FEDERAL UNIVERSITY OF SÃO CARLOS

CENTER OF EXACT SCIENCES AND TECHNOLOGY

GRADUATE PROGRAM IN COMPUTER SCIENCE

ENHANCING SOLAR FLARE FORECASTING: A MULTI-CLASS AND MULTI-LABEL CLASSIFICATION APPROACH TO HANDLE IMBALANCED TIME SERIES

SÉRGIO LUISIR DÍSCOLA JUNIOR

ADVISOR: PROF. DR. MÁRCIO MERINO FERNANDES

São Carlos – SP

June, 2019

FEDERAL UNIVERSITY OF SÃO CARLOS

CENTER OF EXACT SCIENCES AND TECHNOLOGY

GRADUATE PROGRAM IN COMPUTER SCIENCE

ENHANCING SOLAR FLARE FORECASTING: A MULTI-CLASS AND MULTI-LABEL CLASSIFICATION APPROACH TO HANDLE IMBALANCED TIME SERIES

SÉRGIO LUISIR DÍSCOLA JUNIOR

A thesis presented to the Graduate Program in Computer Science (PPG-CC) from the Federal University of São Carlos, as part of the requirements for obtaining a Doctorate in Computer Science. Concentration Area: Signal and Image Processing and Computer Architecture
Advisor: Prof. Dr. Márcio Merino Fernandes

São Carlos – SP

June, 2019

UNIVERSIDADE FEDERAL DE SÃO CARLOS

Centro de Ciências Exatas e de Tecnologia
Programa de Pós-Graduação em Ciência da Computação

Folha de Aprovação

Assinaturas dos membros da comissão examinadora que avaliou e aprovou a Defesa de Tese de Doutorado do candidato Sérgio Luisir Díscola Junior, realizada em 21/06/2019:

[illegible]

Prof. Dr. Márcio Merino Fernandes
UFSCar

[illegible]

Prof. Dr. Renato Bueno
UFSCar

[illegible]

Profa. Dra. Solange Oliveira Rezende
USP

[illegible]

Prof. Dr. Alisson Dal Lago
INPE

[illegible]

Prof. Dr. Robson Leonardo Ferreira Cordeiro
USP

To God and all my family, specially my wife, Marcela, and my daughter, Marina.

ACKNOWLEDGMENTS

I would like to thank my advisor Márcio Merino Fernandes for the helpful hints and support during the doctorate; José Roberto Cecatto for all support about Solar Flares; Marilde Terezinha Prado Santos for the several reviews and discussions about the project; Emerson Carlos Pedrino for all his tips about the doctorate stages; Paulo Cesar Donizeti Paris for all technical support in the laboratory; IFSP for the opportunity of improving my knowledge; CNPq, Fapesp and CAPES for the financial support; and God and all my family for encouraging and guiding me. A special thanks to my wife, Marcela and my daughter, Marina for giving all strength possible to me during the doctorate.

RESUMO

Explosões solares são enormes liberações de energia do Sol. Elas são classificadas em cinco níveis (A, B, C, M e X) de acordo com seus possíveis danos à Terra e podem produzir grandes impactos nos sistemas de comunicação, ameaçando atividades humanas que dependem de satélites e GPS. Portanto, prevê-las antecipadamente pode reduzir tais impactos. No entanto, a previsão de explosões solares apresentam desafios significativos: (a) a sequência de dados deve ser rastreada devido à sua influência no fenômeno; (b) as características solares e os intervalos de dados que influenciam neste fenômeno não estão estabelecidos; (c) a previsão deve ser realizada em tempo razoável; (d) os dados são altamente desbalanceados, (e) as classes adjacentes às vezes são difíceis de distinguir, (f) a maioria das abordagens realizam previsão binária (agregando classes de explosões solar), ao invés de prover uma previsão multi-classe. Este trabalho de doutorado propôs um método que aborda esses desafios simultaneamente, diferentemente de trabalhos anteriores, que tendem a lidar com um desafio por vez.

Para tanto, inicialmente, nós objetivamos prever explosões solares para um horizonte de poucos dias. Foi proposto o método SeMiner, o qual permite a previsão de explosões, dados valores observados passados. O método SeMiner processa séries temporais de Raios-X em sequencias empregando o algoritmo "*Series-to-Sequence*" (SS) através de uma abordagem de janela deslizante configurada pelo especialista do domínio. Este método considera uma sequencia de instâncias no processo de mineração, lidando com o desafio (a). Após isso, a seleção de características é aplicada a fim de determinar o intervalo de dados na série temporal, que mais influencia o processo de previsão. Isto tratou o desafio (b). Então, as sequencias processadas são submetidas a classificadores tradicionais para gerar um modelo que prevê níveis de Raios-X futuros. O SeMiner alcançou 73% de acurácia para uma previsão de 1 dia, 71% e 79%, respectivamente para TPR e TNR.

Um segundo passo foi desenvolvido através da paralelização do algoritmo SS, o qual melhorou o seu desempenho. Este desenvolvimento lidou com a questão (c), implementando esta otimização através da plataforma CUDA. Esta implementação permitiu um "speedup" de 4.36 no seu tempo de processamento devido à distribuição do processamento entre as GPUs ("Unidades Gráficas de Processamento").

O terceiro passo foi composto pela melhoria do método SeMiner. Este passo lidou com os desafios restantes através do desenvolvimento de um novo método chamado ECID ("*Ensemble of classifiers for imbalanced datasets*"). Para cada classe de explosão solar, o ECID

aplica uma amostragem aleatória estratificada para treinar classificadores base binários, fortalecendo suas sensitividades para uma dada classe em um cenário desbalanceado. Esta etapa tratou a questão (d).
Através de uma abordagem de "*Bootstrap*" modificada, o ECID usa um método de agregação que combina os resultados dos classificadores base, possibilitando uma previsão multi-classe e multi-label. Desta forma, o desafio (e) foi trabalhado. Os resultados mostraram que o ECID é bem adequado para previsão de explosões solares, alcançando um TPR médio de 91% e uma precisão média de 97% em um horizonte de previsão de um dia.

Palavras-chave: Explosão solar, previsão, séries temporais, mineração de dados, classificação em séries temporais, seleção de características, conjunto de dados desbalanceados

Abstract

Solar flares are huge releases of energy from the Sun. They are categorized in five levels according to their potential damage to Earth (A, B, C, M, and X) and may produce strong impacts to communication systems, threatening human activities dependent on satellites and GPS. Therefore, predicting it in advance may reduce their negative impacts. However, solar flare forecasting has significant challenges: (a) the sequence of data influences the phenomena and should be tracked; (b) the features and intervals that cause and influence the phenomena are not defined; (c) the forecasting should be performed in an affordable time; (d) the data is highly imbalanced, (e) adjacent classes are sometimes difficult to distinguish, (f) the majority approaches perform binary forecasting (aggregating solar flare classes), instead of multi-class, as actually required. This work proposed a method that tackles these challenges simultaneously, being different from previous works, which tend to handle a challenge per time.

First, we aimed to forecast the X-ray levels expected for the next few days. We proposed the SeMiner method that allows the labels prediction given past observations. SeMiner processes X-ray time series into sequences employing the new Series-to-Sequence (SS) algorithm through a sliding window approach configured by a domain specialist. This method allows to consider the sequence of instances in the mining process, handling challenge (a). Next, feature selection is employed in order to determine which interval of data in the time series, most influences the forecasting process, handling challenge (b). Then, the processed sequences are submitted to a traditional classifier to generate a model that predicts future X-ray levels. SeMiner reached 73% of accuracy for a 2-day forecast, 71% and 79%, respectively for True Positive and True Negative Rates.

Second, we parallelized SS to increase its performance, in order to tackle issue (c), by implementing it in CUDA platform. This implementation allowed a speedup of 4.36 in its time processing due to the distribution of the processing among the GPUs (Graphics Processing Unit).

Third, we improved SeMiner to tackle the remaining challenges by developing a new method called Ensemble of classifiers for imbalanced datasets (ECID). For each solar flare class, ECID employs a stratified random sampling for training binary-class base inducers, strengthening their sensitivity to a given class in a very imbalanced scenario, which tackled issue (d).

Using a modified bootstrap approach, an aggregation method combines the inducers results, enabling a multi-class and multi-label forecasting and thus, handling the issue of adjacent classes (challenge (e)). The results showed that ECID is well-suited for forecasting solar flares, achieving a maximum mean of True Positive Rate (TPR) of 91% and a Precision of 97%, in a time horizon of one day.

Keywords: Solar flare, forecasting, time series, data mining, classifiers, time series classification, feature selection, imbalanced dataset

GLOSSARY

AIA – *Atmospheric Imaging Assembly*

AR – *Active Region*

CME – *Coronal Mass Ejection*

CUDA – *Compute Unified Device Architecture*

DM – *Data Mining*

ECID – *Ensemble of Classifiers for Imbalanced Datasets*

EC – *Ensemble of Classifiers*

GAPIS – *Grupo de Arquitetura e Processamento de Imagens e Sinais*

GOES – *Geostationary Operational Environmental Satellite*

GPS – *Global Positioning System*

GPU – *Graphics Processing Unit*

HF – *High Frequency*

INPE – *National Institute for Space Research from Brazil*

JSOC – *Joint Science Operations Center from Stanford Solar Center*

NOAA/SWPC – *National Oceanic and Atmospheric Administration / Space Weather Prediction Center*

ROC – *Receiver Operating Characteristic*

SDO/HMI – *Solar Dynamics Observatory/Helioseismic and Magnetic Imager*

SFF – *Solar Flare Forecasting*

SOHO/MDI – *Solar and Heliospheric Observatory/Michelson Doppler Imager*

SS – *Series to Sequence algorithm*

Seminer – *Sequence Miner*

VHF – *Very High Frequency*

List of Terms

- **Feature**: A feature is a data field representing a physical phenomenon.
- **Instance**: Instances are the values of a set of predetermined features.
- **Observations**: Observations are the values of one and only one predetermined feature in a given instant of time *t*.
- **Attributes**: An attribute is a data field, representing a characteristic of a data object. In this thesis it does not represent a physical phenomenon.
- **Dataset**: A dataset is a set of instances.
- **Solar dataset**: a solar dataset is a dataset composed of features that describe the solar flare phenomenon.
- **Solar features**: a solar feature describes any feature used in solar flare phenomenon.
- **Event**: an event is a synonym of solar flare in this thesis.
- **Photosphere**: the Photosphere is the visible surface layer of the Sun.
- **Chromosphere**: the Chromosphere is a Sun's layer approximately 400km above the Photosphere.
- **Corona**: Corona is the outer layer of the Sun.
- **Sunspot**: Sunspots are magnetic regions of the Sun with much stronger magnetic strengths than Earth's magnetic field. Solar flares occurs near Sunspots.
- **Expert systems**: an expert system implements computational decision-making to simulate rules used by humans.
- **Gaussian distribution**: a Gaussian distribution is a continuous probability distribution used for represent random variables in statistics area.

List of Figures

List of Tables

CONTENTS

I INTRODUCTION

Chapter 1

INTRODUCTION

According to Wackler (2015), "*Space weather refers to the dynamic conditions of the space environment that arise from interactions with emissions from the sun, including solar flares, solar energetic particles, and coronal mass ejections. These emissions can affect Earth and its surrounding space, potentially causing disruption to electric power transmission; satellite, aircraft, and spacecraft operations; telecommunications; position, navigation, timing services; and other technology and infrastructure.*". In Brazil, INPE heads the EMBRACE program, which aims to "*develop and operate a program of space weather*" (http://www2.inpe.br/climaespacial/por tal/the-embrace-program/). Among the phenomena described in the Space Weather definition, solar flares are important events that must be predicted in order to prevent their impacts on Earth's devices.

Solar flares are sudden releases of large amounts of energy (10^{25} - 10^{32} erg) from the active regions of the solar atmosphere (HOLMAN, 2006). These phenomena can last from tens of seconds to few hours, depending on the intensity, and may emit a broad spectrum of electromagnetic waves from radio up to X-rays or even gamma-rays. Considering the X-ray flux measured at the wavelength range of 1-8 Angstrom, solar flares are categorized into 5 classes, named, *A, B, C, M and X*, respectively in order of their strength. Depending on the intensity, a solar flare causes impacts on High Frequency (HF) and Very High Frequency (VHF) radio communication (TSURUTANI et al., 2009). Changes in the ionosphere caused by X-ray flux variation emitted by the Sun can also interfere in satellite communication, since it uses high-frequency signals (BASU et al., 2010). In addition, solar flares can impact on the Global Positioning System (GPS), and when an associated Coronal Mass Ejection (CME) is produced electricity power grids are also affected, causing blackouts in extreme cases. Depending on the severity of the predicted solar flare, different institutions or companies must be warned in order to trigger their mitigation procedures.

Several computational methods have been proposed in the literature for Solar Flare Forecasting (SFF), but there are still important open challenges: the imbalanced characteristic of the problem; the physical phenomenon which is yet not fully understood by the astrophysicists (PRIEST; FORBES, 2002; RABOONIK et al., 2016); and, the handling of solar flare classification into levels and sub-levels. It is imperative to develop an operational and optimized solar flare forecasting method that deals with these challenges to allow solar flare forecasting as accurate as possible.

1.1 Context

The application domain of this thesis work is within the solar flare forecasting task. This task is important to avoid its related damages. The forecasting is classified according to the X-ray emitted by the Sun and its related impacts. The forecasting methods found in the literature may use several different possible features because it was not detected the ones that definitely lead to the most accurate results. Also, these methods are implemented using different approaches, mainly using statistical methods or machine learning techniques. Finally, there are a diverse set of significant open issues that must be tackled to improve the existing solar flare forecasting methods. Next, we describe the context that this thesis work is inserted.

1.1.1 Solar flare classification

Solar flare forecasting methods perform their predictions based on a classification that depends on the peak flux emitted by Sun during the event. Each class is related to its possible damages to Earth's electronic devices and technological systems and services (LEE et al., 2012). Table 1.1 shows the traditional classification of solar flares with their respective X-ray ranges and possible effects on Earth.

Table 1.1: Description of solar flare classes (TANDBERG-HANSSEN; EMSLIE, 2009)

Solar flare classification	Associated X-ray flux - I (W/m^2)	Possible effects on Earth
B	$I < 1E$-06	none
C	$1E$-06 $\leq I < 1E$-05	Possible effects on space missions.
M	$1E$-05 $\leq I < 1E$-04	Blackout in radio transmissions and possible damages in astronauts outside spacecraft.
X	$I \geq 1E$-04	Damage to satellites, communication systems, power distribution stations and electronic equipment

Solar flares that emit up to 1E-6 W/m^2 are classified as B and do not cause damage to Earth. If the intensity is in the range of 1E-6 to 1E-5 W/m^2, they are classified as C and may have a low effect on Earth and space missions. Explosions classified as M occur when the X-ray released is between 1E-5 and 1E-4 W/m^2 and can cause blackouts in radio transmissions and damages in space missions. Finally, solar flares of class X are the highest and can damage satellites, GPS systems, power stations, and electronic equipment. Each class is divided into sub-levels, according to the highest amount of energy released during the event. Thus, a more specific classification of flares is given by B1.0 to B9.9, C1.0 to C9.9, M1.0 to M9.9, and X1.0 to higher levels. Where the indexes 1.0 to 9.9 correspond to the intensity of X-rays within the solar flare class range. For example, if an event emitted 3E-07 W/m^2, it is fully classified as B3.0 (TANDBERG-HANSSEN; EMSLIE, 2009).

1.1.2 Data about Solar Activities

In order to better understand the context of solar flare forecasting, it is needed to understand which solar data are available in the literature to use in the methods. There are several data sources used to forecast solar flares, like: (1) X-ray emitted by Sun; (2) magnetic features from the Sun's Photosphere; (3) topology of the active regions, among others.

There is no consensus, and there is no definitive study regarding the causes and physical explanations of this phenomenon. So, it has not yet been defined which data features are most relevant to use in solar flare forecasting. However, there are indications in the literature that the time series of magnetic features of the active regions, extracted from images called magnetogram vectors, can be used satisfactorily to perform the prediction task. Another important feature is the X-ray intensity emitted by the Sun. This information is available on the NOAA website (a North American organization for studies about space climate) as a time series with sampling periods of one or five minutes. There are several other sets of features that can be used, such as the international sunspot number, the area of solar active regions, MacIntosh magnetic classification and the 10.7 cm Radio flux - measured in solar flux unity (sfu) - emitted by the Sun. However, in this work, we intended to use time series X-rays flux and magnetic features in the forecasting method.

Regarding solar flares occurrence, the ones classified as C are events with low frequencies and flares of classes M and X are rare. According to (LEE et al., 2012) there were only 7400 C-class flares, 1200 flares classified as M and 120 as X between 1996 and 2000. These values represent a percentage of 0.47%, 0.08% and 0.006% of the total number of observations of X-ray intensities at a sampling period of 5 minutes.

The pattern of distribution of solar flares considering frequency and strength, among other solar parameters is well correlated and approximately follows the average 11-year Solar Activity Cycle. Roughly, this cycle presents an initial period with quite few sunspot groups and occurrences of low intensity (B, C) flares, as well as lower total irradiance and 10.7 cm radio flux values. This phase is designated by *solar minimum epoch*. It is followed by a phase when a gradual increase is observed in those parameters, mainly the frequency of occurrences and intensity of flares. It reaches its peak after around 4-5 year, a period called solar maximum.

Regarding the volume of data, there are at least 30 features obtained from the magnetogram vector and also a time series of X-rays. The time series obtained from the magnetogram vector has a sampling period of 12 minutes, while the X-ray time series, a sampling period of 5 minutes. Considering a full Solar Activity Cycle (11 years) and that each instant in the solar Photosphere can exist multiple Active Regions with their respective magnetogram time series, it is expected to have a large amount of data to be processed by the forecasting method. This doctorate work used 7 years of solar data.

Therefore, the data employed in this work have the following characteristics:

1. Highly imbalanced, because the severe and extreme solar flares are rarer;
2. Periodicity due to the Solar Cycle characteristic;
3. Large amount of solar data to consider in the forecasting task.

1.1.3 Approaches to solar flares forecasting

There are two main approaches used by methods that forecast solar flares: (1) forecasting with statistical methods based on data probability distribution; (2) forecasting with data mining techniques.

The first approach often studies the distribution of solar events within an interval, trying to build statistical models. One of the first statistical methods used to forecast solar flares is presented in McIntosh (1990). This work describes an expert system called *Theo*, which consists of a set of rules based on knowledge of domain specialists (astrophysicists) about solar characteristics (MCINTOSH, 1990). In Gallagher, Moon e Wang (2002), it was proposed a statistical model by estimating the probability of each solar event through a Poison distribution analysis. Another related work that uses statistical tools is found in Barnes et al. (2007). This work assumes a Gaussian distribution of solar flare events applied to a statistical approach named *Discriminant Analysis*. It aims to give probabilities for the occurrence of a phenomenon for different groups. For instance, in solar flare forecasting, it would estimate the individual probability of occurrence of flares of classes C, M and X.

Recently, many solar flare forecasting methods have been developed using data mining techniques, taking advantage of machine learning algorithms. These works usually differ in aspects such as: what is foreseen, the solar features used, the techniques for the preprocessing step, and the adopted classification methods.

Most forecasting methods present their results in different ways: some works group solar flare classes to give a positive answer and some works forecast solar flares in individual classes. Considering classes C, M, and X, some methods return positive for classes greater than or equal to C (AHMED et al., 2013), others consider positive for forecasts greater than or equal to M (NISHIZUKA et al., 2017; BOBRA; COUVIDAT, 2015; LI; ZHU, 2013; YU et al., 2010), and others produce individual probabilities for each class (C, M, X).

In several works found in the literature regarding solar flare forecasting, it can be observed that a very challenging task is to define the most significant features used in the solar flare forecasting process. We can find papers that use magnetogram vector (BOBRA; COUVIDAT, 2015; YU et al., 2009, 2010), sunspot number, sunspot area (GALLAGHER; MOON; WANG, 2002), radio flux, X-ray flux (LI; ZHU, 2013), among others, as input to the process of solar flare forecasting. A significant feature used in solar flare forecasting is the intensity of the X-ray flux emitted by the Sun because it establishes a relation between a solar flare event and its impact on Earth using an event classification range.

The most important difference between forecasting methods is the preprocessing method used to prepare raw data to be submitted to the classifier. In general, the forecasting methods enable classifiers to predict future events through the mapping of values of solar data observed in a particular instant of time with events occurred in a particular time window. This approach does not consider the evolution of a time series in a particular period to perform the mapping. This is the reason why those methods do not consider the evolution of a solar data time series in the mapping, which causes the loss of valuable information for the forecasting process. Some forecasting methods set solar data snapshots to the class of a solar flare occurred after those observed data (NISHIZUKA et al., 2017; BOBRA; COUVIDAT, 2015; AHMED et al., 2013; YUAN et al., 2010; COLAK; QAHWAJI, 2009). However, there are also works that map subseries of the solar data into events observed in the future, so that they adequately consider the historical evolution of solar data (LI; ZHU, 2013; YU et al., 2010).

Classification methods used in the data mining process also differ among works in the literature. We find forecasting methods using Support Vector Machines (BOBRA; COUVIDAT, 2015; ZAVVARI et al., 2015; QAHWAJI; COLAK, 2007), Artificial Neural Networks (ZAVVARI et al., 2015; LI; ZHU, 2013; AHMED et al., 2013; WANG et al., 2008), C4.5 decision trees (ZHANG; LIU; WANG, 2011; HUANG et al., 2010; YU et al., 2010, 2009; GALLAGHER; MOON; WANG, 2002), Naive Bayes (ZAVVARI et al., 2015) and Bayesian Networks (ZHANG; LIU; WANG, 2011).

Current forecasting methods usually are not flexible enough to be set up with the specialist knowledge: their configuration is set up once, and no parameter can be modified afterward. Parameters that need specialist knowledge such as solar features, the size and period of both, training and testing data sets, the period that is considered to map values and future events, are all defined a single time, at the beginning of the process. Few methods allow the configuration of some of those parameters, as presented in Li e Zhu (2013).

1.1.4 Open issues

Considering the data characteristics and the approaches used, we found important open issues in this application domain that could improve current solar flare forecasting methods:

- The data are highly **imbalanced** and previous works have not deeply dealt with this issue.
- The occurrence of the studied phenomenon has a cyclic behavior, so the **evolution** of the time series is a significant aspect that should be employed by the forecasting model.

- Solar flares are classified in levels (A, B, C, M and X) and sub-levels (from 1.0 to 9.9). Hence, solar flares classified in the boundary of their subsequent class have similar impacts on Earth's devices. For example, a solar flare classified as C9.9 has similar impacts to an M1.0 flare. Thus, if a forecasting method predicts a C event, the actual event may be a C1.0 to C9.9, i.e., it may have different impacts. Thus, it is important to differentiate possible solar flares that are in the boundary of stronger ones;

- The knowledge of domain specialists is essential in the forecasting process. He/she knows information as the solar data characteristics, and the most significant features that should be used in the forecasting method. These parameters usually differ according to the period of the Solar Cycle and/or the last occurrences of the event series. Thus, a method that can be configured with the astrophysicist's knowledge may improve the forecasting results.

- Most of the solar flare forecasting methods use data from one to two days of advance to predict solar flares. However, it is important to employ computational processing to select the best intervals that have the data that best distinguishes solar flares;

- Current methods perform binary forecasting. But, as the phenomenon is actually multi-class, solar flare methods must provide this type of forecasting.

1.2 Problem definition

The problem we deal with in this thesis is how to design a solar flare forecasting method that handles the identified open issues using Data Mining.

Solar flare forecasting is the task of predicting which solar flare classes may occur, given the data collected before a certain forecasting horizon. Figure 1.1 shows an illustration of the forecasting definition. The data collected in a period of size c is used as input of the forecasting model. The time unit size is given in days in this thesis. In the figure, the cell named *Jump* (of size j) is the interval between the last data collected and the period of the forecasting result. The Forecasting Horizon is the interval (of size f) ahead the *Jump* period that the forecasting method is able to perform the predictions. For example, consider c and f set as one day and j set as zero. In this way, we hypothetically collect solar data on a certain Wednesday to be used by the forecasting method that performs the forecasting of the following Thursday (as the forecasting horizon is actually the next day).

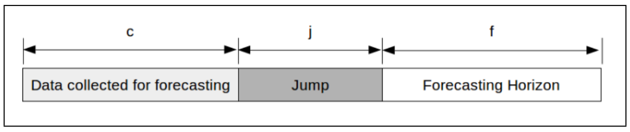


Figure 1.1: Illustration of Forecasting definition

Solar flare forecasting methods may use **Data Mining** (DM) techniques in order to produce the prediction model. This process is divided in the following steps (MAIMON; ROKACH, 2010):

- Establishment of the DM goals;
- Data gathering;
- Data preprocessing: data transformation and cleaning, selection of the DM task (classification, regression or clustering) and algorithms;
- Evaluation of the results;
- Deployment of the learning model in operational environment.

According to the open issues and the DM steps, we mapped the computational challenges that we faced in this thesis.

The first main challenge was to handle the time series historical evolution in the solar flare forecasting process. The domain specialist from INPE (National Institute for Space Research) who supported our project stated that the analysis of solar time series of two days before the forecasting horizon is reasonable to be considered as a training instance. Original solar time series comes in the form of a continuous record of values sampled in regular intervals. As we may need about two days of data to distinguish the solar flare forecasting, we have to transform the original continuous time series in a set of sub-series. Each of them should contain data collected in two days and be labeled with the events occurred in the forecasting horizon. Hence, our dataset is composed of instances of labeled sub-series used in the forecasting method. Thus, this first challenge leads us to a *time series classification context*, so that we incorporated the historical evolution of solar data into the designed method.

The second challenge is that the datasets used in the solar flare forecasting process are highly imbalanced (LEE et al., 2012). In this situation, there are much more instances belonging to a given class than the others. The classification method tends to classify the testing instance with the label of the majority class. Two problems, related to this challenge are cited in Galar et al. (2012):

- Imbalanced data problem: insufficient instances belonging to the minority class may incur in performance problems;
- Non-separable data problem: the difficulties to distinguish different classes due to instances of similar feature values in this context.

In the literature, several approaches were developed to deal with those problems. For this purpose, these approaches may modify the dataset distribution, the classification algorithm or also implement a mixture of both strategies (KRAWCZYK, 2016). Although these approaches are interesting, they do not handle all the problems listed above. Thus, we had to design alternatives to overcome this challenge.

The third challenge is the need to distinguish events classified in the boundary of two solar flare classes. For this purpose, no work in the literature was found. To handle this, we proposed a multi-label approach that employes Ensemble of Classifiers (EC). As EC methods aggregate results from several base classifiers, it is possible to extend the approaches to perform a multi-label approach.

Finally, as mentioned above, the physics regarding solar flares are not fully understood. Consequently, another problem to be faced is the selection of the features that best distinguish solar flares.

In summary, we faced the problem of solar flare forecasting by performing a multi-class and multi-label classification of imbalanced time series of solar features taking into account the astrophysicist's (domain specialist) knowledge.

1.3 Research questions

Based on the identified open issues, some research questions arised:

1. Can time series classification methods be used to forecast solar flares taking into account the historical evolution?
2. How to forecast solar flares in a multi-class and multi-label manner within a highly imbalanced dataset?
3. How to deal with possible ambiguity of adjacent classes in the solar flare forecasting process?

4. Which information of the astrophysicist, should the forecasting method use? When (in the method) should we use such information?

1.4 Hypothesis

From the open issues of current solar flare forecasting methods, we formulated our hypothesis:

A new method to classify time series that considers the data evolution, and employs balancing techniques and Ensemble of Classifiers will improve the solar flare forecasting state-of-the-art.

1.4.1 Validation criteria of the project hypothesis

The thesis was carried out with the support of a specialist from INPE. During the project, he disseminated the knowledge and practices of his research group so that the fundamentals of this thesis could be established. It was also discussed the metrics that should be used to validate the method as well as the aimed results with him.

In the forecasting process of this domain, a false alarm is less costly than a fault, because an unidentified high solar flare may have a severe and costly impact. As our problem is multi-class and multi-label and we use time series classification methods, we based the validation in metrics related to this scenario. Though classification metrics were developed for binary classification, we used customized variations based on Branco, Torgo e Ribeiro (2016). Among the main metrics, we find *recall* or *True Positive Rate (TPR)*, *True Negative Rate (TNR)*, *accuracy*, *precision*, *F-measure* and *error*. All these metrics have their relevance, but when considering the specificities of our domain, *recall* and *error* for each class were considered the most representative. Thus, the thresholds established to validate the hypothesis considered the specialist needs and is presented next:

- Minimum *recall* for each class: 70%;
- Maximum *error* for each class: 30%;
- Forecast horizon: one day;
- The maximum length of historical data: eleven years (one Solar Activity Cycle);

1.5 Project goals

The following general and specific goals were established.

1.5.1 General goal

The general goal of this project was to develop a solar flare forecasting method that deals with the open issues identified in the literature. The developed forecasting method handles the evolution of the time series of X-ray flux, and the magnetogram vector using time series classifiers, dataset balancing techniques and Ensemble of Classifiers.

1.5.2 Specific Goals

In order to achieve the proposed general goal, the following specific goals were defined:

1. Design of the pre-processing step that prepares the raw data to be used as input of the forecasting method;
2. Design of the pre-processing step that splits the full dataset into subsets balanced according to each class of solar flare;
3. Design of the pre-processing step that allows data mining classifiers to forecast future events through the historical evolution of time series;
4. Design of the forecasting task that is fed with the generated balanced datasets to produce forecasting results for each subset;
5. Design of how to produce the multi-label forecasting results;
6. Selection of the solar features to be used in the forecasting method.

Each specific goal builds a module, so each goal was achieved when those modules were implemented and tested against its requirements. The validation of the full method was made using the metrics defined for the hypothesis validation presented in Section 1.4.1.

1.6 Scientific methods and materials

This section describes the scientific methods used in this research project. Then, the materials used to develop, validate and deploy this project are described.

1.6.1 Scientific methods

The scientific method of this research project was based on Wazlawick (2009) and consisted of:

- Review of related works;
- Identification of the open issues in the literature;
- Survey of the needs of INPE's space weather program;
- Formulation of the project goal and hypothesis;
- Identification of the hypothesis validation criteria;
- Development of a model that integrates existing techniques and new methods to perform solar flare forecasting;
- Execution of the experiments to validate the hypothesis.

The requirements and validation of the project were carried out with the domain specialist (INPE astrophysicist) through:

- Meetings
- Emails
- Questionnaires

1.6.2 Materials

As shown in Table 1.2 We collected data from three primary sources which are freely available for research purposes.

Table 1.2: Data source

X-ray time series	**Source**	**Website**
X-ray time series	Website of National Oceanic and Atmospheric Administration - Space Weather Prediction Center - Operating Unit of the U.S. Dept of Commerce	https://satdat.ngdc.noaa.gov/sem/goes/data/new_avg
Magnetogram vector	Joint Science Operations Center (JSOC) - Stanford Solar Center	http://jsoc.stanford.edu/
Solar Event List	Website of National Oceanic and Atmospheric Administration - Space Weather Prediction Center - Operating Unit of the U.S. Dept of Commerce	ftp://ftp.swpc.noaa.gov/pub/warehouse/

Additionally, we used the softwares and programming languages presented in Table 1.3 during the development of the thesis work.

Table 1.3: Software and programming languages used in this thesis work

Type of software and programming languages	Name	Version
Operating System	Linux	16.04
Database Management System	Postgres	10.4
Programming Language	Java	8
Programming Language	C	-
Programming Language	CUDA C	Toolkit 8.0

The development was performed at the GAPIS (Grupo de Arquitetura e Processamento de Imagens e Sinais") laboratory of the Department of Computing at Federal University of São Carlos (UFSCar), using its infrastructure of hardware and software. The hardware configuration available for the project is listed in Table 1.4

Table 1.4: Infrastructure available for the project

Environment	Configuration
GAPIS lab - machine-1	CPU: Intel i5 RAM: 8 GB Graphics Processing Unit: Geforce 960X, with 1024 CUDA cores (GPUs) and 2GB of memory. Operating System: Windows 10
GAPIS lab - machine-2	CPU: Intel Core i5 RAM: 8 GB Disk: 1 TB Operating System: Linux Ubuntu 16.04

1.7 Main contributions

The main contributions of this work were the development of two solar flare forecasting methods: SeMiner and ECID.

SeMiner was developed and tested using a number of strategies aiming to improve parameters such as accuracy, true positive and true negative rates. This method uses one solar feature in order to perform the forecasting. More specifically, this method deals with a crucial step in the process of solar flare forecasting: the data preprocessing, using the knowledge of the domain specialist. Other distinguishing characteristics (compared to other forecasting methods) are: the historical evolution of the solar time series was considered to produce the forecasting model; and, we identified the most significant time intervals to be considered in the forecasting. In the scope of our experiments, the best time interval was within two days for *current window* and comprised both, the initial and end periods of the first day, and the last 16 hours of the second day. Finally, performance optimization was implemented in SeMiner. The core module of SeMiner, the SS algorithm, was parallelized using CUDA, targeting parallel execution in GPUs. The strategy showed to be 4.36 times faster than its pure C sequential version.

We also developed a method called ECID (Ensemble of classifiers for imbalanced datasets). This method provides a tool to support the specialist decision in the solar flare forecasting task. It gives multi-class and multi-label forecasting. This strategy differentiates from other forecasting methods because its result indicates the classes most probably to occur in a forecasting horizon. This result also supports the domain specialist when it is needed to differentiate possible events that occur in the boundary of higher events. This was possible, because, for each solar flare class, ECID employs a stratified random sampling for the training of base inducers, strengthen its sensitivity. Using a modified bootstrap approach, its aggregation method combines the inducers results enabling multi-class forecasting, which can also be multi-label in case of adjacent classes. Finally, it is able to handle more than one solar feature to provide its results.

1.8 Text structure

This text is structured in the following manner:

Chapter 2 presents the theoretical concepts needed for the development of the project. So, it describes the solar flare background needed for the comprehension of the problem. This chapter also presents classification methods used as base inducers in our Ensemble of Classifiers. Next, it presents some time series classification approaches and techniques used to minimize the problems related to imbalanced datasets in classification methods. Finally, metrics applied in the validation of forecasting methods are described.

In Chapter 3, related works regarding computational techniques used in time series classification of imbalanced datasets are listed and discussed. Additionally, the characteristics and results of solar flare forecasting methods found in literature are described and analyzed.

Chapter 4 describes the first method developed in this work, called SeMiner (Sequence Miner). It shows the conceptual basis, justifications, and explanation of the characteristics of SeMiner.

Chapter 5 describes the second method developed in this doctorate work, called ECID (Ensemble of Classifiers for Imbalanced Dataset). It shows the conceptual basis, justifications, and explanation of the characteristics of ECID.

Chapter 6 describes the experiments and results performed with SeMiner and its parallelized version. It also presents the experiments and results obtained with ECID. Finally, it discusses the validation of the thesis-hypothesis based on the result of the experiments.

Finally, Chapter 7 details the most relevant contributions, the scientific publications and the future works.

II BACKGROUND AND RELATED WORK

Chapter 2

BACKGROUND

In this chapter, we present the concepts that formed the basis for the development of this thesis: solar flares, data mining classification, time series classification techniques to handle imbalanced datasets, and validation metrics for forecasting methods.

The solar flare context requires several concepts that may be understood in order to design a forecasting method, thus in this chapter we begin describing the problem of solar flares and an important concept, which is the X-ray background level. Next, the concept of active regions is presented. Also, solar data is provided from several sources, formats and types. As mentioned, the physics of this event is not fully understood, so that current research may select the features which best distinguishes the event. Hence, we provide a description of the main solar features characteristics used in current forecasting methods.

Our solar flare forecasting method was designed using a time series classification approach that used techniques for handling imbalanced data sets. These time series classification methods usually use traditional classifiers, also shown in this chapter. Afterwards, we present the main approaches for time series classification methods and how to handle imbalanced datasets. These methods inspired us in the formulation of our method.

Finally, we present the main metrics used in the validation of a data mining classification task and how they may fit in the multi-class and multi-label approach.

2.1 The problem of solar flares

Solar flares are sudden releases of large amounts of energy (10^{25}-10^{32} erg) from active regions of the solar atmosphere (HOLMAN, 2006), and it was first observed on September of 1859 in the Carrington Event (CARRINGTON, 1859). These phenomena last from tens of seconds to a few hours and release energy in the form of radio waves, X-rays or even gamma rays. It can interfere on High Frequency (HF) and Very High Frequency (VHF) radio communication, since the terrestrial Ionosphere is disturbed by the strong presence of X-rays, causing the reflection of HF and VHF waves at certain frequencies and absorbing others (TSURUTANI et al., 2009). The flares can also interfere in satellite communication since it uses high-frequency signals (BASU et al., 2010).

