PROGRAMA DE DOCTORADO EN ESTADÍSTICA, OPTIMIZACIÓN Y MATEMÁTICA APLICADA

Incremental algorithm for Decision Rule generation in data stream contexts

Nuria Mollá Campello

Director de la tesis
Dr. D. Alejandro Rabasa Dolado

Codirectores de la tesis
Dr. D. Joaquín Sánchez Soriano
Dr. D. Antonio Ferrándiz Colmeiro

Universidad Miguel Hernández de Elche

2022

La presente tesis doctoral se presenta en la modalidad convencional y como indicios de calidad se refiere a los siguientes resultados:

1. Mollá, N.; Rabasa, A.; Rodríguez-Sala, J.J.; Sánchez-Soriano, J. & Ferrándiz, A. (2022) Incremental Decision Rules Algorithm: A Probabilistic and Dynamic Approach to Decisional Data Stream Problems. Mathematics. 10, 16. https://doi.org/10.3390/math10010016.
2. Mollá, N.; Heavin, C. & Rabasa, A. (2022). Data-driven decision making: new opportunities for DSS in data stream contexts, Journal of Decision Systems, DOI: 10.1080/12460125.2022.2071404.

Universidad Miguel Hernández de Elche

D. Alejandro Rabasa Dolado, director de esta tesis doctoral, declaro que el presente trabajo, titulado

Incremental algorithm for Decision Rule generation in data stream contexts

presentado por Doña Nuria Mollá Campello para obtener el título de doctora, fue llevado a cabo bajo mi supervisión en el Programa de Doctorado en Estadística, Optimización y Matemática Aplicada (EOMA) de la Universidad Miguel Hernández de Elche.

Elche, diciembre de 2022

El director,

Dr. Alejandro Rabasa Dolado

Universidad Miguel Hernández de Elche

D. Joaquín Sánchez Soriano, codirector de esta tesis doctoral, declaro que el presente trabajo, titulado

Incremental algorithm for Decision Rule generation in data stream contexts

presentado por Doña Nuria Mollá Campello para obtener el título de doctora, fue llevado a cabo bajo mi supervisión en el Programa de Doctorado en Estadística, Optimización y Matemática Aplicada (EOMA) de la Universidad Miguel Hernández de Elche.

Elche, diciembre de 2022

El codirector,

Dr. Joaquín Sánchez Soriano

Teralco Group

D. Antonio Ferrándiz Colmeiro, codirector de esta tesis doctoral, declaro que el presente trabajo, titulado

Incremental algorithm for Decision Rule generation in data stream contexts

presentado por Doña Nuria Mollá Campello para obtener el título de doctora, fue llevado a cabo bajo mi supervisión en el Programa de Doctorado en Estadística, Optimización y Matemática Aplicada (EOMA) de la Universidad Miguel Hernández de Elche.

Elche, diciembre de 2022

El codirector,

Dr. Antonio Ferrándiz Colmeiro

Universidad Miguel Hernández de Elche

D. Domingo Morales González, coordinador del Programa de Doctorado en Estadística, Optimización y Matemática Aplicada (EOMA) de la Universidad Miguel Hernández de Elche, declaro que el presente trabajo, titulado

Incremental algorithm for Decision Rule generation in data stream contexts

presentado por Doña Nuria Mollá Campello para obtener el título de doctora, fue llevado a cabo en el Programa de Doctorado en Estadística, Optimización y Matemática Aplicada (EOMA) de la Universidad Miguel Hernández de Elche.

Elche, diciembre de 2022

El coordinador del programa de doctorado,

Dr. Domingo Morales González

Universidad Miguel Hernández de Elche y Teralco Solutions

El trabajo presentado se realiza en el marco de un convenio de colaboración entre la Universidad Miguel Hernández de Elche y la compañía Teralco Solutions S.L. Para la realización de esta tesis se cuenta con una ayuda contemplada en el Programa Estatal de Promoción del Talento y su Empleabilidad, en el marco del Plan Estatal de Investigación Científica y Técnica y de Innovación 2017-2020, Ayudas para la formación de doctores en empresas "Doctorados Industriales".

Se reconoce la inestimable ayuda del Ministerio de Ciencia, innovación y universidades MCIN/AEI/10.13039/501100011033 y de Teralco Solutions, a través de la beca con identificación DIN2018-010101 para la formación de doctores en empresas.

El pulso largo y sin medida del tiempo todo lo mueve. No hay nada oculto que no pueda sacar a la luz, ni nada conocido que no pueda volver a la oscuridad.

Sófocles.

Agradecimientos

Para todas las personas que me han acompañado en este proceso y lo han hecho posible. A mi madre Manoli y a mi padre José Luis, que me han dado la oportunidad de estudiar con su esfuerzo y trabajo, más del que podré agradecerles nunca. Gracias por ser un ejemplo y por darme los valores y la valentía de ser lo que soy hoy. A mi hermano, que ha compartido penas y glorias conmigo, espero ver pronto tu firma en un proyecto mucho más grande que este. A mis abuelos, por todo el amor que me han hecho sentir en todos estos años. Gracias por demostrarlo cada sábado desde que tengo uso de razón, y gracias por venir hasta Irlanda para verme, con 84 años y cargando paraguas. A mi tía Lola, que siempre ha sido y será un referente de la persona que quiero ser. Gracias tía, por tu coherencia y tu ejemplo, gracias por ser valiente cuando era imposible serlo, por luchar para que hoy, mujeres como yo podamos firmar trabajos como este. Es un orgullo tenerte a mi lado. A mis tíos y tías, que me acompañan cada sábado en el campo y me hacen sentir su apoyo. A mi tía Lumi y mi iaia Amada, que me acompañan cada día en lo que soy hoy. A mis primas y primos, por los momentos bonitos que hemos compartido y por todos los que se vienen. A mi tía Ana y Elena, por dar a quienes necesitan y por estar orgullosas de ser quien son. A toda mi familia, gracias.

A mis amigas: A las que conozco desde siempre, a las que parece que conozca desde siempre y a las que no conozco todavía. A las mayores, a las más jóvenes y a las de mi edad. A las que veo todas las semanas y a las que veo una vez al mes, o cada seis meses porque están estudiando o cada dos años porque están viajando. A las que vivís conmigo el día a día y a las que vivís fuera pero decidís compartir vuestro tiempo igualmente. Sois mi orgullo y mi energía, gracias.

A mis directores por el apoyo y el trabajo de todos estos meses, especialmente a Álex por años de investigación sin más jerarquías que la que impone tu conocimiento sobre el tema. Gracias a los tres por la confianza y la libertad, ha sido un honor trabajar a vuestro lado y espero que sigamos ese camino. A todas las compañeras de la Universidad Miguel Hernández con las que he podido caminar, tanto las que iban por delante como las que van por detrás. A las compañeras de Teralco que han hecho mi trabajo más fácil, y a las personas responsables de que esto haya ocurrido. Gracias por la confianza, ahora empieza lo divertido.

Thanks to Cork and its people, for the many lessons I have learnt there, and for making my stay so easy. Especially, thanks to Professor Ciara Heavin, who gave me the opportunity to work with her and taught me many things in a professional and personal way. Thank you Ciara, for your honesty and your support. It has been a real pleasure and an honor to research next to you. I hope this is the first step of a long path together. To all the people I met in Cork, those who will stay in my life and those who I will keep in my mind, thank you. Et merci à toi, Juliette, pour accompagner dans la fin de ma thèse. Pour ton soutien et ton amour, pour les voyages, passés et futurs. Merci.

Index

Figures

Tables

List of acronyms

ADWIN	Adaptive Window
API	Application Programming Interface
ASDW	Active Stream Data Warehouse
BA	Business Analytics
BI	Business Intelligence
CNN	Convolutional Neural Network
CPU	Central Processing Unit
CREA	Classification Rules Extraction Algorithm
CVFDT	Concept-adapting Very Fast Decision Tree
DAKAR	Discovery of Actionable Knowledge And Recommendations
DL	Deep Learning
DR	Decision Rules
DRS	Decision Rule Set
DSM	Data Stream Mining
DSS	Decision Support System
DT	Decision Tree
Ec_i	Eje confianza inferior
Ec_s	Eje confianza superior
EFDT	Extremely Fast Decision Tree
Es_i	Eje soporte inferior
Es_s	Eje soporte superior
EVFDT	Efficient Very Fast Decision Tree
FACIL	Fast and Adaptive Classifier by Incremental Learning
FHDT	Fuzzy Hoeffding Decision Tree
GAHT	Green Accelerated Hoeffding Tree
IBL	Instance-Based Learning
ID3	Iterative Dichotomiser 3
IDRA	Incremental Decision Rules Algorithm
IDRAv1	Incremental Decision Rules Algorithm version 1
IDRAv2	Incremental Decision Rules Algorithm versión 2
KNN	K-Nearest Neighbour
KPI	Key Performance Indicators
ML	Machine Learning
MNBT	Minimum/Maximum Norms-based Binary Tree
MOA	Massive Online Analysis
NBT	Norm-based Binary Tree
NFR	Non-Final Rule
NN	Neural Network
NTER	Not Totally Expanded Rules
OKR	Objective Key Results
PRS	Potential Rule Set
RAM	Random Access Memory
RBS	Rules Based on Significance

RIDRA	Reactive Incremental Decision Rules Algorithm
RS	Rule set
SAMOA	Scalable Advanced Massive Online Analysis
SEA	Streaming Ensemble Algorithm
SVM	Support Vector Machine
TV	Television
UINP	Uneven Interval Numerical Pruning
VFDR	Very Fast Decision Rules
VFDT	Very Fast Decision Trees
VFDTc	Very Fast Decision Trees for continuous attributes
VFDTcNB	Very Fast Decision Trees for continuous attributes Naïve Bayes
VFML	Very Fast Machine Learning

Resumen

Actualmente, la ciencia de datos está ganando mucha atención en diferentes sectores. Concretamente en la industria, muchas aplicaciones pueden ser consideradas. Utilizar técnicas de ciencia de datos en el proceso de toma de decisiones es una de esas aplicaciones que pueden aportar valor a la industria. El incremento de la disponibilidad de los datos y de la aparición de flujos continuos en forma de data streams hace emerger nuevos retos a la hora de trabajar con datos cambiantes. Este trabajo presenta una propuesta innovadora, Incremental Decision Rules Algorithm (IDRA), un algoritmo que, de manera incremental, genera y modifica reglas de decisión para entornos de data stream para incorporar cambios que puedan aparecer a lo largo del tiempo. Este método busca proponer una nueva estructura de reglas que busca mejorar el proceso de toma de decisiones, planteando una base de conocimiento descriptiva y transparente que pueda ser integrada en una herramienta decisional. Esta tesis describe la lógica existente bajo la propuesta de IDRA, en todas sus versiones, y propone una variedad de experimentos para compararlas con un método clásico (CREA) y un método adaptativo (VFDR). Conjuntos de datos reales, juntamente con algunos escenarios simulados con diferentes tipos y ratios de error, se utilizan para comparar estos algoritmos. El estudio prueba que IDRA, específicamente la versión reactiva de IDRA (RIDRA), mejora la precisión de VFDR y CREA en todos los escenarios, tanto reales como simulados, a cambio de un incremento en el tiempo.

Abstract

Nowadays, data science is earning a lot of attention in many different sectors. Specifically in the industry, many applications might be considered. Using data science techniques in the decision-making process is a valuable approach among the mentioned applications. Along with this, the growth of data availability and the appearance of continuous data flows in the form of data stream arise other challenges when dealing with changing data. This work presents a novel proposal of an algorithm, Incremental Decision Rules Algorithm (IDRA), that incrementally generates and modify decision rules for data stream contexts to incorporate the changes that could appear over time. This method aims to propose new rule structures that improve the decision-making process by providing a descriptive and transparent base of knowledge that could be integrated in a decision tool. This work describes the logic underneath IDRA, in all its versions, and proposes a variety of experiments to compare them with a classical method (CREA) and an adaptive method (VFDR). Some real datasets, together with some simulated scenarios with different error types and rates are used to compare these algorithms. The study proved that IDRA, specifically the reactive version of IDRA (RIDRA), improves the accuracies of VFDR and CREA in all the studied scenarios, both real and simulated, in exchange of more time.

Chapter 1
Introduction

This chapter comprises a brief introduction to the problem that wants to be solved in this dissertation. The many different problems and challenges that arise within the continuous analysis of data streams are described, as well as the main logic that lead the knowledge for these environments and the characteristics of the systems that are meant to contain the solutions proposed in this work.

Also, this chapter contains an explanation of the main objectives that are aimed with this research, together with a brief definition of the proposed solution and the materials used and a description of the structure and results of the presented work.

Some of the content of this chapter is based on the publications of the journal Mathematics titled *Incremental Decision Rules Algorithm: A Probabilistic and Dynamic Approach to Decisional Data Stream Problems* (Mollá et al. 2022b) and the Journal of Decision Systems titled *Data-driven decision making: new opportunities for DSS in data stream contexts* (Mollá et al. 2022a). The first publication presents the introduction and problem definition used to justify the initial version of the proposed method, IDRAv1. The second publication compiles information about the Decision Support Systems (DSS) and their opportunities in data stream contexts. A combined and extended version of these two initial works is presented in this chapter.

1.1. Introduction to the problem

The introduction of technology in every field of society implies that the amount of data generated is increasing widely and quickly. The use of this information to select the best option in each moment is a field of study with a great interest for researchers and companies. Data analysis has been changing fast and improving the decision-making process until the development of powerful engines able to make highly accurate predictions for given situations. These technologies have been evolving fast along with the generation rate of data and the need to exploit them. Nowadays, solutions are being implemented to extract knowledge and optimise processes in many fields such as industrial systems (Alzghoul and Löfstrand, 2011), product rating (Anto et al., 2016) or supply chain management (Han and Zhang, 2021). Also, the increase of the multimedia content that can be consumed online generates huge amounts of information that can be analysed through different techniques (Deng and Li, 2019; Reed and Kranch, 2017). Additionally, the growth of social networks and their use, along with the great opportunities and huge amount of data produced by these applications have favoured the inclusion of this data in a great number of studies. In this sense, many sentiments or opinion analysis (Rodrigues and Chiplunkar, 2019; Selvan and Moh, 2015; Anto et al., 2016) or prediction tasks applied to many sectors from TV ratings (Egebjerg et al. 2017) to road traffic (Alkouz and Al Aghbari, 2019), drug safety (Alharbi et al. 2017) or cyber-security (Altalhi and Gutub, 2021). These successful cases, together with many more favourable examples, have drawn the attention over the artificial intelligence and machine learning techniques. The benefits and opportunities that have been exposed in the last years have raised the interest in these fields by the companies.

The massive amount of data generated in these cases need to be stored in a system prepared to work with it generating knowledge and dealing with it in a meaningful way. We introduce here the term business analytics (BA), a set of skills, techniques and technologies that use data and statistical methods from previous business performance to predict the trend and expected outcomes and consequently create a business plan. Business analytics includes techniques such as data mining, statistical analysis, and predictive modelling to provide a knowledge base from historical and present data that help make informed decisions. Evans and Lindner (2012) define business analytics as the convergence of three main disciplines: statistics, business intelligence and information systems. Business intelligence (BI) focusses on the description of data that could be used to guide future actions. Thus, we can differentiate the terms by saying that BI proposes a descriptive approach, while business analytics focuses on the whole process, including prediction and prescription (Tableau, 2021). The descriptive analysis is the use of quantitative information to summarise features that could be read and lead to relevant conclusions. On the contrary, predictive analytics is a variety of statistical techniques that aim to analyse and make predictions about future events. Data mining, statistical predictive modelling and machine learning are some of the fields included in this branch of analytics. In business, it is especially

relevant to consider and identify risks and opportunities within these predictive analytics. Studying the relationships among some factors allows the decision maker to know and explain better the context. The last type of analysis that we could highlight is prescriptive analytics. These methods try to optimise an objective variable by maximising or minimising some key values. To do that, they will prescribe necessary actions to improve the results and reduce risks of a given business. In summary, the descriptive analytics will represent information to be able to define what could we analyse and how to do it; the predictive analytics will try to forecast that *what* and the prescriptive analytics will indicate how to act to aim it. Many of these analytics could combine in some applications, involving more than one single analytic type. In this sense, the work presented in this dissertation belongs to predictive analytics, since the algorithm aims to classify instances of an objective variable. Nevertheless, the application of this knowledge to a DSS that aims to advise decision makers in their actions and risks will make it part of the prescriptive analytics.

In this sense, a new dimension is introduced in this analysis, the time. In the mathematical field, time series started working with this new dimension, considering time as a critical factor to study the trends of information. Thus, time series could be defined as sequences of data ordered by time measures and which order is important to understand the whole system logic. Time series analysis studies the statistics of a time series to extract the meaningful patterns found in data. Several characteristics could be detected in time series analysis, such as trend, stationarity or seasonality. These characteristics could help the modelling process during the time series forecasting phase. These methods try to predict future values based on previously observed examples, considering the evolution of the time series. These predictions may be framed in fields such as statistics, signal processing, pattern recognition, weather forecasting or other applications within machine learning or data mining.

The data generation and acquisition process has changed significantly during the last years due to the improvement of data collection and computational power. Elements such as the data availability, the computational power and the statistical tools will define the capacity of an analytic system (see Figure 1.1). The opportunities that arise with technological advancement need to be addressed considering the specific characteristics of these new scenarios to optimise the processes. The rise of the speed and amount of data, combined with online availability in the form of a stream, entails a great opportunity to improve data-driven decision systems. In this new situation, with a larger amount of data available online, the systems have the opportunity to continually integrate these sources into the base of knowledge.

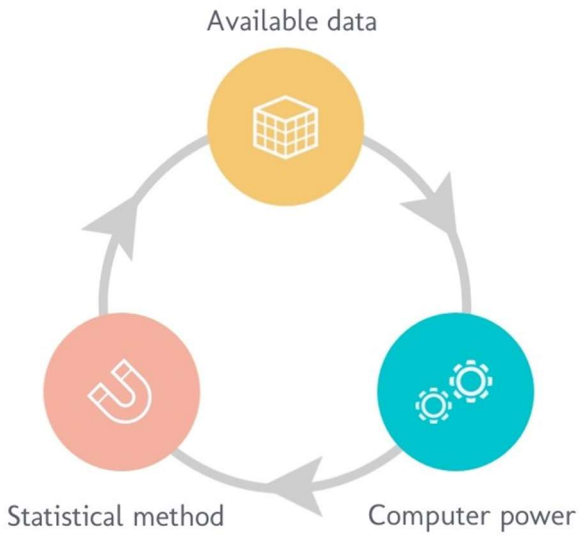


Figure 1.1. Relation between elements in an analytic system.

This new type of data, known as data streams or data flows could be understood as the mentioned concept of time series, but with some significant differences between these two concepts. A data stream is a flow of examples received continuously by a system over time. Despite data stream analysis considers time as a dimension for their analysis of the data flows, this is not a requirement, contrary to time series analysis in which this is one of the main attributes of an analysis. Another difference between these two fields is the data structure, while time series data is previously stored in a system, a data stream is a potentially infinite, high-speed, non-stationary flow that comes continuously into a system and cannot be stored. This incapability to keep the data stream in memory, require statistical and forecasting methods that aggregate new seen data to the system without the need to ever access it again. The machine learning traditional approach provides batch processing with finite training sets while, for data stream approach the model update is accomplished through potentially infinite training sets (see Figure 1.2).

Many of the artificial intelligence or machine learning techniques use vast amounts of data to generate bases of knowledge later used to estimate or optimise processes. But these bases of knowledge are, generally, based on previous batch data. The vast amount of data being generated by some systems make it necessary to introduce continuous analysis of the data streams. These new approaches seem to have a great potential in highly precision demanding or uncertain scenarios. The possibility of an incremental or adaptive base of knowledge that led decisions in these cases, might reduce the error of the system and arise a wide range of opportunities for an information system. These data stream mining techniques face challenges for the online contexts such as incorporating new instances on the fly, unbounded training sets and changes in data distribution. The potential change in data distribution might arise anytime in a data stream and being able to detect and adapt the model to these concept drifts is a challenge in the data stream mining field.

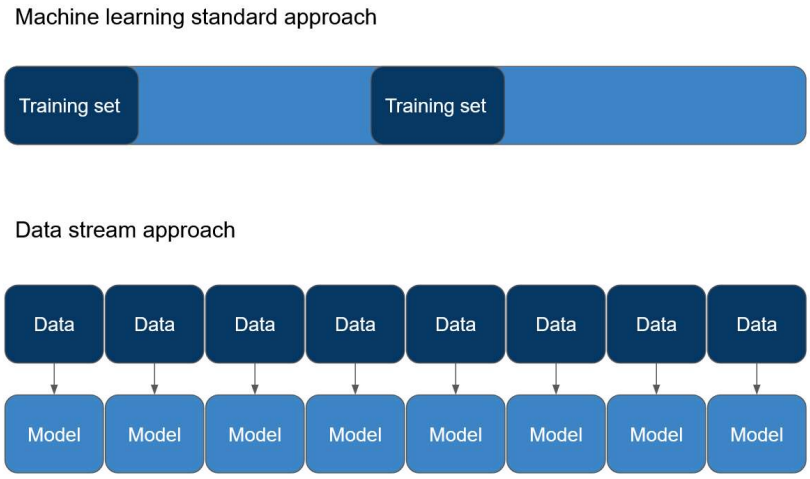


Figure 1.2 Comparison of ML standard approach and Data stream approach (Source: Bifet, 2021).

1.1.1. Online learning concepts and techniques

Standard supervised machine learning techniques used to analyse batch data follow premises defined along the years. Nevertheless, these traditional approaches face new challenges when analysing data streams due to their unbounded structure, their potentially infinite size and the exposure to drifts in data distributions.

Besides the data stream context, this work is focused on the supervised learning techniques, particularly on the classification techniques. In this sense, the new challenges raised by the new structure of data, change the needs of the tools used to analyse it within a system. New logic have to be considered to optimise the analytic purposes of data stream mining. Previous contributions made for the traditional and the online approach are extended in Chapter 2.

Apart from the differences in acquisition method and in the data nature, the main difference between the batch processing and the data stream approach are concept drifts. This term primarily refers to change of the relation between the input data and the target variable over time. Many different types of drifts can be observed in a data stream depending on different factors such as the speed of change or the affected variable (Gama et al., 2014).

Two types of drifts can be found in the literature, the real concept drift and the virtual drift. The real concept drift refers to a change in the probability of a label for the target variable with or without a change in the input set of variables, while the virtual drift refers to a change in the input data that does not affect the probability of the label target variable. Virtual drift was defined by Widmer and Kubat (1993) and is understood as a temporal misrepresentation of data due to incompleteness that might lead to changes in the decision boundary (Tsymbal, 2004) but in reality, has no effect on the target concept (Delany et al., 2005).

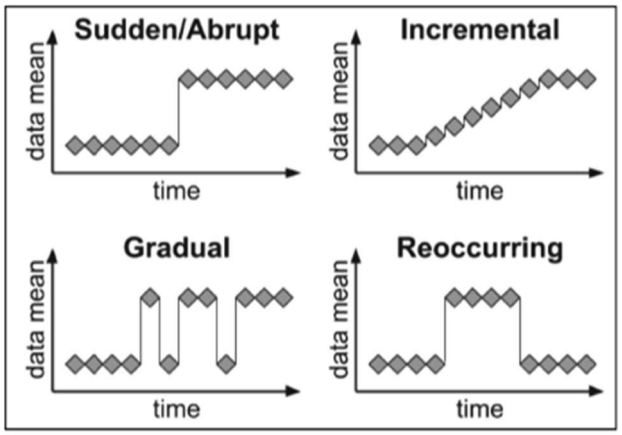


Figure 1.3 Types of concept drifts over time (Source: Gama et al., 2014).

Different changes in data distribution over time can have different impacts over a decisional system. The main patterns of changes seen in data over time are illustrated in Figure 1.3. It is important to differentiate elements such as the noise of a system or outlier values from a concept drift (Chandola et al., 2009). The different types of concept drifts are characterised by the nature of the change and its speed. The abrupt concept drift is a drift that happens suddenly in the system while the incremental and gradual drifts happen progressively for continuous or discrete variables.

From this problem, the necessity of detecting and adapting to these changes emerges to avoid the degradation of the model integrated in a data-driven system. These predictive models are most times required to (1) detect the concept drift and adapt, if necessary, (2) differentiate drifts from noise and (3) compute examples faster than time arrival with memory constraints. From these needs, the idea of the online adaptive learning procedure emerges and it is formally defined by Gama et al. (2014) as follows. A decision model is a function L that maps the input variables (X) to the target (y): $y = L(X)$. A learning algorithm specifies how to build a model from a set of data instances. The online adaptive learning procedure (1) predicts a value for the target variable ($\hat{y}$) when a new instance arrives using model L_t. After some time, when the true label of y_t is received, the (2) diagnosis can be carried out by estimating the loss or error of the system as $f(\hat{y}_t, y_t)$. After this, the seen example (X_t, y_t) is used to update the model L_{t+1}. The updating process and the computational resources needed differ between methods. Handling data streams online can be approached in different ways such as a partial memory access that allows the system to regularly access data to train; a sliding window where data is structured as chunks; or instance integration where an example is processed on the fly right when it arrives. The best scheme might be selected by considering aspects like the memory restrictions or the structure of the system. Elements such as the learning process, the updating method, the estimation loss function or the change detection method are part of the online adaptive learning procedure. These elements are different from one method or algorithm to another and define their suitability for a specific problem.

The online adaptive learning procedure is a learning process for data streams also defined as incremental or adaptive learning. Both terms, incremental learning and adaptive learning, use data streams to integrate knowledge in a system following similar but slightly different approaches (Mollá et al., 2022a). We refer to incremental learning when a method of machine learning continuously integrates knowledge in the system. In this sense, adaptive learning is similar to incremental learning but with a slight difference in the requirements of the system. Incremental algorithms need some parametric definition made by the user, while adaptive algorithms automatically estimate those parameters to the best fitting ones in each moment depending on data distribution.

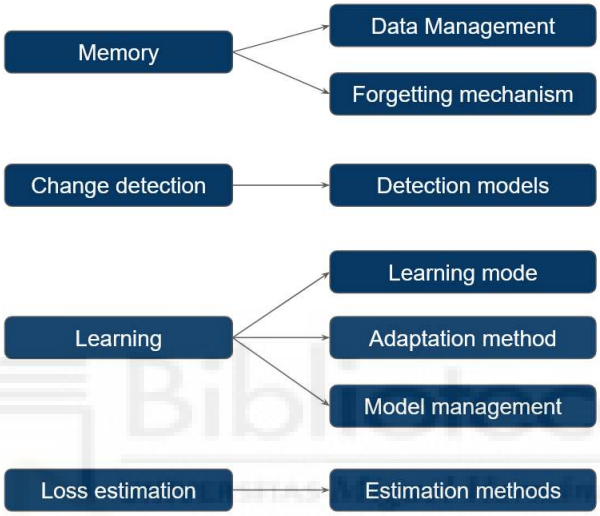


Figure 1.4 Modules of adaptive learning systems (Source: Gama et al., 2014)

Figure 1.4 shows the modules of adaptive learning systems proposed by Gama et al. (2014). In the next paragraphs the most important characteristics of the memory, the change detection, the learning and the loss estimation processes are presented and discussed following the premises of the referred work.

Memory

In an online system, the memory is a key factor that needs to be considered and designed based on system characteristics. Regarding data management, the assumption that the most recent data is the most valuable for predictions, makes adaptive models try to learn from the last available data. The system can store either one single example in memory, for algorithms that continuously learn from streams with no need to access previous examples later; or can store multiple examples, for those algorithms that need to retrieve older examples at some point. The whole data stream cannot be stored in memory due to its potential infinitude, so only a part of the data can be kept for future needs, this is defined as a time window. All the instances that move out of the window

are erased from the system. In this case, determining an appropriate size for this fixed sized window becomes a challenge. A short window can help having a more accurate representation of data during uncertain or changing periods, but during stable periods a larger window might get a higher accuracy rate. Trying to solve this situation some systems use sliding windows with variable size over time.

Since an online system is exposed to continuous incoming data, some forgetting mechanism might be needed in order to erase the deprecated knowledge. The abrupt forgetting refers to the approach in which the previous examples are either inside or outside of the sliding window. Thus, the system will consider only those examples included in the window with no difference among them. In the case of the gradual forgetting approach, the whole memory is considered in the system but with different weights associated with their age. In this case, this method assumes that the most recent cases are the most valuable and therefore they have the greater weights. The value and weight of an example is then associated with its age in a descending order.

Change detection

Change detection refers to those mechanisms that search for the explicit detection of concept drift in data distribution. These techniques try to determine the changing points and the type of change occurring in a given moment in time (Basseville and Nikiforov, 1993). Different dimensions are considered for change detection, such as (1) methods based on sequential analysis, (2) methods based on control charts, (3) methods based on differences between distributions on two different time windows, and (4) heuristic methods. It is also possible that some adaptive or incremental algorithms have no explicit change detection mechanism. The advantage of including an explicit change detection method is the faster response and the available information in the system.

Learning

The learning process refers to the generalisation of knowledge based on previous seen examples. Specifically for data stream contexts, the learning process also includes an updating mechanism that allows the model to adapt to changes. Thus, a learning model is generated and updated after a new label or set of labels are available. This update can be done through different approaches: (1) retraining, that discards previous model and builds a new one from scratch with the data available, and (2) incremental, that adapts the model over time. Retraining methods need data to be kept in memory to remodel, normally using batch-learning approaches. An initial model is built with all the available data, and then when new data arrives to the system, the last model is discarded and a new one is trained with the new data and old data (Zeira et al. 2004). In the case of the incremental approach, the updating of the system is made every time after receiving a new example without losing the previous knowledge. They might need to access previous data or summary of the data. Within the incremental logic we

can also find online learning, which updates the current model depending on whether it misclassified the most recent example; and streaming learning, that are online algorithms for processing high-speed data streams.

After the learning process, the adaptation quality needs to be addressed. The adaptation methods can be whether (1) blind or (2) informed. The blind adaptation refers to that process that is made with no explicit change detection. These methods normally use fixed-size sliding windows to adapt the system after a given number of examples, or example weighting that assigns the importance of the examples depending on their age. The informed adaptation follows a reactive strategy and uses triggers to act when a change is detected. These actions can be applied to the whole model or to a certain region of the data space. In the case of ensemble methods, a model management process will be needed after the adaptation process (Scholz and Klinkenberg, 2007).

