chapter is organized as follows. Section 4.2 presents the problem of *early and revocable classification*, and a novel algorithm to solve this problem, Section 4.3 discusses the intuitions behind the origin of the costs, this would help the user to set them. The novel approach is evaluated through extensive experiments in Section 4.4. Perspectives and future work are discussed in Section 5.5.

4.2 A new framework for revocable decisions

Suppose that while the measurements x_t about time series $\mathbf{x}_T$ unfold from time $t = 1$ to $t = T$, the decision-maker can change its mind as many times as it sees fit and ends up triggering a sequence of predictions $\mathcal{D}_\ell = \langle \hat{y}_{t_1}, \ldots, \hat{y}_{t_\ell} \rangle$ about the class of the input time series. The *final* cost incurred will be:

$$g(\mathcal{D}_\ell|\mathbf{x}_T, y) = \mathrm{C}_m\left(\hat{y}_{t_\ell}|y\right) + \mathrm{C}_d(t_\ell) + \sum_{\substack{i=1 \\ \hat{y}_{t_i}, \hat{y}_{t_{i+1}} \in \mathcal{D}_\ell}}^{\ell-1} \mathrm{C}_{cd}(\hat{y}_{t_{i+1}}|\hat{y}_{t_i}) \tag{4.1}$$

where t_ℓ is the timestamp of the last change of decision yielding the prediction $\hat{y}_{t_\ell} = h_{t_\ell}(\mathbf{x}_{t_\ell})$.

Formally, the problem is now to find a sequence of decisions $\mathcal{D}^\star \in \mathbb{D}_T$ that minimizes Equation 4.1:

$$\mathcal{D}^\star = \underset{\mathcal{D} \in \mathbb{D}_T}{\mathrm{ArgMin}}\, g(\mathcal{D}|\mathbf{x}_T, y) \tag{4.2}$$

where $\mathbb{D}_T$ is the set of all possible sequences of maximum length T.

Non-myopia of second order: When, at time t, only a partial knowledge $\mathbf{x}_t$ is available about the incoming time series. A sequence of decisions $\mathcal{D}_k = \langle \hat{y}_{t_1}, \ldots, \hat{y}_{t_k} \rangle$ has been taken so far, and the question is to see if changing the last decision $\hat{y}_{t_k}$ now, at time t, is favorable, because it would bring a better expected cost, and it would not seem better to postpone such a possible change to a later time $t + \tau$. Note that this is where *second order* considerations enter the optimization problem. In order to decide if now is a good time to change decision, one has to look if another change of decision is likely to happen in the future, at any time $t + \tau$ (see the term $P_{t+\tau}(\hat{y}|\hat{y}_{t_k}, \mathbf{x}_t)$ of Equation 4.4).

The cost of adding a new decision at time $t + \tau$, can be estimated as:

$$f^{rev}_{\tau}(\mathcal{D}_k, t+\tau \,|\, \mathbf{x}_t) = \mathbb{E}^{t+\tau}_{(\hat{y},y)\in\mathcal{Y}^2}[\mathrm{C}_m(\hat{y}|y)|\mathbf{x}_t] + \underbrace{\sum_{\substack{i=1 \\ \hat{y}_{t_i},\hat{y}_{t_{i+1}}\in\mathcal{D}_k}}^{k-1} \mathrm{C}_{cd}(\hat{y}_{t_{i+1}}|\hat{y}_{t_i})}_{\text{cost of past changes}} + \underbrace{\mathbb{E}^{t+\tau}_{\hat{y}\in\mathcal{Y}}[\mathrm{C}_{cd}(\hat{y}|\hat{y}_{t_k})|\mathbf{x}_t]}_{\text{expected value at } t+\tau} + \mathrm{C}_d(t+\tau) \tag{4.3}$$

The expected cost of changing decision is defined as follows for $(1 \leq \tau \leq T - t)$:

$$\mathbb{E}^{t+\tau}_{\hat{y}\in\mathcal{Y}}[\mathrm{C}_{cd}(\hat{y}|\hat{y}_{t_k})|\mathbf{x}_t] \;=\; \sum_{\hat{y}\in\mathcal{Y}} P_{t+\tau}(\hat{y}|\hat{y}_{t_k}, \mathbf{x}_t)\, \mathrm{C}_{cd}(\hat{y}|\hat{y}_{t_k}) \tag{4.4}$$

Given that the notation $\mathcal{D}_{k+1}$ is used to denote the sequence of decisions $\langle \hat{y}_{t_1}, \ldots, \hat{y}_{t_k}, \hat{y}_t \rangle$, with $\hat{y}_t = h(\mathbf{x}_t)$, the criterion for changing decision at time t becomes:

$$\text{criterion} = \begin{cases} & \hat{y}_t \neq \hat{y}_{t_k} \\ \text{and} & \underset{\tau\in\{0,\ldots,T-t\}}{\text{ArgMin}}\; f^{\text{rev}}_{\tau}(\mathcal{D}_k, t+\tau \,|\, \mathbf{x}_t) \;=\; 0 \\ \text{and} & f^{\text{rev}}_{\tau=0}(\mathcal{D}_{k+1}, t \,|\, \mathbf{x}_t) < f^{\text{rev}}_{\tau=0}(\mathcal{D}_k, t_k \,|\, \mathbf{x}_t) \end{cases} \tag{4.5}$$

A decision is thus taken at time t only if:

- (i) the current prediction $\hat{y}_t$ would differ from the last one $\hat{y}_{t_k}$.
- (ii) if it seems that now is the best time to make a new decision.
- (iii) if the estimated cost with the new prediction would be less than the engaged one with the previous decision.

An interesting case occurs *when changing decision is costless*: $\forall y, y' \in \mathcal{Y} \times \mathcal{Y},\ \mathrm{C}_{cd}(y|y') \;=\; 0$. Equation 4.3 becomes:

$$f^{\text{rev}}_{\tau}(\mathcal{D}_k, \tilde{t} \,|\, \mathbf{x}_t) \;= \mathbb{E}^{t+\tau}_{\hat{y}\in\mathcal{Y}}[\mathrm{C}_m(\hat{y}|y)|\, \mathbf{x}_t] \;+\; \mathrm{C}_d(\tilde{t}) \tag{4.6}$$

Then, the strategy is to change decision when the gain in the expected misclassification cost with a new decision offsets the increased delay cost.

Now a question is: what would be the **optimal sequence of decisions**?

Theorem: [Optimal sequence of decisions] Let us assume that $\forall(y, y') \in \mathcal{Y}^2$, $C_{cd}(y|y') > 0$. Then, for any time series $\mathbf{x}_T$ of class y, the optimal sequence of decision is reduced to a **one decision** sequence where the optimal time[1] $t^\star$ is defined by: $t^\star = \text{ArgMin}_{1 \leq t \leq T}\{C_m(\hat{y}_t|y) + C_d(t)\}$.

Proof: Let $\mathcal{D}_k = \langle \hat{y}_{t_1}, \ldots, \hat{y}_{t_k} \rangle$ be a sequence of decisions taken at times $\{t_1, \ldots, t_k\}$. Then the cost paid at time T is: $\sum_{i=1}^{k-1} C_{cd}(\hat{y}_{t_{i+1}}|\hat{y}_{t_i}) + C_m(\hat{y}_{t_k}|y) + C_d(t_k)$ which cannot be less than: $C_m(\hat{y}_{t^\star}|y) + C_d(t^\star)$.

Theorem 4.2 shows that it is better to make the optimal decision at the right time rather than revoking a decision since this can only lead to sub-optimal sequences of decisions. However, in practice, the ground truth y is unknown, and it may be unavoidable to make a first decision, because it seems the optimal time to do so, only to find later that it should be changed.

It must be noted that the *criterion* (Eq. 4.2) does not specify how and when to make **the first prediction** $\hat{y}_{t_1}$. Since a decision is mandatory in the framework of decision making, we assume that a "no decision" is associated with an infinite cost: $f_\tau^{\text{rev}}(\varnothing \,|\, \mathbf{x}_t) = +\infty$, forcing a decision before T, according to the non-myopic strategy defined by $f_\tau(\mathbf{x}_t)$ in its irrevocable regime.

One goal of our research is to evaluate the added value of explicitly taking into account the cost of the changes of decision with respect to a revocable strategy which would not. Accordingly, we implemented two algorithms, based on the ECONOMY-γ algorithm presented in Chapter 3.

1. The first one is named ECO-REV-CU for cost unaware (as in Equation 4.6).
2. The second is named ECO-REV-CA for cost aware (as in Equation 4.3).

A generic algorithmic implementation of the revocable decision-making criterion as defined in Equation 4.2 is presented in Algorithm 5.

Algorithm 5 GENERIC REVOCABLE REGIME ALGORITHM

Input: K: number of groups

decisions $\leftarrow \langle \hat{y}_{t_1} \rangle$
$t_{prev} \leftarrow t_1$
for all t= t_1+1...T **do**
  $\tau^\star \leftarrow \text{ArgMin}_{\tau \in \{0 \ldots T-t\}} f_\tau^{\text{rev}}(\text{decisions}, t+\tau|\, \mathbf{x}_t)$
  $cost_{new} \leftarrow f_{\tau=0}^{\text{rev}}(\text{decisions}, t+\tau^*|\, \mathbf{x}_t)$
  $cost_{prev} \leftarrow f_{\tau=0}^{\text{rev}}(\text{decisions}, t_{prev}|\, \mathbf{x}_t)$
  if $\hat{y}_t \neq \hat{y}_{t_{prev}}$ **and** $\tau^* = 0$ **and** $cost_{new} < cost_{prev}$ **then**
    $t_{prev} = t$
    decisions $\leftarrow$ decisions $\cup\, \hat{y}_t$
  end if
end for
return *decisions*

Complexity Analysis

We present here the time complexity of the two proposed algorithms. First, let us define some notations:

- *Learn(m)*: time complexity for learning a single classifier;

[1] Actually several optimum may occur at different times, and then any one of them can be chosen.

- *Predict*: time complexity of inference phase of a classifier on a time series;
- *Partitioning*: time complexity for partitioning a set of time series;
- K: number of groups in the data partition;
- m: number of time series within the dataset.

The *training stage* consists of multiple steps: (i) learning a classifier for each timestamp, in a $\mathcal{O}(T.Learn(m))$ complexity; (ii) partitioning the training set, in $\mathcal{O}(Partitioning)$; (iii) computing predictions for all examples in the training set at each timestamp in order to compute confusion matrices, $\mathcal{O}(T.m.Predict)$; (iv) the prior of each class in each group must be computed in $\mathcal{O}(|Y|.K.m)$; (v) The expected cost of decision changes is computed for all future timestamps at each timestamp (T^2 in complexity) according to Eq.4.3 which results in a complexity of $\mathcal{O}(|Y|^2.T^2.K)$ Finally, the overall time complexity of ECO-REV-CU is $\mathcal{O}(T.Learn(m) + Partitioning + T.m.Predict + |Y|.K.m)$ and ECO-REV-CA is $\mathcal{O}(T.Learn(m) + Partitioning + T.m.Predict + |Y|.K.m + |Y|^2.T^2.K)$.

For the *testing part*, the time complexity of estimating the cost expectancy of future time step is similar to the *irrevocable* regime which is $\mathcal{O}(T^2.|Y|^2.K)$ as presented in Chapter 3. Taking into account the final decision and the intermediate predictions of the classifiers, this complexity becomes $\mathcal{O}(Predict.T^2.|Y|^2.K)$.

4.3 Origin of the costs

Early classification of time series approaches aim to trigger decisions at the right time, by reaching a good trade-off between the *earliness* and the *accuracy* of their decisions. To achieve this, a balance must be found between penalizing late decisions and penalizing prediction errors. *Decision costs* are key to make this antagonistic trade-off choice, as they allow us to evaluate the cost of waiting for new measures vs. the cost of making a decision now. The objective of this section is to understand the origins of these costs.

Figure 4.2 describes a binary ECTS problem, where the actions to be performed depend on the predicted class and are described by two *Directed Acyclic Graphs* (DAG). These DAGs characterize the sequence and the relationships between the unit tasks which compose them (e.g. task 1 must be completed before starting task 2). Here, the DAGs of tasks are *fixed*, they do not depend on the decision time.

The total cost of a decision can be decomposed by:

(i) the *delay* cost, denoted by C_d, which reflects the need to execute the DAG of actions corresponding to the new decision in a constrained time, and in a parallel way.

(ii) the *decision* cost, which corresponds to the consequences of a bad decision, or the gains of a good decision (denoted by C_m).

(iii) the *revocation* cost, which is the cumulative cost of the mistakenly performed tasks belonging to the DAG of previously made bad decisions, and which are not reusable for the new decision (denoted by C_{cd}).

These costs need to be expressed in the same unit, because they are summed up in order to reflect the quality of the decisions made and their timing.

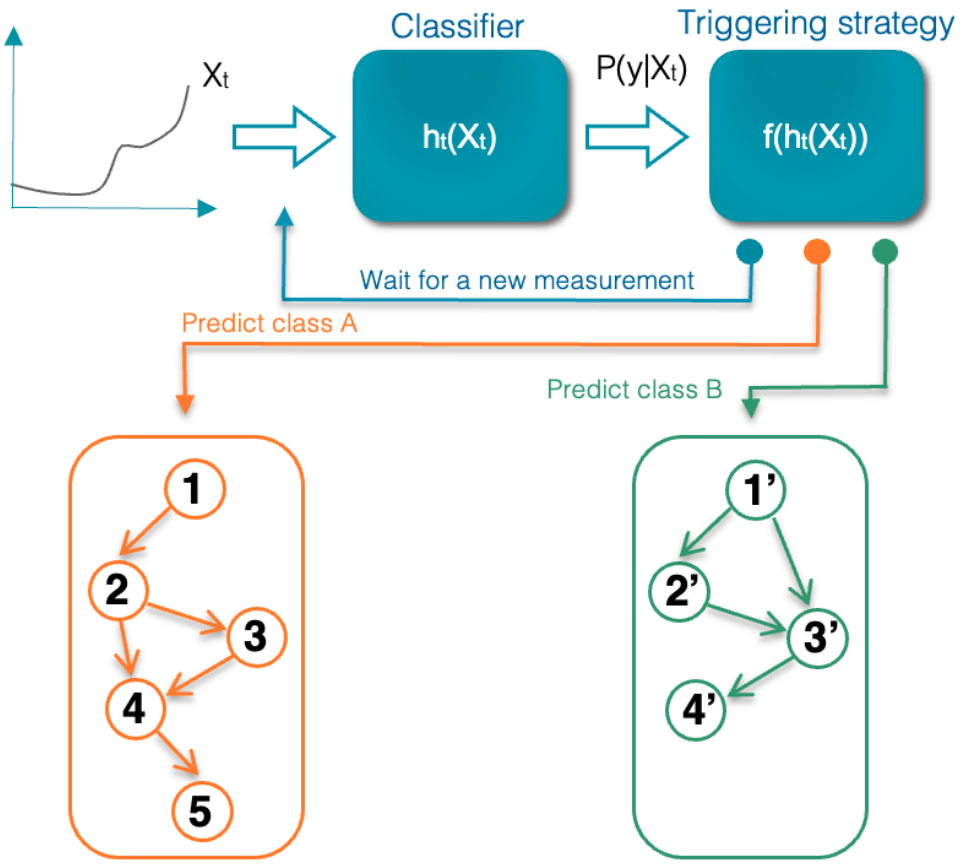


FIGURE 4.2: DAGs of tasks to be performed after the triggering of a decision.

The delay cost

It represents the cost of postponing a decision. In the particular case of ECTS problems, the delay cost is present in all the works described in scientific literature. But it can be explicitly defined as in (Achenchabe, Bondu, and Cornuéjols, 2021; Mori, Mendiburu, et al., 2015), or implicitly as in most approaches. For instance, the authors in (Xing, Pei, and Philip, 2009) trigger all the decisions at the *minimum prediction length*, which correspond to the early moment such that no prediction differs from those applied to the full-length training time series (based on a KNN classifier). This approach thus *implicitly* assumes that the delay cost is very low, by favoring the accuracy of decisions at the expense of their earliness. In (Mori, Alexander Mendiburu, Miranda, et al., 2019), the authors propose to model the trade-off between earliness and accuracy as a multi-objective criterion and explore the Pareto front of multiple dominant solutions. This approach is useful in applications where earliness and accuracy can not be evaluated in a commensurable way, and it provides a collection of optimal solutions each corresponding to a particular value of the delay cost.

For a better understanding, let us examine what happens once a decision is triggered in the simple ECTS problem. Figure 4.3 represents a *classifier* and a *triggering strategy*. At each time step $t \in [0, T]$, the classifier predicts the conditional distribution $P(y|\mathbf{x}_t)$ based on the input incomplete time series $\mathbf{x}_t = \langle x_0, x_1, \ldots, x_t \rangle$. Then, the triggering strategy either decides to *postpone* the decision until a new measurement x_{t+1} is available, or to *trigger* the decision by predicting the class value. In this first scenario, let us consider that triggering a decision at time t implies performing a given *task* (namely α or β) which depends on the predicted class (respectively A or B).

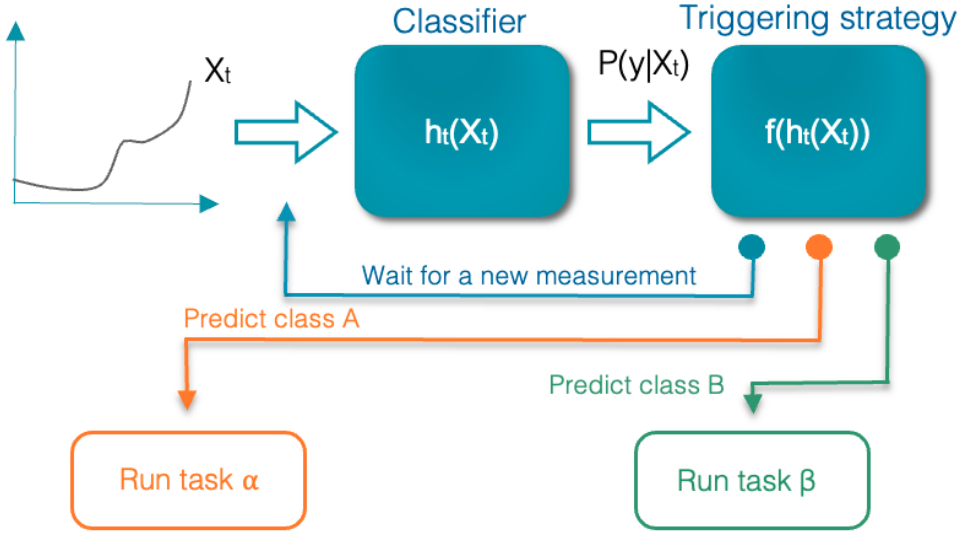


FIGURE 4.3: Tasks to be performed after the triggering of a decision.

Given that this task (α or β) must be completed *before* the deadline T, the problem is to determine how the cost of performing this task evolves depending on the trigger time t. In practice, the delay cost takes the form of a parametric function (e.g., a constant (Xing, Pei, and Philip, 2009), linear (Achenchabe, Bondu, and Cornuéjols, 2021) or exponential (Beibel, 2000) function), whose form characterizes the additional cost to delay the execution of the tasks.

A *constant cost*, one where there is no penalty associated with delaying the decision, would mean that these tasks are achievable in an arbitrarily short time $T - t < \epsilon$. In

practice, an irreducible amount of time is needed to perform the tasks using a single worker. To reduce this time, the tasks need to be parallelized using several workers, incurring an extra-cost when building the global result from sub-tasks. Formally, a constant delay cost would mean that the tasks are *infinitely parallelizable*, i.e. they can be divided into *independent* and *arbitrarily small* sub-tasks, and that there is no extra-cost in building the global result.

More generally, the delay cost is necessarily an increasing function (monotonic or piecewise) depending on the time remaining before the decision deadline, and it may depend on the decision made (i.e. the predicted label). In addition, it should tend to $+\infty$ when the time remaining to perform these tasks $T - t$ tends to zero (Beibel, 2000). For example, this delay cost may be modeled by $1/(T-t)^{\sigma}$, with a single parameter σ which influences the increase in cost when $(T - t) \to 0$.

The decision cost

Taking into account the *decision* cost is a very common feature in the literature, particularly in the field of cost-sensitve learning (Elkan, 2001). These techniques take as input a function $\mathrm{C}_m(\hat{y}|y) : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}$ which defines the cost of predicting $\hat{y}$ when the true class is y. The aim is to learn a classifier which minimizes these costs on new data.

The revocation cost

By contrast, the study of the *revocation* cost is very limited in the literature. To our knowledge, The work presented in this chapter is the only one that considers this problem, and this work shows that assigning a cost to decision changes is a first lead to manage the *reactivity vs. stability dilemma*, and to design *non-myopic to revocation risk* approaches (i.e. discussed later in challenges #8 and #9 in Chapter 6). The origin of this cost can be explained in the light of the tasks to be performed once a decision is triggered (see Figure 4.2). For instance, let us consider the first decision noted by $(A, \hat{t_1})$, in which the system predicts at time $\hat{t_1}$ that the input time series belongs to the class A. This decision is then revoked in favor of a new decision $(B, \hat{t_2})$. The cost of changing this decision, denoted by $\mathrm{C}_{cd}(B|A)$, can be defined as the cost of the actions already performed between $\hat{t_1}$ and $\hat{t_2}$ which turn out to be useless for the new decision, i.e. which cannot be reused in the DAG of tasks corresponding to the new predicted class B. In order to define the costs of decision changes, it is necessary to identify the *common tasks* between the DAGs of the different classes and to evaluate their execution time. In addition, if these tasks are *not perishable*, the entire sequence of the past decisions must be taken into account to identify the already completed tasks which are now useful for the achievement of the current DAG of tasks.

4.4 Experiments

The experiments aim at measuring the true added value of a revocable strategy. Specifically, the question is twofold. *First*, does such a strategy recognize useful changes of decisions: those that increase the performance? *Second*, does it pay off to implement a revocable strategy that takes into account the costs of changing decisions by comparison to a naive one that would not consider these costs? In the following,

we report results obtained on 34 datasets (see Section 4.4.2) for a whole range of values for the delay cost C_d and the cost incurred if changing decision C_{cd}.

4.4.1 Implementation choices

In our experiments, the ECO-REV-CU algorithm is simply the ECONOMY-γ algorithm allowed to be reiterated after each decision. It thus does not take into account the costs associated with changing decisions, whereas ECO-REV-CA does. More technically, ECO-REV-CA approximates $\mathbb{E}^{t+\tau}_{\hat{y}\in\mathcal{Y}}[C_{cd}(\hat{y}|\hat{y}_{t_k})|\mathbf{x}_t]$ in Equation 4.4 by using the groups of time series, denoted by $\mathcal{G}$:

$$\begin{aligned}\mathbb{E}^{t+\tau}_{\hat{y}\in\mathcal{Y}}[C_{cd}(\hat{y}|\hat{y}_{t_k})|\mathbf{x}_t] &\approx \sum_{\mathfrak{g}_k\in\mathcal{G}} P(\mathfrak{g}_k|\mathbf{x}_t)\mathbb{E}^{t+\tau}_{\hat{y}\in\mathcal{Y}}[C_{cd}(\hat{y}|\hat{y}_{t_k})|\mathfrak{g}_k] \\ &= \sum_{\mathfrak{g}_k\in\mathcal{G}} P(\mathfrak{g}_k|\mathbf{x}_t) \sum_{\hat{y}\in\mathcal{Y}} P_{t+\tau}(\hat{y}|\hat{y}_{t_k},\mathbf{x}_t)\, C_{cd}(\hat{y}|\hat{y}_{t_k})\end{aligned} \tag{4.7}$$

Then, the probability $P_{t+\tau}(\hat{y}|\hat{y}_{t_k},\mathfrak{g}_k)$ entering in the term $\mathbb{E}^{t+\tau}_{\hat{y}\in\mathcal{Y}}[C_{cd}(\hat{y}|\hat{y}_{t_k})|\mathfrak{g}_k]$ is estimated in a frequentist way as the proportion of time series predicted to belong to $\hat{y}_{t_k}$ at time t_k, and for which the classifier changed its decision at time $t+\tau$ by predicting the class $\hat{y}$.

4.4.2 Data and feature extraction

Experiments will be restricted to binary classification problems, and in order to be able to directly compare our results with those reported in Chapter 3, we chose to use the same 34 datasets that are taken from the UEA & UCR Time Series Classification Repository[2] (A. Bagnall et al., 2017). It is important to note that the revocable framework presented here could as well accommodate multi-class classification problems.

Each training set is built with 70% of the examples randomly uniformly selected, while the remaining 30% are used as test set (note that in each dataset, all time series have the same length). In addition, each training set is divided into three disjoint subsets: (*i*) 40% for training the Xgboost $chen_xgboost_2016classifiers\{h_t\}_{t\in\{1,\dots,T\}}$ that are the base classifiers used in the ECONOMY-γ method, which offer a good trade-off between computing time and accuracy; (*ii*) 40% for estimating the probabilities in f^{rev}_τ and f_τ; and (*iii*) the remaining 20% for optimizing the number of groups $|\mathcal{G}|$ in ECONOMY-γ which is its only hyper-parameter.

In order to give equal weight to all data sets in the comparison, it is important that they offer the same number of opportunities for decision changes. This is why the instants for potential changes are sampled every n% of the length of the times series in each data set (in our case, n= 5%). For each possible length, 60 features on the statistical, temporal and spectral domains are extracted using the Time Series Feature Extraction Library tsfel, and are used for training the classifiers $\{h_t\}_{t\in\{1,\dots,T\}}$.

4.4.3 The evaluation criterion

The cost incurred using an early classification system on a time series $\mathbf{x}_T$ is the sum of three costs, the cost of misclassification, the delay cost incurred at the time of the

[2] Available at: http://www.timeseriesclassification.com

last decision, and the sum of the costs associated with all changes of decision if any:

$$\text{Cost}(\mathbf{x}_T) = \text{C}_m(h_{t_l}(\mathbf{x}_{t_l})|y) + \text{C}_d(t_l) + \sum_{i=1}^{|\mathcal{D}_l|-1} C_{cd}(\hat{y}_{i+1}|\hat{y}_i) \tag{4.8}$$

In order to evaluate a method, we compute its mean performance on the test set $\mathcal{T}$:

$$AvgCost(\mathcal{T}) = \frac{1}{|\mathcal{T}|}\sum_{i=1}^{|\mathcal{T}|} \text{Cost}(\mathbf{x}_T^i) \tag{4.9}$$

4.4.4 Description of the experiments

In our experiments, we compared three algorithms: ECONOMY-γ which is an irrevocable decision-maker, ECO-REV-CU which is the revocable version of ECONOMY-γ but unaware of the costs of changing decision, and ECO-REV-CA which is aware of these changing costs. We are thus able to measure the added-value of the revocable strategy (ECO-REV-CU *vs.* ECONOMY-γ) and the added-value of being aware of the costs of changing decision (ECO-REV-CA *vs.* ECO-REV-CU).

For a given application, the various costs, relative to misclassifications, delays and changes of decision, must be provided by the domain expert[3]. For our experiments, we explored the performance of the three methods on a wide range of cost values:

- The *misclassification cost* was set to $\text{C}_m(\hat{y}|y) = 1$ if $\hat{y} \neq y$, and $= 0$ if not.
- The *delay cost* was assumed to be linear with a positive slope: $C_d = \alpha \times \frac{t}{T}$ starting from very low α = { 0.0001, 0.00025, 0.0005, 0.00075}, to low α = {0.001, 0.0025, 0.005, 0.0075}, to medium values α = {0.01, 0.025, 0.05 ,0.075} and to high values α = {0.1, 0.25, 0.5, 0.75, 1}.
- The *cost of changing decision* was set to $\text{C}_{cd}(\hat{y_1}|\hat{y_2}) = \beta$ if $\hat{y_1} \neq \hat{y_2}$, and $= 0$ otherwise. The parameter β being taken in the same set of values as α[4]

The AvgCost criterion defined in Equation 5.5 was evaluated on the 34 test sets for all cost values, and the Wilcoxon signed-rank test was performed for all the range of cost values, in order to assess whether the observed performance gap between methods is significant (*"+" and "-"*) or not (*"∘"*).

4.4.5 Results and analysis

Before comparing the methods, it is important to measure the proportion of time series that offer useful opportunities for revocable decisions. Those are the ones where the first decision taken by an irrevocable strategy, here ECONOMY-γ, turns out not to be optimal. For the 34 datasets under study, it turns out that (*i*) for a low delay cost $C_d = 0.0025 \times \frac{t}{T}$ only 3% of the first decisions can be usefully revoked; (*ii*) for a medium delay cost $C_d = 0.025 \times \frac{t}{T}$ this percentage rises to 3.6%; and (*iii*) for a high delay cost $C_d = 0.5 \times \frac{t}{T}$ this percentage reaches 8%. These figures show that, for these datasets and this range of cost values, opportunities for a revocable strategy to overcome an irrevocable one seem seldom. (see (Eco-rev, 2021) for a complete detailed analysis over all the 34 datasets and all couples of values (C_{cd}, C_d)).

[3] For instance, for condition of septic shock, every hour delay in antibiotic treatment leads to 8% increase in the risk of mortality (Khoshnevisan and Chi, 2021a)

[4] α and β were chosen in a very large spectrum of values so as not biasing the results.

However, the *first lesson* is that both revocable methods ECO-REV-CU and ECO-REV-CA get significantly better results than the irrevocable method ECONOMY-γ on a wide range of delay cost C_d and decision change cost values β (see Figures B.13(a) and B.13(b)). The *second lesson* is that it pays off to use a strategy which takes into account the costs of changing decision. Indeed, ECO-REV-CA beats ECONOMY-γ on a wider range of conditions than ECO-REV-CU.

Both revocable strategies fail to overcome the irrevocable one, ECONOMY-γ, when β is large (i.e. more than 0.1), and then ECO-REV-CU fails more often than ECO-REV-CA. This behavior is not surprising since, when it is very costly to delay a decision, the best strategy is generally to make a very early decision and not to revise it afterwards.

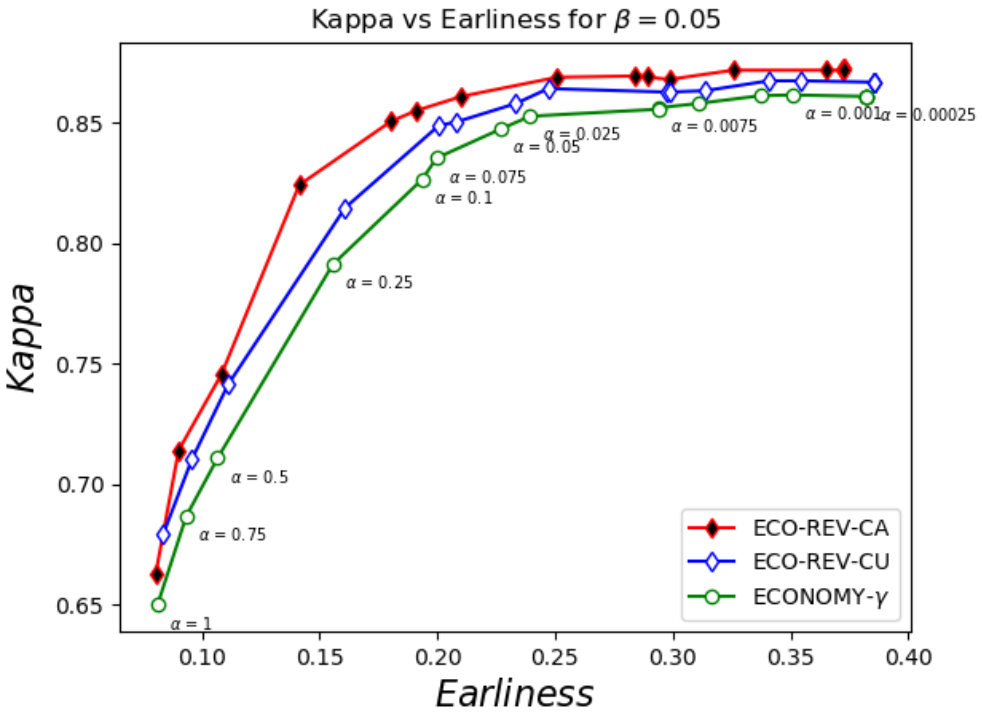
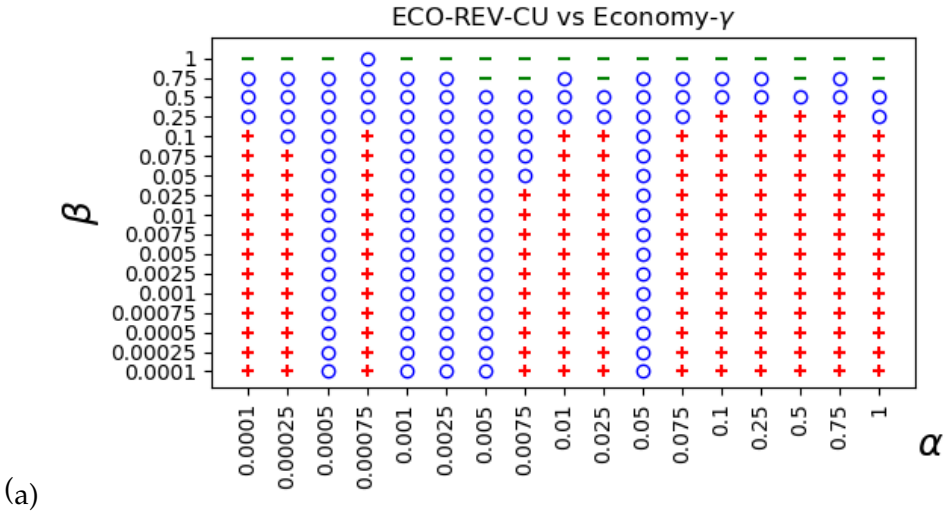


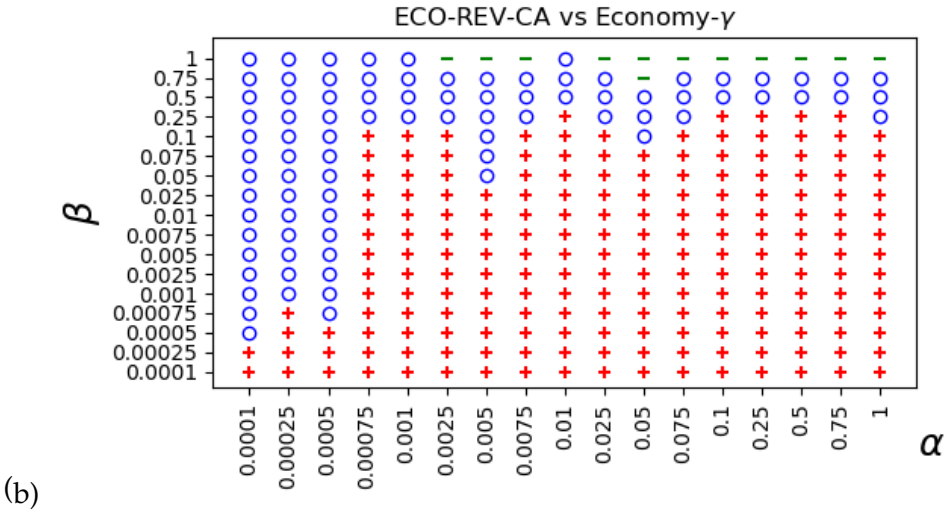
FIGURE 4.4: Average *Earliness* vs. Average *Kappa* score obtained over all the 34 datasets for $\beta = 0.05$ and by varying the slope α of the delay cost. The reader may find the same behavior for other β values in the (Eco-rev, 2021)

Figure B.13(c) shows the results of the Wilcoxon signed-rank test between the two revocable strategies. It appears that the cost aware approach ECO-REV-CA performs significantly better than the cost unaware approach ECO-REV-CU, for almost one third of the pairs of values (α, β). As the slope of the delay cost α grows, ECO-REV-CA becomes significantly better than ECO-REV-CU for an increasing larger range of values for β. This means that when the delay cost is rather high, it pays off to use a revocable strategy that takes into account the cost of changing decision. In addition, the Friedman test (Nemenyi, 1962) shows that ECO-REV-CA is on average better ranked than ECO-REV-CU in 96% of pairs (α, β). (Further details are available in (Eco-rev, 2021)).

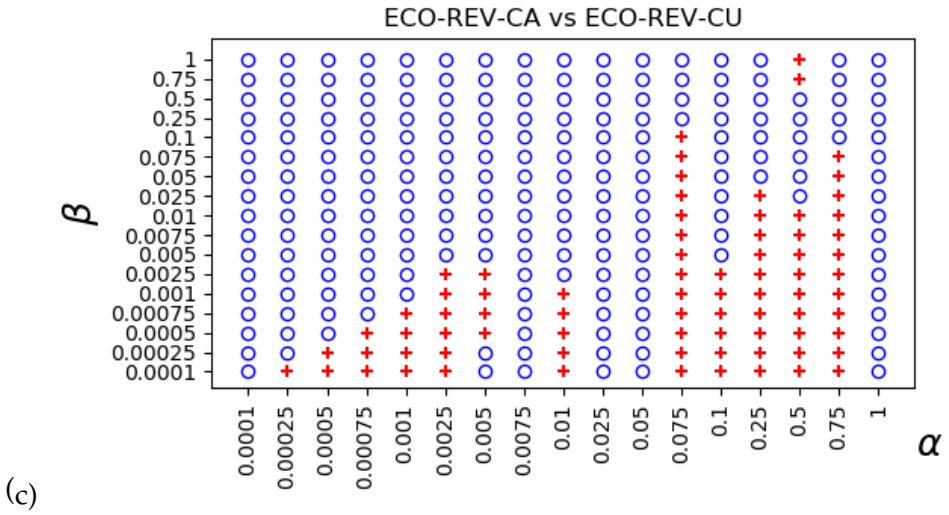
In order to get a global view of the merits of each method, we have drawn *Pareto curves* (see Figure 4.4) with respect to the average Cohen's *kappa* score (Cohen, 1960) and the average *earliness*, which is defined as the mean of the last triggering times normalized by the length of the series $earliness = Avg\{t_\ell / T\}$. These two quantities



(a)



(b)



(c)

FIGURE 4.5: (a) ECO-REV-CU *vs.* ECONOMY-γ; (b) ECO-REV-CA *vs.* ECONOMY-γ; (c) ECO-REV-CA *vs.* ECO-REV-CU. Wilcoxon signed-rank test applied on the *AvgCost* criterion over the 34 test sets, for a range of couples of values α and β, with "$+$" indicating a significant success of the first approach, "$\circ$" an insignificant difference and "$-$" indicating a significant failure of the first approach.

are averaged over the 34 datasets by varying α in the range of values defined in Section 4.4.4, and for $\beta = 0.05$. The Pareto curves confirm that (*i*) the baseline irrevocable ECONOMY-γ method is dominated by the two revocable strategies; (*ii*) and ECO-REV-CA dominates ECO-REV-CU. More finely, it is apparent that, as the slope α of the delay cost increases, from 0.00025 to 1, all methods first maintain a high kappa, before being unable to maintain it as they are forced to make decisions too early. Still, the ECO-REV-CA algorithm is the one that best resists.

Overall, our experiments show the interest of using revocable strategies for the early classification of time series in a wide range of delay and change of decision costs.

4.5 Perspectives and future work

Until now, the problem of early classification of time series was addressed by triggering irrevocable decisions. In other words, once a decision to classify the incoming time series is taken, the process of acquiring new measurements is stopped, and this process ends by outputting the class prediction. For the first time, this chapter defines the revocable regime of this problem, which is a more generic problem, where new measurements are still acquired even if a prediction is triggered, and multiple decisions are allowed to be taken. The notion of *second order* non-myopia has been introduced as well as its corresponding optimization problem. Two versions of an algorithm have been implemented, one which takes into account the cost of changing the decision and the second which does not. Extensive experiments have shown that the algorithm which explicitly takes into account the cost of changing decisions, significantly overcomes the algorithm that does not. In addition, both proposed algorithms outperform the irrevocable scheme. The potential impact of these results on applications such as predictive maintenance, intensive care units or autonomous vehicles, to name a few, is noteworthy.

For future work, more limitations of the ECTS problem should be explored, for example, extending the revocable setting to open time series with indefinite length and different labels for each portion instead of a single label for the whole time series. In the work presented in this chapter, decisions are supposed not to affect the true label, and another scenario is to consider *early outcome classification* where decisions taken can affect the final outcome (i.e. true label), namely in medical applications, if we try to predict the patient death, actions taken by the doctor could change the final result.

In the next chapter, new hypotheses of the classical problem of early classification will be relaxed. The first limitation is that labels are associated with the full-length time series. Second, time series are of finite length. This new setting is challenging since it changes the problem drastically. A novel algorithm to solve this problem while optimizing the earliness-accuracy trade-off will be presented.

Chapter 5

Early classification in open time series

Abstract

In numerous applications, for instance in predictive maintenance, there is a pression to predict events ahead of time with as much accuracy as possible while not delaying the decision unduly. This translates in the optimization of a trade-off between earliness and accuracy of the decisions, that has been the subject of research for time series of finite length and with a unique label. And this has led to powerful algorithms for Early Classification of Time Series (ECTS). This chapter, for the first time, investigates such a trade-off when events of different classes occur in a streaming fashion, with no predefined end. In the *Early Classification in Open Time Series* problem (ECOTS), the task is to predict events, i.e. their class and time interval, at the moment that optimizes the accuracy vs. earliness trade-off. Interestingly, we find that ECTS algorithms can be sensibly adapted in a principled way to this new problem. We illustrate our methodology by transforming two state-of-the-art ECTS algorithms for the ECOTS scenario. Among the wide variety of applications that this new approach opens up, we develop here a predictive maintenance use case that optimizes alarm triggering times, thus demonstrating the power of this new approach.

The content of this chapter has been submitted to a conference chapter:

- When to Classify Events in Open Time Series. **Y Achenchabe**, A Bondu, A Cornuéjols, V Lemaire - The 14th Asian Conference on Machine Learning **ACML**, 2022.

5.1 Introduction

In intensive care units (Shekhar et al., 2021), in control rooms of electrical power grids (Dachraoui, Bondu, and Cornuejols, 2013), in government councils assessing emergencies, in many kinds of contexts, it is essential to make timely decisions in absence of complete knowledge of the true outcome. The issue facing the decision-makers is that, usually, the longer the decision is delayed, the clearer is the likely outcome (e.g. whether the patient is critical or not) but, also, the higher the cost that will be incurred if only because earlier decisions allow one to be better prepared. Formally, this problem translates into optimizing online the trade-off between the earliness and the accuracy of the decision. Early Classification of Time Series (ECTS) deals with time series of finite length and a single decision per time series.