We can observe the Active Region (NOAA AR 2297) in Figure 2.1 located in the yellowish/bluish parts close to the center of the solar disk. The abrupt release of energy in these regions is known as solar flares (HOLMAN, 2006). In Figures 2.2 and 2.3, it is possible to observe the instants before and after of the solar flare classified as X that occurred in March of 2015. Flares of this class are considered the most intense ones and may have the highest impacts on Earth's devices.

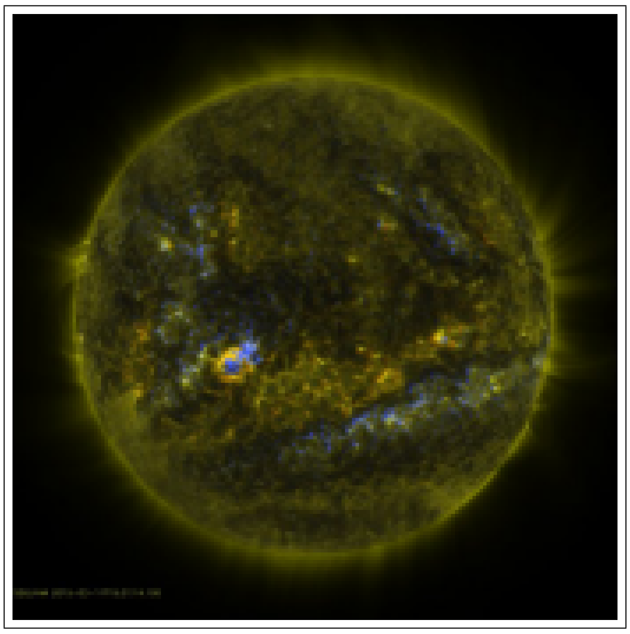


Figure 2.1: Image recorded on March 11th, 2015 at 15:37 UT (Universal Time), showing the active region AR 2297 that produced the X-class solar flare observed on that day.

The X2.1 solar flare, produced on March 11, 2015, by the AR 2297 shown in Figure 2.1, started at 16:11 UT, reached its peak at 16:22 UT and finished at 16:29 UT. Figure 2.2 was extracted at 16:07, shortly before the event. Figure 2.3 was obtained at 16:37, shortly after the event. Thus, one can observe the evolution of the Active Region (in blue and yellow highlighted aspects) from where the flare originated.

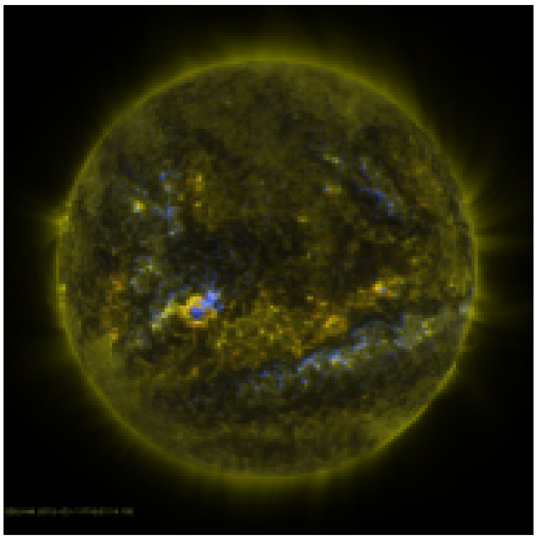


Figure 2.2: Image recorded of the solar disk on March 11th, 2015 shortly before (16:07 UT) the observed X-class solar flare.

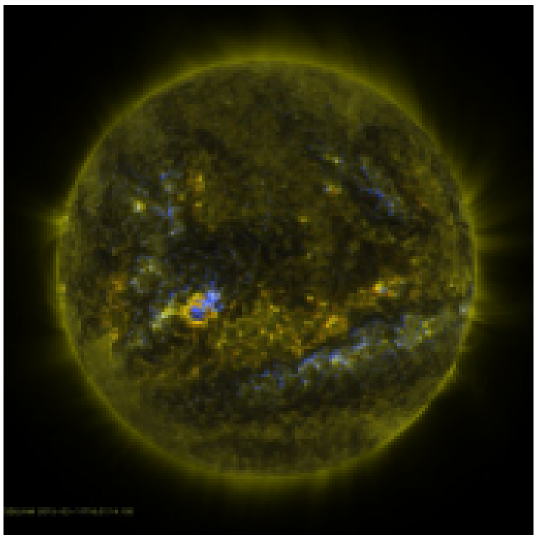


Figure 2.3: Image recorded of the solar disk on March 11th, 2015 shortly after (16:37 UT) the observed X-class solar flare.

Though we can find several computational methods in the literature for the forecasting of solar flares, this area of research still needs improvement, because astrophysicists do not have the full understanding of the physical phenomenon behind solar flares yet (PRIEST; FORBES, 2002; RABOONIK et al., 2016). Consequently, it is of utmost importance to develop accurate, operational and optimized forecasting models for solar flares.

Thus, this thesis developed a method of solar flare forecasting based on the concepts and methods of the area of *Data Mining*. These tools were selected and improved taking into account the characteristics of this application domain, the events, and the solar data. For this, the direction and feedback of the domain specialist were fundamental to understand the various concepts involved in this phenomenon. An essential idea in this scenario is to differentiate the X-ray background level from the solar flare, as explained in the following.

2.1.1 The Difference between solar flares and X-ray background levels

Solar flares, as already mentioned, are a sudden release of energy from Sun that may impact on Earth's environment and/or electronic devices. According to the SWPC website a flare is identified by a sequence of 4 minutes presenting a steep monotonic increase in the 1-8 Angstrom solar X-ray flux that is recorded, independent of its previous flux intensity level. The maximum of the flare corresponds to the peak level observed in the X-ray flux, which determines its intensity. The background level can be obtained by the average X-ray flux calculated from 20% of the lowest flux values in an interval of 8-10 hour.

Thus, it is essential to emphasize the concept of X-ray background level, since the release of a high-intensity X-ray flux is not enough to conclude that the Sun produced a solar flare (WEBSITENOAA, 2017; DRIEL-GESZTELYI LIDIA; GREEN, 2015).

To check if an active region emitted a solar flare, we have to look for an abrupt variation of X-ray emission. For example, if the Sun releases 1E-06 W/m^2 of X-ray flux and this intensity is increasing smoothly, then we can say that the X-ray background level is of level C. It can reach gradually to 1E-05 W/m^2 even though the active region didn't produce a solar flare (WEBSITENOAA, 2017). In this case, instead of the occurrence of a solar flare, there may have been occurred an increase in the X-ray background level.

The abrupt variation of energy characterizes solar flares. Thus, if the X-ray suddenly rises from 1E-07 to 1E-05 W/m^2, than we can say that an Active Region produced a solar flare (WEBSITENOAA, 2017).

Figure 2.4 presents an example of the referred difference. Next to date Jun-11 we can observe a sudden increase of the X-ray background level (red curve) from level A to B, reported as a solar flare of class B1.5. Observing day Jun-12, there was a smooth increase from level A to level B, but, in this case, it was not reported a solar flare because the increase was not abrupt.

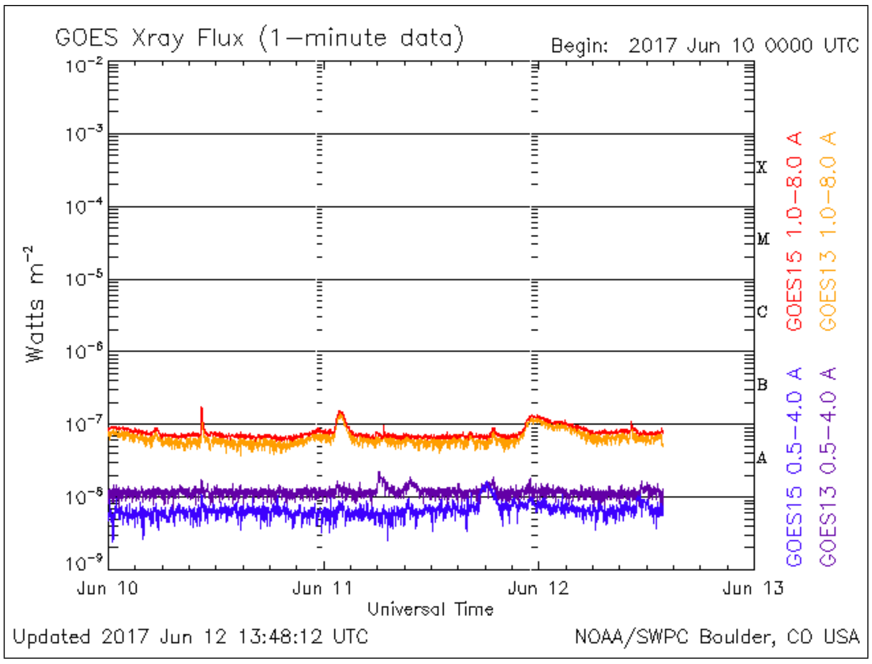


Figure 2.4: Solar flare x Background level example

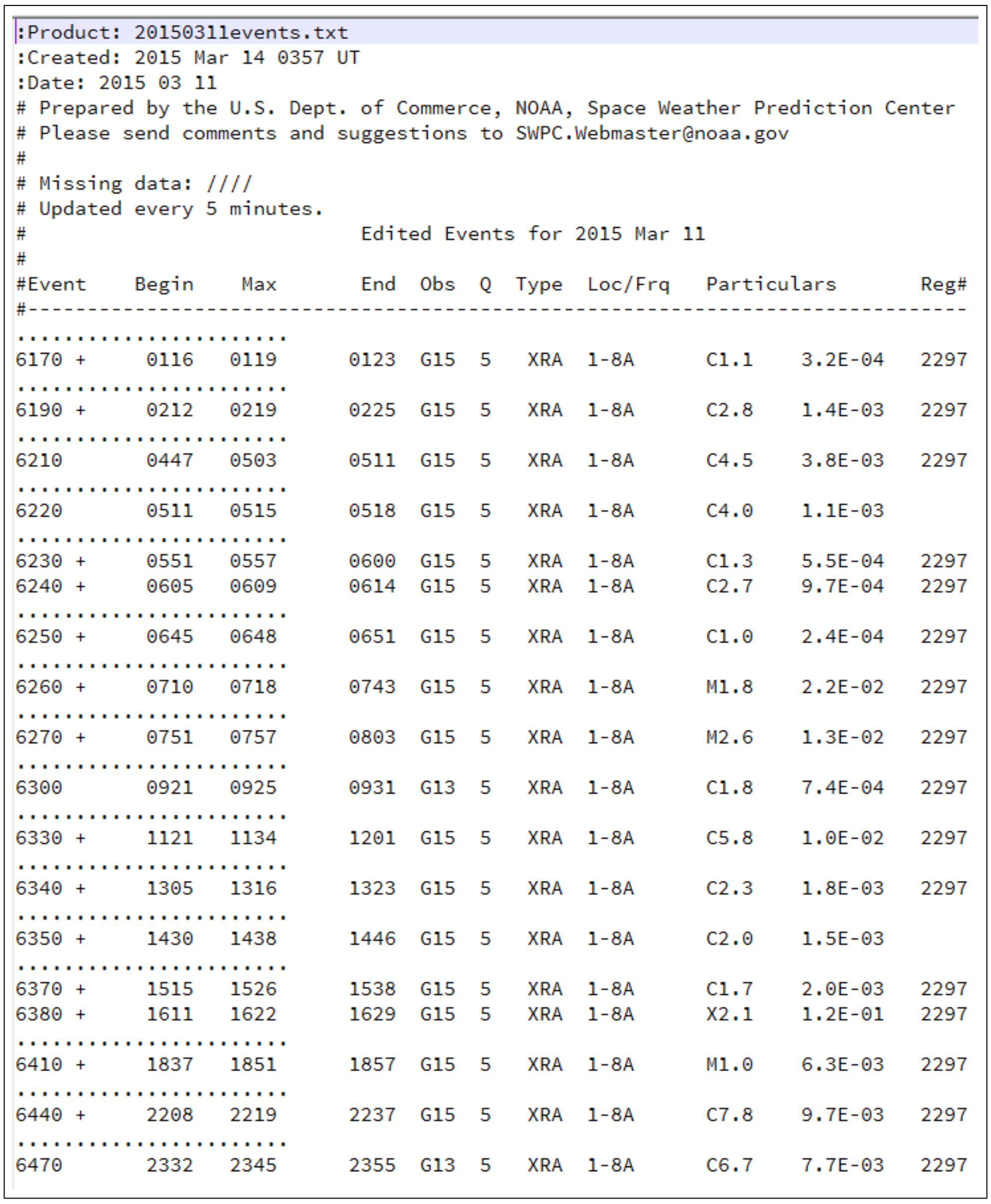
2.1.2 Active Region

At the photosphere, an active region is in the form of bipolar/multipolar magnetic fields showed as dark sunspot groups. These field lines is in the form of arcades. In a simplified description, active regions is the location in the solar atmosphere - mainly high chromosphere and low corona - where a solar flare occurs or can occur. This region is dynamic and occasionally the field lines can reconnect and abruptly release the large amount of stored magnetic energy generating a solar flare (DRIEL-GESZTELYI LIDIA; GREEN, 2015).

Driel-Gesztelyi Lidia; Green (2015) gives the following formal definition for Active Regions:

Active regions are the totality of observable phenomena in a 3D volume represented by the extension of the magnetic field from the Photosphere to the Corona, revealed by emissions over a wide range of wavelengths from radio to X-rays and γ-rays (only during flares) accompanying and following the emergence of strong twisted magnetic flux (kg, $\geq$ 1020 Mx) through the Photosphere into the Chromosphere and Corona.

Each Active Region (AR) has a unique identification, called AR number. As the AR has an intense magnetic activity, because it is composed of magnetic fields and lines, a set of magnetic properties, like its size and strength may be sampled using specific instruments on board of satellites. Thus, there is a time series of values of each magnetic feature for each AR number. Also, NOAA/SWPC (NOAAEVENTREPORT, 2019) provides reports that maps AR with its solar flares. An issue regarding this report is that sometimes it does not link the solar flares with its Active Region. Thus, the development of a forecasting model must take this gap into account. Figure 2.5 shows an example of the NOAA/SWPC (NOAAEVENTREPORT, 2019) reports.



```
:Product: 20150311events.txt
:Created: 2015 Mar 14 0357 UT
:Date: 2015 03 11
# Prepared by the U.S. Dept. of Commerce, NOAA, Space Weather Prediction Center
# Please send comments and suggestions to SWPC.Webmaster@noaa.gov
#
# Missing data: ////
# Updated every 5 minutes.
#                            Edited Events for 2015 Mar 11
#
#Event    Begin    Max       End  Obs  Q  Type  Loc/Frq   Particulars       Reg#
#-------------------------------------------------------------------------------
.......................
6170 +     0116   0119      0123  G15  5   XRA  1-8A      C1.1    3.2E-04   2297
.......................
6190 +     0212   0219      0225  G15  5   XRA  1-8A      C2.8    1.4E-03   2297
.......................
6210       0447   0503      0511  G15  5   XRA  1-8A      C4.5    3.8E-03   2297
.......................
6220       0511   0515      0518  G15  5   XRA  1-8A      C4.0    1.1E-03
.......................
6230 +     0551   0557      0600  G15  5   XRA  1-8A      C1.3    5.5E-04   2297
6240 +     0605   0609      0614  G15  5   XRA  1-8A      C2.7    9.7E-04   2297
.......................
6250 +     0645   0648      0651  G15  5   XRA  1-8A      C1.0    2.4E-04   2297
.......................
6260 +     0710   0718      0743  G15  5   XRA  1-8A      M1.8    2.2E-02   2297
.......................
6270 +     0751   0757      0803  G15  5   XRA  1-8A      M2.6    1.3E-02   2297
.......................
6300       0921   0925      0931  G13  5   XRA  1-8A      C1.8    7.4E-04   2297
.......................
6330 +     1121   1134      1201  G15  5   XRA  1-8A      C5.8    1.0E-02   2297
.......................
6340 +     1305   1316      1323  G15  5   XRA  1-8A      C2.3    1.8E-03   2297
.......................
6350 +     1430   1438      1446  G15  5   XRA  1-8A      C2.0    1.5E-03
.......................
6370 +     1515   1526      1538  G15  5   XRA  1-8A      C1.7    2.0E-03   2297
6380 +     1611   1622      1629  G15  5   XRA  1-8A      X2.1    1.2E-01   2297
.......................
6410 +     1837   1851      1857  G15  5   XRA  1-8A      M1.0    6.3E-03   2297
.......................
6440 +     2208   2219      2237  G15  5   XRA  1-8A      C7.8    9.7E-03   2297
.......................
6470       2332   2345      2355  G13  5   XRA  1-8A      C6.7    7.7E-03   2297
```

Figure 2.5: Example of NOAA/SWPC Solar Event report

Note, in Figure 2.5, that event #6380 shows the X2.1 solar flare occurred at AR #2297. This information is of great importance to our work because it allows to map a solar flare with its magnetic features.

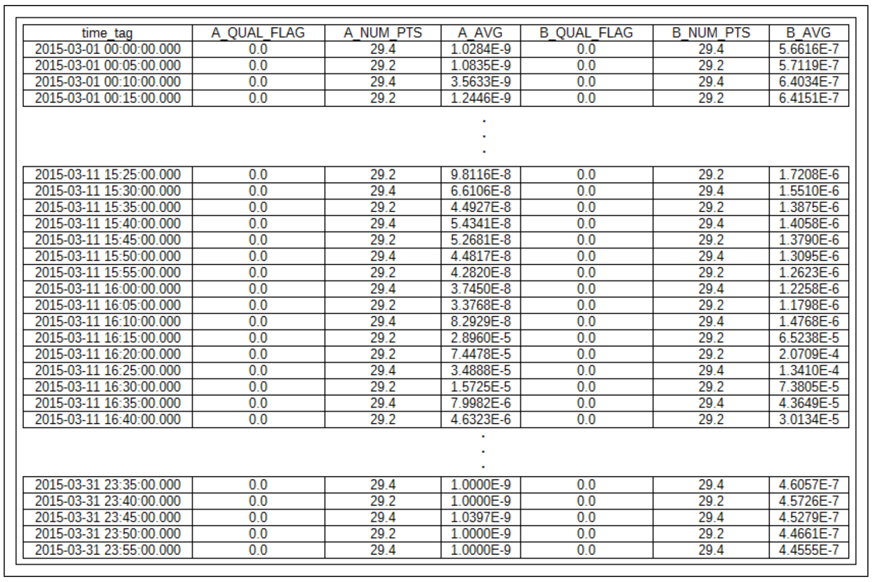
2.1.3 Solar features

A question that arises is: Which solar features indicate that a possible solar flare is in course to occur?

There is no agreement about which features influence on solar flares (PRIEST; FORBES, 2002). But, many studies suggest several features that tightly indicate the occurrence of this phenomenon. One of the most used is the X-ray intensity emitted by Sun and recorded through sensors located at GOES satellite. Another set of features are the magnetic properties of Active Regions. Mainly two instruments collect these features: SOHO/MDI and SDO/HMI. Magnetic features are extracted from images called magnetograms. These images establish an integer value for each pixel. This value is proportional to the magnetic activity of that location. With this image, it is possible to extract the size of magnetic lines, their strength, and many other properties. Also, some works use the topology aspects of sunspots in the forecasting process. Finally, few works use the Chromosphere physical properties to forecast solar flares.

We present, in following, examples of solar feature types and their sources:

- Soft X-ray flux emitted by Sun: this data is relevant to solar flare forecasting because it establishes a direct relation between X-ray and the impacts it may produce. It consists of a time series with a sample period of 1 or 5 minutes and the intensity of the X-ray flux within 1-8 Angstroms passband is recorded by a sensor located at GOES satellite records. Table 1.1 shows the relation between X-ray intensity, solar flare classification, and its possible impacts. Figure 2.6 shows an example of this time series.

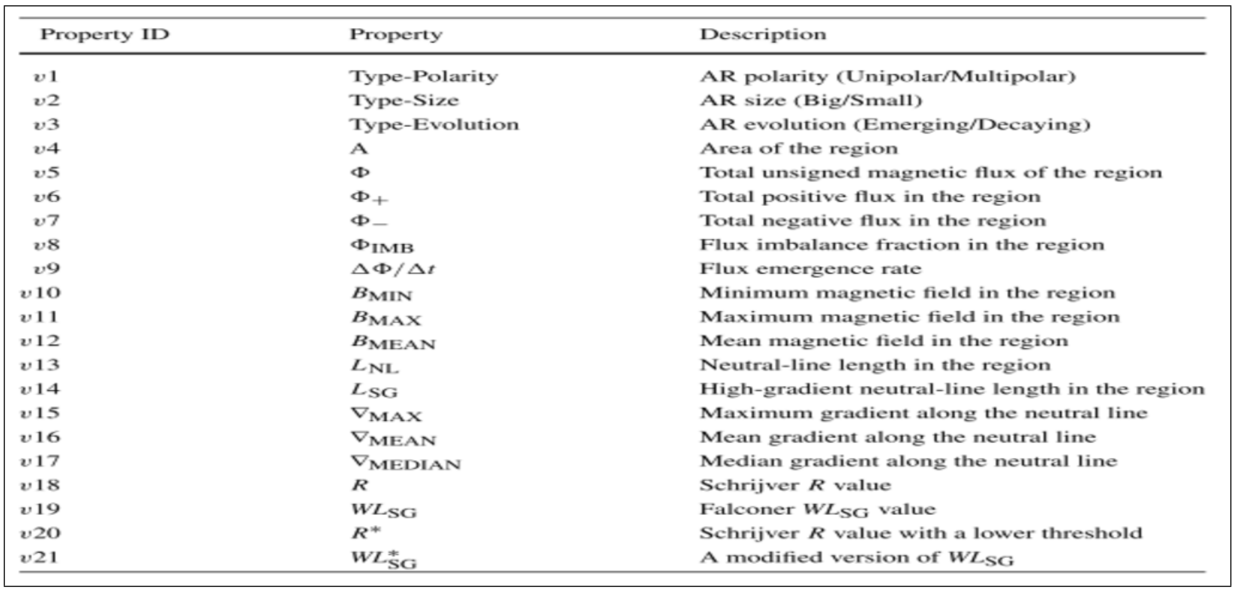


time_tag	A_QUAL_FLAG	A_NUM_PTS	A_AVG	B_QUAL_FLAG	B_NUM_PTS	B_AVG
2015-03-01 00:00:00.000	0.0	29.4	1.0284E-9	0.0	29.4	5.6616E-7
2015-03-01 00:05:00.000	0.0	29.2	1.0835E-9	0.0	29.2	5.7119E-7
2015-03-01 00:10:00.000	0.0	29.4	3.5633E-9	0.0	29.4	6.4034E-7
2015-03-01 00:15:00.000	0.0	29.2	1.2446E-9	0.0	29.2	6.4151E-7
			⋮			
2015-03-11 15:25:00.000	0.0	29.2	9.8116E-8	0.0	29.2	1.7208E-6
2015-03-11 15:30:00.000	0.0	29.4	6.6106E-8	0.0	29.4	1.5510E-6
2015-03-11 15:35:00.000	0.0	29.2	4.4927E-8	0.0	29.2	1.3875E-6
2015-03-11 15:40:00.000	0.0	29.4	5.4341E-8	0.0	29.4	1.4058E-6
2015-03-11 15:45:00.000	0.0	29.2	5.2681E-8	0.0	29.2	1.3790E-6
2015-03-11 15:50:00.000	0.0	29.4	4.4817E-8	0.0	29.4	1.3095E-6
2015-03-11 15:55:00.000	0.0	29.2	4.2820E-8	0.0	29.2	1.2623E-6
2015-03-11 16:00:00.000	0.0	29.4	3.7450E-8	0.0	29.4	1.2258E-6
2015-03-11 16:05:00.000	0.0	29.2	3.3768E-8	0.0	29.2	1.1798E-6
2015-03-11 16:10:00.000	0.0	29.4	8.2929E-8	0.0	29.4	1.4768E-6
2015-03-11 16:15:00.000	0.0	29.2	2.8960E-5	0.0	29.2	6.5238E-5
2015-03-11 16:20:00.000	0.0	29.2	7.4478E-5	0.0	29.2	2.0709E-4
2015-03-11 16:25:00.000	0.0	29.4	3.4888E-5	0.0	29.4	1.3410E-4
2015-03-11 16:30:00.000	0.0	29.2	1.5725E-5	0.0	29.2	7.3805E-5
2015-03-11 16:35:00.000	0.0	29.4	7.9982E-6	0.0	29.4	4.3649E-5
2015-03-11 16:40:00.000	0.0	29.2	4.6323E-6	0.0	29.2	3.0134E-5
			⋮			
2015-03-31 23:35:00.000	0.0	29.4	1.0000E-9	0.0	29.4	4.6057E-7
2015-03-31 23:40:00.000	0.0	29.2	1.0000E-9	0.0	29.2	4.5726E-7
2015-03-31 23:45:00.000	0.0	29.4	1.0397E-9	0.0	29.4	4.5279E-7
2015-03-31 23:50:00.000	0.0	29.2	1.0000E-9	0.0	29.2	4.4661E-7
2015-03-31 23:55:00.000	0.0	29.4	1.0000E-9	0.0	29.4	4.4555E-7

Figure 2.6: Example of X-ray time series

Figure 2.6 shows the file obtained from NOAA/SWPC FTP site, which stores a daily set of X-ray time series ($https://satdat.ngdc.noaa.gov/sem/goes/data/new\_avg/$). This X-ray time series example has a sample period of 5 minutes. There are two columns that inform the data quality of each sample (A_QUAL_FLAG and B_QUAL_FLAG). Also, there are two auxiliary fields (A_NUM_PTS and B_NUM_PTS) and, finally and most important, two attributes containing X-ray measured within 0.05-0.4 nm passband, and the ones measured within 0.1-0.8 nm passband (A_AVG and B_AVG). B_AVG is the time series that has a direct relation to the solar flare classification.

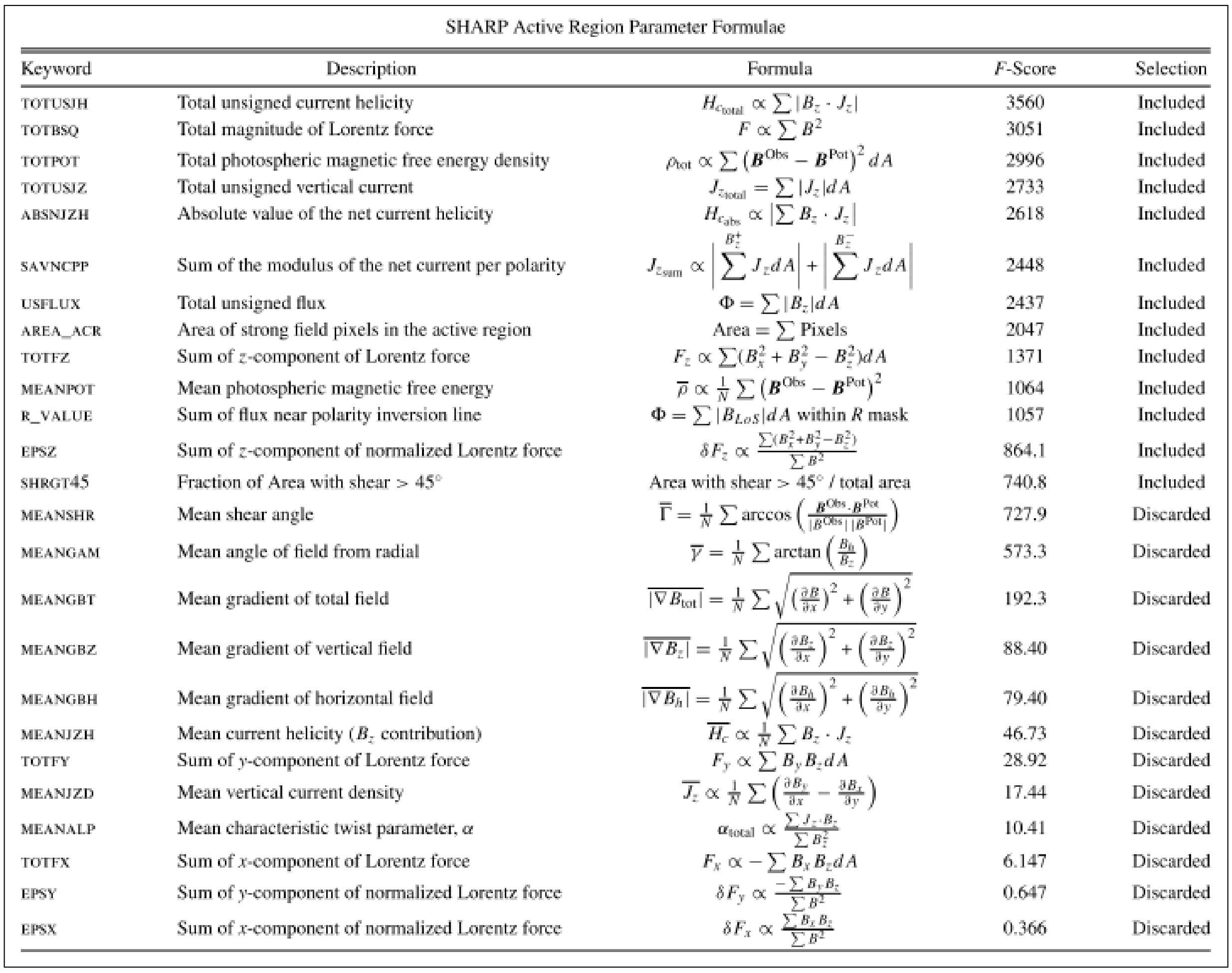
- Magnetic features obtained from SOHO/MDI: these images are called Line-of-sight magnetogram, and it may provide the magnetic field strengths, size, area, etc, found in the active regions. The instrument that records the magnetogram is called *Michelson Doppler Imager (MDI)*, and it is on-board of the *Solar and Heliospheric Observatory (SOHO) spacecraft (SOHO/MDI)*. A list of the possible extracted features from this instrument is shown in (AHMED et al., 2013). This work applies the magnetic features shown in Figure 2.7 in the forecasting process (the physic's meaning of each property is out of the scope of this work).



Property ID	Property	Description
$v1$	Type-Polarity	AR polarity (Unipolar/Multipolar)
$v2$	Type-Size	AR size (Big/Small)
$v3$	Type-Evolution	AR evolution (Emerging/Decaying)
$v4$	A	Area of the region
$v5$	Φ	Total unsigned magnetic flux of the region
$v6$	Φ_+	Total positive flux in the region
$v7$	Φ_-	Total negative flux in the region
$v8$	Φ_{IMB}	Flux imbalance fraction in the region
$v9$	$\Delta\Phi/\Delta t$	Flux emergence rate
$v10$	B_{MIN}	Minimum magnetic field in the region
$v11$	B_{MAX}	Maximum magnetic field in the region
$v12$	B_{MEAN}	Mean magnetic field in the region
$v13$	L_{NL}	Neutral-line length in the region
$v14$	L_{SG}	High-gradient neutral-line length in the region
$v15$	∇_{MAX}	Maximum gradient along the neutral line
$v16$	∇_{MEAN}	Mean gradient along the neutral line
$v17$	∇_{MEDIAN}	Median gradient along the neutral line
$v18$	R	Schrijver R value
$v19$	WL_{SG}	Falconer WL_{SG} value
$v20$	R^*	Schrijver R value with a lower threshold
$v21$	WL^*_{SG}	A modified version of WL_{SG}

Figure 2.7: Line-of-sight magnetic features presented in (AHMED et al., 2013)

- Magnetic properties obtained from SDO/HMI: these images are also called Line-of-sight and vector magnetogram. A newer instrument, the *Helioseismic and Magnetic Imager (HMI)* on board of the *Solar Dynamics Observatory*, records these set of images. This space observatory substituted SOHO/MDI and can produce different types of images named line-of-sight and vector magnetogram. In Bobra e Couvidat (2015), it is extracted the magnetic features from this source to use in their solar flare forecasting method. Figure 2.8 shows the list of magnetic properties used in that work.