Loss estimation

Error estimation is important for supervised methods in order to give feedback of the performance and success rate of the model. Thus, the system could report a severe drop in accuracy due to a change in data distribution. Also, this metric can be used to measure the reliability of the system when susceptible decisions are made. Performance evaluation metrics such as accuracy, recall, mean absolute scaled errors or root mean square error can be used to estimate the quality of a model at a specific moment in time. Detecting low quality levels in a model could warn the decision makers that the predictions made by the system are not so reliable, and also indicate that some actions need to be taken in order to improve the model.

1.1.2. Online learning for information systems

Once we have described the problem in all its complexity, it is important to describe the characteristics of the systems that could contain the proposed solution. One of the main characteristics of this thesis is the industrial nature of the work and the method proposed. The new methodology presented in this assay is meant to be part of a BI tool that helps decision makers through a DSS.

As already introduced, data streams require sophisticated, adaptive, and novel analysis techniques. According to Le et al. (2014), data stream mining (DSM) is concerned with the analysis of vast amounts of data as it is generated in real-time. Data stream classification allows for the labelling of previously unseen data instances, which can be a challenge to handle (Din et al., 2021). The data elements in the stream arrive online (Kolajo et al., 2019). A system has little control over the sequence of data elements that arrive for processing, this may occur either within a data stream or across data streams. Data streams are potentially unbounded in size (Kolajo et al., 2019). According to Wares et al. (2019), when an element from a data stream has been

processed, discarded or archived, it cannot be retrieved unless it is explicitly stored in memory, which typically is compared with the size of the data streams.

The kind of decision support provided over data streams is quite different from 'traditional' decision-making, these decisions are usually 'tactical' rather than 'strategic' (Chatziantoniou and Doukidis, 2009), for example, energy consumption monitoring (Fong et al., 2018). DSS for data streams may be designed and developed for general or specific purpose. According to a recent study by Lu et al. (2020) developing a new generation of adaptive DSS for real-time decision-making is imperative. They assert that self-learning and self-adaptive features are significant characteristics for the next generation of data-based decision-making (Lu et al., 2020).

In this sense, some adaptive or incremental learning methods have been applied already to many different decisional contexts such as prognostics and health management (Zhang et al., 2019), renewable energy systems (Severiano et al., 2021) or driving cycle prediction (Liu et al., 2021) among others. Particularly in the field of DSS, the interest in efficiently converting data streams into operational intelligence is increasing. In this sense, some authors such as Wang et al., (2018) or Yang et al. (2016) have proposed the application of streaming methods to DSS. Yet, as claimed by Mollá et al. (2022a) a lot of challenges and, therefore, opportunities are still not faced when we talk about data streams in DSS.

Implementing incremental or adaptive algorithms, as well as concept drift detection mechanisms in these cases, and integrating them in a DSS may reduce the impact changes have on the model's precision. Improving the engine system's accuracy will positively affect the reliability of the DSS and its recommendations, subsequently reducing any costs associated with incorrect decisions. Previous research in this field has proven that adaptive or incremental logic in changing contexts improves not only the response time of the model but also, in most cases, its accuracy (Domingos and Hulten, 2000). A DSS comprises a model and data, if that data changes, the expected answer/solution might change or might need further attention from the decision maker. When exposed to a potentially infinite data stream, a DSS might find these changes at any point and, if the DSS is unprepared, it might result in misleading recommendations. Being able to adapt to these changes automatically might lead to reduced errors and identify new opportunities to add value beyond the existing traditional approach.

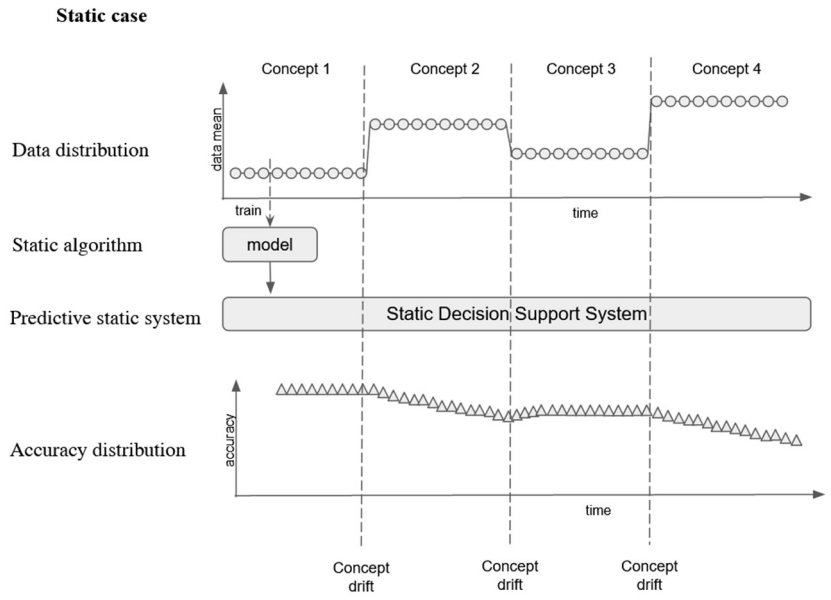


Figure 1.5 Traditional static DSS on data stream contexts (Source: Mollá et al., 2022a).

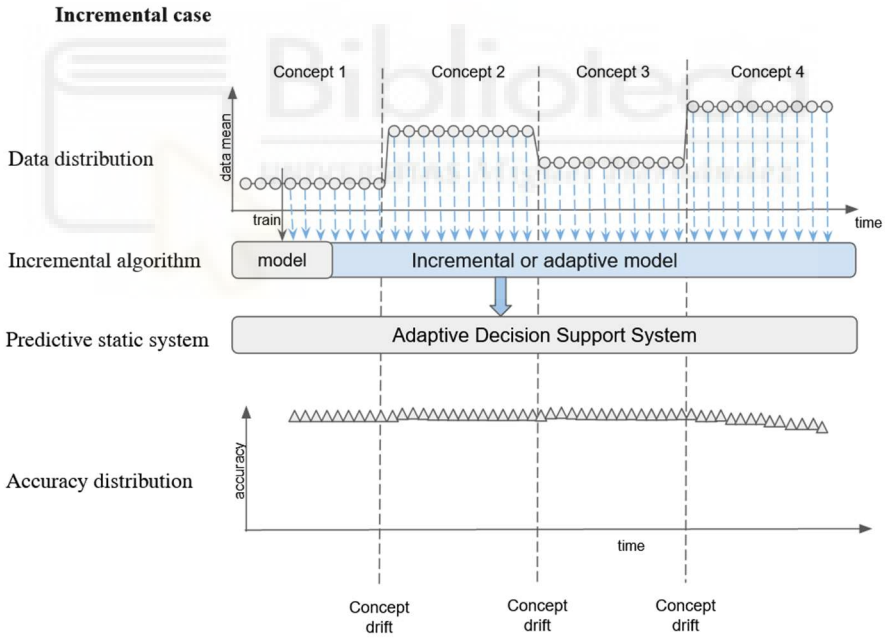


Figure 1.6. Adaptive DSS on data stream contexts (Source: Mollá et al., 2022a).

Figure 1.5 illustrates a traditional static structure and Figure 1.6 shows the proposed modifications (shaded) to be implemented in the adaptive DSS. These alterations expect to stabilise the degradation of performance caused by concept drifts in data. In both diagrams, (Figures 1.5 and 1.6), the data distribution varies abruptly creating four concepts with different means over time. In the static case (Figure 1.5), an initial model is built and then integrated in the DSS as a constant knowledge base. This might lead

to a slow degradation of the performance of the model when applied to other concepts. A model trained with outdated data might become unreliable and its application to an uncertain or changing context may lead to an increase in system errors. These static algorithms integrated in a DSS do not assimilate these new concepts in the base of knowledge and, therefore, they are exposed to this loss of accuracy and the increase of the error in the decision-making process. In the case of an adaptive DSS, the data distribution of new concepts is assimilated by the system through different techniques. These actions aim to update the model and prevent the uncontrolled increase in the prediction error. In Figure 1.6, new data is integrated into the model and, therefore, in the DSS, compensating, in most cases, for its degradation. Figures 1.5 and 1.6 are high-level schemes that represent the general idea of a traditional and an adaptive DSS to compare both logic. Nevertheless, the efficiency and sustainability of a model inside a DSS will depend not only on its logic but also on the algorithm that generates the base of knowledge, the error conditions or the concept drift impact introduced in the system. As illustrated by Mollá et al. (2022a) implementing adaptive techniques in changing data stream contexts improves the accuracy of the system and therefore the quality of decisions proposed by the DSS.

To summarise, the challenges that surround the data stream mining field could determine the problem in different ways. In this sense, memory constraints, the number of variables studied, the number of classes to predict, the type of learning approach or the error levels will be part of the problem definition. As well as these many factors, components such as the computational power, the speed of data update, the time requirements and the concept drift are particularly important when defining the premises. The static models might be especially affected when abrupt and significant changes are present in data stream. Thus, a problem arises when trying to use traditional data mining methods with changing data flows. The expected concept drift might lead to the drop in accuracy of the static models due to their lack of adaptability. Thus, the specific frame this work is focused on is the not-so-fast discrete problems that need to generate a base of knowledge that could adapt in case a change in data distribution takes place and integrate this knowledge into a decision tool.

1.2. Objectives

Once the problem and the context are introduced, we establish the premises and objectives of this work by hypothesising that:

> *Introducing an incremental logic to a classification algorithm in data streams contexts lessens the degradation of the system, maintaining higher accuracy levels and providing better classification results in a data-driven system.*

Following this hypothesis, we establish the main objective pursued by this dissertation is the proposal of a new algorithm that integrates an incremental logic in a classification rules algorithm. As hypothesised before, we expect this incremental approach to improve the accuracy levels for data streams used in data-driven systems. Among other objectives pursued by this research, there is the automatization of the concept drift detection and readjustment of the model. This specific objective is aimed with the second and third versions of the proposed methodology as we will see in Chapter 3 and Chapter 4. Another goal of this investigation is to optimise the rule structure used by the classification method for decisional context. Using a rule structure more informative that provides the decision maker with a wider description of the context might be preferable in BI systems, especially considering the future integration of these rules in a DSS. Also, in order to simplify the rule sets generated by the proposed algorithm and following the same logic than its antecedent, CREA proposed by Rodriguez-Sala (2014), an adaptation of the post-pruning method presented by Almiñana et al. (2012), RBS, is sought. After these methods are proposed and implemented, a deep and wide study is aimed in order to test the new algorithm with different approaches or versions in many different conditions or scenarios. The purpose of this work is to study this new methodology in a variety of simulated conditions as well as in several real scenarios with data streams and compare the results with the traditional static approaches and the state-of-the-art techniques. This study, the results obtained and the discussion that arise from them is presented in Chapter 4.

1.3. Materials and methods

This dissertation aims to propose a new method to approach the classification problems in data stream contexts from the decisional perspective of BI. To do this, the proposed solution uses the decision rules to model the knowledge and to be able to integrate it in a DSS. The design of the methodology, as well as the experiments and the discussions carried out along this work are made considering the industrial approach of the proposal and the necessity to include this solution in a BI engine. The ability to integrate an online solution in a data-driven solution opens a wide field of applications and new opportunities as explained by Mollá et al. (2022a). Many disruptive events recently seen such as global resources crisis, political instabilities or the emergence of new pandemics confirm the value of adaptive systems that deal with uncertain and changing context in stream.

The novel approach presented in this dissertation, called Incremental Decision Rules Algorithm (IDRA), is a new solution that analyses data streams and, therefore, is continuously learning from new data. With this new technique, a variety of challenges arise, as already explained in the introduction to the problem. In this work, we define the conditions of this novel proposal and the different approaches studied for the data stream challenges exposed. A total of three versions of the new methodology are presented in this dissertation, IDRAv1, IDRAv2 and Reactive IDRA or RIDRA. These

variants have the same inner logic but different ways of dealing with the adaptive learning process. For instance, the concept drift detection and the forgetting method are implicit in the first version of the proposal and explicit in the other two, while the incremental learning adaptation is common to all the versions. On the contrary, the rule set management and the structures used in the three approaches are slightly distinct from each other. A deeper explanation of these versions and its differences is presented in Chapter 3, while the computational experiments carried out to compare them and the associated discussion are presented in Chapter 4.

The experiments performed in this dissertation are executed in a computer provided with a Processor Intel i7 with 1.10GHz and 16GB RAM. The Python version used is Python 3.9.12 and, apart from this, the MOA framework (Bifet et al. 2010), implemented in Java, is used to launch some of the experiments. No further materials are necessary in this research.

1.4. Structure and results of the work

This dissertation is structured as follows: Chapter 2 introduces the previous works available in the data stream mining field in general and, specifically, in the classification field. Chapter 3 contains a wide and deep explanation of the main characteristics and the functional description of the novel algorithm in all its versions. In Chapter 4, the computational experience is explained for all the simulated cases, as well as the real ones. This chapter also comprises the results obtained from comparing IDRA, in all its versions with the state-of-the-art method and the static antecedent, as well as the discussion that arises from this. In this chapter, IDRA versions show to perform with better accuracy than the static and the state-of-the-art solutions for the most common simulated scenario, the high impact one and the two real cases, being RIDRA better than all the solutions in all cases except from one of the real cases. Finally, Chapter 5 briefly summarises the conclusions of this research and the opportunities for future research in data stream contexts from different perspectives.

Chapter 2
Literature review

This chapter describes the previous works related with the current research to try to define the state of the art and the contributions that could be made within this field. In this sense, a wide variety of algorithms with different approaches for data streams problems are presented. In this chapter we will pay special attention to those methods that use decision rules, since that structure is used by the technique proposed in this dissertation. Thus, those approaches can be used as a reference point to compare them with the proposal of this work.

Some of the content of this chapter is based on the paper published in the journal Mathematics under the title *Incremental Decision Rules Algorithm: A Probabilistic and Dynamic Approach to Decisional Data Stream Problems* (Mollá et al., 2022b) and in the Journal of Decision Systems under the title *Data-driven decision making: new opportunities for DSS in data stream contexts* (Mollá et al., 2022a).

2.1. Data stream algorithms

This chapter aims to describe, in a meaningful way, the previous work made in the context of the classification problems for data streams, the incremental decision rules algorithms and the adaptive DSS. The characteristics of the antecedents, along with the potential of the mentioned fields will describe the possibilities of this proposal. In this case, the data stream analysis has many related tasks that need to be solved. Among these tasks we can find the data profiling (Schirmer et al., 2019; Caruccio et al., 2021), the queries for data streams (Zaniolo, 2012; Ghanem et al. 2007) or the online analysis of data in which this work is framed. As described by Gaber et al. (2005), the data stream solutions could be categorised in data-based and task-based ones. On the one hand, data-based solutions examine only a subset of the whole dataset, this being either a vertical or horizontal transformation of it. On the other hand, task-based solutions modify existing techniques or invent new ones to address data stream problems.

Within the data stream analysis, a wide variety of techniques have been implemented to deal with its challenges. Some traditional methods such as decision trees (Domingos and Hulten, 2000; Bifet et al. 2017), decision rules (Gama and Kosina, 2011; Ferrer-Troyano et al., 2006), clustering methods (Aggarwal, 2003; Puschmann et al. 2017) or association rules techniques (Jiang and Gruenwald, 2006) have their adaptation prepared to analyse data streams. These methods can be considered either part of the supervised or the unsupervised learning techniques. The unsupervised algorithms learn patterns from data that have no label, while the supervised algorithms use tagged data to learn and then test their efficiency. Two of the biggest techniques of these two learning methods are the clustering (unsupervised) and the classification (supervised). Figure 2.1 compares these two approaches in terms of publication trends to show the research interest they have within the data streams field.

As we can observe in Figure 2.1, the trend of the data stream classification has continued growing along the years, having a promising rate of almost 750 publications per year in 2021, while the data stream clustering stabilised its publications around 600 per year from 2018 with a slight drop in 2021. These results correspond to the search of *data stream classification* and *data stream clustering* terms in the Web of Science (Clarivate, 2022).

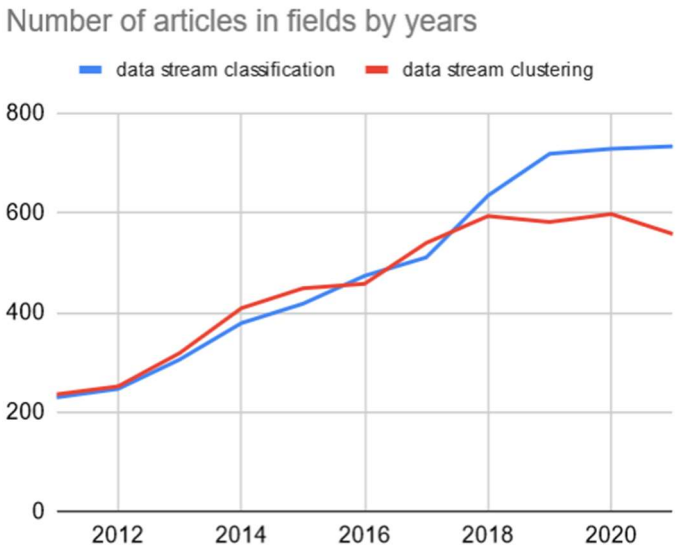


Figure 2.1 Number of articles for classification and clustering methods using data streams in the Web of Science (own source)

2.1.1. Clustering

Clustering or data segmentation is the process of arranging objects into sets called clusters depending on their similarity. Thus, the data objects contained in the same cluster would be similar among them and dissimilar to data objects in other clusters. As defined by Nguyen et al. (2015), the adaptation of clustering algorithms typically extends traditional methods to allow them working with data streams. Thus, the new versions can satisfy constraints, such as bounded memory, single-pass, real-time processing, or concept drifts. Similar to traditional clustering methods, data stream clustering can be divided into five main classes: (1) Hierarchical algorithms, (2) Partitioning based algorithms, (3) Grid based algorithms, (4) Density based algorithms and (5) Model based algorithms.

The distance between clusters that will define if two clusters can be merged during the stream process could be of four types: (1) minimum distance, (2) maximum distance, (3) mean distance and (4) average distance. Minimum distance and mean distance are the most popular ones due to their lower computational cost. Examples such as STREAM (O'Callaghan et al., 2002) or ClusStream (Aggarwal et al., 2003) use mean distance while MR-Stream (Wan et al., 2009) use minimum distance as a criterion for cluster merging. Some more recent examples collected by Zubaroğlu and Atalay (2021) in this field are the Adaptive streaming k-means, a partitioning based online algorithm proposed by Puschmann et al. (2017) or Improved Data Stream Clustering Algorithm, a density-based algorithm proposed by Yin et al. (2017).

2.1.2. Classification

In this work, due to the nature of the proposed method, we will delve into the classification methods that belong to the supervised learning techniques. In data stream systems, when a new instance is received, the true label associated with a studied variable may be expensive to calculate in time and resources. Therefore, an approximation of this value is estimated using machine learning techniques. Particularly, classification methods estimate a label for this studied variable, best known as an objective variable, underlining possible patterns that may affect it.

According to the division referred by Srivani et al. (2020), the techniques of big data classification are categorised into eight groups depending on the base methodology: (1) Convolutional Neural Network (CNN) based techniques, (2) Neural Network (NN) based techniques, (3) Deep learning (DL) based techniques, (4) K-Nearest Neighbour (KNN) based techniques, (5) Fuzzy based techniques, (6) Support Vector Machine (SVM) based techniques, (7) Optimization based techniques, and (8) Decision Tree based techniques. In this sense, the techniques based on Decision Trees are considered the most relevant ones to the scope of this dissertation, but previous works from other groups are also presented in next paragraphs.

Ramírez-Gallego et al. (2017) proposed a KNN classifier using Apache Spark Structured Streaming to process the very fast flow of data. This classifier, called DS-RNGE, distributes the examples on a tree structure, with a top-level tree for query routing and a set of distributed subtrees to search in parallel. An instance selection is introduced for updating and eliminating outdated instances from the case-base and, therefore, endows the method with an adaptive logic. Another example of KNN classifier is the one included in the work of Hassanat (2018). This method uses a KNN classifier to insert new instances to train a binary search tree and speed up the searching process for text samples. Two approaches are used in this work, both of them using Norm-based Binary Tree (NBT) and Minimum/Maximum Norms-based Binary Tree (MNBT). Another previous work that is framed as a classification technique is IBLStreams, proposed by Shaker and Hüllermeier (2012). This methodology develops an instance-based learning classifier for data streams that adds or removes instances from the model based on temporal and spatial relevance and consistency. This method monitors the error rate and the standard deviation during training process.

Decision tree methods

We focus now on the decision tree-based techniques. These techniques are considered especially relevant for this dissertation due to their similarity with the presented work. In this sense, a few proposals are presented in this part to illustrate the previous research made in this field. One of the most important techniques in classification tasks are decision trees. Among the contributions in streaming decision trees, the Hoeffding tree, also known as Very Fast Decision Trees (VFDT) and proposed by Domingos and

Hulten (2000), is considered one of the most relevant works. This method builds an adaptive tree using the Hoeffding bound to decide the exact number of examples that are necessary at each node to guarantee constant time and memory per example. The Hoeffding bound is defined by:

$$\epsilon = \sqrt{\frac{R^2 \cdot \ln\left(\frac{1}{\delta}\right)}{2n}} \quad (1)$$

where R is the range of the random variable, δ is the desired probability of the estimate not being within of its expected value, and n is the number of instances included in the node. The output model given by this algorithm is asymptotically nearly identical to that given by a batch conventional learner. From this initial version, over the years, a family of algorithms have been developed by different researchers, following the same premises but trying to improve the results. One of these evolutionary versions is CVFDT, extended by Hulten et al. (2001), is an algorithm for mining decision trees from data streams with concept drifts. This version builds an alternative subtree and makes it grow until the old one becomes inaccurate, and it can be replaced by the new one. A toolkit for mining high-speed data streams called VFML (Very Fast Machine Learning) is presented, also by Hulten and Domingos (2003), based on these two successful algorithms. Other similar work based on this algorithm is the proposal made by Gama et al. (2003) called VFDTc. This method extends VFDT in two aspects: (1) with the possibility of dealing with numerical attributes and (2) with the use of more powerful classification methods at the tree leaves such as naïve bayes classifiers (VFDTcNB). Following this VFDT logic we can find an efficient version proposed by Li and Liu (2008). This version, named EVFDT from Efficient VFDT, extends the initial system by presenting a pruning method called Uneven Interval Numerical Pruning (UINP) to process numerical attributes in a more efficient way. This method also applies naïve bayes classifiers in order to reduce the scale of the trees. Another improved version of Hoeffding Tree is the Hoeffding Anytime Tree "Extremely Fast Decision Tree" (EFDT) presented by Manapragada et al. (2018). This version obtains a significant improvement in accuracy with a minor modification of the implementation of Hoeffding Tree available in MOA. Recently, an extension of this work has been published by Garcia-Martin et al. (2022) presenting a version with lower energy consumption and therefore lower memory footprint, maintaining the same levels of accuracy. This proposal, Green Accelerated Hoeffding Tree (GAHT), achieves the same competitive performances as EFDT and ensemble versions of Hoeffding tree while reducing the consumption of energy up to 70%. Another work that could be considered a part of this family is the recent work proposed by Ducange et al. (2021). This method introduces a new fuzzy technique based on Hoeffding Decision Tree called Fuzzy Hoeffding Decision Tree (FHDT). This extension endows these trees with a fuzzy logic and is able to build trees more robust to noisy and vague data. This algorithm improves the accuracy of the systems and reduces their

complexity in exchange for a higher processing time consumption. All this family of algorithms is based on the premises of the initial VFDT and/or the Hoeffding limit. Despite reasonable results, Hoeffding bound has been proven to be formally incorrect in solving such problems. Rutkowski et al. (2013) discuss this idea and propose a generalisation of Hoeffding inequality called McDiarmid inequality as split criteria.

Another decision tree classifier for data streams is streamDM-C++, an extremely fast decision tree in C++ presented by Bifet et al. (2017). This method proved to improve the solutions included in the VFML toolkit (Hulten and Domingos, 2003) and it is available as an open-source software (Bifet et al., 2015). In the work that introduces streamDM-C++, the authors remark three dimensions of interest to study: accuracy, memory and time, and defend the classification methods as one of the most used techniques in data mining. This dissertation follows these premises and the ones led by Kosina and Gama (2013) to evaluate the novel proposal. In this case, two of the three mentioned dimensions (accuracy and time) are explored for all the experimental data streams in this dissertation.

Decision rules methods

Even if the decision trees solutions for data streams have been widely used, their structure may be complex depending on their depth and the number of changes that they need to assimilate. A deep tree might make it especially difficult for experts to read information. This knowledge, derived from a tree structure, can also be represented in easier ways such as rules (Quinlan, 1987). A decision rule is a logic predicate structured as a condition "IF antecedent THEN label" in the way that an antecedent implies a consequent. The antecedent is a conjunction of conditions for the attributes included in the dataset and the values in their domains. Thus, the antecedent describes the attributes that an instance needs to fit to be classified with that rule, while the consequent gives a prediction label. These decision or classification rules may contain the same information as the tree branches, however in a modular and more interpretable way. Unlike tree branches, each decision rule is independent and can be understood in isolation from the rest of the set. In these structures, each rule corresponds to the path from the root to a leaf, and there are as many rules as leaves, with these rules being as complex as the decision tree itself. This new approach might be more interesting for decisional contexts due to its readability and natural integration in a decision system engine (Bertini, 2020). Each rule has support and confidence metrics, which express the relative use of a rule in a set (support) and the probability of a successful prediction for that rule (confidence). These metrics will represent the quality and predictive capabilities of every rule of a set.

Among the decision rules techniques, the first rule learner designed for processing data streams is the FACIL algorithm proposed by Ferrer et al. (2005). This method is based on filtering the examples lying near to the decision boundaries (border examples). Consistent rules (those including only one type of example) classify new test examples by covering them, while inconsistent rules (those mixing positive and negative

examples, border examples) classify them by distance as the nearest neighbour algorithm. The best-known decision rules algorithm for data streams, and the one used to compare the proposal of this work, is the Very Fast Decision Rules (VFDR) proposed by Gama and Kosina (2011). VFDR is a single pass algorithm that learns ordered and/or unordered rules with either numerical or categorical variables. The rules expand considering those literals that minimise the class labels' entropy of the examples covered by each rule. This algorithm uses the Hoeffding bound (see (1)) to determine the number of observations after which a rule can be expanded, or a new rule can be induced. This threshold is checked only after a given number of examples (*Nmin*) to improve efficiency. Thus, the rule set may adapt every *Nmin* instance and would increase its predictive capacity over time. The limited size of the rule sets and the low time requirements of VFDR make it a high-quality data stream method.

FACIL and VFDR are the two best-known methods to deal with classification problems in data streams using decision rules. Nevertheless, some other proposals have been made in this field. eClass is a fuzzy-rule-based classifier proposed by Angelov and Zhou (2008) to solve classification problems following a fuzzy logic. This proposal is based on eTS-type fuzzy systems and includes two versions of the algorithm, eClass0 and eClass1. In this case, the logic of these two versions is different since eClass1 is able to manage multiple outputs. Another relevant contribution is G-eRules by Le et al. (2014). This work proposes a Gaussian distribution as a new heuristic method to improve the previous version (eRules) and make it computationally more efficient. The previous eRules version uses a sliding window approach and Prism classifier (Cendrowska, 1987) to work on data streams.

The algorithm proposed in this work is compared with VFDR since it is the main reference within the field of decision rules for data streams and it is publicly available. Notably, it is important to highlight that these two algorithms differ slightly in nature. For VFDR the time may prevail over accuracy, while our new approach will try to provide more accurate solutions in an affordable time for the decision problem at hand. Therefore, the algorithm selection would depend on the problem we have to face and its associated restrictions.

Table 2.1: Summary of the clustering techniques exposed in this chapter.

Clustering technique	Brief description	Reference
STREAM	It is a memory efficient, single-pass algorithm for k-median clustering	O'Callaghan et al., 2002
ClusStream	It generates the stream clustering process using pyramidal time windows and micro-clustering	Aggarwal et al., 2003
MR-Stream	It is a density-based clustering algorithm that provides a hierarchical and multiresolution view of the clusters	Wan et al., 2009
Adaptive streaming k-means	It defines the number of clusters based on the data distribution and creates clusters from the incoming data streams as the traditional k-means algorithm	Puschmann et al., 2017
Improved Data Stream Clustering Algorithm	It proposes a method to combine large high-speed data streams clustering with an intrusion detection mechanism	Yin et al., 2017

Table 2.2: Summary of the classification techniques exposed in this dissertation.

Classification technique	Classifier	Brief description	Reference
DS-RNGE	KNN	A lazy learning solution for massive data streams using distributed metric tree in Spark	Ramírez-Gallego et al., 2017
NBT/MNBT	BT/KNN	It inserts training examples into a binary search tree based on the Euclidian norm and a KNN classifier	Hassanat, 2018
IBLStreams	Instance-based learning (IBL)	It is an instance-based algorithm that works for classification and regression problems and it is implemented in the MOA framework	Shaker and Hüllermeier, 2012
VFDT	DT	Very fast decision tree based on Hoeffding bound	Domingos and Hulten, 2000
CVFDT	DT	Very fast decision tree with concept drift detection (extension to VFDT)	Hulten et al., 2001
VFML	Tool	Very fast machine learning. A toolkit for mining high-speed data streams	Hulten and Domingos, 2003
VFDTc and VFDTcNB	DT	It extends VFDT working with numerical attributes and improves the classification methods using Naïve Bayes	Gama et al., 2003
EVFDT	DT	Efficient very fast decision tree. Presents a pruning method (UINP) to improve the use of numerical attributes	Li and Liu, 2008
EFDT	DT	Extremely Fast Decision Tree or Hoeffding Anytime Tree. Minor modification of the implementation of Hoeffding Tree available in MOA	Manapragada et al., 2018
GAHT	DT	Green Accelerated Hoeffding Tree. Extension of EFDT but with lower energy consumption	Garcia-Martin et al., 2022
FHDT	DT	Fuzzy Hoeffding Decision Tree. New fuzzy technique based on Hoeffding Decision Tree	Ducange et al. (2021
streamDM-C++	DT	An extremely fast decision tree in C++ and available in Apache SAMOA	Bifet et al., 2017
FACIL	DR	First rule learner designed for processing data streams	Ferrer et al., 2005
VFDR	DR	Very Fast Decision Rules. A single pass algorithm that learns ordered and/or unordered rules with either numerical or categorical variables	Gama and Kosina, 2011
eClass	DR	Evolving classifier. A fuzzy-rule-based (FRB) classifier based on eTS-type fuzzy systems	Angelov and Zhou, 2008
G-eRules	DR	Rule-based method that uses a sliding window and Prism algorithm with a gaussian distribution to classify data streams	Le et al., 2014

Data Stream Tools

There are several frameworks or toolkits designed to simulate and analyse data streams. The best-known ones are Massive Online Analysis (MOA) (Bifet et al., 2010), Apache SAMOA (De Francisci and Bifet, 2015), Spark Structured Streaming (Apache Spark, n.d.) or Very Fast Machine Learning (VFML) (Hulten and Domingos, 2003). Apache SAMOA is an open-source platform for mining big data streams while Spark Structured Streaming is an extension of the Spark API that allows natively both batch and streaming real-time data flow processing from various sources. These two frameworks use distributed environments that, now, do not fit the requirements of our problem. Nevertheless, a distributed implementation could be considered in the future to optimise the integration of the system in a production environment. In this sense, due to the extended integration of Spark systems in productive projects, Spark Structured Streaming could be one of the preferred options. Another data stream tool, VFML is one of the first toolkits for mining high-speed data streams and it is mostly written in C. Finally, MOA is a software environment for implementing algorithms and running experiments for online learning from evolving data streams. This software is one of the most important environments to analyse data streams and has a vast number of implementations and available documentation. This framework is mostly written in Java, and it is the environment used to run the VFDR algorithm, implemented under the name of DecisionRules.