In the previous chapter, we have dealt with the first limitation of the ECTS problem, which consists of the irrevocability of decisions. We have proposed a novel algorithm that considers the cost of revision or what we call the cost of changing a decision. Experiments showed that it is better to be able to revise decisions than to implement an irrevocable decision strategy. In addition, it is worth considering the non-myopic of *second order* approach.

While the ECTS framework is well suited to some real-world problems, it suffers from some limitations that will be tackled in this chapter:

1. Labels are associated with the full-length time series: it means that we know that all measurements acquired belong to the same class label, and the start and the end are clearly defined.

2. Time series are of finite length: decisions should be made before a specific deadline which is the maximal length of the time series.

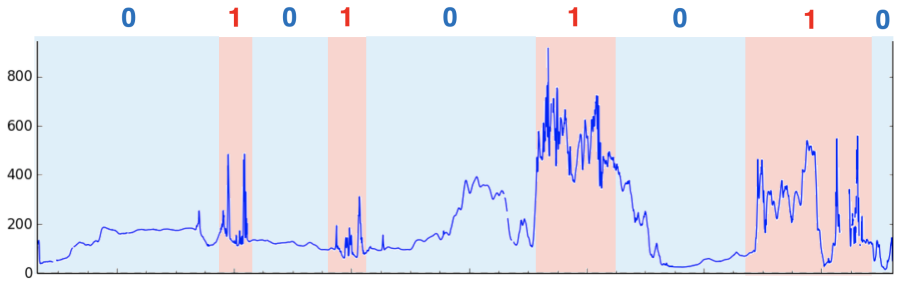


FIGURE 5.1: Example of a part of an open time series where events of possibly different lengths are here labeled with '0s' and '1s'.

Applications abound where the measurements come in an open time series with no time bounds and where different events arise, possibly of different lengths, each with its own class (see Figure 5.1). In this setting, the Early Classification of these events in an Open Time Series (ECOTS) raises three issues if one wants to adapt the ECTS approach.

1. What should the *classifiers* in ECOTS do? In the ECTS framework, they take incomplete time series $\mathbf{x}_t$ as input and make prediction about the class of the associated complete, but still unknown, time series $\mathbf{x}_T$. But, in ECOTS, the notion of complete time series does not make sense anymore.

2. How to solve the *earliness-accuracy* trade-off in the ECOTS setting? In the ECTS framework, the start and length of time series are known which allows one to measure the earliness. But how to define it in ECOTS?

3. How to build the *training sets* in the framework of ECOTS where there are no more individual time series with their associated class, but open time series with events of different classes and durations?

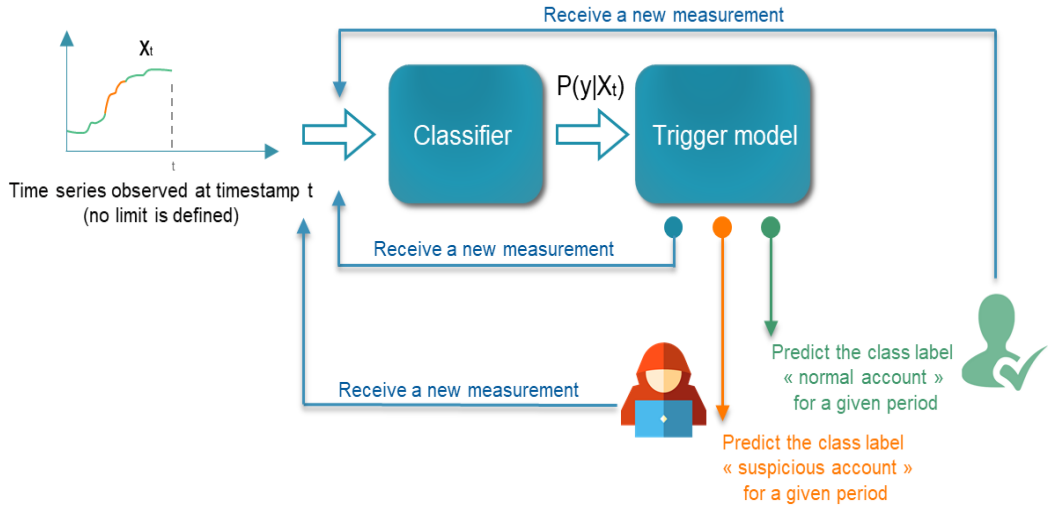


FIGURE 5.2: General schema of ECOTS approaches

The ECOTS procedure is illustrated in Figure 5.2. For an indefinite amount of time, new measurements are received at each timestamp, and the incoming time series consists of multiple chunks having different class labels. At testing phase, the user has not the knowledge about when a specific portion with different class starts and when it ends, contrarily to the classical problem of ECTS where the user knows that the incoming time series ends at a certain moment as well as only one class label is associated with the complete time series.

The goal of this chapter is, first, to define properly the ECOTS problem and, second, to present a methodology to adapt any ECTS approach to it by answering the three questions raised above. As a result, we show i) how the role of the classifiers must be thought anew and transformed; ii) how the earliness-accuracy trade-off translates to the new scenario and what the decision triggering system becomes. We illustrate our methodology by transforming two state-of-the-art ECTS algorithms for the ECOTS scenario. Among the wide variety of applications that this new approach opens up, we develop here a predictive maintenance use case that optimizes alarm triggering times, thus demonstrating the power of this new approach.

In order to avoid confusion, it is important to note that the data stream literature (Silva et al., 2013) focuses on classification of incoming data points at fixed horizon under memory constraints and evolving properties of data. Whereas, in this chapter, we focus on identifying the optimal moment of the classification, the one that optimizes the *earliness-accuracy* trade-off in a stationary environment.

The chapter is organized as follows. Section 5.2 formally draws a parallel between the ECTS problem and the ECOTS one, leading to a generic approach capable of transposing any ECTS algorithm into an ECOTS one. We then show, in Section

5.3, how to adapt these methods to the ECOTS problem, before comparing their performances on experiments in Section 5.4. The conclusion, in Section 5.5, underlines what has been performed in this work, and provides directions for future research.

5.2 ECOTS in the perspective of ECTS

This section defines the ECTS and ECOTS problems in turn, and presents our proposed methodology to adapt any ECTS approach to solve the ECOTS problem.

5.2.1 The ECTS problem

In the ECTS setting, each classifier h_t ($1 \leq t \leq T$) is learned from the truncated training time series up to time t: $\{(\mathbf{x}_t^j, y_j)\}_{(1 \leq j \leq m)}$ (see Figure 5.3). It is expected that the accuracy of the classifiers grows as t increases from the first time step $t = 1$ to the last one $t = T$.

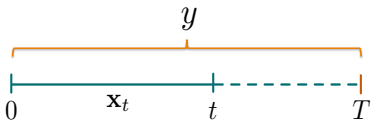


FIGURE 5.3: In the ECTS setting, the classifier h_t sees the incoming time series $\mathbf{x}_t$ and predicts a label $\hat{y}$ of the complete time series $\mathbf{x}_T$ of true class y.

The problem, given an incoming time series, is to choose a time t for which the expected cost of misclassification $\mathrm{C}_m(\hat{y}|y)$, where $\hat{y} = h_t(\mathbf{x}_t)$ and y is the true class, plus the delay cost $\mathrm{C}_d(t)$, is minimal. Formally, the combined expected cost is given by:

$$\begin{aligned} f(\mathbf{x}_t) &= \mathbb{E}^{\,t}_{(\hat{y},y)\in\mathcal{Y}^2}[\mathrm{C}_m(\hat{y}|y)|\mathbf{x}_t] + \mathrm{C}_d(t) \\ &= \sum_{y\in\mathcal{Y}} P_t(y|\mathbf{x}_t) \sum_{\hat{y}\in\mathcal{Y}} P_t(\hat{y}|y,\mathbf{x}_t)\,\mathrm{C}_m(\hat{y}|y) + \mathrm{C}_d(t) \end{aligned} \tag{5.1}$$

where $\mathbb{E}^{\,t}{}_{(\hat{y},y)\in\mathcal{Y}^2}[\,]$ is the expectancy at time t, over the variables y and $\hat{y}$. $P_t(y|\mathbf{x}_t)$ is the probability of the class y given an incomplete time series $\mathbf{x}_t$, and $P_t(\hat{y}|y,\mathbf{x}_t)$ is the probability that the classifier h_t makes the prediction $\hat{y}$ given $\mathbf{x}_t$ as input while y would be its true label.

The objective of the trigger function is to identify the best time $t^\star$ for triggering the decision while receiving online the measurements of the time series, the one that minimizes Equation 5.1.

5.2.2 The ECOTS problem

In this chapter, contiguous instants with the same label are called chunks or events (see Figure 5.1). In the ECOTS scenario, we suppose that we have a training data set of m labeled chunks $\{(\mathbf{x}^j, y_j)\}_{(1 \leq j \leq m)}$ coming from an open time series, where each $\mathbf{x}^j$ has length $l_{\mathbf{x}^j}$, and where y_j is the corresponding label. The ECOTS problem consists in predicting as soon as possible these events, i.e. their class label and time limits.

5.2.3 Proposed transposition of ECTS approaches into ECOTS ones

We propose to simplify the ECOTS problem by considering point-wise predictions instead of predicting whole chunks (i.e. class and time limits), that is to make independent predictions of the labels associated with each single timestamp t_p in the open time series. From such predictions, chunks could be reconstituted, for instance by gluing time stamps with the same predicted class.

We posit that the unfolding time series is observed over a finite time interval which depends on the present time t. In the following, we will use the term time window and note it $\mathbf{x}_{(t-w,t)}$ at time t when its size is w (w past measurements are available at time t).

Then, the target time stamp t_p can be in the future ($t_p > t$), for example if there are warning signs that a machine will break down, or in the past ($t_p < t$) if it was necessary to wait until t_p was in the observation time window to be able to identify its class, as it can happen during a computer attack for example. Therefore, at any time t, for a target timestamp t_p, the transposed ECOTS problem is to decide whether t is the best time for the prediction $\hat{y}_{t_p}$, the class associated with t_p, or whether to postpone this decision to the next time step $t+1$, which will bring a new measurement.

It is expected that, as the window of observation $\mathbf{x}_{(t-w,t)}$ comes closer to t_p, with increasing t, it is easier to make a reliable prediction about its class, but, at the same time, the cost of delaying prediction increases. (See Figure 5.4). Note that we assume that there is a maximal value η_M for the horizon, above which no precursor signal can be detected, and a minimal one η_m after which it is no longer useful to detect the event.

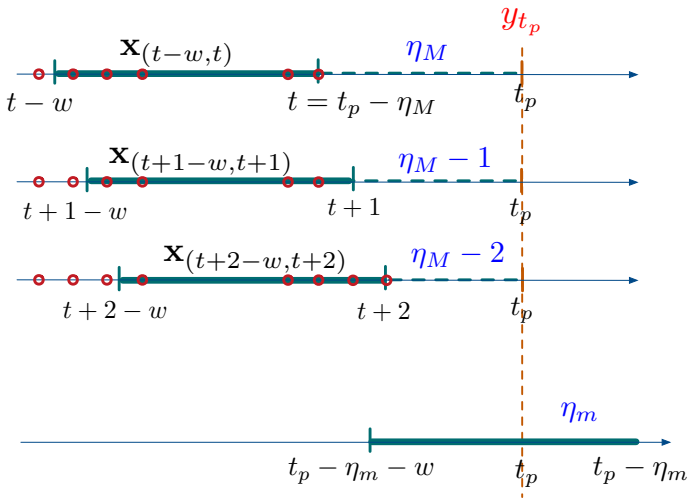


FIGURE 5.4: The fixed point in time where to make a prediction is t_p, of true label y_{t_p}. As the measurements become available from $t_p - \eta_M$ to $t_p - \eta_m$, different classifiers h_η come into play with an advancing sliding window and a diminishing horizon. The triggering system selects the best time to make a prediction $\hat{y}_{t_p}$.

We now turn to the three questions raised in the introduction.

1- In the classical ECTS problem (see Section 5.2.1 and Figure 5.3), each classifier h_t observes a time series $\mathbf{x}_t$ that is increasingly large as t approaches the time limit T. In the ECOTS setting, as we propose to see it, by contrast, each classifier h_η observes a sliding window $\mathbf{x}_{(t-w,t)}$ of the same number w of observations. And each one makes a prediction about the class of the time t_p positioned at a given horizon η

(positive or negative) from t (see Figure 5.4). Given a maximal horizon $\eta_M > 0$, the first classifier that can make a prediction about a time t_p is h_{η_M} viewing the window $\mathbf{x}_{(t_p-\eta_M-w,t_p-\eta_M)}$. And the last classifier is h_{η_m} viewing the window $\mathbf{x}_{(t_p-\eta_m-w,t_p-\eta_m)}$. Thus, instead of having a set of classifiers h_t ($1 \leq t \leq T$) as in the ECTS setting, we now have a set of classifiers h_η ($\eta_m \leq \eta \leq \eta_M$) with various horizons. (See Figure 5.4).

2- The second question to solve is to adapt the earliness-accuracy trade-off to the ECOTS problem. In the proposed transposition, the gain of information over time is due to a time window $\mathbf{x}_{(t-w,t)}$ that gets closer to the target timestamp t_p, and no longer to additional measurement gathered over time as in ECTS. In addition, the single prediction triggered for each time series in ECTS is replaced by a prediction for each possible target timestamp in the open time series. These two essential but easy to carry out modifications allow to transpose most of the ECTS approaches, since the optimization criterion they use for their triggering strategy can remain essentially the same. As a proof of concept, in Section 5.3, we show how two state of the art methods for ECTS can be adapted to the ECOTS problem.

3- In the proposed approach, the values of the window size w, and of the bounds η_m and η_M of the horizon have to be chosen from a training set. But which training set?

In the ECOTS problem, it is assumed that the relationship between the symptoms of an event and the event itself are *stationary* over time (e.g. a given malfunction of a machine keeps the same telltale signs and the same characteristics whenever its appearance in time. Or the symptoms associated with a patient who should undergo an urgent heart operation stay the same, fortunately for the doctors, and for their training). From this property, in the same way as doctors can be trained using independent episodes in an hospital history about heart attacks, it is possible to use subsequences of the open time series, as long as they are independent, to build training datasets in order to learn the classifiers h_η (see Section 5.4.1).

5.3 Adapting two state-of-the art ECTS approaches

In Section 5.2.3, we have shown how to translate an ECTS problem into an ECOTS one by modifying the definition and purpose of *the classifiers*. In the following, we demonstrate how the *triggering strategy* used in ECTS can be adapted to deal with ECOTS. For this, we consider one of the best performing myopic strategies known to date, described in (Mori, Alexander Mendiburu, Dasgupta, et al., 2017) and the best non-myopic approach in the literature: the ECONOMY-γ strategy described in Chapter 3. The first one relies ultimately on confidence criteria, while the second one explicitly optimizes the accuracy versus delay cost trade-off.

5.3.1 The SR approach

The SR approach (Mori, Alexander Mendiburu, Dasgupta, et al., 2017) uses a trigger-model which involves 3 parameters $(\gamma_1, \gamma_2, \gamma_3)$ in order to decide if the current prediction is reliable (output 1) or if it is preferable to wait for more data (output 0):

$$Trigger^{\gamma}\left(h_\eta(\mathbf{x}_t)\right) = \begin{cases} 0 & \text{if } \gamma_1\, p_1 + \gamma_2\, p_2 + \gamma_3\, \frac{t}{T} \leq 0 \\ 1 & \text{otherwise} \end{cases} \tag{5.2}$$

where p_1 is the largest posterior probability estimated by the classifier h_η, p_2 is the difference between the two largest posterior probabilities, and the last term $\frac{t}{T}$

represents the proportion of the incoming time series that is visible at time t. The parameters $\gamma_1, \gamma_2, \gamma_3$ are real values in $[-1, 1]$ to be optimized using training data.

Algorithm 6 Adapted SR approach in the ECOTS scenario

Input: t: current moment.
t_p: target that belongs to [t+η_m, t+η_M].
w, η_m, η_M: window size, minimum and maximum horizon.
1: **for all** $\eta = t_p - t, \ldots, \eta_m$ (step=-1) **do**
2: sliding_window = $\mathbf{x}_{(t_p-\eta-w, t_p-\eta)}$ # sliding window: adds new measurement and deletes the first one.
3: compute A = $\gamma_1\, p_1 + \gamma_2\, p_2 + \gamma_3\, \frac{\eta_M - \eta}{\eta_M - \eta_m}$ using the updated sliding window.
4: **if** $A > 0$ or $\eta == \eta_m$ **then**
return η
5: **end if**
6: **end for**

In the ECOTS problem, the trigger function for a target at horizon $\eta = t_p - t$ becomes:

$$Trigger^{\gamma}\left(h_\eta(\mathbf{x}_{(t-w,t)})\right) = \begin{cases} 0 & \text{if } \gamma_1\, p_1 + \gamma_2\, p_2 + \gamma_3\, \frac{\eta_M - \eta}{\eta_M - \eta_m} \leq 0 \\ 1 & \text{otherwise} \end{cases}$$

The last term of Equation (5.2) is replaced by $\frac{\eta_M - \eta}{\eta_M - \eta_m}$, which represents the relative position of the current horizon η in the considered range of horizons $[\eta_m, \eta_M]$. Algorithm 6 highlights the adapted procedure of the SR approach to choose the optimal horizon for a given target t_p.

5.3.2 The ECONOMY-γ approach

ECONOMY-γ is a non-myopic cost-based approach (Chapter 3), which is capable of estimating the expected cost of making a prediction for any time $t + \tau$ ($1 \leq \tau \leq T - t$) in the future, defined as:

$$\begin{aligned} f_\tau(\mathbf{x}_t) &= \mathbb{E}^{t+\tau}_{(\hat{y},y)\in\mathcal{Y}^2}[\mathrm{C}_m(\hat{y}|y)|\mathbf{x}_t] \;+\; \mathrm{C}_d(t+\tau) \\ &= \sum_{y\in\mathcal{Y}} P_{t+\tau}(y|\mathbf{x}_t) \sum_{\hat{y}\in\mathcal{Y}} P_{t+\tau}(\hat{y}|y,\mathbf{x}_t)\, \mathrm{C}_m(\hat{y}|y) \;+\; \mathrm{C}_d(t+\tau) \end{aligned} \tag{5.3}$$

In practice, the terms $P_{t+\tau}(\hat{y}|y, \mathbf{x}_t)$ and $P_{t+\tau}(y|\mathbf{x}_t)$ are not tractable. A partitioning of the training data into K groups $\mathfrak{g}_k \in \mathcal{G}$ (see Chapter 3) is required to make them computable, yielding the following approximation:

$$f_\tau(\mathbf{x}_t) \approx \sum_{\mathfrak{g}_k\in\mathcal{G}} P_{t+\tau}(\mathfrak{g}_k|\mathbf{x}_t) \sum_{y\in\mathcal{Y}} P_{t+\tau}(y|\mathfrak{g}_k) \sum_{\hat{y}\in\mathcal{Y}} P_{t+\tau}(\hat{y}|y,\mathfrak{g}_k)\, \mathrm{C}_m(\hat{y}|y) \;+\; \mathrm{C}_d(t+\tau)$$

The optimal decision time, at time t, is thus estimated to be:

$$\tau^\star \;=\; \underset{\tau\in\{0,\ldots,T-t\}}{\mathrm{ArgMin}}\; f_\tau(\mathbf{x}_t) \tag{5.4}$$

The idea is to estimate the cost of a decision for all future time steps, up until $t = T$, based on the current knowledge about the incoming time series $\mathbf{x}_t$. The decision is postponed unless $\tau^* = 0$, that is when it is expected that there will be no better trade-off in the future. If so, the prediction $h_t(\mathbf{x}_t)$ is returned and the classification process is terminated. Otherwise, the decision is postponed to the next time step, and

Eq. (5.4) is computed again, this time with $\mathbf{x}_{t+1}$ as input. The process goes on until a decision is made or $t = T$ at which point a prediction is forced.

Algorithm 7 Adapted ECONOMY approach in the ECOTS scenario

Input: t: current moment.
t_p: target that belongs to [t+η_m, t+η_M].
w, η_m, η_M: window size, minimum and maximum horizon.
1: **for all** $\eta = t_p - t, \ldots, \eta_m$ **do**
2: compute $f_\eta(\mathbf{x}_{(t-w+1,t)})$
3: **end for**
4: $\eta^\star = \text{ArgMin}_{\eta_m \leq \eta \leq t_p - t} f_\eta(\mathbf{x}_{(t-w+1,t)})$
5: **if** $\eta^\star$ == t_p-t or $t_p - t == \eta_m$ **then**
6: return $\eta^\star$
7: **end if**

In ECOTS, as time t increases, the task is to label each time step t_p as it appears in the span of the horizon: $[t + \eta_m, t + \eta_M]$ (see Figure 5.4). While in Equation (5.3), the term $\mathbb{E}^{t+\tau}_{(\hat{y},y)\in\mathcal{Y}^2}[\mathrm{C}_m(\hat{y}|y)|\mathbf{x}_t]$ involves the calculation of the confusion matrices for future time steps $t + \tau$ knowing the current incoming time series $\mathbf{x}_t$, the adaptation to ECOTS requires that the confusion matrices are now computed for the various horizons from η_m to η_M and then used to estimate the cost of decision for each horizon η:

$$f_\eta(\mathbf{x}_{(t-w,t)}) = \mathbb{E}^{\eta}_{(\hat{y},y)\in\mathcal{Y}^2}[\mathrm{C}_m(\hat{y}|y)|\mathbf{x}_{(t-w,t)}] + \mathrm{C}_d(\eta)$$

and the best horizon:

$$\eta^\star = \underset{\eta_m \leq \eta \leq t_p - t}{\text{ArgMin}} f_\eta(\mathbf{x}_{(t-w,t)})$$

The decision to classify the data point t_p is triggered at time t either because $t + \eta^\star = t_p$ (i.e. corresponding to the optimal cost) or when $t_p = t + \eta_m$ (i.e. it is not possible to wait any longer), see Algorithm 7 for more details.

The cost of delay $\mathrm{C}_d(t)$, which is an *increasing* function of t in ECTS is a *decreasing* function $\mathrm{C}_d(\eta)$ of the horizon η in ECOTS. Indeed, as the target that we want to label approaches (η decreasing), the cost of the decision increases.

Note that the time and space complexities of the ECONOMY-γ approach adapted to ECOTS are the same than in the original approach. In the ECTS setting, at testing phase, computing the cost at each timestep is in $\mathrm{O}(T^2)$ at worst case, and in ECOTS setting is in $O((\eta_M - \eta_m)^2)$.

5.4 Experiments

We have proposed a principled method to adapt any ECTS approach into an ECOTS one (see Sections 5.2 and 5.3). The aim of the experiments is to validate that the adaptation of the SR and ECONOMY-γ approaches is efficient in the ECOTS setting. In addition, we illustrate the applicability of the proposed approaches for predictive maintenance using real data from the industrial domain.

This section aims at answering the following questions:

1. How efficient is the proposed framework for adapting any ECTS approach to the ECOTS problem compared to baseline algorithms designed for the ECOTS problem?
2. How do these approaches behave when the delay cost increases, and when the misclassification cost becomes very imbalanced?
3. How these approaches adapt their decision time to the observed data?

Our *source code* is shared for full reproducibility of the experiments Github. This also allows interested researchers to extend the experiments to other open time series datasets.

5.4.1 Experimental protocol

Data description:

We use an open real dataset (data, 2020) from one of the Schwan's factories. It contains 100 multivariate time series corresponding to 100 machines monitored over time for a period of 1 year (January 2015 to January 2016) with measurements collected every hour. Each time series is a multi-dimensional data table whose rows indicate the temporal domain and columns include telemetry features (pressure, rotation, voltage and vibration), 5 Boolean columns encoding different types of device errors which are premises correlated with a future failure and a last column which indicates the presence or absence of a failure (the variable to be predicted). This makes 8761 rows and 10 columns for each machine. The whole dataset contains 3919 errors and 761 failures for a total number 876,100 timestamps. This dataset is extremely imbalanced with 0.08% of timestamps associated with the abnormal class (i.e. failure). There is on average 7 failures per time series, with a minimum of 0 and a maximum of 19 failures per machine during the observed year.

Problem statement:

Traditionally, the problem of predictive maintenance is solved by fixing a horizon for predictions (e.g. if a technician needs at least 12 hours to take preventive actions before the machine fails actually, then a fixed horizon would be chosen as η =12 hours). Our goal is to use the ECOTS approaches to make this horizon *adaptive* to the observable part of the time series at hand.

Evaluation criterion:

Ultimately, the value of using an early classification method is defined by the *average cost* incurred using it, as in Chapter 3. Given an open time series $\mathcal{S}$ (e.g. a machine monitored over a year), observed on a finite time interval of sufficient length N. This time period is composed of time stamps $t \in [1, N]$, labeled by the class y_t. As time increases from 1 to N, the ECOTS system makes predictions for each time step t: $\hat{y}_t$ while the true class is y_t. In addition, for each $t \in [1, N]$, the prediction is made using a classifier $h_{\eta_{t^\star}}$ corresponding to the triggering horizon $\eta_{t^\star}$, thus incurring a delay cost $C_d(\eta_{t^\star})$. Hence, the formula:

$$AvgCost(\mathcal{S}) = \frac{1}{N}\sum_{t=1}^{N}\big(C_m(\hat{y}_t|y_t) + C_d(\eta_{t^\star})\big) \tag{5.5}$$

Computing the costs in the experiments:

In real applications, the decision costs would be provided by domain experts. In order to study the behavior of the different ECOTS algorithms, a large range of values has been considered for the misclassification and the delay costs.

The cost of misclassification: Since we deal with a predictive maintenance problem, we make the assumption that the cost of missing a failure is much higher than the cost of sending the technical team. We thus consider four different misclassification costs $C_m = \begin{bmatrix} TN & FN \\ FP & TP \end{bmatrix}$, by varying the importance of false negatives:

$$C_m^{(1)} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}, C_m^{(2)} = \begin{bmatrix} 0 & 10 \\ 1 & 0 \end{bmatrix}, C_m^{(3)} = \begin{bmatrix} 0 & 100 \\ 1 & 0 \end{bmatrix}, C_m^{(4)} = \begin{bmatrix} 0 & 1000 \\ 1 & 0 \end{bmatrix}.$$

The cost of delaying decision: The delay cost $C_d(\eta)$ is provided by the domain experts for an actual use case, and could be of any form. In our experiments, we set it as a linear function of horizon, with coefficient, or slope, α: $C_d(\eta) = \alpha \times \frac{\eta_M - \eta}{\eta_M - \eta_m}$. The larger α, the higher the cost of postponing the decision and the greater the incentive to make prediction for large horizons η. When α is very high, the gain in misclassification cost by waiting to be closer to the target cannot compensate for the increase of the delay cost, and it is better to make a decision early on, that is for large horizons, close to η_M. If, on the contrary, α is very low compared to the misclassification cost, it does not hurt to wait until the target t_p is close to the sliding window $\mathbf{x}_{(t_p-\eta-w,t_p-\eta)}$. Our experiments were run over a large range of values of $\alpha \in$[0.001, 0.01, 0.1, 1, 10, 100, 1000].

Training the collection of classifiers and ECOTS algorithms

Data split and extraction: We splitted the set of time series into four parts: 50% for training the classifiers, 20% for testing the ECOTS algorithms, 15% for validating the ECOTS algorithms and 15% for estimating the confusion matrices. This split is the same as in Chapter 3. Subsequences of size w were extracted from the training open time series by doing the following steps: (*i*) time stamps t_p, aka targets, were set within the time series, spaced with $w + \eta_M$ time units in order to avoid overlaps between samples; (*ii*) $\eta_M - \eta_m$ subsequences of size w were extracted around each target, each one dedicated to the training of the classifier h_η (see Figure 5.4).

Choice of the parameters w, η_m, η_M: These parameters depend on the problem that is being solved and the data associated with it. One of the key ingredients of early classification methods is the information gain measured by the AUC. Generally, the expected cost of misclassification decreases as the target being classified gets closer to the sliding window. A window size of $w = 10$ has been chosen to study the behavior of the ECOTS problems, since it shows a significant information gain over various horizons using AUC. We refer the reader to the supplementary material for AUC curves as a function of horizon with different sliding window sizes. They exhibit equivalent information gain curves, which means that this dataset is not very sensitive to the choice of w. The parameter η_M can be chosen according to the AUC, for our experiments we have chosen $\eta_M = 50$ as the AUC reaches 0.5 which corresponds to the random model, while, for η_m, we chose the end of the sliding window: $\eta_m = -w$.

Training the collection of classifiers: As mentioned in Section 5.2.2, a set of classifiers h_η for different horizons η such that $(\eta_m \leq \eta \leq \eta_M)$ has to be trained. Extracted features[1] from sliding windows include simple statistics: *min*, *max*, *mean*, *median* and the *count* of each type of errors. For our experiments, we trained XGboost models by fine tuning parameters within the following grid of values[2]: *min chlid weight* $\in [1, 5, 10]$, *gamma* $\in [0.5, 1, 1.5, 2, 5]$, *subsample* $\in [0.6, 0.8, 1]$, *colsample* by tree $\in [0.6, 0.8, 1]$, *max depth* $\in [3, 4, 5, 10]$. The parameter *scalePosWeight* is set to the the ratio of positive examples over negative ones in order to take into account the fact that the dataset is imbalanced. The combination of parameters that minimize the total cost[3] is chosen on a validation set (20% of the set used for training the classifiers), then the model is learned on the whole training set. For a fair comparison, the same collection of classifiers is used for all ECOTS algorithms.

ECOTS algorithms: The competing approaches considered in this paper are described below as well as their optimized hyper-parameters.

- **Late baseline**: the last classifier in the collection $\tilde{h}_{\eta_m}$ is used. This is the last time that a prediction can be made.
- **Early baseline**: the first classifier in the collection $\tilde{h}_{\eta_M}$ is used. This corresponds to the earliest possible prediction with the largest horizon in the future.
- **Confidence-based Classifiers (CC)**: for a fixed target t_p, this method takes a decision as soon as the confidence of the classifier regarding the class of interest exceeds a given threshold, optimized as a meta-parameter for each value of α using validation set. If this never happens, then $t_p = t + \eta_m$ and the prediction is forced using $\tilde{h}_{\eta_m}$.
- **Economy-γ** (see Section 5.3.2): for each value of α, the number of groups K used in the method is optimized in the range [1,5] using a validation set.
- **SR** (see Section 5.3.1): for each value of α, the parameters γ_1, γ_2 and γ_3 were optimized in the range $[-1, -0.5, 0, 0.5, 1]^3$ using a validation set.

Note that the "late" and the "early" baselines are not adaptive, while the "Confidence-based" method adapts its decisions to the current input. One goal of the experiments is to compare these methods with ones that have been *translated* from the ECTS framework: **Economy-γ** and **SR**.

5.4.2 Results and analysis

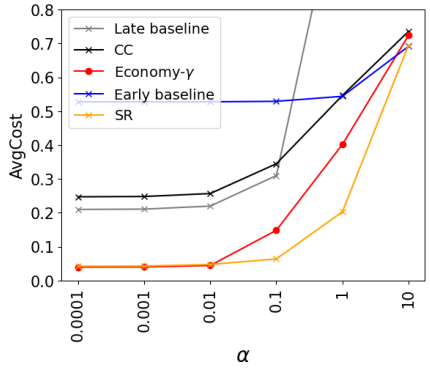
In this section, detailed answers to the questions raised in the introduction of Section 5.4 are given, supported by numerical results.

Efficiency of the proposed framework: In Figure 5.5, one can note interesting patterns for the four matrices of misclassication costs and the large range of values of α and therefore of delay cost functions. When the cost of delaying decision is high ($\alpha \geq 10$), the optimal strategy is to make predictions immediately (i.e. the "early" baseline), for the largest value of the horizon η_M. When the delay cost is low, ($\alpha \leq 0.01$), taking late

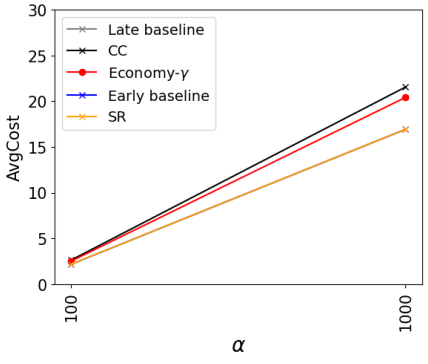
[1] Reproducible using our source code available on Github.

[2] The interested reader can refer to the official documentation for more details: `https://xgboost.readthedocs.io/en/stable/`

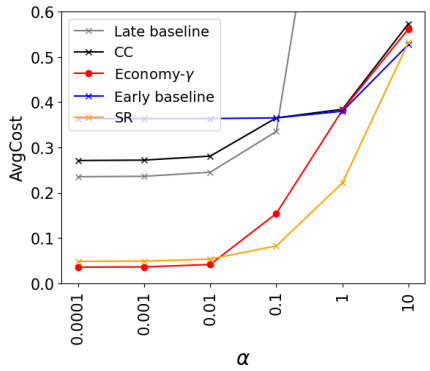
[3] Given the cost of false positives and false negatives, the total cost is computed on a validation set as the sum over wrongly predicted samples weighted by the corresponding cost.



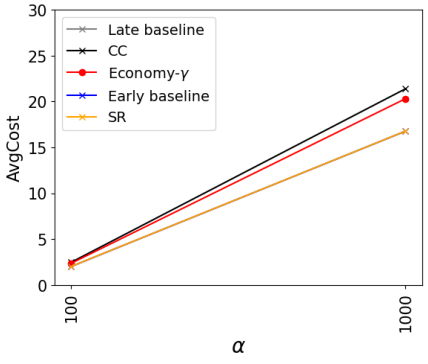
(A) $C_m^{(1)} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$



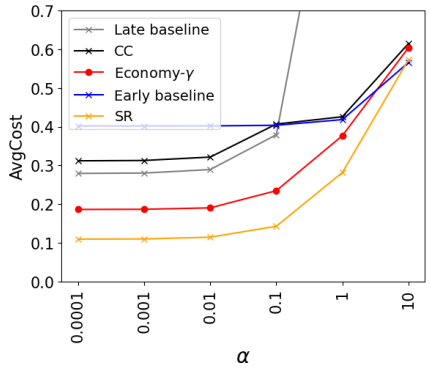
(B) $C_m^{(1)} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$



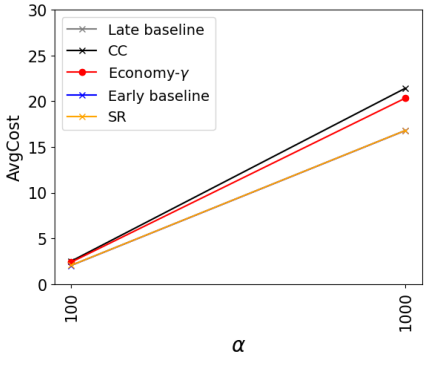
(C) $C_m^{(2)} = \begin{bmatrix} 0 & 10 \\ 1 & 0 \end{bmatrix}$



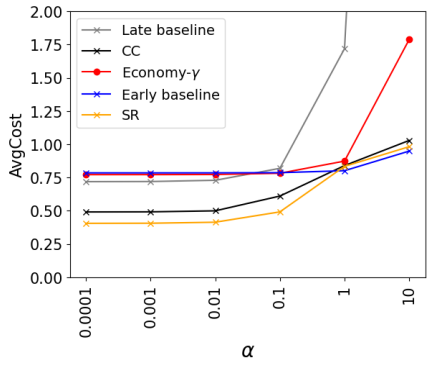
(D) $C_m^{(2)} = \begin{bmatrix} 0 & 10 \\ 1 & 0 \end{bmatrix}$



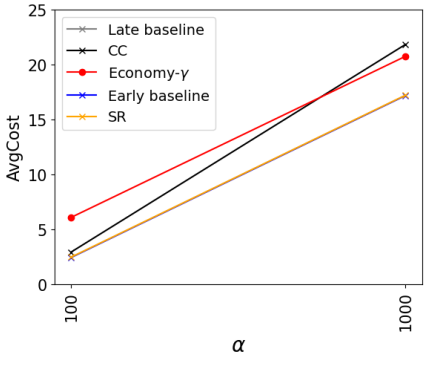
(E) $C_m^{(3)} = \begin{bmatrix} 0 & 100 \\ 1 & 0 \end{bmatrix}$



(F) $C_m^{(3)} = \begin{bmatrix} 0 & 100 \\ 1 & 0 \end{bmatrix}$



(G) $C_m^{(4)} = \begin{bmatrix} 0 & 1000 \\ 1 & 0 \end{bmatrix}$



(H) $C_m^{(4)} = \begin{bmatrix} 0 & 1000 \\ 1 & 0 \end{bmatrix}$

FIGURE 5.5: *AvgCost* of ECOTS algorithms computed on the test set for different values of the α parameter of the delay cost (x-axis), and for different values of misclassification cost.

decisions is a good strategy even though it is not optimal (i.e. the "late" baseline). It is apparent that the CC method essentially switches from one baseline strategy to the other one as α increases, and therefore seems to realize the best of the two strategies adaptively.

At the same time, both methods "imported" from ECTS: **Economy-γ** and **SR**, noticeably overcome **CC**. They are able to better control the horizon of decision when α is low, thus achieving significantly better performance (see more on this in the discussion below on the ability to adapt the triggering times), and they perform as well as the competitors for high values of α. In particular, this shows that the often preferred, almost by default, confidence-based methods (e.g. CC) are being overtaken by more formally based methods translated from ECTS.

The experimentation on this dataset taken from a predictive maintenance problem, leads to the conclusion that the adapted **Economy-γ** and **SR** methods seem to be specially interesting under a wide range of conditions.

Effect of the delay and misclassification costs: In all the situations corresponding to the subfigures of Figure 5.5, the average cost sharply increases when the delay cost become very high (note the logarithmic scale on the x-axis). Indeed, decisions have to be made early so as to avoid high delay costs, but this is at the price of false positives and negatives which may incur high cost, specially for the $C_m^{(3}$ and $C_m^{(4)}$ cost matrices.

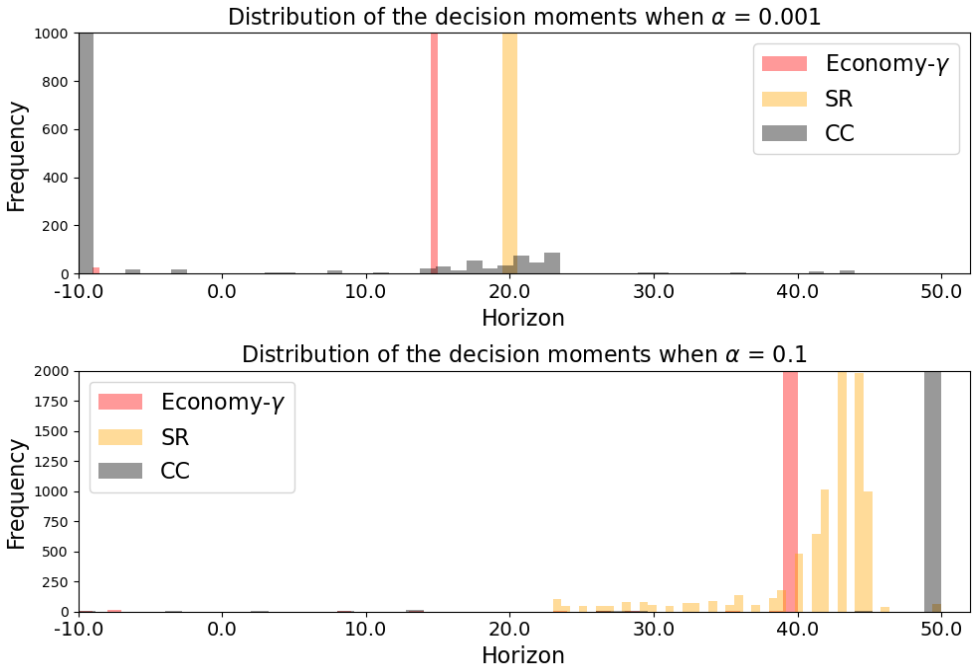


FIGURE 5.6: Distribution of the decision moments for ECONOMY-γ, SR and CC algorithms, for $\alpha = 0.001$ and $\alpha = 0.1$ both for $C_m^{(2)} = \begin{bmatrix} 0 & 10 \\ 1 & 0 \end{bmatrix}$.

Ability to adapt the triggering moment according to observed data: In order to better understand the properties of the adaptive approaches, we show in Figure 5.6 the distribution of the decision moments of the three methods: ECONOMY-γ, CC and SR. We have chosen the scenario $C_m = C_m^{(2)}$ (The method behaves similarly for other values of $C_m = C_m^{(.)}$ and additional figures are given in the supplementary material). One immediate finding is that both ECONOMY-γ and **SR** are more ready to consider

intermediate horizons of prediction than **CC**. For $\alpha = 0.1$, **SR** is more prone to spread its decision horizons than ECONOMY-γ, which shows how this efficient approach adapts the horizon. It may explain its superior performance in this instance. More Figures with different values of α and the cost of misclassification are available in the Appendix D.

5.5 Perspectives and future work

In this chapter, we formally defined for the first time in the literature the problem of *early classification in open time series* (ECOTS). We have provided a generic and well-grounded methodology that allows the adaptation of *early classification of time series* (ECTS) approaches to the ECOTS problem. This recipe specifies how to translate the earliness-accuracy trade-off, how to modify the decision triggering system and what the classifiers become in the new setting.

We have illustrated our methodology by adapting two state-of-the-art ECTS algorithms to the ECOTS scenario and demonstrated the value of the resulting algorithms by applying them to a real-world dataset related to predictive maintenance. Experiments attest to the ability of the new ECOTS algorithms to seek the best trade-off between the earliness and the accuracy of the decisions when events unfold over open time series.

Our work paves the way for a wealth of applications where measurements are made over long periods of time, and decisions about upcoming events need to be triggered as early, but also as accurately, as possible. This includes applications in healthcare, predictive maintenance, autonomous driving, and decision aid in agriculture, to name a few.

For future work, it would be interesting for the scientific community to build a reference benchmark composed of an extensive collection of datasets from various application domains. This would allow comparing ECOTS approaches in a meaningful way, using statistical tests to compare competing approaches. Furthermore, The revocable algorithm detailed in Chapter 4 can be used in the ECOTS setting to revoke decisions that have been made. The adaptation seems trivial using the methodology proposed in this chapter. Another line of research concerns the evaluation criterion proposed in this chapter, which is very simplistic. It evaluates each timestamp independently of the other ones. However, a more meaningful way to evaluate ECOTS approaches is to do it by chunk.

In the next chapter, a generic problem that incorporates the future research directions listed above will be defined, and a list of challenges will be presented to the community for further research.