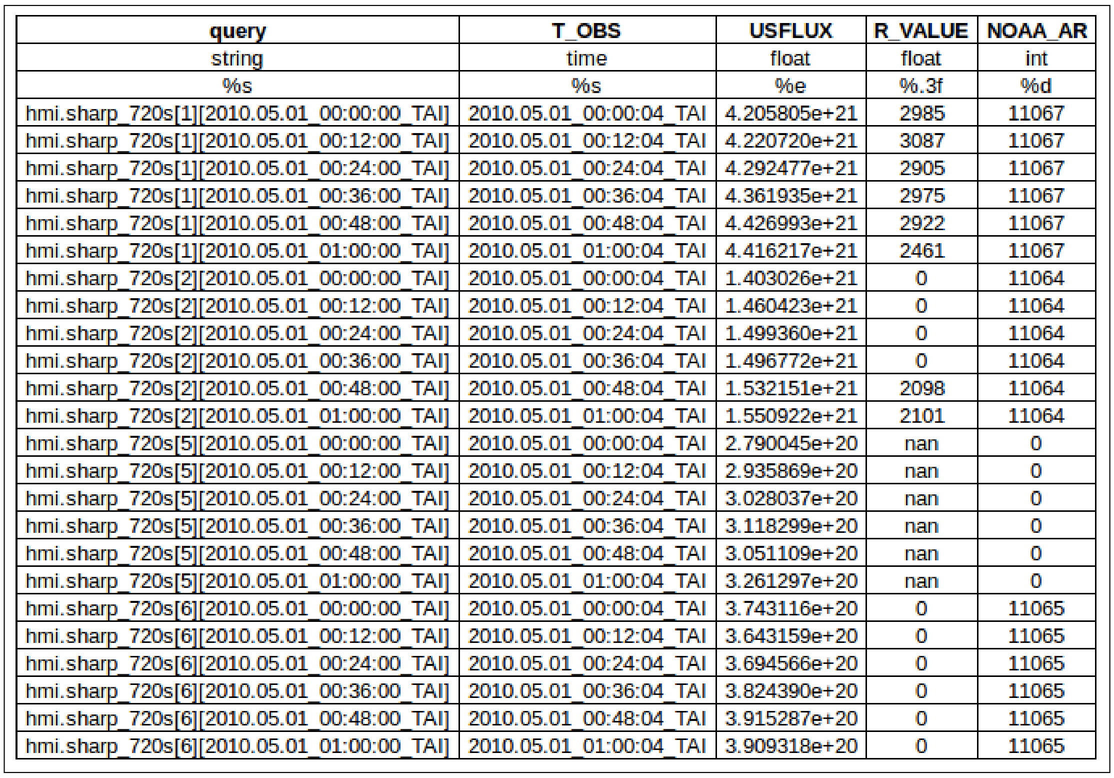


SHARP Active Region Parameter Formulae

Keyword	Description	Formula	F-Score	Selection
TOTUSJH	Total unsigned current helicity	$H_{c_{\text{total}}} \propto \sum \lvert B_z \cdot J_z \rvert$	3560	Included
TOTBSQ	Total magnitude of Lorentz force	$F \propto \sum B^2$	3051	Included
TOTPOT	Total photospheric magnetic free energy density	$\rho_{\text{tot}} \propto \sum \left(\boldsymbol{B}^{\text{Obs}} - \boldsymbol{B}^{\text{Pot}}\right)^2 dA$	2996	Included
TOTUSJZ	Total unsigned vertical current	$J_{z_{\text{total}}} = \sum \lvert J_z \rvert dA$	2733	Included
ABSNJZH	Absolute value of the net current helicity	$H_{c_{\text{abs}}} \propto \lvert \sum B_z \cdot J_z \rvert$	2618	Included
SAVNCPP	Sum of the modulus of the net current per polarity	$J_{z_{\text{sum}}} \propto \left\lvert \sum^{B_z^+} J_z dA \right\rvert + \left\lvert \sum^{B_z^-} J_z dA \right\rvert$	2448	Included
USFLUX	Total unsigned flux	$\Phi = \sum \lvert B_z \rvert dA$	2437	Included
AREA_ACR	Area of strong field pixels in the active region	$\text{Area} = \sum \text{Pixels}$	2047	Included
TOTFZ	Sum of z-component of Lorentz force	$F_z \propto \sum (B_x^2 + B_y^2 - B_z^2) dA$	1371	Included
MEANPOT	Mean photospheric magnetic free energy	$\overline{\rho} \propto \frac{1}{N} \sum \left(\boldsymbol{B}^{\text{Obs}} - \boldsymbol{B}^{\text{Pot}}\right)^2$	1064	Included
R_VALUE	Sum of flux near polarity inversion line	$\Phi = \sum \lvert B_{LoS} \rvert dA$ within R mask	1057	Included
EPSZ	Sum of z-component of normalized Lorentz force	$\delta F_z \propto \frac{\sum (B_x^2 + B_y^2 - B_z^2)}{\sum B^2}$	864.1	Included
SHRGT45	Fraction of Area with shear > 45°	Area with shear > 45° / total area	740.8	Included
MEANSHR	Mean shear angle	$\overline{\Gamma} = \frac{1}{N} \sum \arccos \left(\frac{\boldsymbol{B}^{\text{Obs}} \cdot \boldsymbol{B}^{\text{Pot}}}{\lvert \boldsymbol{B}^{\text{Obs}} \rvert \lvert \boldsymbol{B}^{\text{Pot}} \rvert} \right)$	727.9	Discarded
MEANGAM	Mean angle of field from radial	$\overline{\gamma} = \frac{1}{N} \sum \arctan \left(\frac{B_h}{B_z} \right)$	573.3	Discarded
MEANGBT	Mean gradient of total field	$\overline{\lvert \nabla B_{\text{tot}} \rvert} = \frac{1}{N} \sum \sqrt{\left(\frac{\partial B}{\partial x}\right)^2 + \left(\frac{\partial B}{\partial y}\right)^2}$	192.3	Discarded
MEANGBZ	Mean gradient of vertical field	$\overline{\lvert \nabla B_z \rvert} = \frac{1}{N} \sum \sqrt{\left(\frac{\partial B_z}{\partial x}\right)^2 + \left(\frac{\partial B_z}{\partial y}\right)^2}$	88.40	Discarded
MEANGBH	Mean gradient of horizontal field	$\overline{\lvert \nabla B_h \rvert} = \frac{1}{N} \sum \sqrt{\left(\frac{\partial B_h}{\partial x}\right)^2 + \left(\frac{\partial B_h}{\partial y}\right)^2}$	79.40	Discarded
MEANJZH	Mean current helicity (B_z contribution)	$\overline{H_c} \propto \frac{1}{N} \sum B_z \cdot J_z$	46.73	Discarded
TOTFY	Sum of y-component of Lorentz force	$F_y \propto \sum B_y B_z dA$	28.92	Discarded
MEANJZD	Mean vertical current density	$\overline{J_z} \propto \frac{1}{N} \sum \left(\frac{\partial B_y}{\partial x} - \frac{\partial B_x}{\partial y} \right)$	17.44	Discarded
MEANALP	Mean characteristic twist parameter, α	$\alpha_{\text{total}} \propto \frac{\sum J_z \cdot B_z}{\sum B_z^2}$	10.41	Discarded
TOTFX	Sum of x-component of Lorentz force	$F_x \propto -\sum B_x B_z dA$	6.147	Discarded
EPSY	Sum of y-component of normalized Lorentz force	$\delta F_y \propto \frac{-\sum B_y B_z}{\sum B^2}$	0.647	Discarded
EPSX	Sum of x-component of normalized Lorentz force	$\delta F_x \propto \frac{\sum B_x B_z}{\sum B^2}$	0.366	Discarded

Figure 2.8: Features extracted from vector magnetogram used in (BOBRA; COUVIDAT, 2015)

As SDO/HMI generates images, numerical values must be calculated to extract the time series of each magnetic property. The Joint Science Operations Center (JSOC) - Stanford Solar Center website provides a tool that calculates these numerical values for the magnetic field properties in almost real time. Figure 2.9 shows an example of those time series.



query	T_OBS	USFLUX	R_VALUE	NOAA_AR
string	time	float	float	int
%s	%s	%e	%.3f	%d
hmi.sharp_720s[1][2010.05.01_00:00:00_TAI]	2010.05.01_00:00:04_TAI	4.205805e+21	2985	11067
hmi.sharp_720s[1][2010.05.01_00:12:00_TAI]	2010.05.01_00:12:04_TAI	4.220720e+21	3087	11067
hmi.sharp_720s[1][2010.05.01_00:24:00_TAI]	2010.05.01_00:24:04_TAI	4.292477e+21	2905	11067
hmi.sharp_720s[1][2010.05.01_00:36:00_TAI]	2010.05.01_00:36:04_TAI	4.361935e+21	2975	11067
hmi.sharp_720s[1][2010.05.01_00:48:00_TAI]	2010.05.01_00:48:04_TAI	4.426993e+21	2922	11067
hmi.sharp_720s[1][2010.05.01_01:00:00_TAI]	2010.05.01_01:00:04_TAI	4.416217e+21	2461	11067
hmi.sharp_720s[2][2010.05.01_00:00:00_TAI]	2010.05.01_00:00:04_TAI	1.403026e+21	0	11064
hmi.sharp_720s[2][2010.05.01_00:12:00_TAI]	2010.05.01_00:12:04_TAI	1.460423e+21	0	11064
hmi.sharp_720s[2][2010.05.01_00:24:00_TAI]	2010.05.01_00:24:04_TAI	1.499360e+21	0	11064
hmi.sharp_720s[2][2010.05.01_00:36:00_TAI]	2010.05.01_00:36:04_TAI	1.496772e+21	0	11064
hmi.sharp_720s[2][2010.05.01_00:48:00_TAI]	2010.05.01_00:48:04_TAI	1.532151e+21	2098	11064
hmi.sharp_720s[2][2010.05.01_01:00:00_TAI]	2010.05.01_01:00:04_TAI	1.550922e+21	2101	11064
hmi.sharp_720s[5][2010.05.01_00:00:00_TAI]	2010.05.01_00:00:04_TAI	2.790045e+20	nan	0
hmi.sharp_720s[5][2010.05.01_00:12:00_TAI]	2010.05.01_00:12:04_TAI	2.935869e+20	nan	0
hmi.sharp_720s[5][2010.05.01_00:24:00_TAI]	2010.05.01_00:24:04_TAI	3.028037e+20	nan	0
hmi.sharp_720s[5][2010.05.01_00:36:00_TAI]	2010.05.01_00:36:04_TAI	3.118299e+20	nan	0
hmi.sharp_720s[5][2010.05.01_00:48:00_TAI]	2010.05.01_00:48:04_TAI	3.051109e+20	nan	0
hmi.sharp_720s[5][2010.05.01_01:00:00_TAI]	2010.05.01_01:00:04_TAI	3.261297e+20	nan	0
hmi.sharp_720s[6][2010.05.01_00:00:00_TAI]	2010.05.01_00:00:04_TAI	3.743116e+20	0	11065
hmi.sharp_720s[6][2010.05.01_00:12:00_TAI]	2010.05.01_00:12:04_TAI	3.643159e+20	0	11065
hmi.sharp_720s[6][2010.05.01_00:24:00_TAI]	2010.05.01_00:24:04_TAI	3.694566e+20	0	11065
hmi.sharp_720s[6][2010.05.01_00:36:00_TAI]	2010.05.01_00:36:04_TAI	3.824390e+20	0	11065
hmi.sharp_720s[6][2010.05.01_00:48:00_TAI]	2010.05.01_00:48:04_TAI	3.915287e+20	0	11065
hmi.sharp_720s[6][2010.05.01_01:00:00_TAI]	2010.05.01_01:00:04_TAI	3.909318e+20	0	11065

Figure 2.9: Magnetogram features' time series example

Figure 2.9 shows the resulting time series of a query performed in JSOC website starting on 2010-05-01 00:00 and ending on 2010-05-01 01:00. The time series sampling period is 12 minutes/observation, and we can see the magnetic fields in the columns USFLUX and R_VALUE for AR regions 11067, 11064 and 11065 varying from 12 to 12 minutes (instants 0, 12, 36 and 48). This example shows only two magnetic properties time series: USFLUX and R_VALUE; though, we can extract any feature presented in Figure 2.8.

- Sunspot features: some works of solar flare forecasting use sunspot magnetic topology. McIntosh (1990) shows a classification of sunspots according to their topology. Figure 2.10 shows a graphical description of the McIntosh classification.

- Chromosphera data: according to Nishizuka et al. (2017), data from the Chromosphere also indicates the occurrence of solar flares. *Atmospheric Imaging Assembly (AIA)* onboard of SDO is the instrument that records these data. In Nishizuka et al. (2017), the images provided by AIA/SDO are sources used for calculating the features derived from the Chromosphere, named *Chromospheric (UV) brightening area, Total chromospheric (UV) brightening and Maximum intensity of chromospheric (UV) brightening*.

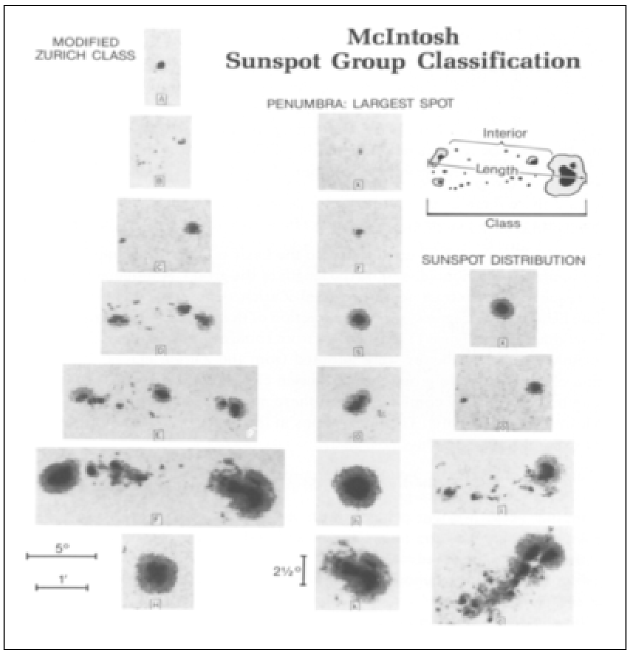


Figure 2.10: McInstosh Sunspot Classification (MCINTOSH, 1990)

2.1.4 The Problem of solar flares - important considerations

There is no agreement on which features fully determine the solar flares. Thus, we performed several practical experiments using the X-ray intensity emitted by the Sun and the magnetic features shown in Figure 2.8 to identify the most significant features for the forecasting process, because:

1. X-ray establishes a direct relation with solar flare classes and their disturbances, impacts and risks to the Earth environment;
2. Magnetic features extracted from SDO/HMI are the most recent features analyzed in current works of solar flare forecasting.

We also used the sunspots topology, but this feature did not result in any improvement.

The data employed in this project have some important characteristics:

1. Periodicity: due to the Solar Cycle characteristic;
2. Highly imbalanced: because intense and extreme solar flares are rarer;
3. Has a large volume;
4. Adjacent classes are difficult to distinguish.

The data characteristics lead us to deal with the forecasting problem using imbalanced time series classification methods. These methods usually use data mining classification methods to deal with the imbalanced data. The most relevant classifiers are shown next.

2.2 Classifiers

In a general overview, the *Data Mining (DM) process* applies the following steps (MAIMON; ROKACH, 2010):

1. Determination of DM goals;
2. Data collection;
3. Preprocessing: data transformation and cleaning, selection of the DM task, and choice of a specific DM algorithm;
4. Execution of the mining algorithm in the processed data;
5. Evaluation of the results;
6. Deployment of the model in an operational environment.

Our solar flare forecasting method follows these steps as it is designed through a DM process. The DM task chosen in this thesis was classification.

Some of the main data mining classifiers used in time series forecasting and classification are K-Nearest Neighbor, Support Vector Machine, Decision Trees and Artificial Neural Networks. These classifiers may be used as initially designed or may be modified to meet a specific domain requirement. We show in this section the original versions, and in the following sections, we show their possible usages in imbalanced and/or time series datasets.

2.2.1 K-Nearest Neighbor (K-NN)

K-NN is a method of *Instance-based learning*. It is a supervised method, where each training instance must be previously labeled. The idea behind the method is to use a distance function, like Euclidean or Manhattan, to compare each testing instance with the ones in the training dataset. The resulting classification is chosen depending on the label of the k closest training instances. For example, if k is set to 5, the method will label the testing instance with the most frequent class of the five-nearest training instances. The most simple algorithm of this method (SILVA; PERES; BOSCHARIOLI, 2016) is shown in Algorithm 1.

Algorithm 1: K-NN basic algorithm

```
   input  : testing_data, training_data
   output: label
1  begin
2      For each training_data, calculate the distance between training_data and
         testing_data;
3      Order training_data according to the distance obtained in the last step;
4      Select $k$ first examples;
5      Classify testing_data with the label that occur most frequently in the $k$ first
         examples of the ordered list;
6  end
```

The best choice of *k* depends on the data that it is working. If we assume higher values of *k*, it may lead to a reduction of the noise effect on the classification, but in the counterpart, it makes the boundaries between classes more difficult to be distinguished (SILVA; PERES; BOSCHARIOLI, 2016). In the other hand, assuming a small value of *k* can turn the method more noise sensitive.

2.2.2 Support Vector Machine (SVM)

SVM is a statistical learning machine algorithm introduced by Cortes e Vapnik (1995) in 1995 based on Vapnik's *Statistical Learning Theory*. The main idea of SVM is to estimate a function that minimizes the risk of misclassification. It separates two classes through a hyperplane so that when the data is linearly separable, a function can determine to which class a certain tuple belongs. Figure 2.11 illustrates this situation.

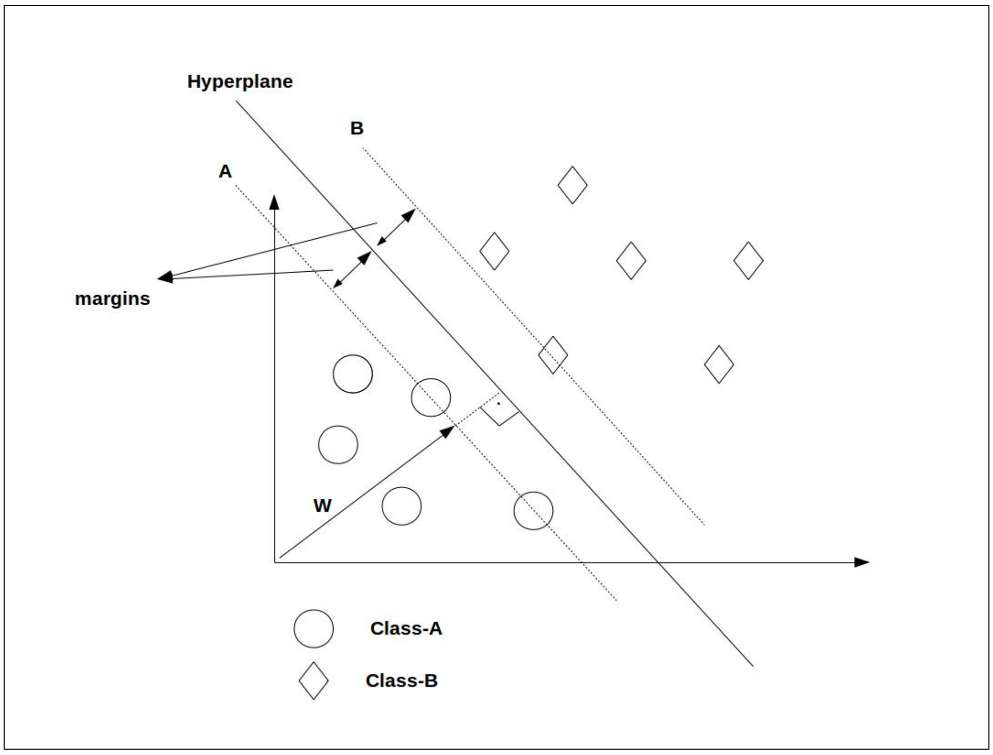


Figure 2.11: SVM example - Linear separable data (SHARMA et al., 2013)

Note that each class, represented by a circle or a diamond, can be separated by a Hyperplane (line), as shown in Figure 2.11. The best Hyperplane to separate the tuples of each class is defined as the one containing the largest margin between the set of instances of each class. The distance between the hyperplane and the nearest instances of each class should be the same and these are called *Support Vectors*. SVM uses a statistical concept called *Constrained Quadratic Optimization Problem* to find the Hyperplane that best separates each class (CORTES; VAPNIK, 1995).

If the data can not be linearly separated, like Figure 2.12 shows, an additional strategy should be used by SVM (CORTES; VAPNIK, 1995).

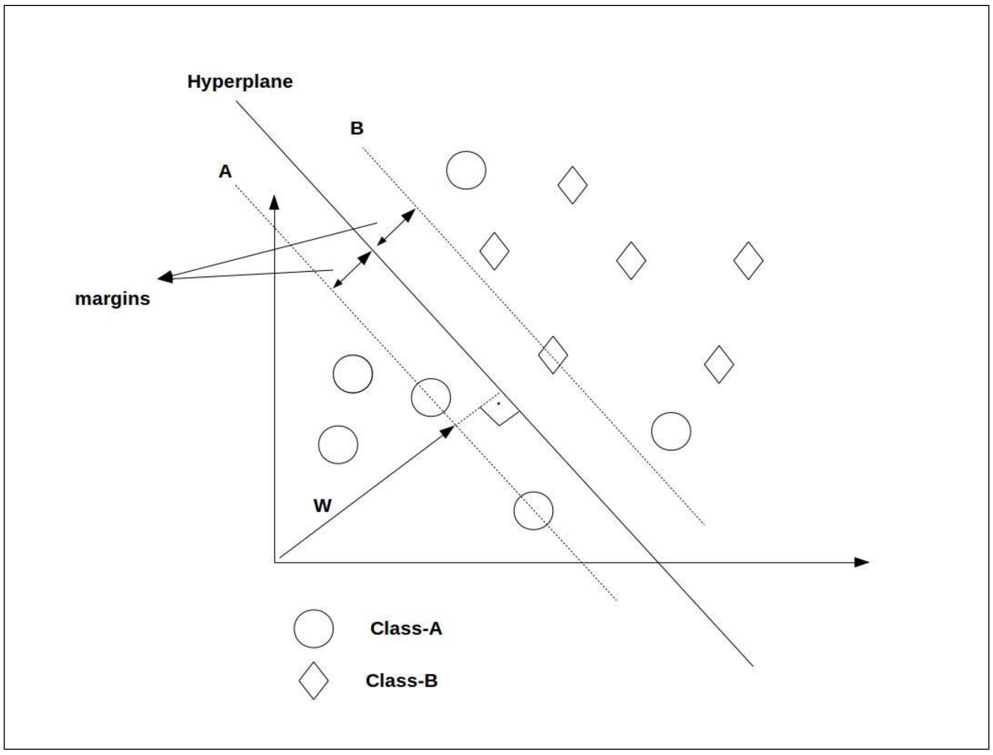


Figure 2.12: SVM example - Non-linear separable data (SHARMA et al., 2013)

The non-linear data must feed a function called *Kernel*. This function increases the data dimension so that the new data can be separable by a hyperplane, instead of by a simple line.

Usually, SVM avoids the overfitting problem because the model produced (a function that implements the hyperplane) needs information of a reduced set of instances, the Support Vectors (BALL; BRUNNER, 2010). These instances don't change so often as the dataset varies so that we can say that the model doesn't overfit. Some implementations of SVM allow to set up weights for different classes. This is a useful characteristic to employ when working with imbalanced datasets.

2.2.3 Decision Trees

Decision Trees is a class of algorithm that constructs a tree with a set of internal nodes and leaves, designing a classification model. The internal nodes are features of a certain dataset. Each sub-trees derived from these internal nodes are obtained through the possible results that these features should assume. This structure is repeated until a definitive result (the prediction) is obtained at the leaf node. This tree is built using a training set.

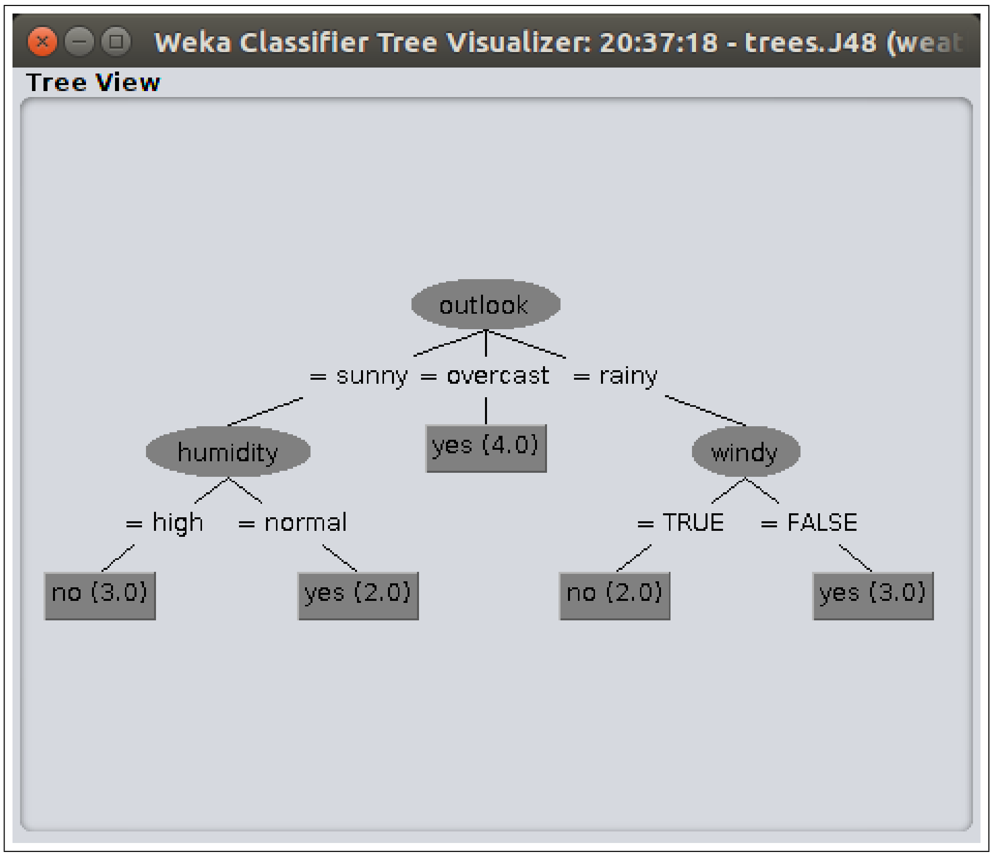


Figure 2.13: A Decision Tree example from Weka

Figure 2.13 shows an example generated by Weka tool (HALL et al., 2009) of a Decision Tree algorithm result. The implementation used is called J48, which consists of a Java implementation of the C4.5 Decision Tree Algorithm (QUINLAN; ROSS, 1993). We can see that the nodes are composed of the training features: *outlook, humidity, and windy*. We can observe that the tree does not contain the feature named *temperature*, even though it is in the training dataset of this example. This is due to the fact that the Decision Tree also analyzes how important each feature is in order to predict the testing instance. This is done using some metrics like the degree of purity of a certain feature, a statistical metric called Entropy and the information gain criteria.

2.2.4 Artificial Neural Networks

Artificial Neural Networks (ANN) simulates the cells of the human brain called neurons and their interconnections. For this purpose, the artificial neuron implemented in ANN is a function that maps input into outputs.

Frank Rosemblatt proposed the Artificial Neuron called *Perceptron* in 1955, which formed the basis for the neural networks (YADAV; YADAV; KUMAR, 2015). This model is composed of the processing units defined by *McCullock - Pitts*. Each processing unit produces an output that is the sum of weighted inputs. So, a general unit performs the following steps: (1) receives signals (input data); (2) each signal is multiplied by the weight assigned for that input; (3) the weighted sum of all the signals is calculated, producing the called activity level; (4) finally, the final answer of the processing unit depends on the level of activity achieved in the previous step. This is decided by an activation function. The weighted sum is calculated using Equation 2.1 and the activation function is given by Equation 2.2.

$$v_0 = \sum_{j=1}^{d} (x_j * w_{0j}) + b_0 \tag{2.1}$$

$$y_0 = f(v_0) \tag{2.2}$$

Figure 2.14 shows the processing unit general model.

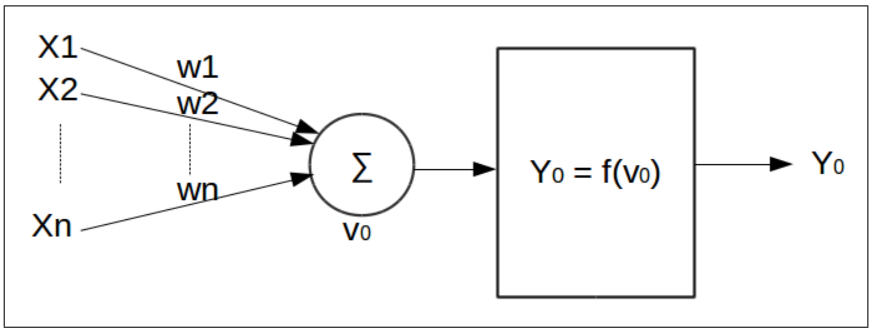


Figure 2.14: Neural network processing unit

The weight of each input is obtained through iterative analysis of the error between the predicted and the expected values. The weight estimation is updated to minimize this error. The method that contains just one Neuron (Perceptron) is the simplest one. The training phase consists on iteratively updating the weights of the Neuron. This model can be extended so that a set of neurons are connected and consequently, composing an Artificial Neural Network (ANN). There are several implementations of ANN, named: Cascade Correlation Neural Network, Multilayer Perceptron Neural (MLP) Network, Learning Vector Quantization, etc. Next, we present an overview of MLP.

Multilayer Perceptron Neural Network (MLP)

Figure 2.15 shows an example of MLP. This sort of ANN is composed by an Input Layer (IL), a Hidden Layer (HL) and an Output Layer (OL).

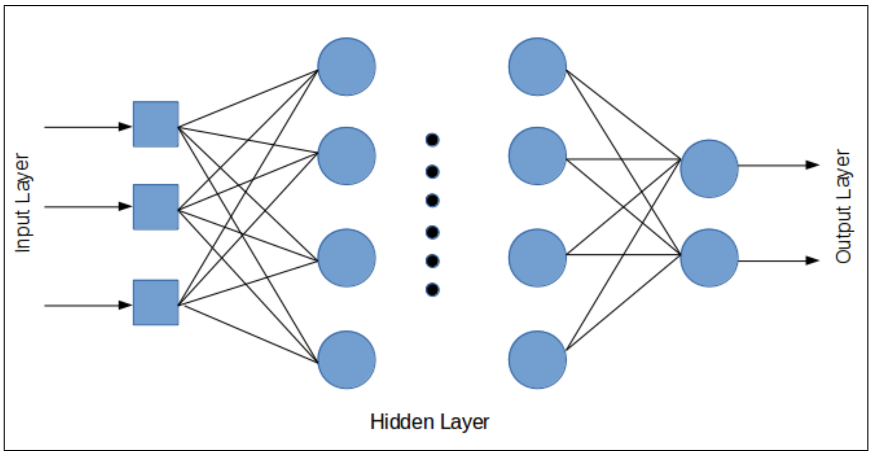
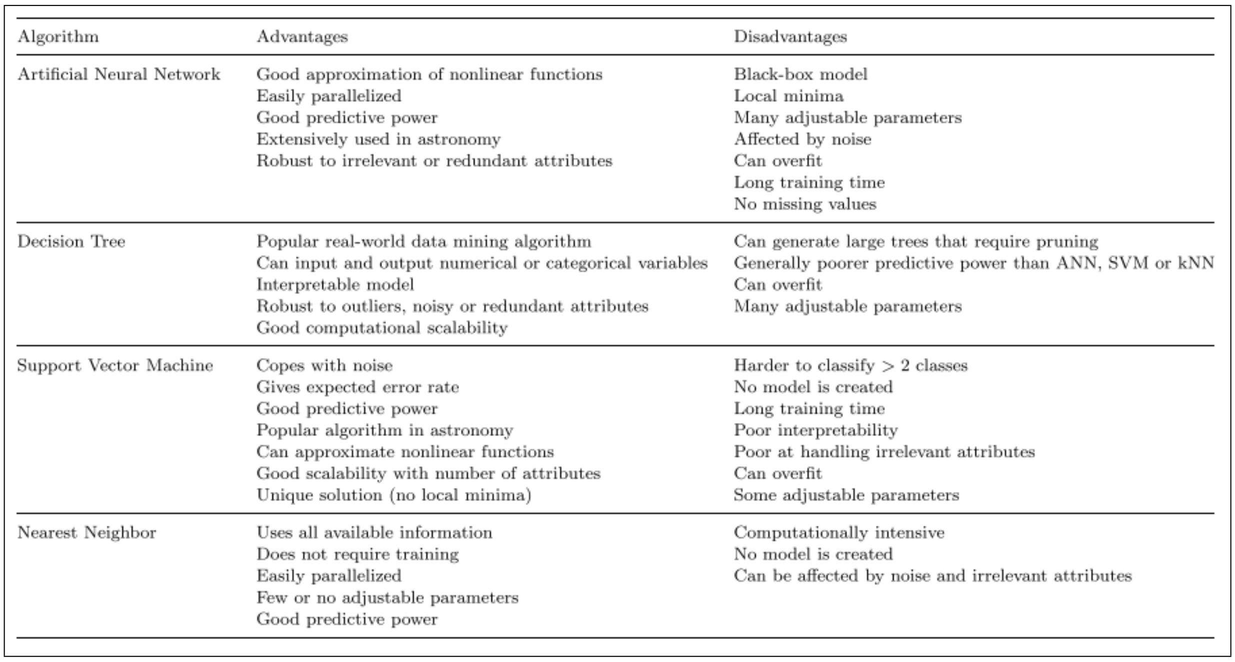


Figure 2.15: MLP Illustration

The IL is responsible for getting input information, applying the respective weights and feeding HL. Then, HL calculates its output using their weights. The output is obtained by calculating the sum of the received values from HL, applying their weights and producing the final output in OL. The second step of the algorithm must update the weights of all the layers. For this purpose, an algorithm of error backpropagation is performed. In this algorithm, the error is calculated using the MLP output compared with the actual value of the training dataset. Then, this error is propagated for the hidden layer and finally for the input layer. Thus, the weights are updated and the next iteration starts again until a certain error threshold is achieved. Note that MLP may have more than one hidden layer (BALL; BRUNNER, 2010). Figure 2.15 shows the general schema of a MLP.

2.2.5 Classifiers - Important Considerations

Figure 2.16 shows a comparison found in Ball e Brunner (2010) among data mining classification methods. According to this comparison, ANN and SVM have great potential to deal with non-linear input data and have good predictive power. Decision Trees although have good computational scalability achieves not such good results. Whereas KNN is easily parallelized, having good predictive power in the cost of being computationally intensive (BALL; BRUNNER, 2010). As we discuss later, SVM and KNN are very often used in solar flare forecasting methods found in the literature.



Algorithm	Advantages	Disadvantages
Artificial Neural Network	Good approximation of nonlinear functions Easily parallelized Good predictive power Extensively used in astronomy Robust to irrelevant or redundant attributes	Black-box model Local minima Many adjustable parameters Affected by noise Can overfit Long training time No missing values
Decision Tree	Popular real-world data mining algorithm Can input and output numerical or categorical variables Interpretable model Robust to outliers, noisy or redundant attributes Good computational scalability	Can generate large trees that require pruning Generally poorer predictive power than ANN, SVM or kNN Can overfit Many adjustable parameters
Support Vector Machine	Copes with noise Gives expected error rate Good predictive power Popular algorithm in astronomy Can approximate nonlinear functions Good scalability with number of attributes Unique solution (no local minima)	Harder to classify > 2 classes No model is created Long training time Poor interpretability Poor at handling irrelevant attributes Can overfit Some adjustable parameters
Nearest Neighbor	Uses all available information Does not require training Easily parallelized Few or no adjustable parameters Good predictive power	Computationally intensive No model is created Can be affected by noise and irrelevant attributes

Figure 2.16: Comparison of data mining classification methods (BALL; BRUNNER, 2010)

In order to apply classifiers to forecast future events, observed tuples should be labeled with time-posterior classes. The majority of the solar flare forecasting methods labels an observed instance as Yes or No depending on if there were a flare in the next few hours. Usually, the dataset does not contain the evolution of the solar features. However, it is essential to consider the historical evolution while labeling observed instances. This was one of our goals in this work.

In the solar flare domain, classifiers are employed in time series classification to analyze the original time series and its evolution. Techniques used in time series classification are discussed next.

2.3 Time Series Classification

A dataset used for solar flare forecasting contains several solar features. Each solar feature is measured in a determined sample rate, producing a time series. Hence, the solar flare dataset is composed of a set of time series, one for each feature. In this way, the solar flare forecasting problem can be treated as a problem of time series forecasting. Next, we present some time series definitions according to Esling e Agon (2012).

Definition 2.1. An univariate time series T is an ordered set of observations of a variable $T = (t_1, t_2, ..., t_\text{n})$, where $t_n \in \mathbb{R}$ is a real value of the variable T measured at time n.

A multivariate time series is a set of time series T related with a specific domain and observed in equal sample rates.

Definition 2.2. Given a time series T of length n, a time series subset S *of* T is a series of length $m \leq n$ consisting of contiguous time instants from T: $S = (t_k, t_{k+1}, ..., t_{k+m-1})$, where $1 \leq k \leq n - m + 1$. The set of all subsets of T of length m is denoted as S_m^T.

Definition 2.3. A time-series dataset DS is an unordered set of time series.

Definition 2.4. Time series classification: given an unlabeled time series T, there is a function f that assigns T to one class c_i from a set $C = C_i$.

Definition 2.5. Time series forecasting is the predictive task of forecasting values of T for time instant $ts > t$, where ts is the instant of time in which it is intended to predict, and t is the instant of time of the current observation.

In literature, it is found forecasting methods that use Regression Statistical Models (RSM) applied to Machine Learning Methods (MLM). We found several RSMs for time series forecasting like Auto-Regressive Moving Average model (ARMA) (MAKRIDAKIS, 1994), Seasonal Auto-regressive Integrated Moving Average (SARIMA) (BROCKWELL; DAVIS, 2002) and Hidden Markov Models (JOSHI; SRIVASTAVA, 2014). There are works that use Support Vector Machine (SVM), K-Nearest Neighbor, Decision Trees and Neural Networks aiming to forecast time series (DONGXIAO; TIANNAN; BINGYI, 2016; PEDRO; COIMBRA, 2015; CHEN; GOO; SHEN, 2014; MEKANIK et al., 2013). These works have in common the capability to predict future numerical values of the time series analyzed.

However, the concept of time series forecasting used in this thesis is that there is a model that given an observed instance (or time series subsets), it returns a class of a future event. In this case, the predicted event is the *future event*, which is formally defined as Y_{t+h} = f(<description Of The Past>), where the $< descriptionOfThePast >$ is the time series subset or observed data used to describe the set of values sampled in the past, and the function f is the model that returns the class of the forecasting event (OLIVEIRA; TORGO, 2015).

The forecasting process can be seen as a type of time series classification problem because the goal is to map observed data to one or more class(es) (as stated in Definition 2.4). As the main data structure of solar flare domain is composed of time series and the concept of forecasting is implicitly a type of classification, we focused our solution on the development of a suitable time series classification method that forecasts a solar flare class. In following, we described the BAGNALL et al. taxonomy for time series classification.

Bagnall et al. (2017) categorizes time series classification methods in the following categories: *Whole intervals, Phase Dependent Intervals, Phase Independent Shapelets, Dictionary based and Combinations*:

2.3.1 Whole-based

Methods that fall in this category consider the whole time series in order to perform the classification task. These methods usually perform the following steps to complete the classification task: (1) calculate the distance between the testing instance and the training dataset, and (2) labels the testing instance with the same class of the closest training instance found in the first step. These methods have the challenge of how to choose the best distance measure according to the training time series pattern. Some domains present training time series aligned with the majority of the testing instances, but other domains face the problem of scaling misalignments. This issue determines how the distance measure is calculated. One-nearest neighbor (1-NN) is an example of a *Whole-based* method. When this classifier is used with time series as input, it computes the distance between the training set instances and the testing instance. Afterward, it labels the testing instance with the closest training instance found.

Time series classifiers may use different types of measures, like *Euclidean Distance* or *Dynamic Time Warping (DTW)* (XI et al., 2006). The first calculates the distance between two time series using the equation $E_d = \sum_1^n (t_{test} - t_{training_i})^2$, where n is the length of the time series, t_{test} is the testing instance, and$t_{training_i}$ is the i^{th} training instance. This measure is simple, fast and applicable in many domains. However, it fails to compare misaligned time series(AGRAWAL; FALOUTSOS; SWAMI, 1993).

Another distance measure is *Dynamic Time Warping (DTW)*. DTW is called an elastic measure, because it is not sensitive to scale misalignments between the training and testing instances (SAKOE; CHIBA, 1978; RAKTHANMANON et al., 2013). DTW calculates the distance of all the points of the testing instance to each point of the training set. The sum of these distances produces a path, called *Warping Path (WP)*. The resulting distance is the path that minimizes the *WP*. So, as it considers the distance between all points of the time series, it avoids the misalignment issue. Though DTW is time-consuming, it is highly parallelizable, like implemented in Rakthanmanon et al. (2013). Many derivations of the original method were developed (JEONG; JEONG; OMITAOMU, 2011; MARTEAU, 2009; BATISTA et al., 2014) to optimize its processing. However, those works fail to compare not aligned time series.