2.2. Adaptive Decision Support Systems

Business and technology researchers have defined and examined ways to improve decision support, decision systems, and subsequently, managerial, and organisational decision-making for over 100 years, with the potential contribution of computing technology coming into particular focus in the last 60 years. DSS emerged due to the availability of technologies for collecting, storing, manipulating, and displaying data and information to support managerial decision-making (Morton, 1971). Existing research asserts that the support provided information systems for semi-structured and unstructured decisions should be characterised as DSS. Gorry and Morton (1971) defined DSS as “interactive computer-based systems, which help decision makers utilize data and models to solve unstructured problems”. Further, according to Sprague and Watson (1979), at its core a DSS involves a database, a model, a user interface, and a decision maker. A computer-based decision system consists of three interacting components (1) a Language System, (2) a Knowledge System, (3) a Problem Processing System (Ginzberg and Stohr, 1982). More recently, Eom (2022, p. 1) defined DSS as “an interactive human–computer decision-making system that supports decision makers rather than replaces them, utilising data and models”. In some cases, data does not take the form of persistent relations, but rather arrives in multiple, continuous, rapid, time-varying data streams (Babcock et al., 2002). Advances in networking, data collection, storage, and dissemination have enabled this

new type of data-intensive solution, which rely on data streams for decision-making (Babcock et al., 2002; Le et al., 2014). Traditionally, DSS data were stored statically and persistently in a database. Increasing volume and intensity of information and data streams create new opportunities and challenges for DSS experts, data scientists, and decision makers. Examples of such applications include financial applications (e.g. streams of transactions), wireless sensor networks (e.g. measurements), traffic management (e.g. streams of packets), mobile records (e.g. streams of calls or mobile data), web management (e.g. click streams), data privacy or social networks (Babcock et al., 2002; Chen et al., 2022; Laforet et al., 2021; Zhang et al., 2011; Ducange et al., 2019)). In recent years, several related concepts have been emerging and developing. In this sense, terms such as Active Stream Data Warehouse (ASDW) (Bimonte et al., 2019), integrated DSS (IDSS) (Baldin et al., 2021) or mobile DSS (Guo and Díaz López, 2013; Kozina et al. 2018) are being defined or evolving to adapt new systems in data streams contexts. Novel data stream contexts require that we move beyond static DSS modelling techniques to support data-driven decision-making. Implementing incremental and/or adaptive algorithms may help to solve some of the challenges arising from data streams.

Bertini (2020) poses the necessity of using explainable and transparent decisional methods when automatic predictions are made so a decision-maker can understand and therefore trust the actions recommended by the system. In this sense, the author claims that white-box models, such as rule-based models, are preferable to black-box models. Nevertheless, not much attention has been paid to rule-based algorithms in data stream problems due to their accuracy level. These models do not provide acceptable rates of accuracy in many applications (Bertini, 2020) and therefore are replaced by other techniques that might be less explainable. Thus, we can identify the potential value of a highly precise and explainable solution that contributes to transparent and trustworthy rules in a data stream decision support system. In this framework is where we can find the proposal of this work explained in detail in next chapters.

In their work, Trépos et al. (2013) defend the value of decision rules in data mining tasks, focusing on the concept of actionability. This term defines the applicability of a set of rules to a problem given characteristics as their size or complexity. The authors claim that generating the set of rules is an important part of the process but also it is important that the generated knowledge can be applied straightforwardly to a system, like a DSS. Also in this work, a methodology that enhances the actionability of a decision rule set is proposed under the name of DAKAR (Discovery of Actionable Knowledge And Recommendations).

2.3. Findings and research direction

In previous sections, we presented the different techniques available in the state-of-the-art to solve classification problems in streaming. Then, we exposed the value of certain explainable techniques, like decision rules, to feed decisional systems, and after that we presented the possibilities of integrating this adaptive power to DSS. In this section, using the knowledge given in previous ones, we try to delimit the gap found in the literature to frame our work in a meaningful and innovative direction.

In this sense, as far as we know, no previous proposal has been defined based on its potential integration in a DSS. In contrast, the chosen methodology leads to an original technique specifically designed to fit an information system (IS) that could be used to make decision in different fields. The final goal of this work and of the project drafted by Teralco Solutions Co. is to provide a BI solution that, integrating the selected methodology, could respond and adapt to changes that take place in different business contexts. In this sense, the use of elements such as Key Performance Indicators (KPI) or Objective Key Results (OKR) allow us to define and standardise the strategic goals of a company and study them. In this framework, the design of IDRA is thought to be able to predict and monitor these indicators and, therefore, to allow the system to early warn a critical situation or a deflection in the business planning. Thus, providing an incremental logic to a BI solution could report multiple benefits in the strategic plans of companies. Among them, the control over the risk of certain actions or the timely disinvestment in specific decadent product or project could lead to avoid significant losses for the company.

Usually, data stream solutions prioritise the time over the model accuracy. This is understood in the context of fast data arrival seen in sensors, network traffic or streaming video processing among others. In the case of the business solutions, the data arrival is not so fast as in the mentioned scenarios, but the benefits from constant and accurate analysis might be extremely valuable. In this sense, IDRA proposes an innovative methodology based in a different paradigm that prioritises the accuracy of the model over the time, always assuring an affordable time response. In this sense, the state-of-art method, VFDR, is a very fast solution based on a very small set of rules (so the searches are very fast) that changes quickly. On the contrary, IDRA changes this paradigm, incrementing the available rules, so the time of search also increases, but the durability, accuracy and stability of the model is expected to improve with them.

Chapter 3
Incremental Decision Rules Algorithm: IDRA

The Incremental Decision Rules Algorithm (IDRA) is a new incremental proposal based on traditional decision rule methods. This new technique is focused on solving supervised classification problems in data stream contexts providing relevant information for the decision-making process. This algorithm is a novel approach designed to fit DSS whose motivation is to give accurate responses in an affordable time for a decision situation. The content of this chapter 3 is based on the paper published in the journal Mathematics under the title *Incremental Decision Rules Algorithm: A Probabilistic and Dynamic Approach to Decisional Data Stream Problems* (Mollá et al. 2022b). This publication presents the initial version of the proposed method. The extensions and new versions of the algorithm are extended in this chapter.

3.1. Methodology design and justification

After reviewing the literature available for data stream analysis, specifically the works with explainable methods such as decision rules, and the potential of adaptative decisional systems, we design our proposal based on the identified opportunities. As seen in section 2.3, the gap detected between the data stream analysis and the DSS fields is promising. In this sense, we propose a methodology with a new paradigm of incremental learning that prioritise the collection of knowledge to maintain the accuracy and stability of the models that feed the DSS, enriching with this the decision-making process.

When we study the main methodologies available in the literature for data stream analysis using explainable methods such us the decision rules, we highlight the method proposed by Gama and Kosina (2011), VFDR. This methodology is a reference of the data stream techniques and it becomes a paradigm in data stream analysis with its approach. VFDR tries to keep in memory the lowest number of rules possible, changing, erasing and/or replacing them continuously along time. Thus, the time of the rule search is extremely low and it allows the algorithm to response timely in very fast scenarios. We understand that this method, as many others that derive from its idea, sacrifices part of the accuracy of the models to get very fast solutions that could be applied to stream context. A wide range of these techniques, specially VFDR, have proved to have good and very fast results in time-demanding data stream contexts. Nevertheless, when we talk about decisional contexts, the demand of the systems prioritises the accuracy of the predictions over the time, without renouncing to the constant analysis that incremental or adaptive systems provide. In this sense, motivated by the application of continuous analytic techniques to decisional context, we propose a new algorithm that includes these new ideas by design. IDRA, is designed to model the system using explainable and concrete rules. Nevertheless, this approach ends up generating significantly more rules than in very fast algorithms and it could delay the searches in the system in an unsustainable way. To avoid that, IDRA divides the knowledge engine, in this case the rule set, in two collections based on the upper support axis (Es_s) (Almiñana et al., 2012): the potential rule set and the decisional rule set (see Figure 3.6). Thus, the model rules are divided depending on the state and the use that they might have, allowing IDRA to search among only relevant and significant rules (decision rule set) and add new incipient knowledge to another set (potential rule set) that does not need to be considered in the predictive process. Therefore, IDRA is designed to deal only with a subset of the rules in the decision process, reducing with this the search times of the algorithm. This design allows IDRA to have a bounded and controlled knowledge base that, even if the number of rules increases significantly, will not increment the algorithm response times. Then, IDRA can assure timely predictions independently from the number of instances previously studied and the rules that the system decide to build.

Besides this, other important design decisions might be taken regarding the algorithm construction and its outcomes. In this sense, we need to define structures and tactics to face data stream problems based on DSS perspectives. First of all, we need to design the interface of the method that will communicate with the DSS. Thus, we can identify how the methodology needs to response and which techniques are preferred to solve each challenge of the streaming scheme. In this sense, to enrich the decision-making process, IDRA offers a probability distribution of the classes for the studied variable. This information might be significant to understand the risk of the actions, as well as the linked consequences, e.g.:

1) *A1 = {value}, A2 = {value}* → *Class 1 (52%)*
2) *A1 = {value}, A2 = {value}* → *Class 1 (52%), Class 2 (47%), Class 3 (1%)*

In this example, the rule type 1 is a hard classifying rule with an associated probability or confidence (52%) and a cost. On the other hand, the rule type 2 is a soft classifying rule that represents the associated probabilities for each one of the classes of the studied variable. When applied to hard classification, following a majority class criterion, both structures classify the same way, *Class 1* with 52% of confidence. Nevertheless, in decisional and managemental contexts, not only the probability of success is important, but also the costs of the decisions. We estimate the associated costs of the actions for *Class 1, Class 2* and *Class 3* such that,

Costs of actions:
Class 1 = 100u, Class 2 = 20u, Class 3 = 500u, being *u* a unit of effort.

With this new information, the rule type 2 has a potential benefit over the rule type 1. In this new situation, the managers could detect that, even if the *Class 1* is the most probable one, the *Class 2*, apart from being close to *Class 1* with 5% of probability difference, also entails the lowest cost, and therefore, the minimum risk. In this case, IDRA implements its rules based on the rule type 2. Thus, the methodology could provide a highly valuable information to the managers and decisions makers through the DSS.

Besides the interface that communicates IDRA and the DSS, and its associated potential, other design decisions need to be made to deeply define the functionality of the method. As seen in the Chapter 1, adaptive learning systems are designed considering several modules (see Figure 1.4). Now, based on the work of Gama et al., (2014), we will define and justify the IDRA approaches to each of these modules in the context of decisional or managemental problems. In some of these modules, IDRA is designed in different ways (versions) to study how each strategy affects the methodology, trying to prioritize those designs that optimise the model performance.

Memory

Respecting to the memory, Gama et al. propose a division of the taxonomy of memory properties in data management methods and forgetting mechanisms.

Data management

Regarding the data management, IDRA is designed to be as independent as possible from the data storage. Our methodology is thought to store the seen data in the form of rules, not needing to see again the data, except in the case of a concept drift. In this sense, different versions of IDRA are designed to work with single examples and with multiple examples to study the advantages and challenges that each perspective is arising. IDRAv1 (see section 3.4.1.) is the simplest approach with a single example storage, that only builds the set of rules and expects it to change along with data. Thus, no previous data is needed to rebuild the model, since no model rebuilding is performed. On the other hand, IDRAv2 (see section 3.4.2.) and RIDRA (see section 3.4.3.) store multiple examples in memory, to rebuild the model when a change is detected. In these cases, the minimum examples that need to be stored is the system would be the fixed size of the training set.

Forgetting mechanism

The most common way to deal with data stream that change in an unpredictable manner, it is to forget the outdated examples. In this sense, different approaches are studied through several versions for IDRA method. IDRAv2 (see section 3.4.2.) uses a sliding window of the static size of the training set, assuring that if a drift is detected, there is a new training set available to rebuild the model. RIDRA (see section 3.4.3.) also uses a sliding window but, in this case, an adaptive window (Bifet and Galdavà, 2007). On the contrary, IDRAv1 (see section 3.4.1.) does not contain any explicit forgetting method. Thus, the changes in the data distribution are integrated in the system only through the own incremental logic of the algorithm, endowing IDRAv1 with a low reactive behaviour. By studying this feature, or the lack of it, we can highlight the importance of these mechanism and the inner adaptability potential of our method, and the necessity of a quicker reactive solutions depending on the conditions of the scenario.

Change detection

Another module of the adaptive learning systems that needs to be considered in the design of the methodology is the change detection techniques. In this sense, several decisions have been made to reflect the variety of approaches used to detect the concept drifts in a stream system. As already mentioned in previous sections, RIDRA (see section 3.4.3.) uses an adaptive window (Bifet and Galdavà, 2007) to detect the changes, in this case by monitoring the data distributions of two different time windows. IDRAv2 (see section 3.4.2.) uses a detection method based on monitoring the statical metric of accuracy. Thus, if the efficiency of the model is lowering along

time, its degradation is understood as a change in the data distribution. Again, IDRAv1 (see section 3.4.1.) does not contain a detection method to study the necessity and value that these techniques contribute to incremental systems.

Learning

Regarding the learning process, different techniques and mechanisms could be used to generalise and extract information from the stream examples that arrive to the system. In this module, design decisions regarding the learning mode or the adaptation method are presented. In this case, we consider that the learning process is maintained always through a single model that evolves along time.

Learning mode

Whenever the system receives a new labelled example, the base of knowledge needs to include it. The selected technique used to update the model is the incremental mode even if some versions of IDRA (IDRAv2 and RIDRA) use a retraining mode when changes are detected to abruptly forget previous concepts. With an incremental learning mode, IDRA receives instances one-by-one and updates the decision rules that shape the base of knowledge and its statistics.

Adaptation method

Respecting to the adaptation method used in IDRA, again several available techniques are selected to test and estimate the impact of each one in the system. In the case of IDRAv1 (see section 3.4.1.), the blind adaptation strategy is used while in IDRAv2 (see section 3.4.2.) and RIDRA (see section 3.4.3.) use an informed strategy. In this case, IDRAv1 has no change detection method, therefore a blind adaptation happens with the data. Blind approaches are limited due to its slow reaction to concept drift in data, nevertheless, reference methods like VFDT (Domingos and Hulten, 2000) use this strategy. On the contrary, IDRAv2 (see section 3.4.2.) and RIDRA (see section 3.4.3.) use an informed strategy based on the change detection methods explained in previous sections. The degree of impact that potential replacements have in IDRA systems are limited to local replacements. Decision rules are considered granular models since the restructure of the information affects only particular elements of the base of knowledge. This restricted impact justifies the selection of decision rules as the structure of information, as well as their natural integrability in a DSS.

In brief, there are many different designs known and studied in the literature. Despite the large number of options, the design decisions are made under the criteria of optimising data stream decision problems and its implementation in adaptive DSS, as well as studying the impact of different techniques that could be applied to our methodology. In this sense, the design decisions are represented based on the taxonomy proposed by Gama et al. (2014) and shown in Table 3.1 for each version of IDRA, explained on detail in section 3.4.

Table 3.1. Design decision made under the taxonomy division of Gama et al. (2014)

Module	Issue	Design decision		
		IDRAv1	IDRAv2	RIDRA
Memory	Data management	Simple example	Multiple examples	Multiple examples
Memory	Forgetting mechanism	No explicit method	Fixed sliding window	Adaptive sliding window
Change detection	Detection models	No explicit method	Monitoring accuracy	Monitoring data distribution
Learning	Learning model	Incremental	Incremental and retrain	Incremental and retrain
Learning	Adaptation method	Blind	Informed	Informed
Learning	Model management	Single model	Single model	Single model

3.2. Formal specification of IDRA

3.2.1. General concepts

The general concepts to consider are based on the nature of the data streams that feed the continuous systems considered in this work. These data streams are potentially infinite and unbounded, so the methods that generate the knowledge base for these systems need to be prepared for unplanned situations. In this sense, the incremental or adaptive algorithms try to integrate the new data received in the system to improve its performance. Actions like explicit change detection, the updating or remodelling processes or the reactivity of an algorithm define its main characteristics and, therefore, the optimal scenarios to be applied to. Thus, the specific scenarios that we focus on in this assay are decisional scenarios in BI systems. We assume that in these cases searching for an accurate, time affordable and updated response would be preferable over a very fast but less accurate response. In decisional contexts, accountability is an important characteristic for providing quality decisions. Techniques such as the decision rules that allow the decision maker to understand the basis of a system recommendation are highly valued in this kind of scenario.

Following these premises, a novel decision rule technique based on high accurate time affording responses is proposed. Incremental Decision Rules Algorithm (IDRA) (Mollá et al., 2022b) is a new method that uses exhaustive decision rule sets to predict instances and update the system when the true label of this instance is received. This approach tries to provide an optimised method for decisional contexts. IDRA uses decision rules, a technique with a high readability value, and proposes a rule structure based on the empirical probability distribution of classes to classify instances while maximising the information available for the decision-maker. Thus, a decision maker can consider all the possible classes, and their associated probabilities, before an action

is taken, and therefore external conditions, such as the cost, the risk, or the trade-off of the action, can be considered. In this sense, IDRA also studies the error of the system based on a confusion matrix to allow the decision maker to be aware of those situations that are predicted with more or less accuracy by the system. Our proposal aims to approach and solve problems in data stream contexts that favours more accurate bases of knowledge with interpretable structures such as the decision rules in not so time-demanding contexts.

Three versions of IDRA with different characteristics are developed to determine which one is performing better in a variety of scenarios. IDRA combines several existing methods adapting them to the incremental logic. The Classification Rules Extraction Algorithm (CREA) (Rodríguez-Sala, 2014) is used to generate the initial rule set that IDRA will update and adapt over time. After this, and due to the exhaustive nature of these rule sets that tend to explain all cases by trying to cover all the possible combinations of explicative variables, a post-prune method is required to reduce the dimensionality. An adapted version of Rules Based on Significance method (RBS) (Almiñana et al., 2012) is used to post-prune the rule set and select the most relevant rules from the system.

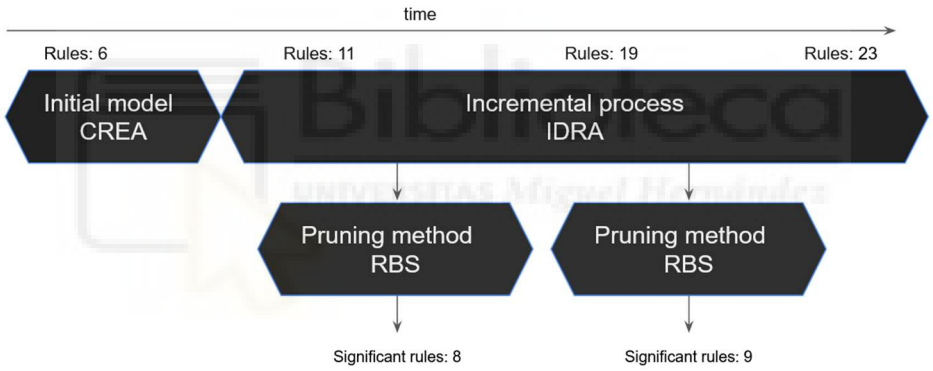


Figure 3.1 Relation between IDRA and antecedents (own source).

3.2.2. Main characteristics of IDRA

Antecedents: Classification Rules Extraction Algorithm (CREA)

Incremental Decision Rules Algorithm (IDRA) is an algorithm based on Classification Rules Extraction Algorithm (CREA) (Rodríguez-Sala, 2014) and, therefore, on Iterative Dichotomiser 3 (ID3) (Quinlan, 1986). CREA introduces a non-recursive approach to the ID3 decision tree generation process, which extracts the decision rules with no need to keep the tree structure in memory. This feature of the algorithm makes it a good antecedent for an incremental data stream adaptation since a new instance seen is assimilated by the system with no need to keep it in the memory.

This algorithm builds an ordered rule set from a discrete dataset, in which every rule has two key metrics: support and confidence. Under this premise, CREA generates all

the possible rules for an antecedent, meaning one rule for each consequent. All these rules have the same support and split up the confidence regarding the data distribution. IDRA changes the structure of the rules unifying all the same-antecedent rules in a single rule with all the classes and the probability distribution of classes. Thus, the traditional hard rules have consequents defined as $[w_i:p_i]$, where pi is the probability of the w_i class, while the IDRA structure of consequent can be defined as $[w_1:p_1,w_2:p_2,\ldots,w_k:p_k]$, where p_j is the probability of the w_j class. With this new structure, the number of rules decreases considerably, and with it the evaluation time, while at the same time more information is provided for the decision-making process. Then, we can say that our method uses soft rules to hard classify instances. IDRA decides which is the predicted class by a majority class criterion, in which the class with higher confidence in the consequent distribution is selected.

This algorithm includes a filtering method based on the Rules Based on Significance (RBS) criteria introduced by Almiñana et al. (2012) and detailed in the next subsection. With this method, only the significant rules, those located in significant regions, with relevant levels of support and confidence, are used to predict labels.

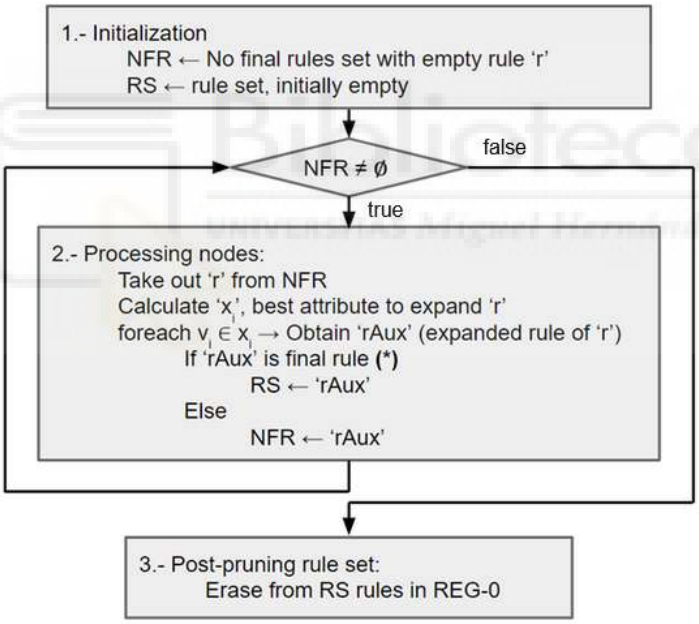


Figure 3.2 Scheme of CREA method (own source).

The CREA algorithm works with two different types of rules, those called totally expanded rules and those called not totally expanded rules. A totally expanded rule is one in which the antecedent has an assigned value for each attribute of the dataset. On the contrary, a not totally expanded rule is a rule in which one or more than one descriptive attribute is not present in the antecedent of the rule. The rules are generated following an iterative process in which the descriptive attributes are expanded based on the information gain criteria (Quinlan, 1986). This process continues until this rule is considered final, that means that all the attributes are expanded (totally expanded

rule) or that all the values of the predictive variable belong to the same class (not totally expanded rule).

The first step of CREA method (see Figure 3.2) is the initialization of the necessary variables. A rule set that will store the set of built rules (RS), a default rule and a list of no final rules (NFR). The first rule of the NFR set will always be an empty rule (a rule that has not expanded any attribute yet) used as a root that will be expanded in the iterator. After these initializations, an iterative process starts until all the extracted rules are final, so that the no final rules set (NFR) is empty. A rule is expanded based on the best information gain attribute defined with the gain ratio metric. Then, another iterative process starts in which several rules are expanded for each value (v_i) of the selected attribute (x_i) obtaining the $rAux$ rule.

Next, CREA will keep expanding attributes until the rule is either a final not totally expanded rule or a final totally expanded rule. In the case of a final not totally expanded rule, the algorithm will create a consequent with the only class it remains, while in the case of a final totally expanded rule, it will create a rule with each one of the remaining classes as a consequent. After that, the rule or rules are inserted in the rule set RS. Once the rule set is completely built, the post-pruning process starts by calculating the RBS regions of significance, and the rules are filtered based on the significance region to which they belong. With this procedure, CREA builds the set of rules RS with no need to keep any seen instance on memory and then filters the set to keep the most significant rules based on RBS post-pruning method.

Antecedents: Rules Based on Significance (RBS)

Another of the antecedents used by IDRA is the Rules Based on Significance (RBS) criteria introduced by Almiñana et al. (2012). With this post-prune method, only the significant rules, those located in significant regions, with relevant levels of support and confidence, are used to predict labels. A two-dimensional space is used to locate every rule in a given rule set (see Figure 3.3). The X axis is used to represent the frequency of the antecedent, while the Y axis reflects the rule frequency or accuracy. Thus, all rules could be represented in a specific point of the space. This space can be divided into regions of significance to establish which rules are relevant based on the location of these rules. The method, using Trial-and-Error methodology and empirical tests, defines a minimal and maximal bound of antecedent frequency and rule frequency to divide the space based on the average values of the classes and the attributes of the variables in the antecedent. The minimal and maximal axes for each of the RBS divisions (Ec_i, Ec_s, Es_i and Es_s) define the regions of significance (see Figure 3.3). These minimal and maximal axes are formed by the mean values of support or confidence of the rules that locate above or below of the axis Ec and Es. The values of Ec and Es are defined such that,

$$Ec = \frac{1}{|Class|} \tag{2}$$

$$Es = \frac{1}{\prod_i |A_i|} \tag{3}$$

The RBS method builds 3 regions of significance (see Figure 3.3). Region 1 contains those rules with high support level but low confidence, known as inverse rules. These rules define what probably will not happen given an antecedent. Region 2 contains those rules that have a high support level and high confidence level, what we call direct rules. These rules refer to what is most likely to happen for a given antecedent. Finally, region 3 defines those rules with an insufficient support level, whose classification cannot be trusted. The rules not included in these regions belong to the so-called Region 0 (blank), a region for non-significant rules, and they will be removed from the final rule set.

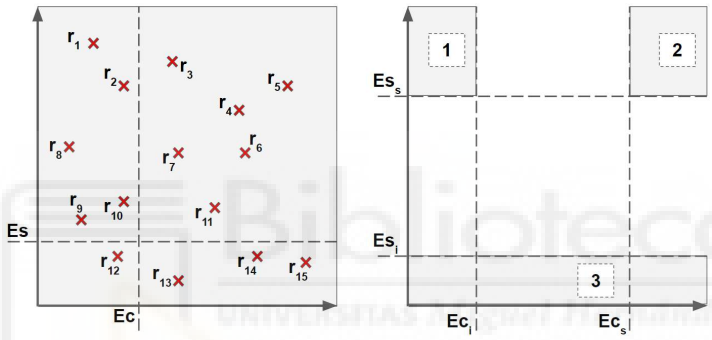


Figure 3.3 Rule distribution and significance region division of RBS (Source: Almiñana et al., 2012).

Considering the new structure of rules that IDRA uses, the definition of the significance regions needs to be rethought. If we try to represent an IDRA rule in the two-dimensional space used by RBS, we can observe that the same rule (r_i with antecedent A_i) is not represented by a single point but by a series of points that represent each of the consequents of the rule in the graph (see Figure 3.4). The axis Es_i and Es_s are the level of support while Ec_i, and Ec_s are the levels of confidence defined by the RBS algorithm to create the significant regions. For this new structure of rules, that consider the confidence distribution of the consequents, the regions need to be reformulated. The new soft rules structure of IDRA maintains the support as a metric. The axis Es_s defines in the traditional RBS method those supports that are significant (region 1 and region 2). Thus, only the upper support axis (Es_s) is considered to post-prune IDRA rule sets. Thus, following these premises, we redefine the significance space to select decision rules. In Figure 3.4, only r_1, r_2 and r_3 would be used to take decisions on the system. Axis Ec_i, Ec_s and Es_i are not considered in the post-pruning process of IDRA.

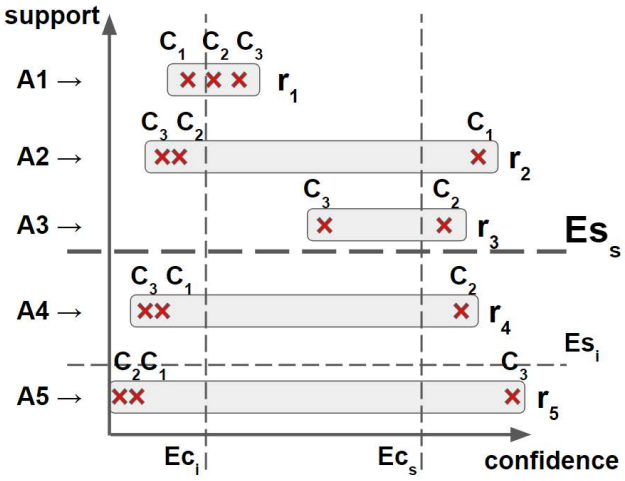


Figure.3.4 Representation of IDRA antecedent rules in RBS space (own source).

Figure 3.5 shows the comparison between the classical rule representation (left) and the soft rule structure implemented in IDRA (right) for a two-dimensional space based on the support and confidence of them. In the traditional representation, the total number of rules is five while in the soft rule structure of IDRA the number drops to only two rules with the same information being represented.