Chapter 6

Challenges and uses cases of ML-EDM

Abstract

We started by studying in-depth the *early classification of time series problem* (ECTS) in Chapter 3. The first limitation of ECTS has been tackled in Chapter 4, which is the irrevocability of decisions. Then in Chapter 5 early classification of time series with indefinite length and having different labels per portion has been addressed. In this chapter, we have taken a step back and thought about a more general problem, called Machine Learning based Early Decision Making (ML-EDM), which consists in optimizing the decision times of models in a wide range of settings where data is collected over time. After defining the ML-EDM problem, ten open problems are identified and proposed to the scientific community to further research in this area. These open problems have important application perspectives, discussed in this chapter.

The content of this chapter has been submitted as journal paper:

- Open challenges for Machine Learning based Early Decision-Making research. A Bondu, Y Achenchabe, A Bifet, F Clérot, A Cornuéjols, J Gama, G Hébrail, V Lemaire, PF Marteau - **SIGKDD** explorations journal, 2022.

6.1 Introduction

In the previous chapters, we started by studying in-depth the *early classification of time series problem* (ECTS) in Chapter 3. The first limitation of ECTS has been tackled in Chapter 4, which is the irrevocability of decisions. Then in Chapter 5 early classification of time series with indefinite length and having different labels per portion has been addressed.

In this chapter, we propose research directions for extending ECTS toward a more generic problem that we call *Machine Learning based Early Decision-Making* (ML-EDM). Ten challenges are proposed in order to develop ML-EDM approaches for a wide range of problems. Some of them have been addressed in this thesis, the rest is left to the scientific community for further research. These challenges are grouped into different categories, including learning tasks, types of data, online early decision-making, and revocable decisions. Then, some examples of applications of the ML-EDM techniques are provided. At last, we conclude with perspectives for the development of the ML-EDM field in the coming years.

The rest of the chapter is organized as follows. Section 6.2 first defines the ML-EDM problem, shows how a triggering strategy can be learned, and positions ML-EDM with respect to Reinforcement Learning. A series of ten challenges is then proposed in order to develop ML-EDM approaches for a wide range of problems. Section 6.3 considers a variety of *learning tasks*, and Section 6.4, a variety of *data types*. Section 6.5 gives some leads to address the problem of *online* ML-EDM. Section 6.6 extends ML-EDM to *revocable decisions*. Section 6.8 gives an overview on the proposed challenges, and makes a synthesis of long and short term application perspectives. Then, Section 6.9 provides some examples of *applications* of the ML-EDM techniques. At last, Section 6.10 concludes with perspectives for the development of the ML-EDM field in the coming years.

6.2 Definition of ML-EDM

This section defines what ML-EDM is by answering the following questions:

- **A-** What is an early decision?
- **B-** How to learn a triggering strategy from training data?
- **C-** Can a triggering strategy be learned by Reinforcement Learning?

Question A - *What is an early decision?*

Basically, Early Decision Making consists in: (*i*) observing pieces of information over time ; (*ii*) deciding when to make a decision ; and (*iii*) making the decision itself. In the following, increasingly complex decision-making problems are considered in order to progressively lead to a general definition of ML-EDM.

Two types of problems can be distinguished (Hansson, 1994). Decision-making *under ignorance* refers to a category of problems where the set of possible outcomes is known, but no information about their probabilities is available. By contrast, decision-making *under uncertainty* deals with problems where the probabilities of the possible outcomes are known, or partially known.

Optimal Stopping Problem (Shepp, 1969) is a canonical case of interest, where the decision to make is simply to stop receiving new pieces of information. More formally, $\{X_i\}$ is a sequence of random variables observed successively, whose joint distribution is known. Let $\{r_i\}$ be a sequence of reward functions, such that r_i is a function of the observed values $x_1, \ldots, x_i$. The objective is to maximize the reward, deciding after observing the value of the random variable X_i, either to stop and accept the reward r_i, or to observe the value of the next random variable X_{i+1}. A number of optimal stopping problems have been extensively studied in the literature, such as:

- The *Shepp's urn* (Shepp, 1969) which is filled with a known number of \$1 bills, and a known number of anti-bills of -\$1. Here, the reward is the sum of the bills gathered until the end of the game. The objective is to maximize our payoff by stopping to draw objects in this urn at the best time.
- The *secretary problem* (Ferguson, 1989) consists in selecting the largest possible value (which is unknown), among a sequence of values of known size observed in a uniform random order. At each step the choice is, either to stop and keep the last observed value, or to continue.

These two problems involve decision making *under uncertainty*, since the system under study is perfectly known and the probability of the possible outcomes can be estimated. For instance, in the Shepp's urn the probability of getting a bill or an anti-bill in the next draw is available, since the content of the urn is known at any time. In the secretary problem, the rank of the last value among the previously observed values approximates the rank in the entire set of values, since the observed values constitute a uniform sample of all values.

As in Early Decision Making problem, Shepp's urn and the secretary problem imply a trade-off between *early* and *accurate* decisions.

On the one hand, there is a *time pressure* which pushes to trigger early decisions. In a Shepp's urn, the number of objects is finite and if all of them are drawn, our payoff is bad, i.e. equal to the number of bills minus the number of anti-bills. In the secretary problem, the number of values is known. The more values are observed, the less future opportunity remains to select a high value.

On the other hand, there is a *gain of information* (about what's left in the urn) over time which tends to delay the decisions. In the Shepp's urn problem, the sample of already drawn objects grows over time, which provides useful information to be compared to the known quantities of bills and anti-bills. For the secretary's problem, the sample of already drawn values grows over time, and the last observed value can be compared to this sample.

From here on, the decision-making problems presented in the following are part of *supervised learning*. A set of labeled examples, which takes different forms depending on the problem, is assumed to be available.

The ECTS problem can be considered as a particular instance of optimal stopping, where the decision to be made consists in: (*i*) stopping receiving new measurements ; and (*ii*) predicting the class of the incoming time series. The hypothesis $h \in \mathcal{H}$ is assumed to be available, allowing to predict the class $y \in \mathbb{Y}$ of the incoming series at any time, such that $h(\mathbf{x}_t) = \hat{y}$. In this case, the reward function $r(\mathbf{x}_t, t, \hat{y}, y)$ depends on the observed measurements $\mathbf{x}_t = \langle x_1, \ldots, x_t \rangle$; the decision time t ; the predicted class $\hat{y}$; and the true class y. The following loss function can be defined:

$$\mathcal{L}(h(\mathbf{x}_t), t, y) = \mathcal{L}_{prediction}\left(h(\mathbf{x}_t), y\right) + \mathcal{L}_{delay}(t) \tag{6.1}$$

where $\mathcal{L}_{prediction}(.)$ is the cost of making a potentially bad prediction which can be expressed as a cost matrix, and $\mathcal{L}_{delay}(t)$ is a monotonically increasing function of t representing the cost of delaying the decision until t[1]. The best decision time t^* is given by the optimal triggering strategy $Trigger^*$ defined as:

$$Trigger^*(.) = \begin{cases} 1 & \text{if } t = t^* = Argmin_{t \in [1,T]} \mathcal{L}(h(\mathbf{x}_t), t, y) \\ 0 & \text{otherwise} \end{cases} \tag{6.2}$$

where the decision is forced at $t = T$ if it was not taken before.

Here, the trade-off between *early* and *accurate* decisions takes the following form. On the one hand, the delay cost $\mathcal{L}_{delay}(t)$ incurred in making a decision urges to make an early decision. On the other hand, the cost of making a bad prediction $\mathcal{L}_{prediction}$ is assumed to decrease over time, as the description of the incoming time series becomes richer. This decision making problem is *under uncertainty*, since the hypothesis h is capable of estimating the distribution of the possible outcomes $P(y|\mathbf{x}_t)$, at any time.

In practice, ECTS approaches trigger decisions at $\hat{t}$, hopefully the closest as possible to the optimal time t^*, at least in terms of cost: $\mathcal{L}(h(\mathbf{x}_{\hat{t}}), \hat{t}, y) - \mathcal{L}(h(\mathbf{x}_{t^*}), t^*, y)$ must be small. Triggering such a decision is an *online* optimization problem, since $\hat{t}$ must be chosen based on a partial description $\mathbf{x}_t$ of the incoming time series $\mathbf{x}_T$ (with $t \leq T$), and the reward function can be defined as:

$$r(\mathbf{x}_t, t, h(\mathbf{x}_t), y) = \begin{cases} -\mathcal{L}(h(\mathbf{x}_t), t, y) & \text{if } t = \hat{t} \text{ or } t = T \\ 0 & \text{otherwise} \end{cases} \tag{6.3}$$

where the risk equals to 0 when no decision is made, given that the decision is forced at $t = T$ resulting in an important risk due to the delay cost.

In the rest of this section, and for readability reasons, the deadline T is still considered as finite and known, as in the ECTS problem. In Section 6.6, another setting is studied where T is indeterminate, i.e. where the successive measurements are observed as a data stream.

Early decisions to be located in time constitute a more challenging problem, which consists of both making a decision for each incoming time series, but also predicting a *time period* associated with the decision. For example, maintenance operations on hydroelectric dam turbines can only be performed when the electricity demand is at a low enough level. There are therefore periods where maintenance is possible and periods where this is not desirable. The objective here is to determine as early as possible *whether* and during *which period* it will be possible to shut down the turbines, within the day (if $[1, T]$ corresponds to one day). In this case, the ground truth $(y, (s, e))$ consists of a class $y \in \mathbb{Y}$, associated with a certain time period $[s, e]$, defined by a *start* timestamp $s \in [1, T]$ and a *end* timestamp $e \in [s, T]$.

At testing time, the objective is twofold: triggering the decision as early as possible, while also predicting the associated time period $[s, e]$. Let us consider a decision denoted by $(h(\mathbf{x}_{\hat{t}}), (\hat{s}, \hat{e}))$, where $h(\mathbf{x}_{\hat{t}})$ is the class predicted at $\hat{t}$ (the triggering time), and $[\hat{s}, \hat{e}]$ is the associated predicted time period. The loss function $\mathcal{L}$ has to be redefined as a function of the following parameters:

[1]Note that the delay cost $\mathcal{L}_{delay}(t)$ could depend on the class y of the time series. For instance, in the emergency department in a hospital, the cost of delaying a decision when there is internal bleeding is not the same as the one in case of gastroenteritis, where the early symptoms could look the same. Here, for reasons of readability, we make $\mathcal{L}_{delay}$ depend only on t.

$$\mathcal{L}\left(\underbrace{(h(\mathbf{x}_t),(\hat{s},\hat{e}))}_{\text{predictions}},\ \underbrace{\hat{t}}_{\text{triggering time}},\underbrace{(y,(s,e))}_{\text{ground truth}}\right) \tag{6.4}$$

The loss function $\mathcal{L}$ needs to be specified further, depending on the considered application. In general, this loss function should account for two aspects: (*i*) the *quality* of the predictions ; (*ii*) the *time overlap* between the decisions made and the true decisions. For example, Figure 6.1 shows a situation where the decision made is correct, since the predicted class (see the second line) matches the ground truth (see the first line). But these two decisions do not coincide exactly in time, as the predicted time period is earlier than the ground truth.

FIGURE 6.1: Example of a time-lagged decision.

By extension, ML-EDM considers ***multiple early decisions to be located in time*** (i.e. in the time period $[1,T]$) which is necessary in numerous applications. For example, consider a set of servers used to trade on a stock exchange platform (where $[1,T]$ corresponds to the platform's hours of operation during the day). For each server, key performance indices (e.g., CPU, RAM, network) are recorded over time. The ground truth consists of a sequence of states (e.g., overload or nominal) associated with the corresponding time periods. In this application, the task is to detect overload periods as early as possible.

Thus, in this problem, the true decisions $\{y_i,(s_i,e_i)\}_{i=1}^{k_{\mathbf{x}}}$ consists of a sequence of varying length $k_{\mathbf{x}}$, which is specific to each individual $\mathbf{x}$. Each element of this sequence is a decision to be *located in time*, which consists of a class $y_i \in \mathbb{Y}$ associated with a certain time period $[s_i,e_i]$. For a given individual $\mathbf{x}$, the time periods $\{(s_i,e_i)\}_{i=1}^{k_{\mathbf{x}}}$ constitute a *time partition*, each interval $[s_i,e_i]$ being associated with the true class y_i (e.g. in predictive monitoring, this time partition would correspond to the successive states, up or down, of a given device).

Here, the online optimization problem to be addressed is more complex than the previous one, since it consists in triggering a sequence of decisions as soon as possible, without knowing the number of true decisions $k_{\mathbf{x}}$, and also ignoring the time periods associated with each true decision. Let us consider that a ML-EDM approach triggers a sequence of decisions $\{h(\mathbf{x}_{\hat{t}_{i'}}),(\hat{s}_{i'},\hat{e}_{i'})\}_{i'=1}^{\hat{k}_{\mathbf{x}}}$; where $\hat{k}_{\mathbf{x}}$ is the number of decisions made ; where $\{\hat{t}_{i'}\}_{i'=1}^{\hat{k}_{\mathbf{x}}}$ represents the associated triggering times ; and where $\{(\hat{s}_{i'},\hat{e}_{i'})\}_{i'=1}^{\hat{k}_{\mathbf{x}}}$ represents the predicted time periods associated to the decisions which forms a partition of the time period $[1,T]$. In the scenario of *multiple early decisions to be located in time*, a loss function $\mathscr{L}$ needs to be defined as a function of the following parameters:

$$\mathscr{L}\left(\underbrace{\{h(\mathbf{x}_{\hat{t}_{i'}}),(\hat{s}_{i'},\hat{e}_{i'})\}_{i'=1}^{\hat{k}_{\mathbf{x}}}}_{\text{predictions}},\ \underbrace{\{\hat{t}_{i'}\}_{i'=1}^{\hat{k}_{\mathbf{x}}}}_{\text{triggering times}},\underbrace{\{y_i,(s_i,e_i)\}_{i=1}^{k_{\mathbf{x}}}}_{\text{ground truth}}\right) \tag{6.5}$$

This equation shows the loss function used to evaluate an approach after the deadline T, when predictions have been taken for all instants in the time period $[1, T]$. The loss function $\mathscr{L}$ can be expressed in many different ways, depending on the application considered. In practice, *mapping rules* need to be defined to match the decisions made to the true ones (see Section 6.2.1).

Note that the problem of making predictions for all instants in $[1, T]$ points to the issue as whether decisions made can be revoked, or not, before T. In case decisions are irrevocable, once a decision has been made, let us say $(y, (s, e))$, then it is no longer possible to change the prediction of the class for all times $t \in (s, e)$. This renders the optimization problem dependent upon previous decisions, and it becomes more constraining for application cases. Revocable decision are studied in Section 6.6.

One of the major issues of ML-EDM is to make the time periods associated with the decisions taken $\{(\hat{s}_{i'}, \hat{e}_{i'})\}_{i'=1}^{\hat{k}_{\mathbf{x}}}$ coincide with those associated with the true decisions $\{(s_i, e_i)\}_{i=1}^{k_{\mathbf{x}}}$. In this respect, two situations can be distinguished which have an important impact on the triggering times $\{\hat{t}_{i'}\}_{i'=1}^{\hat{k}_{\mathbf{x}}}$:

- In the case of a decision making problem *under ignorance*, the occurrence of the next true decision is independent of data observed before it occurs. For example, in a predictive monitoring application, this means that there is no premise in data correlated to future failures. In this case, the triggering times $\hat{t}_{i'}$ will all happen after s_i, that is the beginning of the time period associated with the corresponding true decision (a situation where $\hat{t}_{i'} > s_i$ is called a *negative* prediction horizon).

- In the case of a decision making problem *under uncertainty*, data observed at the current time $\mathbf{x}_t$ may provide useful information to predict the occurrence of the next true decision. The partition of true decisions $\{(s_i, e_i)\}_{i=1}^{k_{\mathbf{x}}}$ is generated by a stochastic process which can be modeled conditionally to the observed data (as in (Frazier and Angela, 2007) which deals with a stochastic decision deadline). In a predictive monitoring application, this means that observed data include premises which are correlated to future failures. The triggering moments $\hat{t}_{i'}$ can then be located before s_i (a situation where $\hat{t}_{i'} < s_i$ is called a *positive* prediction horizon).

ML-EDM aims at designing approaches which correctly manage both situations, i.e. decision making under *ignorance* or *uncertainty*, without being informed about it. If the observed data contains useful information for predicting the timing of the next true decision, the approach should be able to exploit it. If not, the approach should still perform properly by triggering decisions under negative prediction horizons.

The ***deadline*** T after which decisions are forced is an important component, that takes different forms depending on the problem. In the simple case of ECTS, only one decision needs to be made before the incoming time series is complete. Thus, the deadline T is defined as the *maximum size* of the input series, which is known in advance during training. By contrast, in the more complex case of ML-EDM where multiple decisions to be located in time must be taken, the deadline T is defined as a *maximum delay* allowed to detect the start of a true decision (i.e. a bound on negative decision horizons). In practice, two situations can be distinguished:

- Some applications do *not support the absence of decision*, and the entire considered time period must be partitioned by the successive decisions. This is the case for

instance when moderating content on social networks, where discussions are continuously going on between users and where each part of these discussions must be classified as *appropriate* or *not* (see Section 6.9.4). In this case, no decision is not allowed, and all decisions are subject to the cost $\mathcal{L}_{delay}$ and thus constrained by the deadline T.

- By contrast, in some applications, a *nominal operating state* exists which is almost permanent, and for which there is no decision deadline. This is for instance the case in predictive maintenance applications, where there is no urgency or even a deadline to detect the absence of failure. In this case, the delay cost $\mathcal{L}_{delay}$ and also the deadline T apply only to the other decisions (e.g. failures categorized by severity level) excluding the nominal state.

At the end, ML-EDM aims to develop approaches which allow for easy adaptation to all cases, whether the deadline T is applicable to all decisions, or whether there exists a nominal operation state which bypasses this deadline.

Question B - *How to learn a triggering strategy from data?*

As a summary, this section shows that learning a triggering strategy follows the usual general principles of Machine Learning approach, with the particularity to consider *time-sensitive* loss functions (i.e. which depend on when decisions are triggered, as in Equations 6.1, 6.4 and 6.5).

In practice, the optimal triggering strategy is not available and it must be approximated by a learned function, such as $Trigger^{\gamma} \approx Trigger^{*}$, where $\gamma \in \Gamma$ is a set of parameters to be optimized within the space of parameters Γ of a chosen family of triggering strategies.

In addition, the hypothesis h is supposed to be learned previously during the training phase, making the system capable of predicting y at any time $t \in [1, T]$. This hypothesis is defined by a set of parameters $\theta \in \Theta$.

To illustrate what a triggering strategy is, let us consider an example from the ECTS literature. The SR approach, described in (Mori, Alexander Mendiburu, Dasgupta, et al., 2017), involves 3 parameters $(\gamma_1, \gamma_2, \gamma_3)$ to decide if the current prediction $h(\mathbf{x}_t)$ must be chosen (output 1) or if it is preferable to wait for more data (output 0):

$$Trigger^{\gamma}\left(h(\mathbf{x}_t)\right) = \begin{cases} 0 & \text{if } \gamma_1 p_1 + \gamma_2 p_2 + \gamma_3 \frac{t}{T} \leq 0 \\ 1 & \text{otherwise} \end{cases} \tag{6.6}$$

where p_1 is the largest posterior probability estimated by h, p_2 is the difference between the two largest posterior probabilities, and the last term $\frac{t}{T}$ represents the proportion of the incoming time series that is visible at time t. The parameters $\gamma_1, \gamma_2, \gamma_3$ are real values in $[-1, 1]$ to be optimized, as described more generally in the following.

In the simple case of ECTS, a single decision has to be made for each time series $\mathbf{x} \in \mathbb{X}$ (see Equation 6.1). Thus, the *risk* associated with any triggering strategy $Trigger^{\gamma}$ belonging to any family Γ, is defined as follows, given the previously learned hypothesis h^{θ} within the family Θ:

$$R(Trigger^{\gamma}|h^{\theta}) = \mathop{\mathbb{E}}_{\mathbb{X},\mathbb{Y}}\left[\mathcal{L}(h^{\theta}(\mathbf{x}_{\hat{t}}), \hat{t}, y)\right] \tag{6.7}$$

where $\hat{t}$ is determined by γ, the parameters of the triggering strategy.

Similarly, the risk can be defined in the more complex case of ***multiple early decisions to be located in time***. Let T_{part} be the set of all possible partitions of the time domain $[1, T]$, having a varying number of time intervals k. The risk can be defined as:

$$R(Trigger^{\gamma}|h^{\theta}) = \mathop{\mathbb{E}}_{\mathbb{X},T_{part},\mathbb{Y}^k}\left[\mathscr{L}(\{h^{\theta}(\mathbf{x}_{\hat{t}_{i'}}),(\hat{s}_{i'},\hat{e}_{i'})\},\{\hat{t}_{i'}\},\{y_i,(s_i,e_i)\})\right] \tag{6.8}$$

where $\{\hat{t}_{i'}\}$ and $\{(\hat{s}_{i'},\hat{e}_{i'})\}$ are determined by γ, and given h^{θ}. In Equation 6.8, the risk is an expectancy on three random variables, drawing triplets from the join distribution $P(\mathbf{x},\{(s_i,e_i)\},\{y_i\})$. The first element corresponds to the input data[2], which is an individual $\mathbf{x} \in \mathbb{X}$. The two other consist of the ground truth, which is composed of: (*i*) a partition of the time domain $\{(s_i,e_i)\} \in T_{part}$ with a particular number of time intervals, denoted by $k \in [1,T]$; (*ii*) and a set of class labels $\{y_i\} \in \mathbb{Y}^k$ for each time interval.

Now, the objective is to approximate the optimal triggering strategy $Trigger^*$ by finding $\gamma^* \in \Gamma$ which minimizes the risk, such that:

$$\gamma^* = Argmin_{\gamma\in\Gamma} R(Trigger^{\gamma}|h^{\theta}) \tag{6.9}$$

The joint distribution $P(\mathbf{x},\{(s_i,e_i)\},\{y_i\})$ is unknown, thus Equation 6.8 can not be calculated ; however a training set $\mathcal{S}$ which samples this distribution is supposed to be available. The risk can be approximated by the *empirical risk* calculated on the training set $\mathcal{S} = \{\mathbf{x}^j,\{y_i^j,(s_i^j,e_i^j)\}\}_{j\in[1,n],i\in k_{\mathbf{x}_j}}$, as follows:

$$R_{emp}(Trigger^{\gamma}|h^{\theta}) = \frac{1}{n}\sum_{j=1}^{n}\mathscr{L}\left(\{h(\mathbf{x}^j_{\hat{t}_{i'}}),(\hat{s}^j_{i'},\hat{e}^j_{i'})\},\{\hat{t}_{i'j}\},\{y^j_i,(s^j_i,e^j_i)\}\right) \tag{6.10}$$

where $\hat{t}_{i'j}$ is the triggering time of the *i-th* made decision of the *j-th* individual.

At the end, training a ML-EDM approach can be viewed as a *two-step* Machine Learning problem: (*i*) *first*, the hypothesis h^{θ} must be learned in order to predict the most appropriate decision $h^{\theta}(\mathbf{x}_t)$, at any time $t \in [1,T]$; (*ii*) *second*, the best triggering strategy defined by γ^* must be learned, given the hypothesis h^{θ} and given the family Γ, such that:

$$\gamma^* = Argmin_{\gamma\in\Gamma} R_{emp}(Trigger^{\gamma}|h^{\theta}) \tag{6.11}$$

Question C - *Can a triggering strategy be learned by Reinforcement Learning?*

To sum up, this section shows that learning a triggering strategy of a ECTS approach can be cast as a Reinforcement Learning (RL) problem, with rewards well chosen, and it might be expected that provided with sufficient training, RL learning may end up with a good approximation of an efficient decision function.

[2]Notice that the notation $\mathbf{x} \in \mathbb{X}$ in Equations 6.7 and 6.8 is an abuse that we use use to simplify our purpose. In all mathematical rigor, the measurements observed successively constitute a family of time-indexed random variables $\mathbf{x} = (\mathbf{x}_t)_{t\in[1,T]}$. This stochastic process $\mathbf{x}$ is not generated as commonly by a distribution, but by a filtration $\mathbb{F} = (\mathcal{F}_t)_{t\in[1,T]}$ which is defined as a collection of nested σ-algebras (Klenke, 2013) allowing to consider time dependencies. Therefore, the distribution $P(\mathbf{x},\{(s_i,e_i)\},\{y_i\})$ should also be re-written as a filtration.

Reinforcement learning (Sutton and Barto, 2018) aims at learning a function, called a policy π, from states to actions: $\pi : \mathcal{S} \rightarrow \mathcal{A}$. Rewards can be associated with transitions from states $s_t \in \mathcal{S}$ to states $s_{t+1} \in \mathcal{S}$ under an action $a \in \mathcal{A}$. Rewards are classically denoted $r(s_t, a, s_{t+1}) \in \mathbb{R}$. In all generality, the result of an action a in state s_t may be non deterministic and one among a set (or space) of states. The optimal policy $\pi^\star$ is the one that maximizes the expected gain from any state $s_t \in \mathcal{S}$. This gain, denoted R_t starting from the sate s_t, is defined as a function of the rewards from that state (e.g. a discounted sum of the rewards received). In order to learn a policy, value functions can be considered, such as the state-value function $v_\pi(s)$ classically defined as:

$$v_\pi(s_t) \doteq \mathbb{E}_\pi[R_t \,|\, s_t] = \sum_{a \in \mathcal{A}} \pi(a|s_t) \sum_{s_{t+1}, r} p(s_{t+1}, r \,|\, s_t, a) \left[r(s_t, a, s_{t+1}) + \gamma \, v_\pi(s_{t+1}) \right] \tag{6.12}$$

where $\mathbb{E}_\pi[\cdot]$ denotes the expected value of a random variable given that the agent follows the policy π and t is any time step. In the case of a non deterministic policy, $\pi(a|s_t)$ denotes the probability of choosing action a in state s_t and $p(s_{t+1}, r \,|\, s_t, a)$ the probability of reaching state s_{t+1} and receiving the reward r given that the action a has been chosen in state s_t. And γ is a discounting factor: $\gamma < 1$.

In our case, the agent aims to learn a triggering strategy given the previously learned classifier h^θ, and the state $s_t = (t, \mathbf{x}_t)$ is the current time t and the observed data at current time. The instantaneous reward $r(s_t, a)$ only depends on the current state s_t and the action taken a (i.e. prediction now, or postponed to a later time). Finally, the discounted factor γ, usually present in RL for reasons of convergence over infinite episodes, is equal to 1 in our case, since we always deal with finite episodes with forced decisions after a maximum delay. So that the equation (6.12) simplifies to:

$$v_\pi(s_t) = \sum_{a \in \mathcal{A}} \pi(a|s_t) \, r(s_t, a) + v_\pi(s_{t+1})$$

when, during learning, the agent takes a decision, it updates the value of the state s_t using:

$$v_\pi(s_t) = r(s_t, a) + v_\pi(s_{t+1})$$

where s_{t+1} is the state after having taken the action a in state s_t.

As the equation above shows, the core observation in RL is that the value function for a state s_t (i.e. an estimation of the expected gain from that state) is related to the value function of states s_{t+1} that may be reached from s_t. In that way, information gathered further down a followed path can be back-propagated to previous states thus allowing increasingly better decisions from those states to be made.

For instance, in game playing, rewards may happen both during play (e.g. the player just lost a pawn) and at the end of the game (e.g. the player is chess mate). Similarly, one could cast the ECTS problem as a RL problem where, at each time step, the "player" is in state $s_t = (t, h(\mathbf{x}_{t_k}), \mathbf{x}_t)$ and should choose between making a prediction (e.g. $h(\mathbf{x}_t)$) with an associated reward: $r_t = -\mathcal{L}(h(\mathbf{x}_t), t, y) = -\mathcal{L}_{prediction}\left(h(\mathbf{x}_t), y\right) - \mathcal{L}_{delay}(t)$ or postpone the decision, with no immediate associated reward, that is $r_t = 0$. If no decision has been made before the term of the episode (e.g. when $t = T$) a decision is forced (see Figure 6.2). Provided with enough time series to train on, and sufficient training in the form of "playing" these time series, a reinforcement learning agent may end up with a policy $\widehat{\pi}$ that approximates a good early triggering strategy, one that would converge over time, after a very large number of "plays" on the training time series, to the optimal decision function $\pi^\star$

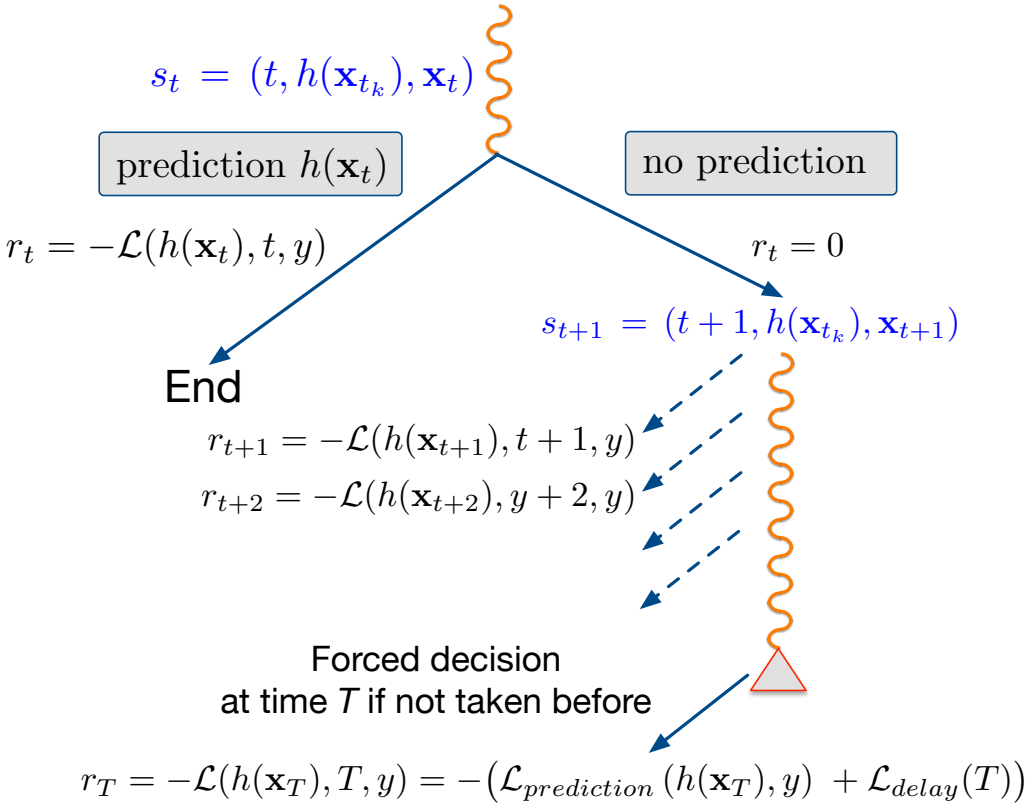


FIGURE 6.2: A part of a ECTS "game" when learning an optimal policy while "playing" a training time series. When a prediction is made, the game stops, otherwise it continues until a prediction is made or the term of the episode is reached.

(See Equations 6.2 and 6.11).

The RL framework is very general. It uses immediate and delayed rewards. As shown in this section, there is in principle no obstacle to apply RL to the learning of a good triggering strategy. However, if used directly, the generality of RL is paid for by a need for a large number of "experiments". In addition, the state space is continuous in the case of the ECTS problem, thus an interpolating functions must be used in order to represent the values such as $v_\pi(s)$ and this entails the choice of a family of functions and setting their associated parameters. Another approach, the one favored in the current literature for ECTS (Achenchabe, Bondu, and Cornuéjols, 2021), is to choose functions for representing the expected values of decision times, and thus providing a ground for the triggering strategy.

This has the merit of incorporating prior knowledge of the trade-off between earliness and accuracy, at the cost of making modelling choices that may bias the method of estimating the expected future cost.

6.2.1 In practice, how to define the loss function ?

The loss function $\mathscr{L}$ in Equation 6.5 can be expressed in many different ways, depending on the application considered. In practice, *mapping rules* need to be defined to match the decisions made to the true ones. In Equation 6.5, the purpose is to map the indices i' to i, considering that the number of decisions made may be different than it should be ($\hat{k}_\mathbf{x} \neq k_\mathbf{x}$). In particular, these rules address the following questions: (*i*) how long should we wait before considering that a true decision has been missed? (see a rule example in Figure 6.3) (*ii*) when the number of decisions made is too large, how to identify the undue decisions? (e.g. Figure 6.4) (*iii*) what is the minimum time overlap between a decision made and the corresponding true one? (e.g. Figure 6.5) And of course, these mapping rules are specific to each application.

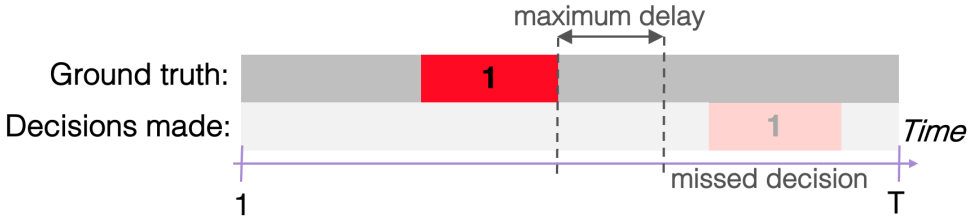


FIGURE 6.3: Maximum delay after which a decision is considered as missed.

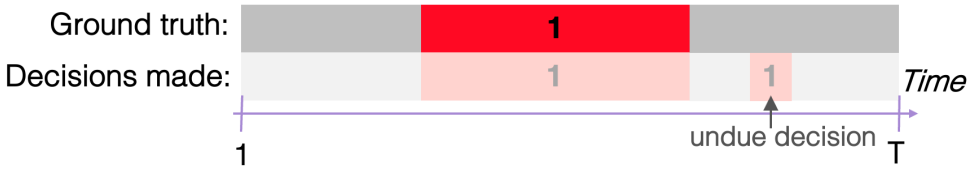


FIGURE 6.4: A decision is undue if no true decision exists in the time interval.

In its general form, the loss function $\mathscr{L}$ should involve several decision costs mentioned below. Their origin is further detailed in Chapter 4. For the moment, let us consider that the following costs are fixed, deterministic, and given as input to an ML-EDM approach:

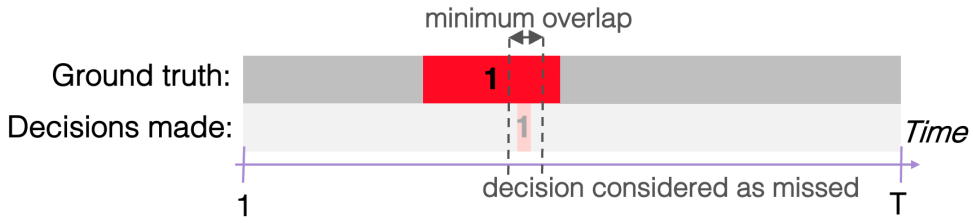


FIGURE 6.5: Minimum overlap to consider that a decision is not missed.

- a prediction cost $\mathcal{L}_{prediction}$,
- a delay cost $\mathcal{L}_{delay}$,
- a time overlap cost $\mathcal{L}_{overlap}$,
- a cost of missing the decision $\mathcal{L}_{missing}$,
- a cost of an extra and undue decision $\mathcal{L}_{delete}$.

As in the ECTS problem, the *prediction* cost $\mathcal{L}_{prediction}$ accounts for a potentially bad prediction and it can be expressed as a cost matrix. The delay cost $\mathcal{L}_{delay}$ depends on the trigger time $\hat{t}_{i'}$ and the time period associated to the i-th true decision $[s_i, e_i]$. Figure 6.6 gives an example where a delay cost is paid since the triggering time (see the green vertical line) is located after the beginning of the period associated with the decision. The *overlap* cost $\mathcal{L}_{overlap}$ accounts that the predicted periods $\{(\hat{s}_{i'}, \hat{e}_{i'})\}_{i'=1}^{\hat{k}_{\mathbf{x}}}$ might not coincide temporally with the periods of the true decisions $\{(s_i, e_i)\}_{i=1}^{k_{\mathbf{x}}}$. For instance in Figure 6.7, the decisions made (shown in the second line) are out of sync with the truth decisions (see the first line), which results in four overlapping periods. The interested reader may refer to (Tatbul et al., 2018) which addresses the evaluation of models by considering such temporal overlap. Finally, the costs of missing a decision $\mathcal{L}_{missing}$ and making an additional undue one $\mathcal{L}_{delete}$ account that the number of decisions made can be different than it should be ($\hat{k}_{\mathbf{x}} \neq k_{\mathbf{x}}$). Figure 6.8 shows a situation where the first anomaly (represented by the class 1) is not detected, incurring a missing cost, and where a false detection occurs at the end leading to a delete cost.

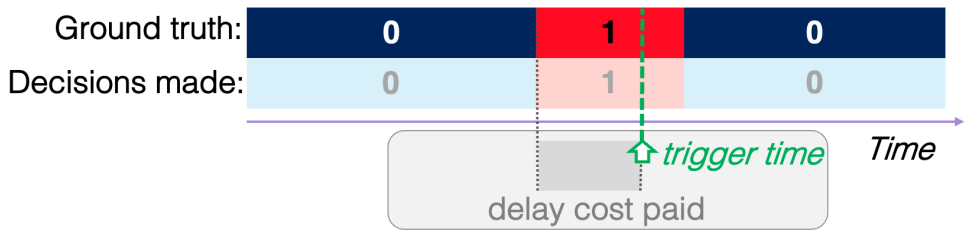


FIGURE 6.6: Example of paid delay cost.

6.2.2 In practice, how to evaluate a ML-EDM approach?

In some applications, decision costs are available as prior knowledge. It is the case for instance in (Khoshnevisan and Chi, 2021b), where the objective is to detect as early as possible patients suffering from septic shock, and where the cost of delaying decisions

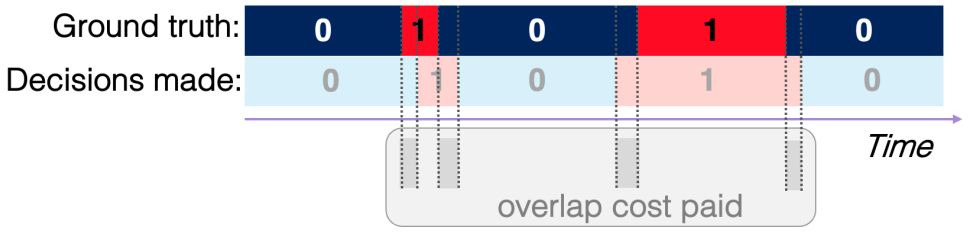


FIGURE 6.7: Example of paid overlap cost.

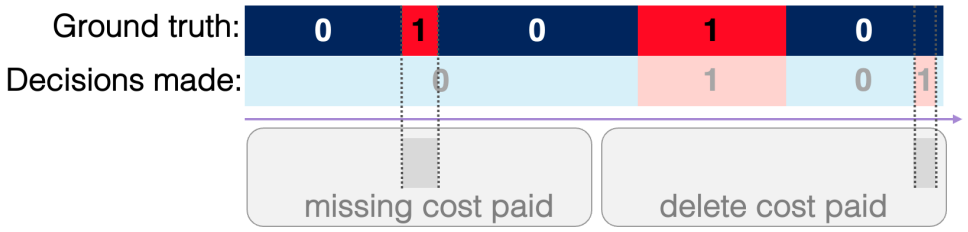


FIGURE 6.8: Example of a missing decision and an extra undue one.

is perfectly known. When available, *decision costs* are of great help in evaluating ML-EDM approaches. Indeed, each decision made can be evaluated by the amount of costs actually incurred, considering: (*i*) the triggering moment ; (*ii*) the ground truth ; (*iii*) and the value of the decision costs (i.e. $\mathcal{L}_{delay}$, $\mathcal{L}_{prediction}$, and $\mathcal{L}_{revoke}$).

In the particular case of ECTS problem, a cost-based evaluation criterion is proposed in Chapter 3 called *AvgCost*, which simply corresponds to the *empirical risk* calculated on a set of test individuals $\mathcal{S}$ as following:

$$\begin{aligned} AvgCost(\mathcal{S}) &= \sum_{(\mathbf{x}_T,y)\in\mathcal{S}} \mathcal{L}(h(\mathbf{x}_{t^*}),t^*,y) \\ &= \sum_{(\mathbf{x}_T,y)\in\mathcal{S}} \mathcal{L}_{prediction}\left(h(\mathbf{x}_{t^*}),y\right) + \mathcal{L}_{delay}(t^*) \end{aligned} \tag{6.13}$$

where t^* is the triggering moment predicted for $\mathbf{x}$. *AvgCost* can also be interpreted as the average cost paid by the user on a particular set of examples, and this quantity should be minimized.

In the case of early and revocable time series classification (see chapter 4), multiple decisions to classify the time series at hand are taken between timestamp 1 and T, the cost of revocation is added to Eq. 6.13 and becomes:

$$\begin{aligned} AvgCost(\mathcal{S}) = \sum_{(\mathbf{x}_T,y)\in\mathcal{S}} [&\mathcal{L}_{prediction}\left(h(\mathbf{x}_{t^*_{\text{last}}}),y\right) + \mathcal{L}_{delay}(t^*_{\text{last}}) \\ &+\mathcal{L}_{revoke}\left(\{(h(\mathbf{x}_{\hat{t}_i}),\hat{t}_i)\}_{i\in[1,D^{\mathbf{x}}]}\right)] \end{aligned} \tag{6.14}$$

where $t^*_{\text{last}} = t_{D^{\mathbf{x}}}$ is the moment of the last decision. Furthermore, the loss of revocation is the sum of the multiple costs of decisions changes, the ideal situation is to have the less possible decision changes per time series, in order to guarantee a stable decision making algorithm.

$$\mathcal{L}_{revoke}\left(\{(h(\mathbf{x}_{\hat{t}_i}),\hat{t}_i)\}_{i\in[1,D^{\mathbf{x}}]}\right) = \sum_{i=1}^{D^{\mathbf{x}}-1} C_{cd}(h(\mathbf{x}_{\hat{t}_{i+1}})|h(\mathbf{x}_{\hat{t}_i}))$$

In the case of dealing with open time series (Chapter 5), we proposed to evaluate decisions for each timestamp, the cost of delay becomes the cost of horizon.

In cases where decision costs are unavailable, a *multi-criteria* evaluation can be considered to take into account the different costs. For instance, in (Mori, Alexander Mendiburu, Miranda, et al., 2019), the *earliness* and *accuracy* of decisions are evaluated separately, and the Pareto optimal front is made up of the dominant approaches considering both criteria.