Figure 2.17 shows examples of time series in which the usage of DTW as distance measure is interesting. The two time series on top belong to the same class, and the two on the bottom belong to another class. The time series of the same class are visually similar, but presents some differences in scaling, peaks and troughs.

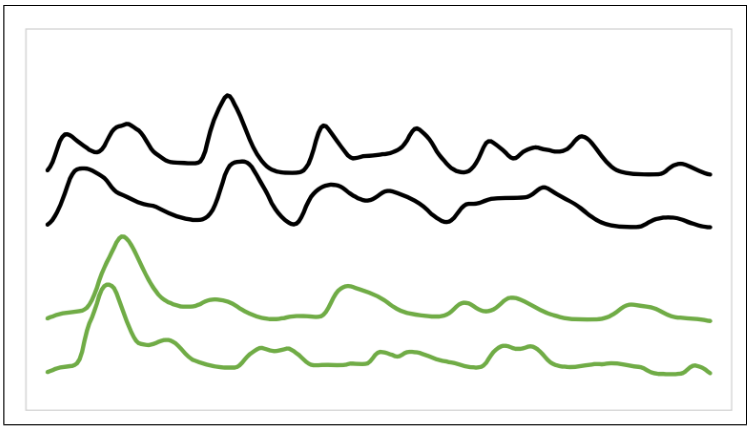


Figure 2.17: Example of time series that may be classified using *Whole-based* methods. Adapted from (BAGNALL et al., 2017)

Methods that compare the *whole time series* may get confused with noises. Additionally, a specific interval can best describe a given class than the whole time series. Thus, another set of algorithms tries to overcome this issue by seeking patterns in one or more time series range(s).

2.3.2 Intervals-based

Intervals-based methods select a subset of the time series in certain intervals to compare them. The general steps of the algorithms that follow this category are: (1) Select the intervals; (2) Obtain new features through the processing of the selected intervals, and; (3) Submit the data to a classifier.

These methods have two challenges: how to determine the most significant intervals of data, and which data transformation techniques should be performed in the selected data to improve the classification accuracy. Some methods decrease the intervals size until they reach a good accuracy. As examples, we can find *Support Vector Machines of interval-based features, Time Series - bag of features and Learned pattern similarity* (RODRÍGUEZ; ALONSO; MAESTRO, 2005; BAYDOGAN; RUNGER; TUV, 2013; BAYDOGAN; RUNGER, 2016).

Figure 2.18 shows examples of time series suitable to be treated with Intervals-based methods. These time series present a well defined noisy and discriminatory regions in specific intervals. In this example, data patterns located at specific intervals determine the correct class.

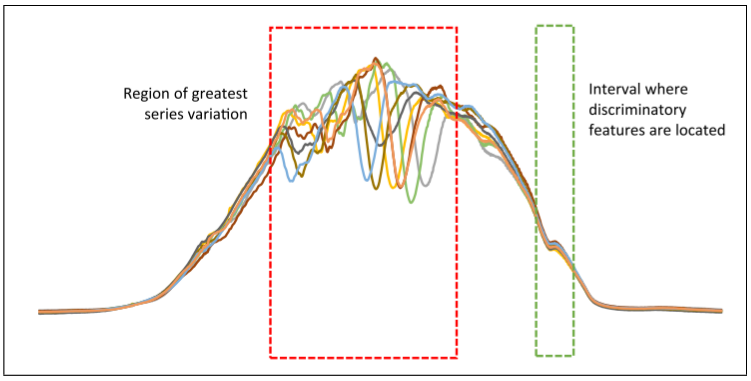


Figure 2.18: Example of time series suitable to be handled with Intervals-based methods. Adapted from (BAGNALL et al., 2017)

Methods that implement the concept of time series *intervals* have the advantage of considering the patterns in one or more intervals through the time series, but it also constrains the classification within time series of the same length, and the ranges of comparison are also fixed.

2.3.3 Shapelets-based

The methods of this type try to find shapelets. A Shapelet is the most discriminant, phase-independent subsequence in a given time series that define the classes of the problem independently of the time interval (BOSTROM; BAGNALL, 2015). Thus, the same pattern may be seen several times in a specific time series. In general, they run the following steps: (1) find a set of shapelets using a distance measure; (2) build a new training dataset containing the distances between the time series and the shapelets; (3) build a model using a classifier; (4) transform the testing instances by calculating their distances to the shapelets; (5) classify using the created model.

Shapelets-based methods have the advantage of being more flexible in the classification task since it analyzes several parts of the time series independently. Otherwise, they need to look for the shapelets (patterns) in the time series, so that they usually run slowly. Some *Shapelets*-based methods are *Time series shapelets, Fast Shapelet Transform, Fast shapelets and Shapelet transform* described in Ye e Keogh (2011), Ji et al. (2018), Rakthanmanon e Keogh (2013), Bostrom e Bagnall (2015). These methods differ from the type of selection criteria of the shapelet.

These methods may classify binary (YE; KEOGH, 2011) and multi-class datasets (BOSTROM; BAGNALL, 2015), which is an attractive characteristic for the solar flare domain. On the other hand, another important issue that our problem may face is that solar flares datasets are multi-label, and this sort of methods are not prepared to deal with this issue. Additionally, the slow execution time to run them is a limitation to our problem.

Figure 2.19 shows three time series that are analyzed against one shapelet. We observe that time series classified as *27 and 28* have reasonable matches with the shapelet, leading to a correct classification, and class 32 had no good match. In fact, the *shapelets*-based approach is recommended in problems having patterns that should be compared with any time series interval.

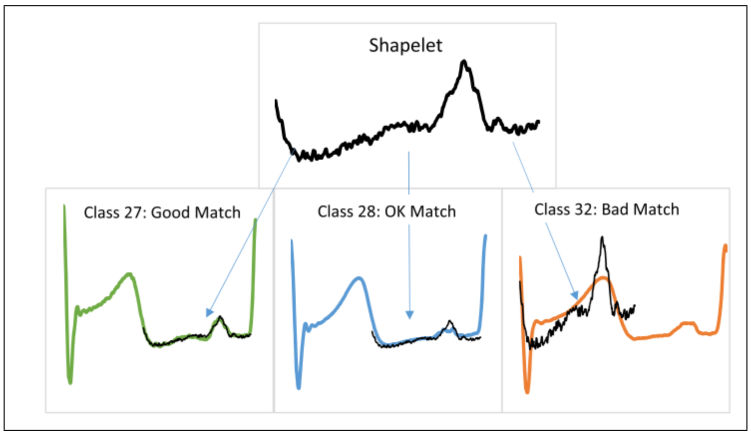


Figure 2.19: Example of time series suitable to be handled with *Shapelets*-based methods. Adapted from (BAGNALL et al., 2017)

2.3.4 Dictionary-based

Sometimes classes are determined by the frequency of patterns occurrences in the time series, instead of the match of a single occurrence of the pattern. This singularity may cause a misclassification if *Shapelets* are being used. Thus, the repetition of a certain pattern in a certain frequency is looked for methods that implement *Dictionary-based* concept.

Dictionary-based methods implement the following steps: (1) Given a training instance, find a set of patterns that lead to its class; (2) For each pattern, count the frequency in the training instance; (3) Build a new training dataset with the frequency of each pattern found; (4) Run a classifier to obtain a classification model; (5) Transform a testing instance by counting the frequency of the patterns found; (6) Submit the transformed testing instance to the built model.

These methods usually differ in how the patterns are selected. In Lin, Khade e Li (2012), Senin e Malinchik (2013), Schäfer (2015), the methods that implement this concept are: *Bag of patterns, Symbolic aggregate approximation-vector space model and Bag of SFA symbols.*

As the *Shapelet*-based methods, the *Dictionary*-based algorithms are also computationally expensive due to the high requirements needed to find the dictionary of patterns used. In fact, the high-computation cost of the *Dictionary*-based methods avoid us to employ them in our proposed method.

Figure 2.20 shows two classes of time series of movements regarding two species of worms (BAGNALL et al., 2017). A normal worm has a continuous with some few abnormal movements. An abnormal worm has frequent abnormal movements. This fact is shown in the two top time series that represents a normal worm and, consequently, few abnormal movements. The two series on the bottom have frequent abnormal patterns. So, *Dictionary*-based methods may be used in this kind of domain, because it classifies a time series according to the frequency in which a given pattern occurs.

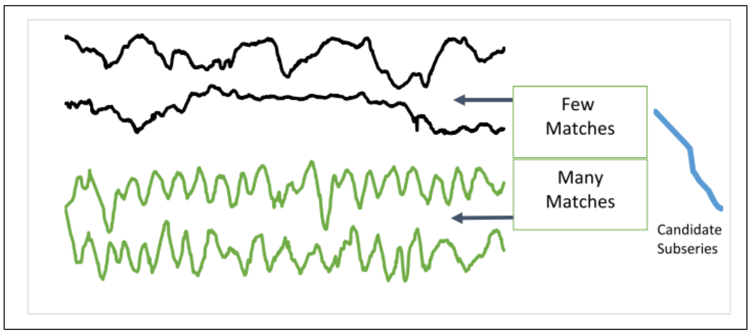


Figure 2.20: Example of time series suitable to be handled with *Dictionary*-based methods. Adapted from (BAGNALL et al., 2017)

2.3.5 Combination-based

Methods of this category usually implement a data pre-processing (also called transformation) and a classification. It uses two or more time series classification approaches. Ensemble of Classifiers is an example of a Combination-based method. The main advantage of these methods is that they minimize the drawbacks of the base algorithms, but they are still specific for certain domains. Two example methods of this category are: *DTW features and Collection of Transformation Ensembles* (KATE, 2016; BAGNALL; JANACEK, 2014).

2.3.6 Time Series Classification - Important considerations

In this work, we preprocessed the dataset to map the observed time series with future solar flares. Then, a time series classification is performed with this dataset to provide the intended forecast. In this task, we faced the problem in which there are few differences among time series that lead to solar flares of classes C, M and X, and their sub-levels. Hence, in this thesis, we employed Combination-based approach to minimize the effects of this problem.

Another issue faced was the highly imbalanced dataset, because the highest solar flares are infrequent. Thus, we faced the problem of highly imbalanced dataset classification. The original versions of time series classification methods were not designed to deal with imbalanced datasets, being unsuitable for treating imbalanced data. In this way, we applied to the time series classification methods some strategies to handle imbalanced datasets.

2.4 Classifying imbalanced datasets

Many domains produce datasets which contain very few instances of a class. The imbalanced dataset causes the imbalanced classification problem, where the classifier misclassifies instances of the minority class.

This issue leads to the learning models that labels the majority of instances (BRANCO; TORGO; RIBEIRO, 2016). As the classification method processes too many observations of the frequent class and very few occurrences of the rare class, it produces a biased result. So, the learning model tends to classify the majority of the testing data as the most frequent class, generating many errors (LONGADGE; DONGRE, 2013).

As the most energetic solar flares (M and X) are very rare, and they have the highest misclassification costs, existing or new strategies to handle the imbalanced data of solar flares are very relevant in the context of forecasting these events.

Many methods have been proposed in the literature to tackle the problem of data imbalance. Different taxonomies ((BRANCO; TORGO; RIBEIRO, 2016), (LONGADGE; DONGRE, 2013) and (KRAWCZYK, 2016)) categorize the approaches used by the methods that handle the imbalance issue. Krawczyk (2016) presents a general taxonomy used to categorize methods that handle imbalanced classification. These categories are described next.

1. **Data Level**: the solutions that use this approach make modifications in the input data to minimize the effects of the imbalanced dataset. Galar et al. (2012) states that Class Distribution is the proportion of instances belonging to each class in a dataset. Usually, methods that implement the **Data Level** approach change the class distribution to give more weight to the minority classes in which the user is generally more interested.

2. **Algorithm Level**: in this type of approach, the classification algorithm is changed to take into account the misclassification costs of either the majority or the minority classes. The algorithm is also set to be sensitive to the user preference.

3. **Hybrid Methods**: in this approach, a combination of the solutions given in the first two approaches are used to obtain the best benefits of each one and minimize their drawbacks.

Each of these approaches have advantages and disadvantages as show in Table 2.1.

Table 2.1: Advantages and disadvantages of the approaches used to tackle the problem of classification in imbalanced datasets (KRAWCZYK, 2016).

Strategy	Advantages	Disadvantages
Data Level	- It is possible to use any learning method; - The distribution is chosen directly related with the user target.	- The modification of the data distribution usually does not reflect in a correct prediction; - The learning model may incur in the problem of over-fitting.
Algorithm Level	- The user target preferences are incorporated directly to the learning method.	- The developer may know the whole algorithm of the method, which may be difficult; - If any change occurs in the data distribution, the developer may change and recompile the method.
Hybrid	- The combination of the previous approaches determines the advantages of this approach.	- The combination of the previous approaches determines the disadvantages of this approach.

2.4.1 Data Level

Data Level approaches preprocess the dataset before its submission to the classifier. For this purpose, the original distribution is modified to handle the imbalance issue. The dataset is modified by undersampling or oversampling its instances. Although the correct determination of the data distribution is a challenging task, the user preference is considered at the beginning of the learning process. Distribution changes can be performed using random sampling, data cleaning, one-class learning, synthetic instances creation, among other methods.

Random sampling is used to perform undersampling of the instances that belong to the majority class or to perform oversampling to the instances that belong to the minority class. If oversampling is used, instances of the minority class are replicated, which may result in unfair weights. On the other hand, undersampling may cause loss of information and overfitting (Abd Elrahman; ABRAHAM, 2015). The sample distribution can be chosen empirically. Most employed distributions are generated by setting the classes frequencies to 2:1 or 3:1 (majority: minority classes).

Data cleaning is a technique used to remove noisy instances or overlapping regions in order to undersample the instances that belong to the majority class. Methods found in the literature may look for overlapping regions using the closest neighbors and decide whether remove or not the instance, as the *Tomek Link Method* (BATISTA; PRATI; MONARD, 2004); or, find the minimal subset of the training set that performs like the original dataset (*Condensed Nearest Neighbor* (HART; P., 1968)).

One-class learning techniques may be used to perform undersampling because they remove instances of the minority class, leaving just the ones that belong to the majority class. One-class classification methods look for patterns in the testing data to identify instances that belong to the majority class, consequently the ones not identified belong to a different class. Usually, a threshold is used by these methods to identify the class patterns. It is hard to set the threshold to classify the instances correctly. Despite this drawback, this technique is still useful in many domains (BRANCO; TORGO; RIBEIRO, 2016).

Synthetic instances creation is a set of techniques used to perform oversampling with artificially created instances. Among them, it is found the Synthetic Minority Over-sampling Technique (SMOTE). In SMOTE, instances of the minority class are oversampled by introducing new synthetic examples along the line that connects one or more of the *k*- nearest neighbors. This strategy leads to a more general learning model produced by the classifier as shown in Chawla et al. (2002). The original SMOTE method first undersamples instances of the majority class. Second, it creates synthetic examples to balance the bias towards the majority and minority classes. SMOTE has a potential limitation of creating over generalized models, which motivated the creation of new methods based on that. However, the original one is still a feasible method.

2.4.2 Algorithm Level

Another approach to handle imbalanced dataset problems is to modify the classifier algorithm. This approach has the advantage of setting the user preference directly to the classification method. This is usually done by implementing strategies to deal with different weights for each class and/or including misclassification costs information in the algorithm, turning them cost-sensitive. These strategies set the classifiers to deal with the imbalanced issue because the modified methods are prepared to receive and take into account the classes (minority or majority) that are most significant to the user preferences (BRANCO; TORGO; RIBEIRO, 2016). Among the modified methods, it is found the k-nearest neighbor, Support Vector Machines, Neural Network, Decision Trees and Ensemble of Classifiers. The modification of existing methods requires knowledge of their implementation, and how to quantify the costs involved in misclassification as well as the benefits obtained from correct classifications. However, sometimes this cost is difficult to obtain.

On this thesis problem, the cost of forecasting errors in the solar flare domain is not entirely quantifiable. The cost depends on the effect on each Earth's device (HUANG; WANG; DAI, 2012). In fact, in our work, we did not modify traditional classifiers, but we modified the datasets and employed an Ensemble of Classifiers, employing a Hybrid method in this thesis.

2.4.3 Hybrid Methods

Hybrid Methods combine **Data and Algorithm Levels** techniques to maximize the advantages and minimize the drawbacks of each one (KRAWCZYK, 2016). Methods that follow this approach are implemented by combining sampling schema with a set of classifiers and, optionally, post-processing the classifier output (WOŹNIAK; GRAÑA; CORCHADO, 2014). Some of the hybrid methods also use learning algorithms that implement a weighting schema given by the user preferences through cost-sensitive analysis (Senzhang Wang et al., 2012). Additionally, Ensemble of Classifiers (EC) methods like Boosting and Bagging can be used to classify imbalanced data (FERNÁNDEZ et al., 2017; GALAR et al., 2012) employing data preprocessing. This mixture of techniques aims to give significant importance to the instances labeled as the minority class. EC are methods that work with a set of classifiers to increase the performance of a single base classifier. They were not originally designed to deal with imbalanced datasets, but certain modifications allow these methods to achieve good results in domains with imbalanced datasets (GALAR et al., 2012). ECs are often used with sampling methods (which implements the Data Level approach).

In the following, we describe two sampling schemas (Random Sampling and SMOTE), and, two EC methods (AdaBoosting and Bagging) employed with EC in this work.

2.4.4 Sampling Algorithms

Sampling algorithms may be implemented randomly or using some heuristic (PRATI; BATISTA; MONARD, 2009). Two of the main algorithms of sampling were *Random under sampling* and *SMOTE*. We had employed random undersampling on our work due to its simplicity, good results and the feasibility of applying them in our domain. Next, we describe both algorithms.

Random Undersampling With Replacement

Random undersampling is the random sampling schema used in the majority class. It can perform sampling with or without replacement. If no replacement is used, then the algorithm will randomly select an instance from the original dataset, which can not be used in any other selection. On the other hand, in algorithms that use sampling with replacement, one instance that is randomly selected, can be sampled again.

Sampling methods are used as a starting point to tackle the imbalanced issue since it aims to obtain a balanced distribution of instances of different classes (PRATI; BATISTA; MONARD, 2008). It may cause loss of information because it removes instances and, thus, information on the training dataset (BRANCO; TORGO; RIBEIRO, 2016). The use of sampling with replacement may specialize the model causing the overfitting problem. Though, if the sampling is done by the insertion of new synthetic instances, like SMOTE, it may introduce noise to the training dataset. So, empirical tests may be performed to find the most feasible method. The main advantages of random undersampling are that it is simple to implement and it runs fast. Next, Algorithm 2 presents the pseudo-code of the random-sampling method implemented in this thesis.

Algorithm 2: RandomUnderSampling(D_o,$P\%$)

```
input  : Original Dataset $D_o$, Percentage of under sampling $P\%$
output: Sampled Dataset $D_s$
1 begin
2     $majorityclass$ = class with most instances in $D_o$;
3     $minorityclass$ = class with less instances in $D_o$;
4     $D_{majority}$ = dataset composed with the instances of $majorityclass$ from $D_o$;
5     $D_{minority}$ = dataset composed with the instances of $minorityclass$ from $D_o$;
6     $D_s = D_{minority} \cup randomWithReplacement(D_{majority}, P)$;
7     return $D_s$
8 end
```

In our thesis work, the sampled dataset is a union of all the instances classified with the minority class with a random undersampling with replacement of the majority class. The main challenge of this approach is to find the most proper distribution between the majority and minority classes. Some authors recommend a distribution of 2:1 or 3:1 for extremely imbalanced datasets, but we obtained the best results with a 1:1 distribution (as shown in our experiments).

SMOTE

The idea behind SMOTE is to oversample the instances of minority class to change the balance of the original dataset. This method aims to obtain a balanced result between True Positives and False Positive of the minority class prediction (FERNANDEZ et al., 2018). The algorithm creates synthetic instances by randomly selecting a subset of the minority class instances in the first moment. Then, it finds the K-nearest neighbors of the instances of the subset and randomly selects N instances among them. Finally, it calculates the difference between each sample and its N-neighbors, multiplies this difference by a random number between 0 and 1, and adds this result to the first sample, producing a new synthetic instance. The detailed process is shown in Algorithm 3 (FERNANDEZ et al., 2018). SMOTE was also combined with random undersampling with replacement of the majority class to balance instances of the minority and majority classes (CHAWLA et al., 2002). The application of both methods increased the area under ROC curve of many datasets tested. The best distribution of classes was obtained empirically.

Algorithm 3: SMOTE(T,N, k): adapted from (CHAWLA et al., 2002)

```
input  : Number of minority class samples $T$; Amount of SMOTE $N\%$; Number
         of nearest neighbors $k$
output: (N/100)*$T$ synthetic minority class samples
1  begin
2      (* If N is less than 100%, randomize the minority class samples as only a
         random percent of them will be SMOTED. *)
3      if $N < 100$ then
4          Randomize the $T$ minority class samples;
5          $T = (N/100) * T$;
6          $N = 100$;
7      end
8      $N = (int)(N/100)$ (* The amount of SMOTE is assumed to be in integral
         multiples of 100.*) ;
9      $k$ = Number of nearest neighbors ;
10     numattrs = Number of attributes ;
11     Sample[][]: array of original minority class samples ;
12     newindex: keeps a count of number of synthetic samples generated,
         initialized to 0 ;
13     Synthetic[][]: array for synthetic samples ;
14     ( * Compute k nearest neighbors for each minority class sample only. * )
15     for i = 1 to T do
16         Compute k nearest neighbors for i, and save the indices in the nnarray ;
17         (* This part of the code starts generating the synthetic samples*);
18         while $N \leq 0$ do
19             Choose a random number between 1 and $k$, call it $n$. This step
                 chooses one of the $k$ nearest neighbors of $i$ ;
20             for attr = 1 to numattrs do
21                 Compute: dif = Sample[nnarray[nn]][attr] - Sample[i][attr] ;
22                 Compute: $gap = randomnumberbetween0and1$ ;
23                 $Synthetic[newindex][attr] = Sample[i][attr] + gap * dif$
24             end
25             $newindex++$ ;
26             $N = N - 1$ ;
27         end
28         (* End of the code that generates the synthetic samples*);
29     end
30 end
```

2.4.5 Ensemble of classifiers

An ensemble is a set of base classifiers, called inducers, whose results are combined to produce the final prediction. This composition let the overall prediction system be benefited with the strengths of each classifier. There are two types of ECs: the stage independent and the stage dependent.

In the first type, the training dataset feeds several inducers in parallel, then different models are built, and the testing instance is submitted to these models. Finally, the results of each model are combined to calculate the final prediction.

In the second type, a training dataset feeds an inducer so that a model is produced. Then, this model is tested against the original dataset, and an error schema is calculated. With this error, weights are given to the inducers, and a new model is produced. The algorithm ends when it reaches a certain minimum error rate (KRAWCZYK et al., 2017).

Usually, an ensemble of classifiers is more effective than each of its components working alone. The classification obtained is given by a strategy of the composition of votes of the individual classifiers. This strategy of votes may be done by selecting the class with the highest number of votes.

An issue of the ensembles is the need to generate diversity for diversifying the learning models, increasing the ensemble accuracy and reducing overfitting. An example of a strategy to increase the diversity among classifiers is the *Bootstrap Aggregating* (Bagging) where the base classifiers (usually induced by the same algorithm) are submitted to potentially different sub-samples of the training set (MENDES-MOREIRA et al., 2012).

The algorithms of the two main EC, Bagging and Boosting, are presented in following.

Bagging

Bagging is called an independent Ensemble of Classifiers because each base inducer does not depend on the results of any other inducer of the ensemble. The first step of the algorithm is to create I subsample datasets with the same size, according to the Bootstrap technique. The datasets created must be subsets of random samples with replacement of the original dataset. Then, each dataset trains the base inducers producing I models. Finally, a test instance is submitted to each model, and the final prediction is the most voted class (BREIMAN, 1996). Algorithm 4 presents the pseudo-code of Bagging.

Bagging is a simple algorithm that may be easily parallelized due to have independent inducers. Another compelling advantage of this method is the diversity of classifiers that may be used in the ensemble. This characteristic increases the power of generalization of the method in imbalanced datasets. Another advantage is that it joins the data preprocessing with the diversity of classifiers. This strategy reduces the need to apply weights to features and instances in the prediction process. Because of it, this method inspired our thesis work. In our proposed method, we used the diversity of classifiers and voting schema to deal with the solar flare prediction, a highly imbalanced class problem.

Algorithm 4: Bagging(*S*,*L*,*I*) (DU et al., 2012)

```
  input : training sample $S$, Classifier $L$, iterations $I$
  output: Result $L_E$
1 begin
2     for $i$=1 to $I$ do
3         $S_i$ = boostrap sample from $S$ ;
4         $L_i$ = train a classifier on $S_i$ via $L$ ;
5     end
6     $L_E = \underset{y \in Y}{argmax} \sum_{i:L_i(x)=y} 1$ ;
7 end
```

AdaBoosting

AdaBoosting is a type of Ensemble of classifiers that iteratively calculates *weights* for each training instance and the called *voting power* for its inducers in T iterations (GALAR et al., 2012). The number of T depends on: (1) achieving the desired accuracy ; (2) defining a number; (3)no longer increasing the accuracy (after new iterations). The tasks involved in each iteration is described in the following and detailed in Algorithm 5.

1. Initialize each training point with equal weights: line 3 of Algorithm 5;
2. Submit the training points with their respective weights to the weak classifier, so that it returns a classification model that minimizes the error rate: line 5 of Algorithm 5;
3. Calculate the error rate of the model obtained through the previous weights: line 6 of Algorithm 5;
4. Calculate the called *voting power* α_t of the weak classifier through its error rate: line 11 of Algorithm 5;

5. For each instance of the training dataset, re-calculate the new weights according to the previous ones and the weak classifier error rate. In this step, the weight increases if the instance was misclassified, and decreases, otherwise: lines 12 to 19 of Algorithm 5.

The final classification is given by the sign function of a weighted sum of each trained weak classifier (line 21 of Algorithm 5). The main advantage of AdaBoosting is that it turns weak classifiers, which return a result a few better than random, to strong ones using a simple algorithm. On the other hand, as the algorithm is sequential, it is difficult to parallelize. In the context of multi-class classification in an imbalanced dataset, the original algorithm was not prepared because: (1) it was initially developed for binary classification, and (2) minority class was not tackled in this work. However, changes in the instance weighting mechanism combined with data pre-processing produced interesting methods derived from AdaBoosting, like $AdaBoost.M1$ and $AdaBoost.M2$ (FREUND; SCHAPIRE, 1997; SCHAPIRE; SINGER, 1999).

Algorithm 5: AdaBoosting(S,T,I) adapted from (GALAR et al., 2012)

```
input  : Training set $S = x_i, y_i$,i = 1,...,N; and $y_i \in -1, +1$; $T$: Number of
         iterations; $I$: Weak learner
output: Boosted classifier: $H(x) = sign(\sum_{t=1}^{T} \alpha_t h_t(x))$
begin
    (* $h_t$ are the induced classifiers (with $h(x) \in -1, +1$) and their assigned
      weights, respectively)
    $W_1(i) = 1/N$ for $i = 1, ..., N$;
    for t=1 to T do
        $h_t = I(S, W_t)$;
        $\varepsilon_t = \sum_{i, y_i \neq h_t(x)} W_t(i)$;
        if $\varepsilon_t > 0.5$ then
            $T = t - 1$;
            return;
        end
        $\alpha_t = \frac{1}{2} ln(\frac{1-\varepsilon_t}{\varepsilon_t})$;
        foreach instance i classified in $h_t$ do
            if i is misclassified then
                $W_{t+1}(i) = \frac{1}{2} \times \frac{1}{\varepsilon_t} \times W_t(i)$;
            end
            else
                $W_{t+1}(i) = \frac{1}{2} \times \frac{1}{1-\varepsilon_t} \times W_t(i)$;
            end
        end
    end
    return $H(x) = sign(\sum_{t=1}^{T} \alpha_t h_t(x))$;
end
```

2.4.6 Classifying imbalanced datasets - Important considerations

In our work, we combine time series classification approaches with techniques that handles imbalanced datasets to perform the forecasting task. To handle the imbalanced dataset issue, we employed random sampling and Bagging. These methods are fast and allow establishing weights to different classes through the change of the distribution of the dataset. As our domain is multi-class, random sampling was performed to built balanced datasets for each solar flare class. These datasets feed binary inducers that composed our Ensembled-based solution, generating a fair result. Next, the metrics used to validate forecasting methods are presented.

2.5 Validation metrics for forecasting methods

Several metrics can be used to validate forecasting methods and many of them were defined for time series classification method (BRANCO; TORGO; RIBEIRO, 2016). In this way, an important concept is the *Confusion Matrix*, presented in Table 2.2. This matrix maps **observed** and **predicted** values by the forecasting method.

Table 2.2: Confusion Matrix

		Predicted classification	
		Class-A	**Class-B**
Observed classification	**Class-A**	TP	FN
	Class-B	FP	TN

The rows are related to the observed (actual) classification, and the columns represent the predicted classification (forecasting). In this example, Class-A corresponds to the **Positive** class and Class-B to the **Negative** class. So, if an actual Class-A observation is foreseen as Class-A, we obtain a True Positive (TP), because the forecast matches the actual observation for the Positive class. If an actual Class-A observation is foreseen as Class-B, we obtain a False Negative (FN), because the forecast misses a Positive class. Looking at the second row, the Confusion Matrix results False Positive (FP) if a Class-B observation is foreseen as Class-A, because the method predicted a Positive class instead of the actual observation (Class-B, or Negative Class). Finally, it produces a True Negative (TN) if the forecasting method predicts a Class-B (or Negative Class) for an actual Class-B(Negative Class) observation. So, in the Confusion Matrix, **TP**, **FN**, **FP** and **TN** are the numbers of True Positive, False Negative, False Positive and True Negative resulted for a specific testing dataset and they are used in the formula of several metrics for the validation of forecasting methods.

2.5.1 Accuracy

Accuracy is the ratio of correctly classified instances by the total number of instances of the testing dataset. Equation 2.3 shows the formula of this metric.

$$Accuracy = \frac{TP + TN}{TP + FN + TN + FP} \tag{2.3}$$

2.5.2 Precision

Precision is the ratio of instances that are correctly classified as a particular class C by the total number of instances classified as C. This rate can be interpreted as *What percentage of instances was correctly classified as C from all class-C forecasting results?* Equation 2.4 shows the formula of this metric.

$$Precision = \frac{TP}{TP + FP} \tag{2.4}$$

2.5.3 Recall or True Positive Rate

Recall or True Positive Rate (TPR) is the ratio of instances that are correctly classified as a particular class C by the total number of instances that belong to C. This rate can be interpreted as *What is the percentage of relevant forecasts obtained from the model over the total amount of actual relevant events (solar flares) that have already occurred?* Equation 2.5 shows the formula of this metric.

$$Recall = \frac{TP}{TP + FN} \tag{2.5}$$

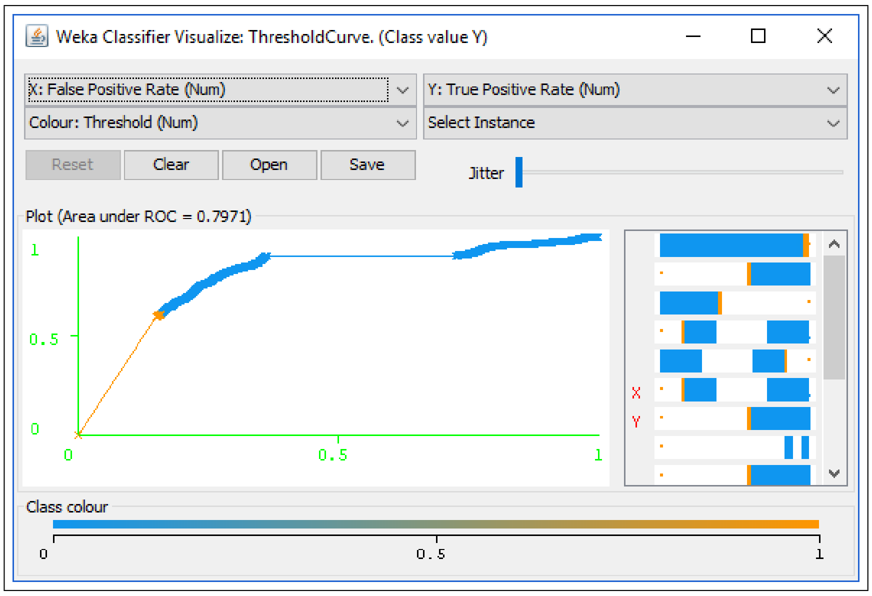
2.5.4 Specificity

Specificity or True Negative Rate is the ratio of instances that are correctly classified as the Negative class by the total amount of instances that belongs to the Negative class. Equation 2.6 shows the formula of this metric.

$$Specificity = \frac{TN}{TN + FP} \tag{2.6}$$

2.5.5 Receiver Operating Characteristic (ROC) Area

The ROC curve is obtained by plotting the cumulative distributive function of True Positive Rate against the False Positive Rate at different thresholds. So, it relates the probability of detection (TPR) to the false-alarm (FPR). If the area under this curve approaches 1.0 and the Accuracy is high, it can be said that TPR and TNR are balanced so that the forecasting method is performing well for both classes (BRANCO; TORGO; RIBEIRO, 2016).


Weka Classifier Visualize: ThresholdCurve. (Class value Y)

X: False Positive Rate (Num)

Y: True Positive Rate (Num)

Colour: Threshold (Num)

Plot (Area under ROC = 0.7971)

Figure 2.21: An example of ROC Curve

2.5.6 Skill Scores

The validation of a forecasting method should take into account the balance between True Positive Rate and True Negative Rate. If we have an imbalanced dataset with many more Negative instances than Positive ones, and we analyze these metrics individually, it can lead us to a wrong interpretation. For instance, a method reaching high accuracy could have classified correctly almost all the Negative instances and missed almost all the Positive ones. Considering the imbalanced nature of the solar dataset, if we analyze only Accuracy, or only Precision, or only Recall, or only Specificity, we may not have a definitive analysis of the forecasting method results.

An interesting way to validate a forecasting method is using Skill scores, a metric that compares the evaluated method with a benchmark.

$$SkillScore = \frac{A_{\text{forecast}} - A_{\text{reference}}}{A_{\text{perfect}} - A_{\text{reference}}} \tag{2.7}$$

Equation 2.7 shows a general formula for general Skill scores where:

- A_{forecast} is the accuracy of the evaluated forecasting method;
- $A_{\text{reference}}$ is the accuracy of the forecasting method used as benchmarking;
- A_{perfect} is the accuracy of a perfect forecasting.

A well-known Skill score is called Heidke Skill Score (HSS) which measures the improvement of the evaluated method over a random forecast (see Equation 2.8).

$$HSS = \frac{TP + TN - E}{P + N - E} \tag{2.8}$$

where:

- $P = TP + FN$
- $N = TN + FP$
- $E = \frac{(TP+FP)x(TP+FN)+(FP+TN)x(FN+TN)}{P+N}$

This metric is strongly dependent on how severe is the imbalance of the testing dataset. However, its characteristic of comparing forecasting methods with different training and testing dataset makes it frequently used as a validation metric in the solar flare forecasting domain. The forecasting methods that result in HSS closer to 1.0 are considered satisfactory.

Another Skill score metric used is the Hanssen-Kuiper Skill Score, also known as True Skill Statistics (TSS), which calculates the difference between the recall and the false alarm rate. TSS is presented in Equation 2.9.

$$TSS = \frac{TP}{TP+FN} - \frac{FP}{FP-TN} \tag{2.9}$$

TSS achieves results from -1 to 1, where -1 represents bad performance and +1, perfect one. According to Bobra e Couvidat (2015), this metric deals well with imbalanced testing datasets and it does not depend on the attributes and the datasets employed. For this reason, we considered this metric when evaluating our proposed method.

2.6 Final considerations

This chapter presented the background concepts needed to develop the proposed method. Our work concentrated on the forecasting of solar flares, even though our first results forecast the background levels. We employed magnetic features of Active Regions found in the magnetogram vector as well as the time series of X-ray intensity emitted by the Sun as the input of our method.

We employed classifiers by mapping observed values with future events. Moreover, our work used sampling and the general schema of Ensemble of Classifiers to combine the results and produce final forecasting as shown in Chapter 5. To validate the proposed forecasting method, we used the metrics presented in this chapter. Next chapter, the related works are presented.

Chapter 3

Related works

In our domain, we are interested in the forecasting of five classes of solar flares (A, B, C, M, and X) leading to a multi-class problem. Also, in a specific forecasting day, several classes of solar flares may occur. As a consequence, we also face a multi-label problem. So, solar flare forecasting is a multi-class and multi-label problem. Another essential issue to handle is that the main computational methods used to forecast time series obtain the future value of this time series. As commented before, we are interested in, given an unlabeled time series of solar data, predict the label of a future solar flare. Thus, it is necessary to adapt the classification problem to predict the label of a future instance. Finally, the solar data is highly imbalanced. Thus, we start this chapter describing some computational methods used in other application domains that are closely related to our needs. They handle imbalanced time series, like ours. Some are used to perform classifications of current labels of imbalanced time series, which partially meets our problem. And others may be used to forecast future events, like our needs.