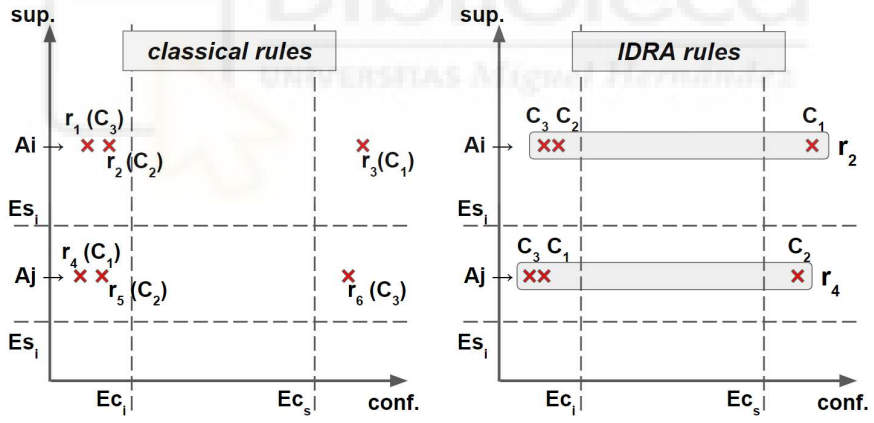


Figure 3.5 Comparison between the representation of classical and IDRA rules (own source).

Incremental Decision Rules Algorithm: IDRA

The Incremental Decision Rules Algorithm (IDRA) is a new proposal based on the traditional ID3 method (Quinlan, 1986) and the optimised rule extraction method CREA (Rodriguez-Sala, 2014). IDRA extends the functionality of CREA to data stream changing contexts adding an incremental logic to the decision rules. This new disposal will help IDRA to assimilate new data and adapt the predictive rule set. IDRA is especially designed for decisional data stream contexts. The management environment will favour an accurate and time-affordable response over a very-high-speed and less precise solution. IDRA trains an initial model that provides the rule set

that will evolve incrementally over time, adding or ignoring rules. This initial model is built using the CREA algorithm, with a slightly different approach to the rule structure. This new version uses soft rules, with a (empirical) probability distribution of classes, to make a hard classification, so that only one class is given as a prediction. After this initial model is generated, the incremental learning stage begins, in which the new instances are integrated in the model.

To make predictions, IDRA works with a decision rule set (DRS) that contains those rules used to predict or support decisions, and a potential rule set (PRS) that includes those rules that are not yet relevant enough, although they may change in importance at some moment during the stream. A rule might belong to the decision rule set or the potential rule set based on its support and a variable boundary defined following the RBS minimum support axis (Es_s). This boundary is recalculated after a given number of examples; therefore, the decision rule set (DRS) is constantly changing its size. Thus, the algorithm ignores the less relevant rules until they become important inside the rule set and uses the most relevant ones to classify until they lose importance. The division between the DRS and the PRS, and the threshold used to split the two rule sets is represented in the Figure 3.6. In IDRA, a new rule is added to the potential rule set every time an unseen example comes. This means that when that rule gains enough importance (there are a minimum of occurrences) this will be used to predict similar examples that come to the system. Thus, IDRA provides a consistent system that can predict with a high accuracy rate and with a certain control over the complexity of the rule set due to its post-prune method.

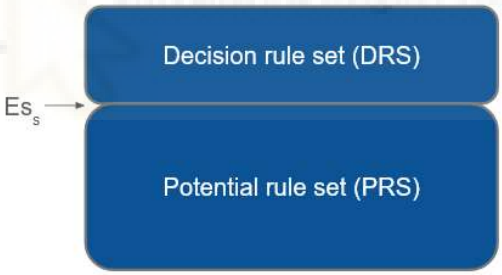


Figure 3.6 Division between the DRS and the PRS using the Es_s limit (own source).

IDRA introduces rules whose consequents comprise a probabilistic distribution of different class labels, what we call soft rules. This means that a single rule might contain several confidence rates relating to different classes, so there is no single metric to measure the inference capacity of that rule. Unifying the confidences for all the consequents of the rules for IDRA helps to reduce the number of rules of the systems and therefore the search times (see Figure 3.5). Also, this characteristic is especially valuable in decisional systems where decisions might need to be made knowing the whole available information. Nevertheless, when we want to compare the different rules and their classification quality, we lose the dimension of confidence because each rule has several metrics. To solve this problem, a new dimension is proposed in order to be able to compare the predictive quality of a group of rules. This

new metric is based on the entropy of the confidence metrics of a specific rule. To solve this problem, IDRA uses the entropy of the precision rates as a metric to compare rules. The lower the entropy is in a rule, the greater the difference among their class label confidences and therefore, the better capacity to predict. Thus, a rule with a single-class confidence of 100% would have a Shannon entropy of 0, while a rule with a confidence equally distributed into two classes would have a Shannon entropy of 1. The best inference capacity is given by a Shannon entropy of 0 and the worst inference capacity by the maximum possible Shannon entropy is given by this formula:

$$H_{max} = log_2(\#class) \tag{4}$$

This entropy measure will be used to detect those rules with low classification value that are replaceable in the system. This process is happening after a given number of instances in the stream, called evaluation window, a required parameter of the algorithm, together with the entropy tolerance level.

3.3. Functional description of IDRA

In this section, the functionality of IDRA and its general characteristics are described to understand the underlying logic of this incremental algorithm. Due to the industrial character of this work, the code of the algorithm is not publicly distributed. Nevertheless, pseudocode of IDRA in its different versions is provided and explained in this assay in order to understand and contextualise the results shown in next sections.

Figure 3.7 shows a high-level scheme of the proposed algorithm when an instance arrives to the system. When a new instance is observed, IDRA finds the best fitting rule in the decision rule set (DRS), which gives a class label. After that, the algorithm searches for the perfectly fitting rule, that is the rule that matches all its antecedents with all the attributes of the example. Should that rule exist, the statistics are updated, while if it does not exist, a new rule is induced and added to the potential rule set (PRS). After a given delay, the true label arrives or can be calculated, so the algorithm updates the statistics of the model and sets the accuracy and error rates. IDRA informs these rates using a confusion matrix, so decision makers could see which predictions are reliable and which ones should be more deeply studied in a decisional situation. Beside that possibility, the algorithm is continually tracking its performance and updating this value based on the predictive results of the model. The accuracy shows how the model is performing during the stream and it could be used to detect changes that have an impact on it, in a given evaluation window.

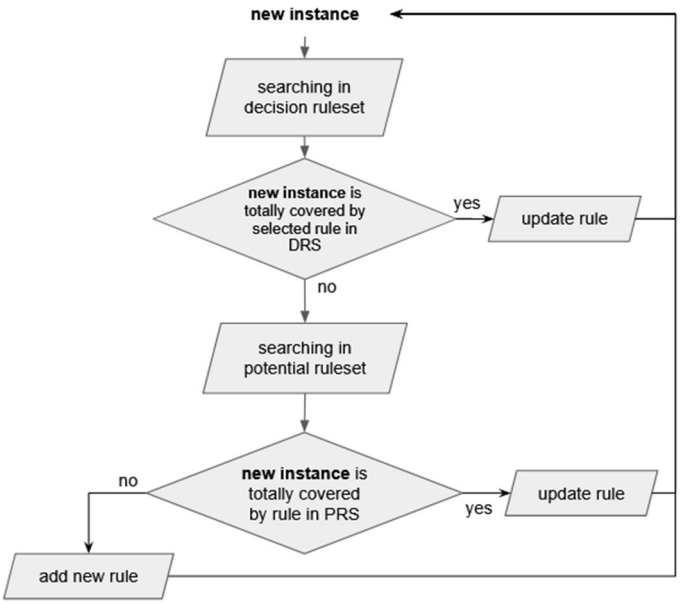


Figure 3.7 High-level flowchart of IDRA (own source).

The evolution of the algorithms from static to dynamic logic arises new challenges derived from the nature of the data. These new challenges require explicit or implicit tactics to deal with them and approach solutions to minimise their impact. Concepts like the evaluation window, the forgetting method or novelty assimilation are studied in the next subsections to expose the different solutions proposed by IDRA for these challenges.

Novelty assimilation

When we study a potentially infinite data stream, any of the descriptive variables or the class could suffer the unexpected emergence of a new value never seen before. Techniques such as novelty detection methods are learning tasks that identify new or unknown concepts that learning systems did not consider during the training process (Gama et al., 2010, Chapter 9). Novelty detection represents an important challenge in online systems since concepts are constantly exposed to changes. To deal with this potential problem, IDRA does not integrate an explicit novelty detection method, but it takes action to include all new concepts in the system either if they happen in the descriptive variables or in the objective variable. If a new value is seen in a descriptive variable, the algorithm will create new rules considering this new concept from that moment on, while if a new class arises, all the rules will progressively integrate this new value in their consequents. This moment might cause a drop in accuracy in the system and a drastic change in rules. In these cases, the versions of IDRA that include explicit forgetting methods may be preferable.

Forgetting methods

Another of the challenges that derive from infinite data streams is the way the systems deal with deprecated knowledge. When a change happens in a data stream, the previous rules used to model might become obsolete and, therefore, the accuracy of the system will drop if it keeps using them. In this case, each version of IDRA has a different approach to deal with this situation. In the first version, the algorithm will trust its own logic to replace those deprecated rules in the system for those with better results. The second version of IDRA, however, includes an explicit detection method that uses the accuracy drop to detect the change in data. The third version of IDRA integrates a specific method to detect changes in the data distribution. Both, the second and third versions rebuild the model with the last data available in case a change is detected. These different approaches will be extended in subsection 3.3.

New metrics proposed

IDRA selects the decision rule that needs to be used based on the order of the rules in the rule set. This order will be determined by a criterion that could be the support or a new metric called trending. Other metrics could also be used to order the set of rules but only these two have been tested so far. The trending metric proposes a method to reflect a partial support belonging to the last evaluation window in the stream. Using this metric as criteria will allow IDRA to forget really fast the previous knowledge so it might be preferable in uncertain contexts. The metric used as a criterion in IDRA directly affects the rules that are selected to make decisions due to the nature of the algorithm. The decision rule set (DRS) and the potential rule set (PRS) are ordered based on this metric. The algorithm selects the first rule available in the DRS to make the prediction, so the order of this rule set directly affects the selected rule.

Figure 3.8 shows how to compare the trending metric to the support metric. A data stream is analysed by IDRA and the corresponding metric is calculated based on the established criteria. Every rule in the rule set has a support or trending metric and its value will define which rule is selected from the decision rule set. The support metric takes its first value from the initial model built by IDRA. After that, the model will update that value every time that rule is used by the system. Considering that this rule is used once in every window, the final support value would be incremented by s_w.

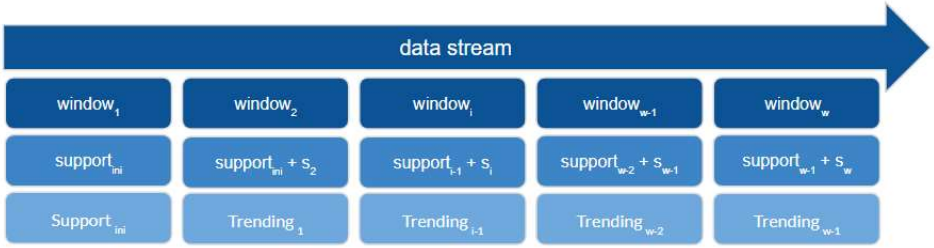


Figure 3.8 Comparison of trending and support metrics for an IDRA rule (own source).

In the case of the trending metric, the initial value is also defined by the first model generated by IDRA. From that moment, the trending will change its value based on a partial support of the previous window. Considering the same case as in the support metric, if the evaluated rule is used once in every window, the trending of this rule would be one. It is important to highlight that this criterion is also applied to the calculation of the RBS values, so it will also affect the value of the minimum support axis (Es_s) now considering the window size to establish a relative Es_s. This metric searches to give IDRA a reactive behaviour in the case of high impact recurrent changes scenarios. Trending is only considered in the first version of IDRA since that reactive behaviour is achieved by other versions of IDRA with explicit change detection and drastic rebuilding process.

Table 3.2. Change of distribution of classes for a *rule$_i$* that is used 250 times every window.

$rule_i$	Window 1	Window 2	Window 3
class A	100	130	40
class B	60	40	60
class C	90	80	150
Trending of $rule_i$	**250**	**250**	**250**

In this case, one version of IDRA uses this metric not only as a partial support of the rule (see Figure 3.8) but also as a partial probability of the classes (see Table 3.2). This method searches to quickly act upon changes in the distribution of the predicted classes. To illustrate this situation, we consider the case of a $rule_i$ that is used a total of 250 times inside a specific evaluation window. The trending rate of that $rule_i$ would be 250 for all the windows studied (three) and the trending rate of the classes A, B and C would distribute this trending rate of the $rule_i$ among them all. Thus, if a change in the distribution of the classes happens, the model would be fast enough to detect it and predict based on the newest data available in the stream. In Table 3.2, a specific example of this situation is illustrated. Slight changes in the class importance are perceived between the first and second evaluation windows. Nevertheless, in the case of the third evaluation window, we can clearly see that a change in the occurrences of the consequents happens and therefore, the predictions of the model might be affected when using $rule_i$ to predict. In this case, the model will start labeling the instances with the class C when using $rule_i$ for the next windows since a change in the probability of the classes has been detected.

Evaluation window

As presented in the forgetting method challenge description, a variety of techniques that aim to remove deprecated instances from a system use an evaluation window. This interval searches to establish a point in time when the previous knowledge learnt by an algorithm is forgotten and new distribution in data can be integrated easily. In this sense, some techniques could be more sensible to this threshold depending on the use they give it inside the learning process and the importance and presence it has in the

overall of the process. In this sense, the input parameter of the size of the evaluation window can have a great impact on the results of the algorithm while using the trending metric in the IDRA algorithm. If the evaluation window is larger, IDRA would consider more instances and its behaviour would be more constant. Nevertheless, if the evaluation window is smaller, the algorithm will quickly change, and its behaviour would be more changeful. Thus, we can see that an adequate size of the evaluation window might have a great impact in the performance of an algorithm that uses this technique.

The challenges presented in the previous subsections and the proposed methods contained by the different versions of IDRA are summarised in Table 3.3. These characteristics are integrated or not in the three versions of IDRA depending on the logic of that specific approach. The different versions of the proposal are extended in the subsection 3.3.

Table 3.3. Characteristics included in each version of IDRA.

	IDRAv1	IDRAv2	RIDRA
Evaluation window	X	X	
Explicit forgetting		X	X
Trending metric	X		
Novelty assimilation	X	X	X

3.4. Versions of IDRA

In this subsection, the different versions of IDRA are explained in detail. Based on the general concepts and the functional description explained in previous sections, the differences between versions are explained in the next subsections.

3.4.1. IDRA version 1 or IDRAv1

IDRA version 1 is the first version of the presented algorithm. This method is the version introduced in the work by Mollá et al. (2022b) and represents the first adaptation of the CREA algorithm to streaming contexts. This version works with an auxiliary rule set, composed by the totally expanded rules of all the not totally expanded rules of the rule set ($DRS + PRS$). The totally expanded rules contained in this auxiliary rule set will replace the not totally expanded rules in the DRS and the PRS if they do not meet the given requirements regarding the entropy level of the rules. This parameter, entropyTolerance (see Algorithm 1), is defined in the input of the algorithm, so different values can be tried to test how the demanding or lax thresholds might affect the algorithm performance.

Since this first version of IDRA has no explicit forgetting method implemented, it might use the trending to work with a partial metric that refers to a distribution within

a specific evaluation window. The use of the trending or the support metric would define the criteria used to select rules and to order the set of rules. These criteria would be received by the system in the form of an input of the algorithm.

Algorithm 1: IDRAv1: Incremental Decision Rule Algorithm (version 1)

```
input:  D: training data set
        DS: data stream
        Window: window size
        criteria: { support, trending }
        Entropy_tolerance
output: RS: rule set
begin
    Let RS←{ }
    Let defaultrule r←∅
    Let NFR←{r}
    while NFR is not empty do
        r←Extract any rule from NFR
        xi←BestGainRatioAttribute(r, D)
        foreach vi ∈ xi do
            r'←ExpandRule(r, xi=vi)
            if r' is a final rule then
              RS←RS ∪ ExpandRule(r',[w1:p1, w2:p2, . . . , wk:pk ])
            else
              NFR←NFR ∪ {r'}
    UpdateRBSThreshold()
    defineDRSandPRS(RS, RBSThreshold)
    Let NTER←{r ∈ RS, r is not totally expanded rule}
    AuxRS←ExpandAllRules(NTER)
    foreach instance {x, y} in DS do
        r←bestRule(x, DRS, criteria)
        if exists TotallyCoveringRule(x, RS) then
            updateTotallyCoveringRuleStats(x, y, RS)
        else
            PRS←PRS ∪ newTotallyCoveringRule(x, y)
        if r is not totally expanded then
            update(r, AuxRS)
        if Window check is Required then
            UpdateRBSThreshold( )
            defineDRSandPRS(RS, RBSThreshold)
            checkEntropy(NTER, AuxRS, entropyTolerance)
            if criteria is trending then
              resetTrends(RS)
        sort(RS,criteria)
```

Algorithm 1 (IDRAv1) requires as an input a training data set, a stream data set, the size of the evaluation window, the criteria used to select the rules and an entropy threshold. Thus, IDRAv1 will start creating an initial model based on the CREA method (see Figure 3.2) and then, a streaming data process will follow. IDRAv1 starts creating an initial rule set (RS), a default empty rule (r) and a no final rule set (NFR) containing this empty default rule r. After this initialization process, a loop starts extracting any rule of the NFR set and then selecting an attribute based on the information gain criterion (Quinlan, 1986) with the BestGainRatioAttribute function and expanding all its values with the ExpandRule function. After that, IDRAv1 checks

if this expanded rule is final, inserting r to the rule set RS if it is, and to the no final rule set (NFR) if it is not. After all the rules are final and the NFR set is empty, the initial rule set is built, and the data streaming process starts. Based on the RBS post-pruning method and the described changes applied to it, the so-called RBS values are initialised using the UpdateRBSThreshold function. Based on this threshold, the decision rule set (DRS) and the potential rule set (PRS) are delimited with the defineDRSandPRS function. This method divides the RS, based on the RBS threshold, in a set of rules that are used to make predictions (DRS) and a set of rules that are expected to gain importance before being used in the decision-making process. After this, an auxiliary rule set ($AuxRS$) is created for all the not totally expanded rules ($NTER$). This auxiliary rule set contains all the totally expanded rules of each not totally expanded rule. That means that if there is a not totally expanded rule in RS, such that, $x = \{x_1, x_2, \dots, x_{i-1}, x_{i+1}, \dots, x_{c-1}\}$ in $AuxRS$ will be a set of totally expanded rules for all $v_i \in x_i$.

After the creation of the auxiliary rule set, the incremental learning process starts. For each instance received by the data stream, a prediction is made using the bestRule function, which selects the best rule of the DRS based on a given criteria. Several criteria are tested to study the behaviour of the algorithm in each of them. One of the metrics studied is the traditional support. Apart from this one, the new trending metric is proposed to study how the algorithm performs under a fast adaptation method based on forgetting data distributions after two evaluation windows. Next in the algorithm pseudocode, IDRAv1 searches for a rule in RS (DRS and PRS) that totally covers the descriptive attributes of the instance, updating the statistics of the rule if it is found or creating a new rule in the PRS if it is not. Thus, the algorithm slowly integrates new instances not observed before, increasing their importance after each occurrence. When a new set of explanatory attributes (or antecedent) is observed enough times, the associated rule climbs position in the PRS and it might move to the DRS if it meets the significance conditions established in the RBSThreshold. In the case that the best rule in RS is a not totally expanded rule, the statistics of the corresponding auxiliary rules of $AuxRS$ will also be updated to build auxiliary rules, as significant as possible, whose consequents represent the updated class distribution shown on the data stream. After that, if a window check is required, several statistics within the model are updated. The RBS threshold is recalculated and, therefore, the DRS and the PRS are divided again based on this new limit. At this point, the entropy of the consequents of each not totally expanded rule is calculated based on the Shannon entropy (Shanon, 1948). Those rules whose Shannon entropy is above the entropyTolerance limit, will be replaced by the equivalence rules of $AuxRS$. The checkEntropy function will also erase the replaced rules from the $NTER$ and the equivalent rules from the $AuxRS$. Thus, those not totally expanded rules with consequents that possess a not clear classification option (the Shannon entropy is far from the perfect case, $H = 0$), evolve to more concrete rules that might classify with better Shannon entropy rates (closer to 0). After that, in the case that the criteria used to select the predictive rules is trending metric, the trends of the rule set RS are reseted using the function resetTrends. Finally, IDRAv1 sorts the RS based on the selected criteria (support or trending) and rules

might reorder, producing changes in the system rule preferences when a classification action is taken. It is important to remember that IDRA selects the first found rule in the rule set that meets the antecedent, so the order of the rules is important to define which rules are used to predict.

3.4.2. IDRA version 2 or IDRAv2

IDRA version 2 is the next version of the presented algorithm. This version erases the auxiliary rule set for the totally expanded rules and includes an explicit forgetting method. This forgetting process includes the rebuilding of the set of rules, the recalculation of the RBS threshold and the new definition of DRS and PRS when the performance of the algorithm is affected to a certain level. This performance is measured using the accuracy of the model, so this process takes place when a drop in this metric is detected within a window. Thus, if the model suffers, within the same window, a drop in its accuracy of more than a given threshold (accuracy drop tolerance), the algorithm will consider that a change has taken place. When this happens, the model will redefine the set of rules (DRS and PRS), as well as the RBS threshold. The accuracy drop tolerance needs to be related with the evaluation window size in order to avoid unnecessary model rebuilding, as well as the problem arising from disregarding a meaningful impact in accuracy.

IDRA version 2 also calculates the entropy of the rules belonging to the rule set, nevertheless this metric is only used to compare the classification quality of different rules, and there is no entropy tolerance metric involved in this version. Although the algorithm is prepared to use either support or trending criteria to select the decision rules used to predict, only the support metric is considered for this version. This decision is taken based on the fact that this second version includes an explicit forgetting method, and the value of the support and class probabilities update when a rebuilding need is detected. Thus, the algorithm does not need any partial metric to be used to fulfil this purpose. IDRA version 2 aims to simplify the searching process in the main rule set (RS) and the auxiliary rule set ($AuxRS$) handled by the first version of IDRA. Thus, IDRAv2 erases the auxiliary set of rules and adds a naïve rebuilding method when a drop of accuracy happens between two evaluation windows. These changes might considerably reduce the number of rules contained by the system and therefore, the times of search needed to operate with them.

Algorithm 2: IDRAv2: Incremental Decision Rule Algorithm (version 2)

```
input:  D: training data set
        DS: data stream
        Window: window size
        Accuracy_drop_tolerance
output: RS: rule set
begin
    Let RS←{ }
    Let defaultrule r←∅
    Let NFR←{r}
    while NFR is not empty do
        r←Extract any rule from NFR
        xi←BestGainRatioAttribute(r, D)
        foreach v_i ∈ x_i do
            r'←ExpandRule(r, x_i=v_i)
            if r' is a final rule then
              RS←RS ∪ ExpandRule(r',[w1:p1, w2:p2, . . . , wk:pk ])
            else
              NFR←NFR ∪ {r'}
    UpdateRBSThreshold()
    defineDRSandPRS(RS, RBSThreshold)
    foreach instance {x, y} in DS do
        r←bestRule(x, DRS, support)
        if exists TotallyCoveringRule(x, RS) then
            updateTotallyCoveringRuleStats(x, y, RS)
        else
            PRS←PRS ∪ newTotallyCoveringRule(x, y)
        if Window check is Required then
            if Accuracy drop is detected then
              rebuild(RS)
            UpdateRBSThreshold( )
            defineDRSandPRS(RS, RBSThreshold)
        sortBySupport(RS)
```

3.4.3. Reactive IDRA or RIDRA

The third version of IDRA presented in this assay is the reactive IDRA or RIDRA. This version of the proposed algorithm does not wait to detect a drop in the accuracy of the model. Instead, this version includes an explicit method to detect possible changes in the data distribution of the objective variable. The explicit technique used to identify a shift is the well-known adaptive window method, ADWIN by Bifet and Galdavà (2007). This method changes the logic of a fixed sliding window for a statistically meaningful window size recalculated online. ADWIN resizes the evaluation window considering the rate of change observed in the data contained by the window itself.

ADWIN creates a sliding window W with the most recent instances x_i read by the system (see Algorithm 3). The size of this window will increase until the average of a sufficiently large sub-window is distinct enough from the rest of the data of the window. When that happens, the older examples of the window are dropped from the system. The value of ε_{cut} defines how different two subwindow distributions need to

be to decrease the size of the window to the newest examples. The ε_{cut} threshold for a partition of two subwindows W_0 W_1 of W is computed such that: n_0 and n_1 are the lengths of W_0 and W_1, while the length of W is n such that $n = n_0 + n_1$. In addition, $\hat{\mu}W_0$ and $\hat{\mu}W_1$ are the averages of the values contained in W_0 and W_1, and μW_0 and μW_1 their expected values. To obtain totally rigorous performance guarantees the threshold is defined such that:

$$m = \frac{1}{\frac{1}{n_0}+\frac{1}{n_1}} \ (harmonic\ mean\ of\ n_0\ and\ n_1) \tag{5}$$

$$\delta' = \frac{n}{\delta'}\ and\ \varepsilon_{cut} = \sqrt{\frac{1}{2m}} ln \frac{4}{\delta'} \tag{6}$$

This statistical test checks the average value of data in W_0 and W_1 considering ε_{cut} as the threshold. In this case, δ' is used to avoid problems with multiple hypothesis, since different sizes of W_0 and W_1 are tested, and the global error is expected to be below δ.

When this technique is applied to IDRA, the model will consider all the examples contained in the sliding window defined by ADWIN. If a change is not detected in the variable distribution, the sliding evaluation window will enlarge and IDRA will keep the incremental process without applying any forgetting method. Nevertheless, if a meaningful difference between two sub-windows is detected, the method will assume that a change has happened in data, and therefore IDRA will rebuild the model using data from the most recent window.

Algorithm 3: ADWIN: Adaptive Windowing Algorithm

```
1  Initialize Window W
2  foreach t > 0 do
3      do W ← W ∪ {xt}
4          repeat Drop elements from the tail of W
5              until |μ̂W0−μ̂W1|≥ ∈cut holds
6                for every split of W into W=W0·W1
7          output μ̂W
```

Thus, this reactive version of IDRA analyses the distribution of data to detect a meaningful change in order to optimise the rebuilding process. Using this additional method to identify shifts in data adds some complexity to the algorithm and, therefore, some time. A substantial improvement in accuracy might justify the incrementation of time. The reactive version of IDRA or reactive IDRA searches to optimise the algorithm by reducing the unnecessary rebuilding of the set of rules. It can happen that this rebuilding process is computationally less complex than the analysis carried out by the ADWIN method to avoid it. This depends on factors such as the data stream nature, the number of explicative variables and the possible values of these variables. This situation might be considered when deciding which version of the algorithm is better in a specific scenario.

Algorithm 4: Reactive IDRA: Reactive Incremental Decision Rule Algorithm

```
input:  D: training data set
        DS: data stream
output: RS: rule set
begin
    Let RS←{ }
    Let defaultrule r←∅
    Let NFR←{r}
    Let ADWIN Window W
    while NFR is not empty do
        r←Extract any rule from NFR
        xi←BestGainRatioAttribute(r, D)
        foreach v_i ∈ x_i do
            r'←ExpandRule(r, x_i=v_i)
            if r' is a final rule then
              RS←RS ∪ ExpandRule(r',[w_1:p_1, w_2:p_2, . . . , w_k:p_k ])
            else
              NFR←NFR ∪ {r'}
    UpdateRBSThreshold()
    defineDRSandPRS(RS, RBSThreshold)
    foreach instance {x, y} in DS do
        r←bestRule(x, DRS, support)
        addElement(instance, W)
        if changeDetected(W) then
            rebuild(RS)
        if exists TotallyCoveringRule(x, RS) then
            updateTotallyCoveringRuleStats(x, y, RS)
        else
            PRS←PRS ∪ newTotallyCoveringRule(x, y)
        sortBySupport(RS)
```

Algorithm 4 shows the pseudo-code of the reactive version of IDRA (RIDRA). It illustrates how the model is now only rebuilt when a change is detected in the window. This change allows the system to optimise the learning process since the reaction of the algorithm is justified by a meaningful change in data distribution. Nevertheless, the continuous logic of the ADWIN method presents a limitation when applied to RIDRA. Since the values evaluated in the adaptive window are normalised between 0 and 1, RIDRA can only work with binary classes. This limitation can lead to the choice of other versions over this one due to incompatibilities with the analytic problem.

Also, this new approach dispenses with parameters that conditionate the results of the algorithms. In this sense, in the previous versions, restrictions like the evaluation window would determine the moment in which the model is evaluated, and therefore defining a non-optimum value might lead to a lower efficiency of the system. Removing the parameters of the scheme means that the algorithm is not exposed to these kinds of inappropriate rates and therefore, reduces the probability of being affected by them. As a consequence of not including parameters, and as it can be seen in its pseudo-code, this reactive version of IDRA does not implement a post-pruning method. For IDRAv1 and IDRAv2, the RBS method is used to establish a threshold (called RBSThreshold) that allows the system to divide the rule set RS in a decision rule set (DRS) and potential rule set (PRS). This threshold was determined based on

the total amount of instances seen by the system, and it was recalculated after a given number of instances (evaluation window). By erasing the window size parameter, the recalculation process of the RBSThreshold needs to be reformulated as well. The only moment in which RIDRA implements a recalculation process is when a change in the window is detected (see line 23 of Algorithm 4). This change can happen after an unpredictable number of instances, so the recalculation of this threshold and, therefore, its adequation might be unreliable too. To avoid this, the post-pruning method is erased from the process and all the rules are included in the same rule set ordered by its metric of support. This fact, added to the implementation of the adaptive sliding windowing technique, might increase the time the algorithm spends in the search of the rules. Moreover, this new implementation provides the system with a more independent behaviour due to its non-parametric nature.

Chapter 4
Computational experience

Once the novel incremental technique has been presented in Chapter 3, Incremental Decision Rules Algorithm (IDRA) is now tested in this chapter against the antecedent CREA and the state-of-the-art method VFDR. The design of the computational experience, as well as the data used to test the different algorithms are studied. Also, an analysis of the required parameters used by the first two versions of IDRA is presented, in addition to the final results of the algorithms and the discussion of the outcomes.