In the more general case of ML-EDM where multiple early decisions have to be made, a cost-based evaluation requires more prior knowledge. Indeed, mapping rules would have to be defined in order to match the triggered decisions with the true ones, the cost of overlap $\mathcal{L}_{overlap}$ between predicted and true time periods, as well as the costs of missing a decision $\mathcal{L}_{missing}$ and triggering an undue one $\mathcal{L}_{delete}$ (see Equation 6.5 in Section 6.2 and Figures 6.3 to 6.8).

The respective performances, merits and limits of both approaches should be studied empirically by a comparison of RL based ECTS approaches, such as (Martinez, Ramasso, et al., 2020), with approaches that explicitly exploit the form of the optimization criterion designed for ECTS as in (Achenchabe, Bondu, and Cornuéjols, 2021).

6.3 Learning tasks

The formal definition of ML-EDM provided in Section 6.2 involves the ground truth. However, in many applications, it is extremely hard or costly to obtain, especially in the case of anomaly detection (e.g. fraud, cyber-attacks, predictive maintenance). In these application domains, there are several issues: (*i*) labels can be extremely expensive to obtain as they each require an examination from an expert ; (*ii*) the labels provided by experts can be uncertain ; and (*iii*) the class of anomalous observations is often poorly represented and drifts over time. For example, cyber-attack techniques are very diverse and change with time. Faced with these difficulties, anomaly detection is often addressed using unsupervised approaches, by assuming that the anomalies are outliers. In this case, the problem comes down to modeling the normal behavior of the system, if possible using historical data that are cleaned of anomalies. Then, it is necessary to define the notion of outlier to be able to assign an eccentricity score to the new observations. Note that this type of modeling can be considered as a *first step* to manage non-stationarity, since in this case the stationarity assumption only concerns the normal behavior of the system (this assumption could be removed in future work).

Challenge #1: extending non-myopia to unsupervised approaches
An unsupervised ML-EDM problem could be to decide, as soon as possible, whether a partially observed time series $\langle x_1, x_2, \ldots, x_t \rangle$ will be an outlier (or not) when fully observed at time T. In this case, the *accuracy vs. earliness* trade-off still exists.

On the one hand, an early detection is inaccurate by nature because the outlier series is unreliably detected, based on few observed measurements. On the other hand, delaying the detection of anomalies can be very costly. For instance, a cyber-attack which is not detected immediately gives time to the hackers to exploit the security issue found.

The non-myopic property has shown a big potential in the ECTS problem under the supervised setting. An approach is non-myopic if it takes into account the

information about the future to make the current decision, it does not only decide if the current moment is the optimal one to trigger a decision, but also estimates when in the future the decision is going to be optimal.

Designing ML-EDM approaches to tackle *unsupervised* learning tasks is challenging in several respects: (*i*) learning a triggering strategy with the goal of achieving a good trade-off between earliness and accuracy of its decisions cannot be achieved in the Machine Learning framework and should be formalized in another way ; (*ii*) developing unsupervised *non-myopic* approaches is very difficult, as the training set does not contain anomalous series, thus the triggering strategy cannot learn from their continuations.

The extension of ML-EDM both to *online* scenarios (discussed later in Section 6.5) and to *unsupervised* tasks is of particular interest, because combined they would enable a new generation of *monitoring systems* (Abellan-Nebot and Subirón, 2010) to be developed. In this case, the learning task would consist in detecting online the start and end of the outlier chunks: (*i*) without requiring labels to learn the model ; (*ii*) by considering the trade-off between *accuracy* and *earliness* to trigger the decisions at the right time.

Challenge #2: addressing other supervised learning tasks
The formal description of ML-EDM proposed is generic, in the sense that the type of the target variable y can easily be changed. By definition, the ECTS approaches in the literature are limited to *classification* problems, but they could naturally be extended to other supervised learning tasks. For instance, predicting a *numerical* target variable from a time series is a problem known as *Time Series Extrinsic Regression* (TSER)(Tan et al., 2021). In some domains, TSER approaches are very useful and allow applications such as the prediction of the daily energy consumption of a house, based on the last week's consumption, temperature and humidity measurements. *Early* TSER would consist of predicting the value of the numerical target variable as soon as possible, while ensuring proper reliability. Another example of a supervised task for which ML-EDM approaches could be developed is time series *forecasting* (Chatfield, 2000). Basically, a forecasting model aims to predict the next measurements of a time series up to a horizon ν, $Y = \langle x_{t+1}, x_{t+2}, \ldots, x_{t+\nu} \rangle$ from the recent past measurements $X = \langle x_{t-w}, \ldots, x_{t-1}, x_t \rangle$. Using a forecasting model, in a an online and *early* way, would consist of adapting the forecast horizon $t + \nu$ according to the observed values in X, by modeling the trade-off between the *accuracy* and the *earliness* of these predicted values.

The ML-EDM problem should also be adapted to *semi-supervised* learning, which is of great help when the ground truth is only partially available. More generally, the collected ground truth may be imperfect for various practical reasons, such as the labeling cost, the availability of experts, the difficulty of defining each label with certainty, etc. This problem has recently gained attention in the literature through the field of *Weakly Supervised Learning* (WSL) (Z.-H. Zhou, 2018) which aims to list these problems and provide solutions.

As detailed in (Nodet, Lemaire, Bondu, Cornuéjols, and Ouorou, 2021), the collected labels may suffer from three main deficiencies: (*i*) inaccuracy ; (*ii*) non-adaptability ; (*iii*) or even incompleteness. More precisely:

i) *Inaccuracy* of labels is commonly modeled as noise: the probability distribution that a label is corrupted may be uniform (i.e. completely at random), may

depend on the class value (i.e. at random), or even it may depend on the instance by varying in the input space (i.e. not at random);

ii) *Non-adaptability* of labels gathers a variety of situations, such as Transfert Learning (Zhuang et al., 2020) and Multi-instance Learning (Carbonneau et al., 2018), where the training labels may be available in another target domain or a sub-domain (i.e. proxy domain), the labels may come from a slightly different concept than the one to be learned (i.e. proxy labels), or labels can be associated with individuals defined in a slightly different way (i.e proxy individual) ;

iii) *Incompleteness* of labels is related to partially labelled training sets. The objective is to use the entire training set, including the unlabeled examples, to achieve better classification performance than learning a classifier only from labeled examples. Active Learning (Settles, 2009), Semi-Supervised Learning (Seeger, 2000), Positive Unlabeled Learning (Bekker and Davis, 2020), Self-Training (Ennaji, Mammass, El Yassa, et al., 2012) and Co-training (Blum and Mitchell, 1998) are suitable techniques for this situation.

Challenge #3: early weakly-supervised learning
The extension of ML-EDM to weakly-supervised learning is an interesting challenge, as it would allow to better address applications where the ground truth has corruptions or is incomplete (which includes semi-supervised learning). However, the weakly-supervised learning is a very large domain with many types of supervision deficiencies to be studied. From a practical point of view, the priority is probably to extend ML-EDM to label noise, and more specifically to bi-quality learning (Nodet, Lemaire, Bondu, and Cornuéjols, 2020), where the model is trained from two training sets: (*i*) one trusted with few labels ; (*ii*) the other, untrusted, with a large number of potentially corrupted labels. This would allow interesting applications, such as in cyber security where few labels are investigated by an expert, and the majority of labels are provided by rule-based systems. The major difficulty in designing *bi-quality learning* ML-EDM approaches is to learn a triggering strategy from these two training sets, which models the compromise between *accuracy* and *earliness* in a robust way to label noise. Another interesting avenue would be to adapt *Active Learning* (Settles, 2009) approaches to ML-EDM, with the goal of labeling examples which improve both *accuracy* and *earliness* of the decisions. Such approaches would be particularly helpful when early decisions have to be made, and when labeling examples is very costly as, again, it is the case in cyber security applications.

6.4 Types of data

The ML-EDM definition involves measurements (i.e. scalar values) acquired over time. However, this is only for reasons of simplicity of exposition. Ideally, ML-EDM approaches should be *data type agnostic*, i.e. they should operate for any data type as long as measurements are made over time and decisions are online.

Below, we outline data types that are present in applications where ML-EDM could be used.

i) *Multivariate time series* consist of successive measurements each containing more than one numerical value. More formally, a multivariate time series of length t is defined as $\mathbf{x}_t = \langle (x_1^1 \dots x_1^k), \dots, (x_t^1 \dots x_t^k) \rangle$, where k is the number of numerical values composing each time-indexed vector. For example, the

data from an accelerometer can be represented by a multivariate time series composed of the three variables representing the accelerations along the X, Y, and Z axes.

ii) More *complex signals* exist, such as video streams which involve higher dimensionality. Typically, a video stream is composed of frames, i.e. the measurements are images. Each time-indexed frame can be considered as a signal that is indexed by the position (x, y) of the pixels. The values of this multi-indexed signal are the pixel, defined by four values which encode the RGB components and the luminance of each pixel. Each time-indexed frame can be considered as a 2D signal composed of 3 channels which describe the RGB components. In addition, video streams could be multimedia when they include speech and transcript (text or sign language translation).

iii) *Data streams* is another type of data which can contain both numeric and categorical variables (Bifet et al., 2018). Successive measurements, also called *tuples*, are received in an uncontrolled order and speed. For example, an Internet of Things (IOT) device such as a connected security camera may send measurements in an irregular flow. These measurements may be composed of categorical variables such as an alarm type, and numerical variables such as a signal encoding a short video sequence.

iv) Another type of data is *evolving graphs* which consist of graphs whose structure changes over time (Latouche and Rossi, 2015). A typical example of an evolving graph is the one which represents the social network of the customers of a telecom provider (i.e. the nodes of the graph) and their interactions through the phone network (i.e. the edges of the graph). A new customer may appear, leading to the creation of a new *node* in the graph ; or two customers may meet in real life and initiate phone calls resulting in the creation of new *edges* in the graph. Several types of learning tasks can be considered, such as predicting the next changes in the graph structure, or the classification of parts of the graph (e.g. nodes, edges, sub-graphs).

v) Successive snapshots of *relational data* (Džeroski, 2009) should be consider to design new ML-EDM approaches. More precisely, relational data consists of a collection of tables having logical connections between them. (e.g. in a relational database system, two tables can be linked together by a foreign key). Generally, raw data stored in information systems can be represented by this type of data, which makes relational data a very widespread data type. For instance, consider that the customers of a company are described in a main table, each row containing the information about a particular customer. The contracts subscribed by the customers can be represented by a secondary table, linked to the main table with a one to many relationship.

   Like other types, relational data can evolve over time: (*i*) the connections between tables can change; (*ii*) as well as the structure of the tables; (*iii*) or even the values of the information stored in the tables.

vi) *Text* is another widespread type of data. There are many application cases where text data is collected over time, and a decision has to be made at a certain moment. For example, this is the case of an e-mail exchange to sell a second-hand car, where the seller realizes after several exchanges that it is most likely a scam.

An application example is the moderation of social networking platforms, with early deletion of inappropriate contents and automatic closure of fraudulent accounts (see Section 6.9.4). In the Machine-Learning literature, there is little work which considers early decision on texts (Y. He et al., 2018; Xia, Xuan, and J. Yu, 2020) even though this is likely a future application area of considerable interest, and the development of ML-EDM methods would enable to optimize the time of decision making for text data.

Challenge #4: data type agnostic ML-EDM
Ideally, the new developed ML-EDM approaches should be *data type agnostic*, i.e. they should operate for any data type presented above. To do so, a pivotal format needs to be defined in order to learn the triggering strategies in a generic way. For instance, each learning example could be characterized by a series of T predictions indexed by time (corresponding to the output of the learned hypothesis $h(\mathbf{x}_t)$ for each time step $t \in [1, T]$), as well as by $\{y_i, (s_i, e_i)\}_{i=1}^{k_\mathbf{x}}$ the ground truth composed of the true decisions to be made over time for this individual. In the particular case of ECTS, some approaches can easily be adapted to become agnostic to data type (Achenchabe, Bondu, and Cornuéjols, 2021; Mori, Alexander Mendiburu, Eamonn Keogh, et al., 2017; Mori, Alexander Mendiburu, Miranda, et al., 2019). In contrast, others have been designed to be very specific to time series (Ghalwash, Radosavljevic, and Obradovic, 2014; G. He, Duan, R. Peng, et al., 2015; Xing, Pei, and Philip, 2009; Xing, Pei, P. S. Yu, et al., 2011), especially with the search of features (e.g. shapelets) occurring early in the time series and helping to discriminate between classes. More generally, future work in ML-EDM should definitely promote data type agnostic approaches, to allow the use of these techniques in a wide range of application conditions.

6.5 Online Early Decision Making

In the specific case of Early Classification of Time Series (ECTS), an important *limitation* is that the training time series: (*i*) have the same length T ; (*ii*) correspond to different *i.i.d* individuals ; (*iii*) have a label which characterizes the whole time period of length T. There are obviously applications where this formulation of the problem is relevant (Alipour-Fanid et al., 2019; Dachraoui, Bondu, and Cornuejols, 2013; Fahrenkrog-Petersen et al., 2019; Gupta et al., 2020; Loyola et al., 2017; Rußwurm, Tavenard, et al., 2019; Sharma and Kumar Singh, 2020; Teinemaa et al., 2018), especially in cases where the *start* and *end* of the time series are naturally defined (e.g. a day of trading takes place from 9:30am to 4pm, during the opening hours of the stock exchange).

The development of *online* ML-EDM approaches could overcome these limitations and enable a new range of applications. For this purpose, let us consider that the input measurements are observed without interruption, in the form of a *data stream* (Joao Gama, 2012). In the case of a *classification problem*, an online ML-EDM approach would consist in identifying *chunks* in the input data stream (i.e. fixed time-windows defined by their start and end timestamps) and *categorizing* them according to a predefined set of classes. For example, in a predictive maintenance scenario (Ran et al., 2019) such an approach would operate on a continuous basis to detect periods of system malfunction as soon as possible.

Challenge #5: online and early predictions to be located in time
In the case of a *classification* problem, the training data consist of the measurements observed from the stream during the training period, denoted by $\mathbf{x} = \langle x_1, x_2, \ldots, x_{|\mathbf{x}|} \rangle$,

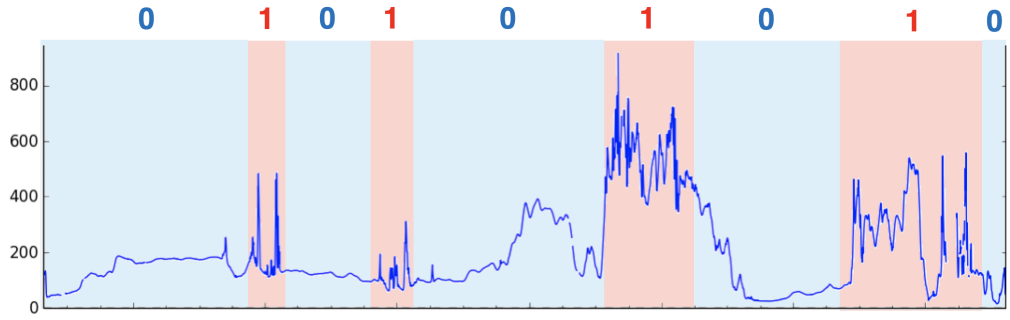


FIGURE 6.9: Example of a data stream labeled by chunks over a time period

associated with their labels $\mathbf{y} = \langle y_1, y_2, \ldots, y_{|\mathbf{x}|} \rangle$. A *labeled chunk* is formed by the consecutive measurements, between the timestamps t_a and t_b, if their labels share the same value (i.e. if $\{y_i\}_{i \in [t_a, t_b]}$ is a singleton). As shown in Figure 6.9, the data stream defined over the training period is labeled by chunks of variable size. For example, these chunks could represent the periods of failure and nominal operation in a predictive maintenance scenario. During the deployment phase, the model is applied online on a data stream whose measurements are observed progressively over time. This model is expected to provide *predictions located in time*, since it needs to predict the *beginning* and the *end* of each chunk, associated with the predicted *class* which characterizes the state of the system during this chunk.

Challenge #6: online accuracy vs. earliness trade-off
Designing *online* ML-EDM approaches requires redefining the accuracy vs. earliness trade-off for online decisions. The main issue is that a data stream is of indeterminate length: (*i*) its *beginning* may be too old to be considered explicitly, or can even be indeterminate ; (*ii*) its *end* is never reached, since it is constantly postponed by the new measurements that arrive. In the particular case of ECTS, it is precisely the fact that the input series has a maximum length T, known in advance, that leads to force triggering the decision when the current time t becomes close to the *deadline* T.

The rest of this paragraph presents an example of adapting the *accuracy vs. earliness* trade-off to online decisions, which is a summary of the work presented in Chapter 5.

Let us consider a predictive maintenance problem for which a classifier has been trained in batch in order to detect the beginning and the end of abnormal chunks (see Figure 6.10). The prediction of the classifier focuses on a fixed timestamp s and the question is to determine if this timestamp corresponds (or not) to the beginning of an abnormal section. The input features used by the classifier are extracted from a sliding window $\mathbf{x}_t = \langle x_{t-w}, \ldots, x_{t-1}, x_t \rangle$ of length w. As shown in Figure 6.10, the sliding window $\mathbf{x}_t$ moves over time as it gets closer to s. At first, the timestamp s is located in the future ($s > t$). Making a good prediction is difficult since the potentially anomalous part of the stream is not yet visible in $\mathbf{x}_t$. In this case, the classifier has to detect the early signs of an anomaly. Then, the timestamp s enters the $\mathbf{x}_t$ window (at time $t = 4$). The prediction becomes easier to perform, since a part of the potentially abnormal chunk is visible in $\mathbf{x}_t$. The last possible moment to trigger the decision is reached when the timestamp s is getting ready to exit the sliding window $\mathbf{x}_t$.

Finally, the accuracy vs. earliness trade-off occurs as follows: (*i*) on the one hand, the accuracy of the decisions increases over time due to the classification task that becomes easier as the $\mathbf{x}_t$ window shifts ; (*ii*) on the other hand, predictive maintenance applications require early decisions which allow to anticipate breakdowns, or at least

to detect it early. Ultimately, this proposal consists of changing the definition of *what is predicted* as normal or abnormal. Here, the observation to be scored is no longer a time series of finite length, but a particular measurement of the input data stream identified by its timestamp. This proposal only partially addresses the problem, as the predictions for each timestamp would have to be consolidated in order to predict the start and end of each chunk. There are certainly other ways to adapt the accuracy vs. earliness trade-off to online decisions that would be valuable to investigate.

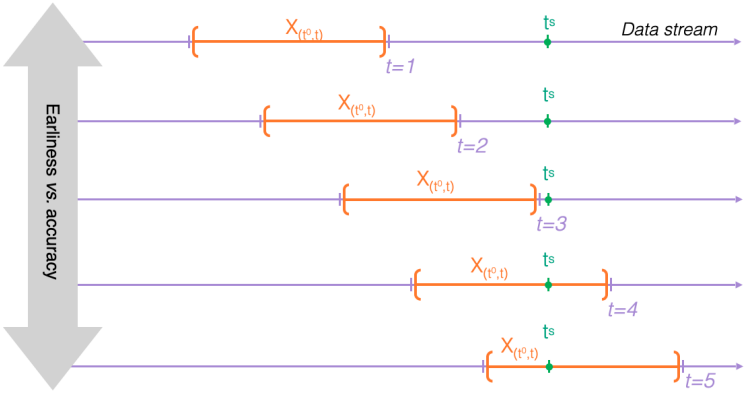


FIGURE 6.10: Illustration of the earliness vs. accuracy trade-off for online decisions

Challenge #7: management of non-stationarity in ML-EDM
It is not always realistic to assume stationarity of the data. In practice, data collected from a stream may suffer from several types of drifts: (*i*) the distribution of the measurements within the sliding window $\mathbf{x}_t$ can vary over time, this is called *covariate-shift* (Quiñonero-Candela et al., 2009); (*ii*) the prior distribution of the classes $P(y)$ can be subject to such drifts; (*iii*) and the concept to be learned $P(y|\mathbf{x})$ can also change when *concept drift* occurs (Joao Gama, 2010).

To manage these non-stationarities, a first family of approaches maintains a decision model trained using a sliding window of most recent examples. This is a blind approach, in the sense that there is no explicit drift detection. The main problem is deciding the appropriate window size.

A second family of approaches, explicitly *detects* the drifts (João Gama et al., 2014; Lemaire, Salperwyck, and Bondu, 2014) and triggers actions when necessary, such as re-training the model from scratch, or using a collection of models in the case of ensembles. In this case, detecting concept drift can be considered as similar to the anomaly detection problem, and ML-EDM approaches could be used to tackle it in future work. A popular idea is to train the decision model using a growing window while data is stationary, and shrink the window when a drift is detected. This kind of approaches can easily be adapted to *online* ML-EDM, since they decouple model training and non-stationarity management.

In the case of incremental concept drift, a third family of approaches consists in continuously adapting the model by training it online from recent data. This kind of *adaptive* approach is much more challenging to adapt to *online* ML-EDM. Indeed, as in ML-EDM problems (see Figure 4.3), two kinds of models are used: (*i*) the *predictive model(s)*, which can categorize the input data stream at any time ; (*ii*) the *triggering strategy* which makes the decisions at the appropriate time. The main challenge in developing adaptive drift management methods for the *online* ML-EDM problem is that the parameters of the *predictive models* and of the *triggering strategy* must be

updated jointly. These two kinds of models are highly dependent: updating the parameters of one has an impact on the optimal parameters of the other.

By contrast, in standard ML-EDM approaches which operate in batch mode, the parameters of the predictive models are first optimized, and then the parameters of the triggering strategy are optimized in turn given the parameters of the classifiers (see paragraph B in Section 6.2). This two-step Machine Learning scheme is definitely not valid for managing drift online (Krempl et al., 2014). Adaptive drift management for the *online* ML-EDM problem has not yet been addressed in the literature and constitutes an interesting research direction. In drift detection systems, there is a trade-off between fast detection and the number of false alarms. Moreover, in problems where the target (e.g. the labels) is not always available or available with a delay requires unsupervised or semi-supervised drift detection mechanisms. The ML-EDM framework, improving the compromise between earliness and accuracy, can provide new approaches for drift detection.

6.6 Revocable decisions

In many situations, one can take a decision and then decide to change it after some new pieces of information become available. The change may be burdensome but nevertheless justified because it seems likely to lead to a much better outcome. This can be the case when a doctor revises what now seems a misdiagnosis.

Similarly, ML-EDM should be *extended* to consider such a revocation mechanism. In the classical ML-EDM problem, a prediction $h(\mathbf{x}_{\hat{t}})$ cannot be changed once the decision is triggered at time $\hat{t} \leq T$. Whereas, the extension of ECTS to *revocable* decisions (see Chapter 4) allows a prediction to be modified several times before the deadline T. On the one hand, the revocation of a decision generates a higher delay cost $\mathcal{L}_{delay}$, as well as a cost of changing the decision $\mathcal{L}_{revoke}$. On the other hand, new data observed in the meantime provide information that makes the prediction more reliable, thus tending to decrease the misclassification cost $\mathcal{L}_{prediction}$. Ultimately, the main issue is to identify the appropriate decisions to revoke, in order to minimize the global cost.

Such an extension to revocable decisions could be of great interest: (*i*) in applications where the cost of changing decisions is low, i.e. the DAGs associated with each possible decision share reusable tasks (see Chapter 4) ; (*ii*) in applications involving online early decision making (see Chapter 5). There are many use cases where the need to *revoke* decisions appears clearly. For instance, the emergency stop system of an autonomous car brakes as soon as an obstacle is suspected on the highway, and releases the brake when it realizes, as it gets closer, that the suspected obstacle is a false positive (e.g. a dark spot on the road).

Developing ML-EDM approaches capable of appropriately revoking its decisions involves solving the two following challenges:

Challenge #8:
reactivity *vs.* stability dilemma for revocable decisions
The first issue is to ensure that a decision change is driven by the information provided by the recently acquired measurements, and not caused by the inability of the system to produce a stable decision over time. This problem is not trivial. On the one hand, the system needs to be reactive by changing its decision promptly when necessary. On the other hand, the system is required to provide stable decisions over time by avoiding excessively frequent and undue changes. Thus, a trade-off exists

between the *reactivity* of the system and its *stability* over time. To our knowledge, the work presented in Chapter 4 is only one that uses such a cost of decision change (Achenchabe, Bondu, Cornuéjols, and Lemaire, 2022a), in order to penalize revocation of too many decisions. The *reactivity vs. stability* dilemma of revocable decisions is understudied in the literature, and it would be interesting for the scientific community to work on this question.

Challenge #9: extending non-myopia to revocation risk
Non-myopic ML-EDM approaches are capable of estimating the information gain that will be provided by future measurements, based on the currently visible ones. In other words, these approaches are able to predict the reliability improvement of a decision in the future. Thus, a decision is triggered when the expected gain in miss-classification cost at the next time steps does not compensate the cost of delaying the decision (Chapter 3). In the case of revocable decisions, an important challenge is to estimate the future information gain by taking into account the *risk of revocation* itself. Specifically, a decision that will probably be revoked afterward should be delayed due to this risk. Conversely, a decision which promises to be sustainable should be anticipated. Designing *non-myopic to revocation risk* approaches could be an important step forward to (*i*) optimize the first trigger moment, and (*ii*) reduce the number of undue decision changes. The approach proposed in Chapter 4 constitutes a first step in this direction, by assigning a cost to decision changes and considering it in the expectation of future costs. To the best of our knowledge, this is the only approach which provides this interesting property. It is not clear whether alternative methods are possible. This is an interesting topic for further studies by the scientific community.

6.7 Origin of the decision costs

Figure 6.11 describes a binary ECTS problem, where the actions to be performed depend on the predicted class and are described by two *Directed Acyclic Graphs* (DAG). These DAGs characterize the sequence and the relationships between the unit tasks which compose them (e.g. task 1 must be completed before starting task 2). Here, the DAGs of tasks are *fixed*, they do not depend on the decision time.

The total cost of a decision can be decomposed by:

(i) the *delay* cost, denoted by $\mathcal{L}_{delay}$, which reflects the need to execute the DAG of actions corresponding to the new decision in a constrained time, and in a parallel way (already detailed in Section **??**);

(ii) the *decision* cost, which corresponds to the consequences of a bad decision, or the gains of a good decision (denoted by $\mathcal{L}_{prediction}$).

(iii) the *revocation* cost, which is the cumulative cost of the mistakenly performed tasks belonging to the DAG of previously made bad decisions, and which are not reusable for the new decision (denoted by $\mathcal{L}_{revoke}$) ;

When expressed in the same unit, these different types of costs can be summed up in order to reflect the quality of the decisions made, and their timing. Thus, Equation 6.1 becomes:

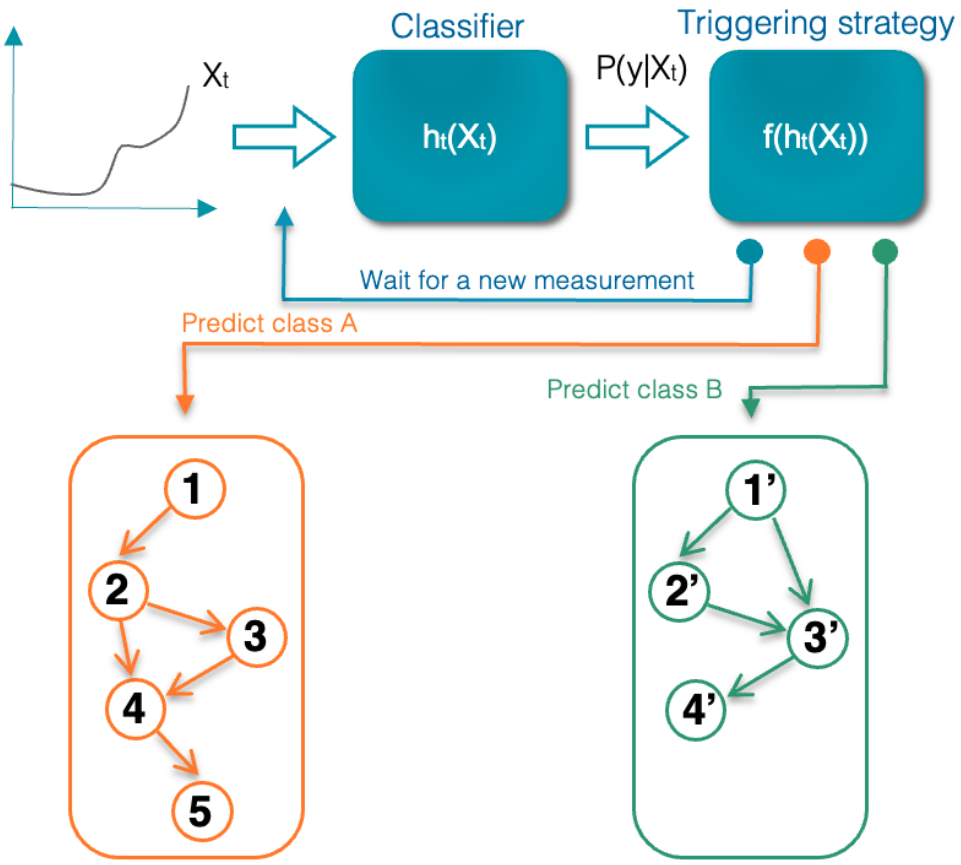


FIGURE 6.11: DAGs of tasks to be performed after the triggering of a decision.

$$\mathcal{L}(h(\mathbf{x}_t), t, y) = \overbrace{\mathcal{L}_{delay}(t)}^{(i)} + \overbrace{\mathcal{L}_{prediction}\left(h(\mathbf{x}_t), y\right)}^{(ii)} + \underbrace{\mathcal{L}_{revoke}\left(h(\mathbf{x}_t) | \{(h(\mathbf{x}_{\hat{t}_i}), \hat{t}_i)\}_{i \in [1, D_t^{\mathbf{x}}]}\right)}_{(iii)} \tag{6.15}$$

where $\{(h(\mathbf{x}_{\hat{t}_i}), \hat{t}_i)\}_{i \in [1, D_t^{\mathbf{x}}]}$ represents the sequence of the previously made decisions and their associated triggering time, with $\hat{t}_i < t, \forall i \in [1, D_t^{\mathbf{x}}]$.

Term (*ii*): Taking into account the *decision* cost is a very common feature in the literature, particularly in the field of cost-sensitve learning (Elkan, 2001). These techniques take as input a function $\mathcal{L}_{prediction}(\hat{y}|y) : \mathcal{Y} \times \mathcal{Y} \to \mathbb{R}$ which defines the cost of predicting $\hat{y}$ when the true class is y. The aim is to learn a classifier which minimizes these costs on new data.

Term (*iii*): By contrast, the study of the *revocation* cost is very limited in the literature. To our knowledge, (Achenchabe, Bondu, Cornuéjols, and Lemaire, 2022a) is the only one article article that considers this problem, and this work shows that assigning a cost to decision changes is a first lead to manage the *reactivity vs. stability dilemma*, and to design *non-myopic to revocation risk* approaches (i.e. discussed later in challenges #8 and #9). The origin of this cost can be explained in the light of the tasks to be performed once a decision is triggered (see Figure 6.11). For instance, let us consider the first decision noted by $(A, \hat{t_1})$, in which the system predicts at time $\hat{t_1}$ that the input time series belongs to the class A. This decision is then revoked in favor of a new decision $(B, \hat{t_2})$. The cost of changing this decision, denoted by $\mathcal{L}_{revoke}((B, \hat{t_2})|(A, \hat{t_1}))$, can be defined as the cost of the actions already performed between $\hat{t_1}$ and $\hat{t_2}$ which turn out to be useless for the new decision, i.e. which cannot be reused in the DAG of tasks corresponding to the new predicted class B. In order to define the costs of decision changes, it is necessary to identify the *common tasks* between the DAGs of the different classes and to evaluate their execution time. In addition, the entire sequence of the past decisions must be taken into account to identify the already completed tasks which are now useful for the achievement of the current DAG of tasks. For instance, the cost $\mathcal{L}_{revoke}((A, \hat{t_3})|\{(A, \hat{t_1}), (B, \hat{t_2})\})$ can be reduced by the tasks executed between $\hat{t_1}$ and $\hat{t_2}$, if these tasks are *not perishable*, i.e. the results are identical to those that would be obtained by re-executing these tasks at $\hat{t_3}$.

Challenge #10: scheduling strategy and time-dependent decision costs
In this thesis, the DAGs of tasks are supposed to be *fixed*, i.e. not depending on the decision time. However, a different problem could be considered (see Figure 6.12) where the DAGs of tasks are generated by a *scheduling strategy* depending on: (*i*) the decision made ; (*ii*) and the decision time. Such a scheduling strategy is useful in applications where the actions to be performed after a decision can be *adapted* to a time budget available to perform them. Two situations may occur: (*i*) ideally, a decision is triggered early enough to allow the scheduling strategy to generate a *complete* DAG of tasks which is optimal given the decision made (as in Figure 6.11) ; (*ii*) on the contrary, in the case of a too late decision, the scheduling strategy needs to build the DAG so that it can be achieved in the remaining time (e.g. by parallelizing some tasks, by changing or removing some of them). For instance, when flying an airplane, the tasks to be performed for an emergency landing are not the same as for

a normal landing, and there is a range of situations with different emergency level, and therefore corresponding to different time budgets.

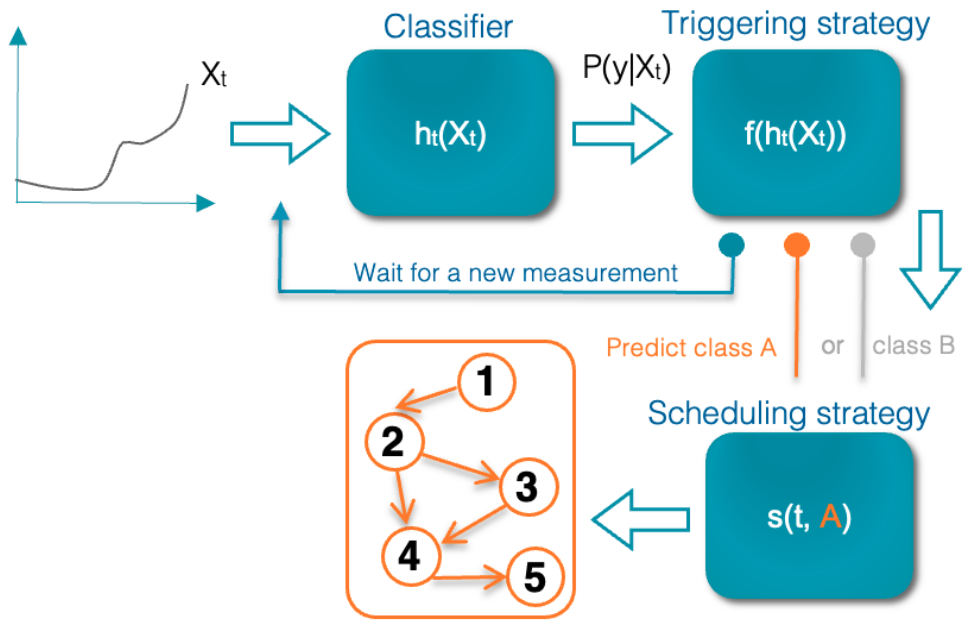


FIGURE 6.12: DAG of tasks to be performed after the triggering of a decision, generated by a scheduling strategy.

Such a time-dependent scheduling strategy radically transforms the ML-EDM problem and the way it can be formulated. In particular, the *triggering* and *scheduling* strategies become mutually dependent:

1. *Decision costs depend on the generated DAG of tasks:* all the previously discussed costs result from the structure of the DAG to be performed conditionally to the decision made: (*i*) the relationships between the tasks ; (*ii*) their execution time ; (*iii*) the conditions of their reuse when they are common to several DAGs. Since the structure of the DAG to be performed now depends on the decision time, the decision costs can no longer be considered as fixed, and they are available only after scheduling.

2. *The optimal decision time depends on the cost values:* on the other hand, the triggering strategy aims to optimize the decision time based on the cost values. As described in Equation 6.11, the triggering strategy is learned by minimizing the empirical risk, which is itself estimated using a loss function based on the costs.

Finding an optimal triggering strategy when the scheduling strategy is itself time-dependent makes ML-EDM a quite difficult challenge as the scheduling strategy is only known through its interactions with the triggering strategy. In this case, Reinforcement Learning seems to be a possible option to address the problem. The scheduling strategy could then be considered as part of the environment, and a contributor to the reward signal by determining the decision costs for each decision taken at a particular time. However, this line of attack remains to be investigated in order to assess its merit.

In many applications, fortunately, the implementation of a scheduling strategy is much simpler, especially when the variation of decision costs over time is known in advance (or modeled, and thus are partially known). We will place ourselves under

this assumption in the rest of this chapter. The preceding remarks are reminders that if considered in all its complexity, ML-EDM becomes a very difficult problem. As we will see below, addressing the case where the costs are assumed to be time dependent but with a known form, already offers interesting challenges and corresponds to a variety of applications.

6.8 Overview on challenges

This section provides an overview of the previously presented challenges, indicating references which address part of these challenges (see the second column of Table 6.1), and summarizing the main prospects for applications in the short and long term (see the last column of Table 6.1). Table 6.1 organizes the proposed challenges by category, using colors to identify: (*i*) those related to changing the learning task ; (*ii*) those related to online ML-EDM ; (*iii*) and those related to revocable decisions.

6.9 Usecases

ML-EDM approaches can be applied to a wide range of applications, such as cyber security (Zoppi et al., 2021), medicine (Khoshnevisan and Chi, 2021b), surgery (Samuel and Cuzzolin, 2021). This section develops some key use cases and identifies possible advances in near future, if the proposed challenges are met.

6.9.1 Early classification of fetal heart rates

There are no precise figures on the number of deaths in childbirth due to poor oxygenation. According to the Portuguese Directorate-General for Health, the number of children who died due to hypoxia in 2013 was 192 fetuses. This is a critical example where making informed early decisions is critical, literally meaning a difference between life and death. Cardiotocography techniques are used to assess fetal well-being through continuous monitoring of fetal heart rate and uterine contractions (Luzietti et al., 1999).

Fetal well-being results from the normal functioning of the transfer of maternal blood to the placenta and, through its proper functioning, the transfer of oxygen present in maternal blood to fetal blood (P., 2015). Labor is a potentially threatening situation to fetal well-being, as strong uterine contractions stop the flow of maternal blood to the placenta, compromising fetal oxygenation (P., 2015).

Hypoxia, resulting from lack of oxygen, represents a large part of unsuccessful deliveries. In addition, more than 50% of deliveries with poor outcome are caused by failure to recognize fetal heart rate patterns (Chinnasamy, Muthusamy, and Gopal, 2013).

In this field, ML-EDM techniques could be of great help to detect the early warning signs of complications during childbirth. This application can be addressed as an ECTS problem, as a fetal heart rate signal constitutes a time series. The goal there is twofold: (*i*) classifying the birth outcome before having access to complete time-series ; (*ii*) optimizing the triggering moment, when this prediction is made. The extension of ECTS techniques to revocable decisions would be very relevant (see challenges #8 and #9) allowing for active monitoring of the children's well-being on a continuous basis, until delivery. In addition, two particular aspects need to be taken into account in developing an efficient approach: (*i*) the prediction cost $\mathcal{L}_{prediction}$ is highly asymmetrical since a false negative can mean the death of the baby or the mother; (*ii*) the deadline T which represents the moment of delivery is uncertain and varying. Thus, the deadline T corresponds to the occurrence of an event (i.e. the birth) which can be modeled as random variable as in (Frazier and Angela, 2007; Kochenderfer, 2015).

6.9.2 Digital twin in production systems

Digital Twin (DT) is an important active concept in the area of Industry 4.0. With the development of low cost sensors and efficient IoT communication facilities, almost all production systems are now equipped with several sensors enabling real time monitoring and helping in decisions about maintenance, or when failures occur. Production systems with sensors coupled with computers are also called cyber-physical systems (CBS). In this section, we consider digital twins (DT) of cyber-physical systems (CBS) which are in operation.

The main digital twin applications (Fuller et al., 2020) are related to smart cities, manufacturing, healthcare and industry. The role of the DT is thus to use the data

streams coming from the sensors of the CBS in order to constantly calibrate simulation models of different components of the system. Indeed, this offers several opportunities, namely (1) detection of anomalies when the system deviates from the simulation model ; (2) diagnostic of dysfunctions when they occur ; (3) exploration of different scenarios for system evolution in case of dysfunction ; (4) recommendation for repair actions.

These approaches were, for instance, well illustrated in the European H2020 project MAYA (see http://www.maya-euproject.com/).

Effective maintenance management methods are vital, and industries seek to minimize the number of operational failures, reduce their operational costs, and increase their productivity. In this context, The availability of large volume of data coming from sensors of a CBS makes the use of Machine Learning techniques, supervised or unsupervised, very appealing. Typical unsupervised ML approaches are related to anomaly detection (Ruff et al., 2021) where an alarm should be triggered when the behavior of the CBS differs from normal running. Typical supervised ML approaches in the context of manufacturing and industry are related to predictive maintenance (Carvalho et al., 2019; Ran et al., 2019), where classification models are used to predict categories that correspond to possible failures. Predictive Maintenance (PdM) is a data-driven approach that emerged in Industry 4.0 as a prominent field of research. It uses statistical analysis, Machine Learning (ML) models, and Deep Learning (DL) solutions for modeling complex systems behavior, identifying trends and predicting failures.

In both anomaly detection and predictive maintenance, ML models need to be updated continuously with incoming new data, and their outputs are used by operators for decision making. Consequently, DT's using machine learning models can benefit from early decisions and ML-EDM techniques. The **multiple early decision to be located in time** concept introduced in Section 6.2 is particularly relevant in this context, since maintenance operations often imply several periods of time.

We review below some challenges of the chapter in light of this domain. Challenge #1 (extending non-myopia to unsupervised approaches) is relevant, since an efficient anomaly detection system requires unsupervised approaches which can be combined with physics-based simulations of the different components. Challenge #2 (other supervised tasks) is also appropriate since both classification and regression problems appear (e.g. breakdown occurrence, prediction of energy consumption).

Regarding challenge #3 (weakly supervised learning), the main problem is not that training data is of poor quality but that interesting data (e.g. failures) are often rare, leading to the need for data augmentation using simulation. Of course Challenge #4 (data type agnostic) is especially relevant for DT's, since a system is always composed of several heterogeneous components with many sensors generating data of heterogeneous type, for instance tabular data, multivariate time series, images, audio signals such vibrations, possibly videos, ...). In this situation, the update of one component or one or several sensors would be much easier and cheaper if ML-EDM were data type agnostic. DT's operating at a system level leads to complex prediction models and complex decisions since the different components operate differently but in interaction (cf. challenge #5). The ability to manage non-stationarity (cf. challenge #7) is obviously central in DT's: aging and wearing of equipment lead to covariate and concept drifts which must be taken into account. Moreover, prediction models may need to be recalibrated after maintenance operations.