Besides, several solar flare forecasting methods that use classification methods are closely related to ours, though not handling the issues identified. These closely related solar flare forecasting methods are also presented in this chapter.

3.1 Classification methods for imbalanced time series

In this section, we present some of the main classification methods for imbalanced time series found in the literature. These methods usually change the training data distribution to deal with the imbalanced dataset and next, submit this modified data to a time series classifier.

In Cao et al. (2011), a method called SPO: Structure Preserving Oversampling for Imbalanced Time Series Classification is proposed. It uses the oversampling of minority class instances to tackle the imbalanced learning issue. For this purpose, it produces new synthetic instances by estimating the covariance structure of the minority class. Also, it performs processing that minimizes the loss function of the model. The preprocessed training dataset is submitted to SVM to build the classification model and finally, predict the test instances. These characteristics lead the model to minimize noisy intervals and the over-fitting problem. This method was designed and tested for binary classification, which is a limitation of it.

In Liang e Zhang (2012), it is presented a comparison between the oversampling method SPO and the usage of several under-sampling ratios with KNN as the classification method. It was used multi-class datasets and the conclusion of the paper was that under-sampling with KNN worked as well as SPO. The main conclusion is that using a faster method (undersampling with KNN - due to the smaller dataset generated by the under-sampling) is as good as the usage of a method that requires more computational processing (SPO - due to the higher number of instances processed generated by the oversampling).

In Liang (2013), it is presented a method called Hybrid sampling composed of over and undersampling, as well as the addition of an ensemble method: Bagging. It tried to overcome the drawbacks found in random over/undersampling techniques and the original Bagging methods in time series classification. For this purpose, the author applies a random over-sampling in the positive instances and a random under-sampling in the negative ones. Then, it generates a set of balanced bootstrap samples with a distribution of 50% for each class. Finally, it submits the bootstrap to the base inducers of a Bagging method. The limitation is that they only tested in binary datasets and the authors used just J48 as base inducer.

In Dhurjad e Banait (2015), the authors took into account the fact that *...due to sequential nature of time series data, variables which are close time are extremely correlated...* to develop the proposed method. They also considered the imbalanced time series issue. They proposed a technique called Majority Weighted Minority Oversampling used to pre-process the input data and build the classification model. This sampling schema tried to improve original SMOTE algorithm by identifying the minority class instances that are harder to learn and assigning to these samples the Euclidean distance from the nearest negative class instance as a way to weight each instance. Through this, the method generates synthetic instances using a clustering technique. It uses statistical techniques of eigenvalue regularization to turn the final model more generalized. The modified dataset is submitted to an SVM classifier. The authors tested in imbalanced binary datasets, achieving good precision and recall results. The limitation of this method is that it only works with the binary classes problem.

In Alonso González, González e Diez (2002), it is presented a method that defines literals that best discriminate the classes of a domain. These literals are categorized according to the frequency of a range of values and if the values are increasing or decreasing. Next, the method implements a *Boosting* adaptation in which, to classify a new example, it calculates a weight for each class, and then the example is assigned to the class with higher weight. For each base inducer, the label of the test instance is analyzed. If it matches, then, for each class, its weight is updated adding the weight of the class for the literal and subtracting, otherwise. The final classification corresponds to the one with the highest weight. This method deals with the called *early classification* issue. This issue is related to the capability of the classification method to predict the time series label given a sub-series of it. The main advantage of this method is that it serves to classify multi-class time series. However, it does not handle the imbalanced data problem.

In Xing, Pei e Yu (2012), it is presented a method that defines the term early classification as the possibility of finding a sub-series of length l in the training time series to perform classification as accurate as the one obtained using the full-time series. For this purpose, they developed a method that splits the training time series into clusters using an agglomerative hierarchical clustering method, named MLHC (DING; HE, 2005) to find the minimum length of the training time series and, then they run 1NN to perform the early classification. Even though this method can forecast future events, it only works with univariate time series and binary class problems.

In He et al. (2015), it is presented a method that performs early classification in multivariate time series. First, the algorithm scans for each feature of the training dataset to extract the shapelet candidates. Then, a subset of the shapelet candidates with length lower than the original is selected, they are called core shapelets. In this step, the shapelets candidates are selected by looking for the highest F-measure generated by the evaluated shapelet. The shapelets for each feature are used to build a classifier for early classification. Next, the method finds the core shapelets for each feature by clustering these shapelets and finding the most significant ones using a measure that depends on recall, precision, and earliness. Earliness is a measure, defined as the minimum length of the time series that may distinguish a specific class. Using the core shapelets, the method builds a classification model, based on query by committee (QBC) for each set of shapelets selected from each feature. Then, a testing instance is submitted to the model, and the method counts the majority result. Finally, the test instance is labeled as the predominant class. Though the method was effective in the early classification task, this method does not perform well with imbalanced data, according to the authors.

In He et al. (2018), it is presented a method that handles the problem of early classification in imbalanced multivariate time series. For this purpose, the authors developed an Ensemble of classifier that: (1) splits the majority class instances into subsets of the same length as the minority class instances; (2) creates *n* datasets grouping each majority subset with the minority ones, so that each subset contains the same length; (3) builds a base inducer, for each balanced subset, by obtaining the called *associate pattern*, which is an association of the best shapelets that best distinguish each feature; (4) uses test instances to be classified in an ensemble model. The final result is a weighted sum of the results given by the base inducers that compose the ensemble. This method is one of the first to handle the imbalanced issue in time series classification. As a drawback, it does not perform multiclass classification, only binary one.

Table 3.1 shows a comparison among the methods in terms of the most relevant characteristics to the domain of this thesis. It also categorizes the methods according to the Time Series Classification and Classification in imbalanced datasets taxonomies described in Chapter 2.

Table 3.1: Comparison of time series classification methods

Paper	**Method**	**Time series classification approach**	**Approach for the classification in imbalanced datasets**	**Classifier**	**Type of classification**	**Dataset Characteristic**
Cao et al. (2011)	SPO: Structure Preserving Oversampling for Imbalanced Time Series Classification	Whole-based	Data level (random sampling / SMOTE)	SVM	Binary	Imbalanced
Liang e Zhang (2012)	Undersampling with KNN	Whole-based	Data level (random sampling / SMOTE)	level	(random sampling	Imbalanced
Liang (2013)	Hybrid sampling	Whole-based	Combination (Sampling + Ensemble of Classifiers)	Decision Tree	Binary	Imbalanced
Dhurjad e Banait (2015)	Majority Weighted Minority Oversampling	Whole-based	Data level (random sampling / SMOTE)	SVM	Binary	Imbalanced
Alonso González, González e Diez (2002)	Boosting interval-based literals	Shapelets-based	Not applied	Boosting based	Multiclass	Balanced
Xing, Pei e Yu (2012)	Early classification on time series	Intervals-based	Not applied	1NN	Binary	Balanced
He et al. (2015)	Early classification on multivariate time series	Shapelets-based	Not applied	Specific Ensemble of classifiers	Multi-class	Balanced
He et al. (2018)	An ensemble of shapelet-based classifiers on inter-class and intra-class imbalanced multivariate time series at the early stage	Shapelets-based	Combination (Sampling + Ensemble of Classifiers)	Specific Ensemble of classifiers	Binary	Imbalanced

We note that the close related methods are distributed in *Whole-intervals*, *Intervals-based* and *Shapelets-based*. As we will see in the next chapter, we take as the hypothesis that solar flares may be distinguished by analyzing the whole time series or a sub-series of it. Thus, we used in our work two approaches: *Whole based* and *Intervals based*.

The methods listed above have their major improvements in the area of time series classification and presented important solutions that are applied to our problem. Mainly, they implement a sampling schema, find a sub-series pattern and use a classification method (traditional or an Ensemble of classifiers) to solve the problems related to imbalanced time series in multiclass domains. Another interesting approach is to find sub-series of the training time series to perform early classification. However, we can notice that two important characteristics of solar flare forecasting were not tackled: the issue of multi-label prediction and the adjacent class problem. Consequently, even those methods inspired us to perform sampling with an ensemble of classifiers; these two issues still had to be faced in our work.

3.2 Solar flare forecasting methods

There are two main approaches used to forecast solar flares: (1) forecasting with statistical methods based on data probability distribution; (2) forecasting with data mining techniques.

The first approach often studies the distribution of solar events within an interval, trying to find the best match and to build statistical models. One of the very first statistical methods used to forecast solar flares is presented in McIntosh (1990). This work describes an expert system called *Theo*, which consists of a set of rules based on the knowledge of the domain specialists (astrophysicists) about solar characteristics (MCINTOSH, 1990).

In Gallagher, Moon e Wang (2002) it was proposed a statistical model by estimating the probability of each solar event through a *Poison* distribution analysis. This method was adopted by the website of *Trinity College Dublin and Dublin Institute for Advanced Studies* that provides solar data from several international observatories called *Solar Monitor*. Another related work that uses statistical tools is Barnes et al. (2007). This work assumes a Gaussian distribution of the solar flare events applied to a statistical approach named *Discriminant Analysis*, which aims to give probabilities for the occurrence of a phenomenon for different groups. For instance, in solar flare forecasting, it would estimate the individual probability of occurrence of flares of classes C, M and X.

Recently, many solar flare forecasting methods have been developed using data mining techniques, taking advantage of machine learning algorithms. Those works usually differ in aspects such as what is foreseen, the preprocessing step, and the adopted classification methods.

The manner that the forecasting methods present their results also differs. Most methods performs binary classification and considers classes C, M, and X as Positive (AHMED et al., 2013); others consider Positive for forecasts greater than or equal to M-class (NISHIZUKA et al., 2017; BOBRA; COUVIDAT, 2015; LI; ZHU, 2013; YU et al., 2010), and very few works forecast individual probabilities for each class (C, M, X).

There are various classification methods employed in forecasting. We found forecasting methods using Support Vector Machines (BOBRA; COUVIDAT, 2015; ZAVVARI et al., 2015; QAHWAJI; COLAK, 2007), Artificial Neural Networks (ZAVVARI et al., 2015; LI; ZHU, 2013; AHMED et al., 2013; WANG et al., 2008), C4.5 decision trees (ZHANG; LIU; WANG, 2011; HUANG et al., 2010; YU et al., 2010, 2009; GALLAGHER; MOON; WANG, 2002), Naive Bayes (ZAVVARI et al., 2015) and Bayesian Networks (ZHANG; LIU; WANG, 2011). Most works try different classifiers, looking for the one that achieves the best results in the application.

The general difference between traditional classification and forecasting methods is the preprocessing method used to prepare the raw data to be submitted to the classifier. In general, the forecasting methods maps observed data with events that occurred in a specific time window (NISHIZUKA et al., 2017; BOBRA; COUVIDAT, 2015; AHMED et al., 2013; YUAN et al., 2010; COLAK; QAHWAJI, 2009). That approach does not consider the evolution of a time series in a particular period to perform the mapping, causing information loss. However, there are also works that map subseries of the solar data into events observed after them, so that they adequately consider the historical evolution of solar data (LI; ZHU, 2013; YU et al., 2010).

Current forecasting methods usually are not flexible enough to be set up with the specialist knowledge: their configuration is set up once, and no parameter can be modified afterward. Parameters that need expert knowledge such as solar features, the size and period of both, training and testing data sets, the period that is considered for mapping values and future events, are all defined a single time, at the beginning of the process. Few methods allow a dynamic configuration of those parameters, as presented in Li e Zhu (2013).

The following section presents a summary of the main characteristics of the works mostly related to ours.

3.3 Closely related solar flare forecasting methods

In Nishizuka et al. (2017), it is presented a solar flare forecasting method which uses line-of-sight magnetogram, vector magnetogram, 1600A broadband filtergram images, and the light curves of the soft X-ray emission as input of three different classifications methods (K-NN, SVM and Extremely Randomized Trees) in order to perform the forecasting of: (1) only class X flares; (2) flares of classes M and X. The main goals of the work are: (1) to include X-ray flux and a chromospheric new set of features (extracted from the filtergram) in the forecasting process besides the usual magnetic features; (2) compare three different classifiers; (3) rank the data features according to their importance in the forecasting system. The method presented in this paper consists of four steps: (1) Construction of the input database. The following set of images and sources were used: line-of-sight and vector magnetograms obtained from SDO and GOES; 1600A broadband filtergram images; and, the light emissions of soft X-rays; (2) Discovery of the Active Regions (AR) of each vector magnetogram and tracking of them (because each AR moves across the visible part of the Sun); (3) Feature extraction: for each AR, the features are extracted and each tuple are labeled as Positive if it produced an M or X flare in the following 24 hours of observation; (4) Machine learning training and testing: KNN, SVM and Extremely Randomized Tree (ERT) were tested using the dataset obtained from the previous step. When comparing the results of the three classifiers, KNN achieved the best results: TSS = 0.91, TPR = 0.93 and a TNR = 0.99. It also obtained an important rank of the features through the application of ERT. It was found that the evolution of the X-ray released is the most important feature employed in the forecasting process.

In Bobra e Couvidat (2015), it is presented a framework for predicting solar flare using the magnetogram vector of active regions. It used Univariate *Feature Selection* for feature selection across 25 features from the vector magnetogram of active areas. It also used Support Vector Machine in the prediction process and True Skill Statistics metrics to analyze the obtained results. The main advantages of the method are the usage of a considerable amount of magnetogram data, but the disadvantages are: it had not considered flares of class C in the training dataset; False Negatives were relatively high (28,6%) and the *precision* obtained was 50%.

In Ahmed et al. (2013), it is proposed a method that used magnetic field properties extracted from SOHO/MDI line-of-sight magnetograms as input to the forecasting system. Then, it employs two feature selection methods, and the obtained database feeds a *Cascade Correlation Neural Network*, which produces the forecasting model. The work has two major goals: (1) use a machine learning algorithm (Neural Network) with feature selection methods (Correlation-based and Minimum Redundancy Maximum Relevance) to predict flares higher than class-C1.0, and (2) determine which attributes have better capability to predict solar flares. The feature selection task selected 6 of the original 21 attributes of Magnetic Fields (MF). The MF properties that are related to the polarity-separation line were seen to be the most significant according to the feature selection. It was also shown that the accuracy of the model using the selected features and the full data are almost the same. The best result achieved by this method was HSS = 0.72, TPR = 0.659 and TNR = 0.992. The method missed 0.341 of the solar flares, indicating a need for improvement.

In Li e Zhu (2013), a method for solar flare forecasting is presented. It uses sunspot area, McIntosh classification, magnetic classification, and radio flux as the input of the preprocessing method that turns each observation of the Active Regions in a sequence through a Sliding Window approach. This approach enables the method to consider the historical evolution of an Active Region in the forecasting process. In order to label each sequence, it calculates the importance of each Active Region, named *Itot*, using the sum of occurrences of individual solar flare classification events: ($Itot = 1.0 \times qtyC + 10.0 \times qtyM + 100.0 \times qtyX$). This formula gives weights to the number of flare occurrences for each class in steps of 10, starting from the lowest solar flare class (C) up to the strongest class (X). Each Active Region is observed for three days, and the size of the window is equal to two days. This parameter settings, according to solar astrophysics, are enough to foreseen future events. The sequence feeds a classification method, which builds the forecasting model. In this work, the author used *MultiLayer Perceptron Network (MLP) and Learning Vector Quantization (LVQ)* in a set of experiments. They achieved 84.3% of accuracy, 67.9% of TPR and 84.9% of TNR using MLP, and 82.8% of accuracy, 67.8% of TPR and 83.7% of TNR using LVQ. They conclude that using the historical evolution of Solar data, the results increased using either MLP and LVQ classifiers. Hence the data evolution should be considered to improve the forecasting method.

In Yu et al. (2010), it presented a work that extracts three predictors, named the maximum horizontal gradient, the length of the neutral line, and the number of singular points from Solar and Heliospheric Observatory/Michelson Doppler Imager longitudinal magnetograms. These predictors feed a sliding window algorithm similar to the one used in Li e Zhu (2013) to build sequences that take into account the historical evolution of the predictors. Then, each sequence serves as the input of a *Maximum overlap discrete wavelet transform and a feature selection method* to decompose the sequences of predictors into four frequency bands. In each band, the maximum, the mean, the standard deviation, and the root mean square are extracted. These features are submitted to a feature selection method. Next, the selected predictors feed a C4.5 decision classifier that builds the forecasting model. It was found that the performance of the short-term solar flare prediction model based on the multiresolution predictors is greatly improved.

In Colak e Qahwaji (2009), a solar flare forecasting method integrated with an automatic feature extraction from SOHO/MDI images was presented. The McIntosh classification and the total area of sunspots are automatically extracted from MDI vector magnetogram. These features feed two Neural Networks. The first predicts if a sunspot will produce flares. If so, the second one forecasts the probabilities of class C, M and X . The NN is trained with a dataset produced using an algorithm (QAHWAJI; COLAK, 2007) that associates sunspots with a solar flare catalog provided by NOAA. This algorithm maps whether sunspots produced solar flares (24 hours after its observation) and its intensity through its time occurrence and localization. The results obtained were measured using five metrics: Quadratic Rate (QR), Probability of Detection (POD), False Alarm Rate (FAR), Percent Correct (PC) and Heidke Skill Score (HSS). For forecasts of 24 hours of antecedence, the method resulted for QR, POD, FAR, PC and HSS: 0.141, 0.772, 0.319, 0.811 and 0.493 for class C; 0.05, 0.865, 0.688, 0.944 and 0.470 for class

M; and, 0.022, 0.917, 0.967, 0.981 and 0.169 for class X. These results were good in terms of accuracy, but for class X flares, the method should be improved due to its low HSS.

3.4 Solar flare forecasting methods discussion

This section presents a summary of the main characteristics of each forecasting method shown above, comparing them according to their:

- Classifier: the data mining classification method used in the related work;
- Validation metric: the metric used to validate the forecasting method;
- Input features: the Solar features that feed the forecasting method;
- Output: it is the result of the solar flare forecasting and contains the criteria to consider the result as a solar flare forecasting (Positive result), or no event (Negative result);
- Dataset size: it corresponds to the period covered by the dataset used in the training phase of the mining process;
- Mapping of observed values and future events: it describes how the method maps the observed values with solar flares;
- Characteristics: it lists the issues handled by each solar flare forecasting method.

Table 3.2 shows a comparison among the classifiers employed in the forecasting process. Most methods used traditional classifiers. But, Nishizuka et al. (2017) used an Ensemble of Classifiers. However, it did not consider the time series evolution, missing essential data for the forecasting process.

Table 3.2: Classifiers used in each solar flare forecasting method

Paper	KNN	SVM	NEURAL NETWORK	C4.5	Ensemble of Classifiers
Nishizuka et al. (2017)	X	X	-	-	X
Bobra e Couvidat (2015)	-	X	-	-	-
Ahmed et al. (2013)	-	-	X	-	-
Li e Zhu (2013)	-	-	X	-	-
Yu et al. (2010)	-	-	-	X	-
Yuan et al. (2010)	-	X	-	-	-
Colak e Qahwaji (2009)	-	-	X	-	-

Table 3.3 shows the features used in the selected methods. As we can see, the features used varies among the methods. The solar flare phenomenon is a physical phenomenon not fully understood and consequently, the features that distinguish its flare classes are still not determined. However, we can notice that magnetic properties are promising because several works use them satisfactorily. Though X-ray flux is not used frequently, we showed that considering the evolution of X-ray time series, we could improve the forecasting results.

Table 3.3: Input features used in each solar flare forecasting method

Paper	X-ray flux (1-8 Angs)	Magnetic Properties (SOHO/MDI)	Magnetic Properties (SDO/HMI)	Chromosphere properties	Sunspot topology	Radio Flux
Nishizuka et al. (2017)	X	-	X	X	-	-
Bobra e Couvidat (2015)	-	-	X	-	-	-
Ahmed et al. (2013)	-	X	-	-	-	-
Li e Zhu (2013)	-		-	-	X	X
Yu et al. (2010)	-	X	-	-	-	-
Yuan et al. (2010)	-	X	-	-	-	X
Colak e Qahwaji (2009)	-	-	-	-	X	-

Table 3.4 lists the metrics used to validate the forecasting methods. We observe that most methods employ different metrics.

Table 3.4: Metrics used to validate each solar flare forecasting method

Paper	TPR	FPR	TNR	FNR	FAR	MSE	ACC	HSS	TSS
Nishizuka et al. (2017)	x	x	x	x	-	-	-	-	-
Bobra e Couvidat (2015)	-	-	-	-	-	-	-	-	x
Ahmed et al. (2013)	x	x	x	x	x	x	x	x	-
Li e Zhu (2013)	x	-	x	-	-	-	x	-	-
Yu et al. (2010)	x	-	x	-	-	-	-	x	-
Yuan et al. (2010)	x	-	x	-	-	-	x	-	-
Colak e Qahwaji (2009)	-	-	-	-	x	-	x	x	-

Table 3.5 shows the forecasting output of the methods. The majority aggregates classes C, M, and X in the positive label to predict a solar flare. Since the datasets are highly imbalanced, there is an inherent challenge to distinguish the rarest classes like M and X. Thus, the forecasting of individual classes is still an open issue in this domain.

Table 3.5: Output of each solar flare forecasting method

Paper	Individual (A, B, C, M and X)	Positive ($\geq C$)	Positive($\geq M$)	Positive ($\geq X$)
Nishizuka et al. (2017)	-	-	X	X
Bobra e Couvidat (2015)	-	-	X	-
Ahmed et al. (2013)	-	X	-	-
Li e Zhu (2013)	-	-	X	-
Yu et al. (2010)	-	-	X	-
Yuan et al. (2010)	-	-	-	X
Colak e Qahwaji (2009)	X	-	-	-

Table 3.6 lists how the training dataset is prepared to enable traditional classifiers to perform the forecasting task. There are two possibilities found in the literature. The first is to label snapshots of solar flare features according to the event that occurred in the next 24 hours. The second type found was to calculate a weighted sum of all the events that occurred after the observation to label the feature instance. On the one hand, in the first approach, the forecasting method does not consider any information about the evolution of solar features over time. On the other hand, the second approach still doesn't consider the full information of the evolution because the classifier is deprived of the sub-series considered. It just provides this information when calculating the label of the observed instance. However, more information about evolution should be taken in the process to improve the forecasting process.

Table 3.6: Mapping of observed values and future events of each solar flare forecasting method

Paper	Positive (Solar flare within 24 hours from the observation)	Positive (weighted sum of the solar flares occurred within the next 24 hours)
Nishizuka et al. (2017)	X	-
Bobra e Couvidat (2015)	X	-
Ahmed et al. (2013)	X	-
Li e Zhu (2013)	-	X
Yu et al. (2010)	-	X
Yuan et al. (2010)	X	-
Colak e Qahwaji (2009)	X	-

Table 3.7 refers to the size of the datasets that each method used as training and testing

inputs. The size is proportional to the number of years that each dataset is about, as shown in the table. COLAK; QAHWAJI uses three data of three solar cycle (34 years), while BOBRA; COUVIDAT uses just 4 years of data. This difference is related to the different solar features used. The first author uses magnetic features, which have been collected for a small period (since 2010), while the second uses the topology of sunspots, which have been collected for a long time.

Table 3.7: Size of the dataset used in each solar flare forecasting method

Paper	Size
Nishizuka et al. (2017)	5 years
Bobra e Couvidat (2015)	4 years
Ahmed et al. (2013)	14 years
Li e Zhu (2013)	12 years
Yu et al. (2010)	12 years
Yuan et al. (2010)	10 years
Colak e Qahwaji (2009)	34 years

Finally, Table 3.8 presents the characteristics addressed by each forecasting method. This table shows that few works tackle the imbalanced issue as well as the undifferentiated class problem. None of them uses multi-label approach, and just two performs multi-class forecasting.

Table 3.8: Characteristics of each solar flare forecasting method

Paper	Individual forecasting	Adjacent class	Imbalanced dataset handling	Historical Evolution of time series	Multi-label issue
Nishizuka et al. (2017)	NO	NO	NO	NO	NO
Bobra e Couvidat (2015)	NO	NO	YES	NO	NO
Ahmed et al. (2013)	NO	NO	NO	NO	NO
Li e Zhu (2013)	NO	NO	NO	YES	NO
Yu et al. (2010)	NO	NO	NO	YES	NO
Yuan et al. (2010)	YES	YES	NO	NO	NO
Colak e Qahwaji (2009)	YES	YES	NO	NO	NO

An important aspect to analyze is the numeric results obtained in the closely related methods. Table 3.9 shows the experimental results collected in each experiment described in the papers. In most of the literature works, many experiments were performed, differing in the classifier used and the training dataset pre-processing method.

Table 3.9: Experimental results of the solar flare forecasting methods

	TPR	TNR	ACC	FPR	Precision	TSS	F-Measure
Nishikawa (=X)	0,90	0,99	0,99	0,0003	0,89	0,91	0,89
Nishikawa (>=M)	0,91	0,99	0,99	0,002	0,92	0,91	0,92
Bobra (>=M)	0,71	0,98	0,97	-	0,80	0,70	0,75
Li (>=M)	0,73	0,78	0,77	-	-	-	-
Ahmed (>=C)	0,67	0,99	0,97	0,006	-	-	-
Yu (>=M)	0,85	0,87	-	-	-	-	-
Yuan (= X)	0,83	0,73	0,74	-	-	-	-
Colak (C,M,X)	0,81	-	-	0.30	-	-	-
Colak (=X)	0,98	-	-	0.97	-	-	-

The values shown in the table represents the best results obtained by each method in their experiments. These works were validated using a cross-validation technique or splitting the dataset in 70% for training and 30% for testing. One important drawback of using cross-validation in these works is that the way that the training dataset is composed turns the training phase biased because the sampling period of the instances is in terms of minutes. For example, let two consecutive instances i_1 and i_2 sampled at times *t* and *t+1*. The probability that i_1 belongs to the same class of i_2 is high because the sampling period usually is of minutes. Also, the values of the features that compose both instances do not have a significant change in minutes. Thus, if a cross-validation approach is used, there is a high probability that the training and the testing datasets contain consecutive instances, like i_1 and i_2. Consequently, if i_2 is being tested, there is a high chance to receive the same class of i_1 (that could be in the training dataset). Thus, this validation method is not fair, and can label previous events using posterior data.

3.5 Final considerations

In this chapter, we presented the related works regarding time series mining methods used in the forecasting process, and some of the leading solar flare forecasting methods. It was presented their strengths, weakness, and gaps, according to our assumptions, that this thesis aimed to handle. In summary, previous works did not deal with the issues that our work intended to deal:

- imbalanced solar flare dataset;
- evolution of solar flare time series;

- multi-class and multi-label forecasting;
- adjacent class forecasting.

We present our proposed methods to achieve these requirements in the following chapters.

III DEVELOPED WORK

Chapter 4

SEMINER FORECASTING METHOD

According to Section 3.2 of Chapter 3, several methods that use data mining in the forecasting process were proposed in the literature. The majority of these methods predict groups of solar flare classes as positive and the remaining classes as negative. Usually, grouping classes $\geq C$ or $\geq M$ as positive. Another significant characteristic of these methods is that they do not take into account the time series evolution. Instead, they use snapshots of the magnetic features to compose the training dataset of the classifiers. Also, current methods do not analyze the periods of sub-series in which data may better distinguish solar flares. Additionally, the domain specialist could not adequately tune current forecasting methods up. Finally, several works use cross-validation for their validation. As discussed in Chapter 3, in this domain, this type of validation may cause a biased result.

In this context, and taking into account our goals, we developed the SeMiner method. The main purpose of this method was to tackle:

1. the lack of methods that use the evolution of the solar time series in the forecasting process;
2. the lack of methods that incorporate the domain specialist knowledge in the forecasting process;
3. the lack of methods that analyze the time periods that best distinguish solar flares.

4.1 First considerations

We first developed a forecasting method called Sequence Miner (SeMiner), which is composed of the typical data mining steps. The core of SeMiner is the Series to Sequence (SS) algorithm, which considers the evolution of the solar data and the domain specialist knowledge. The SeMiner method initially processes solar time series into sequences, using the SS algorithm. Observed sub-series are mapped according to events occurred after them. The specialist can adjust SS by setting up the forecasting horizon, the sequence size, and the size of the time window used to analyze future events. In the second step, the processed sequences are submitted to a feature selection method that determines the most significant sub-series that distinguish solar flares. Then, the produced dataset is submitted to a classifier, resulting in a learning model that predicts solar events in advance. Finally, in the third step, the testing time series are submitted to the learning model. The SS algorithm was also optimized using parallelization techniques for Graphics Processing Units (GPU) through the CUDA-NVIDIA framework.

In order to design SS, the time series of X-ray flux was analyzed during and before the event. Our central assumption was that a sub-series of a certain length taken before an event could distinguish a solar flare. This assumption was based on the information given by the domain specialist:

1. Every solar flare occurs above the X-ray background level;
2. The analysis of the time series from one to two days before the forecasting event may distinguish solar flares.

To illustrate these information, Figure 4.1 shows the evolution during three days of X-ray flux near to four different X-class solar flares.

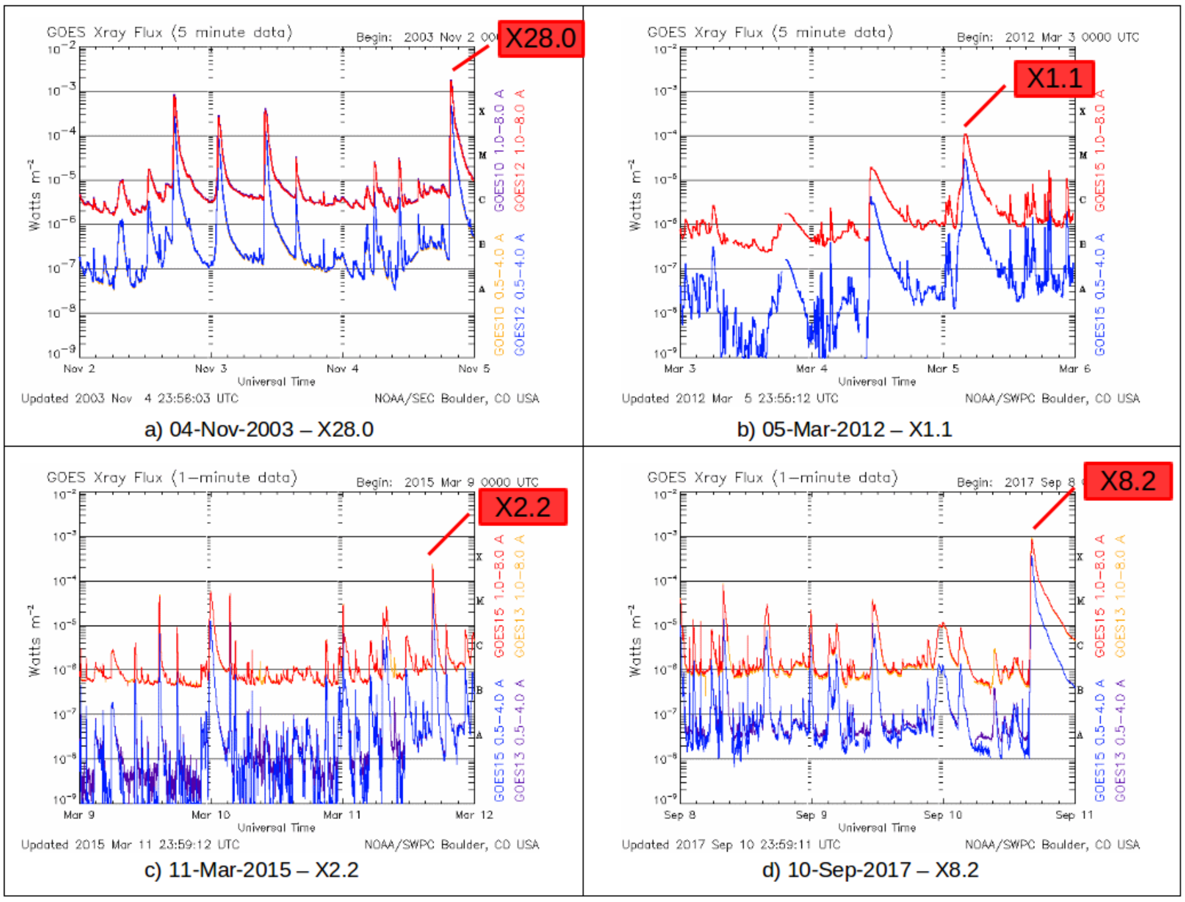


Figure 4.1: Example of a 3-day plot of X-ray flux evolution near X-class solar flares: a) Solar flare of X28.0 class occurred on 04-Nov-2003. b) Solar flare of X1.1 class occurred on 05-Mar-2012. c) Solar flare of X2.2 class occurred on 11-Mar-2015. d) Solar flare of X8.2 class occurred on 10-Sep-2017.

Figure 4.1a, b, c and d shows X-class solar flares of different levels occurred in 2003, 2012, 2015 and 2017, respectively. The first is the strongest X-class event reaching an X-ray intensity of 28.0E-4 W/m^2, the second reached 1.1E-04 W/m^2, and the last two, emitted 2.2E-04 and 8.2E-04 W/m^2, respectively. Another characteristic observed in the figure is the background level that precedes the events. According to the Daily Solar Data report provided at (ftp://ftp.swpc.noaa.gov/pub/indices/old_indices/), the background level occurred in each day of Figure 4.1a, b, c and d were respectively, C2.3, C1.0, B6.8 and B4.9.

Thus, we can observe that all X-class solar flares illustrated in the figure had occurred above the X-ray background level of each day. Besides, we can notice that the background level during two days before remains approximately similar, as shown in the figure.

Based on these two characteristics of the phenomenon, SS lists all possible sub-series (with configured length) that may compose specific windows in order to prepare the dataset. Then, it labels each of these sub-series with the maximum solar flare occurred in a period after that.

Figure 4.2-a,b,c shows a brief example that selects three sub-series belonging to three different windows from the time series of Figure 4.1-d. Notice that each window is slid a certain *d* interval (as shown in Figures 4.2b and c). Then, each sub-series is labeled with the maximum solar flare occurred in the next day. In this example, all the sub-series shown is labeled as X.

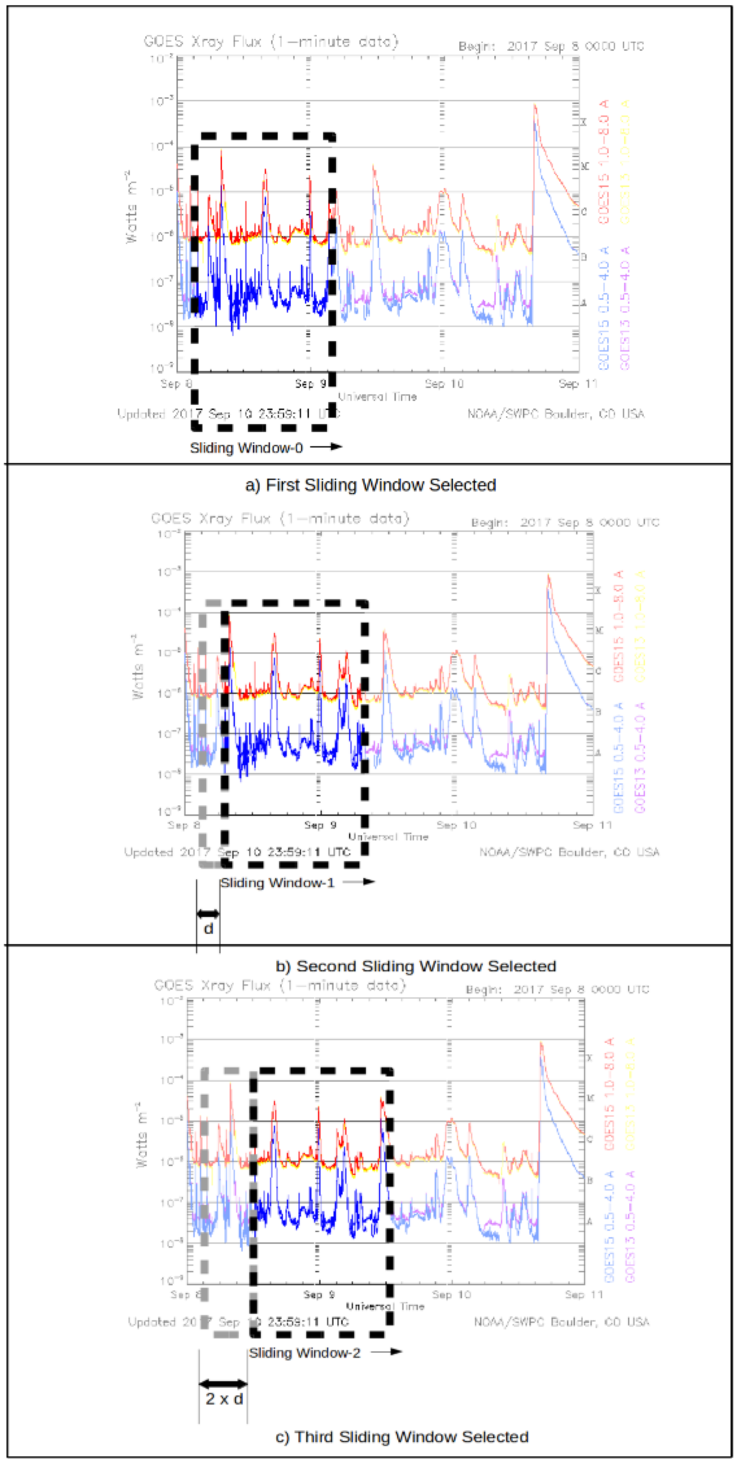


Figure 4.2: Example of the sliding window evolution: a) first window selected. b) the second selected window c) the third selected window.