The content of this chapter is based on the paper published in the journal Mathematics under the title *Incremental Decision Rules Algorithm: A Probabilistic and Dynamic Approach to Decisional Data Stream Problems* (Mollá et al. 2022b). This publication presents the results of the initial version of the proposed method, IDRAv1. The computational experience of the new versions of the algorithm are presented in this chapter for the first time.

4.1. Design of the computational experiences

This section presents the conditions of the computational experiences launched to test the proposed novel approach presented in this work. Considering the application of IDRA is framed to data stream context, all the presented experiments will be carried out following the same premises. All the versions of the algorithm, as well as the static approach (CREA) and a state-of-art reference, are exposed to the same conditions in the computational experiments. A wide variety of experiences are designed using different conditions and type of data, both real and simulated, to try to generalise the conclusions of this study. Exposing the methodologies to different situation through a broad computational experience, we aim to study how valuable and solid the proposal is, and how it is the relation with the available methods. Using a state-of-art simulation technique guarantees that the generated data is generalized for any kind of problem and that there is no specially designed scheme that improve the results of IDRA. Thus, the simulations could be repeated to compare these results with other new solutions, knowing that the accuracy levels compiled in this chapter are the mean accuracies of a total of five experiments, with different seeds, per scenario. Using simulated data streams, allow us to expose the methodologies to a large variety of error conditions to prove the value of high accurate IDRA solution, as well as to repeat the experiments and represent an average metric that prove to be generalised. Thus, we aim to study how the proposed methodology behaves in a large number of scenarios, to demonstrate that the results provided have sufficient quality, and to compare it with the state-of-art methodologies. On the other hand, using real data stream permits to test the methodologies in real decisional scenarios. With the public marketing dataset, we guarantee the repeatability of the experiment, since it can be freely download, tested with a new methodology and compared with the techniques exposed in this work. Also, testing the methods in a real scenario of a client, shows the potential applicability to real decisional cases seen in the industry. In addition, this scenario allows us to make conclusions through the power and usability of our solution, and it push us to design and define how we want IDRA to communicate with the user of the final DSS.

The method used to evaluate all the models is the interleaved test-then-train or prequential evaluation. In this scheme, every individual example is used to test the model before it is used to update it. Thus, the model is always being tested on instances never seen before. With test-then-train evaluation, all examples are considered to compute accuracy, while prequential only uses those contained in a sliding window. In this case, the test-then-train scheme is used with no sliding window.

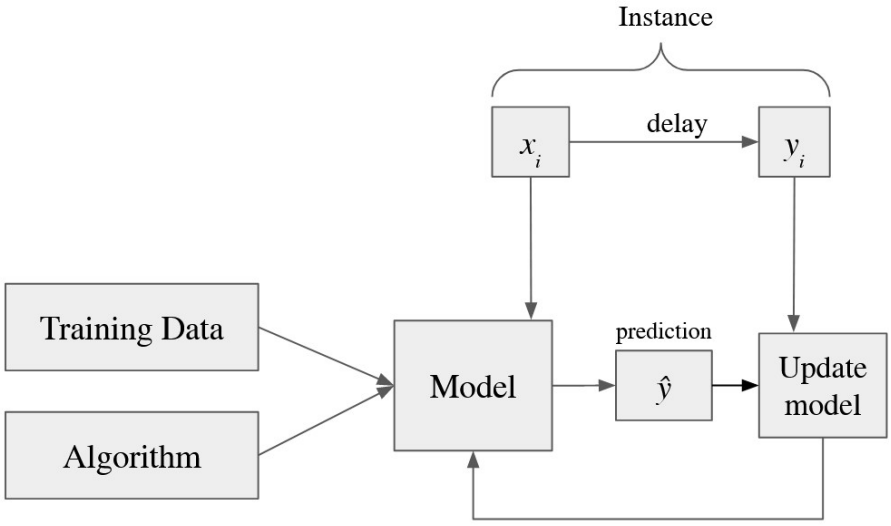


Figure 4.1. Scheme of the test-then-train logic when a new instance (x_i, y_i) arrives (own source).

Figure 4.1 shows a high-level scheme of the test-then-train evaluation method used for the experiments. In this approach, the initial model is built using a batch of previous data that we refer to as train data. After that, a new instance arrives to the system. An instance is formed by a set of discrete descriptive attributes, $\boldsymbol{x}$ and a predictable variable, y, i.e., instance $I = \{\boldsymbol{x}, y\}$ such that $\boldsymbol{x} = \{x_1, x_2, \dots, x_{c-1}\}$ where the values of x_i are defined as $v_i \in x_i, i = 1, 2, \dots, c-1$.

The model makes a prediction $\hat{y}$ based on the descriptive attributes received x_i. After a given time, the real label of the objective variable, y is received by the system. At that moment, the prediction made by the model is compared with the real value collected and a metric of the system error is updated. At this moment we say the model is being tested, since we can see how accurate our model is when exposed to never-seen-before examples. After the partial test, the knowledge acquired by the examination of this last instance (x_i, y_i) is integrated into the model in a new training process. Thus, a single instance has been used to test the performance of the system and, after that, to integrate new knowledge into the incremental model.

In the case of the static algorithms, this evaluation method is not applicable since no updating process takes place. To evaluate this type of algorithm in the same conditions, an adaptation needs to be done to the test-then-train process. In these cases, the model is trained only with the train data and the rest of data received by the system is used to test it, with no re-training or incremental process involved. In real productive scenarios, after a model is generated and implemented in a system, it is exposed to new instances never seen before in order to make predictions and compare the results with the later received real classes. To evaluate the traditional static algorithms in this work, a productive environment is simulated under the data stream conditions specific for each case.

The main purpose of the designed simulated experiments is to study the behaviour of the proposed algorithm and all its versions in abrupt changing data stream contexts, using different magnitudes of impact. All IDRA versions are compared with the state-of-the-art decision rule algorithm VFDR and a static rule oriented optimised ID3 (CREA) in terms of performance and time. In this sense, a wide variety of experiments are designed to study how different types of error affect each algorithm in simulated and real scenarios. All the algorithms are implemented in Python except for the state-of-the-art method, VFDR, whose version is integrated in the MOA framework (Bifet et al. 2010) and therefore implemented in Java. The computer used for these experiments has a Processor Intel i7−10710U CPU @ 1.10GHz, 16GB RAM, Windows 10 20H2. The Python version used to run the experiment is Python 3.9.13.

One of the limitations of this study is related to the time performances of the tested algorithms. The version of VFDR used in this assay is integrated in the MOA framework (Bifet et al. 2010) and therefore implemented in Java, while CREA and all the versions of IDRA are written in Python. Even in this case, when the hardware conditions are the same, the different implementation languages of the algorithms might make the time comparison not so relevant. Nevertheless, the different nature of the algorithms is not a reason why the two algorithms cannot be compared (Destefanis et al. 2016). Thus, the time results are shown as a valuable metric of the algorithms to evaluate which problems could be handled assuming certain time requirements and to compare the different versions of IDRA and these ones with the static approach. The study compares the mean accuracy of each method and presents the computational times to know whether an algorithm is suitable for a specific problem or not.

4.1.1. Simulated data

The proposed algorithm is supposed to run in decisional context with data streams. To evaluate the performance of the new method and try to explore different scenarios, several datasets are simulated under the same conditions with different seeds. The use of the same seeds in different computers might lead to slightly different sets of data. This fact might justify marginal variations in the results of future experiments using the same algorithm implementations. In this sense, a total of five experiments are launched for each scenario and the mean accuracy obtained by the different algorithms is used to compare their performance in each situation. All simulated datasets have a total of 60,000 instances, 600 of them are used to train the initial model and the remaining 59,400 to simulate the stream. These proportions are taken from the work by Kosina and Gama (2013).

To simulate the changing data, the SEA method (Streaming Ensemble Algorithm) proposed by Street and Kim (2001) is used. This technique generates abrupt changing datasets formed by three numerical variables (x_1, x_2, x_3), two of which are relevant (x_1 and x_2). These numerical explicative variables take values between 0 and 10, while the objective variable has only two values labelled following the next expression:

$$x_1 + x_2 \leq \theta \quad (7)$$

In this expression, the x_1 and x_2 are the values of the two relevant variables, and θ is the changing threshold used to generate different concept drifts. Should the addition of the first two attributes ($x_1 + x_2$) be less than or equal to θ, the example would be labelled as negative (−) and if it is more than θ, it would be labelled as positive (+). Following this logic, four concepts (three drifts) are generated by varying the value of θ, each one of them containing 15,000 examples. The default values of θ are 9,8,7 and 9.5. All the datasets simulated under these premises generate distributions around 64% positive labelling and 36% negative labelling. This method is selected mostly as it fits the simulation requirements of a decisional context. The changes are abrupt, however its magnitude does not have a high impact on the data. That explains the decisional environments quite well, in which the context may change abruptly yet the scope is in most cases controlled and delimited in an expected range. An abrupt change with a high impact is not seen so often in decisional contexts, however it is possible. The behaviours of all the algorithms are also studied in a highly demanding scenario with a considerable difference between the values of θ. In this case, the threshold limits are set to 9, 15, 8 and 14, generating concept drifts of a high impact.

Simulated data aims to study the algorithm behaviour over a variety of scenarios as realistic as possible. To reproduce the conditions of real data, the simulated streams are exposed to errors that might be caused by miscaptured data in noisy environments, as well as mislabelled classes produced by human mistakes. To measure the algorithms' tolerance to these kinds of error, different levels of noise and mislabelling are included in simulated datasets. Each type of error is generated following a different logic. In the case of noise, this is generated by adding a random error, comprehended between the negative and positive value of the maximum error. That means that the classes were produced following this expression:

$$x_1 + x_2 + noise \leq \theta \quad (8)$$

The maximum error is calculated based on the noise rate and the attributes range. In this case, using SEA generation, the variables take values between 0 and 10, so if the noise level is fixed to 10, the error oscillates within the $[-1, +1]$ range. In the case of mislabelling, the process consists of changing the correct class to the wrong one in each number of examples, depending on the defined level of error. Thus, if the level of mislabelling is set to 5%, 3000 out of 60,000 instances are incorrectly labelled. In real scenarios, this type of error might be caused mainly by human error or wrong assumptions from the system, and it has a considerable impact on the performance of the model.

Table 4.1 shows the different simulated scenarios in which the methods are tested, as it can be observed, a total of 5 times each. The proposed scenarios include the perfect case (No error) with neither noise nor mislabelling, noisy scenarios (10% and 20% rates) and mislabelled scenarios (5% and 10% rates). The most used scenario in the literature is the one that includes 10% noise; therefore, we will consider it to be the most important one. For this reason, the simulated data for the high impact scenario includes a rate of 10% of noise. Other scenarios with different rates and types of errors are also studied to delimit which algorithm is the best for each situation, and which kind of error most affects each method.

Table 4.1. Scenarios, noise and mislabelling levels and threshold limits for simulated datasets.

Scenario type	Experiment	Noise	Mislabel	Threshold θ
No error	5 data streams mean	0%	0%	[9,8,7,9.5]
Noisy	5 data streams mean	10%	0%	[9,8,7,9.5]
Noisy	5 data streams mean	20%	0%	[9,8,7,9.5]
Mislabelled	5 data streams mean	0%	5%	[9,8,7,9.5]
Mislabelled	5 data streams mean	0%	10%	[9,8,7,9.5]
High impact	5 data streams mean	10%	0%	[9,15,8,14]

4.1.2. Real data

Besides the simulated datasets that provide the experiment with a deep study of a variety of conditions that might affect the algorithms, real datasets are also used to test all the methods. In this sense, a set of publicly available data is used, as well as a private data set provided by Sala Group, a car dealership company sited in Alicante province.

The public dataset (Moro et al., 2014; Machine Learning Repository, 2012) contains data from the marketing campaigns of a bank and classifies the subscription of each client to a product. This dataset has a total of 45,211 instances with 11 discrete columns containing information about different aspects of a client. In this case, to simplify the analysis, the five most relevant attributes of each dataset are selected according to gain ratio criteria (Orenes et al., 2021). To the best of our knowledge, this dataset contains neither errors nor mislabelling, so in the next sections it will not be considered. Also, this set of data includes no explicit concept drift, so similar performances of the static and the incremental algorithms (CREA and IDRA versions) are expected.

In the case of the private real dataset, it contains information about the repairments and maintenance service of a car workshop in the mentioned vehicle dealership company. This real private dataset will allow us to extract meaningful business conclusions applied to an authentic scenario, and therefore, contrast these deductions with the opinions of the experts leading in this sector. This scenario is a great opportunity for the new proposal to be exposed to a real production case, and it is especially important in our case due to the industrial nature of this work. A subset of the available database is selected based on expert criteria. The main purpose of this scenario is to analyse the customer retention of the company in the last years, focusing

specifically on the maintenance car service. In this case, the analysed subset contains a total number of 262,387 instances with 5 descriptive variables and a binary class that describes the return or loss of a client in the maintenance service. All the descriptive variables are previously discretised in a way that there are two descriptive variables with 7 values, one with 6 values and others with 3 and 2 values. These variables define a variety of qualities of the owner and the vehicle that are studied in order to characterise each case. In the case of the studied variable, the value of return or loss of a client is not intrinsically gathered in the database but estimated by the system using an amount of time to label a client as lost. Thus, if a client has not come back to the service in an established time, they will be labelled as a missed client, while if they come back before, they will be labelled as retained client. From this process, a given client could be labelled as a lost client and then come back to the service after some time. That means that the system contains a given rate of mislabelled instances. In the case of the studied dataset, the mislabelling rate is around 3,18%.

The use of these two different datasets allows us to test the proposed algorithm with public and contrasted data, as well as applying to real productive cases where the expert knowledge can help us to interpret the results and draw relevant conclusions. Thus, not only the accuracy of the algorithm is proved, but also the value of the results obtained by the methods. Due to the industrial interest of the results of this research work, the code of algorithm is not publicly available. Besides this, the public dataset can be found, tested, and studied with many other state-of-the-art algorithms. Even though this dataset is simplified and shuffled and therefore, the experiment has some random factor, this case proposes a common ground to test other similar techniques from future or previous research.

These two real datasets allow the techniques to be tested over some authentic cases of study and give other researchers the opportunity to compare their proposals with ours using the publicly available dataset, and they allow us to extract meaningful conclusions of the results contrasting them with the opinions of experts in the studied sector using the private dataset. These real scenarios, combined with the six simulated scenarios, try to study the behaviour of the methods in a variety of situations (no error, two rates of noise error, two rates of mislabelling error and a high impact concept drift scenario, see Table 4.1), providing a deep research plan to test the algorithms.

4.1.3. Conditions of the experiments

The experiments carried out in this chapter aim to illustrate how different algorithms with different logic perform in a variety of data stream scenarios and compare them with our proposal. The algorithms that will be studied in this chapter are: (1) CREA, a static algorithm commonly trained with batch data and tested using cross-validation, exposed now to data streams that resemble real productive environments. (2) IDRA and all its versions, the incremental adaptation of the CREA algorithm, which progressively updates its rules with the incoming data streams with no need to keep

examples on memory. (3) VFDR, an adaptive algorithm able to deal with nominal and continuous attributes, as well as ordered and unordered rule sets. To compare the methods in similar conditions, ignoring the basic differences between the static, incremental and adaptive logic, all the algorithms are based on majority class criteria to make predictions and generate ordered rule sets with discrete datasets.

In the case of the simulated datasets, with 60,000 examples, 600 of them are used to train the initial model, while the remaining 59,400 are used to test the static algorithm and test-then-train the incremental and adaptive ones. The same conditions are replicated in the case of the real datasets. A total of 600 instances are used to train and the remaining instances to test or test-then-train depending on the logic of the algorithm. All methods are fed with the test data in the form of a stream. In order to compare the evolution of the different models, an evaluation window is fixed to take measures of the model accuracy. In all cases, we assume a common window size of 1,000 instances. It is considered that this window size is suitable for providing an appropriate detection of changes, while not taking unaffordable time to process the instances. In this sense, the mean accuracy obtained by the model for a given dataset is again used to calculate the mean accuracy of all datasets within the same scenario. In the same way, the mean time spent by an algorithm in a scenario is used as a comparative metric. In the case of accuracy, its evolution is considered relevant to compare the behaviour and qualities of the methods. Every simulated scenario is understood as a set of error parameters. A total of five datasets are simulated for each of those scenarios and the mean metrics of each studied dimension are reported to compare the algorithms.

In the case of the real scenarios, different approaches are considered depending on the dataset. For the unordered bank data, since the examples are unordered, five streams are generated by randomly shuffling the instances of the dataset. All these datasets are shuffled, and the original order is not included among the experiments to avoid an order based on an erroneous attribute. Thus, if an attribute has two values A and B, and the order of the instances is based on this attribute, the learning process of the algorithms might only include one of these values. To avoid this bias, the dataset is mixed in five different ways to test the algorithms. On the contrary, for the car data, since the set is ordered in a meaningful way, only one experiment can be executed on this occasion. The conclusions extracted in this case are only relevant when the original instance order is preserved. The mean accuracies and times obtained by all the algorithms in simulated and real data stream scenarios are shown in Section 4.2.

Table 4.2 comprises all the conditions of the experiments, those using simulated data streams and those using real datasets. In all the cases, the system simulates the reception of instances in the form of a data stream even if the datasets are previously generated. This process is replicated in all the scenarios, so the same conditions are used in order to be able to compare the results.

Table 4.2. Name, nature and known error levels of all the tested scenarios

Scenario name	Nature	Error type	Experiment	Noise	Mislabel
Simul 0	Simulated	No error	5 data streams mean	0%	0%
Simul 1	Simulated	Noisy	5 data streams mean	10%	0%
Simul 2	Simulated	Noisy	5 data streams mean	20%	0%
Simul 3	Simulated	Mislabelled	5 data streams mean	0%	5%
Simul 4	Simulated	Mislabelled	5 data streams mean	0%	10%
Simul 5	Simulated	High impact	5 data streams mean	10%	0%
Bank	Real	Unknown	5 shuffled data streams	-	-
Cars	Real	Mislabelled	1 ordered data stream	-	3,18%

As already mentioned, the rule structure of IDRA differs from the one of CREA. The design of the rules in IDRA comprises all the consequents from all the same-antecedent rules in one that represents the probability distribution of all the possible classes. Trying to reduce the differences between the search times to compare specifically the traditional static logic against the incremental logic of the algorithms, the structure of the rules of the CREA method will be modified to be the same as the IDRA ones. Thus, the CREA method will have rules that contain a distribution of probabilities as a consequent and the prediction will be based on the majority class criteria, as in the case of the rest of algorithms. This slight modification of the static method aims to create equivalent information structures that allow comparing the results of the methods based more on their logic than on their information architectures. In the case of CREA and IDRA, their rule structure is equivalent for all the experiments launched in this work. Nevertheless, for the state-of-art method, VFDR, the rule structure is not changed. This algorithm follows different tactics in the rule generation and changing the structure of the decision rules might affect the nature of the algorithm. In this sense, the traditional rules used by VFDR are conserved for the experiments.

4.2. Parametric study of the IDRA versions

In the different versions of the presented algorithm, several parameters or conditions, such as the tolerance level, accuracy bound, window size, etc. are established prior to the execution of the model. As previously mentioned in this assay, an inadequate value of a parameter might directly affect the performance of an algorithm in certain scenarios. Since these rates are significant and sensible for the accuracy of a method, a specific study is carried out to test different values for these key criteria. Only two of the three presented versions of the IDRA algorithm have parameters to be studied, so only empirical results for the versions IDRAv1 and IDRAv2 are presented. This study will use simulated data to report the performance and select the best criteria for these two versions (IDRAv1 and IDRAv2). Following the same premises exposed in previous sections, the evaluation window used in this parametric study has a size of 1,000 instances. Even if the evaluation window might also affect the results of these

two approaches, a single value is used in all the experiments since this study is considered out of the scope of this work. A wide study of this matter will be proposed as further research. The criteria chosen in next subsections will be used to perform the comparative experiments of the parametric versions (IDRAv1 and IDRAv2) in Section 4.3.

4.2.1. IDRA version 1 or IDRAv1

IDRA version 1 or IDRAv1 (see Algorithm 1) has parameters that need to be defined at the beginning of the analysis, such as the entropy tolerance (entropyTolerance) and the selection rule criteria. Regarding the entropy tolerance, the results of IDRA version 1 (IDRAv1) correspond to a demanding level defined by the formula,

$$\frac{log_2(\#values)}{\#values} \tag{9}$$

This high threshold has led to the only use and maintenance of highly accurate not totally expanded rules in the set of rules. In this assay, only this threshold is studied, but other laxer tolerance levels could be studied to determine the importance of this boundary in the quality of the rule sets.

Moreover, some criteria need to be established in IDRAv1 to order the set of rules and, therefore, to determine which decision rules are selected before, as well as using or not an explicit forgetting method in the system. In this case, as it was already explained when describing the trending metric, the two possible criteria for the system are either the trending or the support of the rules. In the case of the trending, not only a different metric is used to choose the decision rule, but also new criteria are used to select the majority class of the probability distribution. Here, the majority class would be based on the trending criteria and therefore, the system would pick up the class that was most seen in the previous evaluation window for that specific rule. Thus, the selected criterion would fix a strict and naïve forgetting method in the case of the novel metric, while no explicit forgetting method would be used in the case of the support.

In these computational experiments, both metrics are tested in order to determine if the novel criteria is suitable for any data stream scenario and if it improves the traditional metrics. Trending metric, as previously mentioned, is highly dependent on the size of the window. In this case, even if this fact is known and, to simplify the experiments, only one window size is evaluated. Every 1,000 instances, a process of evaluation, parameter recalculation and reset of trends, if necessary, will take place.

Table 4.3. Experiments designed to test the IDRAv1 algorithm.

Experiment name	entropyTolerance	criteria
IDRAv1 support	$\frac{log_2(\#values)}{\#values}$	support
IDRAv1 trending	$\frac{log_2(\#values)}{\#values}$	trending

The first version of IDRA, IDRAv1 uses a criterion, either the support or the trending metric, to order and select the rules, as well as the classes in a probability distribution based on occurrences. The use of the trending will provide the system with an aggressive forgetting method, while using the support will suppose a slower adaptation but a higher knowledge retention rate. Each of these characteristics might be preferable depending on the situation. In the next tables (accuracy rates in Table 4.4 and times in Table 4.5), the results of the algorithm IDRAv1 are shown for the 6 simulated scenarios available.

Table 4.4. IDRAv1 accuracy (%) based on different criteria for simulated scenarios.

Criteria	No error	Noised		Mislabelled		High impact
	0%	10%	20%	5%	10%	Noise 10%
Support	**92.99**	**89.64**	**85.01**	**81.02**	72.53	82.00
Trending	85.12	85.79	83.77	80.66	**72.88**	**82.51**

Table 4.5. IDRAv1 time (s) based on different criteria for simulated scenarios.

Criteria	No error	Noised		Mislabelled		High impact
	0%	10%	20%	5%	10%	Noise 10%
Support	35.12	35.50	32.70	35.39	41.85	35.44
Trending	35.26	35.53	32.21	34.86	42.62	35.74

As it can be seen in Table 4.5, the time differences between the two criteria are not relevant in any of the studied scenarios. Regarding the accuracy, the situation is different (see Table 4.4). Even if in the mislabelling scenarios the two criteria obtain similar results (0.64 points more for support metric in 5% rate case and 0.35 points more for trending metric in 10% rate), in the no error scenario and both of the noisy ones, the support criterion obtains important differences compared with the trending one. In the case of the noisy scenarios, the support improves the trending accuracies in 1.24 points for the 20% rate of noise and 3.85 points for the 10% rate. In the no error case, the gain of the support metric rises to 7.87 points, a pretty significant difference. In contrast, the results of IDRAv1 using the trending metric are slightly better (0.51 points) than using the support metric in the high impact scenario.

Based on these findings, after analysing the differences in accuracy for all the error conditions and since no important variation is presented in the time, we conclude that the support is the best fitting metric for IDRAv1 in these simulated scenarios.

Therefore, these criteria will be used to test IDRAv1 in the comparative experiments shown in Section 4.3.

4.2.2. IDRA version 2 or IDRAv2

IDRA version 2 or IDRAv2 (see Algorithm 2) reduces the number of parameters respecting the first version. In this case, there is no auxiliary rule set, so no entropy tolerance threshold needs to be established. Also, only support criterion is used in the experiments since an explicit forgetting method is included after a given drop in accuracy is detected. The required parameter for this version is defined by Accuracy_drop_tolerance, the tolerance of the system against a drop in accuracy.

IDRA version 2 or IDRAv2 is tested with several tolerance rates in order to determine the best performing percentage for the system with an evaluation window of 1,000 instances. The variation of this threshold might lead to different performances of the algorithm for the same data stream. The use of a too high limit might lead to a never-rebuilding situation in which no forgetting process takes place in the system. To try to avoid this possibility, the studied thresholds are fixed to relative low values. Also, a naïve remodelling criterion is reviewed among the designed experiments. This approach tries to study what will happen in the system if the base of knowledge was forgotten and rebuilt again with no criteria involved, after a given number of instances, in this case the 1,000 ones of the evaluation window.

Among the studied parametrization of the experiments, we can find five different scenarios in which the accuracy drop rate for IDRAv2 is fixed from a 0% rate to a 1% in intervals of 0.25%. We consider the influence of this threshold to be worth studying in this assay, so a variety of experiments with different levels is presented in this chapter. IDRAv2 is studied from a completely naïve scenario that uses no tolerance level (0%) to a less sensitive threshold (1%) as the rebuilding criteria, going through some intermediate values. In total, five different rates are tested as the accuracy drop tolerance rate: 0%, 0.25%, 0.5%, 0.75% and 1%.

As stated before, the second version of IDRA, IDRAv2, uses an accuracy drop threshold to determine when it is necessary for the system to erase all the previous knowledge and redefine the decision rules set. The results of the accuracies and the times obtained by each studied quota are shown in Table 4.6 (accuracy) and Table 4.7 (times) for the simulated scenarios.

Table 4.6. IDRAv2 accuracy (%) based on different thresholds for simulated scenarios.

Threshold	No error	Noised		Mislabelled		High impact
	0%	10%	20%	5%	10%	Noise 10%
0.00%	76.41	76.68	77.44	82.06	**76.57**	77.67
0.25%	85.60	85.14	83.70	80.32	74.56	81.52
0.50%	87.47	87.46	84.05	82.29	74.70	82.60
0.75%	**92.73**	**90.37**	**85.63**	**83.05**	74.88	**85.04**
1.00%	**92.73**	**90.37**	**85.63**	82.84	75.66	**85.04**

Table 4.7. IDRAv2 time (s) based on different thresholds for simulated scenarios.

Threshold	No error	Noised		Mislabelled		High impact
	0%	10%	20%	5%	10%	Noise 10%
0.00%	38.18	42.27	48.68	54.60	66.61	40.11
0.25%	24.73	25.10	25.95	29.94	33.77	27.21
0.50%	21.50	22.33	24.57	25.25	30.77	24.26
0.75%	**19.00**	**20.36**	**23.09**	**24.15**	**30.27**	21.57
1.00%	20.41	21.26	23.73	25.60	31.08	**21.51**

Based on the results showed in Table 4.6 and Table 4.7, the threshold criteria used in the comparative experiments is selected. Table 4.6 shows that those thresholds with better accuracy rates are the 0.75% and the 1%. These two limits have the same results except for a couple of scenarios (mislabelling) in which 0.75% rate beats the 1% for the 5% mislabelling level, and the 1% rate beats the 0.75% threshold in the 10% mislabelling scenario. Table 4.7 shows that the times of IDRAv1 are slightly faster in the case of the 0.75% of accuracy drop. Therefore, since the accuracy levels are quite similar and the times are better, the 0.75% accuracy drop threshold is selected.

4.2.3. Reactive IDRA or RIDRA

In the case of the third version of IDRA, Reactive IDRA or RIDRA (see Algorithm 4) due to its reactive nature dispenses with parameters. This variant waits to detect a change in the distribution of data to act upon the model and modify it in order to avoid an accuracy drop. That supposes that this version makes the system more independent and self-adaptive to changes with no parameters among the input of the algorithm. This characteristic supposes an improvement in the self-sufficiency of the system and, therefore, in a reduction of the human intervention in the parameterization of the method. As mentioned before in this chapter, the input of the algorithms might lead to better or worse performances depending on the adequacy of the selected criteria. Reducing or even dispensing with all the parameters, make the system less exposed to wrong specifications that lead to inefficient schemes.

In this case, RIDRA incorporates a sliding window technique to detect when a new sample deviates from the average value of the window enough to determine that a concept drift has taken place, and therefore, an action, in this case the rebuilding of the base of knowledge, is performed. To elude the use of parameters, the reactive version of the IDRA algorithm renounces the post-pruning process included in the prior adaptations. Thus, RIDRA builds an independent model that is capable of incrementally learning from new examples coming from a data stream, at the same time that it can forget the previous knowledge base, in case a concept drift happens without considering any parameter.

4.3. Results of the computational experiences

As previously presented in Section 4.2, the parameters for the two first versions of IDRA (IDRAv1 and IDRAv2) are selected based on a study of the performances of different criteria or values. The selected criterion for IDRAv1 version is the support, while the rate of accuracy drop for IDRAv2 is 0.75%. In this section, the results of the computational experiments are shown for all the versions of the novel approach IDRA in both simulated and real scenarios. The comparison and discussion of these results will be led in Section 4.4, while the general conclusions of this work will be presented in Chapter 5.

4.3.1. Simulated scenarios results

Once the best parameter values are selected based on a previous study, the experiments are run using a variety of different scenarios. In the case of the simulated scenarios, as it is presented in Subsection 4.1.1 (see Table 4.1). In these next paragraphs, the different tables summarising the obtained results in the experimental evaluation will be shown. Table 4.8 shows the mean accuracies of all models while Table 4.9 shows the mean times obtained for each algorithm in the described scenarios. With these measures, it can be quickly identified whether an algorithm is suitable for the time requirements of a system, or a faster method should be used.