Scenario exploration in manufacturing digital twins

As mentioned above, one interesting characteristic of Digital Twins (DT's) is their ability to anticipate possible evolution of the system using current observation and running simulation models. For instance in (Gabor et al., 2016), a software architecture framework is defined which enables information exchanges between the cyber-physical system and simulation models which are part of the DT. This enables simulations of possible evolution of the CBS if such simulations can be operated in real-time. In (Lugaresi and Matta, 2018), a review of Real-Time Simulation (RTS) models is proposed and describes several approaches for RTS, usually based on Discrete Event Simulation (DES). As reported in this chapter, there are still research challenges in RTS for manufacturing (data management, adaptability, model generation, validation, reactiveness) but some solutions exist. In (Lugaresi, Travaglini, and Matta, 2019), a LEGO toy demonstrator has been developed to prove this concept.

Consequently, one can consider that in manufacturing, Digital Twins will soon have the ability to run in real-time simulations of the evolution of the system, based on Discrete Event Simulation models. These simulations provide different possible outcomes of the system with associated probabilities: typically what an ML-EDM expects as input in a *non-myopic* decision perspective. This case is closely related to the problems addressed in challenge #10 (scheduling strategy and time-dependent decision costs), since the time needed to run simulations is similar to the time budget of a scheduling strategy (cf. Figure 6.12).

6.9.3 Predictive Maintenance of Metro Trains

Advances in networking, machine learning, data analytics, and robotics are allowing vast improvements on industrial processes. Predictive maintenance, in particular, is one technique with high impact in today's industry. Over the years different maintenance strategies have been developed. Three main approaches can be identified: (*i*) *Corrective* maintenance: when an equipment is run until failure. This simplest of method almost always leads to high (unexpected) downtime and thus potentially to critical situations that entail great costs for companies. (*ii*) *Preventive* maintenance: It is based on planning regular replacement of components and/or equipment. Historical failure data and/or the data provided by the equipment manufacturer is used. Although this approach prevents unexpected shutdown, it usually entails unnecessary additional costs and an increased unexploited lifetime. (*iii*) *Predictive* maintenance: It uses direct monitoring of the mechanical condition and other parameters that can determine the operating conditions over time in order to accurately predict the arrival of a breakdown. There are now tools that can process real-time data acquired from different equipment parts to predict any sign of failure.

Data-driven predictive maintenance (PdM) monitors the mechanical conditions or other health indicators of the equipment, and uses advanced statistical or Machine Learning methods to detect operating patterns and dynamically identify operating conditions.

Predictive maintenance of metro trains offers a lot of opportunities for ML-EDM to improve the quality of service of public transportation. Clearly, anomaly detection can be addressed using unsupervised techniques, which in turn implies challenge #1. The data collected on the operations of metro trains can be of various types: multi-variate series, feedback from the drivers or the maintenance agents, and so on, which relate to challenge #4. Of course, challenges #5 and #6 on online predictions with location in time are implied as well. Finally, even though once an alert has been

raised and predictive operations have been scheduled, it is rare that the decisions can be revoked, this is nonetheless an issue that can be considered when further data may lead to a reassessment of the situation (see challenge #8 and #9).

6.9.4 Social networks: societal and psychological risks

Online social networking platforms are more popular than ever. They are now used daily and have become an important part of our lives. They radically transform the way we communicate with each other. However, this transformation comes with many problems on both sides, for users and platforms alike.

For example, *Fake news* spread widely during the covid pandemic, which had an impact on the spread of the virus itself. (Ajao, Bhowmik, and Zargari, 2019) tackled this problem as a binary classification problem where classes are “fake" and “real" news, by using fact checking techniques. *Fake accounts* are also considered a major problem for these platforms, as they are among the main culprits in spreading false information. For instance, (Aydin, Mehmet, and Salur, 2018; Y. Elyusufi, Z. Elyusufi, et al., 2019; Fire et al., 2014; N. Singh et al., 2018) use Machine Learning techniques to detect these fake account based on interactions between users. Fake accounts can also be used for harassment and propagating hate speech, which can induce major psychological risks (Watanabe, Bouazizi, and Ohtsuki, 2018). The detection of depression and risk of suicide has been addressed using Machine Learning techniques in (Castillo-Sánchez et al., 2020; Islam et al., 2018).

Decisions taken by Machine Learning models to prevent such societal and psychological risks on social networks are clearly *time-sensitive*:

- Fake news must be detected as early as possible to limit its spread, and thus its harmful consequences on society. For example, (X. Zhou et al., 2020) focuses on early detection of fake news from the press before it is expressed on social media, and (Yang Liu and Wu, 2020) proposes a deep learning model that achieves an accuracy of more than 90% within 5 minutes of the beginning of the propagation of a fake news and before it is retweeted 50 times.
- The early detection of *fake users* has also been studied in recent work. For example, (Breuer, Eilat, and Weinsberg, 2020) proposes a graph-based approach which uses network connectivity to detect fake users as early as possible.
- Likewise, Detecting as early as possible *depressed* or potential suicidal users is very critical for prevention. This problem has also been addressed under the perspective of early classification in (Leiva and Freire, 2017).

The development of the ML-EDM domain is an opportunity to go further in these applications. In particular, it would be very useful to develop unsupervised and weakly supervised ML-EDM approaches (see challenges #1 and #3). In this application area, ground truth is often unavailable or corrupted. Typically, users rarely declare on social networks their depressive state or their suicidal thoughts ; and if they do, this information is not reliable due to obvious social biases. In the case of social networks, Training data is very complex and consists of multiple sources: streams of texts, a large graph evolving over time etc. Therefore, it would be particularly beneficial to develop ML-EDM approaches which are agnostic to data types (see challenge #4). In the scenario where the user’s state is monitored in a streaming fashion, and where the goal is to identify their state changes as soon as possible, the development of online ML-EDM approaches seems to be very desirable (see challenges #8 and #9).

6.9.5 Autonomous vehicle

An autonomous vehicle is defined in (Thrun, 2010) as capable of sensing its environment and navigating safely without human input.

In order to achieve this ambitious goal, it is necessary to combine advanced technologies from many fields such as: (*i*) electronics and sensors ; (*ii*) software engineering ; (*iii*) telecommunication ; (*iv*) computer security ; (*v*) information processing.

Five levels of vehicle automation have been defined (Milakis, Van Arem, and Van Wee, 2017) as intermediate goals toward full automation: in *level 1*, most functions are controlled by the driver ; in *level 2*, at least one driver assistance system is implemented ; in *level 3*, the driver is able to delegate safety critical functions to the vehicle ; in *level 4*, the vehicle is fully autonomous, but not in all driving scenarios ; in *level 5*, the vehicle is fully autonomous, with performance equal to that of a human driver in all driving scenarios. The development of a fully autonomous vehicle (levels 4 or 5) requires a complex software architecture, which operates numerous functional components (Serban, Poll, and Visser, 2018). More precisely, three classes of components have been identified, corresponding to different levels of control:

i) *Operational* components, which implement basic vehicle control such that lateral and longitudinal vehicle motion, monitoring of the driving environment by detecting objects and events, identification of the vehicle's condition and its position in the environment ;

ii) *Tactical* components, which plan and execute vehicle maneuvers and prepare appropriate responses to incoming events, e.g. trajectory control, lane change, obstacle avoidance, emergency braking, visibility improvement by adapting lighting to environmental conditions;

iii) *Strategic* components, which determine the general itinerary according to the driver's preferences and the traffic conditions, based on a maps database and a path planning algorithm.

Given the high complexity of the tasks to be automated, *Machine Learning* approaches have become an essential element in the design of autonomous vehicles (Ma et al., 2020). Machine learning is therefore used in the development of different classes of components:

i) *Operational* components are the most developed in the literature, and can be classified as follows: (1) *mediated perception* approaches are mostly based on deep learning techniques (Fagnant and Kockelman, 2015; John et al., 2015) and aim to detect a wide variety of objects, such as obstacles, road signs, lanes and traffic lights ; (2) *direct perception* (Bojarski, Del Testa, et al., 2016; Bojarski, Yeres, et al., 2017) consists of end-to-end approaches, which aim to directly manage vehicle controls (e.g., gas pedal, brake, steering wheel) without explicitly dealing with location and mapping ; (3) *localization* approaches (Alcantarilla et al., 2018; Vishnukumar et al., 2017) aim to characterize similarities and discrepancies between the environment observed from sensor data, and a priori maps, in order to accurately locate the vehicle and identify potential obstacles.

ii) *Tactical* components are mostly developed to automate vehicle maneuvers using Machine Learning techniques, such as : (1) advanced scenarios of *automated parking*, as free space recognition, pedestrian detection during the operation (Heimberger et al., 2017) ; (2) *car-following* improvement by predicting the

trajectories of other human-drivers (Gong and Du, 2018) taking into account road conditions to increase safety (L. Li, Ota, and Dong, 2018) ; (3) *trajectory planning* including obstacle avoidance [27], self-driving in urban environment (Sales et al., 2014) and at high speed (Al-Hasan and Vachtsevanos, 2002), sliding control improvement (Akermi, Chouraqui, and Boudaa, 2020).

An autonomous vehicle is an extremely complex system, which must react safely, and in real time, to the vagaries of its environment. The decisions made by the Machine Learning based components (presented above) are definitely *time-sensitive*. In this case, reaching a good compromise between *earliness* and *accuracy* of these decisions is critical. On the one hand, early detection of an obstacle facilitates the planning of a safe evasive trajectory and perceived as such by the passengers. On the other hand, false positives can be generated by too early and not enough accurate detections, causing unnecessary or even dangerous trajectory changes.

Considering the *earliness vs. accuracy* compromise is an emerging and important issue in research for autonomous vehicles. In particular, *cooperative perception* (Kim et al., 2015) has been developed to extend the perceptual range of a connected autonomous vehicle, by sharing real-time information with other surrounding vehicles.

In this scenario, the vehicles' situational awareness is improved, allowing for smoother and safer maneuvers, such as early lane changes or early emergency braking. In some ways, cooperative perception improves both the *earliness* and *accuracy* of decisions by extending the vehicles' perceptual capability.

The development of the ML-EDM field could make it easier to design fully autonomous vehicles. Indeed, the ability of these approaches to make *non-myopic* decisions is an advantage to make self-driving more fluid and safe.

An experienced driver is capable of anticipating what is going to happen on the road: he is able to build a mental picture of the situations that are likely to occur in a few seconds. A non-myopic ML-EDM approach would be able to identify probable continuations of an observed situation on the road, based on related situations encountered in the training data and their continuations(e.g., if a balloon crosses the road, it is probable that a child will run behind).

Most of the challenges presented in this chapter are relevant for the autonomous vehicle. Indeed, training data is very complex in this case, and consists of multiple sources: video, radar, lidar, sensors etc. In addition, training data may change over time, for example with the arrival of a new types of sensors on a next generation of car. It is therefore important to develop ML-EDM approaches which are *agnostic* to data types (see challenge #4). In the case of self-driving, Ground truth contained in the training data includes both the actions to be performed (e.g., emergency braking) and the timing of these actions (e.g., triggering this action 3 seconds before the potential impact). It would be very useful and probably possible to develop ML-EDM methods which learn the evolution of *decision costs* over time, underlying self-driving (see challenge #10). In the case of autonomous vehicles, sensor data is continuously observed, so it is essential to design *online* ML-EDM approaches (see challenges #5 and #6), and driving actions must definitely be *revocable* (see challenges #8 and #9). For example, an emergency brake must not be carried out to the end, if there are finally no obstacles in the way.

6.10 Perspectives and future work

This chapter aims to define the field of ML-EDM, and proposes ten challenges to the scientific community to further research in this area. In particular, ML-EDM has

been defined and positioned with respect to related fields, such as machine learning and reinforcement learning. Three challenges have been presented in relation to the learning task at hand: extending ML-EDM to unsupervised learning (challenge #1), to regression tasks (challenge #2) and to weakly-supervised learning (challenge #3). The development of data type agnostic ML-EDM approaches has been singled out as an important direction of research to extend the domains of application, yielding challenge #4. Extending ML-EDM to the online scenario has also been recognized as important too which raises three challenges (challenge #5, #6 and #7). Being able to revoke decisions properly is significant as well in many applications and raises two challenges (#8 and #9). The origin of the different costs involved in the optimization of decision times has been discussed, leading to a last challenge (challenge #10) to extend the ML-EDM problem to cases where these costs vary over time. Finally, a range of application areas for which ML-EMD could lead to significant progress in the near future have been described, such as anomaly detection, predictive maintenance, patient health monitoring, self-driving vehicles.

The overall objective of this chapter is to define a new field of investigation, and to propose research avenues in order to generate interest from the scientific community.

ML-EDM challenges	SOTA	Main application perspectives
#1 (Section 6.3) Extending non-myopia to unsupervised approaches		In anomaly detection applications, anticipate the deviation of an observed individual from a normal behavior.
#2 (Section 6.3) Addressing other supervised learning tasks		- Adapt ECTS approaches to extrinsic regression problems. - Develop forecasting methods whose prediction horizon can adapt.
#3 (Section 6.3) Early weakly supervised learning (WSL)		Adapt ECTS approaches to the different WSL classification scenarios.
#4 (Section 6.4) Data type agnostic ML-EDM	(Dachraoui, Bondu, and Cornuéjols, 2015) (Achenchabe, Bondu, and Cornuéjols, 2021) (Mori, Mendiburu, et al., 2015; Mori, Alexander Mendiburu, Dasgupta, et al., 2017)	- Identify agnostic approaches in the literature and promote this feature. - Define a pivotal format allowing to develop an ML-EDM library.
#5 (Section 6.5) Online predictions to be located in time		Applications where the arrival of an event (e.g. a failure) must be predicted in advance, as well as its duration.
#6 (Section 6.5) Online accuracy vs. earliness trade-off	(Achenchabe, Bondu, Cornuéjols, and Lemaire, 2022b)	Optimize decision time in online predictive maintenance applications.
#7 (Section 6.5) Management of non-stationarity in ML-EDM		Properly manage the potentially long life of ML-EDM models.
#8 (Section 6.6) Reactivity vs. stability dilemma for revocable decisions	(Achenchabe, Bondu, Cornuéjols, and Lemaire, 2022a)	Applications where undue and excessive decision changes must be avoided.
#9 (Section 6.6) Non-myopia to revocation risk	(Achenchabe, Bondu, Cornuéjols, and Lemaire, 2022a)	Applications where it is necessary to delay decisions which are likely to be changed later.
#10 (Section ??) Scheduling strategy and time-dependent decision costs		- Applications where the variation of the decision costs over time is known or can be modeled. - Applications where the scheduling strategy is only known through its interactions with the triggering strategy.

TABLE 6.1: Overview of the proposed challenges by category: in blue those related to the *learning task*, in green those related to *online* ML-EDM, in yellow those related to *revoking decisions*, and in white the others.

Chapter 7

Conclusion

7.1 Summary

The point of entry in this thesis was the *early classification of time series* (ECTS) problem, studied in Chapter 3 where two antagonist concepts are at stake. In time-critical applications, on the one hand, the sooner the time series is classified, the more rewarding it is. On the other hand, an early classification is more likely to be inaccurate. This is called the *earliness* vs *accuracy* dilemma. However, this trade-off has been formalized for the specific case of ECTS, where time series are of fixed-length T, with a single irrevocable decision, in a supervised learning setting where labeled training data assigns a single label for each time series. The decision irrevocability limitation has been addressed in Chapter 4. In many applications, namely, monitoring in predictive maintenance or medical applications, measurements of an individual come in a streaming fashion, and time series are of indefinite undetermined length, with portions of variable size with potentially different labels. This scenario has been addressed in Chapter 5. Finally, in Chapter 6, we proposed a more generic problem that we called *Machine Learning based Early decision-making* (ML-EDM). It extends the ECTS problem to more challenging decision-making problems. A list of 10 challenges is proposed to the scientific community for further research.

7.2 Identified challenges

At this thesis's last semester, a particular effort has been made to identify the main challenges the community needs to overcome to open the path to more applications. In the following, a list of the proposed challenges is given in Table 7.1.

Some of them have already been addressed in this thesis, without exhaustively exploring these questions. Challenge #4 was addressed in Chapter 3, as the proposed approach is data type agnostic. Challenge #5, and #6 were addressed in Chapter 5 as the individual considered is a fixed timestamp; however, it would be interesting to solve this problem while considering chunks of time series as described in the definition of ML-EDM in Chapter 6. Challenges #8 and #9 were addressed for time series of fixed-length and a single true label. Adapting the work done in Chapter 5 to the revocable setting is trivial.

The rest of the challenges were not addressed in this thesis, and are left to the community as research directions.

ML-EDM challenges	Addressed
#1: Extending non-myopia to unsupervised approaches	
#2: Addressing other supervised learning tasks	
#3: Early weakly supervised learning (WSL)	
#4: Data type agnostic ML-EDM	Chapter 3
#5: Online predictions to be located in time	Chapter 5
#6: Online accuracy vs. earliness trade-off	Chapter 5
#7: Management of non-stationarity in ML-EDM	
#8: Reactivity vs. stability dilemma for revocable decisions	Chapter 4
#9: Non-myopia to revocation risk	Chapter 4
#10: Scheduling strategy and time-dependent decision costs	

TABLE 7.1: Overview of the challenges identified

7.3 Contributions

More specifically, here is a summary of the contributions of this thesis.

In chapter 3 the problem of early classification of time series is studied in depth, the main contribution of this chapter is presenting a general framework for the ECTS problem and implementing three novel methods. Extensive experiments on a benchmark of 45 datasets from different domain areas show the superiority of the proposed method ECONOMY-γ over the state of the art of the field. The second contribution consists in claiming that the cost of delay as well as the cost of misclassification depends on the application area and must be specified in advance by the user, and a new evaluation criterion based on these costs is proposed.

In chapter 4 our contribution is to extend the classical ECTS problem by overcoming one of its limitations, which is decisions are irrevocable. We rigorously define the early and revocable time series classification problem, introduce a the cost of decision change, the origin of the costs introduced in this thesis are discussed as well. We propose a new framework which models the risk of revocation, and is capable of revoking decisions already taken at the optimal moment based on new acquired measurements. Two novel algorithms are proposed, experiments show that these two algorithms have superior performance than the irrevocable regime, which confirms the utility of the framework.

In chapter 5 In the same spirit of chapter 4, two limitations of ECTS are tackled: i) Labels are associated with the full-length time series; ii) time series are of finite length. Our contribution lies in defining this problem and proposing a new algorithm which deals with these limitations, the earliness-accuracy trade-off is also transcribed from ECTS to this new problem that we call ECOTS. Numerical results on a predictive maintenance dataset show promising result in favor of the new approach compared to a classical approach.

In chapter 6 our contribution is to push even more the limits of ECTS towards a generic problem that we call ML-EDM where a Machine Learning model can be applied on data acquired over time, and where the trade-off between the earliness and the accuracy of decisions must be optimized in different scenarios. challenges and use cases are proposed to the community for further research.

7.4 Limitations and Future work

Chapter 6 as a whole aims to propose new research directions for the scientific community for future research in this area. In this section, key ideas will be given to tackling the challenges detailed in the cited chapter.

ML-EDM approaches are limited to supervised classification problems. It is a critical limitation that prevents these approaches from being applied to many applications where labels are very costly to obtain, or when they are available, but too noisy to deal with using classical supervised learning approaches. Three challenges have been proposed to extend these approaches to other learning tasks: (challenge #1) to unsupervised learning, (challenge #2) to regression tasks and (challenge #3) to weakly-supervised learning.

Let us focus on challenge #1, in order to extend the ECONOMY 's non-myopic property to the unsupervised setting, the first issue to overcome is the collection of classifiers which is trained in a supervised way. An unsupervised algorithm should replace this collection. Then, ECONOMY approach is cost-sensitive, the expected misclassification cost for future time steps is based on confusion matrices, which can not be computed without ground truth labels. The new approach should be capable of estimating the expected misclassification cost of deciding in the future time steps in a fully unsupervised way.

Extending ML-EDM approach to regression tasks can have practical implications. Namely, the prediction of the daily energy consumption of a house, based on historical consumption, temperature and humidity measurements. This consists of predicting the value of the numerical target variable as soon as possible, while ensuring proper reliability. A simple approach to do this extension, is considering the regression task as a classification task using space discretization. This is the simplest way to deal with this problem. However, the user must choose the number of splits in the output space.

The extension of ML-EDM to weakly-supervised learning (WSL) is an interesting challenge, as it would allow to better address applications where the ground truth has corruptions or is incomplete (which includes semi-supervised learning). A simple naive idea is to apply WSL algorithms on the classification model and the triggering model.

Then, the second set of challenges is related to online ML-EDM have been proposed. The aim of Challenge #5 is to develop algorithms that can trigger decisions located in the future or the past. For example, predict the state of the machine as soon as possible during the following week. Challenge #6 proposes to deal with the compromise between accuracy and earliness in other scenarios, and not only in the setting of ECTS. Challenge #7 suggests dealing with the non-stationarity of streaming environments, which significantly benefits deploying ML-EDM models into production.

Chapter 5 proposes a first step into dealing with Challenge #5, and #6. Predictions are located in time, made for the past or future time stamps, while optimizing online the earliness-accuracy trade-off in a new scenario, where time series are open (with no predefined length) and multiple labels, each for a different portion of time. However, each decision is associated with a timestamp, and decisions are considered independent, which is a naive assumption since decisions are time-dependent. This work can be extended for the scenario where the classifier's predictions are associated with a period of time instead of a single time stamp. It would require more user parameters to be chosen. For example, one could imagine a discretization of the time axis in different portions for which the user defines the length. Then the ML-EDM

approach predicts the class label for each portion while optimizing the earliness-accuracy trade-off. The work presented in Chapter 5 corresponds to the case where all portion lengths are equal to 1.

Challenge #7 has not been dealt with in this thesis. However, it is essential in order to deploy ML-EDM into production. Streaming environments evolve over time. Consequently, this impacts ML-EDM approaches because they become obsolete over time in the presence of concept drift. Literature that deals with non-stationary environments can be applied to ML-EDM approaches. Multiple questions can be raised: i) How to update the classification model and the triggering model? ii) When to update these models?

Challenge #10 is the hardest one to deal with since it changes how the ML-EDM problem was formulated in chapter 6. Costs in this new setting are generated by a scheduling strategy, they depend on the decision made as well as its timing. Future work should be conducted to explore this research direction using reinforcement learning techniques as a starting point.

Appendix A

Appendix of Chapter 2

In the following, two widely used similarity measures are defined.

Definition A.0.1. ***Euclidean distance***
Given two time series $\mathbf{x}_t$ *and* $\mathbf{y}_t$*, the euclidean distance between them is:*

$$d(\mathbf{x}_t, \mathbf{y}_t) = \sqrt{\sum_{i=1}^{t} (x_i - y_i)^2}$$

The *euclidean distance* is the most intuitive similarity measure. However, if we try to use it naively between two raw time series, we may get very unintuitive results. The reason is that this distance is very sensitive to distortions. They should be removed from data for a more meaningful comparison. Multiple techniques were developed to deal with this problem, *offset translation* which consists of subtracting from the time series its mean before computing the distance, *amplitude scaling* which is a normalization technique that divides by the standard deviation of the time series after subtracting its mean, *removing linear trend*, which consists of fitting the best straight line to the time series, then subtract that line from the time series. *Smoothing* which consists of removing noise by averaging each data point value with its neighbors.

Another well-known similarity measure for time series is **Dynamic Time Warping (DTW)**. It was first introduced in (Bellman and Kalaba, 1959) and then extensively developed for many application areas, including handwriting and online signature matching, sign language recognition, gestures recognition, data mining, time series clustering, and many other areas.

As explained in (Senin, 2008), the DTW algorithm is extremely efficient in minimizing the effects of shifting and distortion in time that we have already discussed. It allows elastic transformation of time series to detect similar shapes with different phases. Given two time series $\mathbf{x}_m = \langle x_1, \ldots, x_t \rangle$ and $\mathbf{y}_n = \langle y_1, \ldots, y_t \rangle$ represented by the sequences of values, DTW yields optimal solution in the O(m.n) time. However, data sequences should be sampled at equidistant points in time (but this problem can be resolved by re-sampling).

Figure A.1 shows the difference in matching between euclidean distance and dynamic time warping, this elastic matching allows one to compute the distance between portions with different phases.

Now the question is how to compute DTW. The procedure is straightforward. However, it is heavily costly compared to Euclidean distance. First a distance matrix of size $m \times n$ should be computed between $\mathbf{x}_m$ and $\mathbf{y}_n$. The algorithm in Figure A.2 shows from line 5 to 7 the initialization of the first column. From lines 8 to 10 the initialization of the first row,$c(i, j)$ in this simple case is the is the absolute value of the

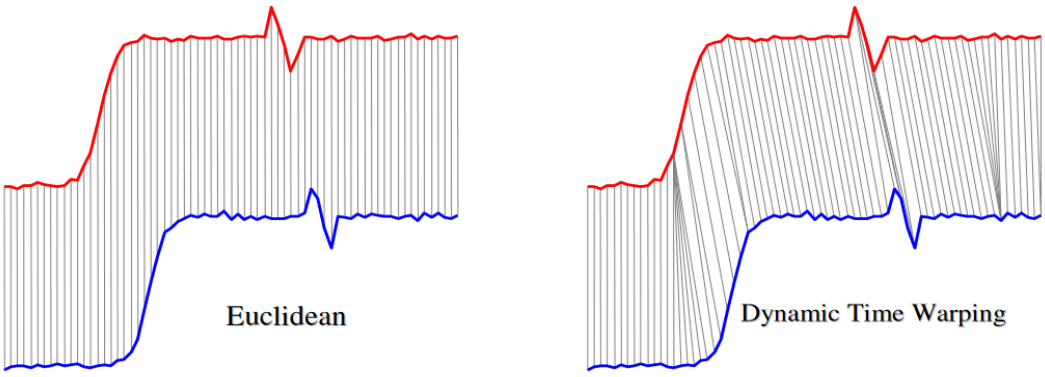


FIGURE A.1: Euclidean distance and DTW matching on the same couple of time series (taken from (Eamonn, 2006))

difference between the x_i and y_j added to the last distance computed. In the general case as shown from lines 11 to 14, the value computed is the distance between the two points added to the minimum of 3 cases: i) moving only in the time series **x**; ii) moving only in the time series **y**; iii) moving in both. This procedure is applied until the matrix is fully computed, then the warping path (i.e., the best matching distance between **x** and **y**) is computed according to the procedure described in Figure A.3. In simple terms, the procedure starts at the end of both time series which is position (m, n), then it looks backward for the minimum distance until it reaches position $(0, 0)$.

Algorithm AccumulatedCostMatrix(X, Y, C)

1: n $\leftarrow |X|$
2: m $\leftarrow |Y|$
3: $dtw[] \leftarrow new\ [n \times m]$
4: $dtw(0,0) \leftarrow 0$
5: **for** $i = 1;\ i \leq n;\ j++$ **do**
6: $\quad dtw(i,1) \leftarrow dtw(i-1,1) + c(i,1)$
7: **end for**
8: **for** $j = 1;\ j \leq m;\ j++$ **do**
9: $\quad dtw(1,j) \leftarrow dtw(1,j-1) + c(1,j)$
10: **end for**
11: **for** $i = 1; i \leq n; j++$ **do**
12: $\quad$ **for** $j = 1; j \leq m; j++$ **do**
13: $\quad\quad dtw(i,j) \leftarrow c(i,j) + \min\{dtw(i-1,j);\ dtw(i,j-1);\ dtw(i-1,j-1)\}$
14: $\quad$ **end for**
15: **end for**
16: **return** dtw

FIGURE A.2: Distance matrix computation (Senin, 2008)

Algorithm OPTIMALWARPINGPATH(dtw)

```
path[] ← new array
i = rows(dtw)
j = columns(dtw)
while (i > 1) & (j > 1) do
  if i == 1 then
    j = j − 1
  else if j == 1 then
    i = i − 1
  else
    if dtw(i−1, j) == min{dtw(i − 1, j); dtw(i, j − 1); dtw(i − 1, j − 1)}
    then
      i = i − 1
    else if dtw(i, j−1) == min{dtw(i − 1, j); dtw(i, j − 1); dtw(i − 1, j − 1)}
    then
      j = j − 1
    else
      i = i − 1; j = j − 1
    end if
    path.add((i, j))
  end if
end while
return path
```

FIGURE A.3: DTW path from (Senin, 2008)

Appendix B

Appendix of chapter 3

B.1 Distribution of decision moments and post optimal decision moments for all values of α

B.1.1 Economy-γ

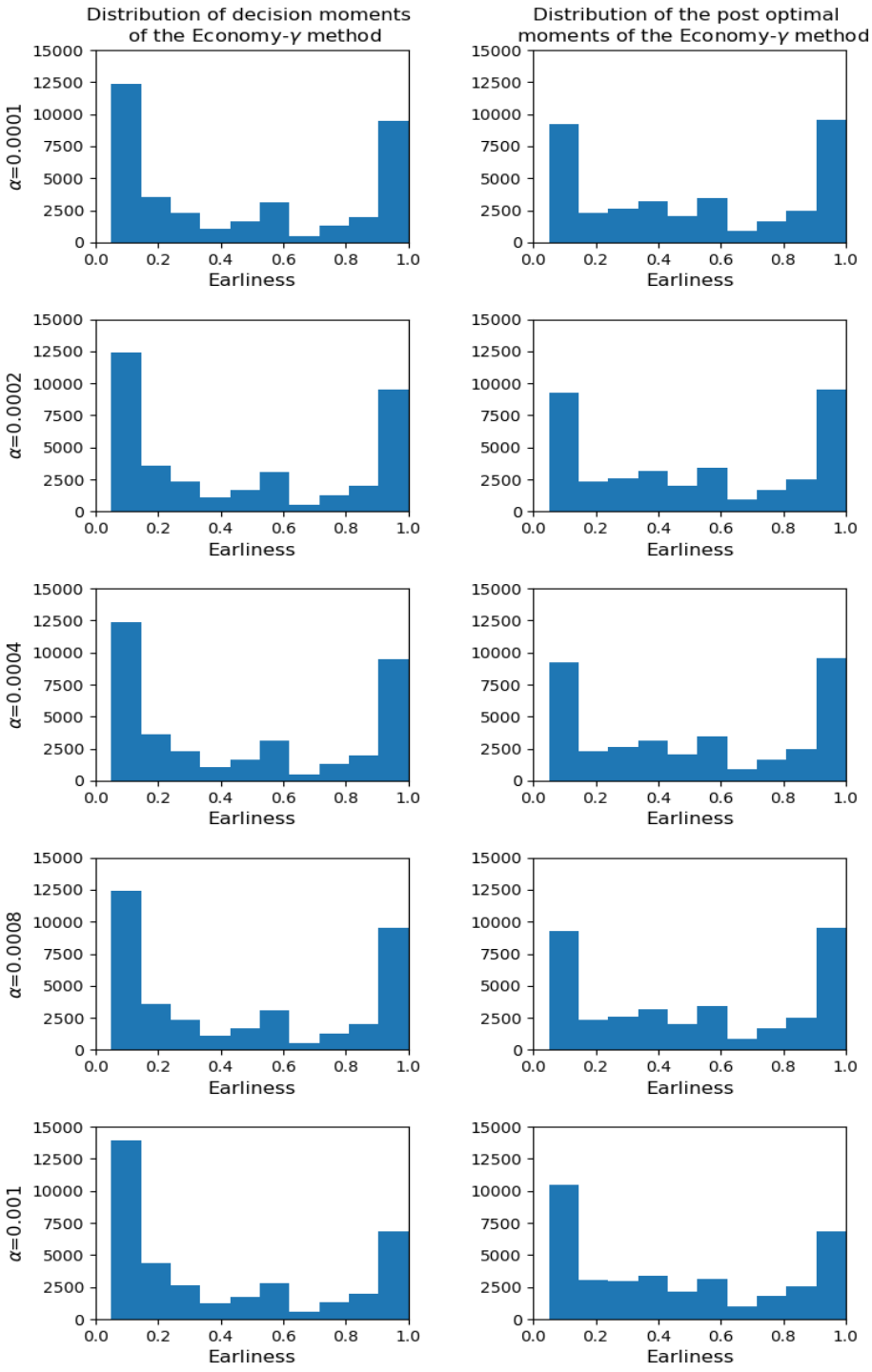


FIGURE B.1: The distribution of decision moments (left column) and post optimal moments (right column) for ECONOMY-γ with $\alpha \in \{0.0001, 0.0002, 0.0004, 0.0008, 0.001\}$

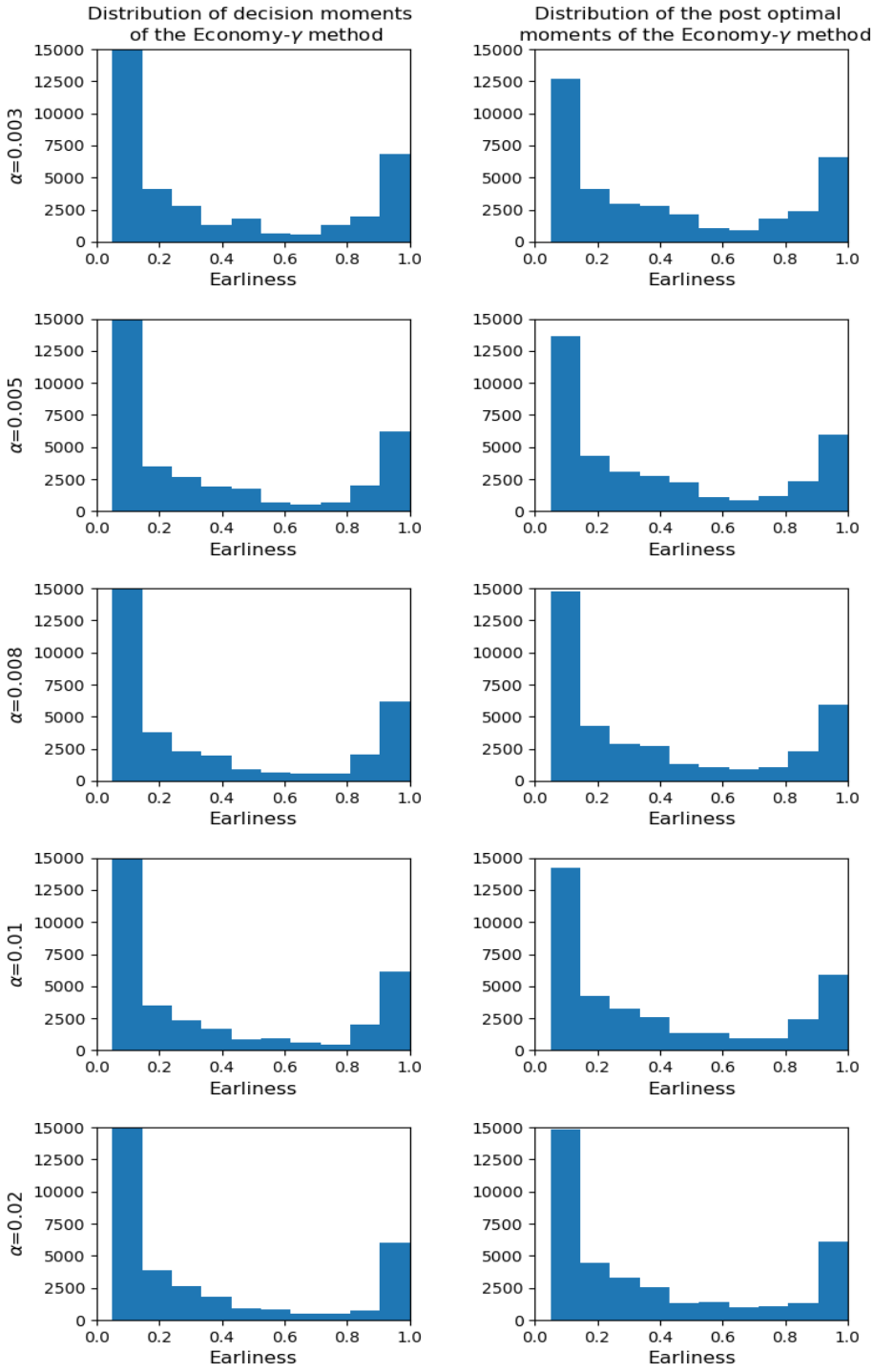


FIGURE B.2: The distribution of decision moments (left column) and post optimal moments (right column) for ECONOMY-γ with $\alpha \in \{0.003, 0.005, 0.008, 0.01, 0.02\}$

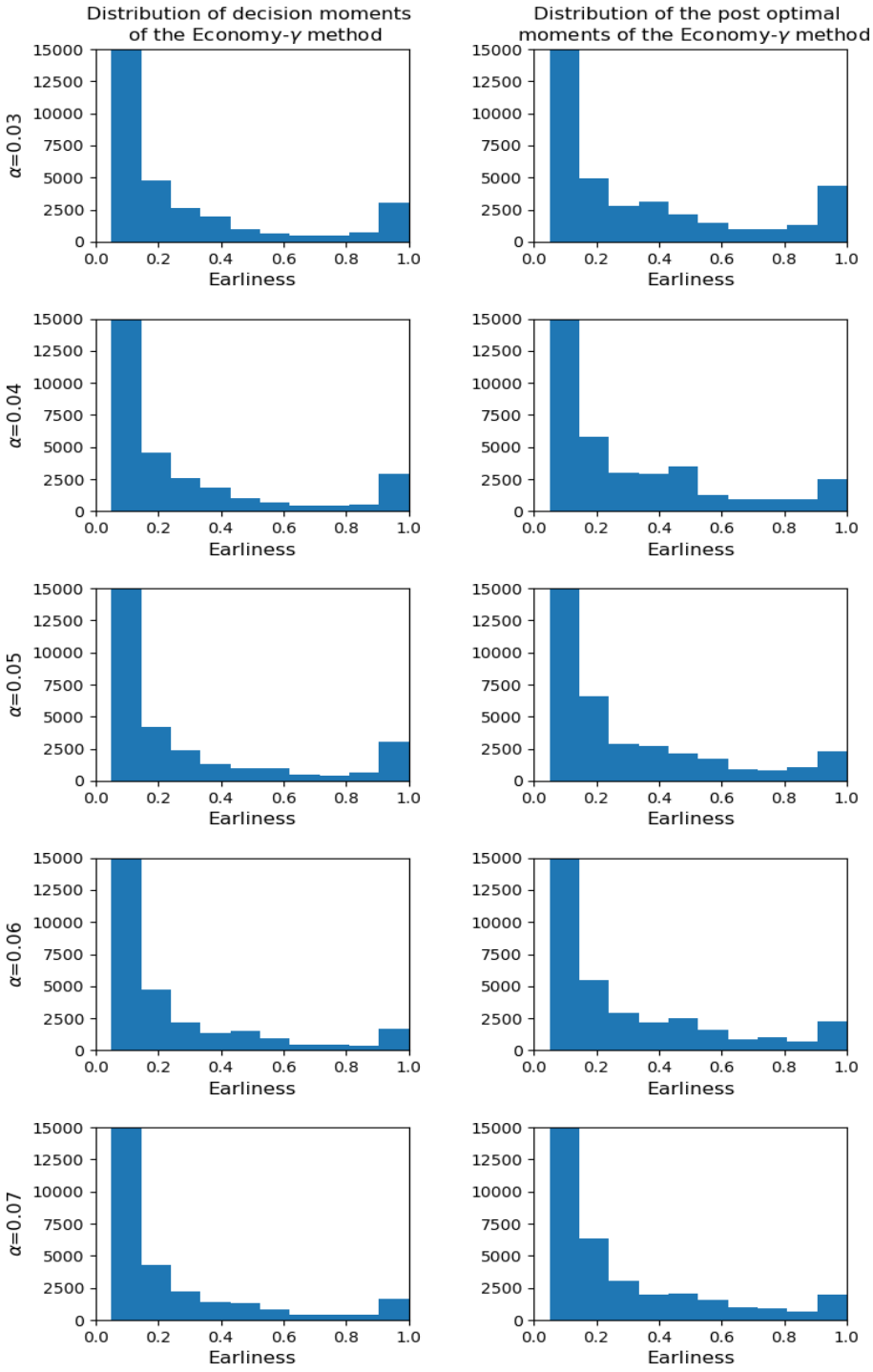


FIGURE B.3: The distribution of decision moments (left column) and post optimal moments (right column) for ECONOMY-γ with $\alpha \in \{0.03, 0.04, 0.05, 0.06, 0.07\}$

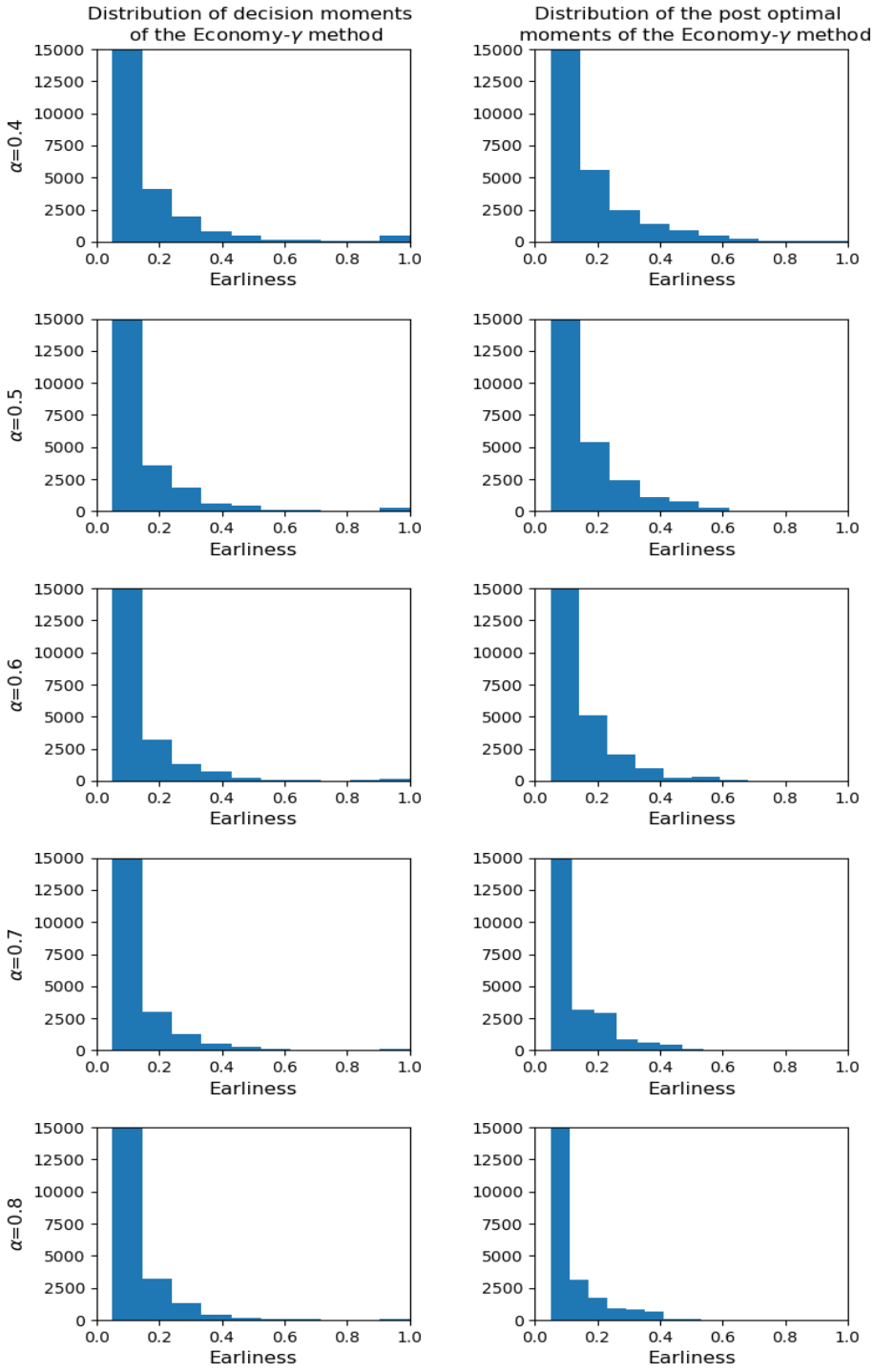


FIGURE B.4: The distribution of decision moments (left column) and post optimal moments (right column) for ECONOMY-γ with $\alpha \in \{0.4, 0.5, 0.6, 0.7, 0.8\}$