SS was designed to select sub-series according to the needs of the domain specialist. We also used a variation of the Early classification concept described in Section 3.1, where the domain specialist set the desirable time window size.

In this manner, we tackled the issues of time series evolution and added specialist knowledge within the forecasting process. The last goal of SeMiner was to find the intervals of the training sub-series that best distinguishes the solar flare classes. For this purpose, after the sub-series are obtained, they serve as input for a feature selection method. As each feature corresponds to a value in a specific time instant, the feature selection algorithm selects the best time intervals to be used in the forecasting. Another advantage of this approach is that it also speeds up the training step, because the feature selection decreases the data size processed by the classification method.

4.2 SeMiner description

As mentioned, SeMiner is a forecasting method where the preprocessing step allows the classification method to produce a forecasting model. SeMiner aims to forecast solar events within a given forecasting horizon or antecedence.

The method input is a time series of X-ray intensity emitted by the Sun, which can produce two possible forecasting results: Yes for C-class or higher; and, No for events bellow C. The modules of SeMiner are illustrated in Figure 4.3.

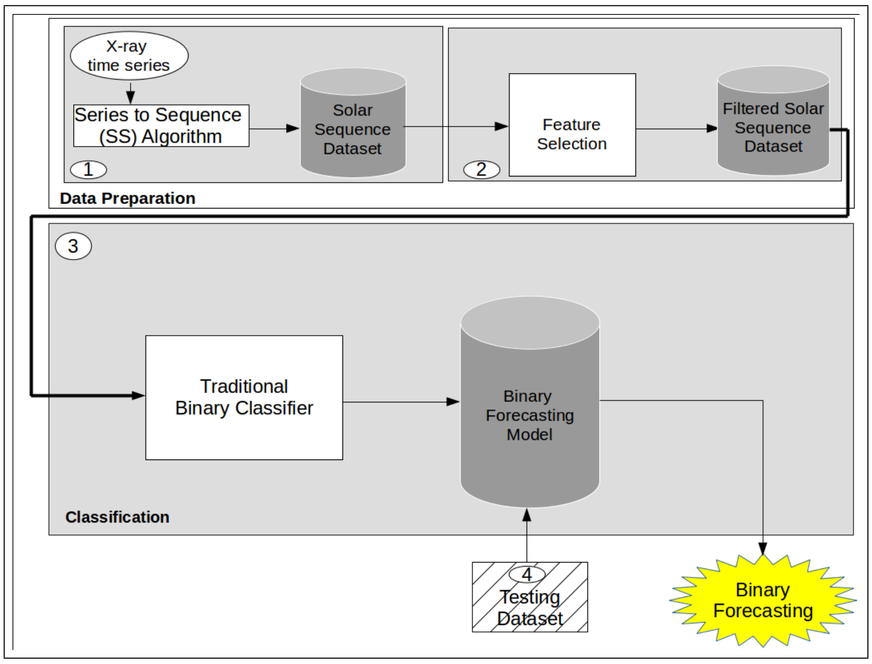


Figure 4.3: SeMiner: Method Overview.

In order to forecast solar flares, the proposed method (SeMiner) encompasses 3 steps: (1) map solar time series onto sequences, done by the Series to Sequence algorithm (SS); (2) perform feature selection; (3) mine the preprocessed solar data through a traditional classification method.

Sections 4.2.1, 4.2.2 and 4.2.3 present the steps executed in SeMiner.

4.2.1 The Series to Sequence (SS) Algorithm

Traditional classifiers use sub-series initially sampled in a timestamp t to predict the class of the same timestamp. However, we aim to build a model in which a sub-series initially sampled in timestamp t, return the class of timestamp $t+h$, where h is the forecasting horizon.

Therefore, how is it possible to forecast future labels using traditional classifiers? SeMiner allows it by preprocessing the time series using the Series to Sequence (SS) algorithm.

Next, an example that illustrates the concepts and tasks performed during the execution of the SS algorithm is presented. Then, its implementations are described.

Table 4.1 presents an example of an X-ray time series mapped with the solar flares occurred in each time instant. It is possible to see that, at *instant 0*, the Sun emitted 3.65 $\times 10^{-7}$ W/m^2 of X-ray intensity, and at this instant, it also produced a B-class solar flare. The SS algorithm uses this time series to map observed values with solar events that occurred after these observations.

Table 4.1: Example of a labeled-unidimensional time series $XRTS(t)$

X-ray Time Series		
t	x(t)	Solar flare class
0	3.65E-7	B
5	3.92E-7	B
10	4.09E-7	B
15	4.04E-7	B
20	3.92E-7	B
25	3.94E-7	B
30	3.84E-7	B
35	3.80E-7	B
40	3.80E-7	B
45	3.83E-6	C
50	3.84E-7	B
55	3.90E-7	B
60	6.47E-7	B
65	6.75E-7	B
70	5.24E-7	B

The main idea of the SS algorithm is to map the evolution of X-ray time series for each window to the maximum solar flare occurred after a specified period. The mapping is called Solar Sequence, and the dataset composed of all the solar sequences is named Solar Sequence Dataset (SSD). A window is a set of observations extracted from the original time series within a period. In SS, it is composed of the *current window, jump and future window*. *Current window* is the data collected in a period of size *c* used as input to the learning model. *Jump* is the interval between the last data collected and the period of the forecasting result. The *future window* is the interval (of size *f*) ahead the *Jump* period that the forecasting method is able to perform the predictions.

As previously said, the sliding window approach builds a set of windows with the same size, but each window begins in a time-shifted from the previous window by a certain number of instances. Table 4.2 shows an example of an execution of this approach. In this example, a window is composed of 12 instances from the X-ray time series. To build the next windows, they are shifted by a certain number of instances, called *step*. In this example, the *step* is one instance. We can see that *window-0* is composed by the instances in the time interval of $0 \leq t \leq 55$ (note that the sampling period of this time series is 5 min, resulting in a time interval containing 12 instances). The next iteration of the sliding window approach builds *window-1*. This window is composed of 12 instances of the X-ray time series, but now this subset begins at the second instance (time instant 5) because, in this example, the step of window shifts is set to one instance. Therefore, *window-1* includes the instances from the time interval of $5 \leq t \leq 60$ (12 instances). The same idea builds *windows 2 and 3*, as shown in Table 4.2.

Table 4.3 shows the values of X-ray intensity and their related solar flares (shown inside the parenthesis) of *windows 0 to 3*, as presented in Table 4.2.

In this example, the size of *current window*, *jump* and *future window* is set to 4 instances in 20 min. Note that the value of attribute *f1* for the first window (*window = 0*) in Table 4.3 (3.65×10^{-7} (B)) corresponds to the X-ray intensity and its solar flare, occurred in time instant 0 of Table 4.2. Similarly, the second feature *f2* equals (3.92×10^{-7} (B)) in time instant 5, and so on until the 12th value of the window is reached. *Window-1* is shown in the next line so that the first value of this window (3.92×10^{-7} (B)) is the second instance of the X-ray time series (time instant 5). This window is composed of the values of the next 12 instances of the original time series as seen in the table. The construction of *windows 2 and 3* follows the same idea and are shown in Table 4.3.

Table 4.2: Sliding window evolution performed in the execution of SS.

X-ray Time Series						
t	**x(t)**	**Solar flare class**	***window-0***	***window-1***		
0	3.65E-7	B	current window	step	***window-2***	
5	3.92E-7	B	current window	current window	step	***window-3***
10	4.09E-7	B	current window	current window	current window	step
15	4.04E-7	B	current window	current window	current window	current window
20	3.92E-7	B	jump	current window	current window	current window
25	3.94E-7	B	jump	jump	current window	current window
30	3.84E-7	B	jump	jump	jump	current window
35	3.80E-7	B	jump	jump	jump	jump
40	3.80E-7	B	future window	jump	jump	jump
45	3.83E-6	C	future window	future window	jump	jump
50	3.84E-7	B	future window	future window	future window	jump
55	3.90E-7	B	future window	future window	future window	future window
60	6.47E-7	B		future window	future window	future window
65	6.75E-7	B			future window	future window
70	5.24E-7	B				future window

Table 4.3: Example of the values of the windows built by SS

	current window				jump				future window			
window	**f1**	**f2**	**f3**	**f4**	**f5**	**f6**	**f7**	**f8**	**f9**	**f10**	**f11**	**f12**
0	3.65E-7 (B)	3.92E-7 (B)	4.09E-7 (B)	4.04E-7 (B)	3.92E-7 (B)	3.94E-7 (B)	3.84E-7 (B)	3.80E-7 (B)	3.80E-7 (B)	3.83E-6 (C)	3.84E-7 (B)	3.90E-7 (B)
1	3.92E-7 (B)	4.09E-7 (B)	4.04E-7 (B)	3.92E-7 (B)	3.94E-7 (B)	3.84E-7 (B)	3.80E-7 (B)	3.80E-7 (B)	3.83E-6 (C)	3.84E-7 (B)	3.90E-7 (B)	6.47E-7 (B)
2	4.09E-7 (B)	4.04E-7 (B)	3.92E-7 (B)	3.94E-7 (B)	3.84E-7 (B)	3.80E-7 (B)	3.80E-7 (B)	3.83E-6 (C)	3.84E-7 (B)	3.90E-7 (B)	6.47E-7 (B)	6.75E-7 (B)
3	4.04E-7 (B)	3.92E-7 (B)	3.94E-7 (B)	3.84E-7 (B)	3.80E-7 (B)	3.80E-7 (B)	3.83E-6 (C)	3.84E-7 (B)	3.90E-7 (B)	6.47E-7 (B)	6.75E-7 (B)	5.24E-7 (B)

SS uses the X-ray values of *current window* and the maximum solar flare classification of the *future window* shown in Table 4.3. The resulting SSD is shown in Table 4.4. Considering Table 4.3, the X-ray values of the *current window* of *window-0* are the values 3.65 $\times 10^{-7}$ to 4.04 $\times 10^{-7}$ of features *f1 to f4*, and the maximum solar flare among features *f9 to f12* of its *future window* is C, and they compose the *solar sequence-0* found in Table 4.4. The same principle guides the construction of *solar sequences* 1, 2 and 3.

Table 4.4: Example of a Solar Sequence Dataset (SSD)

solar sequence	f1	f2	f3	f4	Class
0	3.65E-7	3.92E-7	4.09E-7	4.04E-7	C
1	3.92E-7	4.09E-7	4.04E-7	3.92E-7	C
2	4.09E-7	4.04E-7	3.92E-7	3.94E-7	B
3	4.04E-7	3.92E-7	3.94E-7	3.84E-7	B

The SS algorithm is presented in Algorithm 6. It is fed with the X-ray time series (using the same format as Table 4.1), the *current window* size, the *step* of the consecutive windows, the *jump* and the size of the *future window*.

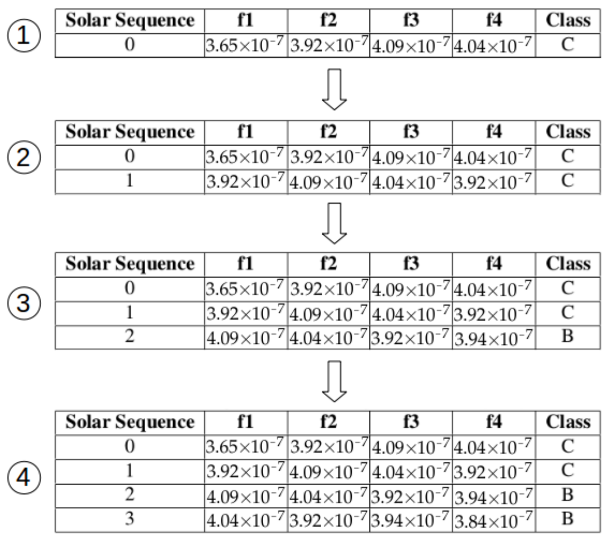
In line 5, SS builds a *window* starting in time instant *t*. The size of the window (*windowSize*) is the sum of the sizes of the *current window, the jump and the future window*. Then, in line 6, the first *c* values of the *window* is extracted and stored in an array called *currentWindow*. In line 7, the *future window* is generated using the *f* values obtained from the starting position of $c + j$ within the *window* array and stored in *futureWindow*. In line 8, *maxSolarFlareClass* receives the maximum solar flare found in the *future window* (*futureWindow*). In line 9, the *solar sequence* composed by the X-ray values of *currentWindow* and the maximum solar flare (*maxSolarFlareClass*) is built and stored in the array *solarSequence*. Line 10 adds the *solar sequence* of time instant *t* in the final set of sequences *SolarSequenceDataset*. Then, the next sequence is computed by returning the processing to line 5. Finally, line 12 returns the dataset with all the computed sequences called *SolarSequenceDataset*. The number of windows and sequences built is set in the *qty_of_windows*.

Algorithm 6: The Series to Sequence (SS) algorithm

```
input : Labeled X-ray time series $XRTS$ of size $timeSeriesSize$
        Current window: $c$
        Step: $s$
        Jump: $j$
        Future window: $f$
output: Set of Solar Sequences generated: $SolarSequenceDataset$

1  begin
2      $qty\_of\_windows \leftarrow (timeSeriesSize - (c + j + f) + 1)/s$;
3      $windowSize \leftarrow c + j + f$;
4      for $t=0$ ; $t < qty\_of\_windows$; $t+=s$ do
5          $window \leftarrow windowGenerator(XRTS, t, windowSize)$;
6          $currentWindow \leftarrow currentWindowGenerator(window, c)$;
7          $futureWindow \leftarrow futureWindowGenerator(window, c, j, f)$;
8          $maxSolarFlareClass \leftarrow searchMaxClassInFutureWindow(futureWindow)$;
9          $solarSequence \leftarrow$
             $mapCurWinAndFutSolFl(currentWindow, maxSolarFlareClass)$;
10         $add(SolarSequenceDataset, solarSequence)$;
11     end
12     return SolarSequenceDataset
13 end
```

An illustration of a hypothetical run of Algorithm 6, based on the data of Table 4.4 is shown in Figure 4.4.



Solar Sequence	f1	f2	f3	f4	Class
0	3.65×10^{-7}	3.92×10^{-7}	4.09×10^{-7}	4.04×10^{-7}	C

Solar Sequence	f1	f2	f3	f4	Class
0	3.65×10^{-7}	3.92×10^{-7}	4.09×10^{-7}	4.04×10^{-7}	C
1	3.92×10^{-7}	4.09×10^{-7}	4.04×10^{-7}	3.92×10^{-7}	C

Solar Sequence	f1	f2	f3	f4	Class
0	3.65×10^{-7}	3.92×10^{-7}	4.09×10^{-7}	4.04×10^{-7}	C
1	3.92×10^{-7}	4.09×10^{-7}	4.04×10^{-7}	3.92×10^{-7}	C
2	4.09×10^{-7}	4.04×10^{-7}	3.92×10^{-7}	3.94×10^{-7}	B

Solar Sequence	f1	f2	f3	f4	Class
0	3.65×10^{-7}	3.92×10^{-7}	4.09×10^{-7}	4.04×10^{-7}	C
1	3.92×10^{-7}	4.09×10^{-7}	4.04×10^{-7}	3.92×10^{-7}	C
2	4.09×10^{-7}	4.04×10^{-7}	3.92×10^{-7}	3.94×10^{-7}	B
3	4.04×10^{-7}	3.92×10^{-7}	3.94×10^{-7}	3.84×10^{-7}	B

Figure 4.4: Illustration of the sequential execution approach used in SS.

In the sequential execution of SS, the *window-1* construction must wait for *window-0* to be fully built. Similarly, *windows 2 and 3* are generated waiting for their respective order in the sequential queue.

Although the algorithm has complexity of order *n*, we optimized it because it can handle as many time series of solar characteristics as the expert wants. In this way, it will become more scalable yet through the parallel implementation.

For this purpose, it was developed a parallelized algorithm using *GPU* (Graphics Processing Unit) capabilities to optimize SS execution, making it more adequate for large datasets. We present the parallelization strategy used in SS in following.

Parallel Implementation of Series to Sequence (SS) Algorithm

The capabilities of *GPUs* architectures were used to distribute the parallelized part of SS into threads executed in the *CUDA* cores of an *NVIDIA* enabled graphical card. The strategy used to parallelize SS was to execute the same instructions in different datasets within parallel threads.

CUDA is a programming model that explores the computing capability of *NVIDIA GPUs* to develop solutions of computationally complex problems. This model runs only under *NVIDIA GPUs* and is consisted of programming directives that extend several sequential programming languages as C, Fortran, etc. *CUDA* provides a certain degree of scalability because it enables queueing of threads among the cores located at the *GPU*. (CORPORATION_NVIDIA, 2017). A certain parallelizable code is submitted to the *GPU* using *CUDA* when an usual C function calls a special *CUDA* function called *kernel*. This *CUDA* function triggers threads according to the developer configuration and the *GPU* card will be responsible to allocate the threads in the *GPU* cores. The execution is divided in groups of threads called *blocks*. And a set of *blocks* is called a *grid* so that the *GPU* organizes and queues each thread execution in the cores (CORPORATION_NVIDIA, 2017). A *CUDA* program is structured as follows (shown in Figure 4.5):

1. In a C function, data from the host computer is copied to the *GPU* memory, called device memory;
2. The *kernel* is triggered by the C function;
3. The *kernel* executes their threads in parallel inside its cores;

4. After all threads are finished, the resulting data are copied back to the host (computer) memory and the original C function takes control of the remaining sequential execution again.

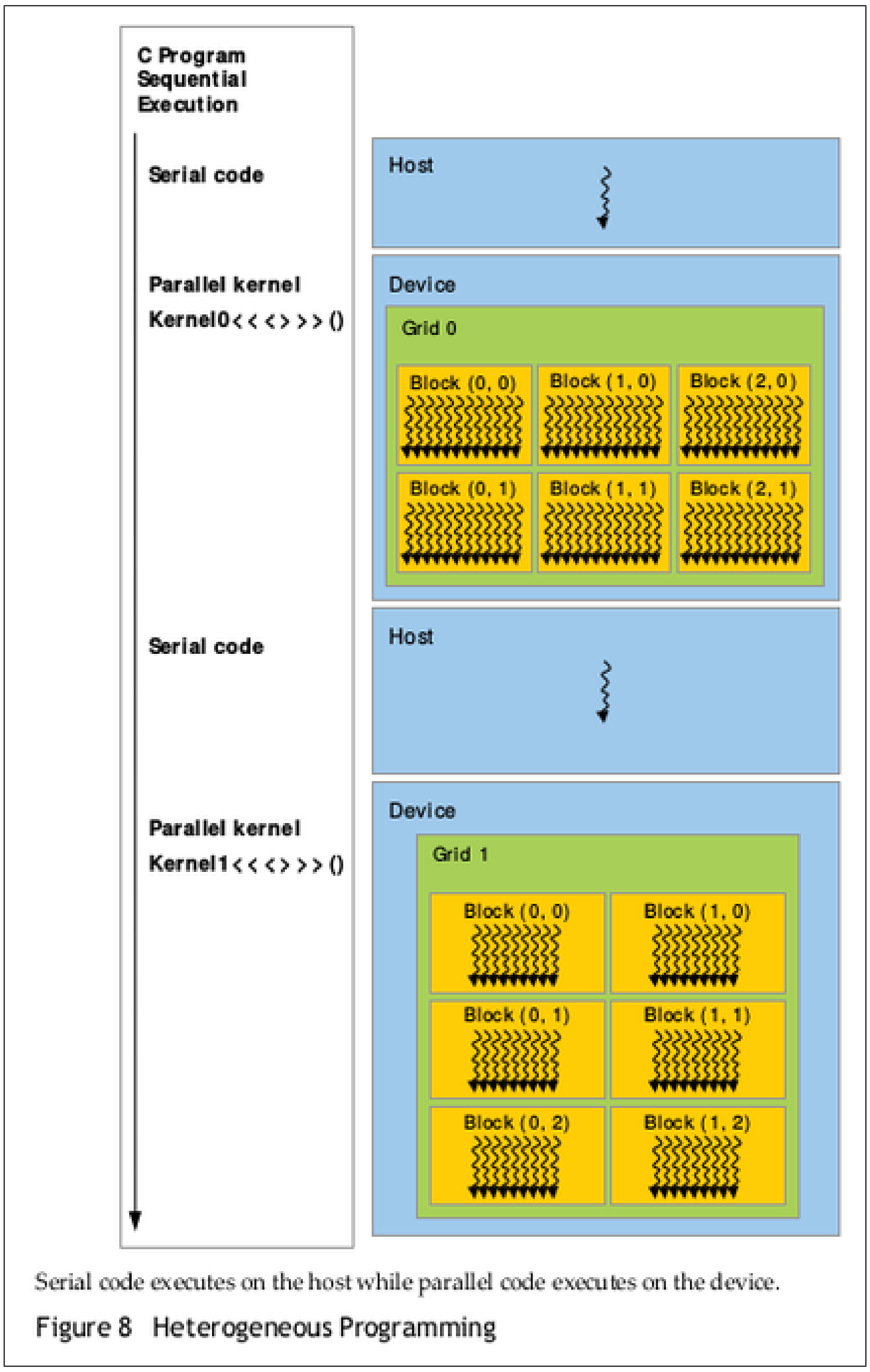
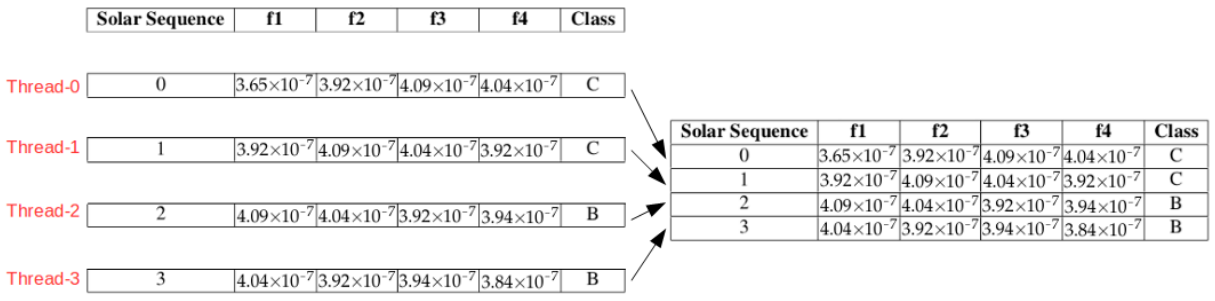


Figure 4.5: Execution Flow in GPU with CUDA - extracted from (CORPORATION_NVIDIA, 2017)

Following the schema mentioned above, the instructions of SS executed in parallel are the ones responsible for the construction of each *solar sequence*. Moreover, the dataset used is the subsets of the X-ray times series that composed the full window (*current window* + *jump* + *future window*) of each step. An hypothetical parallel execution of SS is shown in Figure 4.6.



	Solar Sequence	f1	f2	f3	f4	Class
Thread-0	0	3.65×10^{-7}	3.92×10^{-7}	4.09×10^{-7}	4.04×10^{-7}	C
Thread-1	1	3.92×10^{-7}	4.09×10^{-7}	4.04×10^{-7}	3.92×10^{-7}	C
Thread-2	2	4.09×10^{-7}	4.04×10^{-7}	3.92×10^{-7}	3.94×10^{-7}	B
Thread-3	3	4.04×10^{-7}	3.92×10^{-7}	3.94×10^{-7}	3.84×10^{-7}	B

Solar Sequence	f1	f2	f3	f4	Class
0	3.65×10^{-7}	3.92×10^{-7}	4.09×10^{-7}	4.04×10^{-7}	C
1	3.92×10^{-7}	4.09×10^{-7}	4.04×10^{-7}	3.92×10^{-7}	C
2	4.09×10^{-7}	4.04×10^{-7}	3.92×10^{-7}	3.94×10^{-7}	B
3	4.04×10^{-7}	3.92×10^{-7}	3.94×10^{-7}	3.84×10^{-7}	B

Figure 4.6: Illustration of the parallel execution approach used in SS.

Each thread executed in the *CUDA* cores of the *GPU* builds one *solar sequence* individually and in parallel so that the construction of all *solar sequences* finishes approximately at the same time. As Figure 4.6 shows, *threads 0 to 3* are executed in parallel, and the full dataset (*Solar Sequence Dataset*) is also fulfilled in parallel.

Algorithm 7 presents the *CUDA* kernel algorithm of SS.

It has as input the labeled X-ray time series, the *current window*, the *step*, the *jump* and the *future window*. The output is an array containing all the *solar sequences* called SolarSequenceDataset. This is the algorithm that the kernel function executes in *GPU*. Therefore, this algorithm is triggered in parallel as many times as the number of windows, depending on the size of the X-ray time series and the window size. The resulting dataset (SolarSequenceDataset) will be fulfilled in the global memory of the *GPU* in parallel with all the sequences (SolarSequence) built by each triggered thread. This is possible because each thread produces a unique index according to the order in which it was triggered. This index can be used to obtain the array index of SolarSequenceDataset to place the *solar sequence* built by the thread in its correct place in the array.

The threads running in the *GPU* are executed within a structure of blocks. A block, in *CUDA*, is a set of threads. The function in C that calls the *kernel* function must configure the number of threads per block, and the number of blocks to the parallel execution of SS. Thus, line 2 contains the calculation of the index of the triggered thread. This index is composed by the sum of the identifications of the triggered thread (*threadIdx.x*), its block (*blockIdx.x*) and the number of threads per block (*blockDim.x*). Line 3 shows that the array index used to fulfill the SolarSequenceDataset (*windowIndex*) is calculated by multiplying the thread index obtained in line 2 by the *step* value (*s*) (the shift from two consecutive windows). Then, in line 6, it is decided if the thread generates another *solar sequence*, depending on if the *windowIndex* has reached the predefined number of windows or not. If so, lines 7 to 11 create the *solar sequence* according to the appropriate thread index, and line 12 adds the *solar sequence* to the *SolarSequenceDataset* in the correct place of the array through the calculated *windowIndex*.

Algorithm 7: SS kernel function algorithm

```
input : Labeled X-ray time series $XRTS$ of size $timeSeriesSize$
        Current window: $c$
        Step: $s$
        Jump: $j$
        Future window: $f$
output: Set of Solar Sequences generated: $SolarSequenceDataset$
1  begin
2      $index \leftarrow threadIdx.x + blockIdx.x * blockDim.x$;
3      $windowIndex \leftarrow index * s$;
4      $qty\_of\_windows \leftarrow timeSeriesSize - (c + j + f) + 1$;
5      $windowSize \leftarrow c + j + f$;
6      if $windowIndex \leq qty\_of\_windows$ then
7          $window \leftarrow windowGenerator(XRTS, windowIndex, windowSize)$;
8          $currentWindow \leftarrow currentWindowGenerator(window, c)$;
9          $futureWindow \leftarrow futureWindowGenerator(window, c, j, f)$;
10         $maxSolarFlareClass \leftarrow searchMaxClassInFutureWindow(futureWindow)$;
11         $solarSequence \leftarrow$
             $mapCurWinAndFutSolFl(currentWindow, maxSolarFlareClass)$;
12         $add(SolarSequenceDataset, solarSequence, windowIndex)$;
13     end
14 end
```

Each *NVIDIA GPU* card has a maximum number of threads that can be executed within a block. In our case, we used the graphical card *Geforce 960X*, that can run up to 1024 threads per block. Therefore, the method was configured with 1024 threads per block, and the number of blocks was defined by Equation (4.1).

$$NumberOfBlocks = \frac{qty\_of\_windows + NUMBEROFTHREADSPERBLOCK - 1}{NUMBEROFTHREADSPERBLOCK} \quad (4.1)$$

where:

- $qty\_of\_windows = \frac{(timeSeriesSize - totalWindowSize + 1)}{step}$
- $NUMBEROFTHREADSPERBLOCK = 1024$

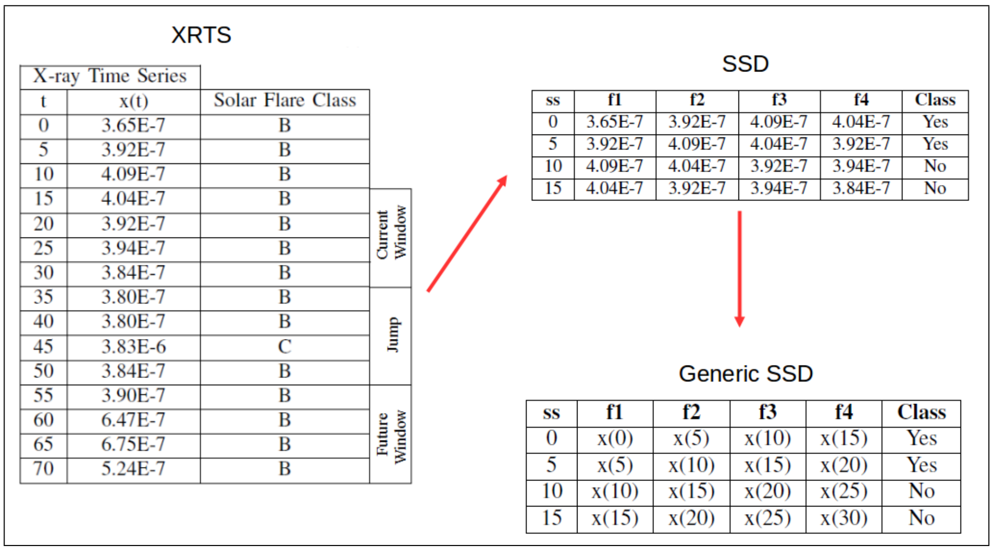
Equation (4.1) calculates the number of blocks because the number of windows may not be divisible by the *NUMBEROFTHREADSPERBLOCK* so that we have to guarantee that the threads will handle all windows.

The total number of windows that SS considers is given by subtracting the total size of the time series by the window size and dividing this result by the step of instances between two adjacent windows.

Therefore, by using Algorithm 7 to implement the *CUDA kernel* function, it was possible to parallelize the creation of the *Solar Sequence Dataset*, as shown in the example of Figure 4.6.

4.2.2 Feature Selection

In a real scenario, the preprocessed sequences usually have a large number of features. However, for large data sequences, the classification algorithms face the problem where the performance and accuracy of the algorithms degrade with the increase of the sequence size. In order to reduce the number of features and consequently to reduce the data to be submitted to the mining process, the preprocessed sequences are submitted to a feature selection process. Feature selection is used for selecting the features that mainly induces to an accurate classification. As usually, not all the features are significant for the classification, in this way, feature selection is used to generate the final dataset with the most representative features. But, what does selecting features mean in the context of SeMiner?



X-ray Time Series		
t	x(t)	Solar Flare Class
0	3.65E-7	B
5	3.92E-7	B
10	4.09E-7	B
15	4.04E-7	B
20	3.92E-7	B
25	3.94E-7	B
30	3.84E-7	B
35	3.80E-7	B
40	3.80E-7	B
45	3.83E-6	C
50	3.84E-7	B
55	3.90E-7	B
60	6.47E-7	B
65	6.75E-7	B
70	5.24E-7	B

ss	f1	f2	f3	f4	Class
0	3.65E-7	3.92E-7	4.09E-7	4.04E-7	Yes
5	3.92E-7	4.09E-7	4.04E-7	3.92E-7	Yes
10	4.09E-7	4.04E-7	3.92E-7	3.94E-7	No
15	4.04E-7	3.92E-7	3.94E-7	3.84E-7	No

ss	f1	f2	f3	f4	Class
0	x(0)	x(5)	x(10)	x(15)	Yes
5	x(5)	x(10)	x(15)	x(20)	Yes
10	x(10)	x(15)	x(20)	x(25)	No
15	x(15)	x(20)	x(25)	x(30)	No

Figure 4.7: Illustration of feature selection within SS algorithm.

According to Figure 4.7, the value of *f1* of the *sequence 0* in SSD is *x(0)* of XRTS, *f2* equals *x(5)* - the second observation occurred in instant five of XRTS, *f3* is *x(10)*, and *f4* is *x(15)*. In the *second sequence* (*ss* = *5*), *f1* is *x(5)*, *f2* = x(10), etc. If we consider *f1*, we have the values starting from time instant 0 as shown in the *Generic SSD* table. If we consider *f2*, we have the values of the time instants starting from instant 5, etc.

Thus, if the selection method chooses *f1*, for example, it means that the value of the time instant 0 is the most significant in the forecasting process. Selecting *f2*, the value of time instant 5 is another significant one, and so on. Thus, selecting features in this example means choosing the most significant time instants that distinguishes solar flares.

Our proposed method may employ any supervised feature selection algorithms to perform dimensionality reduction in the sequence data, like: Starminer (RIBEIRO et al., 2009) or Relief (KIRA; RENDELL, 1992).

Starminer (RIBEIRO et al., 2009) is a statistical association-rule based algorithm, which mines rules associating a feature f_i and a class c_k. If f_i has a uniform and a particular behavior among the instances of the c_k class in the training dataset, it produces rules of the form $f_i \rightarrow c_k$. A rule is generated if the hypothesis that the feature f_i has the same mean over class c_k and the remaining classes, are rejected. The Starminer algorithm returns the features that are presented in the mined rules as the selected ones. We chose Starminer because: 1) it produces rules relating features and the classes that they most describe, which can be validated by domain specialist; 2) it has a low computational cost when compared with most feature selection methods; 3) it selects the most relevant features and discards the irrelevant ones, not returning a ranking.

Relief (KIRA; RENDELL, 1992) is a distance-based feature selection method. A weight w_i is associated to each feature f_i. This weight indicates the relevance of the feature, and it is updated at each iteration. After a given number of iterations, the weight of each feature composes a relevance vector. An interaction consists of randomly selecting an instance x of a class c_k. The dataset instances are ranked according to the distance that they present to instance x, using the Euclidean distance. The closest *same-class* instance is called “near-hit”, and the closest *different-class* instance is called “near-miss”. The feature weight increases if its feature values differ less in the nearby instances of the same class than in the nearby instances of the other class. Even though it has a substantial computational cost, we chose Relief because it is one of the most used feature selection algorithm in the literature.

4.2.3 Mining the solar data sequences

In this step, the data sequences, considering only the relevant features mined by a feature selection approach are submitted to a classifier. A question that may arise is how traditional classification methods will be capable to predict **future** labels using the new dataset? As shown in Table 4.4, the X-ray values of the *current window* (*f1 to f4*) is being related with the maximum solar flare occurred at the *future window*. So that, when a traditional classification method is trained with that preprocessed dataset, it will make a model relating a set of X-ray values with its related future ones. Then, if we apply X-ray values collected in a snapshot not yet analyzed during the classification training phase to the learning model, the output is a future label, named class of the solar flare. In this way, the traditional classification method will be able to predict future labels using current values. It is important to emphasize that the *future window* is shifted from the *current window* by a *jump window*. In the example, the forecasting method will predict labels in advance of 4 instances. If each instance corresponds to 5 minutes, this means that the method will foresee solar flares in a forecasting horizon of 20 minutes.

Summarizing, the Solar Sequence Dataset is submitted to a feature selection method which produces the final dataset that is employed as an input of a traditional classifier. This generates a learning model which is applied to perform the forecasting using unlabeled testing time series.

4.3 Final considerations

In this chapter, we presented the SeMiner method. This first method developed is a typical time series classification method in which a set of sub-series are extracted from the original dataset and labeled. Next, a feature selection method is applied, and finally, a classifier is employed in order to generate the forecasting model. In this method, we aimed to validate the assumption of the existence of patterns in the X-ray time series. For this reason, the individual forecasting of solar flares was not yet performed. SeMiner forecasts groups of solar flares greater than and equal to class C as positive and slower events as negative. The core of the method is the Series to Sequence (SS) algorithm, which transforms windows of the original time series into labeled sequences. The assigned labels are the classes of solar flares that occurred after the observations. Thus, SS maps past observations with future events. The main characteristics of SeMiner are:

1. it uses the evolution of the solar time series in the forecasting process;
2. it incorporates the specialist knowledge in the forecasting process by its parametrization;
3. it finds the intervals of sub-series in which data may better distinguish solar flares through the use of feature selection methods.