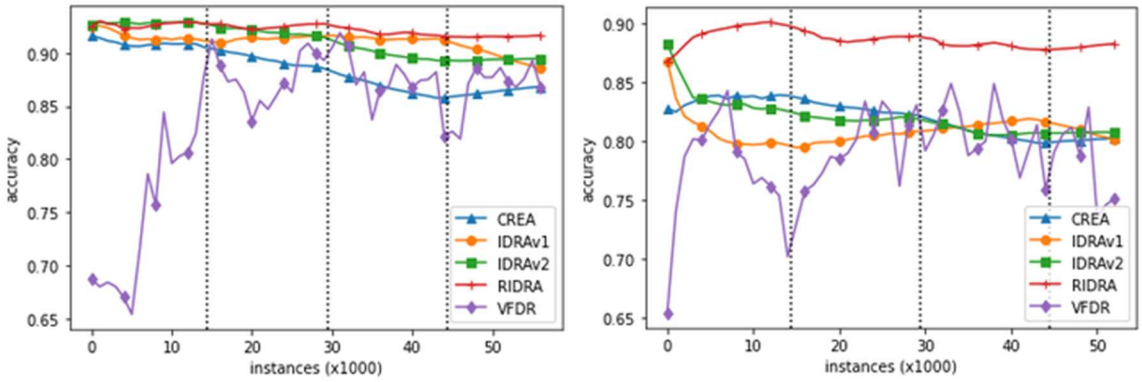
Table 4.8. Mean accuracy (%) of algorithms for simulated scenarios.

	No error	**Noised**		**Mislabelled**		**High impact**
	0%	**10%**	**20%**	**5%**	**10%**	**Noise 10%**
CREA	89.97	87.91	84.04	83.62	75.64	79.43
IDRAv1 support	93.0	89.60	84.80	81.40	74.40	84.10
IDRAv2 (0.75%)	92.73	90.37	85.63	83.05	74.88	85.04
RIDRA	**93.37**	**92.02**	**88.50**	**88.79**	**82.81**	**90.15**
VFDR	82.98	83.56	81.53	78.05	72.57	82.43

Table 4.9. Mean times (s) of algorithms for simulated scenarios.

	No error	**Noised**		**Mislabelled**		**High impact**
	0%	**10%**	**20%**	**5%**	**10%**	**Noise 10%**
CREA	17.98	18.85	19.21	20.45	22.94	18.32
IDRAv1 support	35.12	35.50	32.70	35.39	41.85	35.06
IDRAv2 (0.75%)	19.00	20.36	23.09	24.15	30.27	21.57
RIDRA	29.88	32.71	33.97	39.67	47.97	35.13
VFDR	**4.42**	**4.22**	**4.53**	**4.26**	**3.85**	**3.45**

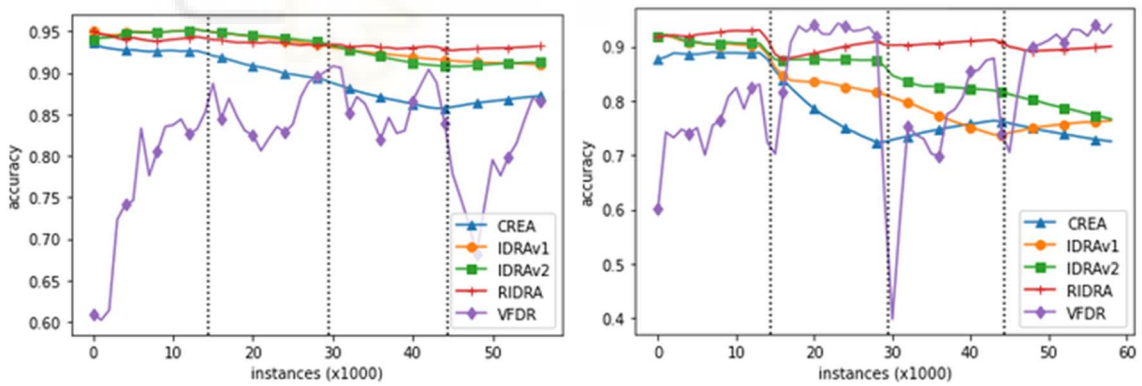
In Figure 4.2, we can see the evolution of the accuracy rate for all the algorithms in two of the simulated experiments. The performance of CREA, VFDR and IDRA versions in the 10% noise and the 5% mislabel scenario are represented in the graphs.



(a) (b)

Figure 4.2 Accuracy evolution of all the studied algorithms in two of the simulated data scenarios. The concept drifts are represented with dashed lines. (a) Accuracy evolution of all the studied algorithms in one of the simulated data streams with 10% noise. (b) Accuracy evolution of all the studied algorithms in one of the simulated data streams with 5% mislabelling.

Figure 4.3 presents the accuracy distribution of the studied methods for (a) a non-error scenario and (b) a high impact concept drift simulation. Clearly, the algorithms have a more stable behaviour in the first scenario, being the three versions of IDRA pretty similar to each other. On the contrary, the performances in the high impact scenario are way more diverse.



(a) (b)

Figure 4.3 Accuracy evolution of all the studied algorithms in two of the simulated data scenarios. The concept drifts are represented with dashed lines. (a) Accuracy evolution of all the studied algorithms in one of the simulated data streams with no error. (b) Accuracy evolution of all the studied algorithms in one of the simulated data streams with high impact changes.

4.3.2. Real scenario results

After the analysis of the simulated scenarios that allow us to study the behaviour of the novel and previous techniques in a variety of situations, the real case results are presented. In this sense, as commented previously in this assay, two real datasets are used to test the algorithms, a publicly available set that compiles information from telemarketing campaigns of a bank and a private dataset composed by the repairs of a car workshop. In Table 4.10, we can see the mean accuracies of all the methods, while in Table 4.11 the mean times for these two cases are shown.

Table 4.10. Mean accuracy (%) of algorithms for real scenarios.

	Bank marketing	**Car workshop**
CREA	93.29	78.47
IDRAv1	93.45	**85.92**
IDRAv2 (0.75%)	93.43	85.87
RIDRA	**93.50**	85.71
VFDR	88.83	83.05

Table 4.11. Mean times (s) of algorithms for real scenarios.

	Bank marketing	**Car workshop**
CREA	11.44	67.04
IDRAv1	13.50	90.31
IDRAv2 (0.75%)	13.56	87.03
RIDRA	20.17	166.59
VFDR	**2.30**	**25.44**

Figure 4.4 shows the accuracy progression along the real data analysis. We can appreciate that the versions of IDRA obtain similar results in both cases (a and b), but CREA differs in the case of the car workshop. IDRA versions and CREA have more stable behaviours in the two scenarios, while VFDR have more variable measures of accuracy.

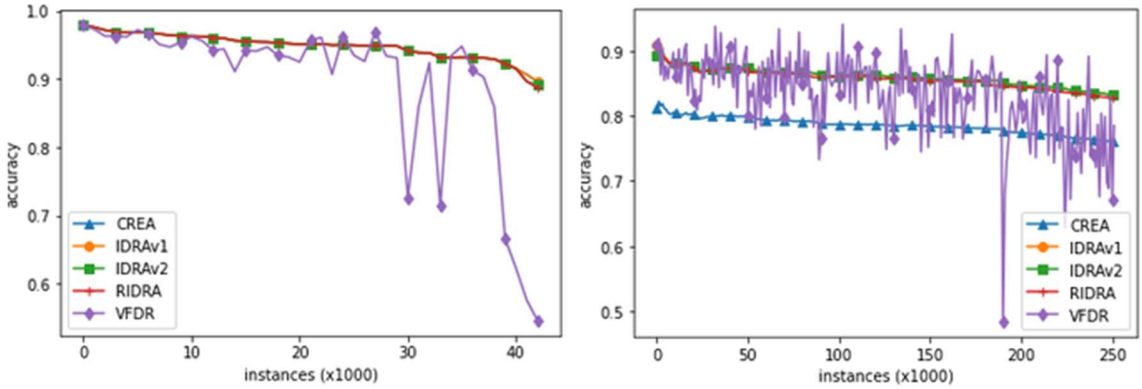


Figure 4.4 Accuracy evolution of all the studied algorithms in the two real data scenarios. (a) Accuracy evolution of all the studied algorithms in one of the generated bank data streams. (b) Accuracy evolution of all the studied algorithms in the car workshop real data stream.

4.4. Discussion

In previous sections, IDRA, in its different versions, is tested in several scenarios together with CREA and VFDR methodologies. The presented experiments try to test the different methodologies in a variety of scenarios using a commonly used simulation method, SEA (Street and Kim, 2001) and adding different levels of error, as well as using real data publicly available and real decision data granted by a client of Teralco Solutions. The results of IDRA in all the simulated scenarios, as well as in the real ones, show a more stable behaviour with, in general, better mean accuracy results, especially when we talk about RIDRA version. Thus, all these experiments, covering a wide variety of situations, reveal the inner potential of the presented methodology in continuous scenarios where the time requirements are not extremely demanding like in very fast schemes. In the case of decisional contexts, systems are exposed to changes but with instances arrival times significantly less critical. These scenarios create opportunities for accurate incremental solutions like IDRA to model the system, using, also, valuable highly explainable structure such as the decision rules. In these cases, the advantages that IDRA methodology could entail in decisional contexts provide benefits respecting to other solutions, both adaptive and static.

4.4.1. Simulated scenarios results

Table 4.8 shows the mean performances obtained by IDRA and its versions. We can see in this table that the accuracies of the proposed technique are, in most cases, higher than the static algorithm and that the state-of-the-art technique for the simulated scenarios. Reactive IDRA or RIDRA obtains the best results of accuracy for all the

scenarios, even though the other versions of IDRA (IDRAv1 and IDRAv2) also perform better than the static and state-of-the-art techniques in most cases. To simplify the comparison, the discussion will try to focus on the results of this last version of the proposed algorithm, RIDRA, due to its better performance. However, the other versions of IDRA are also included in the discourse.

In the benchmark case, the simulated scenario with 10% of noise, IDRAv1 obtains almost 2 points more of mean accuracy than CREA and 6 points more than the VFDR implementation (see Table 4.8). In the case of IDRAv2, the difference with the CREA algorithm is almost 3 points while the difference with VFDR rises to almost 7 points. IDRAv2 improves the performance of IDRAv1 by 0.73 points. In the case of the Reactive IDRA or RIDRA, the gain in accuracy with the static method CREA rises to almost 4 points, while for VFDR it reaches almost 9 points of difference. If we compare the versions of IDRA, RIDRA obtains the best performance for this 10% noisy scenario with 92.02% of accuracy against an 89.64% of IDRAv1 (2.38 points less than RIDRA) and 90.37% of IDRAv2 (1.65 points less than RIDRA). This improvement in accuracy of RIDRA is trade-off for an increase in the time consumption of the process. Compared with the other versions of IDRA, RIDRA obtains similar time results than IDRAv1, but it increments the times of IDRAv2 by around 10 seconds in noisy and no error scenarios and almost 15 seconds in the mislabelling and high impact ones. Regarding the time consumption, the best algorithm is clearly the state-of-the-art method, VFDR, with less than 5 seconds in all the simulated scenarios.

The comparison between models has its highest difference in the non-error simulated scenario (see Table 4.8) in which RIDRA gets 93.37% of mean precision while CREA gets almost 90% and VFDR 83%. The difference between the third version of IDRA (RIDRA) and the state-of-the-art method rises to more than 10 points for this simulated scenario. The time results of these two methods, VFDR and RIDRA, are even though quite substantial (see Table 4.9). For this non-error scenario, VFDR runs the model in a total time of 4.42 seconds, while RIDRA takes almost 30 seconds. That means RIDRA analyses 1.988 instances in one second, while VFDR is capable of processing 13.439 instances per second. This is a substantial difference in processing capacity, but so it is the improvement in accuracy. As already explained in this work, there are situations where a more precise response in an affordable time is preferred to a very fast less accurate return. RIDRA proves to add value to the existing methods for those specific cases.

If we compare the times obtained by the version of IDRA for the simulated scenarios (see Table 4.9), we can see that RIDRA and IDRAv1 have similar time response while the second version (IDRAv2), improves these times consumptions from 10 to 15 seconds, depending on the scenario and its error conditions. The time results of CREA and IDRAv2 are quite similar. Even if CREA is slightly faster in some cases, the differences between the accuracy rates justify and prove the improvement given by this version over the static approach. The fact that IDRAv2 could assimilate new data distributions incrementally and remodel when an abrupt change is affecting the system,

could be seen as a significant advantage over CREA. In the case of IDRAv1 and RIDRA, this advantage is also present, but the times in these two cases are slightly slower than CREA's one. The accuracy rates of IDRAv1 and IDRAv2 are higher than the static method of CREA in most cases, except for the mislabel error scenarios, while the results of RIDRA improve the static approach in all scenarios.

In the high impact simulated scenario (see Table 4.8), the best mean performance is achieved again by the reactive version of the proposed algorithm, RIDRA, with an accuracy rate of 90.15% and a difference of almost 8 points over VFDR implementation and more than 10 points over the static version. The RIDRA version improves IDRAv2 by 5 points and IDRAv1 by 6 points. The performance of RIDRA in the high impact scenario proves its highly adaptive capacity, having an accuracy rate that could be compared to the performance of the rest IDRA versions in the best performing scenarios with noise. If we compare this result with the other algorithms, CREA and VFDR, we can see that the performance of Reactive IDRA (RIDRA) in high impact scenario is slightly better than the best accuracy rate achieved by CREA in the non-error case (0.17 points of difference) and significantly better than the best performance obtained by VFDR in the 10% noise scenario (7.17 points of difference). This fact shows the magnitude of the improvement achieved by the Reactive version of IDRA in this scenario, even though the high impact case shows an extremely changeable and inconsistent data stream that might not be found so often.

In the mislabelling scenarios (see Table 4.8), CREA achieves better accuracy rates than IDRAv1 and VFDR for both rates and improves the IDRAv2 performance in the 10% mislabelling case. This better performance of CREA may be caused by its static nature and the data error distribution. The mentioned implementations include incremental logic in a way the mislabelled instances are incrementally added to the system. CREA, with its static logic, might not include these mistaken examples in the base of knowledge. Despite all this, CREA cannot overcome the results of RIDRA in these scenarios. The mislabelling error has more effect in the performance of all algorithms than the common noise error. In the case of 10% of mislabelled examples, the performance of IDRAv1 drops 7 points from the 5% mislabelling case while for IDRAv2 drops more than 8 points. The least affected algorithms in this increment of the mislabelling rate are VDFR with a drop of 5.48 points and RIDRA with 5.98 points. In all the implementations of IDRA and for VFDR, the mislabelling error, at both the 5% and 10% rate, have more of an effect on the mean accuracy than high impact changes.

If we want to discuss the impact of the different types of errors in a more general way for the different methods, we can compare the drop of accuracy in different scenarios. This comparison can be seen in Table 4.12. This table shows that, in the case of different impacts in the given algorithms, the minimum value of difference between the best- and worst-case scenarios (no error or 10% noise scenario, with the 10% mislabel) is achieved by RIDRA. The second algorithm with less difference is VFDR but, if we look carefully at the best and worst performances, we can see that the worst accuracy of RIDRA and the best accuracy of VFDR only vary by less than 1 point.

That indicates that the accuracy improvement obtained by RIDRA respecting the benchmark technique VFDR is significantly better for these simulated scenarios.

Table 4.12. Differences between the best performing scenario and the worst performing one for all the studied methods.

Algorithm	Best scenario	Best accuracy	Worst scenario	Worst accuracy	Difference
CREA	No error	89.97%	10% mislabel	75.64%	14.33%
IDRAv1	No error	92.99%	10% mislabel	72.53%	20.46%
IDRAv2	No error	92.73%	10% mislabel	74.88%	17.85%
RIDRA	No error	93.37%	10% mislabel	82.81%	10.56%
VFDR	10% noise	83.56%	10% mislabel	72.57%	10,99%

Table 4.13. Instances per second rates for all the studied algorithms in a 10% noise scenario.

Algorithm	Instances per second	Time (ms) per instance
CREA	3.183	0.31
IDRAv1	1.690	0.59
IDRAv2	2.947	0.34
RIDRA	1.834	0.54
VFDR	**14.218**	**0.07**

Table 4.13 shows that the VFDR algorithm has a faster rate of classification capacity than the other algorithms. Nevertheless, as previously explained, in decision contexts with no such demanding speeds, the accuracy rate may prevail over time. The results of each algorithm in a specific scenario may help to get an idea of their potential effectiveness in a particular case or industry. If the data stream arrival time is below these rates or the decision system can afford to have a small delay in classification, then the proposal is suitable for the scenario or production case.

Regarding the accuracy evolution in the simulated scenarios of noise 10% (see Figure 4.2a) the versions of IDRA and static CREA behave similarly before the first concept drift. From this point onwards, the versions of IDRA generally maintain their performance over time, while CREA degrades faster. IDRAv1 is affected by the last change while IDRAv2 is more affected by the second one and then it stabilises. RIDRA maintains its performance constant during the whole data stream analysis. In contrast, CREA has a downward trend over time and it slightly recovers when it reaches the fourth concept. The static method's drop of accuracy is very dependent on the θ used to build the initial set of rules, which explains the mild increase of accuracy for the last concept (from instance 44,400 onwards). In the case of VFDR, the precision changes quicker yet remains below all versions of IDRA accuracy levels most of the time. The accuracy evolutions of CREA and IDRA versions show a greater degradation of CREA performance when new observations arrive to the system. Thus, when the algorithm is exposed to data streams that are potentially infinite, the trend of the methods is a relevant element to consider. Therefore, the incremental quality proves its value, especially if we consider long-term maintenance and the consequences of this uncontrolled degradation in a system run by a static method.

In Figure 4.2b, the performance evolutions are shown for the 5% mislabelling scenario. In this case, we can see that CREA has a similar performance trend to IDRAv2. Again, the RIDRA version is the best one in this experiment, with a stable and maintained high accuracy rate over time. In the case of VFDR, a variable performance is observed during all the stream, stabilising its values in the last two concepts (from instance 29,400 onwards) around the results of CREA, IDRAv1 and IDRAv2.

In the case of Figure 4.3, the accuracy of the methods in the non-error data stream (Figure 4.3a) and the high impact changes scenario (Figure 4.3b) are shown. In the non-error scenario, all the versions of IDRA have similar performance evolution, maintaining their values around the 93% of accuracy (0.93), while CREA tends to decrease the quality of its predictions along the stream. In the case of VFDR, the accuracy level reaches the same levels obtained by IDRA versions at the end of the first and second concepts (around the instances 14,400; 29,400 and 44,400) but after the drift the performance falls again to recover again by the end of the same concept. The results of VFDR are always below the accuracy rates of all the versions of IDRA in this specific case. On the contrary, in the high impact drift case (Figure 4.3b), the VFDR method is able to improve the accuracy obtained by RIDRA at some points, after the reactive version of IDRA experiences a drop after the first concept drift. We can observe falls in all-algorithm performances after each change point (dashed lines). The magnitude of these drops varies a lot among algorithms, being again RIDRA the less affected one. This method again provides a pretty constant accuracy over time even if the impact of the changes in the data stream studied are great.

4.4.2. Real scenarios results

Now that the results of the simulated data streams have been discussed, the outcome of the real scenarios is commented on in this subsection. Table 4.8 contains the mean accuracy rates obtained by the algorithms while Table 4.9 shows the times in seconds. As already mentioned, two real datasets are used as streams of data to evaluate all the algorithms. The bank marketing dataset is publicly available and compiles examples of telemarketing actions of a bank. The car workshop set is a private dataset used to expose the algorithms to a specific real case whose conclusions could be contrasted with experts in the industry. In the case of the bank marketing case, the mean accuracy rates shown in Table 4.10 and Table 4.11 are the mean results of five experiments obtained by shuffling the original not ordered dataset. The car workshop results shown in these tables, represent the outcome of a single experiment, since the data have a meaningful order, the result tables.

In the case of the bank dataset, the results of CREA and IDRA versions are quite similar due to the lack of concept drifts that affect the model. In the case of the mean accuracy obtained by VFDR is almost 5 points lower than the rest of the methods, however the time execution is the fastest of all with a big difference. CREA, IDRAv1 and IDRAv2 have similar times (around 2 seconds of difference between the static

approach and the two first versions of IDRA), while RIDRA increases the time consumption to 20 seconds (9 seconds of difference with CREA and 7 seconds with IDRAv1 and IDRAv2).

The accuracy evolution of the algorithms in one of the shuffled datasets of the bank marketing scenario is represented in Figure 4.4a. The algorithm progress of all the IDRA versions and the static method are quasi-identical in the whole stream. The fact that IDRA versions perform equally to CREA in data streams with no change, and that it can adapt in those that do contain changes, shows the improvements achieved by these new incremental approaches. In contrast, VFDR shows some drops in the accuracy distribution after almost 30,000 seen instances. At this point, the evolution of the state-of-the-art method shows a more variable and inconsistent behaviour while also a slight decrease in the effectiveness of the IDRA versions and CREA model can be perceived.

Moving now to the car workshop scenario, we can see that the static method CREA and the dynamic ones (IDRA versions and VFDR) have a considerable difference in their performances. In this case, the analysed instances are significantly more than the bank telemarketing case with 260,000 instances against a bit more than 45,000. This rise in the number of the evaluated examples, as well as the growth of values of the studied variables have an impact on the time spent by all algorithms in this last real scenario. IDRAv1 and IDRAv2 have similar results in time, while RIDRA version significantly increases the time spent analysing the data. In this case, again, the very fast technique, VFDR, is the fastest option.

The accuracy evolution of the algorithms in this scenario is shown Figure 4.4b. Again, the evolution of VFDR is fast and variable, while the rest of methods stay more stable. All the versions of IDRA perform similarly during all the data stream analysis, while CREA have a similar tendency but with lower accuracy rates. Around the instance 190,000, a fall in accuracy happens in all algorithms, especially in the case of VFDR. This might be due to a change in the data or the context. The meaning of this drop is interpreted in the conclusions of the work, in Chapter 5.

Chapter 5
Conclusions and future research

This chapter presents the conclusions that arise from the experimental study carried out in Chapter 4. The suitability and limitations of the proposal are exposed, as well as the achievement of the objectives set in Chapter 1 and the contributions made from the practical and the theoretical point of view. Additionally, the future research that can follow this work is presented in two different directions. Either towards the implementation of optimization data techniques that increase the data stream quality and therefore the time or performance of the proposed algorithm, or the study of the interaction of an expert with the decision system and its improvement.

The content of this chapter is based on the paper published in the journal Mathematics under the title *Incremental Decision Rules Algorithm: A Probabilistic and Dynamic Approach to Decisional Data Stream Problems* (Mollá et al. 2022b). This publication presents the results and conclusions of the initial version of the proposed method, IDRAv1. The conclusions regarding the new versions of the algorithm are presented in this chapter for the first time.

5.1. Suitability and limitations of IDRA

The presented method IDRA, in its three versions, has been tested in different scenarios and conditions to measure its suitability and effectiveness in many situations. All the versions of the novel technique improve the static method (CREA), in most common scenarios and in high impact ones. Depending on the level and the type of error that the system is exposed to, a version of the algorithm would be preferable to others. The variety of approaches could require some time from the decision maker to define the best fitting one, but a more flexible and specific solution could be provided for those cases. A method selection based on the conditions and specification of the industry and the specific case in which it is applied, even if a time investment is required in the initial stages, could lead to a much better performing system. Defining the time requirements of the systems, the needed accuracy rates, the expected change and the informative level, the best approach could be determined.

The presented technique is especially valuable for decisional contexts where the actions are led or supported by a system using a previous learned base of knowledge. This learning process is the key element used to estimate the class values of the studied variable. Providing an incremental learning approach, the static method is improved considerably for the data stream contexts that contain changes. The use of the decision rules in the system helps the readability of this knowledge and makes IDRA especially suitable to fit a DSS. These decision rules are built in an exhaustive manner, so the decision maker can have a broad idea of what type of individual the rule is describing. This approach might enlarge the rule set, since more variables need to be taken into account. Nevertheless, this contribution has proven to improve the accuracy of the method and, we consider it adds value in the decision-making contexts. Especially in those cases, a high informative level of the rules might by itself be preferable, since it helps to understand the situations we are facing.

Table 5.1. Qualitative comparison of all algorithms based on its main characteristics for changing data streams in the simulated case of 10% noise scenario.

Algorithm	Time requirement (instances/s)	Accuracy rates	Informative level	Adaptable to changes
CREA	Between 2.5 and 5	Between 85% and 90%	Higher	No method
IDRAv1	Less than 2.5	Between 85% and 90%	Higher	Implicit method
IDRAv2	Between 2.5 and 5	More than 90%	Higher	Explicit method
RIDRA	Less than 2.5	More than 90%	Higher	Explicit method
VFDR	More than 5	Less than 85%	Lower	Explicit method

Table 5.1 shows the relative comparison between the different methods based on the described characteristic in the simulated 10% noise scenario. This table tries to identify which approach could fit better considering some specific conditions of the context. For example, in the studied real case of the car workshop, with non-extremely demanding times, and considering that changes are not present in the system on a daily basis, any version of IDRA could be selected. Considering that higher accuracy rates

are always preferable, the reactive version would be the most attractive one, since it is the option that provides better performing models. In the case of the adaptability to changes, the explicit method is supposed to be way quicker than the implicit method and the absence of it. The lack of change adaptation is not especially desirable when a changing data stream context is studied due to the proven degradation of the system. Also, we consider that, for decisional contexts, a more informative system would be preferred, so the use of IDRA versions is justified over the VFDR option.

In the studied case of the bank tele-marketing campaigns, since no changes are occurring in the data, the accuracy rates are similar for the static version (CREA) and the incremental versions (IDRA). Nevertheless, a decision maker might rather have a slightly slower response (in this case, 2s of difference for almost 45,000 instances) if the system is able to adapt to changes that might rise in the data stream. In this case, IDRAv1 or IDRAv2 may be chosen over the static approach.

Figure 5.1 presents a flow diagram for the criteria that might be to select a technique based on characteristics such as the time requirements, the control over the system, the accuracy rates needed or the nature of the studied variable. In this case, the diagram shows that for systems with constant expert control, a static traditional algorithm might be enough. The algorithm CREA is a more than fair solution for those cases as proved by Rodriguez-Sala (2014). For those systems that prefer less expert control, always more expensive to provide, the time requirements are considered. The very fast arrival data stream will need a very fast approach like the one yielded by VFDR. If the arrival times are not that critical, as it could be the case of the decisional contexts, the expected impact of the changes is evaluated. In this case, for those high impact changes scenarios, RIDRA has proven to be the preferred method. Then, the flow diagram evaluates, if a more informative rule structure is valuable so some of the IDRA versions would be preferred. If the system does not value those types of rules, and the time prevails over the accuracy, VFDR is again a good option to run in our system. In the case that the accuracy prevails over time, again a IDRA method will be selected over the very fast technique. Finally, if the studied variable has more than two possible classes, the system would need to run with one of the two first versions of IDRA, IDRAv1 or IDRAv2. In this case, IDRAv2 is preferable since it is an extended and improved version of IDRAv1. On the contrary, if the data stream objective variable has only two classes, RIDRA would be preferred since the accuracy rates obtained by this version have proved to be better than the ones by IDRAv1 and IDRAv2.

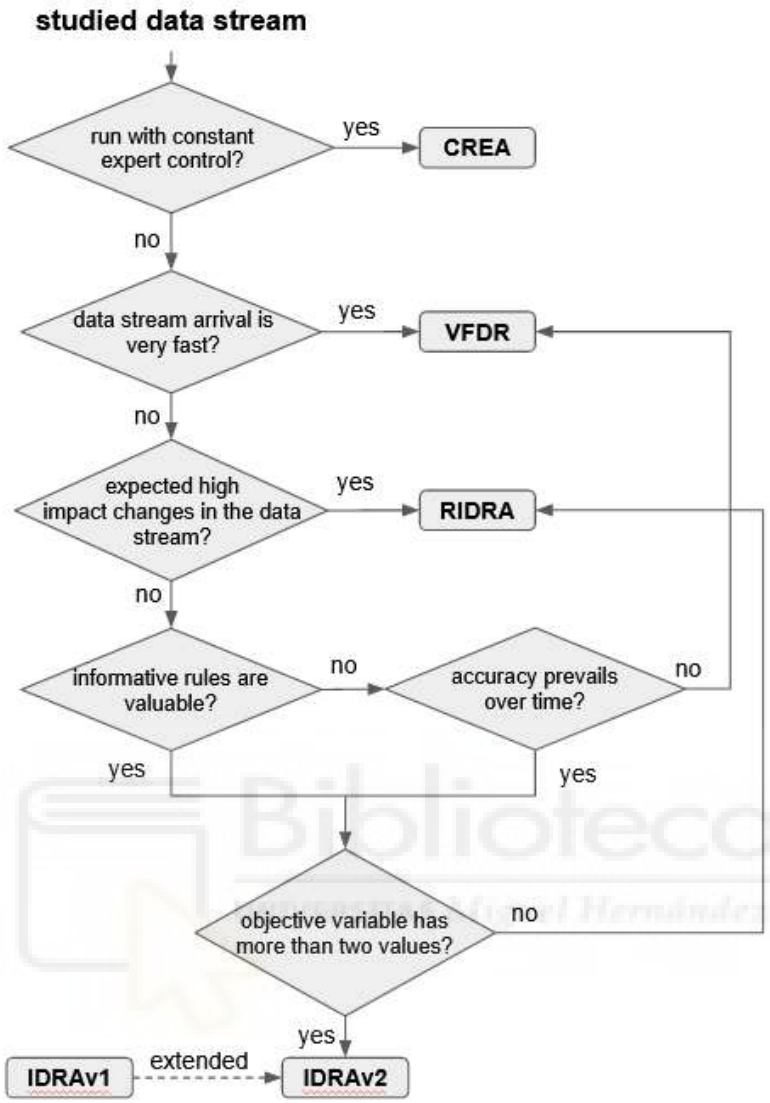


Figure 5.1 Flow diagram of the method selection based on basic premises (own source).

For the data stream scenarios, the number of instances is considered to be potentially infinite. Based on the conclusions of Rodriguez-Sala (2014) regarding the static antecedent in which the IDRA proposal is based on, we consider that the most important factor when referring to the analysis times is the number of instances. In this work, Rodriguez-Sala studies the evolution of the times based on parameters such as the number of columns, the total number of values of these columns (cardinality of the attributes) and the size of the dataset. In the referred work, CREA proves to have a complexity divided in sections, obtaining linear times in the first and last stages, and exponential ones in the intermediate section. This static approach, as already mentioned, considers the number of columns and their cardinality, as well as the number of analysed instances to estimate the needed time. Since the available columns and their values are expected to be constant while the streamed instances tend to be infinite, only the number of examples received by the system will be considered

relevant to determine the time needed by the system. Also, based on the three stages presented by Rodriguez-Sala, we assume that the third one (the last one) is the most relevant one. This is easily understood by the fact that the data stream has not-known end so once this stage is reached, that same formula will be used to estimate the computational costs and the times of the model.