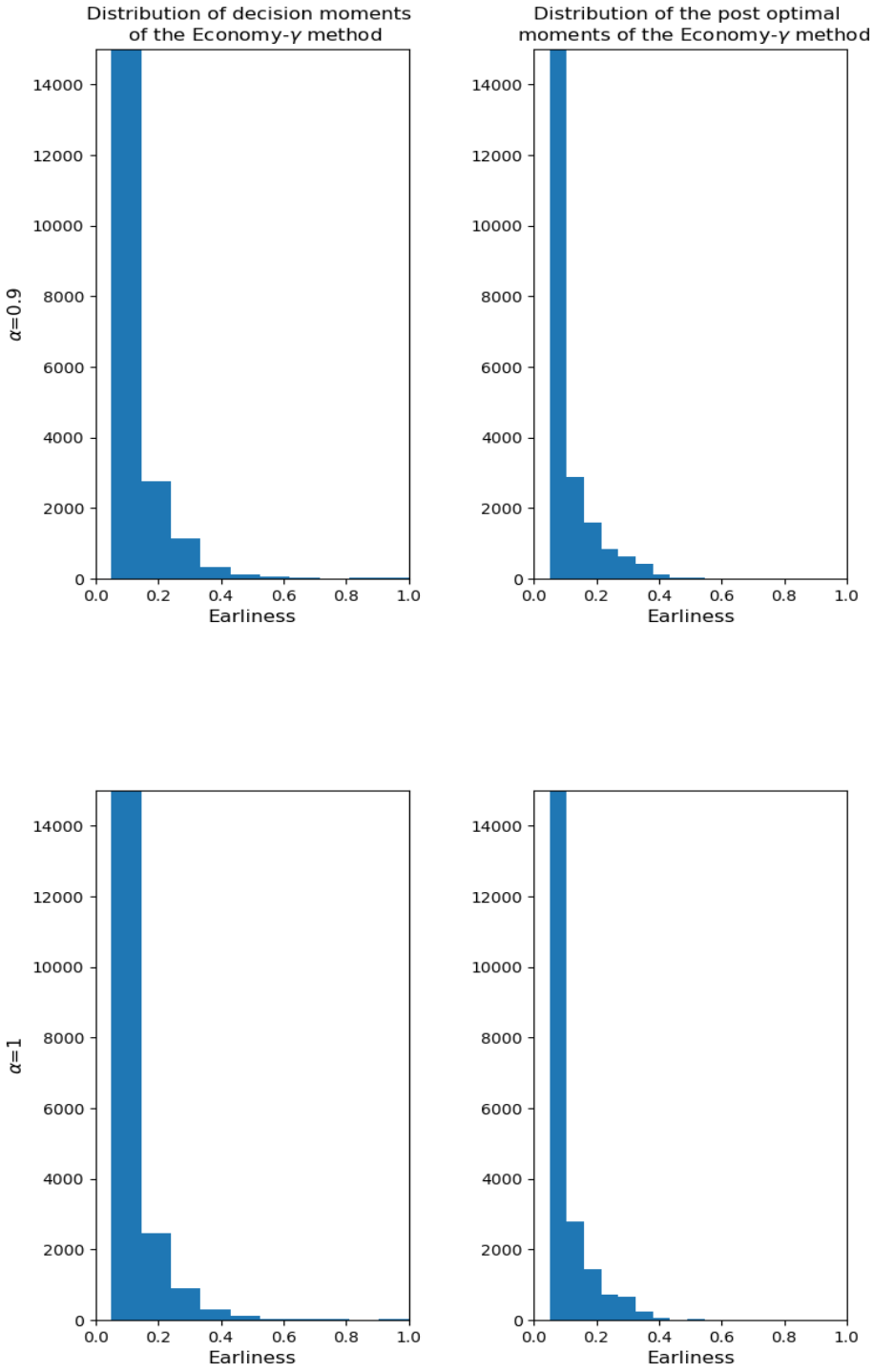


FIGURE B.5: The distribution of decision moments (left column) and post optimal moments (right column) for ECONOMY-γ with $\alpha \in \{0.9, 1\}$

B.1.2 ECONOMY-K

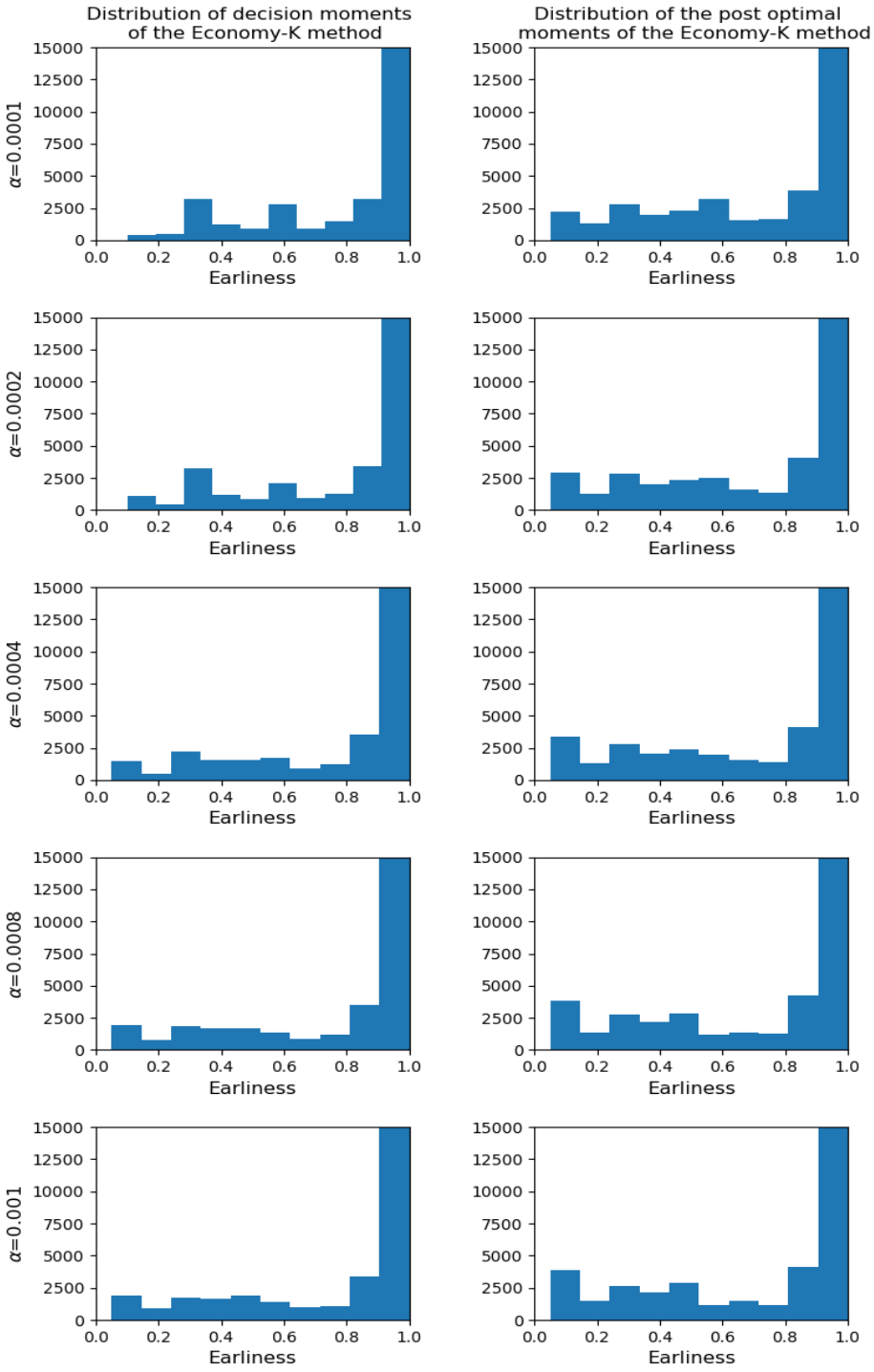


FIGURE B.6: The distribution of decision moments (left column) and post optimal moments (right column) for ECONOMY-K with $\alpha \in \{0.0001, 0.0002, 0.0004, 0.0008, 0.001\}$

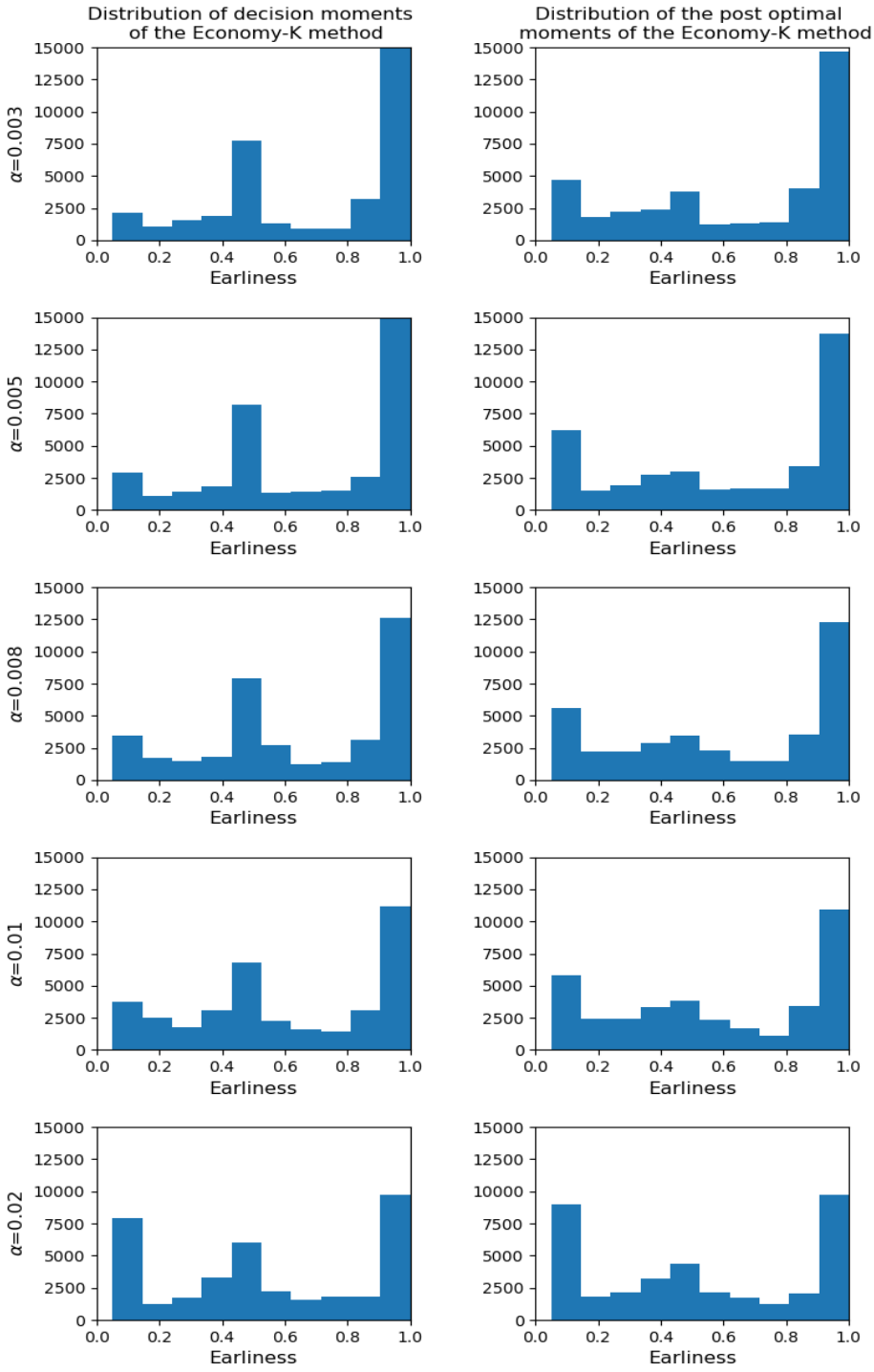


FIGURE B.7: The distribution of decision moments (left column) and post optimal moments (right column) for ECONOMY-K with $\alpha \in \{0.003, 0.005, 0.008, 0.01, 0.02\}$

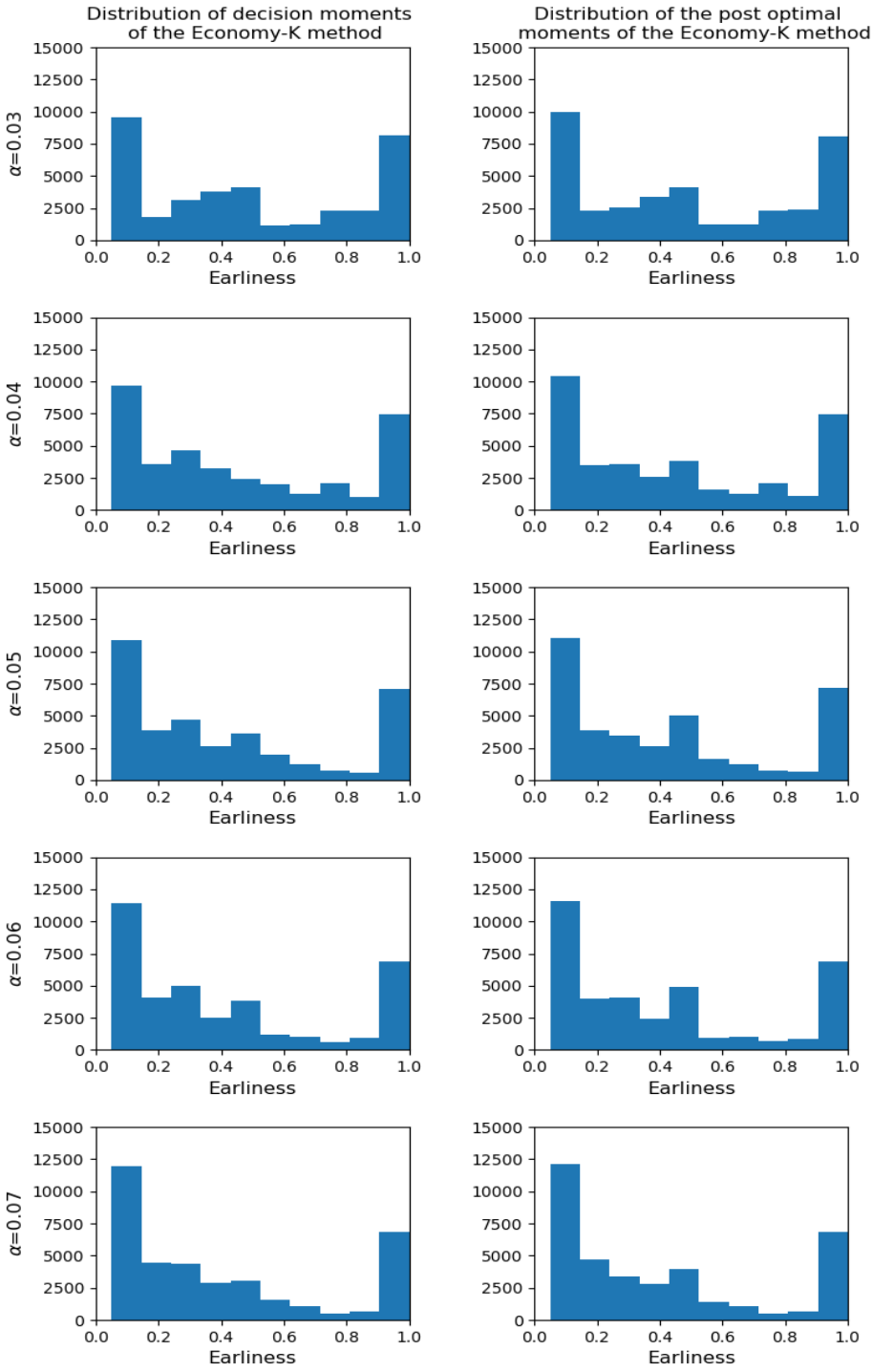


FIGURE B.8: The distribution of decision moments (left column) and post optimal moments (right column) for ECONOMY-K with $\alpha \in \{0.03, 0.04, 0.05, 0.06, 0.07\}$

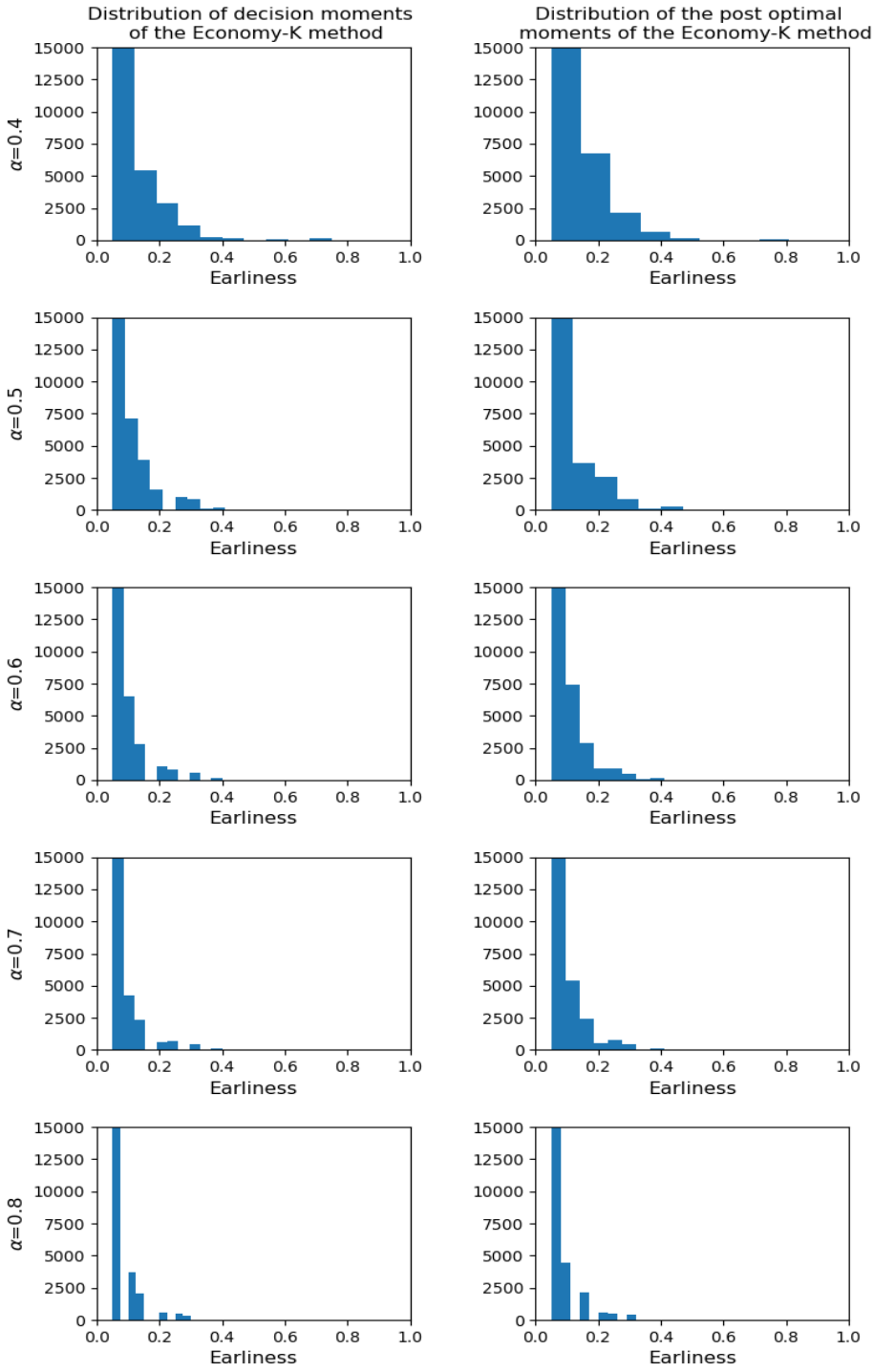


FIGURE B.9: The distribution of decision moments (left column) and post optimal moments (right column) for ECONOMY-K with $\alpha \in \{0.4, 0.5, 0.6, 0.7, 0.8\}$

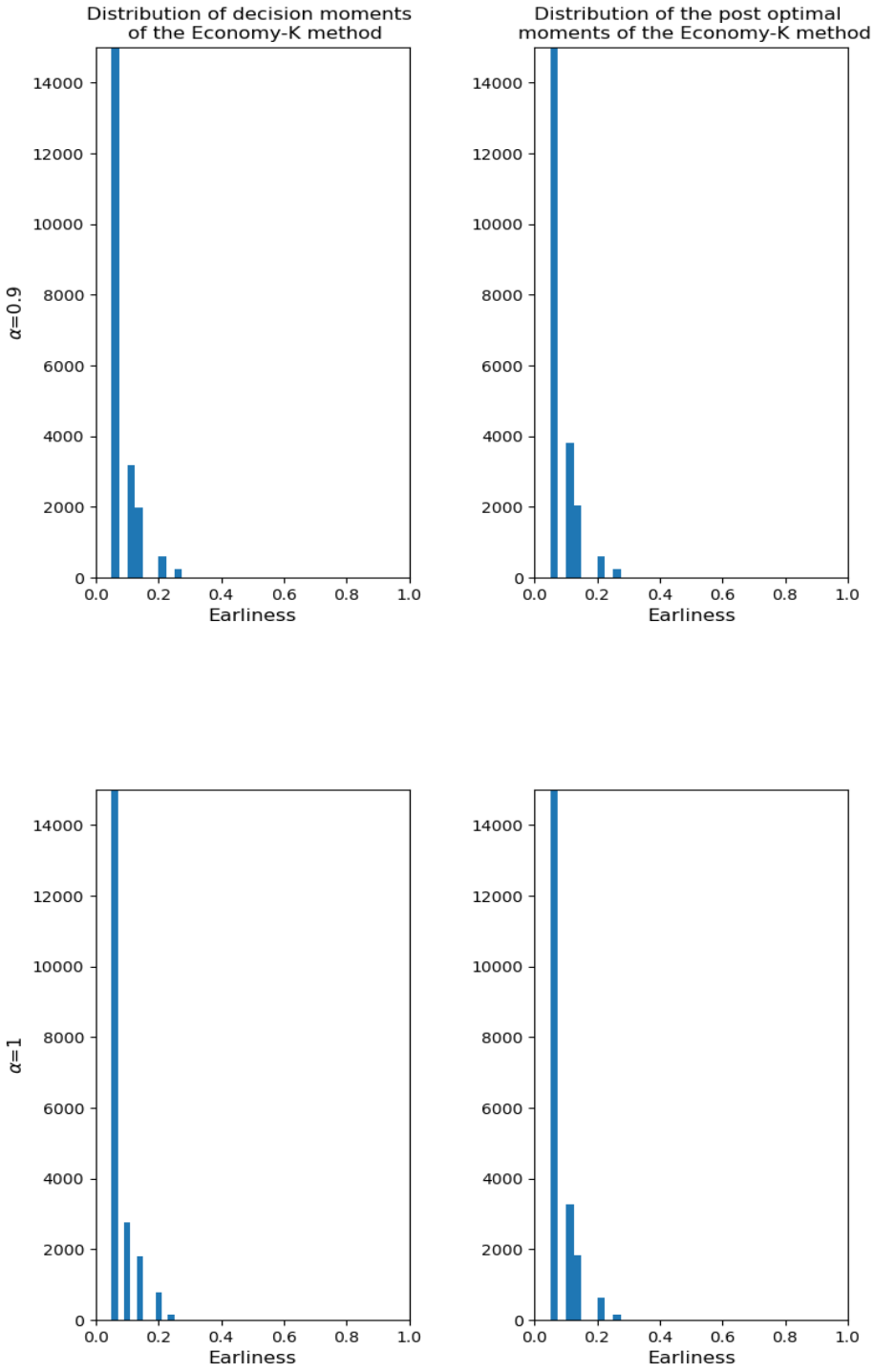


FIGURE B.10: The distribution of decision moments (left column) and post optimal moments (right column) for ECONOMY-K with $\alpha \in \{0.9, 1\}$

B.2 Additional experiments

Similar experiments as the ones described in Section 3.5 but performed on the 45 datasets benchmark proposed by (Mori, Alexander Mendiburu, Dasgupta, et al., 2017).

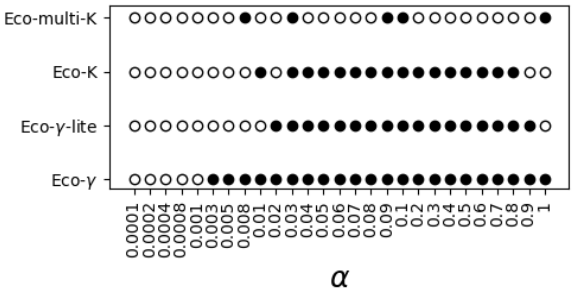
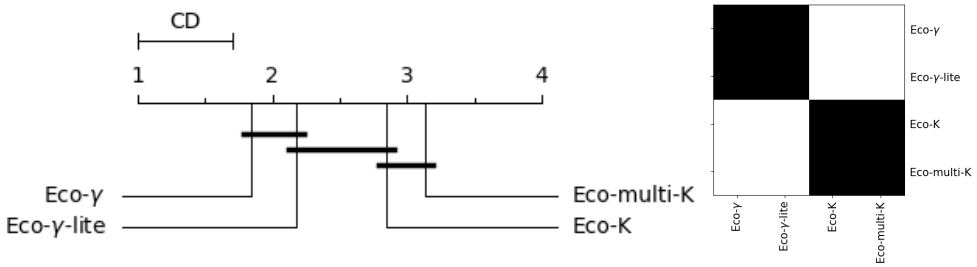


FIGURE B.11: Success of adapting the trigger times - Wilcoxon signed-rank test results for different values of α: black dotes indicate success and circles failures.



(A) (B)

FIGURE B.12: Evaluation based on *AvgCost*: (a) Nemenyi test applied to the 45 datasets; (b) pairwise comparison using the Wilcoxon signed-rank test, with black squares identifying non-significant comparisons.

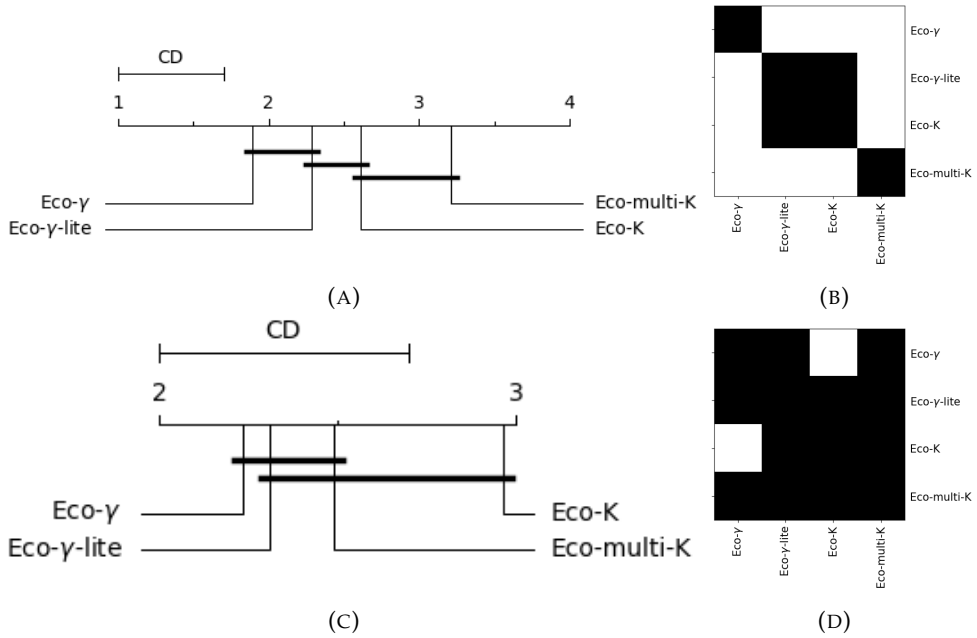


FIGURE B.13: Earliness (a, b) and predictive performance (c, d) comparison of the ECONOMY approaches.

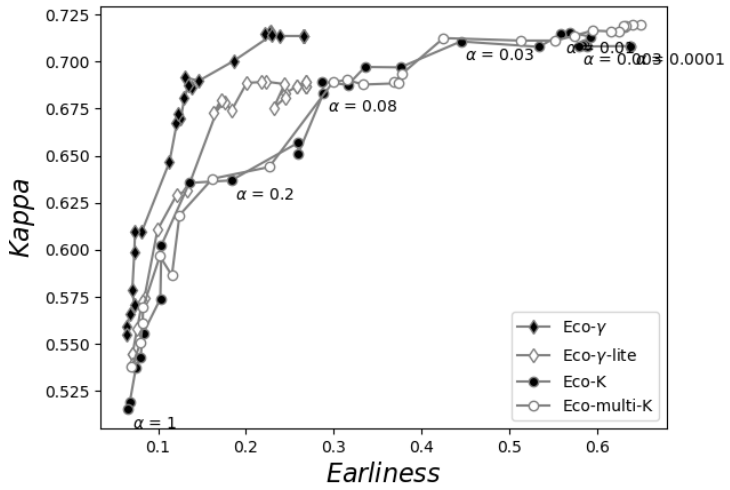
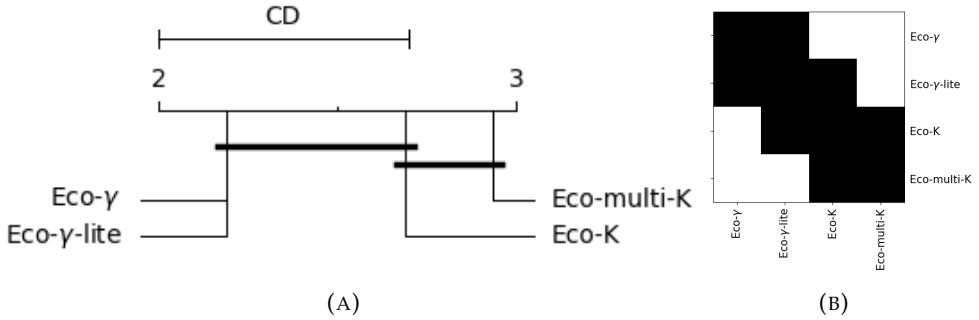


FIGURE B.14: Average *Earliness* vs. Average *Kappa* score obtain over the 45 datasets by varying the slope of the time cost, such as $\alpha \in [10^{-4}, 1]$.



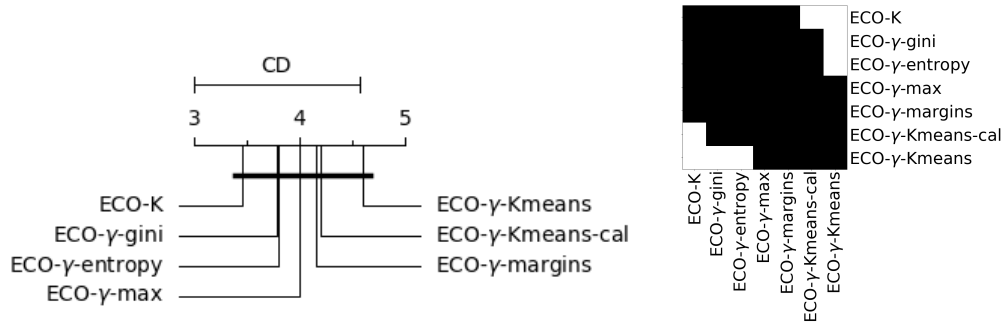
(A) (B)

FIGURE B.15: Evaluation of the quality of online decisions based on Δ_{cost}.

B.3 Additional results for multi-class classification problems

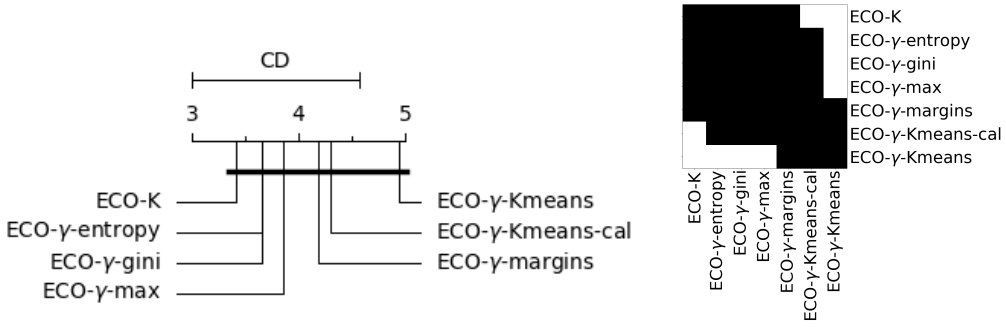
B.3.1 Nemenyi & Wilcoxon results for all α

Below are presented the full results obtained for the 33 datasets from Mori, Alexander Mendiburu, Dasgupta, et al., 2017 with more than 2 classes.



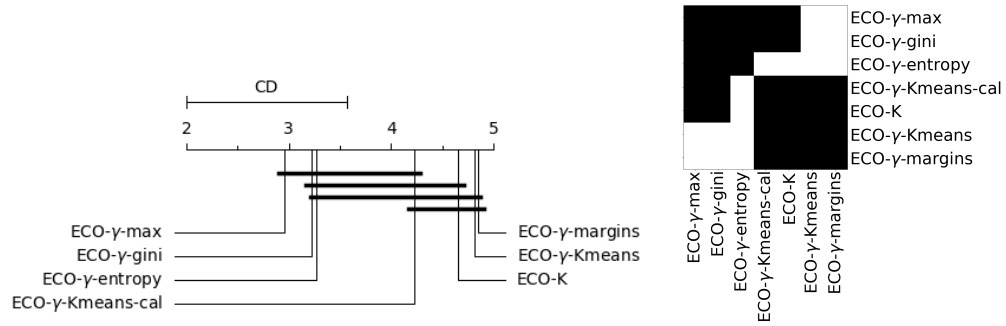
(A) (B)

FIGURE B.16: Comparison of ECONOMY approaches for $\alpha = 0.001$ using (a) Nemenyi and (b) Wilcoxon signed-rank tests



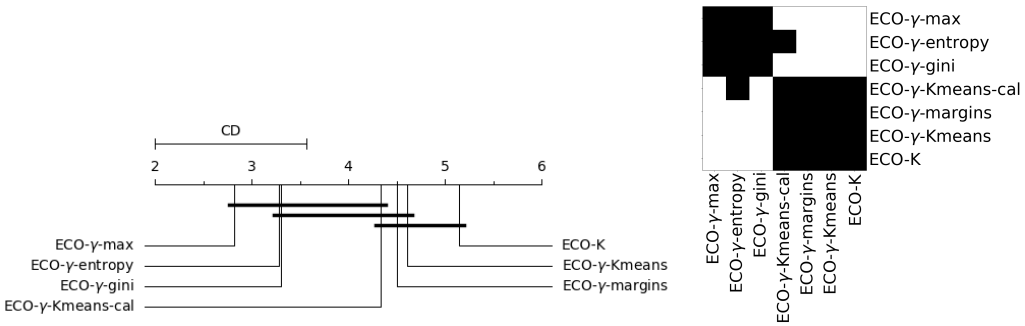
(A) (B)

FIGURE B.17: Comparison of ECONOMY approaches for $\alpha = 0.01$ using (a) Nemenyi and (b) Wilcoxon signed-rank tests



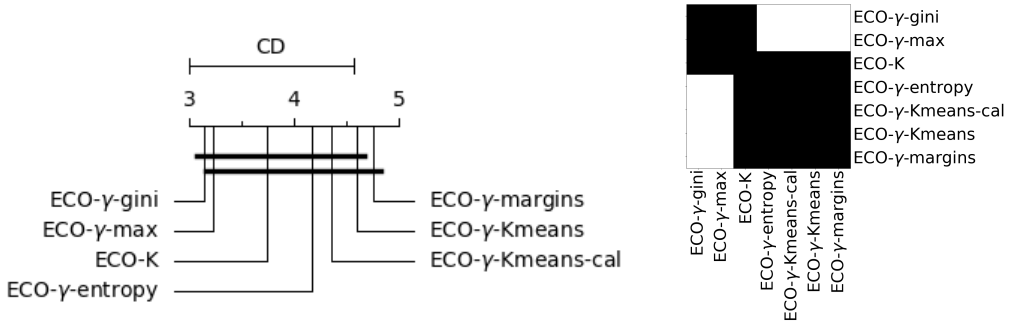
(A) (B)

FIGURE B.18: Comparison of ECONOMY approaches for $\alpha = 0.1$ using (a) Nemenyi and (b) Wilcoxon signed-rank tests



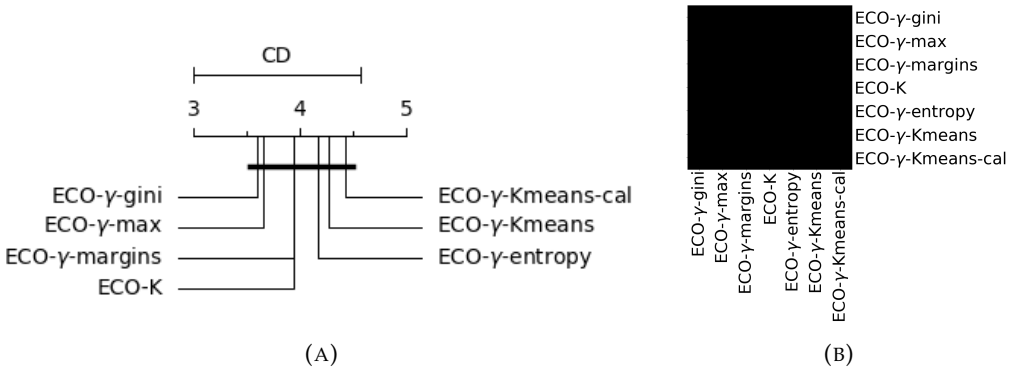
(A) (B)

FIGURE B.19: Comparison of ECONOMY approaches for $\alpha = 0.2$ using (a) Nemenyi and (b) Wilcoxon signed-rank tests



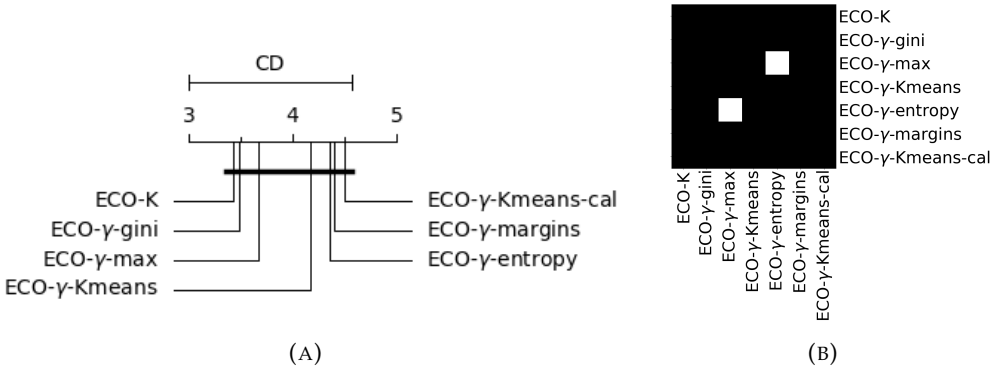
(A) (B)

FIGURE B.20: Comparison of ECONOMY approaches for $\alpha = 0.3$ using (a) Nemenyi and (b) Wilcoxon signed-rank tests



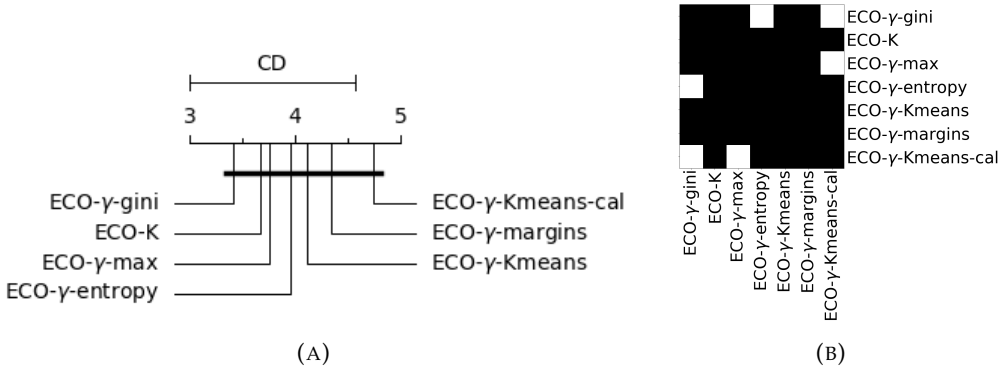
(A) (B)

FIGURE B.21: Comparison of ECONOMY approaches for $\alpha = 0.4$ using (a) Nemenyi and (b) Wilcoxon signed-rank tests



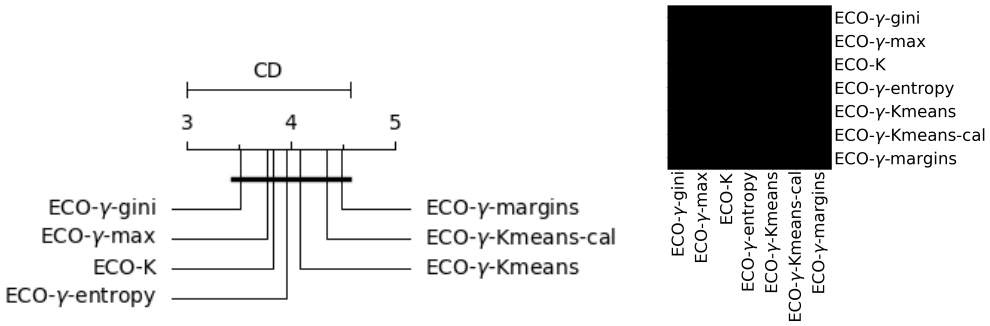
(A) (B)

FIGURE B.22: Comparison of ECONOMY approaches for $\alpha = 0.5$ using (a) Nemenyi and (b) Wilcoxon signed-rank tests



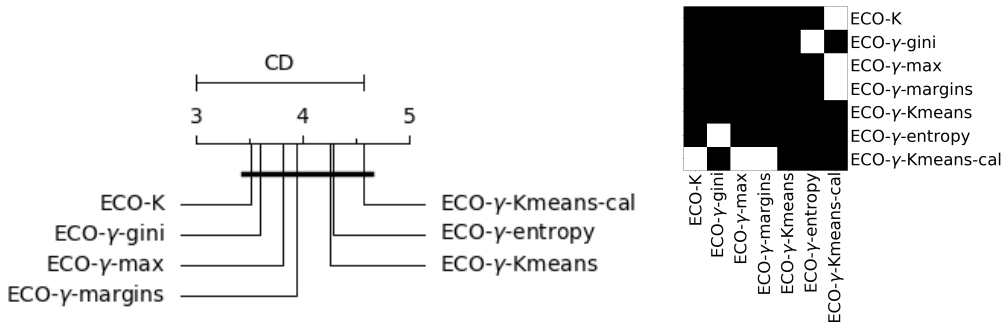
(A) (B)

FIGURE B.23: Comparison of ECONOMY approaches for $\alpha = 0.6$ using (a) Nemenyi and (b) Wilcoxon signed-rank tests



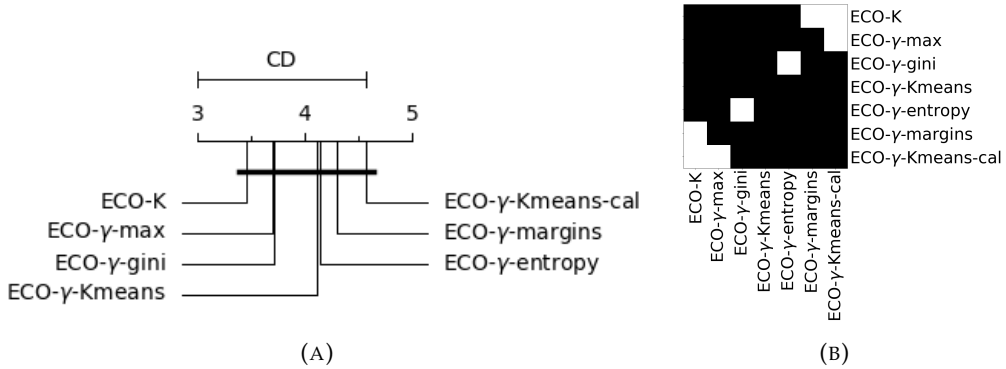
(A) (B)

FIGURE B.24: Comparison of ECONOMY approaches for $\alpha = 0.7$ using (a) Nemenyi and (b) Wilcoxon signed-rank tests



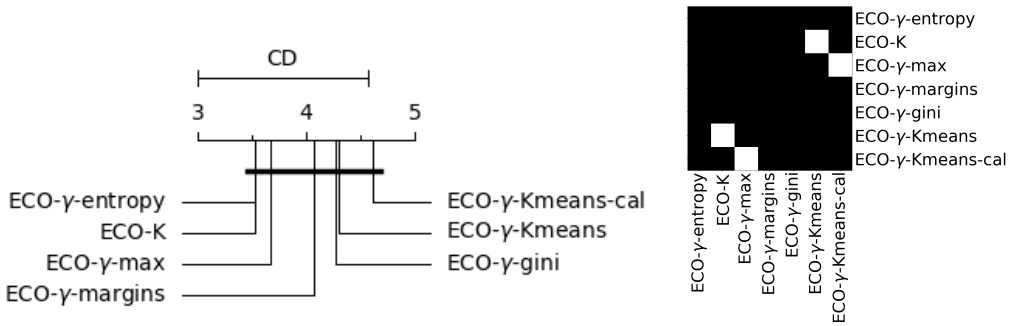
(A) (B)

FIGURE B.25: Comparison of ECONOMY approaches for $\alpha = 0.8$ using (a) Nemenyi and (b) Wilcoxon signed-rank tests



(A) (B)

FIGURE B.26: Comparison of ECONOMY approaches for $\alpha = 0.9$ using (a) Nemenyi and (b) Wilcoxon signed-rank tests



(A) (B)

FIGURE B.27: Comparison of ECONOMY approaches for $\alpha = 1$ using (a) Nemenyi and (b) Wilcoxon signed-rank tests

Appendix C

Appendix of chapter 4

This appendix gives further details on experiments presented in Chapter 4, Section C.1 shows that the percentage of samples for which a revocation is useful is relatively low, that demonstrates that the tackled problem is hard, then Section C.2 presents results of the Friedmann test comparing the two proposed approaches, Section C.3 shows the pareto curves for additional values of β and finally experiments on multi-class classification problems are presented in Section C.4.