The experiments regarding SeMiner are described in Chapter 6.

Next chapter, we present the evolution of SeMiner: the ECID method. As we show, it tackles more specifically the imbalanced data problem and performs a multi-class / multi-label forecasting.

Chapter 5

ECID Forecasting Method

SeMiner was developed to provide mechanisms to take into account the evolution of the time series in the forecasting process. It has parameters that could be tuned up by a domain specialist. And, it also identifies the time periods most related to the flare occurrences. It was the first step towards our goals. However, it still does not handle the problems of: imbalanced dataset and high similarity between data from adjacent classes.

The domain specialist do not fully understand the physical phenomena that influence the forecasting process (BOBRA; COUVIDAT, 2015). We found works that used features derived from magnetogram vector (NISHIZUKA et al., 2017; BOBRA; COUVIDAT, 2015; YU et al., 2010), sunspot area (GALLAGHER; MOON; WANG, 2002), radio flux or X-ray flux (LI; ZHU, 2013). Thus, we added more solar features in the forecasting process, intending to improve the forecasting results.

Another important issue is that solar flare datasets are extremely imbalanced. Most of the previous works of solar flare forecasting perform binary forecasting, classifying solar flare only as positive or negative. In fact, when mapping the multi-class problem to a binary-class problem, the imbalanced issue is masked. Few works predict individual classes. In the later case, they usually use purely statistical methods in the forecasting process. Some methods consider positive results for classes greater than or equal to C (AHMED et al., 2013), others consider positive for forecasts greater than or equal to M (NISHIZUKA et al., 2017; BOBRA; COUVIDAT, 2015; LI; ZHU, 2013; YU et al., 2010). Additionally, as Table 3.1 shows, there is a lack of time series classification methods that deal with multi-class classification in imbalanced datasets.

Also, solar flare forecasting may become a multi-label problem because during a given day it may occur solar flares of classes C, M and X. This is another gap found in literature because current works that use data mining classification methods usually do not provide multi-label forecasting.

An alternative to handle the poor results of the learning in imbalanced data and also produce a multi-label forecasting method is using an Ensemble of Classifiers (EC) (SAGI; ROKACH, 2018; GALAR et al., 2012; RÄTSCH; ONODA; MÜLLER, 2001). The EC main goal is to improve weak classification methods by applying several weak classifiers (also called base inducers) and combining their results. This goal enables EC to produce accurate methods resulting in multi-label classification in imbalanced domains. In this sense, we propose an EC tuned-up for the domain of the solar flare forecasting.

In this context, we developed a method called ECID (Ensemble of Classifiers for Imbalanced Datasets) that uses an extended version of SS algorithm to tackle the open issues identified in this thesis. So, the main purpose of ECID was to tackle the following gaps:

1. The lack of methods that handle the evolution of the solar time series in the forecasting process through the extended version of SS algorithm;
2. The lack of methods that incorporate the specialist knowledge in the forecasting process by configuring SS;
3. The lack of methods that perform individual class forecasting producing a multi-class result for a given day, so that the method provides to the astrophysicist a tool that shows possible solar flare categories that may happen in a given day;
4. The lack of methods that treat the imbalanced dataset issue. For this purpose, ECID uses an Ensemble of Classifiers with a stratified random sampling for the training of the inducers;
5. The lack of methods that perform a multi-label solution, giving alternatives to the astrophysicist to decide the final forecasting when possible adjacent classes are predicted.

5.1 First considerations

ECID uses the historical evolution of solar time series and the specialist knowledge to its setup in order to provide a multi-class and multi-label forecasting. This way, it handles the issues of imbalanced dataset and the high similarity between data from adjacent classes. Figure 5.1 presents an overview of the method.

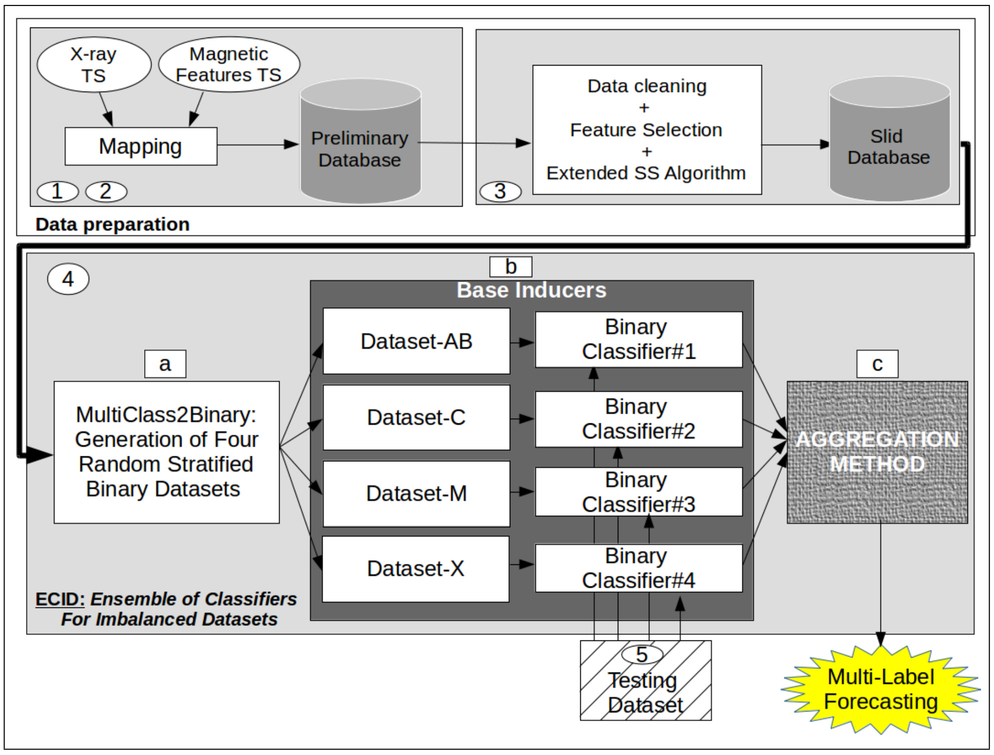


Figure 5.1: Overview of ECID and its pre-processing.

Steps 1, 2 and 3 are performed to prepare the data sources to be submitted to ECID. They are responsible for obtaining, cleaning and transforming the data for the learning task. Traditional classification methods produce models that classify instances with **current events**. For the forecasting purpose, it is necessary to map current values with future events to turn such classifiers in forecasting methods. In this stage, the method prepares the original dataset by using an extended version of the SS algorithm. This extension was needed in order to use more than one solar feature as the forecasting method input, differently from the original version of SS.

Solar data is highly imbalanced because the highest solar flares are extremely infrequent. Table 5.1 shows the class distribution of a solar dataset processed by SS (data collected from 2010 to 2017).

Table 5.1: Class distribution in a solar dataset slid by SS

Class	Number of Instances	% of Instances
AB	33839	50
C	23633	35
M	8968	13
X	1238	2

Thus, we faced the problem of highly imbalanced dataset classification. The original versions of time series classification methods were not designed to deal with imbalanced datasets, being unsuitable for treating imbalanced data. In this way, we applied to the time series classification methods some balancing strategies and an Ensemble of Classifiers to handle imbalanced datasets as explained in Sections 2.3 and 2.4.

In this way, the prepared data are submitted to the proposed ensemble method ECID. The issue of imbalanced data is tackled by employing a strategy based on Bagging (see Section 2.4.5). The difference is that, in our approach, we use stratified random sampling to produce four datasets of different sizes, which are used to train the base inducers. The original version of Bagging, however, generates all the input datasets with the same size.

Thus, the original dataset is divided into four balanced subsamples, one for each forecasting class, using the new MultiClass2Binary Balancing strategy (see Figure 5.1-Step 4(a)). This strategy produces a dataset specifically for the forecasting of class AB, another for C, another for M, and the last for class X. As the original dataset is imbalanced, the dataset for class X has fewer instances compared with the dataset used for class AB, C or M. This schema aims to give more weight to the instances that belong to rarest classes.

The base inducers are weak binary classifiers (see Figure 5.1 - Step 4(b)): the Binary Classifier#1 generates a positive result if a class A or B is predicted by the model and negative, otherwise. Binary Classifier#2 generates a positive result if a class C is predicted and negative, otherwise. Finally, the same logic is applied for classes M and X. The outputs of the base inducers are submitted to the Aggregation Method (see Figure 5.1 - Step 4(c)), which combines the individual votes of each inducer, producing the final multi-class forecasting. The forecasting produced in this step is multi-class.

If more than one class has enough votes, the result is also multi-label, because the Aggregation Method returns more than one class. One characteristic of solar events is that the impact over electronic devices on Earth of a C9.0 solar flare is quantitatively 1,0% greater of a M1.0, while a C9.9 solar flare is quantitatively 10 times greater than a C1.0 flare. In this way, we considered that solar flare C9.9 has similar impacts compared with M1.0 ones. Hence, as the method provides a multi-label result, the astrophysicist may use the method output in order to decide the best classification of the predicted solar flare. For example, if the method returns CM as the forecasting result, it means that probably the solar flare is about C9.0 to M1.0. Differently, if the method returns just C, it will probably indicate to the astrophysicist that an event smaller than a class M may occur. So, the method provides a mechanism to support the astrophysicist in deciding which intensity the solar flare will be. A detailed description of the steps of ECID is given as follows.

5.2 ECID method description

As Figure 5.1 shows, there is initially a data preparation phase that precedes the ECID method. In this phase, the input features from different sources are handled, cleaned, selected and submitted to the extended SS to map past observations to their following events. This step handles:

1. the evolution of the solar feature time series in the forecasting process;
2. the incorporation of the specialist knowledge in the forecasting process;

In the next phase, ECID splits the slid dataset provided by Step-3 into four datasets, one for each class: AB, C, M and X in order to implicitly assign different weights for each class according to their distribution in the original slid dataset. ECID applies an Ensemble of Classifiers which may use any base inducers. The choice of which inducer to use is made empirically. Finally, an aggregation method is used to decide which classes may be assigned to the final forecasting according to a decision rule based on the voting given for each inducer. The output is multi-class and multi-label, since it may contain more than one (of four) class in the final forecasting. Consequently, ECID also handles the following requirements of this thesis:

1. it performs individual class forecasting producing a multi-class result for a given day so that the method provides to the astrophysicist a tool that shows possible solar flare categories that may happen in a given day;
2. it treats the imbalanced dataset issue. For this purpose, ECID uses an EC with a stratified random sampling for training the inducers;
3. it is multi-label, giving to the astrophysicist the possibility of deciding between adjacent classes.

In the next sections, a detailed description of ECID method is presented.

5.2.1 Data preparation

ECID uses two time series of solar features: the time series of X-ray intensity emitted by the Sun, and the time series of magnetic features provided from the instrument SDO/HMI. It also uses a list of solar events in order to compose the dataset used to train and test the model. Next, we present the main information provided by these time series and the list of events. We also describe the preparation steps of the input data required by ECID.

The sampling periods of X-ray time series (XRTS) are of 1 or 5 minutes. The 5-minute series is the average of the series of sampling periods of 1 minute. The unit of measure of X-ray intensity is W/m^2 and X-ray values are collected according to the sampling period. Its values correspond to the integration of the X-rays emitted by the whole solar circle. Table 5.2 contains an example of the format of this time series.

Table 5.2: X-ray time series (XRTS): Instant format: YYYY-MM-DD_HH:Min, and X-ray observations $\in R$

Instant(YYYY-MM-DD_HH:Min)	X-ray observations (W/m^2)
2017-09-07_05:00	1.5E-06
2017-09-07_05:01	1.1E-05
2017-09-07_05:02	2.4E-05
2017-09-07_05:03	2.2E-05
2017-09-07_05:04	1.8E-05
2017-09-07_05:05	1.6E-05
2017-09-07_05:06	1.4E-05
2017-09-07_05:07	1.3E-05
2017-09-07_05:08	1.1E-05
2017-09-07_05:09	1.0E-05
2017-09-07_05:10	9.4E-06
2017-09-07_05:11	8.4E-06
2017-09-07_05:12	7.6E-06
2017-09-07_05:13	6.8E-06
2017-09-07_05:14	6.2E-06
2017-09-07_05:15	5.8E-06
2017-09-07_05:16	5.6E-06
2017-09-07_05:17	5.2E-06
2017-09-07_05:18	4.7E-06
2017-09-07_05:19	4.2E-06
2017-09-07_05:20	3.7E-06
2017-09-07_05:21	3.4E-06
2017-09-07_05:22	3.1E-06
2017-09-07_05:23	2.8E-06

For example, at instant 2017-09-07_05:02, the integration of the X-rays emitted by the whole solar circle was 2.4E-05 W/m^2.

The second time series used was about the magnetic features collected from the Photosphere of Sun. On the JSOC website, there are time series of magnetic features of different sampling periods from 45 or 720 seconds (12 minutes). For the selection of the Magnetic Feature Time

Series (MFTS), we based on the experiments carried out in (BOBRA; COUVIDAT, 2015), so that we used the time series of sampling period of 12 minutes. This work guided us in this choice due to the results obtained and also by the fact that this series comes from new magnetic reading equipment, the SDO/HMI.

MFTS provides observations of the magnetic features of each active region of the solar circle every 12 minutes. Unlike XRTS, which provides the integration of X-rays emitted by the **whole** solar circle. Figure 2.8 shows the list of magnetic features provided by MFTS. Table 5.3 contains an example of this time series.

Table 5.3: Magnetic Feature Time Series (MFTS) - NOAA_AR: Active Region ID; T_REC: Sampling time instant; USFLUX and R_VALUE: magnetic features

NOAA_AR	T_REC(YYYY.MM.DD_HH:Min)	USFLUX	R_VALUE
12673	2017.09.07_05:00	5.4e+22	5160
12673	2017.09.07_05:12	5.4e+22	5137
12673	2017.09.07_05:24	5.5e+22	5136
12673	2017.09.07_05:36	5.4e+22	5163
12673	2017.09.07_05:48	5.5e+22	5177
12674	2017.09.07_05:00	3.8e+22	4292
12674	2017.09.07_05:12	3.8e+22	4365
12674	2017.09.07_05:24	3.8e+22	4295
12674	2017.09.07_05:36	3.8e+22	4263
12674	2017.09.07_05:48	3.8e+22	4226

Observe that the MFTS exemplified in Table 5.3 contains information of two active regions (AR) identified as 12673 and 12674. AR-12673 contains five instances sampled at 12-minute intervals and information about two magnetic features identified as USFLUX and R_VALUE. By the arrangement of this time series, it is noted that it is composed of a set of sub-series, one for each AR. Therefore, at a given instant, it is possible to have more than one instance in MFTS. As an example, the instant 2017.09.07_05:00 contains instances from two active regions. Thus, this instant has more than one instance in the MFTS.

Another data used in ECID is the List of Solar Events (LSE). NOAA provides a daily LSE that contains the period, class, and region of the recorded event. Table 5.4 contains an example of this list.

Table 5.4: Solar event list example: Edited Events for 2017 Sep 07.

event	begin	max	end	obs	q	type	loc_frq	class	intensity	reg
7580	459	502	508	G15	5	XRA	1-8A	M2.4	7.3E-03	2673
7600	619	628	642	G15	5	XRA	1-8A	C8.2	8.4E-03	2673
7640	916	920	924	G13	5	XRA	1-8A	C2.3	7.0E-04	2673
7660	949	954	958	G15	5	XRA	1-8A	M1.4	4.0E-03	2673
7680	1011	1015	1018	G15	5	XRA	1-8A	M7.3	1.4E-02	2673
7780	1420	1436	1455	G15	5	XRA	1-8A	X1.3	1.2E-01	2673
7830	1804	1828	1836	G15	5	XRA	1-8A	C5.2	7.8E-03	2673
7840	1840	1844	1849	G15	5	XRA	1-8A	C4.5	1.8E-03	2673
7860	2054	2057	2059	G13	5	XRA	1-8A	C2.5	4.3E-04	2673
7870	2124	2137	2147	G15	5	XRA	1-8A	C5.4	5.6E-03	2673
7910	2257	2300	2302	G15	5	XRA	1-8A	C2.7	4.4E-04	2677
7940	2350	2359	14	G15	5	XRA	1-8A	M3.9	3.6E-02	2673

The above example is from a list of events that occurred on 07-Sep-2017 provided by NOAA. Each line of the table is a recorded solar event, which is identified by the *event* attribute. The *begin, max, end* attributes contain the instants of start, maximum, and end of the events. The *obs* attribute refers to the name of the satellite that recorded the event. The *q* attribute corresponds to the quality of the information, where 1 means low and 5, excellent quality. The *type* attribute (in which all lines are set as XRA) shows the type of event recorded. It its value is “XRA”, the event is a solar flare. The *loc/frq* attribute refers to the passband of the sensor that has read the X-ray. The *particulars* and *intensity* attributes list the class of the solar flare and its level. Finally, the *reg* attribute consists of the identifier of the AR where the event occurred.

In order to use these three sources of solar information: XRTS, MFTS, and LSE, it is necessary to clean their anomalies. This cleaning is performed during the pre-processing phase in different steps as we detail later. The following anomalies were found in these data sources:

- Erroneous values sampled in the observations of MFTS, usually set to NaN (Not a Number);
- No ARs assigned to some solar flares in LSE;
- Instances not sampled in MFTS: as the sampling period of MFTS is 12 minutes, the sampling of one hour corresponds to five observations of each feature. The data contains

several missing sampled instances. This fact generates less than five observations per hour;

- Erroneous values in XRTS, usually set to a negative number;

XRTS, MFTS, and LSE are used as input data in the solar flare forecasting method ECID. For this purpose, we applied the following steps to the time series aiming to obtain a unique time series:

- (a) As XRTS and MFTS have different sampling periods, initially it may create a unique time series through the equalization of these sampling periods. This new time series is called Initial Equalized Time Series (IETS);
- (b) The next step is responsible for labeling each instance of IETS with the solar flare class listed in LSE. For this purpose, the algorithm searches, in the LSE list, the event occurred and labels IETS instances with the event found. The resulting time series is called Labeled Equalized Time Series (LETS);
- (c) Each instant may contain more than one instance in LETS, but the developed method in this thesis requires a time series with one entry per time. Thus, it is necessary to obtain a unique instance for each instant of observation. This new time series is called Preliminary Database (PD);
- (d) PD still have anomalous data, which is handled in this step.

Step-1a: Equalization of the sampling periods of the XRTS and MFTS time series

The method uses time series of one or more solar features as input data. Time series of X-ray intensity and time series of magnetic features are used as input to the ECID method.

However, the time series have different sample periods. For example, the sampling period of time series of X-rays (XRTS) is 1 or 5 minutes. The time series of magnetic features (MFTS) used in this work has a sampling period of 12 minutes. On the other hand, the ECID method can work only with time series of the same sampling period. Therefore, to work with the series of X-rays in conjunction with the MFTS, it is necessary to equalize the sampling periods of both series, generating a new equalized time series.

We mapped the observations of the original series into a single equalized series. The equalization of sampling periods was done through the insertion of samples related to the series of lower sampling period in the ones with the highest sampling period. As an example, we inserted observations of the series of X-rays (of lower sampling period) in instances of the series of magnetic features (of higher sampling period).

Table 5.5: Initial Equalized Time Series (IETS): XRTS and MFTS equalization.

NOAA_AR	T_REC(YYYY.MM.DD_HH:Min)	USFLUX	R_VALUE	RX_0	RX_1	RX_2	RX_3	RX_4	RX_5	RX_6	RX_7	RX_8	RX_9	RX_{10}	RX_{11}
12673	2017.09.07_05:00	5.4e+22	5160	1.5E-06	1.1E-05	2.4E-05	2.2E-05	1.8E-05	1.6E-05	1.4E-05	1.3E-05	1.1E-05	1.0E-05	9.4E-06	8.4E-06
12673	2017.09.07_05:12	5.4e+22	5137	7.6E-06	6.8E-06	6.2E-06	5.8E-06	5.6E-06	5.2E-06	4.7E-06	4.2E-06	3.7E-06	3.4E-06	3.1E-06	2.8E-06
12674	2017.09.07_05:00	3.8e+22	4292	1.5E-06	1.1E-05	2.4E-05	2.2E-05	1.8E-05	1.6E-05	1.4E-05	1.3E-05	1.1E-05	1.0E-05	9.4E-06	8.4E-06
12674	2017.09.07_05:12	3.8e+22	4365	7.6E-06	6.8E-06	6.2E-06	5.8E-06	5.6E-06	5.2E-06	4.7E-06	4.2E-06	3.7E-06	3.4E-06	3.1E-06	2.8E-06

Table 5.2 shows the XRTS with its sampled observations every one minute. Table 5.3 shows the MFTS with the respective sampled observations every 12 minutes. Note that the equalized series, shown in Table 5.5, is the union of the MFTS with the twelve observations relating to its sampling period. For example, at time 2017.09.07_05:00, the method linked the magnetic features $USFLUX$ and R_VALUE sampled at this time, with the X-ray observations registered from 2017.09.07_05:00 to 2017.09.07_05:12 denoted by RX_0 to RX_{11}, since the number of attributes added to MFTS is equal to its sampling period, i.e., 12. At time 2017.09.07_05:12, the magnetic characteristics of this instant were joined to the following 12 observations of XRTS. Note in the example that the X-ray values sampled from 2017.09.07_05:00 to 2017.09.07_05:12 were filled with the same values at each time regardless of the AR considered.

Formally, we have:

- The X-ray time series is defined as XRTS = {instant_of_X-ray_obs, X-ray_intensity}, where:
  - instant_of_X-ray_obs: instant of the X-ray observation;
  - X-ray_intensity: X-ray value observed at instant_of_X-ray_obs.
- The Magnetic Features time series is defined as MFTS = {instant_of_MF_obs, MF_0, MF_1, .., MF_n}, where:
  - instant_of_MF_obs: instant of the magnetic features observations;
  - MF_0 to MF_n: the n_{th} magnetic features.
- The list of solar event list is defined as LSE = {period_of_the_event, class_of_the_event}, where:
  - period_of_the_event: period of the recorded solar event ;
  - class_of_the_event: class of the solar flare occurred at period_of_the_event.

- The resulting Initial Equalized Time Series is defined as IETS = {t, MF_0, MF_1, .., MF_n,XR_t, XR_{t+1},.., XR_{t+i}}, where
  - $i = SamplePeriod\_MFTS/SamplePeriod\_XRTS - 1$;
  - t = highest_SR_TS_instant = instant of the time series with the highest sampling period (in the previous example, *t = 12*.

This strategy allows us to calculate for each instant of the equalized series statistical measures of X-ray values, that can aid in the process of solar flare forecasting. Among others, it is possible to calculate average, median, minimum, maximum or amplitude of variation. Such metrics can identify trends that are important for the forecasting process. For this reason, this strategy of sampling period equalization was chosen. Thus, we obtained the new time series called Initial Equalized Time Series (IETS) by equalizing the sampling periods of the XRTS and MFTS.

The next step of the method labels IETS with the related solar flare.

Step-1b: IETS labeling

The algorithm looks in LSE the event occurred for each AR of IETS and labels each corresponding instant with the solar flare class encountered. For example, the first instance of IETS corresponds to the sampled observations of AR-12673 at time 2017.09.07_05:00. We can see, in the listing given by Table 5.4, that a class M2.4 solar flare, numbered 7580, was registered in AR-2673 from 04:59 and 05:08. There is a difference in the numbering of ARs between the reports: the same AR is identified by 12673 in IETS and 2673 in LSE, but it is the same AR. Thus, the algorithm labels the first instance of IETS with class M2.4. This strategy is implemented for all IETS instances. If the algorithm does not find an associated event, the given label is AB, denoting that there was no solar flare in that AR/instant. The resulting time series is named Labeled Equalized Time Series (LETS). An example of LETS is presented in Table 5.6.

Table 5.6: Labeled Equalized Time Series (LETS)

NOAA_AR	T_REC(YYYY.MM.DD_HH:Min)	USFLUX	R_VALUE	RX_0	RX_1	RX_2	RX_3	RX_4	RX_5	RX_6	RX_7	RX_8	RX_9	RX_{10}	RX_{11}	class
12673	2017.09.07_05:00	5.4e+22	5160	1.5E-06	1.1E-05	2.4E-05	2.2E-05	1.8E-05	1.6E-05	1.4E-05	1.3E-05	1.1E-05	1.0E-05	9.4E-06	8.4E-06	M2.4
12673	2017.09.07_05:12	5.4e+22	5137	7.6E-06	6.8E-06	6.2E-06	5.8E-06	5.6E-06	5.2E-06	4.7E-06	4.2E-06	3.7E-06	3.4E-06	3.1E-06	2.8E-06	AB
12674	2017.09.07_05:00	3.8e+22	4292	1.5E-06	1.1E-05	2.4E-05	2.2E-05	1.8E-05	1.6E-05	1.4E-05	1.3E-05	1.1E-05	1.0E-05	9.4E-06	8.4E-06	AB
12674	2017.09.07_05:12	3.8e+22	4365	7.6E-06	6.8E-06	6.2E-06	5.8E-06	5.6E-06	5.2E-06	4.7E-06	4.2E-06	3.7E-06	3.4E-06	3.1E-06	2.8E-06	AB

Formally, we have:

- The Labeled Equalized Time Series is defined as LETS = U{IETS(t), class_of_the_event_given_by_LSE}, where:

– IETS(t) is the IETS instance in a certain instant *t*;

– class_of_the_event_given_by_LSE is the class of the event found in LSE for the instant *t*;

– LETS is the union of all instances of IETS in a certain period of time.

Besides the IETS labeling, this step of the method also performs a data cleaning task: all the anomalous values of the X-ray observations are set to NaN (Not a Number), which is a numeric data type value representing an undefined value.

Additionally, we observe that the LETS series may contain multiple instances for the same instant, but ECID can only have as input a series with unique instances for each instant. Therefore, it is necessary to unify different instances of the same instant, which is performed in the next step.

Step-2: Unification of different instances of LETS of a unique instant

The strategy used to unify different instances of LETS is to provide a series with the minimum, maximum and average of each feature. Let us consider only the USFLUX feature for exemplification purposes. At time 2017.09.07_05:00, the algorithm extracts the minimum, maximum and average of this feature. For the *class* attribute, it is generated only the minimum and the maximum, since it is typified as categorical. Table 5.7 shows the resulting series with unified instants for USFLUX and Class. The same strategy is applied to all attributes of LETS.

Table 5.7: Unified LETS: Preliminary Database (PD)

T_REC(YYYY.MM.DD_HH:Min)	**$\text{USFLUX}_{\text{MIN}}$**	**$\text{USFLUX}_{\text{MAX}}$**	**$\text{USFLUX}_{\text{AVG}}$**	**$\text{CLASS}_{\text{min}}$**	**$\text{CLASS}_{\text{MAX}}$**
2017.09.07_05:00	3.809E+25	5.42E+22	1.907E+25	AB	M2.4
2017.09.07_05:12	3.806E+25	5.43E+22	1.906E+25	AB	AB

Summarizing, in a given instant, for each LETS feature, it is created the minimum, maximum and average, as well as the minimum and maximum of the class. Table 5.7 shows only those attributes for the USFLUX and the Class. The attribute *NOAA_AR* of LETS was excluded because the new unified series is bringing compiled information by time.

Formally, we have:

- The Preliminary Database is defined as PD = $\text{U}\{\text{t}, \text{MF}0_{\text{min}}, \text{MF}0_{\text{max}}, \text{MF}0_{\text{avg}}, \text{MF}1_{\text{min}}, \text{MF}1_{\text{max}}, \text{MF}1_{\text{avg}}, .., \text{MFn}_{\text{min}}, \text{MFn}_{\text{max}}, \text{MFn}_{\text{avg}}, \text{XR}_{\text{t}_{\text{min}}}, \text{XR}_{\text{t}_{\text{max}}}, \text{XR}_{\text{t}_{\text{avg}}},.., \text{XR}_{\text{t+i}_{\text{min}}}, \text{XR}_{\text{t+i}_{\text{max}}}, \text{XR}_{\text{t+i}_{\text{avg}}}$, class_of_the_event_given_by_SEL$_{\text{min}}$, class_of_the_event_given_by_SEL$_{\text{max}}\}$, where:

- t = highest_SR_TS_instant = instant of the time series with the highest sampling period;
- $\mathrm{MF}_{\mathrm{j_{min}}}$, $\mathrm{MF}_{\mathrm{j_{max}}}$, $\mathrm{MF}_{\mathrm{j_{avg}}}$ = minimum, maximum and average of the magnetic feature in the instant t of LETS; j is the j_{th} magnetic feature. The total quantity of magnetic features is n;
- $\mathrm{XR}_{\mathrm{t_{min}}}$, $\mathrm{XR}_{\mathrm{t_{max}}}$, $\mathrm{XR}_{\mathrm{t_{avg}}}$ = minimum, maximum and average of the X-ray in the instant t of LETS; t is the t_{th} x-Ray from LETS and its highest value is the sample period of MFTS divided by the sample period of XRTS, given by i;
- class_of_the_event_given_by_SEL$_{\mathrm{min}}$, class_of_the_event_given_by_SEL$_{\mathrm{max}}$ = minimum and maximum class in the instant t of LETS;

The resulting values for the minimum, maximum, and average involving operations with feature values NaN were also set to NaN. The series produced in this step was named Preliminary Database (PD).

Step-3a: Data cleaning

Among the anomalies mentioned, the only untreated ones, until this step, were some missing instances of XRTS or MFTS. Thus, in this step, instances with value NaN were inserted in all the gaps of PD. This is because the SS algorithm requires a series of input with a fixed sampling period.

Step-3b: Feature Selection

Bobra e Couvidat (2015) lists a set of 25 magnetic features possibly able to distinguish solar flares. Thus, we applied a feature selection task over PD in order to filter the most important features to use in ECID. We used a ranking feature selection method and performed experiments using the top three ranked. As we will show in Chapter 6, the top three features were: $\mathrm{XR}_{\mathrm{max}}$, R_VALUE$_{\mathrm{max}}$ and USFLUX$_{\mathrm{max}}$.

Step-3c: Application of SS into PD

In Step-3c, an extended version of Series to Sequence (SS) algorithm was developed to handle multidimensional time series. This version is presented in Algorithm 8. The strategy used to extend the algorithm was to compose the *extended current window* to a set of windows, one for each feature and label the resulting instance with the maximum class found in the future window.

Algorithm 8: The Series to Sequence (SS) algorithm: Extended Version

```
input : Preliminary Database $PD$ of size $timeSeriesSize$
        Current window: $c$
        Step: $s$
        Jump: $j$
        Future window: $f$
output: $SlidDataset(SD)$
1  begin
2      $qty\_of\_windows \leftarrow (timeSeriesSize - (c + j + f) + 1)/s$;
3      $windowSize \leftarrow c \times |solarFeatures| + j + f$;
4      for $t=0; t < qty\_of\_windows; t+=s$ do
5          for $i = 0; i < |solar features of PD|; i++$ do
6              $extendedcurrentWindowofFeature_i \leftarrow$
                 $extendedcurrentWindowGenerator(SD.solarFeature_i, t, c)$;
                 $add(extendedcurrentWindow, extendedcurrentWindowofFeature_i)$;
7          end
8          $window \leftarrow windowGenerator(SD, extendedcurrentWindow, t, windowSize)$;
9          $futureWindow \leftarrow futureWindowGenerator(window, c, j, f)$;
10         $maxSolarFlareClass \leftarrow searchMaxClassInFutureWindow(futureWindow)$;
11         $solarSequence \leftarrow mapCurWinAndFutSolFl(currentWindow, maxSolarFlareClass)$;
12         $add(SD, solarSequence)$;
13     end
14     return SD
15 end
```

As presented in Algorithm 8, SS is fed with *Preliminary Database*, the *current window size*, the *step* of the consecutive windows, the *jump* and the *size of the future window*.

In line 5 to 7, the *extended current window* is created. The subseries of size c of each solar feature, starting in *instant t*, is added to the *extendedcurrentWindow* array. Then, in line 8, a *window* array, composed of the *extended current window*, the *jump* and the *future window* is created. In line 9, the *future window* is extracted from the *window* array using the f values obtained from the starting position of $c + j$ of this array, and stored in the *futureWindow* array. In line 10, *maxSolarFlareClass* receives the maximum solar flare found in *futureWindow*. In line 11, the *Solar Sequence* composed of the sub-series of *currentWindow* and the maximum solar flare (*maxSolarFlareClass*) is built and stored in the array *solarSequence*. Line 12 adds the *Solar Sequence* of time instant t in the final set of sequences *Slid Dataset (SD)*. Then, the next sequence is computed by returning the processing to line 5. Finally, line 14 returns the dataset with all the computed sequences called *Slid Database*. The number of windows and sequences built is set in the *qty_of_windows*.

To better understand the extended version of SS algorithm, consider Table 5.8 as the Slid Database (SD) obtained. The values are not shown in the table due to space limitations. As presented in Table 5.8, the attributes named USFLUX$_{\text{MIN-0}}$ to USFLUX$_{\text{MIN-CWS}}$ (where CWS means the current window size) are the attributes of the *current window* of the feature USFLUX$_{\text{MIN}}$ of Table 5.7. The attributes named USFLUX$_{\text{MAX-0}}$ to USFLUX$_{\text{MAX-CWS}}$ are the attributes of the *current window* of the feature USFLUX$_{\text{MAX}}$. Additionally, the attributes named USFLUX$_{\text{AVG-0}}$ to USFLUX$_{\text{AVG-CWS}}$ are the attributes of the *current window* of USFLUX$_{\text{AVG}}$. Finally, the attribute Class_of_future_Window is the class of the future window as described in Section 4.2. Here, we used just the attribute CLASS$_{\text{MAX}}$ of PD. Note that, for exemplification purposes, we only show the attributes regarding USFLUX, but in the real situation, this strategy is applied considering all the magnetic features selected in the feature selection step.

Table 5.8: Slid Database (SD)

	Extended Current Window									
T_REC	**USFLUX$_{\text{MIN-0}}$**	**...**	**USFLUX$_{\text{MIN-CWS}}$**	**USFLUX$_{\text{MAX-0}}$**	**...**	**USFLUX$_{\text{MAX-CWS}}$**	**USFLUX$_{\text{AVG-0}}$**	**...**	**USFLUX$_{\text{AVG-CWS}}$**	**Class_of_future_window**
...	...	...	...	...	...	...	...	...	...	...
...	...	...	...	...	...	...	...	...	...	...
...	...	...	...	...	...	...	...	...	...	...

Formally, we have:

- $|PD|$ is the number of instances of the Preliminary Database, PD;
- $SD = \{MF_{j_{min\text{-}(t)}}(i), MF_{j_{min\text{-}(t+1)}}(i), ..., MF_{j_{min\text{-}(t+CWS)}}(i),$
  $MF_{j_{max\text{-}(t)}}(i), MF_{j_{max\text{-}(t+1)}}(i), ..., MF_{j_{max\text{-}(t+CWS)}}(i),$
  $MF_{j_{avg\text{-}(t)}}(i), MF_{j_{avg\text{-}(t+1)}}(i), ..., MF_{j_{avg\text{-}(t+CWS)}}(i),$
  $XR_{k_{min\text{-}(t)}}(i), XR_{k_{min\text{-}(t+1)}}(i), ..., XR_{k_{min\text{-}(t+CWS)}}(i),$
  $XR_{k_{max\text{-}(t)}}(i), XR_{k_{max\text{-}(t+1)}}(i), ..., XR_{k_{max\text{-}(t+CWS)}}(i),$
  $XR_{k_{avg\text{-}(t)}}(i), XR_{k_{avg\text{-}(t+1)}}(i), ..., XR_{k_{avg\text{-}(t+CWS)}}(i),$
  $maximumClassOfFutureWindow, 1 \leq i \leq |PD| - windowsize,$
  $0 \leq t \leq i \times highestsamplePeriod\}$, where
  - SD is the Slid Database;
  - $MF_{j_{min\text{-}(t)}}(i)$ is the lowest value of a j_{th} magnetic feature in instant t;
  - $MF_{j_{max\text{-}(t)}}(i)$ is the highest value of a j_{th} magnetic feature in instant t;
  - $MF_{j_{avg\text{-}(t)}}(i)$ is the average value of a j_{th} magnetic feature in instant t;

- $XR_{k_{min\text{-}(t)}}(i)$ is the lowest value of a k_{th} X-ray intensity in instant *t*;
- $XR_{k_{max\text{-}(t)}}(i)$ is the highest value of a k_{th} X-ray intensity in instant *t*;
- $XR_{k_{avg\text{-}(t)}}(i)$ is the average value of a k_{th} X-ray intensity in instant *t*;
- CWS is the defined size of the current window;
- *windowsize* is the defined size of the window;
- $maximumClassOfFutureWindow$ is the maximum solar flare class occurred in the *future window* (for a detailed explanation of current and future window, see Section 4.2).

At this point, the database is already prepared to be handled by ECID as presented in following.