Regarding the restraints of the proposed algorithm, in all its versions, the nature of the analysed data streams could be a limitation, since only discrete data can be processed by IDRA, as well as by its antecedent CREA. Analysed data streams might require a previous discretization process to make them meet this requirement in the case of the continuous variables. IDRA is a supervised algorithm, so it requires a target variable, also known as objective variable, to predict. This variable needs to be also nominal or discrete. This discretization process is normally done by manually defining the intervals of the continuous variable under expert criteria and assigning the value before the analysis for the explicative variables or the objective variable. If there are no expert criteria to be applied to the discretization process, the process will be based on the value distribution of each continuous attribute in a given moment in time. If the value distribution of any explicative variable changes over time, IDRA will not be able to detect this change in any of its versions. If that change of distribution happens in the objective variable, the Reactive version of IDRA, RIDRA, will detect it using the ADWIN method implemented on it. The second version of IDRA, IDRAv2, will react to this change after the expected impact on the accuracy of the model. Including external analysis of the stream distributions of the explicative variables and the objective one, could solve this situation and provide improvements in the analytic problems.

One of the limitations of the novel method presented in this work, specifically in its third version, RIDRA, is, as we could see in the Figure 5.1, the requirement of the objective variable to be bounded to two classes. This approach uses a third-party method, ADWIN, implemented for Python in the package Scikit-multiflow (Montiel et al. 2018). In this case, this implementation uses only continuous variables to study and detect changes in the objective variable. Since the IDRA algorithm analyses only discrete classes, only a conversion between a two-value class variable and a continuous variable is possible. Thus, the negative discrete class could be replaced by a 0 and the positive class by a 1. If more than two classes are studied, this conversion could not take place in a meaningful way and, therefore, the detection method of ADWIN, a basic part of the RIDRA approach, could not be used. In these cases, the other versions could replace the reactive method to preserve the decisional optimised characteristics that IDRA provides to the system.

5.2. Achievement of objectives

The novel incremental approach presented in this work is considered a successful new technique especially relevant in decisional contexts. IDRA has proven, with a wide experimental study, an improvement in accuracy respecting the static traditional methodology and the existing adaptive methods. IDRA achieved the objective of endowing the previous static method (CREA) with an incremental logic that consequently improved its performance for data streams that might potentially change at some point. IDRA, in all its versions, has demonstrated to improve the performance of the static approach in most common cases and when a high impact change happens in the stream. Thus, IDRAv2 and RIDRA achieved to automate an explicit concept drift detection method and adapt the model to the new data distribution by rebuilding the base of knowledge. Also, IDRA successfully integrates a new structure for decision rules that provide broad information of the class probability distributions. Thus, the decision maker could understand better the context and implications of their actions before a decision is made. External conditions, such as the risk, the cost or the trade-off of a plan, can be taken into account.

Apart from adapting the learning static method and restructuring the rules to a soft structure based on probabilities, IDRA also proposes a new way of post-pruning the set of rules. The new structure of IDRA rules and the incremental logic of the method make it necessary to adapt the RBS method proposed by Almiñana et al. (2012). This post-pruning technique simplifies the rule sets and reduces the search times establishing a threshold for the selection of decision rules. In this sense, the modified RBS is integrated successfully in the first two versions of IDRA, IDRAv1 and IDRAv2. The pruning criteria prove to have no impact on the quality of the built models since the performance of IDRAv1 and IDRAv2 are better or comparable with the static version.

Now, comparing the presented method with the previous existing ones, we can say that IDRA provides a new high accuracy option in exchange for a higher time consumption. This option, as already mentioned, is valuable for not so fast contexts that might be exposed to changes or that simply want to run with no expert supervision in an uncertain scenario. For those cases, IDRA provides a time-assumable highly accurate solution with a DSS integrable structure. Also, the characteristics compiled by this approach again endow it with an added value in the case of decisional contexts. In this sense, as it was already exposed in this chapter, the new technique uses exhaustive decision rules to deeply describe the target case that is under study. Thus, the decision maker can understand the exact characteristics that define each case and act based on this knowledge. With this new approach and all the computational experimentation carried on in this work, IDRA has successfully proved to be an optimised solution for these kinds of scenarios, alternatively to the very fast adaptive algorithms. The novel approach did not aim to challenge the very fast solutions present in the state-of-the-art

but propose a new branch of algorithms that fit into highly explicative decisional contexts with no so demanding times.

The suitability and value of this new methodology is justified in this work through an exhaustive experimental design that illustrates the advantages that this new approach apports. In this sense, specifically in the real data scenario of the car workshop, IDRA versions prove to be less affected (see Figure 4.4b real car scenario) by a substantial change that takes place during the stream. An analysis of the data used as a stream, we could detect that this change point belongs to March 2020. This date coincides with the start of the pandemic of COVID-19 that affected all industries around the world. This specific fact poses as a great proof of the high value that the incremental learning process could apport to a system. In this sense, IDRA generates solutions with less variability in the accuracy maximum and minimum levels when compared to the state-of-the-art method VFDR. This could be understood as a measure of the volatility of the methods and could show their reliability compared to its mean accuracy.

The different characteristics of the study presented in this dissertation are summarised in Table 5.2. This table shows the main aspects of the study, specifically the problem to tackle, the main objective of the work, the hypothesis assumed during the process and the final conclusions.

Table 5.2: Summary of the main characteristics of the study.

Problem	Concept drift produces the degradation of models when analysing data streams.
Objective	Maintain or improve the levels of accuracy of the models in data stream contexts.
Hypothesis	Incremental models will provide better classification results for data stream contexts.
Conclusions	The hypothesis is confirmed, at least for the studied scenarios, by using the proposed incremental algorithm.

5.3. Contributions

This work has contributed to the state of the art in different ways. Besides the practical contribution of proposing a new method that provides a high accurate decisional solution for data streams, this thesis also extends the knowledge regarding the adaptive philosophy in decision making processes in organisational settings. In this sense, there is a special value in the combination between the incremental or adaptive logic in data stream contexts with the data-driven solutions for information systems. This approach is presented in Mollá et al. (2022a) and extended in this dissertation. Specifically in the field of DSS, when a system is exposed to a potentially infinite flow of data that might change anytime, if this DSS is not adapted or prepared to detect these changes, its recommendations could be misleading or mistaken, and therefore carry a high cost for the decision makers. Endowing the DSS with an adaptive engine that automatically

integrates the changes produced in the data stream grants an updated base of knowledge that increases both the quality of the DSS and the trust of the user in the system.

When we propose, design and develop IDRA, we aim to provide the incremental engine that fits in that adaptive DSS within a business decision tool. To fulfil this purpose, the knowledge modelled by IDRA uses an explainable structure like the decision rules that could be easily integrated in a DSS and a BI system. IDRA, by design, allows the rule set to grow with any new instance received, nevertheless, as postulated by Trépos et al. (2013) and discussed in section 2.2, we should try to guarantee the actionability of the system. This means that attributes such as the size or the complexity of a set of rules should not affect the response capacity of a system. To assure the actionability, IDRA is provided with a filter method based on an adaptation of the work by Almiñana et al. (2012). This filter reduces the set of rules to those that have a certain level of significance, assuring with that the number of rules is affordable and the included knowledge is relevant for a DSS environment. Ensuring these optimised structures and with the results presented in Chapter 4, we can conclude that the contributions of IDRA are promising in the field of highly accurate solution for decisional contexts. From the industrial point of view, the integration of this algorithm in a BI solution could significantly enrich its results.

5.4. Further research

The research possibilities and the importance of this field has been proven along all this research. In this sense, the future research linked to this novel line of investigation could be led in many different directions that are presented in this subsection. Both, improvements on the presented algorithm using optimization techniques, and the study of further steps in the integration of the new method in an information system are considered.

Regarding the potential improvement that could be made upon the algorithm presented in this work, we consider the possibility of proposing a wider study of the parametrization values and environmental modifications that could lead to an improvement of the performance of IDRA in all its parametric versions. A deep analysis of criteria such as the metric used to order and select rules in the first version of IDRA (especially the new proposed metric of trending), their relevance to the evaluation window size or the study of non-so-demanding values for the entropy threshold. Also, further efforts could focus on implementing different types of memory access to study the optimization of the algorithm and proposing a method for the post-pruning of the rule sets of the reactive version, RIDRA.

Respecting the possibility of improving the quality of IDRA, it could also be done by the optimization of the analysed data streams that will be read by the system. In this sense, an online pre-processing method that automates the optimization of the analysed

data streams is a wide and complex new field that could lead to major benefits. As already explained, IDRA algorithm, in all its versions, works with discrete values. In case these values come from a continuous variable, the system will need to convert them into categories in order to use the algorithm. This process is not especially expensive in terms of time and complexity, so it would be assumable by the big majority of systems. Nevertheless, as in all the data stream contexts, the potential infinitude of the stream could lead to changes that affect the effectiveness of the manually and previously defined boundaries. Posing a new method that processes the data stream online, updating this discretization criteria when the data distribution changes would improve the independence of the algorithms, reducing the expert intervention. This fact could reduce times and, therefore costs, by proposing new ranges when needed, which an expert will accept or modify, with no need to study the whole problem. It is important that these techniques preserve a supportive logic in which an expert takes the final decision, so a minimum human intervention is assured. Thus, the modifications and decisions recommended by those systems are checked before taking place so biased or unfair outcomes could be detected and avoided. Other pre-processing techniques such as an online selection of the most relevant variables for classification or the outlier and null values detection and estimation could also be presented as a relevant contribution in this promising field of study.

Once the algorithm is proposed and tested with different data stream conditions, it needs to be integrated in an information system before being used in a real scenario. The incremental approach proposed in this work is especially designed to be integrated in a DSS and, then in a BI tool. Therefore, the next steps will be to integrate the novel methodology in a DSS that allows the decision maker to interact with the base of knowledge and study this interaction to optimise, at the same time, the framework that contains it. This research is considered relevant, especially from the information system and the BI perspectives.

Capítulo 5
Conclusiones y trabajos futuros

Este capítulo presenta las conclusiones que derivan del estudio experimental llevado a cabo en el Capítulo 4. Se exponen la adecuación y limitaciones de la propuesta, así como la realización de os objetivos establecidos en el Capítulo 1. Además, se presentan los trabajos futuros que se pueden plantear en dos posible direcciones. Esta investigación puede ir dirigidas tanto hacia la implementación de técnicas de optimización de los datos que incrementen la calidad de los flujos de datos y, por tanto, el tiempo o desempeño del algoritmo propuesto, como hacia el estudio de la interacción de una persona experta con el sistema decisional y su mejora.

El contenido de este capítulo se basa en el artículo publicado en la revista Mathematics con el título Incremental Decision Rules Algorithm: A Probabilistic and Dynamic Approach to Decisional Data Stream Problems (Mollá et al. 2022b). Esta publicación presenta los resultados y conclusiones de la versión inicial del método propuesto, IDRAv1. Las conclusiones sobre las nuevas versiones del algoritmo se presentan por primera vez en este capítulo.

5.1 Adecuación y limitaciones de IDRA

El método IDRA, en sus tres versiones, ha sido probado en diferentes escenarios y condiciones para medir su idoneidad y eficacia en multitud de situaciones. Todas las versiones de esta novedosa técnica mejoran el método estático (CREA), en los escenarios más comunes y en los de alto impacto. Según el nivel y el tipo de error al que esté expuesto el sistema, una versión del algoritmo será preferible a otras. La variedad de enfoques podría requerir algo de tiempo por parte del decisor a la hora de definir cuál es la versión más adecuada, pero se podría proporcionar una solución más flexible y específica para esos casos. Una selección de métodos basada en las condiciones y especificaciones del campo de aplicación y el caso específico en el que se aplica, podría conducir a una mejora considerable de sistema. Definiendo los requisitos de tiempo, las tasas de precisión necesarias, el cambio esperado y el nivel de información deseado, se puede determinar el método más adecuado.

La técnica presentada es especialmente valiosa para contextos decisionales donde las acciones son dirigidas o respaldadas por un sistema que utiliza una base de conocimiento previamente aprendida. Este proceso de aprendizaje es el elemento clave utilizado para estimar los valores de clase de la variable objetivo. A través de un enfoque de aprendizaje incremental, el método estático consigue ser mejorado considerablemente para los contextos de flujo de datos que contienen cambios. El uso de las reglas de decisión ayuda a la legibilidad de este conocimiento y hace que IDRA sea especialmente adecuado para adaptarse a un sistema de apoyo a la decisiones (DSS). Estas reglas de decisión se construyen de manera exhaustiva, por lo que la persona encargada de la toma de decisiones puede tener una idea amplia de qué tipo de individuo o caso describe la regla. Este enfoque podría ampliar el conjunto de reglas, ya que es necesario tener en cuenta más variables. Sin embargo, esta contribución ha demostrado mejorar la precisión del método y se considera que agrega valor en los contextos de toma de decisiones. Especialmente en esos casos, un alto nivel informativo de las reglas, por sí mismo, puede ser preferible, ya que ayuda a comprender en profundidad las situaciones a las que nos enfrentamos.

Tabla 5.1. Comparación cualitativa de todos los algoritmos basada en sus principales características para flujos de datos cambiantes en el caso de escenarios con un 10% de error.

Algoritmo	Requisito tiempo (instancias/s)	Ratio de precisión	Nivel informativo	Adaptable a cambios
CREA	Entre 2.5 y 5	Entre 85% y 90%	Alto	Sin método
IDRAv1	Menos de 2.5	Entre 85% y 90%	Alto	Método implícito
IDRAv2	Entre 2.5 y 5	Más de 90%	Alto	Método explícito
RIDRA	Menos de 2.5	Más de 90%	Alto	Método explícito
VFDR	Más de 5	Menos de 85%	Bajo	Método explícito

La Tabla 5.1 muestra la comparación relativa entre los diferentes métodos en el escenario simulado de 10% de ruido. Esta tabla trata de identificar qué enfoque podría

encajar mejor considerando algunas condiciones específicas del contexto. Por ejemplo, en el caso real estudiado del taller de automóviles, con tiempos no extremadamente exigentes, y considerando que no se presentan cambios en el sistema a diario, se podría seleccionar cualquier versión de IDRA. Teniendo en cuenta que siempre son preferibles las tasas de precisión más altas, la versión reactiva sería la más atractiva, ya que es la opción que proporciona modelos con mejor rendimiento. En el caso de la adaptabilidad a los cambios, se supone que el método explícito es mucho más rápido que el método implícito y la ausencia del mismo. La falta de adaptación al cambio no es deseable cuando se estudia un contexto de flujo de datos con variaciones debido a la degradación del sistema comprobada en los experimentos realizados. Además, consideramos que, para contextos decisionales, es preferible un sistema más informativo, por lo que se justifica el uso de las versiones IDRA frente a la opción de VFDR.

En el caso estudiado de las campañas de telemarketing bancario, dado que no se están produciendo cambios en los datos, las tasas de precisión son similares para la versión estática (CREA) y las versiones incrementales (IDRA). Sin embargo, una persona encargada de la toma de decisiones podría tener una respuesta un poco más lenta (en este caso, 2 segundos de diferencia para casi 45 000 instancias) si el sistema es capaz de adaptarse automáticamente a los cambios que pueden surgir en el flujo de datos. En este caso, se puede elegir IDRAv1 o IDRAv2 en lugar del enfoque estático.

La Figura 5.1 presenta un diagrama de flujo de los criterios que podrían ser utilizados para seleccionar una técnica en función de características tales como los requisitos de tiempo, el control sobre el sistema, las tasas de precisión necesarias o la naturaleza de la variable estudiada. En este caso, el diagrama muestra que para sistemas con una persona experta controlando de manera constante, un algoritmo tradicional estático podría ser suficiente. El algoritmo CREA es una solución más que aceptable para esos casos, tal como demuestra Rodríguez-Sala (2014). Para aquellos sistemas que prefieren menos control experto, que siempre es más costoso de proporcionar, se consideran los requisitos de tiempo. El flujo de datos de llegada muy rápida necesitará un enfoque muy rápido como el producido por VFDR. Si los tiempos de llegada no son tan críticos, como podría ser el caso de los contextos decisionales, se evalúa el impacto esperado de los cambios. En este caso, para esos escenarios de cambios de alto impacto, RIDRA ha demostrado ser el método preferido para los casos estudiados. Luego, el diagrama de flujo evalúa si una estructura de reglas más informativa es valiosa, por lo que se preferirían algunas de las versiones de IDRA. Si el sistema no valora este tipo de reglas, y el tiempo prima sobre la precisión, VFDR vuelve a ser una buena opción para ejecutar en nuestro sistema. En el caso de que la precisión prevalezca al tiempo, nuevamente se seleccionará un método IDRA sobre la técnica muy rápida del estado del arte. Finalmente, si la variable estudiada tiene más de dos clases posibles, el sistema necesitaría modelarse con una de las dos primeras versiones de IDRA, IDRAv1 o IDRAv2. En este caso, es preferible IDRAv2 ya que es una versión extendida y mejorada de IDRAv1. Por el contrario, si la variable objetivo del flujo de datos tiene solo dos clases, se preferirá RIDRA ya que las tasas de precisión obtenidas por esta versión han demostrado ser mejores que las de IDRAv1 e IDRAv2.

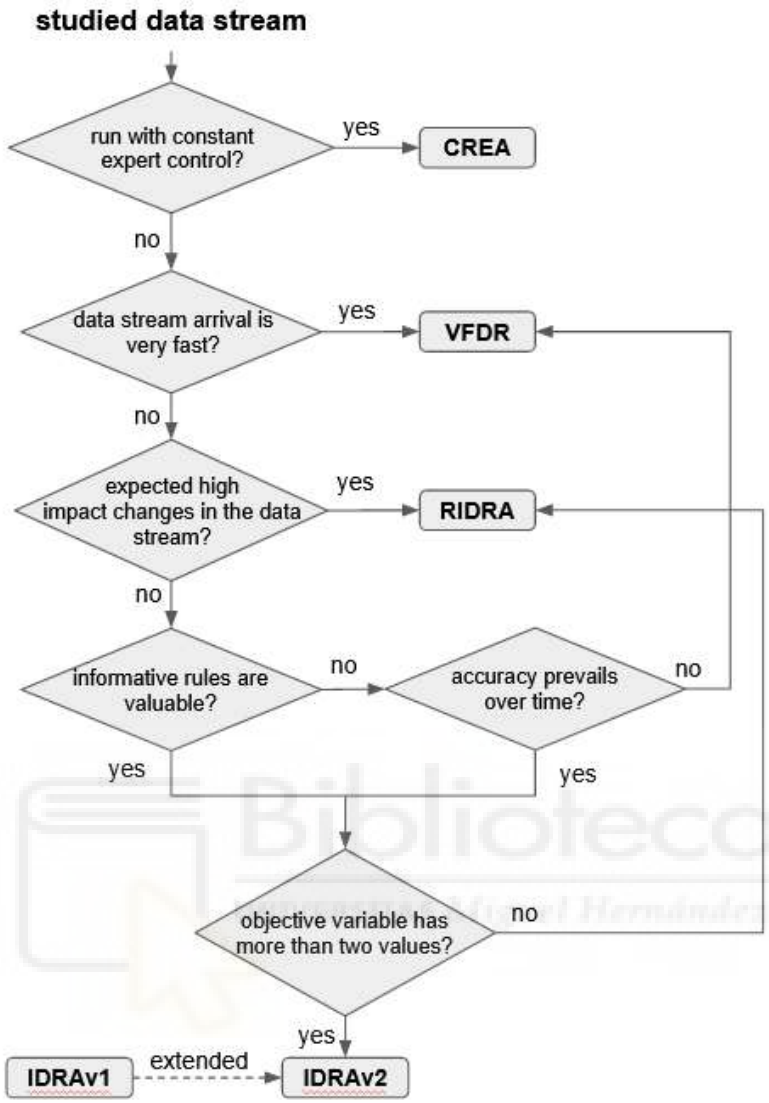


Figura 5.1 Diagrama de flujo de la selección de método basada en premisas básicas (elaboración propia).

Para los escenarios data stream, el número de instancias a analizar se considera potencialmente infinito. Basándonos en las conclusiones de Rodríguez-Sala (2014) sobre el antecedente invariable en el que se basa la propuesta de IDRA, consideramos que el factor más importante al referirnos a los tiempos de análisis es el número de instancias. En este trabajo, Rodríguez-Sala estudia la evolución de los tiempos en función de parámetros como el número de columnas, el número total de valores de estas columnas (cardinalidad de los atributos) y el tamaño del conjunto de datos. En el referido trabajo, CREA demuestra tener una complejidad dividida en tramos, obteniendo tiempos lineales en las etapas primera y última, y exponenciales en el tramo intermedio. Este enfoque no variable, como ya se mencionó, considera el número de columnas y su cardinalidad, así como el número de instancias analizadas para estimar el tiempo necesario. Dado que se espera que las columnas disponibles y sus valores sean constantes, mientras que las instancias transmitidas tienden a ser infinitas, solo la

cantidad de ejemplos recibidos por el sistema se considerará relevante para determinar el tiempo que necesita el sistema. Asimismo, en base a las tres etapas presentadas por Rodríguez-Sala, suponemos que la tercera (la última) es la más relevante. Esto se entiende fácilmente por el hecho de que el flujo de datos no tiene un final conocido, por lo que una vez que se alcanza esta etapa, se utilizará esa misma fórmula para estimar los costos computacionales y los tiempos del modelo.

En cuanto a las restricciones del algoritmo propuesto, en todas sus versiones, la naturaleza de los flujos de datos analizados podría ser una limitación, ya que solo los datos discretos pueden ser procesados por IDRA, así como por su antecedente CREA. Los flujos de datos analizados pueden requerir un proceso previo de discretización para que cumplan con este requisito en el caso de las variables continuas. IDRA es un algoritmo supervisado, por lo que requiere una variable objetivo para predecir. Esta variable debe ser también nominal o discreta. Este proceso de discretización se realiza normalmente definiendo manualmente los intervalos de la variable continua bajo criterio de expertos y asignando el valor antes del análisis para las variables explicativas o la variable objetivo. Si no existen criterios de expertos que aplicar al proceso de discretización, este se basará en la distribución de valores de cada atributo continuo en un momento dado. Si la distribución de valores de cualquier variable explicativa cambia con el tiempo, IDRA no podrá detectar este cambio en ninguna de sus versiones. Si ese cambio de distribución ocurre en la variable objetivo, la versión Reactiva de IDRA, RIDRA, lo detectará mediante el método ADWIN implementado sobre ella. La segunda versión de IDRA, IDRAv2, reaccionará a este cambio después del impacto esperado en la precisión del modelo. La inclusión de análisis complementarios de las distribuciones de las variables explicativas y objetivas podría solucionar esta situación y aportar mejoras en los problemas analíticos.

Una de las limitaciones del método presentado en este trabajo, específicamente en su tercera versión, RIDRA, es, como pudimos ver en la Figura 5.1, el requisito de que la variable objetivo esté acotada a dos clases. Este enfoque utiliza un método de terceros, ADWIN, implementado para Python en el paquete Scikit-multiflow (Montiel et al. 2018). En este caso, esta implementación utiliza únicamente variables continuas para estudiar y detectar cambios en la variable objetivo. Dado que el algoritmo IDRA analiza solo clases discretas, solo es posible una conversión entre una variable de clase de dos valores y una variable continua. Así, la clase discreta negativa podría ser sustituida por un 0 y la clase positiva por un 1. Si se estudian más de dos clases, esta conversión no podría realizarse de forma significativa y, por tanto, el método de detección de ADWIN, básico para la versión RIDRA, no se puede utilizar. En estos casos, las otras versiones podrían reemplazar el método reactivo para preservar las características decisionales optimizadas que IDRA aporta al sistema.

5.2 Consecución de los objectivos

El algoritmo incremental presentado en este trabajo se considera una nueva técnica exitosa especialmente relevante en contextos decisionales con flujos de datos cambiantes. IDRA ha demostrado, con un amplio estudio experimental, una mejora en la precisión respecto a la metodología tradicional estática y los métodos adaptativos existentes. IDRA logró el objetivo de dotar al método estático anterior (CREA) de una lógica incremental que, como consecuencia, mejoró su rendimiento para flujos de datos sujetos a cambios potenciales a lo largo del tiempo. IDRA, en todas sus versiones, ha demostrado mejorar el rendimiento del enfoque estático en los casos más comunes y cuando ocurre un cambio de alto impacto en los datos. Por lo tanto, podemos decir que IDRAv2 y RIDRA han logrado automatizar un método explícito de detección de desviación de concepto y adaptar el modelo a la nueva distribución de datos mediante la reconstrucción de la base de conocimiento. Además, IDRA integra con éxito una nueva estructura de reglas de decisión que proporciona amplia información de las distribuciones de probabilidad de las posibles clases. Así, la persona encargada de la toma de decisiones podría comprender mejor el contexto y las implicaciones de sus acciones antes de decidir. De esta manera, se pueden tener en cuenta condiciones externas como el riesgo, el coste o la compensación de un plan antes de emprenderlo.

Además de adaptar el método estático de aprendizaje y reformular las reglas hacia una estructura suave basada en probabilidades, IDRA también propone una nueva forma de post-poda del conjunto de reglas. La nueva estructura de las reglas IDRA y la lógica incremental del algoritmo hacen necesario adaptar el método RBS propuesto por Almiñana et al. (2012). Esta técnica de post-poda simplifica los conjuntos de reglas y reduce los tiempos de búsqueda estableciendo un umbral para la selección de reglas de decisión. En este sentido, el RBS modificado se integra con éxito en las dos primeras versiones de IDRA, IDRAv1 e IDRAv2. Los criterios de poda demuestran no tener impacto en la calidad de los modelos construidos ya que el rendimiento de IDRAv1 e IDRAv2 es mejor o comparable con la versión estática.

El novedoso enfoque incremental presentado en este trabajo se considera una nueva técnica exitosa especialmente relevante en contextos decisionales. IDRA ha demostrado, con un amplio estudio experimental, una mejora en la precisión respecto a la metodología tradicional estática y los métodos adaptativos existentes. IDRA logró el objetivo de dotar al método estático anterior (CREA) de una lógica incremental que, en consecuencia, mejoró su rendimiento para flujos de datos que podrían cambiar en algún momento. IDRA, en todas sus versiones, ha demostrado mejorar el rendimiento del enfoque estático en los casos más comunes y cuando ocurre un cambio de alto impacto en la corriente. Por lo tanto, IDRAv2 y RIDRA lograron automatizar un método explícito de detección de desviación de conceptos y adaptar el modelo a la nueva distribución de datos mediante la reconstrucción de la base de conocimiento. Además, IDRA integra con éxito una nueva estructura para reglas de decisión que

proporciona información amplia de las distribuciones de probabilidad de clase. Así, el tomador de decisiones podría comprender mejor el contexto y las implicaciones de sus acciones antes de tomar una decisión. Se pueden tener en cuenta las condiciones externas, como el riesgo, el costo o la compensación de un plan.

Comparando ahora el método presentado con los anteriores existentes, podemos decir que IDRA proporciona una nueva opción de alta precisión a cambio de un mayor consumo de tiempo. Esta posibilidad, como ya se mencionó, es valiosa para contextos no tan rápidos que pueden estar expuestos a cambios o que simplemente quieren funcionar sin supervisión experta en un escenario incierto. Para esos casos, IDRA proporciona una solución altamente precisa y asumible en el tiempo con una estructura integrable en un DSS. Asimismo, las características que reúne este enfoque vuelven a dotarlo de un valor añadido en el caso de contextos decisionales. En este sentido, como ya se expuso en este capítulo, la nueva técnica utiliza reglas de decisión exhaustivas para describir en profundidad el caso que se está prediciendo. Así, la persona encargada de la toma de decisiones puede comprender las características exactas que definen cada caso y actuar en base a este conocimiento. Con este nuevo enfoque y toda la experimentación computacional llevada a cabo en este trabajo, IDRA ha demostrado con éxito ser una solución optimizada para este tipo de escenarios y una alternativa más precisa a los algoritmos adaptativos muy rápidos. Esta nueva propuesta no pretende desafiar a los algoritmos de clasificación llamados "muy rápidos" presentes en el estado del arte, sino proponer una nueva rama de algoritmos que encajan en contextos decisionales altamente explicativos con tiempos no tan exigentes.

El algoritmo incremental presentado en este trabajo se considera una nueva técnica que ha probado funcionar de manera exitosa en los escenarios analizados y que resulta especialmente relevante en contextos decisionales. IDRA ha demostrado, con un amplio estudio experimental, una mejora en la precisión respecto a la metodología tradicional estática y a los métodos adaptativos existentes. IDRA ha logrado el objetivo de dotar al método estático anterior (CREA) de una lógica incremental que, en consecuencia, ha mejorado su rendimiento para flujos de datos expuestos a potenciales cambios en algún momento. IDRA, en todas sus versiones, ha demostrado mejorar el rendimiento del enfoque estático en los casos más comunes y cuando ocurre un cambio de alto impacto en la corriente. En estos casos, IDRAv2 y RIDRA han logrado automatizar un método explícito de detección de desviación de conceptos y adaptar el modelo a la nueva distribución de datos mediante la reconstrucción de la base de conocimiento. Además, IDRA integra con éxito una nueva estructura para reglas de decisión que proporciona información amplia de las distribuciones de probabilidad de clase. Así, la persona encargada de la toma de decisiones podría comprender mejor el contexto y las implicaciones de sus acciones antes de tomar una decisión. Se pueden tener en cuenta las condiciones externas, como el riesgo, el coste o los potenciales beneficios de un plan.

La idoneidad y valor de esta nueva metodología se justifica en este trabajo a través de un exhaustivo diseño experimental que ilustra las ventajas que aporta este nuevo enfoque. En este sentido, concretamente en el escenario de datos reales del taller de

automóviles, las versiones de IDRA se muestran menos afectadas (ver Figura 4.4b escenario real de un taller de automóvil) por un cambio sustancial producido a lo largo del tiempo. Un análisis de los datos utilizados concluye que este punto de cambio corresponde a marzo de 2020. Esta fecha coincide con el inicio de la pandemia de COVID-19 que afectó a todas las industrias alrededor del mundo. Este hecho específico se presenta como una gran prueba del alto valor que el proceso de aprendizaje incremental podría aportar a un sistema. En este sentido, IDRA genera soluciones con menor variabilidad en los niveles máximos y mínimos de la precisión en comparación con el método de referencia, VFDR. Esto podría entenderse como una medida de la volatilidad de los métodos y podría mostrar la fiabilidad de IDRA en comparación con su precisión media.

Las diferentes características del estudio presentado en esta tesis se resumen en la Tabla 5.2. En esta tabla se muestran los principales aspectos del estudio, concretamente el problema a abordar, el objetivo principal del trabajo, la hipótesis asumida durante el proceso y las conclusiones finales.