C.1 Percentage of samples for which a revocation is useful

This section aims to check if the cost paid by the user can be improved by revoking the decision made by the irrevocable regime, statistics on results of ECONOMY-γ were computed. Figure C.6 shows the percentage of samples for which the score paid by the user can be improved by revoking the decision of ECONOMY-γ for each dataset and α value.

These statistics show that decisions can be successfully revoked on very few samples which makes the *early and revocable time series classification* problem difficult to solve, because the algorithm needs to identify those samples using incomplete knowledge of the time series in order to revoke the decision on them and keep the decision of the irrevocable regime on the other samples.

Datasets	0.0001	0.0003	0.0005	0.0008	0.001	0.0025	0.005	0.0075	0.01	0.025	0.05	0.075	0.1	0.25	0.5	0.75	1
CBF	1.075	1.075	1.075	1.075	1.075	1.075	1.075	1.075	1.075	1.075	1.075	2.151	3.226	3.226	4.659	5.018	6.452
ChlorineConcentration	12.142	12.142	12.142	12.142	12.142	12.142	12.142	12.142	12.142	16.705	12.142	18.097	20.65	21.036	21.346	23.279	23.511
CinCECGTorso	1.174	1.174	1.174	1.174	1.174	1.174	1.174	1.174	1.174	1.174	1.174	1.174	1.174	0.939	1.643	1.643	1.643
Crop	0.111	0.111	0.111	0.111	0.111	0.139	0.139	0.139	0.139	0.153	0.417	0.486	0.5	1.014	3.681	3.681	3.681
ECG5000	0.267	0.267	0.267	0.267	0.267	0.267	0.533	0.6	0.8	0.867	0.867	0.867	0.933	0.933	1.667	1.6	1.733
ECGFiveDays	0	0	0	0	0	0	0	0	0	0	2.256	2.256	2.256	2.256	10.902	8.271	12.406
ElectricDevices	0	0	0	0	0	0	0	0	0	1.302	1.302	5.028	5.329	8.594	10.877	12.28	13.241
FaceAll	0	0	0	0	0	0	0	0	0	0	0	0	0.296	0.148	0.296	0.444	0.444
FacesUCR	1.63	1.63	1.63	1.63	1.63	1.63	1.63	1.63	1.63	2.519	2.074	3.407	3.407	4.741	5.778	6.519	6.519
FiftyWords	4.044	4.044	4.044	4.044	4.044	4.044	4.044	4.044	4.044	4.044	2.206	4.779	4.779	3.309	5.515	4.412	5.515
FordA	0	0	0	0	0	0	2.031	2.031	2.031	2.573	2.979	3.453	3.859	4.875	5.281	7.177	10.494
FreezerRegularTrain	0.111	0.111	0.111	0.111	0.111	0.111	0.111	0.111	0.111	0.111	0.111	0.111	0.111	0.556	0.556	1.333	1.333
HandOutlines	4.38	4.38	4.38	4.38	4.38	4.623	4.623	4.623	4.623	4.623	6.083	4.623	8.029	8.273	11.922	14.599	15.815
InsectWingbeatSound	1.97	1.97	1.97	1.97	1.97	1.97	4.848	4.848	4.848	4.848	1.667	1.667	2.424	3.333	4.697	5	7.727
ItalyPowerDemand	0.912	0.912	0.912	0.912	0.912	0.912	0.912	0.912	0.912	3.04	3.04	2.432	5.775	8.815	18.541	22.188	25.836
Mallat	0.139	0.139	0.139	0.139	0.139	0.139	0.139	0.139	0.139	0.417	0.417	0.556	0.556	1.528	2.222	3.889	3.611
MedicalImages	9.621	9.621	9.621	9.621	9.621	9.621	9.621	9.621	9.621	9.913	9.913	11.662	12.245	18.659	18.659	23.907	23.907
MelbournePedestrian	0.367	0.367	0.367	0.367	0.367	0.367	0.367	0.367	0.367	0.642	0.642	0.826	0.826	1.743	1.651	2.294	8.532
MixedShapesRegularTrain	1.025	1.025	1.025	1.025	1.025	1.025	1.025	1.025	1.025	1.253	1.253	1.253	1.367	1.822	1.822	2.733	2.164
MoteStrain	3.141	3.141	3.141	3.141	3.141	3.141	3.141	2.094	2.094	3.927	3.927	5.236	6.021	6.806	7.068	8.377	15.445
NonInvasiveFatalECGThorax2	0.619	0.619	0.619	0.619	0.619	0.619	0.619	0.619	0.619	0.619	0.619	0.619	0.619	0.619	0.619	0.708	1.239
PhalangesOutlinesCorrect	15.414	15.414	15.414	15.414	15.414	15.414	15.414	16.165	16.165	14.787	18.045	18.045	18.045	22.306	24.311	28.947	30.827
ProximalPhalanxOutlineCorrect	8.209	8.209	8.209	8.209	8.209	8.209	8.209	8.209	8.209	9.701	9.701	10.821	11.194	17.164	26.493	26.119	26.493
SemgHandGenderCh2	7	7	6.5	6.5	6.5	6.5	6.5	6.5	6.5	6.5	6.5	6.5	6.5	7	11	15	14.5
SonyAIBORobotSurface2	5.442	5.442	5.442	5.442	5.442	5.442	5.442	5.442	5.442	5.442	5.442	5.442	5.442	6.463	8.503	10.204	9.864
StarLightCurves	0	0	0	0	0	0	0.974	1.155	1.191	1.119	1.263	2.382	2.382	2.382	3.428	4.15	4.367
Strawberry	4.407	4.407	4.407	4.407	4.407	4.407	4.407	4.407	4.746	4.746	5.085	5.085	5.085	7.119	13.559	14.237	15.593
Symbols	1.333	1.333	1.333	1.333	1.333	1.333	1.333	1.333	1.333	1.333	1.333	2	2.333	2.333	3	3	10
TwoLeadECG	0.86	0.86	0.86	0.86	0.287	0.287	0.287	0.287	0.86	0.86	0.86	0.86	0.86	2.292	4.585	5.158	8.023
TwoPatterns	1.467	1.467	1.467	1.467	1.467	1.467	1.467	1.467	1.867	1.933	1.933	3.133	3.267	10.067	24.533	25.067	25.4
UWaveGestureLibraryX	1.265	1.265	1.265	1.265	1.265	1.265	1.265	1.265	1.265	1.562	2.455	2.455	2.679	3.646	4.836	6.994	7.887
Wafer	0	0	0	0	0	0.047	0.047	0.047	0.047	0.047	0.14	0.14	0.14	0.14	0.186	0.279	0.279
WordSynonyms	5.515	5.515	5.515	5.515	5.515	5.515	5.515	5.515	5.515	5.882	4.412	6.985	4.412	6.25	8.456	9.559	14.338
Yoga	10.303	10.303	10.303	10.303	10.303	10.303	10.505	10.808	10.808	10.808	11.616	12.626	12.424	13.232	13.434	14.747	14.747

FIGURE C.1: Percentage of samples for which the score paid by the user can be improved by revoking the decision of ECONOMY-γ. Each column represent a different value of α in ascending order.

C.2 Average ranking using the Friedman test

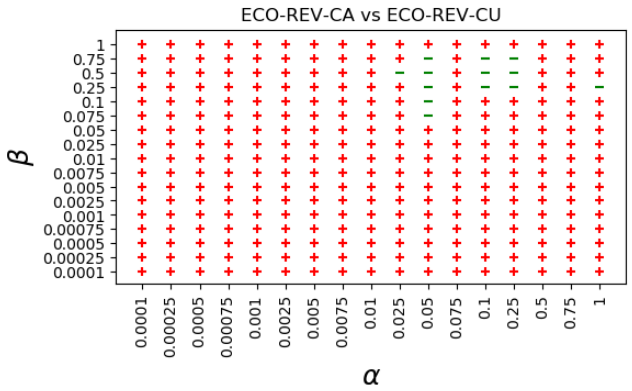


FIGURE C.2: ECO-REV-CA *vs.* ECO-REV-CU: Average ranking based on *AvgCost* using the Friedman test, for different values of α and β: "+" indicates ECO-REV-CA having a better average rank than ECO-REV-CA, and "−" indicates the opposite .

In our paper, we used Wilcoxon signed-rank test to compare the two proposed approaches (see Section 5). In addition, Figure C.2 uses the Friedman test and shows that ECO-REV-CA has better average rank than ECO-REV-CU in 96% of cases.

C.3 Average Earliness vs. Average Kappa for different values of β

This section presents the pareto curve for additional values of β, we notice that the same order holds for all values of β except for β=0.75 and β=1 where ECO-REV-CA becomes worse than ECONOMY-γ, however ECO-REV-CU still performs better.

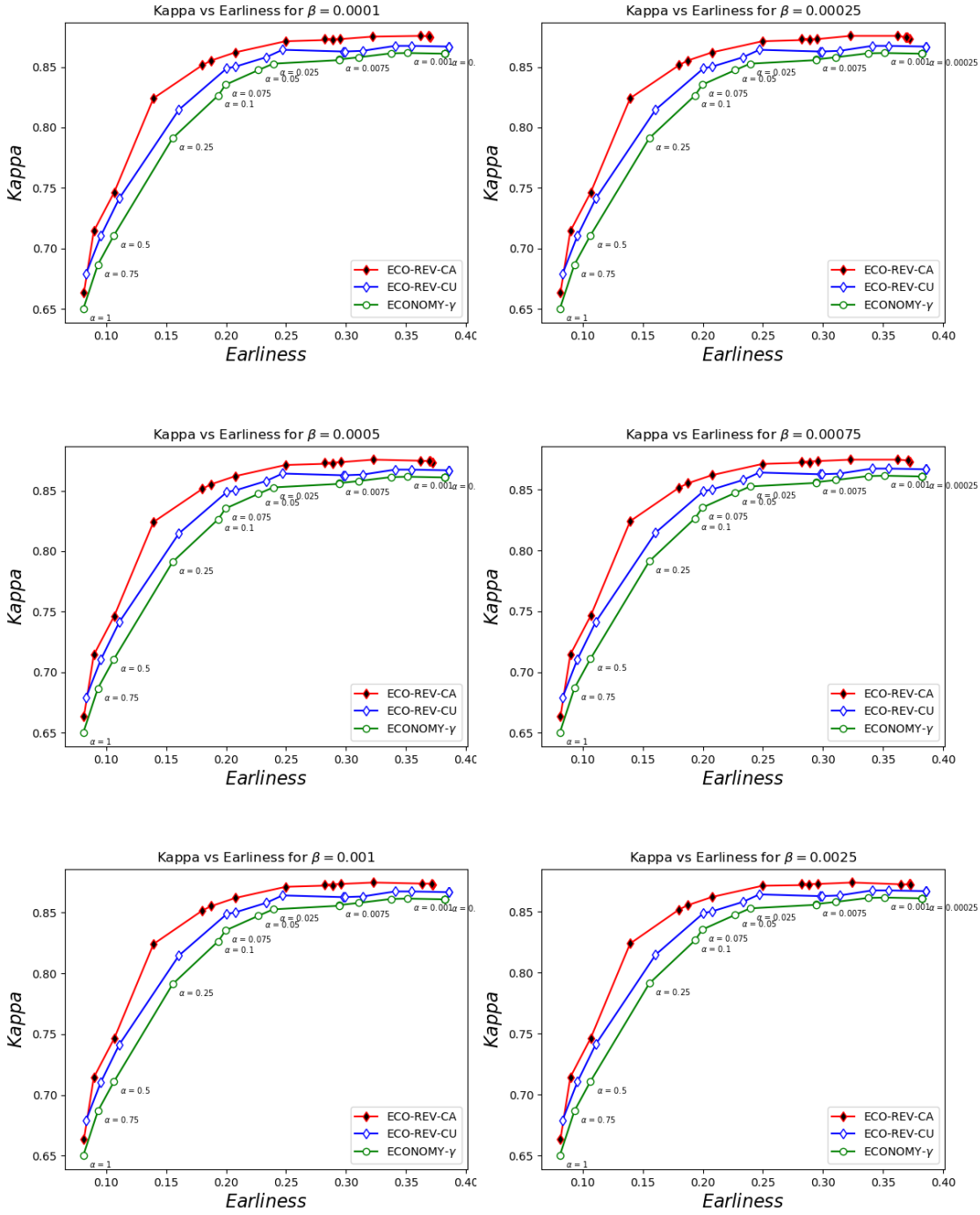


FIGURE C.3: Average *Earliness* vs. Average *Kappa* score obtained over the 34 datasets by varying α of the delay cost.

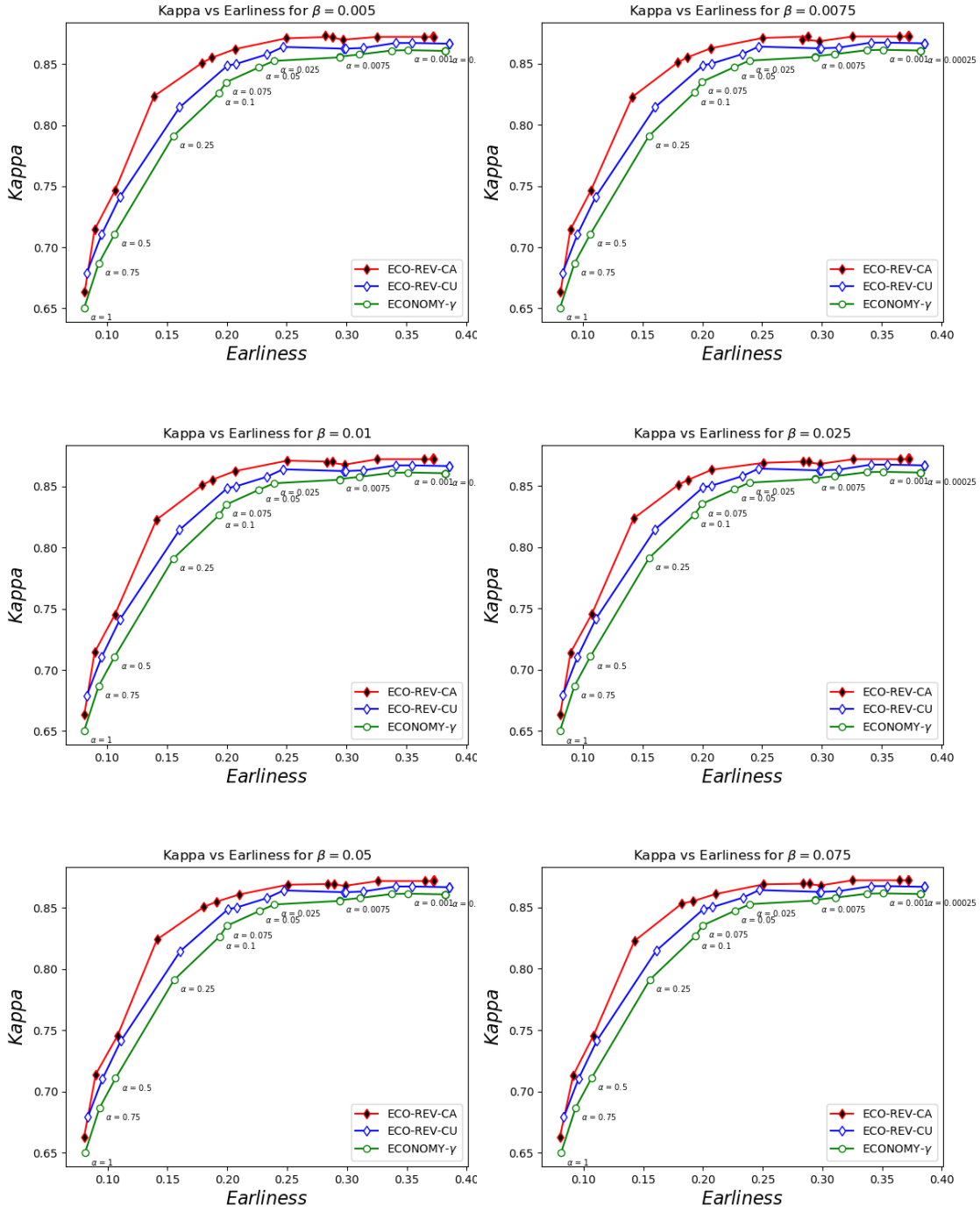


FIGURE C.4: Average *Earliness* vs. Average *Kappa* score obtained over the 34 datasets by varying α of the delay cost.

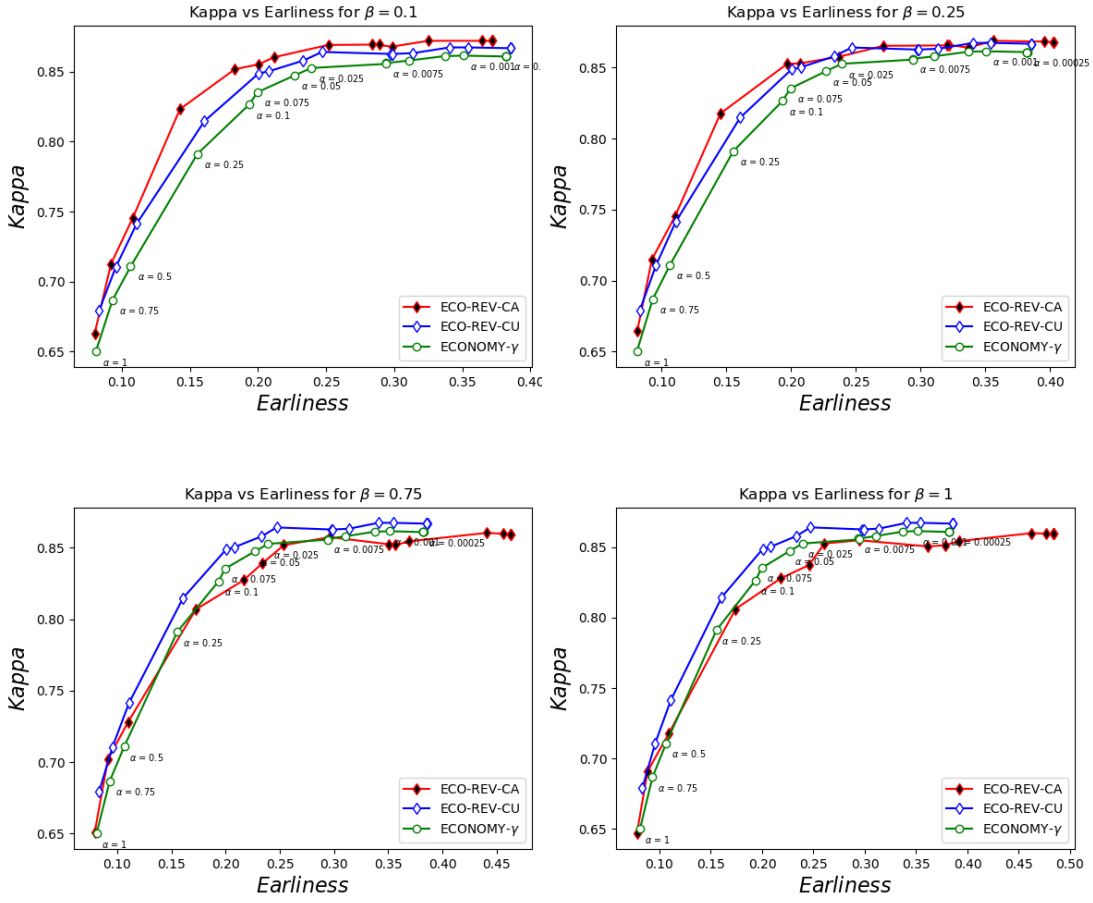


FIGURE C.5: Average *Earliness* vs. Average *Kappa* score obtained over the 34 datasets by varying α of the delay cost.

C.4 Experiments on multi-class early and revocable classification problems

This section gives two simple implementation examples of the ECONOMY framework, in the case of multi-class problems and revocable decisions.

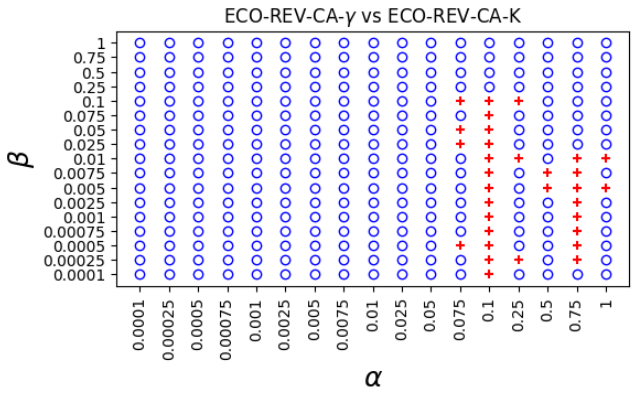


FIGURE C.6: ECO-REV-CA-γ VS. ECO-REV-CA-K: Wilcoxon signed-rank test applied on the *AvgCost* criterion over the 22 test sets, for a range of couples of values α and β, with "+" indicating a significant success of the first approach, "$\circ$" an insignificant difference and "$-$" indicating a significant failure of the first approach.

ECONOMY-γ is by design limited to binary early classification problems. The reason behind this limitation is that the partitioning of time series in the training set is done by discretising the confidence level of the classifiers regarding the positive class. We propose a new method ECONOMY-γ-MC which handles multi-class early classification problems. The intuition behind this method is that time series with similar uncertainty level of classifiers belong to the same group. In fact, instead of discretising the confidence level of the classifiers to the positive class, we propose to discretise the uncertainty of the classifier. This uncertainty is measured using the relative entropy function. The mechanism of predicting the groups at future time steps using Markov chain remains unchanged compared to ECONOMY-γ.

ECONOMY-K partition time series using the clustering algorithm K-means. This approach can deal by design with multi-class classification problems. ECONOMY-K and ECONOMY-γ-MC are used to implement early and revocable framework described in our paper. Experiments were performed on 22 datasets chosen at random from the UAE-UCR archive to compare performance of these two revocable strategies called respectively ECO-REV-CA-K and ECO-REV-CA-γ. Results of Wilcoxon signed-rank test applied on the *AvgCost* criterion are shown in Figure C.6. Further detailed results focusing on *AvgCost* are presented in Table 1 and Table 2.

Datasets	Class	α = 0.0001		α = 0.0025		α = 0.0005		α = 0.00075		α = 0.001		α = 0.0025		α = 0.005		α = 0.0075		α = 0.01	
		G	K	G	K	G	K	G	K	G	K	G	K	G	K	G	K	G	K
Plane	7	0.0953	0.0955	0.0953	0.0956	0.0953	0.0957	0.0953	0.0959	0.0954	0.096	0.0956	0.0969	0.0959	0.0827	0.0962	0.0841	0.0965	0.0694
BirdChicken	2	0.3334	0.3334	0.3335	0.3335	0.3337	0.3337	0.3338	0.3338	0.334	0.334	0.335	0.335	0.3368	0.3368	0.3385	0.3385	0.3402	0.3402
MoteStrain	2	0.0341	0.0446	0.0342	0.0473	0.0343	0.0475	0.0344	0.0477	0.0344	0.0479	0.0349	0.0361	0.0357	0.0381	0.0366	0.0401	0.0374	0.0395
ProximalPhalanxOutlineCorrect	2	0.1197	0.1158	0.1198	0.1159	0.1199	0.1162	0.1201	0.1164	0.1239	0.1166	0.1246	0.1181	0.1258	0.1205	0.127	0.1228	0.1352	0.1252
Fungi	18	0.1775	0.0807	0.1775	0.0808	0.1776	0.081	0.1777	0.0811	0.1778	0.0813	0.1785	0.0822	0.1795	0.0838	0.1806	0.117	0.1816	0.1184
RefrigerationDevices	3	0.4134	0.4134	0.4134	0.4134	0.4136	0.4136	0.4137	0.4137	0.4138	0.4138	0.4145	0.4145	0.4156	0.4156	0.4167	0.4167	0.3597	0.4178
SemgHandMovementCh2	6	0.2993	0.3038	0.2994	0.3039	0.2995	0.3041	0.2996	0.3043	0.2998	0.3046	0.3005	0.3058	0.3017	0.308	0.3069	0.3101	0.3081	0.3122
Ham	2	0.2785	0.4154	0.2785	0.4155	0.2786	0.4156	0.2786	0.4157	0.2787	0.4158	0.279	0.4164	0.2799	0.4173	0.2804	0.4183	0.2809	0.4193
Wine	2	0.3824	0.1765	0.3824	0.1766	0.3825	0.1768	0.3826	0.1769	0.3827	0.1771	0.3831	0.178	0.3838	0.1795	0.3846	0.1811	0.3853	0.1825
HandOutlines	2	0.1193	0.0949	0.1193	0.095	0.1194	0.0951	0.1195	0.0952	0.1196	0.0953	0.1201	0.0959	0.1209	0.0969	0.1216	0.0979	0.1003	0.0989
OliveOil	4	0.2778	0.5001	0.2778	0.5002	0.2779	0.5004	0.2779	0.5007	0.278	0.5009	0.2782	0.5022	0.2787	0.5045	0.2792	0.5067	0.2797	0.5087
FreezerRegularTrain	2	0.0011	0.0023	0.0011	0.0023	0.0012	0.0024	0.0012	0.0024	0.0012	0.0014	0.0014	0.0028	0.0017	0.0033	0.002	0.0047	0.0022	0.0061
Earthquakes	2	0.2015	0.2159	0.2015	0.216	0.2016	0.2161	0.2016	0.2163	0.2017	0.2164	0.2021	0.2173	0.2027	0.2188	0.2031	0.2202	0.2036	0.2217
Trace	4	0.0334	0.0334	0.0335	0.0336	0.0336	0.0338	0.0337	0.034	0.0338	0.0342	0.0345	0.0356	0.0357	0.0378	0.0369	0.0401	0.0381	0.0423
SonyAIBORobotSurface1	2	0.0321	0.0268	0.0322	0.0269	0.0324	0.0271	0.0326	0.0273	0.0327	0.0275	0.0337	0.0233	0.0353	0.0301	0.0369	0.027	0.0385	0.033
ArrowHead	3	0.2188	0.1719	0.2188	0.1721	0.2189	0.1722	0.219	0.1724	0.219	0.1726	0.2195	0.1737	0.2202	0.1756	0.2209	0.1775	0.2216	0.1793
ChlorineConcentration	3	0.219	0.2174	0.2191	0.2175	0.2193	0.2178	0.2195	0.2172	0.2197	0.2174	0.221	0.2187	0.2181	0.2209	0.219	0.217	0.2261	0.2252
Strawberry	2	0.0611	0.0511	0.0611	0.0512	0.0612	0.0513	0.0613	0.0515	0.0614	0.0516	0.0619	0.0557	0.0659	0.064	0.0666	0.0783	0.0673	0.0784
ScreenType	3	0.4408	0.4535	0.4409	0.4537	0.4411	0.4539	0.4413	0.4542	0.4415	0.4544	0.4428	0.4514	0.4448	0.4538	0.4469	0.4695	0.449	0.4761
InsectEPGRegularTrain	3	0.0	0.0	0.0	0.0001	0.0	0.0001	0.0	0.0002	0.0	0.0003	0.0001	0.0006	0.0002	0.0011	0.0004	0.012	0.0005	0.0123
FordA	2	0.3016	0.3016	0.3016	0.3016	0.3016	0.3016	0.3016	0.3016	0.3016	0.3016	0.3017	0.3017	0.3018	0.3018	0.302	0.302	0.3021	0.3021
FacesUCR	14	0.0617	0.1215	0.0617	0.1216	0.0618	0.1218	0.0619	0.122	0.062	0.1221	0.0626	0.1231	0.0635	0.1247	0.0601	0.1263	0.061	0.1279

TABLE C.1: AvgCost for every dataset in the 22 benchmark for ECO-REV-CA-γ and ECO-REV-CA-K given a fixed value of $\beta = 0.01$

Datasets	Class	α = 0.025		α = 0.05		α = 0.075		α = 0.1		α = 0.25		α = 0.5		α = 0.75		α = 1	
		G	K	G	K	G	K	G	K	G	K	G	K	G	K	G	K
Plane	7	0.0985	0.074	0.1017	0.0664	0.1049	0.0736	0.1081	0.0641	0.1275	0.1427	0.1597	0.1792	0.2093	0.2698	0.2124	0.2764
BirdChicken	2	0.3504	0.3504	0.3675	0.3678	0.3846	0.3833	0.4017	0.4822	0.5042	0.5042	0.6851	0.6322	0.7359	0.6984	0.6709	0.6533
MoteStrain	2	0.0518	0.0565	0.0606	0.0821	0.0595	0.097	0.0772	0.1061	0.1236	0.1598	0.169	0.1917	0.1915	0.2124	0.2162	0.2248
ProximalPhalanxOutlineCorrect	2	0.1415	0.1394	0.1249	0.1868	0.1524	0.2204	0.1686	0.2294	0.1994	0.2697	0.2601	0.316	0.3024	0.3285	0.3239	0.341
Fungi	18	0.1879	0.1266	0.1984	0.1403	0.2088	0.1539	0.2193	0.1676	0.2821	0.2497	0.4098	0.3758	0.4782	0.5281	0.6025	0.6529
RefrigerationDevices	3	0.3627	0.4246	0.3727	0.4358	0.3724	0.4471	0.3886	0.4583	0.4075	0.4331	0.4236	0.4706	0.4283	0.4691	0.4533	0.4848
SemgHandMovementCh2	6	0.3149	0.325	0.3263	0.3462	0.3253	0.3943	0.3491	0.3741	0.3922	0.4854	0.4827	0.565	0.6231	0.6231	0.6481	0.6481
Ham	2	0.2842	0.4251	0.2896	0.4349	0.295	0.4446	0.3004	0.4544	0.2993	0.5388	0.3803	0.4771	0.4249	0.5156	0.4518	0.5629
Wine	2	0.3898	0.1914	0.3972	0.1764	0.4046	0.191	0.4121	0.2057	0.4567	0.3168	0.531	0.4769	0.4531	0.669	0.5844	0.6403
HandOutlines	2	0.1045	0.1049	0.1114	0.1148	0.1153	0.1248	0.122	0.1348	0.1655	0.1754	0.2052	0.2359	0.2366	0.2791	0.2595	0.2853
OliveOil	4	0.2825	0.4085	0.2872	0.3661	0.2919	0.3236	0.2966	0.3226	0.3249	0.3218	0.3719	0.3569	0.419	0.3842	0.4661	0.3799
FreezerRegularTrain	2	0.007	0.0064	0.0062	0.0095	0.0074	0.0122	0.0109	0.0153	0.0212	0.0262	0.0361	0.0383	0.0517	0.0507	0.06	0.0632
Earthquakes	2	0.2087	0.2641	0.2352	0.247	0.2412	0.2483	0.2472	0.2495	0.2568	0.2568	0.269	0.269	0.2812	0.2812	0.2934	0.2934
Trace	4	0.0452	0.0555	0.0571	0.0582	0.069	0.0869	0.0853	0.1123	0.2147	0.1399	0.3681	0.3112	0.3768	0.3162	0.3483	0.3541
SonyAIBORobotSurface1	2	0.0659	0.04	0.0845	0.0643	0.065	0.0503	0.067	0.075	0.0801	0.1131	0.1016	0.1284	0.1085	0.1391	0.1412	0.1498
ArrowHead	3	0.226	0.1903	0.2332	0.1762	0.2405	0.2082	0.1584	0.2054	0.2912	0.2949	0.3071	0.301	0.2977	0.4181	0.4369	0.4618
ChlorineConcentration	3	0.2335	0.2404	0.2388	0.2551	0.2474	0.2743	0.2545	0.2871	0.2923	0.3285	0.3425	0.3943	0.384	0.4239	0.4077	0.4456
Strawberry	2	0.0749	0.0754	0.0824	0.0859	0.0844	0.0895	0.09	0.1022	0.1254	0.1504	0.1718	0.2129	0.213	0.2396	0.2371	0.2846
ScreenType	3	0.4635	0.4524	0.477	0.4813	0.4718	0.4817	0.489	0.5051	0.5684	0.6107	0.6575	0.6345	0.6826	0.6429	0.6879	0.6462
InsectEPGRegularTrain	3	0.0012	0.0151	0.0025	0.0059	0.0037	0.0085	0.005	0.0112	0.0125	0.0125	0.025	0.025	0.0374	0.0374	0.0499	0.0499
FordA	2	0.3028	0.3028	0.3041	0.3041	0.3053	0.3053	0.3066	0.3066	0.3141	0.3141	0.3266	0.3266	0.3391	0.3391	0.3516	0.3516
FacesUCR	14	0.0665	0.1375	0.0829	0.1535	0.0968	0.1696	0.1031	0.1856	0.1638	0.2826	0.2626	0.3565	0.3352	0.4482	0.3966	0.5166

TABLE C.2: AvgCost for every dataset in the 22 benchmark for ECO-REV-CA-γ and ECO-REV-CA-K given a fixed value of $\beta = 0.01$

Appendix D

Appendix of chapter 5

D.1 The distribution of decision moments

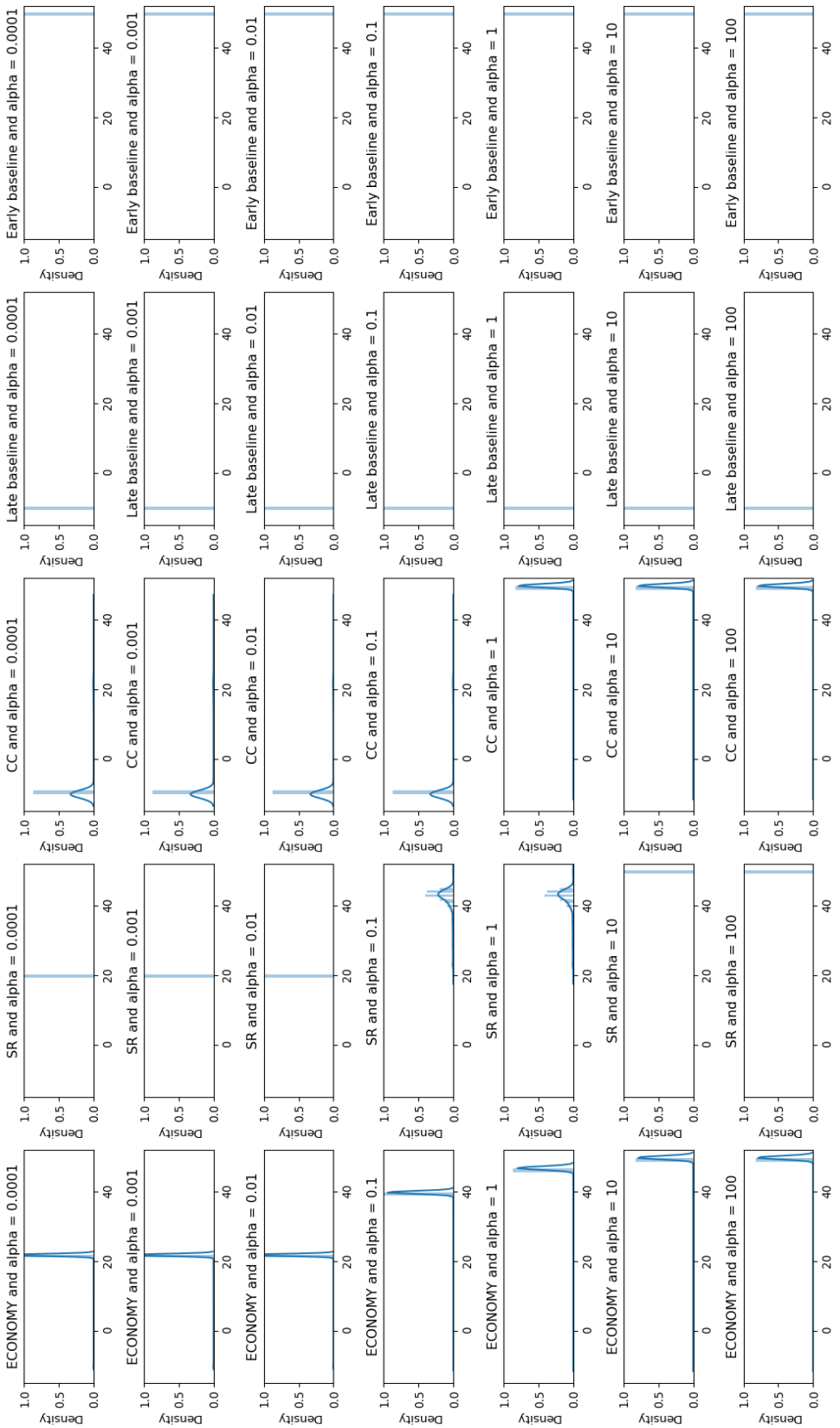


FIGURE D.1: The distribution of decision moments when $C_m^{(1)} = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$