Tabla 5.2: Resumen de las principales características del studio.

Problema	El cambio en los datos produce una degradación en los modelos cuando se analizan flujos de datos continuos.
Objectivo	Mantener o mejorar los niveles de precisión de los modelos para entornos data stream.
Hipótesis	Los modelos incrementales mejoran los resultados de clasificación para contextos data stream.
Conclusiones	La hipótesis se confirma, al menos para los escenarios estudiados, usando el algoritmo incremental propuesto.

5.3 Contribuciones

Este trabajo contribuye al estado del arte de diferentes maneras. Además de la contribución práctica de proponer un nuevo método que proporciona una solución decisional de alta precisión para flujos de datos; esta tesis también amplía el conocimiento sobre la filosofía adaptativa en los procesos de toma de decisiones en entornos organizacionales. En este sentido, hay un valor especial en la combinación entre las lógicas incrementales o adaptativas en contextos de flujo de datos con las soluciones basadas en datos o data-driven para sistemas de información. Este enfoque se presenta en Mollá et al. (2022a) y se amplía en este trabajo. Específicamente en el campo de los DSS, cuando un sistema está expuesto a un flujo de datos potencialmente infinito que puede cambiar en cualquier momento, si este DSS no está adaptado o preparado para detectar estos cambios, sus recomendaciones pueden ser engañosas o erróneas, y por lo tanto suponer un alto coste para la toma de decisiones. Dotar al DSS de un motor adaptativo que integre automáticamente los cambios producidos en el flujo de datos otorga una base de conocimiento actualizada que aumenta tanto la calidad del DSS como la confianza del usuario en el sistema.

Cuando proponemos, diseñamos y desarrollamos IDRA, nuestro objetivo es proporcionar el motor incremental que encaje en ese DSS adaptativo dentro de una herramienta de decisión empresarial. Para cumplir con este propósito, el conocimiento modelado por IDRA utiliza una estructura explicable como las reglas de decisión que podrían integrarse fácilmente en un DSS y un sistema de BI. IDRA, por su diseño, permite que el conjunto de reglas crezca con cualquier nueva instancia recibida. Sin embargo, tal como postulan Trépos et al. (2013) y se discute en la sección 2.2, debemos tratar de garantizar la accionabilidad del sistema. Esto significa que atributos como el tamaño o la complejidad de un conjunto de reglas no deberían afectar la capacidad de reacción y respuesta de un sistema. Para asegurar la accionabilidad, IDRA cuenta con un método de filtrado basado en una adaptación del trabajo de Almiñana et al. (2012). Este filtro reduce el conjunto de reglas a aquellas que tienen un cierto nivel de importancia, asegurando que la cantidad de reglas sea asequible y que el conocimiento incluido sea relevante para un entorno DSS. Asegurando estas estructuras optimizadas y con los resultados presentados en el Capítulo 4, podemos concluir que las contribuciones de IDRA son prometedoras en el campo de la solución de alta precisión para contextos decisionales. Desde el punto de vista industrial, la integración de este algoritmo en una solución de BI podría enriquecer significativamente sus resultados.

5.4 Trabajos futuros

A lo largo de todo este trabajo, las posibilidades de investigación y la relevancia del campo del data stream han quedado demostradas. En este sentido, la investigación futura vinculada a esta novedosa línea de investigación podría encaminarse en las direcciones que se presentan en este subapartado. Se consideran tanto las mejoras en el algoritmo presentado mediante el uso de técnicas de optimización, como el estudio de pasos posteriores en la integración del nuevo método en un sistema de información.

En cuanto a la mejora potencial que se podría obtener sobre el algoritmo presentado en este trabajo, consideramos la posibilidad de proponer un estudio más amplio de los valores de parametrización y modificaciones del entorno que podrían conducir a una mejora del rendimiento de IDRA en todas sus versiones paramétricas. Un análisis profundo de criterios como la métrica utilizada para ordenar y seleccionar reglas en la primera versión de IDRA (especialmente la nueva métrica propuesta de tendencia), su relevancia con respecto al tamaño de la ventana de evaluación o el estudio de valores no tan exigentes para el umbral de entropía. Además, los esfuerzos adicionales podrían centrarse en implementar diferentes tipos de acceso a la memoria para estudiar la optimización del algoritmo y proponer un método para la post-poda de los conjuntos de reglas de la versión reactiva, RIDRA.

Respetando la posibilidad de mejorar la calidad de IDRA, también podría hacerse mediante la optimización de los flujos de datos analizados por el sistema. En este sentido, un método de preprocesamiento online que automatice la optimización de los

flujos de datos analizados es un campo amplio, novedoso y complejo que podría generar grandes beneficios. Como ya se ha explicado, el algoritmo IDRA, en todas sus versiones, trabaja con valores discretos. En caso de que estos valores provengan de una variable continua, el sistema deberá convertirlos en categorías para poder utilizar el algoritmo. Este proceso no es especialmente costoso en términos de tiempo y complejidad, por lo que sería asumible por la gran mayoría de sistemas. Sin embargo, como en todos los contextos de flujo de datos, la potencial infinitud del flujo podría conducir a cambios que afecten la efectividad de los límites definidos previamente de manera manual. Plantear un nuevo método que procese el flujo de datos online, actualizando este criterio de discretización cuando cambie la distribución de datos mejoraría la independencia de los algoritmos, reduciendo la intervención de expertos. Este hecho podría reducir tiempos y, por tanto, costes, al proponer nuevos rangos cuando sea necesario, que un/a experto/a aceptará o modificará, sin necesidad de estudiar el problema en su totalidad. Es importante que estas técnicas conserven una lógica de apoyo en la que una persona experta toma la decisión final, de modo que se asegure una mínima intervención humana. De esta forma, las modificaciones y decisiones recomendadas por dichos sistemas son revisadas antes de que se lleven a cabo para detectar y evitar resultados sesgados o injustos. Otras técnicas de preprocesamiento, como la selección online de las variables más relevantes para la clasificación o la detección y estimación de valores atípicos y nulos, también podrían presentarse como una contribución relevante en este prometedor campo de estudio.

Una vez que el algoritmo ha sido propuesto y probado en diferentes condiciones de flujo de datos, debe integrarse en un sistema de información antes de usarse en un escenario real. El enfoque incremental propuesto en este trabajo está especialmente diseñado para ser integrado en un DSS. Por tanto, los próximos pasos serán integrar esta nueva metodología en un DSS que permita al decisor/a interactuar con la base de conocimiento y estudiar esta interacción para optimizar, al mismo tiempo, el framework que la contiene. Esta investigación se considera relevante, especialmente desde la perspectiva de los sistemas de información y la inteligencia de negocios o business intelligence.

References

Aggarwal, C.C.; Han, J.; Wang, J. & Yu, P.S. (2003). A framework for clustering evolving data streams. In Proceedings of the 2003 VLDB Conference, Berlin, Germany, 9–12 September 2003; Morgan Kaufmann: Burlington, MA, USA, pp. 81–92.

Alharbi, A.N.; Alnamlah, H. & Syed, L. (2017). Using opinion mining techniques on Twitter streaming data regards drug safety issues. In Proceedings of the Second International Conference on Internet of things, Data and Cloud Computing (ICC '17). Association for Computing Machinery, New York, NY, USA, Article 143, 1–5. https://doi.org/10.1145/3018896.3036386.

Alkouz, B. & Al Aghbari, Z. (2020). SNSJam: Road traffic analysis and prediction by fusing data from multiple social networks, Information Processing & Management, Volume 57, Issue 1, 102139, ISSN 0306-4573, https://doi.org/10.1016/j.ipm.2019.102139.

Almiñana, M.; Escudero, L.; Martín, A.P.; Rabasa, A. & Santamaría, L. (2012). A classification rule reduction algorithm based on significance domains. TOP. 22, 397–418.

Altalhi, S. & Gutub, A. (2021). A survey on predictions of cyber-attacks utilizing real-time twitter tracing recognition. J Ambient Intell Human Comput 12, 10209–10221. https://doi.org/10.1007/s12652-020-02789-z.

Alzghoul, A. & Löfstrand, M. (2011). Increasing availability of industrial systems through data stream mining, Computers & Industrial Engineering, 60 (2). Pages 195-205, ISSN 0360-8352, https://doi.org/10.1016/j.cie.2010.10.008.

Angelov, P. P. & Zhou, X. (2008). Evolving Fuzzy-Rule-Based Classifiers From Data Streams. IEEE Transactions on Fuzzy Systems, vol. 16, no. 6, pp. 1462-1475, Dec. 2008, doi: 10.1109/TFUZZ.2008.925904.

Anto, M. P.; Antony, M.; Muhsina, K. M.; Johny, N.; James, V. & Wilson, A. (2016). Product rating using sentiment analysis. 2016 International Conference on Electrical, Electronics, and Optimization Techniques (ICEEOT). pp. 3458-3462, doi: 10.1109/ICEEOT.2016.7755346.

Apache Spark. (n.d.) Spark Structured Streaming: https://spark.apache.org/streaming/ (accessed on 25 May 2022).

Babcock, B.; Babu, S.; Datar, M.; Motwani, R.; & Widom, J. (2002, June). Models and issues in data stream systems. In Proceedings of the twenty-first ACM SIGMOD-SIGACT-SIGART symposium on Principles of database systems (pp. 1–16), 3-6 June, Madison.

Baldin, M.; Breunig, T.; Cue, R.; De Vries, A.; Doornink, M.; Drevenak, J.; Fourdraine, R.; George, R.; Goodling, R.; Greenfield, R.; Jorgensen, M.W.; Lenkaitis, A.; Reinemann, D.; Saha, A.; Sankaraiah, C.; Shahinfar, S.; Siberski, C.; Wade, K.M.; Zhang, F.; Fadul-Pacheco, L.; Wangen, S.; da Silva, T.E. & Cabrera, V.E. (2021).

Integrated Decision Support Systems (IDSS) for Dairy Farming: A Discussion on How to Improve Their Sustained Adoption. Animals, 11, 2025. https://doi.org/10.3390/ani11072025.

Basseville, M. & Nikiforov, I. (1993). Detection of Abrupt Changes - Theory and Application. online, France.

Bertini, JR (2020). Graph embedded rules for explainable predictions in data streams. Neural Networks, 129, 174–192. doi:10.1016/j.neunet.2020.05.035.

Bifet, A. (2021, January). Adaptive Machine Learning for Data Streams [Video]. Instituto Centro de Investigación Operativa, Miguel Hernández University of Elche. https://www.youtube.com/watch?v=0FTF-tqvotA

Bifet, A.; De Francisci Morales, G.; Read, J.; Holmes, G. & Pfahringer, B. (2015). Efficient Online Evaluation of Big Data Stream Classifiers. In Proceedings of the 21th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD '15). Association for Computing Machinery, New York, NY, USA, 59–68. https://doi.org/10.1145/2783258.2783372.

Bifet, A. & Gavalda, R. (2007). Learning from time-changing data with adaptive windowing." In Proceedings of the 2007 SIAM international conference on data mining, pp. 443-448. Society for Industrial and Applied Mathematics.

Bifet, A.; Holmes, G.; Kirkby, R. & Pfahringer, B. (2010). MOA: Massive online analysis. J. Mach. Learn. Res. 11, 1601–1604.

Bifet, A.; Fan, W.; Zhang, W.; Liu, Q.; Tu, D.; Maniu, S.; He, C.; Qian, J. & Zhang, J. (2015). StreamDM C++ Stream Machine Learning in C++. Huawei Noah's Ark Lab. http://huawei-noah.github.io/streamDM-Cpp/ (Last accessed: 2022, May 20)

Bifet, A.; Zhang, J.; Fan, W.; He, C.; Zhang, J.; Qian, J.; Holmes, G. & Pfahringer, B. (2017). Extremely fast decision tree mining for evolving data streams. In Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Halifax, NS, Canada, 13–17 August 2017; Association for Computing Machinery: New York, NY, USA, pp. 1733–1742.

Bimonte, S.; Boussaid, O.; Schneider, M. & Ruelle, F. (2019). Design and Implementation of Active Stream Data Warehouses. International Journal of Data Warehousing and Mining, 15(2), 1–21. doi:10.4018/IJDWM.2019040101.

Caruccio, L.; Cirillo, S.; Deufemia, V. & Polese, G. (2021). Efficient Validation of Functional Dependencies during Incremental Discovery. In Proceedings of the 29th Italian Symposium on Advanced Database Systems, Pizzo Calabro, Italy, 5–9 September 2021.

Cendrowska, J. (1987). PRISM: An Algorithm for Inducing Modular Rules. International Journal of Man-Machine Studies, 27(4):349–370.

Chandola, V.; Banerjee, A. & Kumar, V. (2009). Anomaly Detection: A Survey. ACM Comput. Surv. 41, 3. 15:1–15:58

Chatziantoniou, D. & Doukidis, G. (2009). Data streams as an element of modern decision support. In D.B.A. Mehdi Khosrow-Pour (Ed.), Encyclopedia of information science and technology, second edition (pp. 941–949). IGI Global.

Chen, C.Y.-H.; Okhrin, Y. & Wang, T. (2022). Monitoring network changes in social media. Journal of Business and Economic Statistics, 10(2139), 1–16. https://doi.org/10.1080/07350015.2021.2016425.

Clarivate. (2022) Web of Science. https://www.webofknowledge.com/

De Francisci Morales, G. & Bifet A. (2015). SAMOA: Scalable Advanced Massive Online Analysis. Journal of Machine Learning Research, 16:149–153. URL http://samoa-project.net.

Delany, S.; Cunningham, P.; Tsymbal, A. & Coyle, L. (2005). A Case-based Technique for Tracking Concept Drift in Spam filtering. Knowledge-Based Sys. 18, 4–5, 187–195.

Deng, L. & Li, D. (2019). Multimedia data stream information mining algorithm based on jointed neural network and soft clustering. Multimed Tools Appl 78, 4021–4044. https://doi.org/10.1007/s11042-017-4964-7.

Destefanis, G.; Ortu, M.; Porru, S.; Swift, S. & Marchesi, M. (2016). A statistical comparison of Java and Python software metric properties. In Proceedings of the 7th International Workshop on Emerging Trends in Software Metrics, Austin, TX, USA, 14–22 May 2016; pp. 22–28.

Din, S.U.; Shao, J.; Kumar, J.; Mawuli, C.B.; Mahmud, S.M.H.; Zhang, W. & Yang, Q. (2021). Data stream classification with novel class detection: A review, comparison and challenges. Knowledge and Information Systems, 63(9), 2231–2276. https://doi.org/10.1007/s10115-021-01582-4.

Domingos, P. & Hulten, G. (2000). Mining high-speed data streams. In Proceedings of the Sixth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, New York, NY, USA, 20–23 August 2000; Association for Computing Machinery: New York, NY, USA, 2000; pp. 71–80.

Ducange, P.; Fazzolari, M.; Petrocchi, M. & Vecchio M. (2019). An effective Decision Support System for social media listening based on cross-source sentiment analysis models, Engineering Applications of Artificial Intelligence, Volume 78, Pages 71-85, ISSN 0952-1976, https://doi.org/10.1016/j.engappai.2018.10.014.

Ducange, P.; Marcelloni, F. & Pecori, R. (2021). Fuzzy Hoeffding Decision Tree for Data Stream Classification. International Journal of Computational Intelligence Systems. 14(1) 946–964.

Egebjerg, N. H.; Hedegaard, N.; Kuum, G.; Mukkamala, R. R. & Vatrapu, R. (2017). Big Social Data Analytics in Football: Predicting Spectators and TV Ratings from Facebook Data," 2017 IEEE International Congress on Big Data (BigData Congress), pp. 81-88, doi: 10.1109/BigDataCongress.2017.20.

Eom, S. (2022). Decision Support Systems.

Evans, J. R. & Lindner, C. H. (March 2012). Business Analytics: The Next Frontier for Decision Sciences. Decision Line. 43 (2).

Fahy, C. & Yang, S. (2022) Finding and Tracking Multi-Density Clusters in Online Dynamic Data Streams. IEEE Transactions on Big Data, 8(1), 178-192. https://doi.org/10.1109/TBDATA.2019.2922969.

Ferrer, F.; Aguilar, J. & Riquelme, J. (2005). Incremental rule learning and border examples selection from numerical data streams. J. Univers. Comput. Sci. 8, 1426–1439

Ferrer-Troyano, F.; Aguilar-Ruiz, J.S. & Riquelme, J.C. (2006). Data streams classification by incremental rule learning with parameterized generalization. In Proceedings of the 2006 ACM Symposium on Applied Computing, Dijon, France, 23–27 April 2006. Association for Computing Machinery: New York, NY, USA. pp. 657–661.

Fong, S.; Li, J.; Song, W.; Tian, Y.; Wong, R.K. & Dey, N. (2018). Predicting unusual energy consumption events from smart home sensor network by data stream mining with misclassified recall. Journal of Ambient Intelligence and Humanized Computing, 9(4), 1197–1221. https://doi.org/10.1007/s12652-018-0685-7.

Gaber, M.M.; Zaslavsky, A. & Krishnaswamy, S. (2005). Mining data streams. ACM Sigmod Rec. 34, 18.

Gama, J. (2010). Knowledge Discovery from Data Streams (1st. ed.). Chapman & Hall/CRC.

Gama, J. & Kosina, P. (2011). Learning Decision Rules from Data Streams. In Proceedings of the IJCAI International Joint Conference on Artificial Intelligence, Barcelona, Spain, 16–22 July 2011; AAAI Press/International Joint Conferences on Artificial Intelligence: Menlo Park, CA, USA, pp. 1255–1260.

Gama, J.; Rocha, R. & Medas, P. (2003). Accurate decision trees for mining high-speed data streams. In Proceedings of the Ninth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Washington, DC, USA, 23–27 August 2003; Association for Computing Machinery: New York, NY, USA.

Gama, J.; Zliobaite, I.; Bifet, A.; Pechenizkiy, M. & Bouchachia, A. (2014). A survey on concept drift adaptation. ACM Comput. Surv. 46, 4, Article 44 (March 2014), 37 pages. DOI: http://dx.doi.org/10.1145/2523813.

Garcia-Martin, E.; Bifet, A.; Lavesson, N.; König, R. & Linusson, H. (2022). Green Accelerated Hoeffding Tree. https://doi.org/10.48550/arXiv.2205.03184.

Ghanem, T.M.; Hammad, M.A.; Mokbel, M.F.; Aref, W.G. & Elmagarmid, A.K. (2007). Incremental Evaluation of Sliding-Window Queries over Data Streams. IEEE Trans. Knowl. Data Eng. 19, 57–72.

Ginzberg, M.J. & Stohr, E.A. (1982). Decision support systems: Issues and perspectives. NYU Working Paper No. IS-82-12.

Gorry, G.A. & Scott Morton, M.S. (1971). A framework for management information systems. Sloan Management Review, 13, 55–70.

Guo, X. & Díaz López, A. (2013). Mobile Decision Support System Usage in Organizations" Proceedings of the Nineteenth Americas Conference on Information Systems, Chicago, Illinois, August 15–17, 2013.

Han, C. & Zhang, Q. (2021). Optimization of supply chain efficiency management based on machine learning and neural network. Neural Comput & Applic 33, 1419–1433. https://doi.org/10.1007/s00521-020-05023-1.

Hassanat, A. (2018). Norm-based binary search trees for speeding up KNN big data classification, Computers 7(4), 54.

Hulten, G.; Spencer, L. & Domingos, P. (2001). Mining time-changing data streams. In Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, 23–29 August 2001; Association for Computing Machinery: New York, NY, USA, 2001.

Hulten, G. & Domingos, P. (2003). VFML -- A toolkit for mining high-speed time-changing data streams. http://www.cs.washington.edu/dm/vfml/. (Last accessed: 2022, May 20).

Jiang, N. & Gruenwald, L. (2006). Research issues in data stream association rule mining. ACM Sigmod Rec., 35, 14–19.

Kolajo, T.; Daramola, O. & Adebiyi, A. (2019). Big data stream analysis: A systematic literature review. Journal of Big Data, 6(1), 47. https://doi.org/10.1186/s40537-019-0210-7.

Kosina, P. & Gama, J. (2013). Very fast decision rules for classification in data streams. Data Min. Knowl. Discov., 29, 168–202.

Kozina, Y.; Volkova, N. & Horpenko D. (2018). Mobile Application for Decision Support in Multi-Criteria Problems. 2018 IEEE Second International Conference on Data Stream Mining & Processing (DSMP), 2018, pp. 56-59, doi: 10.1109/DSMP.2018.8478499.

Laforet, F.; Olms, C.; Biczok, R. & Böhm, K. (2021). An ensemble technique for better decisions based on data streams and its application to data privacy. IEEE Transactions

on Knowledge and Data Engineering, 33(12), 3662–3674. https://doi.org/10.1109/TKDE.2020.2977035.

Le, T.; Stahl, F.; Gomes, J.B.; Gaber, M.M. & Di Fatta, G. (2014). Computationally efficient rule-based classification for continuous streaming data. In: Thirty-fourth SGAI International Conference on Artificial Intelligence, 9-11 Dec 2014, Cambridge, England, pp. 21-34.

Li, F. & Liu, Q. (2008). An Improved Algorithm of Decision Trees for Streaming Data Based on VFDT. 2008 International Symposium on Information Science and Engineering, pp. 597-600, doi: 10.1109/ISISE.2008.256.

Liu, C.; Chen, Y. & Zhao, L. (2021). An adaptive prediction method based on data stream mining for future driving cycle of vehicle. Proc. Inst. Mech. Eng. Part D J. Automob. Eng. 235, 1702–1712.

Lu, J.; Liu, A.; Song, Y.; Guangquan, Z., et al. (2020). Data-driven decision support under concept drift in streamed big data. Complex Intell. Syst, 6(1), 157–163. https://doi.org/10.1007/s40747-019-00124-4.

Machine Learning Repository — UCI. (2012, February 14) Bank Marketing Data Set. Available online: https://archive.ics.uci.edu/ml/datasets/Bank%2BMarketing (accessed on 9 December 2021).

Manapragada, C.; Webb, G. I. & Salehi, M. (2018). Extremely fast decision tree. In C-J. Lin, & H. Xiong (Eds.), KDD' 2018 - Proceedings of the 24th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining: August 19-23, 2018, London, United Kingdom, pp. 1953-1962. Association for Computing Machinery (ACM). https://doi.org/10.1145/3219819.3220005.

Mollá N.; Heavin C. & Rabasa A. (2022a) Data-driven decision making: new opportunities for DSS in data stream contexts. Journal of Decision Systems. https://doi.org/10.1080/12460125.2022.2071404.

Mollá, N.; Rabasa, A.; Rodríguez-Sala, J.J.; Sánchez-Soriano, J. & Ferrándiz A. (2022b) A. Incremental Decision Rules Algorithm: A Probabilistic and Dynamic Approach to Decisional Data Stream Problems. Mathematics, 10, 16. https://doi.org/10.3390/math10010016.

Moro, S.; Cortez, P. & Rita, P. (2014). A Data-Driven Approach to Predict the Success of Bank Telemarketing. Decis. Support Syst. 62, 22–31.

Nguyen, HL.; Woon, YK. & Ng, WK. (2015). A survey on data stream clustering and classification. Knowl Inf Syst 45, 535–569. https://doi.org/10.1007/s10115-014-0808-1.

O'Callaghan, L.; Mishra, N.; Meyerson, A.; Guha, S. & Motwani, R. (2002) Streaming-data algorithms for high-quality clustering. Proceedings of the 18th international conference on data engineering, pp 685–694.

O'Driscoll, K. (2022, February). Business Analytics. Cork University Business School, University College Cork.

Orenes, Y.; Rabasa, A.; Rodriguez-Sala, J.J. & Sanchez-Soriano, J. (2021). Benchmarking Analysis of the Accuracy of Classification Methods Related to Entropy. Entropy. 23, 850.

Puschmann, D.; Barnaghi, P. & Tafazolli, R. (2017). Adaptive clustering for dynamic IoT data streams. IEEE Internet Things J 4(1):64–74.

Quinlan, J.R. (1986). Induction of Decision Trees. Mach. Learn. 1, 81–106.

Quinlan, J.R. (1987). C4.5: Programs for Machine Learning; Learning decision lists; Morgan Kaufmann Publishers: San Mateo, CA, USA. pp. 229–246.

Ramírez-Gallego, S.; Krawczyk, B.; García, S.; Woźniak, M.; Benítez, J.M. & Herrera, F. (2017). Nearest Neighbor Classification for High-Speed Big Data Streams Using Spark. IEEE Transactions on Systems, Man, and Cybernetics: Systems, 47, 2727-2739.

Reed, A. & Kranch, M. (2017). Identifying HTTPS-Protected Netflix Videos in Real-Time. In Proceedings of the Seventh ACM on Conference on Data and Application Security and Privacy (CODASPY '17). Association for Computing Machinery, New York, NY, USA, 361–368. https://doi.org/10.1145/3029806.3029821.

Rodrigues, A.P. & Chiplunkar, N.N. (2019). A new big data approach for topic classification and sentiment analysis of Twitter data. Evol. Intel. https://doi.org/10.1007/s12065-019-00236-3.

Rutkowski, L.; Pietruczuk, L.; Duda, P. & Jaworski, M. (2013). Decision Trees for Mining Data Streams Based on the McDiarmid's Bound. IEEE Trans. Knowl. Data Eng. 25, 1272–1279.

Schirmer, P.; Papenbrock, T.; Kruse, S.; Naumann, F.; Hempfing, D.; Mayer, T. & Neuschäfer-Rube, D. (2019). DynFD: Functional Dependency Discovery in Dynamic Datasets. In Proceedings of the 22nd International Conference on Extending Database Technology, Advances in Database Technology-EDBT, Lisbon, Portugal, 26–29 March 2019. OpenProceed-ings.org. pp. 253–264, ISBN 978-3-89318-081-3.

Scholz, M. & Klinkenberg, R. (2007). Boosting Classifiers for Drifting Concepts. Intell. Data Anal.11, 1(2007), 3–28.

Scott Morton, M.S. (1971). Management decision systems; computer-based support for decision making. Division of Research, Graduate School of Business Administration, Harvard University.

Selvan, L. G. S. & Moh T. -S. (2015). A framework for fast-feedback opinion mining on Twitter data streams. 2015 International Conference on Collaboration Technologies and Systems (CTS). pp. 314-318, doi: 10.1109/CTS.2015.7210440.

Severiano, C.A.; de Silva, P.C.L.E.; Cohen, M.W. & Guimarães, F.G. (2021). Evolving fuzzy time series for spatio-temporal forecasting in renewable energy systems. Renew. Energy. 171, 764–783.

Shaker, A. & Hüllermeier, E. (2012). IBLStreams: A system for instance-based classification and regression on data streams. Evol. Syst. 3, 235–249.

Shannon, C.E. (1948). A Mathematical Theory of Communication. Bell Syst. Tech. J. 27, 379–423.

Sprague, R.H., & Watson, H.J. (1979). Bit by Bit: Toward decision support systems. California Management Review, 22(1), 60–68. https://doi.org/10.2307/4116485.

Srivani, B.; Sandhya, N. & Padmaja Rani, B. (2020). Literature review and analysis on big data stream classification techniques. International Journal of Knowledge-based and Intelligent Engineering Systems, 24(3), 205–215. doi:10.3233/KES-200042.

Street, W.N. & Kim, Y.S. (2001). A streaming ensemble algorithm (SEA) for large-scale classification. In Proceedings of the Seventh ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, 26–29 August 2001; Association for Computing Machinery: New York, NY, USA, 2001; Volume 4, pp. 377–382.

Tableau. (2022, May 17). Comparing Business Intelligence, Business Analytics and Data Analytics. https://www.tableau.com/learn/articles/business-intelligence/bi-business-analytics.

Trépos, R.; Salleb-Aouissi, A.; Cordier, MO. et al. (2013). Building actions from classification rules. Knowl Inf Syst 34, 267–298. https://doi.org/10.1007/s10115-011-0466-5.

Tsymbal, A. (2004). The Problem of Concept Drift: Definitions and Related Work. Tech. rep. Department of Computer Science, Trinity College, Dublin.

Wan, L.; Ng, W.K.; Dang, X.H.; Yu, P.S. & Zhang, K. (2009) Density-based clustering of data streams at multiple resolutions. ACM Trans Knowl. Discov. Data (TKDD). 3(3):1–28.

Wang, Y.; Zhang, X. & Wang, Z. (2018). A proactive decision support system for online event streams. Int. J. Inf. Technol. Decis. Mak. 17, 1891–1913.

Wares, S.; Isaacs, J. & Elyan, E. (2019). Data stream mining: Methods and challenges for handling concept drift. SN Applied Sciences. 1(11), 1412. https://doi.org/10.1007/s42452-019-1433-0.

Widmer, G. & Kubat, M. (1993). Effective Learning in Dynamic Environments by Explicit Context Tracking. In Proc. of the Eur. Conf. on Mach. Learn. (ECML). Springer, Berlin, 227–243.

Yang, H.; Li, P.; He, Z.; Guo, X.; Fong, S. & Chen, H. A decision support system using combined classifier for high-speed data stream in smart grid. Enterp. Inf. Syst. 2016, 10, 947–958.

Yin, C.; Xia, L.; Zhang, S. et al. (2018). Improved clustering algorithm based on high-speed network data stream. Soft Comput. 22, 4185–4195. https://doi.org/10.1007/s00500-017-2708-2.

Zaniolo, C. (2012). Logical Foundations of Continuous Query Languages for Data Streams. In Datalog in Academia and Industry. Lecture Notes in Computer Science. Barceló, P., Pichler, R., Eds.; Springer: Berlin/Heidelberg, Germany, 2012; Volume 7494.

Zeira, G.; Maimon, O.; Last, M. & Rokach, L. (2004). Change Detection in Classification Models Induced from Time-Series Data. In Data Mining in Time Series Databases. Vol. 57. World Scientific, Singapore, 101–125.

Zhang, P.; Zhu, X.; Shi, Y.; Guo, L. & Wu, X. (2011). Robust ensemble learning for mining noisy data streams. Decision Support Systems. 50(2), 469–479. https://doi.org/10.1016/j.dss.2010.11.004.

Zhang, L.; Lin, J.; Liu, B.; Zhang, Z.; Yan, X. & Wei, M. (2019). A Review on Deep Learning Applications in Prognostics and Health Management. IEEE Access 2019, 7, 162415–162438.

Zubaroğlu, A. & Atalay, V. (2021). Data stream clustering: a review. Artif Intell Rev 54, 1201–1236. https://doi.org/10.1007/s10462-020-09874-x.