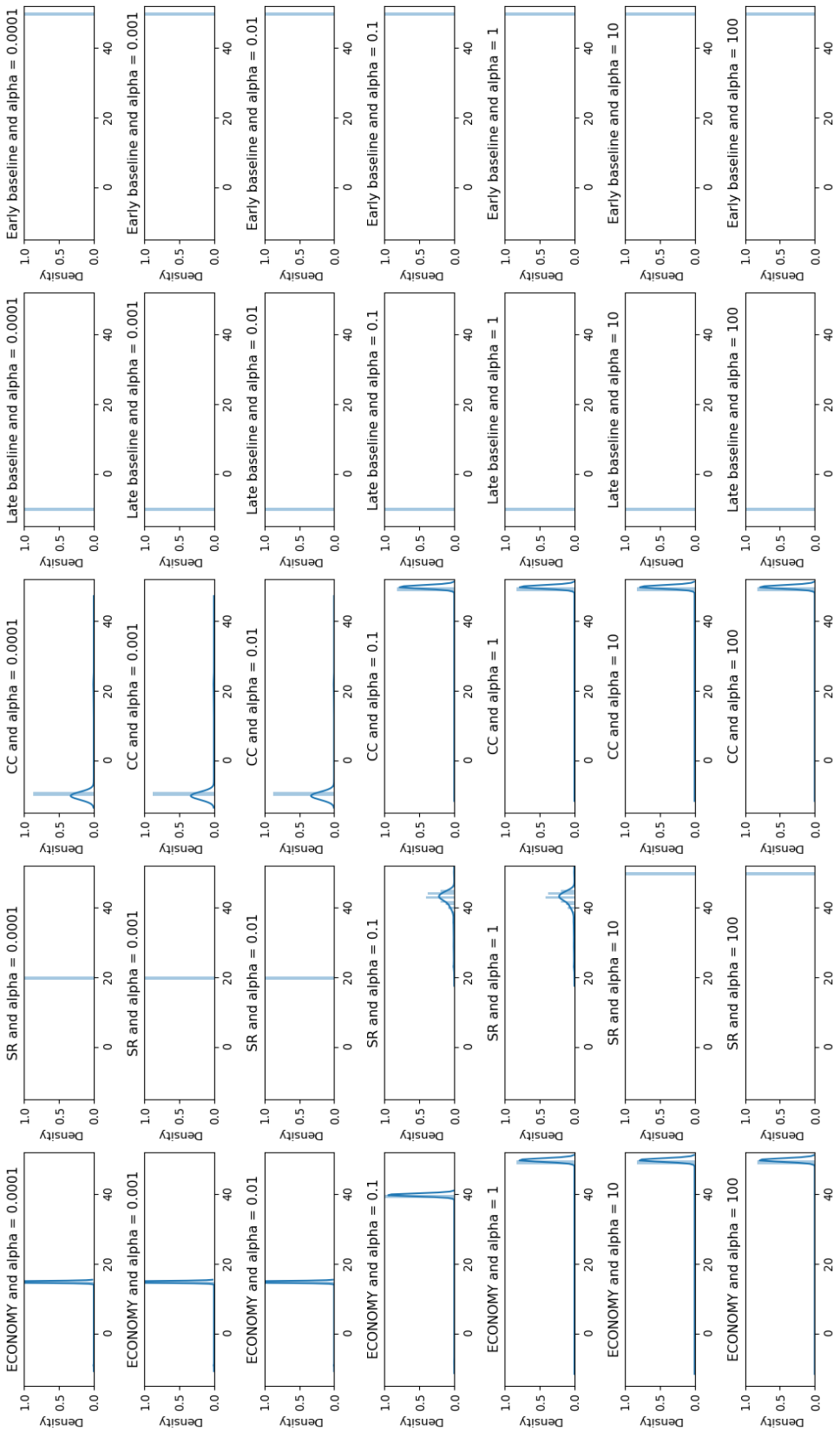


FIGURE D.2: The distribution of decision moments when $C_m^{(2)} = \begin{bmatrix} 0 & 10 \\ 1 & 0 \end{bmatrix}$

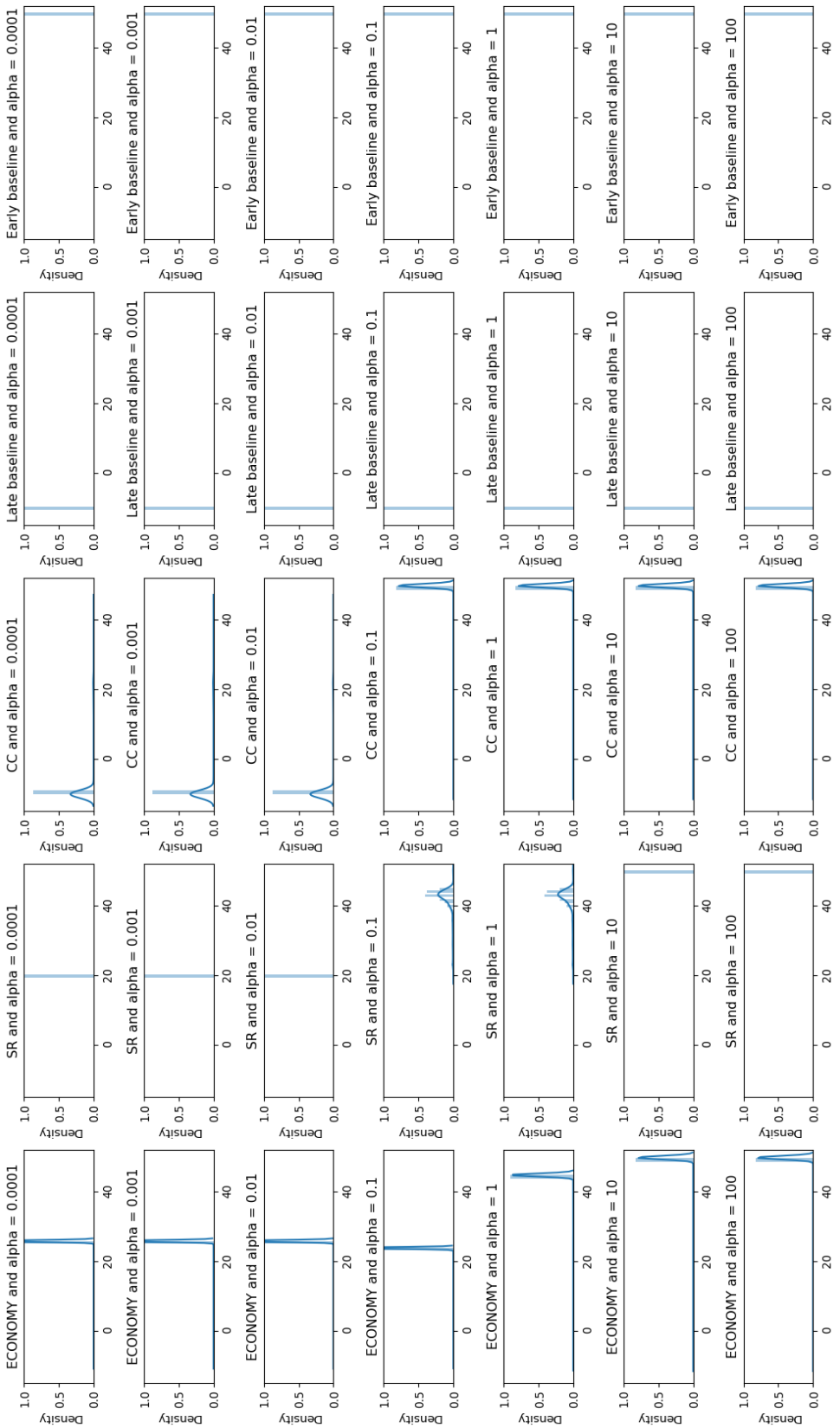


FIGURE D.3: The distribution of decision moments when $C_m^{(3)} = \begin{bmatrix} 0 & 100 \\ 1 & 0 \end{bmatrix}$

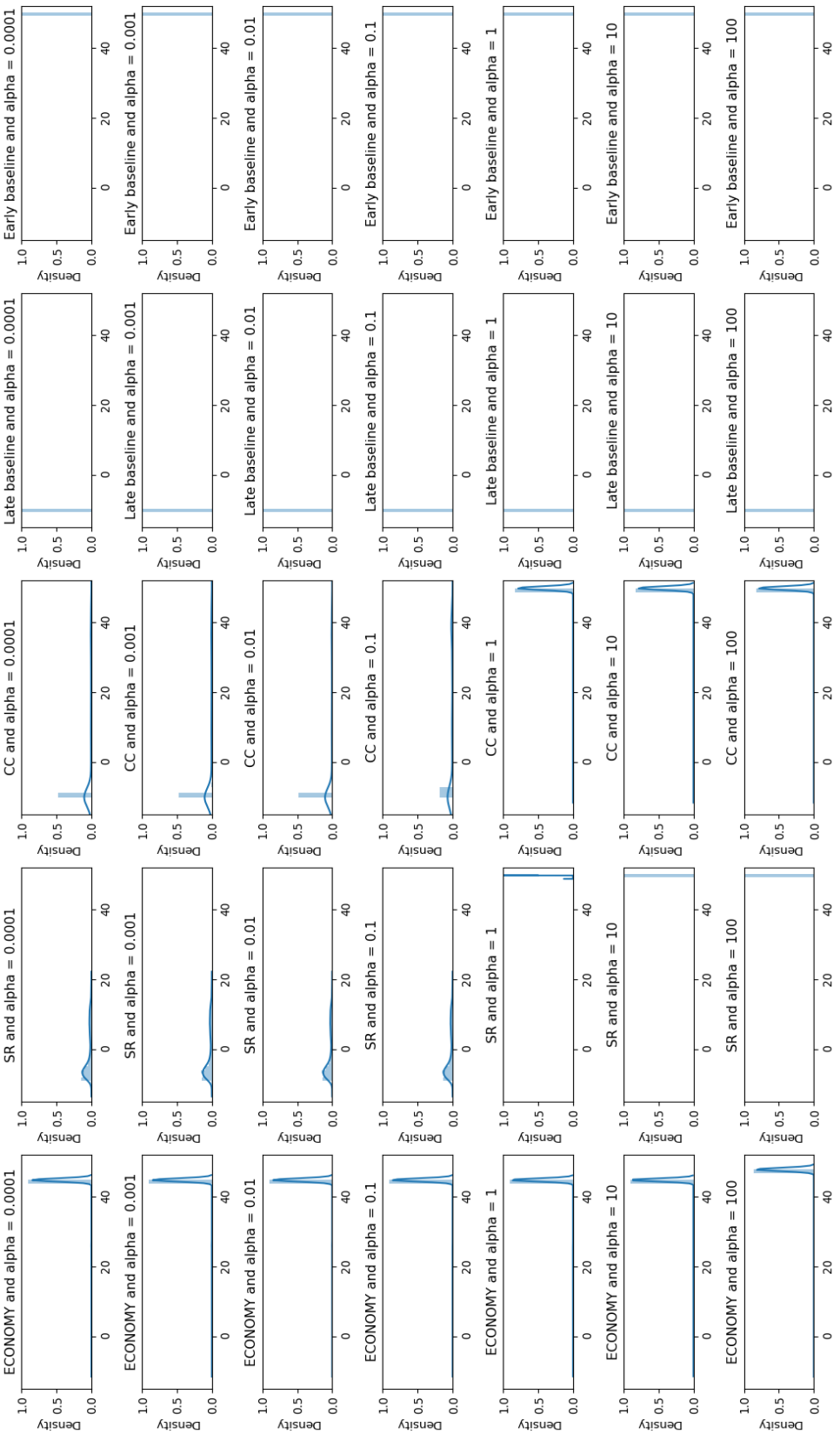


FIGURE D.4: The distribution of decision moments when $C_m^{(4)} = \begin{bmatrix} 0 & 1000 \\ 1 & 0 \end{bmatrix}$

Bibliography

Abanda, Amaia, Usue Mori, and Jose A Lozano (2019). "A review on distance based time series classification". In: *Data Mining and Knowledge Discovery* 33.2, pp. 378–412.

Abellan-Nebot, Jose Vicente and Fernando Romero Subirón (2010). "A review of machining monitoring systems based on artificial intelligence process models". In: *The International Journal of Advanced Manufacturing Technology* 47.1, pp. 237–257.

Achenchabe, Youssef, Alexis Bondu, and Antoine Cornuéjols (2021). "Early classification of time series. Cost-based optimization criterion and algorithms". In: *Machine Learning*.

Achenchabe, Youssef, Alexis Bondu, Antoine Cornuéjols, and Vincent Lemaire (2022a). "Early and Revocable Time Series Classification". In: *In Proceedings of the International Joint Conference on Neural Networks (IJCNN)*.

— (2022b). "ECOTS: Early Classification in Open Time Series". In: *arXiv preprint arXiv:2204.00392*.

Ajao, Oluwaseun, Deepayan Bhowmik, and Shahrzad Zargari (2019). "Sentiment aware fake news detection on online social networks". In: *IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*. IEEE, pp. 2507–2511.

Akermi, Kada, Samira Chouraqui, and Boudjemaa Boudaa (2020). "Novel SMC control design for path following of autonomous vehicles with uncertainties and mismatched disturbances". In: *International Journal of Dynamics and Control* 8.1, pp. 254–268.

Alcantarilla, Pablo F et al. (2018). "Street-view change detection with deconvolutional networks". In: *Autonomous Robots* 42.7, pp. 1301–1322.

Alipour-Fanid, Amir et al. (2019). "Machine learning-based delay-aware UAV detection over encrypted Wi-Fi traffic". In: *2019 IEEE Conference on Communications and Network Security (CNS)*. IEEE, pp. 1–7.

Alonso González, Carlos J and Juan J Rodrıguez Diez (2004). "Boosting interval-based literals: Variable length and early classification". In: *Data mining in time series databases*. World Scientific, pp. 149–171.

Anderson, Hyrum S et al. (2012). "Early Time-Series Classification with Reliability Guarantee". In: *Sandria Report*.

Arthur, David and Sergei Vassilvitskii (2007). "K-Means++: The Advantages of Careful Seeding". In: *Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms*. SODA '07. New Orleans, Louisiana: Society for Industrial and Applied Mathematics, pp. 1027–1035. ISBN: 9780898716245.

Aydin, Ilhan, SEVİ Mehmet, and Mehmet Umut Salur (2018). "Detection of Fake Twitter Accounts with Machine Learning Algorithms". In: *2018 International Conference on Artificial Intelligence and Data Processing (IDAP)*. IEEE, pp. 1–4.

Bagnall, A. et al. (2017). "The Great Time Series Classification Bake Off: a Review and Experimental Evaluation of Recent Algorithmic Advances". In: *Data Mining and Knowledge Discovery* 31 (3), pp. 606–660.

Bagnall, Anthony et al. (2017). "The great time series classification bake off: a review and experimental evaluation of recent algorithmic advances". In: *Data mining and knowledge discovery* 31.3, pp. 606–660.

Balcan, Maria-Florina, Andrei Broder, and Tong Zhang (2007). "Margin based active learning". In: *International Conference on Computational Learning Theory*. Springer, pp. 35–50.

Barandas, Marılia et al. (2020). "TSFEL: Time Series Feature Extraction Library". In: *SoftwareX* 11. https://github.com/fraunhoferportugal/tsfel, p. 100456.

Barber, David (2012). *Bayesian reasoning and machine learning*. Cambridge University Press.

Batista, Gustavo EAPA et al. (2014). "CID: an efficient complexity-invariant distance for time series". In: *Data Mining and Knowledge Discovery* 28.3, pp. 634–669.

Baydogan, Mustafa Gokce and George Runger (2016). "Time series representation and similarity based on local autopatterns". In: *Data Mining and Knowledge Discovery* 30.2, pp. 476–509.

Baydogan, Mustafa Gokce, George Runger, and Eugene Tuv (2013). "A bag-of-features framework to classify time series". In: *IEEE transactions on pattern analysis and machine intelligence* 35.11, pp. 2796–2802.

Beibel, Martin (2000). "A note on sequential detection with exponential penalty for the delay". In: *Annals of Statistics*, pp. 1696–1701.

Bekker, Jessa and Jesse Davis (2020). "Learning from positive and unlabeled data: A survey". In: *Machine Learning* 109.4, pp. 719–760.

Bellman, Richard and Robert Kalaba (1959). "On adaptive control processes". In: *IRE Transactions on Automatic Control* 4.2, pp. 1–9.

Berger, James O (1985). *Statistical decision theory and Bayesian analysis*. Springer Science & Business Media.

Bifet, Albert et al. (2018). *Machine learning for data streams: with practical examples in MOA*. MIT press.

Blum, Avrim and Tom Mitchell (1998). "Combining labeled and unlabeled data with co-training". In: *Proceedings of the eleventh annual conference on Computational learning theory*, pp. 92–100.

Bojarski, Mariusz, Davide Del Testa, et al. (2016). "End to end learning for self-driving cars". In: *arXiv preprint arXiv:1604.07316*.

Bojarski, Mariusz, Philip Yeres, et al. (2017). "Explaining how a deep neural network trained with end-to-end learning steers a car". In: *arXiv preprint arXiv:1704.07911*.

Breuer, Adam, Roee Eilat, and Udi Weinsberg (2020). "Friend or faux: Graph-based early detection of fake accounts on social networks". In: *Proceedings of The Web Conference 2020*, pp. 1287–1297.

Carbonneau, Marc-André et al. (May 2018). "Multiple instance learning: A survey of problem characteristics and applications". In: *Pattern Recognition* 77, pp. 329–353.

Carvalho, Thyago P. et al. (2019). "A systematic literature review of machine learning methods applied to predictive maintenance". In: *Computers & Industrial Engineering* 137, p. 106024. ISSN: 0360-8352. DOI: https://doi.org/10.1016/j.cie.2019.106024. URL: https://www.sciencedirect.com/science/article/pii/S0360835219304838.

Castillo-Sánchez, Gema et al. (2020). "Suicide risk assessment using machine learning and social networks: A scoping review". In: *Journal of medical systems* 44.12, pp. 1–15.

Chatfield, Chris (2000). *Time-series forecasting*. CRC press.

Chen, Yanping et al. (2015). *The UCR Time Series Classification Archive*. www.cs.ucr.edu/~eamonn/time_series_data/.

Chinnasamy, S., C. Muthusamy, and G. Gopal (2013). "An Outlier Based Bi-Level Neural Network Classification System for Improved Classification of Cardiotocogram Data". In: *Life Science Journal* 10.1, pp. 244–251.

Cohen, Jacob (1960). "A coefficient of agreement for nominal scales". In: *Educational and psychological measurement* 20.1, pp. 37–46.

Dachraoui, Asma, Alexis Bondu, and Antoine Cornuejols (2013). "Early classification of individual electricity consumptions". In: *RealStream2013 (ECML)*, pp. 18–21.

Dachraoui, Asma, Alexis Bondu, and Antoine Cornuéjols (2015). "Early classification of time series as a non myopic sequential decision making problem". In: *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer, pp. 433–447.

Das, Gautam et al. (1998). "Rule Discovery from Time Series." In: *KDD*. Vol. 98. 1, pp. 16–22.

data (2020). *Schwan's factory data*. https://github.com/DeeptiChevvuri/Predictive-Maintenance-Modelling-Datasets.

DeGroot, Morris H (2005). *Optimal statistical decisions*. Vol. 82. John Wiley & Sons.

Deng, Houtao et al. (2013). "A time series forest for classification and feature extraction". In: *Information Sciences* 239, pp. 142–153.

Desreumaux, Louis and Vincent Lemaire (2020). "Learning Active Learning at the Crossroads? Evaluation and Discussion". In: *Proceedings of the Workshop on Interactive Adaptive Learning, co-located with (ECML PKDD 2020)*. Ed. by Daniel Kottke et al. Vol. 2660. CEUR Workshop Proceedings. CEUR-WS.org, pp. 38–54. URL: http://ceur-ws.org/Vol-2660/ialatecml%5C_paper3.pdf.

Džeroski, Sašo (2009). "Relational data mining". In: *Data Mining and Knowledge Discovery Handbook*. Springer, pp. 887–911.

Eamonn, Keogh (July 2006). *time series tutorial*. http://didawiki.cli.di.unipi.it/lib/exe/fetch.php/dm/time_series_from_keogh_tutorial.pdf.

Ebihara, Akinori F et al. (2020). "Sequential density ratio estimation for simultaneous optimization of speed and accuracy". In: *arXiv preprint arXiv:2006.05587*.

Eco-rev (2021). *Early and Revocable classification library*. https://github.com/eco-rev/rev-algo. Online; January 2021.

Eisenhower (1944). https://www.eisenhowerlibrary.gov/sites/default/files/research/online-documents/d-day/order-of-the-day.pdf, last visited January 2021.

Elkan, Charles (2001). "The foundations of cost-sensitive learning". In: *International joint conference on artificial intelligence*. Vol. 17. 1. Lawrence Erlbaum Associates Ltd, pp. 973–978.

Elyusufi, Yasyn, Zakaria Elyusufi, et al. (2019). "Social networks fake profiles detection using machine learning algorithms". In: *The Proceedings of the Third International Conference on Smart City Applications*. Springer, pp. 30–40.

Ennaji, Abdelatif, Driss Mammass, Mostafa El Yassa, et al. (2012). "Self-training using a k-nearest neighbor as a base classifier reinforced by support vector machines". In: *International Journal of Computer Applications* 975, p. 8887.

Fagnant, Daniel J and Kara Kockelman (2015). "Preparing a nation for autonomous vehicles: opportunities, barriers and policy recommendations". In: *Transportation Research Part A: Policy and Practice* 77, pp. 167–181.

Fahrenkrog-Petersen, Stephan A et al. (2019). "Fire now, fire later: alarm-based systems for prescriptive process monitoring". In: *arXiv preprint arXiv:1905.09568*.

Ferguson, Thomas S (1989). "Who solved the secretary problem?" In: *Statistical science* 4.3, pp. 282–289.

Fire, Michael et al. (2014). "Friend or foe? Fake profile identification in online social networks". In: *Social Network Analysis and Mining* 4.1, p. 194.

Flach, Peter A (2016). "Classifier calibration". In: *Encyclopedia of Machine Learning and Data Mining*, pp. 1–8.

Frazier, Peter I and J Yu Angela (2007). "Sequential hypothesis testing under stochastic deadlines." In: *NIPS*, pp. 465–472.

Fuller, Aidan et al. (2020). "Digital Twin: Enabling Technologies, Challenges and Open Research". In: *IEEE Access* 8, pp. 108952–108971. DOI: 10.1109/ACCESS.2020.2998358.

Gabor, Thomas et al. (2016). "A Simulation-Based Architecture for Smart Cyber-Physical Systems". In: *2016 IEEE International Conference on Autonomic Computing (ICAC)*, pp. 374–379. DOI: 10.1109/ICAC.2016.29.

Gama, Joao (2010). *Knowledge discovery from data streams*. CRC Press.

— (2012). "A survey on learning from data streams: current and future trends". In: *Progress in Artificial Intelligence* 1.1, pp. 45–55.

Gama, João et al. (2014). "A survey on concept drift adaptation". In: *ACM Comput. Surv.* 46.4, 44:1–44:37. DOI: 10.1145/2523813. URL: https://doi.org/10.1145/2523813.

Ghalwash, Mohamed F and Zoran Obradovic (2012). "Early classification of multivariate temporal observations by extraction of interpretable shapelets". In: *BMC bioinformatics* 13.1, pp. 1–12.

Ghalwash, Mohamed F, Vladan Radosavljevic, and Zoran Obradovic (2014). "Utilizing temporal patterns for estimating uncertainty in interpretable early decision making". In: *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 402–411.

Gong, Siyuan and Lili Du (2018). "Cooperative platoon control for a mixed traffic flow including human drive vehicles and connected and autonomous vehicles". In: *Transportation research part B: methodological* 116, pp. 25–61.

Grabocka, Josif et al. (2014). "Learning time-series shapelets". In: *Proceedings of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 392–401.

Gupta, Ashish et al. (2020). "Approaches and Applications of Early Classification of Time Series: A Review". In: *IEEE Transactions on Artificial Intelligence* 1.1, pp. 47–61.

Hansson, Sven Ove (1994). "Decision Theory–A Brief Introduction". In.

Hartvigsen, Thomas et al. (2019). "Adaptive-halting policy network for early classification". In: *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 101–110.

— (2020). "Recurrent halting chain for early multi-label classification". In: *Proceedings of the 26th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*, pp. 1382–1392.

Al-Hasan, S and G Vachtsevanos (2002). "Intelligent route planning for fast autonomous vehicles operating in a large natural terrain". In: *Robotics and autonomous systems* 40.1, pp. 1–24.

He, Guoliang, Yong Duan, Rong Peng, et al. (2015). "Early classification on multivariate time series". In: *Neurocomputing* 149, pp. 777–787.

He, Guoliang, Yong Duan, Guofu Zhou, et al. (2014). "Early classification on multivariate time series with core features". In: *International Conference on Database and Expert Systems Applications*. Springer, pp. 410–422.

He, Yu et al. (2018). "Time-evolving Text Classification with Deep Neural Networks." In: *IJCAI*. Vol. 18, pp. 2241–2247.

Heimberger, Markus et al. (2017). "Computer vision in automated parking systems: Design, implementation and challenges". In: *Image and Vision Computing* 68, pp. 88–101.

Hills, Jon et al. (2014). "Classification of time series by shapelet transformation". In: *Data mining and knowledge discovery* 28.4, pp. 851–881.

Islam, Md Rafiqul et al. (2018). "Depression detection from social network data using machine learning techniques". In: *Health information science and systems* 6.1, pp. 1–12.

Jeong, Young-Seon, Myong K Jeong, and Olufemi A Omitaomu (2011). "Weighted dynamic time warping for time series classification". In: *Pattern recognition* 44.9, pp. 2231–2240.

John, Vijay et al. (2015). "Saliency map generation by the convolutional neural network for real-time traffic light detection using template matching". In: *IEEE Transactions on Computational Imaging* 1.3, pp. 159–173.

Khoshnevisan, Farzaneh and Min Chi (2021a). "Unifying Domain Adaptation and Domain Generalization for Robust Prediction Across Minority Racial Groups". In: *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer, pp. 521–537.

— (2021b). "Unifying Domain Adaptation and Domain Generalization for Robust Prediction Across Minority Racial Groups". In: *Joint European Conference on Machine Learning and Knowledge Discovery in Databases*. Springer, pp. 521–537.

Kim, Seong-Woo et al. (2015). "The impact of cooperative perception on decision making and planning of autonomous vehicles". In: *IEEE Intelligent Transportation Systems Magazine* 7.3, pp. 39–50.

Klenke, Achim (2013). *Probability theory: a comprehensive course*. Springer Science & Business Media.

Kochenderfer, Mykel J (2015). *Decision making under uncertainty: theory and application*. MIT press.

Krempl, Georg et al. (Sept. 2014). "Open Challenges for Data Stream Mining Research". In: *SIGKDD Explor. Newsl.* 16.1, pp. 1–10. ISSN: 1931-0145. DOI: 10.1145/2674026.2674028. URL: http://doi.acm.org/10.1145/2674026.2674028.

Latouche, Pierre and Fabrice Rossi (2015). "Graphs in machine learning: an introduction". In: *European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning (ESANN), Proceedings of the 23-th European Symposium on Artificial Neural Networks, Computational Intelligence and Machine Learning (ESANN 2015)*, pp. 207–218.

LeCun, Yann et al. (1998). "Gradient-based learning applied to document recognition". In: *Proceedings of the IEEE* 86.11, pp. 2278–2324.

Leiva, Victor and Ana Freire (2017). "Towards suicide prevention: early detection of depression on social media". In: *International Conference on Internet Science*. Springer, pp. 428–436.

Lemaire, Vincent, Fabrice Clérot, and Nicolas Creff (2015). "K-means clustering on a classifier-induced representation space : application to customer contact personalization". In: *Annals of Information Systems, Special Issue on Real-World Data Mining Application* Special Issue on Real-World Data Mining Application, pp. 139–153.

Lemaire, Vincent, Oumaima Alaoui Ismaili, et al. (2020). "Predictive K-means with Local Models". In: *Trends and Applications in Knowledge Discovery and Data Mining*. Ed. by Wei Lu and Kenny Q. Zhu. Springer International Publishing, pp. 91–103.

Lemaire, Vincent, Christophe Salperwyck, and Alexis Bondu (2014). "A survey on supervised classification on data streams". In: *European Business Intelligence Summer School*. Springer, pp. 88–125.

Li, Liangzhi, Kaoru Ota, and Mianxiong Dong (2018). "Humanlike driving: Empirical decision-making system for autonomous vehicles". In: *IEEE Transactions on Vehicular Technology* 67.8, pp. 6814–6823.

Liao, T Warren (2005). "Clustering of time series data—a survey". In: *Pattern recognition* 38.11, pp. 1857–1874.

Lin, Yu-Feng et al. (2015). "Reliable early classification on multivariate time series with numerical and categorical attributes". In: *Pacific-Asia Conference on Knowledge Discovery and Data Mining*. Springer, pp. 199–211.

Lin, Jessica, Rohan Khade, and Yuan Li (2012). "Rotation-invariant similarity in time series using bag-of-patterns representation". In: *Journal of Intelligent Information Systems* 39.2, pp. 287–315.

Liu, Yahui, Xiaolong Jin, and Huawei Shen (2019). "Towards early identification of online rumors based on long short-term memory networks". In: *Information Processing & Management* 56.4, pp. 1457–1467.

Liu, Yang and Yi-Fang Brook Wu (2020). "Fned: a deep network for fake news early detection on social media". In: *ACM Transactions on Information Systems (TOIS)* 38.3, pp. 1–33.

Liu, Yu and X Rong Li (2013). "Performance analysis of sequential probability ratio test". In: *Sequential Analysis* 32.4, pp. 469–497.

Loyola, Juan Martın et al. (2017). "Learning when to classify for early text classification". In: *Argentine Congress of Computer Science*. Springer, pp. 24–34.

Lugaresi, Giovanni and Andrea Matta (2018). "REAL-TIME SIMULATION IN MANUFACTURING SYSTEMS: CHALLENGES AND RESEARCH DIRECTIONS". In: *2018 Winter Simulation Conference (WSC)*, pp. 3319–3330. DOI: 10.1109/WSC.2018.8632542.

Lugaresi, Giovanni, Davide Travaglini, and Andrea Matta (2019). "A LEGO® Manufacturing System as Demonstrator for A Real-Time Simulation Proof of Concept". In: *2019 Winter Simulation Conference (WSC)*, pp. 2025–2036. DOI: 10.1109/WSC40007.2019.9004733.

Luzietti, Roberto et al. (1999). "European Community Multi-Center Trial "Fetal ECG Analysis During Labor": ST plus CTG analysis". In: 27.6, pp. 431–440. DOI: doi:10.1515/JPM.1999.058. URL: https://doi.org/10.1515/JPM.1999.058.

Ma, Yifang et al. (2020). "Artificial intelligence applications in the development of autonomous vehicles: a survey". In: *IEEE/CAA Journal of Automatica Sinica* 7.2, pp. 315–329.

Marteau, Pierre-François (2008). "Time warp edit distance with stiffness adjustment for time series matching". In: *IEEE transactions on pattern analysis and machine intelligence* 31.2, pp. 306–318.

Martinez, Coralie, Guillaume Perrin, et al. (2018). "A deep reinforcement learning approach for early classification of time series". In: *2018 26th European Signal Processing Conference (EUSIPCO)*. IEEE, pp. 2030–2034.

Martinez, Coralie, Emmanuel Ramasso, et al. (2020). "Adaptive early classification of temporal sequences using deep reinforcement learning". In: *Knowledge-Based Systems* 190, p. 105290.

Marzio, M Di and Charles C Taylor (2005). "Kernel density classification and boosting: an l2 analysis". In: *Statistics and Computing* 15.2, pp. 113–123.

Mathukia, Chirag et al. (2015). "Modified Early Warning System improves patient safety and clinical outcomes in an academic community hospital". In: *Journal of community hospital internal medicine perspectives* 5.2, p. 26716.

Milakis, Dimitris, Bart Van Arem, and Bert Van Wee (2017). "Policy and society related implications of automated driving: A review of literature and directions for future research". In: *Journal of Intelligent Transportation Systems* 21.4, pp. 324–348.

Miyagawa, Taiki and Akinori F Ebihara (2021). "The Power of Log-Sum-Exp: Sequential Density Ratio Matrix Estimation for Speed-Accuracy Optimization". In: *International Conference on Machine Learning*. PMLR, pp. 7792–7804.

Mocaër, William, Eric Anquetil, and Richard Kulpa (2021). "Online Spatio-Temporal 3D Convolutional Neural Network for Early Recognition of Handwritten Gestures". In: *ICDAR 2021-16th International Conference on Document Analysis and Recognition*.

Mori, Usue, A Mendiburu, et al. (2015). "Early classification of time series from a cost minimization point of view". In: *Proceedings of the NIPS Time Series Workshop*.

Mori, Usue, Alexander Mendiburu, Sanjoy Dasgupta, et al. (2017). "Early classification of time series by simultaneously optimizing the accuracy and earliness". In: *IEEE transactions on neural networks and learning systems* 29.10, pp. 4569–4578.

Mori, Usue, Alexander Mendiburu, Eamonn Keogh, et al. (2017). "Reliable early classification of time series based on discriminating the classes over time". In: *Data mining and knowledge discovery* 31.1, pp. 233–263.

Mori, Usue, Alexander Mendiburu, Isabel Marta Miranda, et al. (2019). "Early classification of time series using multi-objective optimization techniques". In: *Information Sciences* 492, pp. 204–218.

Muthukrishna, Daniel et al. (2019). "RAPID: early classification of explosive transients using deep learning". In: *Publications of the Astronomical Society of the Pacific* 131.1005, p. 118002.

Nath, Aneesh G et al. (2020). "An early classification approach for improving structural rotor fault diagnosis". In: *IEEE Transactions on Instrumentation and Measurement* 70, pp. 1–13.

Nemenyi, Peter (1962). "Distribution-free multiple comparisons". In: *Biometrics* 18.2, p. 263.

Nodet, Pierre, Vincent Lemaire, Alexis Bondu, and Antoine Cornuéjols (2020). "Importance Reweighting for Biquality Learning". In: *arXiv preprint arXiv:2010.09621*.

Nodet, Pierre, Vincent Lemaire, Alexis Bondu, Antoine Cornuéjols, and Adam Ouorou (2021). "From Weakly Supervised Learning to Biquality Learning: an Introduction". In: *In Proceedings of the International Joint Conference on Neural Networks (IJCNN)*.

Novikov, Andrey (2008). "Optimal sequential tests for two simple hypotheses based on independent observations". In: *International journal of pure and applied mathematics* 45.2, pp. 291–314.

Óskarsdóttir, Maria et al. (2018). "Time series for early churn detection: Using similarity based classification for dynamic networks". In: *Expert Systems with Applications* 106, pp. 55–65.

P., Neilson J. (2015). "Fetal electrocardiogram (ECG) for fetal monitoring during labour". In: *The Cochrane database of systematic reviews* 12.

Parrish, Nathan et al. (2013). "Classifying with confidence from incomplete information". In: *J. of Mach. Learning Research* 14.1, pp. 3561–3589.

Quiñonero-Candela, Joaquin et al. (2009). *Dataset shift in machine learning*. Mit Press.

Rakthanmanon, T and E Keogh (2011). "Fast-shapelets: A fast algorithm for discovering robust time series shapelets". In: *Proceedings of 11th SIAM international conference on data mining*.

Ran, Yongyi et al. (2019). "A survey of predictive maintenance: Systems, purposes and approaches". In: *arXiv preprint arXiv:1912.07383*.

Ruff, Lukas et al. (2021). "A Unifying Review of Deep and Shallow Anomaly Detection". In: *Proc. IEEE* 109.5, pp. 756–795. DOI: 10.1109/JPROC.2021.3052449. URL: https://doi.org/10.1109/JPROC.2021.3052449.

Rußwurm, Marc, Sébastien Lefevre, et al. (2019). "End-to-end learning for early classification of time series". In: *arXiv preprint arXiv:1901.10681*.

Rußwurm, Marc, Romain Tavenard, et al. (2019). "Early classification for agricultural monitoring from satellite time series". In: *arXiv preprint arXiv:1908.10283*.

Sales, Daniel O et al. (2014). "Adaptive finite state machine based visual autonomous navigation system". In: *Engineering Applications of Artificial Intelligence* 29, pp. 152–162.

Samuel, Dinesh Jackson and Fabio Cuzzolin (2021). "Unsupervised Anomaly Detection for a Smart Autonomous Robotic Assistant Surgeon (SARAS) Using a Deep Residual Autoencoder". In: *IEEE Robotics and Automation Letters* 6.4, pp. 7256–7261. DOI: 10.1109/LRA.2021.3097244.

Schäfer, Patrick (2015). "The BOSS is concerned with time series classification in the presence of noise". In: *Data Mining and Knowledge Discovery* 29.6, pp. 1505–1530.

Schäfer, Patrick and Ulf Leser (2020). "TEASER: early and accurate time series classification". In: *Data Mining and Knowledge Discovery* 34.5, pp. 1336–1362.

Seeger, Matthias (2000). "Learning with labeled and unlabeled data". In.

Senin, Pavel (2008). "Dynamic time warping algorithm review". In: *Information and Computer Science Department University of Hawaii at Manoa Honolulu, USA* 855.1-23, p. 40.

Senin, Pavel and Sergey Malinchik (2013). "Sax-vsm: Interpretable time series classification using sax and vector space model". In: *2013 IEEE 13th international conference on data mining*. IEEE, pp. 1175–1180.

Serban, Alexandru Constantin, Erik Poll, and Joost Visser (2018). "A standard driven software architecture for fully autonomous vehicles". In: *2018 IEEE International Conference on Software Architecture Companion (ICSA-C)*. IEEE, pp. 120–127.

Settles, Burr (2009). "Active learning literature survey". In.

Sharma, Anshul and Sanjay Kumar Singh (2020). "A novel approach for early malware detection". In: *Transactions on Emerging Telecommunications Technologies*, e3968.

Shekhar, Shubhranshu et al. (2021). "Benefit-aware Early Prediction of Health Outcomes on Multivariate EEG Time Series". In: *arXiv preprint arXiv:2111.06032*.

Shepp, Larry A (1969). "Explicit solutions to some problems of optimal stopping". In: *The Annals of Mathematical Statistics* 40.3, p. 993.

Silva, Jonathan A et al. (2013). "Data stream clustering: A survey". In: *ACM Computing Surveys (CSUR)* 46.1, pp. 1–31.

Singh, Naman et al. (2018). "Detection of fake profile in online social networks using machine learning". In: *2018 International Conference on Advances in Computing and Communication Engineering (ICACCE)*. IEEE, pp. 231–234.

Singh, Ravinder et al. (2019). "A framework for early detection of antisocial behavior on Twitter using natural language processing". In: *Conference on Complex, Intelligent, and Software Intensive Systems*. Springer, pp. 484–495.

Stefan, Alexandra, Vassilis Athitsos, and Gautam Das (2012). "The move-split-merge metric for time series". In: *IEEE transactions on Knowledge and Data Engineering* 25.6, pp. 1425–1438.

Sutton, Richard S and Andrew G Barto (2018). *Reinforcement learning: An introduction*. MIT press.

Tan, Chang Wei et al. (2021). "Time series extrinsic regression". In: *Data Mining and Knowledge Discovery* 35.3, pp. 1032–1060.

Tatbul, Nesime et al. (2018). "Precision and Recall for Time Series". In: *Advances in Neural Information Processing Systems*. Ed. by S. Bengio et al. Vol. 31. Curran Associates, Inc.

Teinemaa, Irene et al. (2018). "Alarm-based prescriptive process monitoring". In: *International Conference on Business Process Management*. Springer, pp. 91–107.

Thrun, Sebastian (2010). "Toward robotic cars". In: *Communications of the ACM* 53.4, pp. 99–106.

Torkamani, Sahar and Volker Lohweg (2017). "Survey on time series motif discovery". In: *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery* 7.2, e1199.

Vapnik, Vladimir and Akshay Vashist (2009). "A new learning paradigm: Learning using privileged information". In: *Neural networks* 22.5-6, pp. 544–557.

Vishnukumar, Harsha Jakkanahalli et al. (2017). "Machine learning and deep neural network—Artificial intelligence core for lab and real-world test and validation for ADAS and autonomous vehicles: AI for efficient and quality test and validation". In: *2017 Intelligent Systems Conference (IntelliSys)*. IEEE, pp. 714–721.

Wald, Abraham and Jacob Wolfowitz (1948). "Optimum character of the sequential probability ratio test". In: *The Annals of Mathematical Statistics*, pp. 326–339.

Wang, Wenlin et al. (2016). "Earliness-aware deep convolutional networks for early time series classification". In: *arXiv preprint arXiv:1611.04578*.

Watanabe, Hajime, Mondher Bouazizi, and Tomoaki Ohtsuki (2018). "Hate speech on twitter: A pragmatic approach to collect hateful and offensive expressions and perform hate speech detection". In: *IEEE access* 6, pp. 13825–13835.

Xia, Rui, Kaizhou Xuan, and Jianfei Yu (2020). "A State-independent and Time-evolving Network with Applications to Early Rumor Detection". In: *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 9042–9051.

Xing, Zhengzheng, Jian Pei, and S Yu Philip (2009). "Early prediction on time series: A nearest neighbor approach". In: *Twenty-First International Joint Conference on Artificial Intelligence*. Citeseer.

Xing, Zhengzheng, Jian Pei, Philip S Yu, et al. (2011). "Extracting interpretable features for early classification on time series". In: *Proceedings of the 2011 SIAM international conference on data mining*. SIAM, pp. 247–258.

Ye, Lexiang and Eamonn Keogh (2009). "Time series shapelets: a new primitive for data mining". In: *Proceedings of the 15th ACM SIGKDD international conference on Knowledge discovery and data mining*, pp. 947–956.

— (2011). "Time series shapelets: a novel technique that allows accurate, interpretable and fast classification". In: *Data mining and knowledge discovery* 22.1, pp. 149–182.

Zhao, Lei et al. (2019). "Asynchronous Multivariate time series early prediction for ICU transfer". In: *Proceedings of the 2019 International Conference on Intelligent Medicine and Health*, pp. 17–22.

Zhdanov, Fedor (2019). "Diverse mini-batch Active Learning". In: *arXiv:1901.05954 [cs.LG]*.

Zhou, Kaimin et al. (2019). "Early rumour detection". In: *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)*, pp. 1614–1623.

Zhou, Xinyi et al. (2020). "Fake news early detection: A theory-driven model". In: *Digital Threats: Research and Practice* 1.2, pp. 1–25.

Zhou, Zhi-Hua (2018). "A brief introduction to weakly supervised learning". In: *National science review* 5.1, pp. 44–53.

Zhuang, Fuzhen et al. (2020). "A comprehensive survey on transfer learning". In: *Proceedings of the IEEE* 109.1, pp. 43–76.

Zoppi, Tommaso et al. (Apr. 2021). "Unsupervised Anomaly Detectors to Detect Intrusions in the Current Threat Landscape". In: *ACM/IMS Trans. Data Sci.* 2.2. ISSN: 2691-1922. DOI: 10.1145/3441140. URL: https://doi.org/10.1145/3441140.