5.2.2 The proposed method ECID (Ensemble of Classifiers for Imbalanced Datasets)

The preprocessed data is submitted to ECID that employs a modified bootstrap strategy: ECID builds multiple learning models from different subsamples of the training dataset. Specifically, it splits the training dataset into subsamples using a strategy that we named *MultiClass2Binary*.

MultiClass2Binary strategy is the key to provide multi-class forecasting for four classes of solar flares (AB, C, M and X). Here, we grouped instances classified as A and B, because these type of solar flares does not impact Earth's equipment, so that distinguishing them is not interesting for the astrophysicists. As presented in Step-4a of Figure 5.2, this strategy builds four distinct balanced datasets using stratified sampling. The sampling strata are the solar flares classes, and the sampling schema is detailed next. The first dataset is composed of 50% of the tuples classified as classes A or B and 50% of remaining; the second one is composed of 50% of tuples class C and 50%, the remaining; the third is balanced in the same way for class M, and the last one for class X.

Figure 5.2 shows an illustration of an execution of MultiClass2Binary. The Slid Database is highly imbalanced. In this illustration, four datasets are created by random undersampling the original SD. The called AB-Dataset is composed of $|AB|$ tuples classified as AB and $|AB|$ of instances classified as C or M or X. The called C-Dataset is composed of $|C|$ tuples classified as C and $|C|$ of instances classified as AB or M or X. M and X-Datasets follows the same approach.

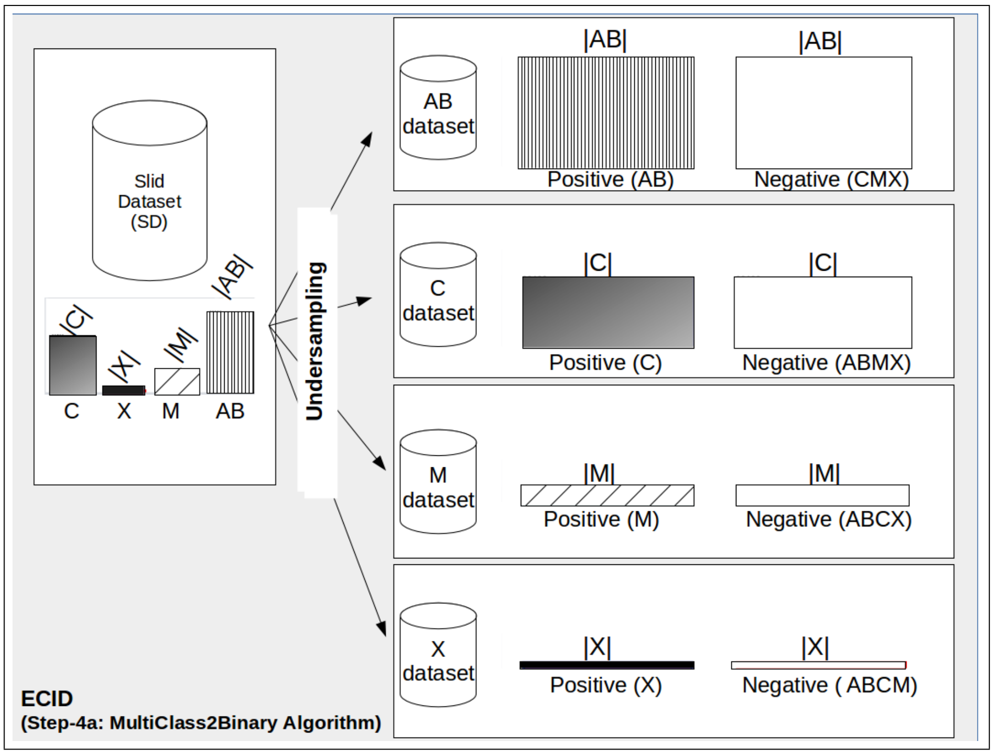


Figure 5.2: Example of the application of the MultiClass2Binary strategy into SD

Each dataset has just two classes, and this splitting mechanism aims to give more weight to the rarest classes. Thus, a binary classification method (or inducer) is employed to perform the forecasting for each of the datasets generated by MultiClass2Binary, as shown in Step-4b of Figure 5.3.

Each dataset is applied to a binary inducer. Accordingly, the **Binary Classifier#1** provides the forecasting for class *AB* or not, **Binary Classifier#2**, for class C or not, **Binary Classifier#3**, for class M or not, and, **Binary Classifier#4**, for class X or not. We propose to employ all inducers of the same type because our experiments did not demonstrate improvements by combining different types of inducer. However, the combination of different inducers may also be performed.

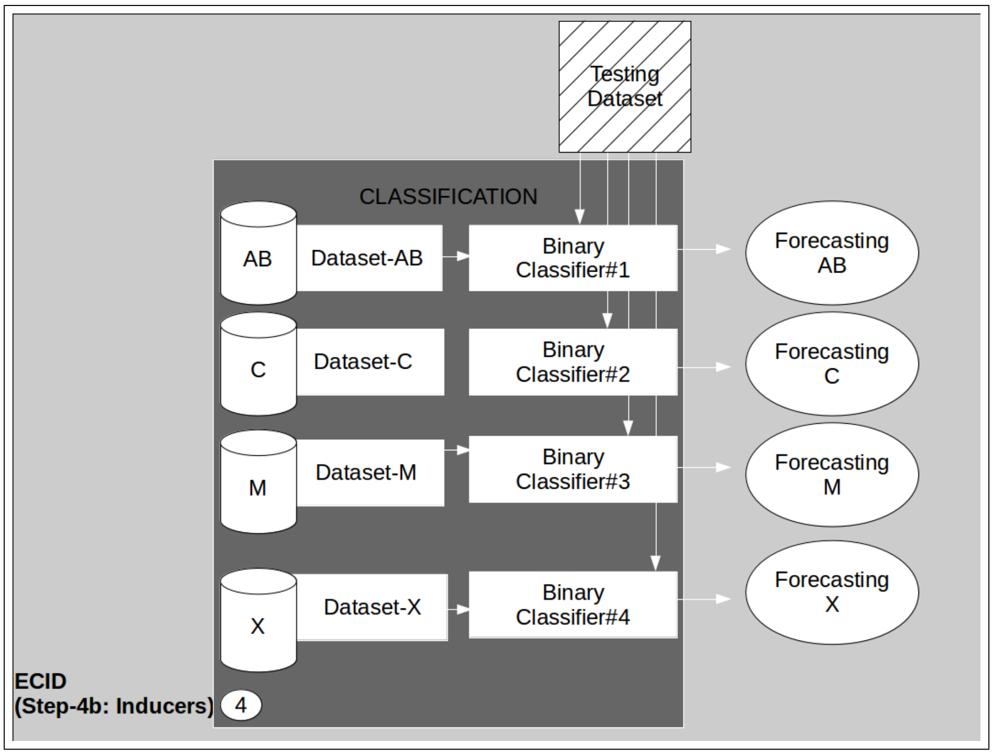


Figure 5.3: Step-4b: Base inducers

The goal of ECID is to return the possible solar flare classes that may occur in a given day. However, SD has 120 instances per day, because the sampling period is 12 minutes. The base inducers are trained with 120 instances per day; consequently, the forecasting result is also given in a 12 minutes sampling period. As there are four training datasets, for a given testing sample, there will be at least four possible results: AB, C, M and X. Table 5.9 shows an example of the base inducer individual forecasting.

Table 5.9: Example of a base inducer individual forecasting

instant	AB_y	C_y	M_y	X_y
Day_1_Instant_00:00	1	1	0	0
Day_1_Instant_00:12	...	...	...	...
Day_1_Instant_00:24	...	...	...	...

The instant column corresponds to the time of the forecasting result. Columns Z_y (where Z = {AB,C,M,X}) is set to 1 if the corresponding inducer produced positive forecasting for the given class. Consider the tuple with Day_1_Instant_00:00, if Binary Classifier#1 produced a positive forecasting, the AB_y column is set to 1; if Binary Classifier#2 produced a positive forecasting, the C_y column is set to 1; and so on.

Note that, the forecasting is originally given in 12 minutes, but the goal of this work is to produce daily forecasting. So, the aggregation method combines the votes of the base inducers producing a daily result as represented in Figure 5.4.

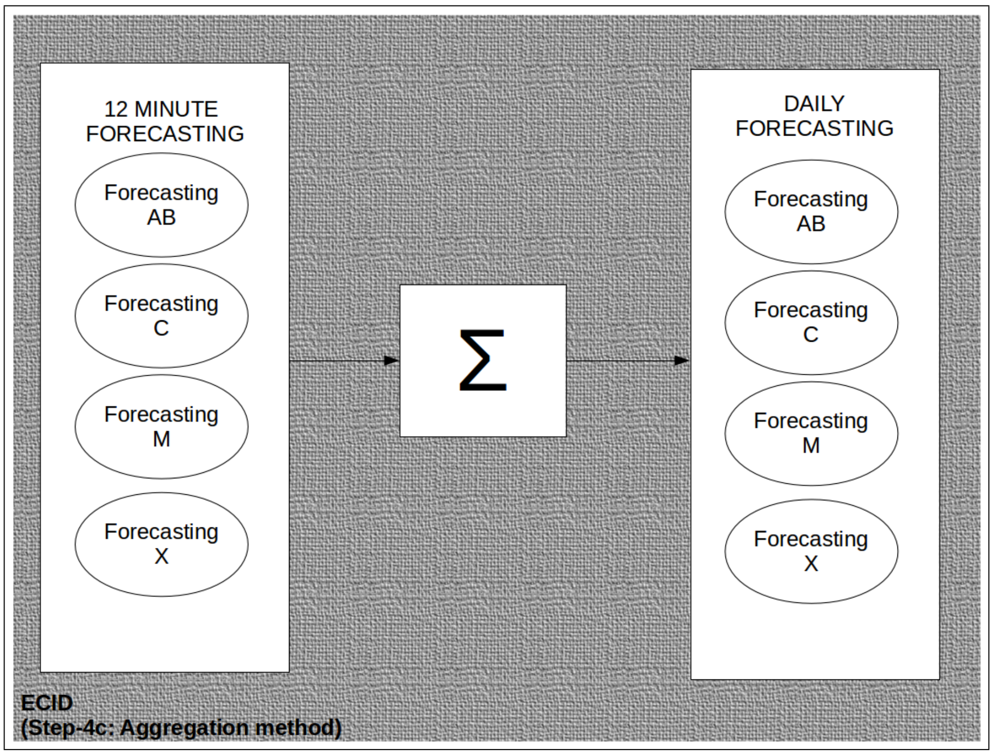


Figure 5.4: Step-4c: The Aggregation Method

The Aggregation method sums all the vote given by each inducers into a daily manner, so that Table 5.10 is produced.

Table 5.10: Example of an aggregated daily forecasting

instant	AB_y	C_y	M_y	X_y
Day_1	80	40	0	0
Day_2	0	0	20	100
Day_3	0	2	90	28

Note that Table 5.9 shows the forecasting for sampling period of 12 minute of the fictitious Day_1. Table 5.10 gives the sum of the votes given by each inducer in a daily manner, so that the sum of votes are given for the fictitious Day_1, Day_2 and Day_3.

After calculating the daily sum of Z_y, the aggregation method decides the final forecasting result for a given day. So, this method uses a decision rule to obtain the final result. For this purpose, it was proposed a parameterized schema, with three parameters (p1, p2 and p3). The parameters p1, p2 and p3 denote the minimum number of daily votes required for the selection of the first, second and third most voted classes. For example, if AB is the most voted and the number of votes is given by $|AB|$, this class will be selected if $p1 \leq |AB|$. If C is the second most voted and the number of votes is given by $|C|$, this class will be selected if $p2 \leq |C|$. If M is the third most voted and the number of votes is given by $|M|$, this class will be selected if $p3 \leq |M|$.

We found empirically that the best results are obtained using $p1 = 24$, $p2 = 6$, and $p3 = 3$. The importance of minority classes is increased because their class frequencies are low.

For example, in Table 5.10, Day_1 has the first and second most voted given by classes AB and C. Day_2 has the first and second most voted given by classes X and M. Day_3, however, has M, X and C as the top three most voted.

Table 5.11: Example of the Forecasting Result Calculation (p1 = 24; p2 = 6; p3 = 3)

Instant	1th Voted	2nd Voted	3rd Voted	Forecasting Result
Day_1	24 <= 80 (AB)	6 < 40 (C)	-	ABC
Day_2	24 <= 100 (X)	6 <= 20 (M)	-	X
Day_3	24 <= 90 (M)	6 <= 28 (X)	3 <= 2 (C)	MX

Table 5.11 orders the classes and the respective votes for each day. In the first day, class AB is selected in the forecasting since $p1 = 24 \leq |AB|$. Another example, in the line of Day_3, the 3^{rd} most voted was class C, with 2 votes. However, as $p3 = 3 \leq |C|$ is not satisfied, class C is not selected in the forecasting result.

Note that if more than one condition is satisfied in this decision rule, a multi-label forecasting is obtained.

Another characteristic of the aggregation method is that it is possible to improve its decision rules by using a classification method. In Chapter 6, it is described some experiments that insert SVM in the aggregation method.

5.3 Final considerations

In this chapter, we presented ECID. In this method, the data is prepared using an extended version of SS. Four datasets are extracted to build models for each solar flare class through the ensemble of base inducers. An aggregation method provide multi-class and multi-label forecasting. Finally, the results can be submitted to a classifier, like SVM, to maximize the separability among the returned classes. The main characteristics of ECID were:

1. it uses the evolution of the solar feature time series in the forecasting process;
2. the specialist knowledge is employed in the configuration of the ECIDs parameters;
3. it handles imbalanced datasets using a balancing schema based in stratified sampling and ensemble of classifiers;
4. it performs a multi-class forecasting;
5. it handles the issue of high similarity between adjacent classes by enabling multi-label forecasting;
6. it enabled binary classifiers to be used as the EC's inducer by giving more weight to the least voted class. This strategy was applied because when the *MultiClass2Binary* algorithm stratified the original dataset, it gave a higher weight to the main class of each balanced stratified dataset.

In the next chapter, the experiments performed using SeMiner and ECID are presented. We also discuss the validation of these methods according to the goals specified in this thesis.

Chapter 6

EXPERIMENTAL RESULTS

This chapter describes the experiments performed in order to test and validate SeMiner and ECID. It also describes the gains obtained by handling the imbalanced dataset in ECID against the original SeMiner method. Finally, it discusses the obtained results against our hypothesis, and also compares our results to the most closely related works of solar flare forecasting.

6.1 SeMiner Experiments

Three experiments were performed to test SeMiner and its core algorithm, SS. The first experiment applied SeMiner to forecast the X-ray background level emitted by the Sun. The second, predicted solar flares using SeMiner. And the last experiment, analyzed the speedup obtained by the parallelized version of SS. All three experiments used the X-ray time series as input. The first two experiments performed a binary forecasting. It was used the Yes and the No classes. The Yes means that SeMiner predicted a solar flare or background level greater than or equal to class C, otherwise the method predicted No.

The metrics used to analyze the results were: True Positive Rate, True Negative Rate and ROC Area.

An overview of these three experiments are described as follows:

1. The first experiment considered the intensity of X-rays in the 1–8 Angstrom passband as input of the SeMiner method. It aimed to forecast the background level of X-ray flux instead of solar flares (the difference between these concepts are in Section 2.1). The method was tested using different classification methods and datasets with the forecasting horizon set to one day. The best results achieved True Positive Rate = 94.3%, True Negative Rate = 86.5% and ROC Area = 90.5%, using the IBK classifier (a variation of

the k-nearest neighbor method-KNN). These results show a strong balance between True Positive (TP) and True Negative (TN) rates, which is a desirable feature considering that the solar dataset is very imbalanced.

2. The second experiment employed SeMiner to forecast solar flares. It was used different feature selection methods and classifiers to check the configuration that provided the best results. For this purpose, the feature selection methods Relief Attribute Evaluation (KIRA; RENDELL, 1992) and StarMiner (RIBEIRO et al., 2009) were employed in SeMiner to select the periods of the dataset that best distinguished the solar flares. Also, this experiment used the following classification methods in SeMiner: J48 (a decision tree implementation), IBK (a K-nearest neighbor implementation), NaiveBayes, OneR and SVM. In this experiment, it was found that the time interval that best distinguish solar flares are within two days before the event. More specifically, it was found that the initial and final periods of the first day, and the remaining 16 hours of the second day are the intervals that contains the data that once used in SeMiner, provides the best results. This finding corroborates the empirical heuristic given by the domain specialist: the previous two days of data are the most important to predict possible future solar flares. Another benefits of using feature selection method in SeMiner is that it reduced SeMiner execution time as well as produced higher accuracy with balanced TPR and TNR. SeMiner achieved: Accuracy = 72.7%, TPR = 70.9% and TNR = 79.7%.

3. The last experiment was concerned with the performance of the parallelized version of SS. It was found that the parallelized version of SS runs about four times faster than the sequential algorithm.

Details about each of those experiments are presented along with a discussion about the results obtained and the main findings.

6.1.1 SeMiner - first experiment: X-ray background level forecasting

The goal of this first experiment was to start validating the SeMiner method by forecasting the X-ray background level of a future day. For this purpose, we used a time series of X-ray flux comprising data sampled from 2014 and 2015. The data was provided by SWPC Primary GOES X-ray satellite in its website (https://www.swpc.noaa.gov/products/goes-x-ray-flux). As mentioned before, this data source provides two types of X-ray: one recorded in the 1-8 Angstrom passband, and another in the 0.5-4.0 Angstrom passband. In order to compose the training and testing dataset, we used the time series of X-ray recorded in the 1-8 Angstrom passband.

The X-ray time series provided at NOAA website (https://satdat.ngdc.noaa.gov/sem/goes/data/new_avg/) is composed of the X-ray observations recorded in intervals of 1 minute, but its classification, as presented in Table 1.1, is not assigned in this report. Thus, it was necessary to assign the correspondent class for each record. For this purpose, we compared each observation of the X-ray time series against the ranges of each class given in Table 1.1 in order to perform this assignment.

For example, consider an X-ray time series composed of the X-ray values given in Table 6.1. Instant-0, 1, 2 and 3 provided X-rays values of 1E-07, 5E-07, 7E-05, 8E-04 W/m^2, respectively. These observations are not originally assigned with any class. Thus, we looked at, for each observation, the ranges of each X-ray background level given in Table 1.1 and performed the assignments.

Table 6.1: Example of the class assignment in an X-ray time series.

Instant	X-ray	Class
0	1E-07	B
1	5E-06	C
2	7E-05	M
3	8E-04	X

In this way, the first instance contains an observation *I = 1E-07*. If we look at Table 1.1, it falls in the range of class B. Thus, this observation is assigned to B. The following observations were assigned using the same approach.

Furthermore, two types of dataset splitting were used to validate this experiment:

- 10 fold cross-validation: the dataset was divided into ten folds. Then, the first part was used as the test set, and the other nine as the training set. Afterward, the second part was used as the test set, and the remaining, as the training set, and so on.
- Fixed dataset splitting: 67% for training, and 33% for testing.

Finally, the tests were performed using data taken from the following periods: 10 days, one month, one year and two years. Table 6.2 shows the setup for this experiment.

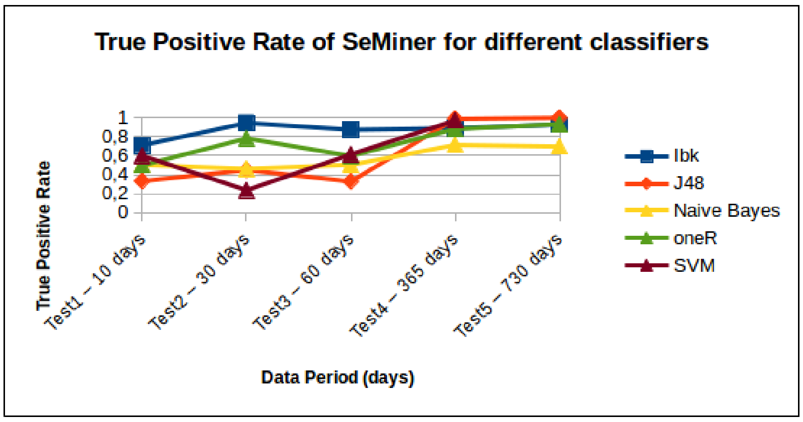
Table 6.2: Setup of the SeMiner experiments. *Test Number* is the number of the test performed; *n* (current window size), *Step* (window step), *j* (jump) and *f* (size of the future window) are measured in numbers of observations; Δt (sampling period) is measured in minutes; $|R|$ is the number of the generated sequences by SS.

Test Number	Test and training set division method	Class Definition of Solar Event	Δt	n	j	f	Step	#observations	Data Period (days)	$\|R\|$
1	Cross-validation: 10 folds	C, M and X	5	288	288	288	1	2880	10	2016
2	Cross-validation: 10 folds	C, M and X	5	288	288	288	1	8640	30	7776
3	Cross-validation: 10 folds	C, M and X	5	288	288	288	1	17280	60	16416
4	Dataset split: 67%: training 33%: testing	C, M and X	5	288	288	288	1	105120	365	104256
5	Dataset split: 67%: training 33%: testing	C, M and X	5	288	288	288	1	210240	730	209376

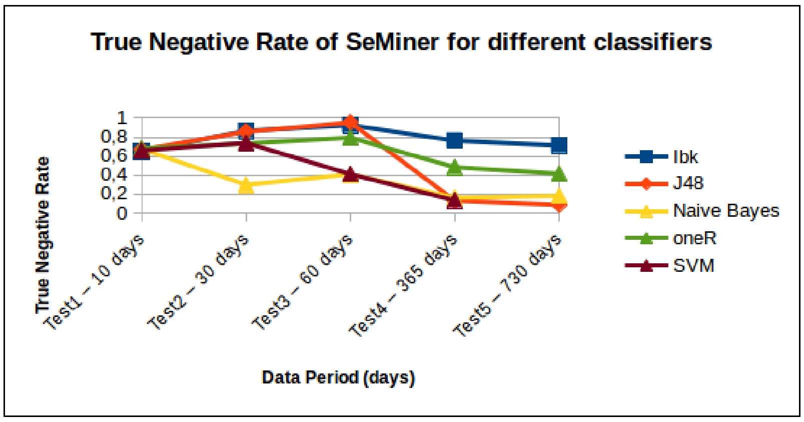
SeMiner allows the usage of any classifier. Therefore, in this experiment, we applied, for tests 1 to 5, the following classifiers to verify which one best forecasts X-ray flux intensity: J48 (C4.5 implementation developed in Java), IBK (a k-nearest neighbor algorithm), Naive Bayes, OneR, and Support Vector Machine (SVM) using the Polykernel function as SVM kernel function. The implementations of these algorithms were taken from The Waikato Environment for Knowledge Analysis (WEKA) tool (HALL et al., 2009).

$Test 1, 2, 3$ were validated with Cross-validation using 10 folds. On the other hand, $Test 4$ and $Test 5$ used 67% of the data for training and the remaining 33% for testing. The classes were categorized as Yes, for occurrences of the intensity of X-rays classified as C, M and X, and No otherwise. $Test 1$ used data collected from 10 days period, while $Test 2$ used 30 days of data, $Test 3$ used 60 days, $Test 4$ used one year of data, and $Test 5$ used 2 years of data containing X-ray flux intensity.

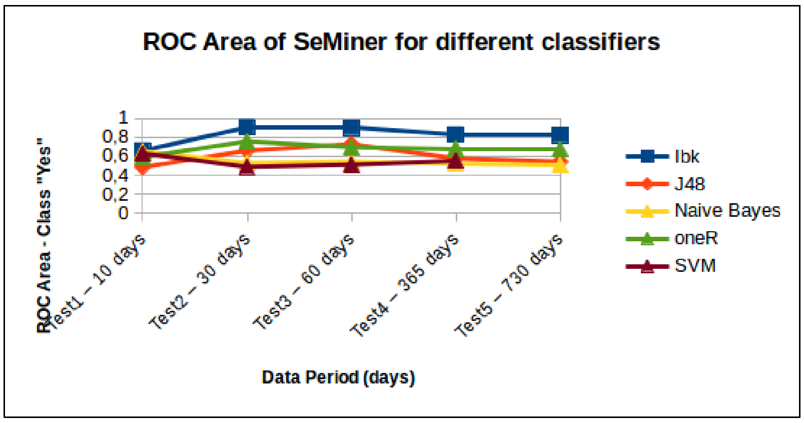
The graphics presented in Figure 6.1 shows the three metrics: True Positive Rate, True Negative Rate and ROC Area versus each test performed for 10, 30, 60, 365 and 730 days of data.



a) TPR of SeMiner tests 1 to 5



b) TNR of SeMiner tests 1 to 5



c) ROC Area of SeMiner tests 1 to 5

Figure 6.1: SeMiner - Results of the first experiment.

The classifier OneR, which constructs one single rule for each data attribute, was employed as the baseline classifier. Despite its simplicity, it performed relatively well (see Figure 6.1), because the input tuple was a preprocessed data sequence, with 288 attributes. In this case, the OneR model produced 288 rules that, using the majority vote strategy, could fairly describe the data.

The J48, an implementation of the C4.5 classifier, presented a significant decrease in the TNR as the dataset size increased. It employs a Decision Tree approach that is very sensitive to data imbalance, losing the learning model specificity.

The use of Naive Bayes resulted in one of the poorest results. The method employs a probability model that assumes independence among attributes. However, the processed sequences have features with high levels of dependency. This is because each feature produced by SS have a shifted sub-series compared to their subsequent features. So that, each feature are strongly correlated.

Even though SVM usually produces accurate models to categorize tuples, it achieved poor results (ROC Area: 0.63, TPR: 0.6 and TNR: 0.658) in the experiment performed. The SVM classifier presupposes a multidimensional data and finds the best hyperplanes that separate the data into classes. However, the datasets used are originally unidimensional, as previously discussed. Additionally, the computational performance of SVM is too low, so that $Test5$ was not completed using SVM as it took too long to complete (more than 3 days).

The ROC Area, TPR and TNR obtained using the IBK classification method were the highest. The test using 30 days of data achieved the best results when combining the three metrics: ROC Area (0.905), TPR (0.943) and TNR (0.865).

This first experiment showed that SeMiner had great potential to predict future events. However, this experiment tested just the background level of X-ray in terms of the classification provided in Table 1.1. In the next experiment, SeMiner was used to predict solar flares.

6.1.2 SeMiner - second experiment: solar flare forecasting

This experiment used SeMiner to perform solar flare forecasting. It was used X-ray time series, provided by NOAA website (⟨http://www.swpc.noaa.gov/products/goes-X-ray-flux⟩), as input for the method.

This second experiment consisted of six tests. It was used StarMiner and Relief Attribute Evaluation as the feature selection methods within SeMiner. Each test used an X-ray time series collected in 2014 as the training dataset to produce forecasting with one day of advance. The data collected during six months of 2015 formed the testing dataset. This data division ensured that the results obtained were not biased due to close instances in both testing and training datasets. Table 6.3 shows the setup of the tests performed.

Table 6.3: SeMiner - Planning of the second experiment

					Data Interval		
Test number	**Feature Selection Method**	**Current Window**	**Jump**	**Future Window**	**Training**	**Testing**	**Classification Method**
1	Not used	1	1	1	1 year 2014	6 months 2015	J48, IBK, NaiveBayes, OneR, SVM (SMO - Weka - Polykernel)
2	Starminer						
3	Relief Attribute Evaluation						
4	Not used	2	1	1			
5	Starminer						
6	Relief Attribute Evaluation						

Two main configurations were considered during this experiment: (1) one day for the *current window*, and; (2) two days for the *current window*. *jump* and *future window* were configured as one day for both phases. The classification methods employed were: J48, IBK, Naive Bayes, OneR, Support Vector Machine (QUINLAN; ROSS, 1993; AHA; KIBLER; ALBERT, 1991; BESNARD; HANKS, 1995; HOLTE, 1993; SCHOLKOPF; BURGES; SMOLA, 1999).

As we can see in Figure 6.2, the NaiveBayes classifier resulted in the largest ROC areas. Test 6 achieved the largest one, reaching 0.799 of the ROC area. On the other hand, we observe that this experiment got an accuracy of 72.7%. This accuracy can be considered high, considering the imbalanced dataset, and is accompanied by balanced TP and TN rates, 70.9% and 79.7%, respectively.

The forecasting method results are presented in Figures 6.2–6.3.

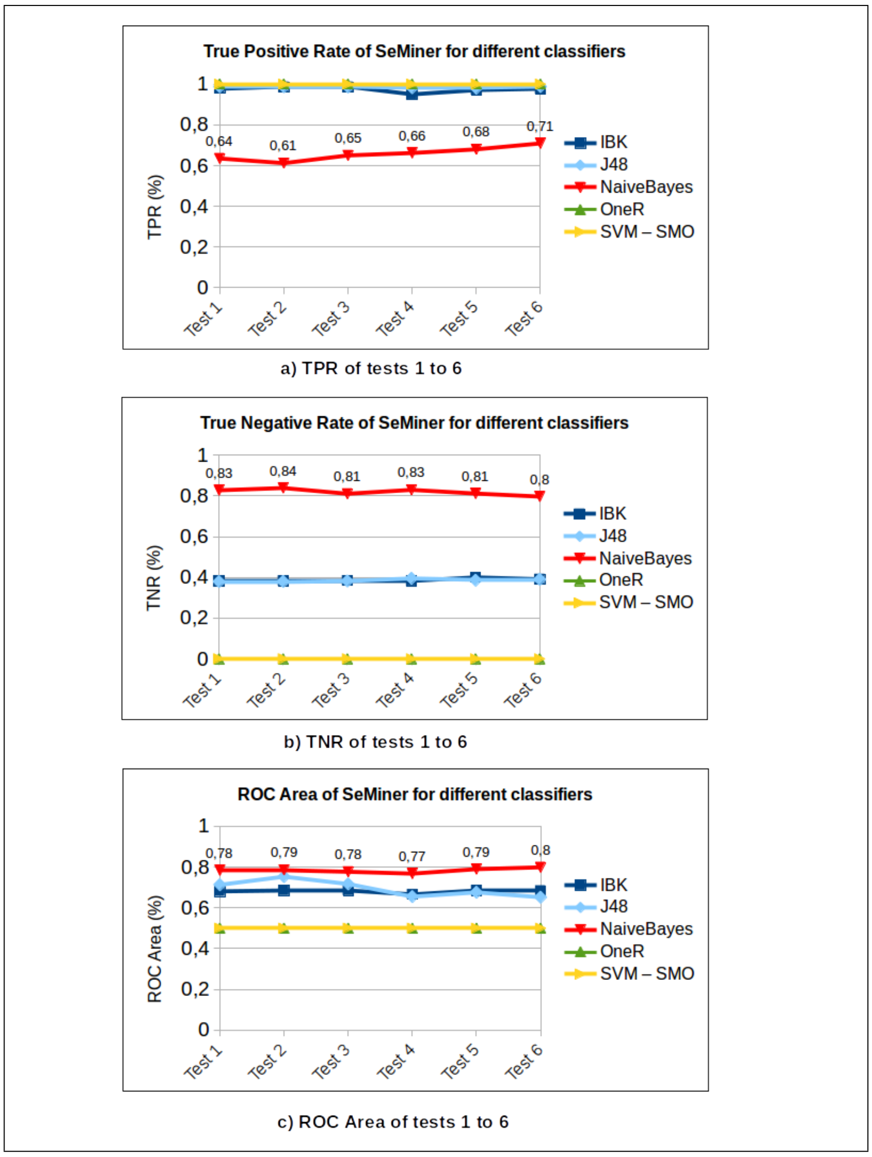


Figure 6.2: SeMiner - Results of the second experiment.

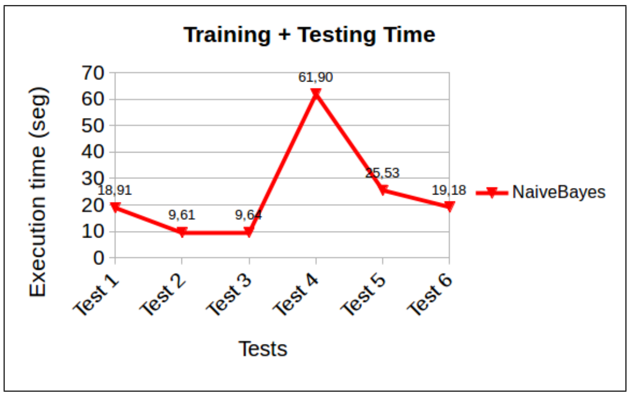


Figure 6.3: SeMiner Total time = Training Time + Testing Time (Time in seconds).

The last aspect analyzed in this experiment was the results produced by the feature selection method in SeMiner.

Selecting features in this domain means choosing the most significant time intervals that distinguish solar flares, like explained in Section 4.2.2. In addition, as feature selection is employed, less data is used by SeMiner and consequently, it increases the speed-up of the overall execution.

In tests 1 to 6, SS algorithm generated 288 features, because the *current window* was set to one day (or 288 instances). As each feature corresponds to an interval of 5 minutes, we have a relation between the selected features and the periods in which it relates. For example, the first twelve features corresponds to 5 x 12 minutes, i.e., the first one hour of the observations within the *current window*.

In this way, when a feature selection method applied to SeMiner choose certain features, actually it is selecting the most significant periods that best distinguishes a solar flare. For this analysis, we have mapped the selected features of each test to the intervals that they are related with.

Figure 6.4 shows the result of this transformation. As shown above, Test 6 achieved the best results. In this test, it was combined the classification method NaiveBayes and the feature selection method Relief Attribute Evaluation using two days as *current window* within SS. Hence, we can observe from the yellow line of Figure 6.4 that the features selected by test 6 actually chose the data comprising the begin and end of the first day and the end of the second day. It means that data collected within these intervals, two days before the solar flare, may distinguish the events according to the Relief Attribute Evaluation method.

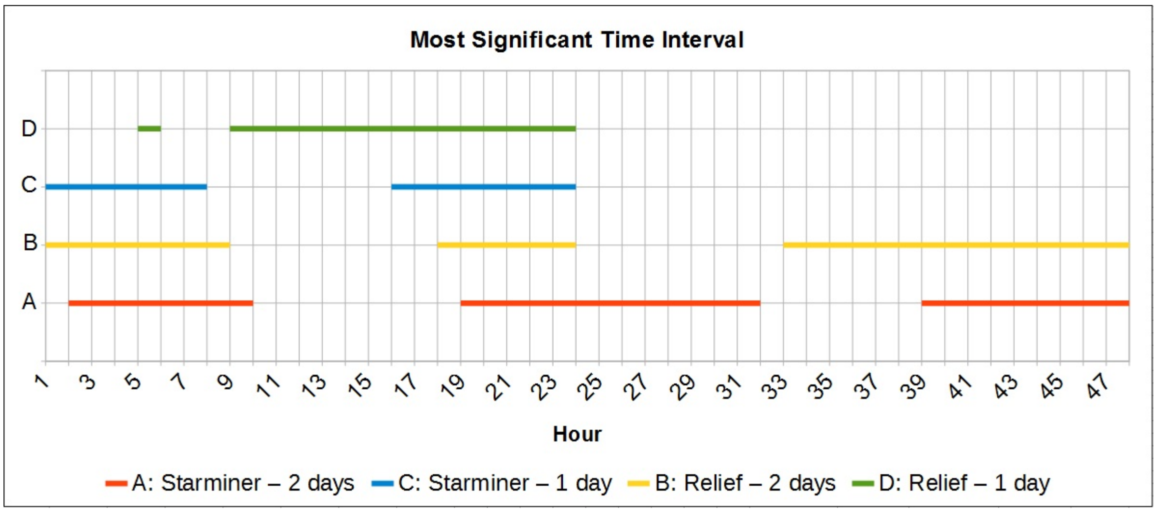


Figure 6.4: SeMiner - Most significant time interval analysis.

In summary, the results of Test 6 were significant due to the following reasons:

1. as shown in Table 6.3, Test 6 is configured with a *current window* of 2 days and, according to the domain specialist, this period has the most significant pattern that can be observed to distinguish a solar flare;

2. when Relief Attribute Selection feature selection method was used in SeMiner, along with two days for the current window, the selected intervals (given in hours) were: $1 \leq instant \leq 9$ (start of the first day), $18 \leq instant \leq 24$ (end of the first day), and $33 \leq instant \leq 48$ (last 16 h of the second day);

3. feature selection along with the Naive Bayes classifier resulted in the best TPR, TNR and ROC area among all the tests performed. Furthermore, it produced a speedup of 3.2, i.e., an execution time 3.2 faster than if no feature selection had been performed ($speedup = time_{noRelief}/time_{withRelief} = 61.9/19.18$);

4. the ROC Area, considering the tests performed with 2 days of current window, increased from 0.77 (without feature selection) to 0.80 (using feature selection);

5. it showed that the usage of SeMiner was efficient in the task of solar flare forecasting.

SeMiner is a binary forecasting method and proved to be successful in the prediction of grouped solar flares classes. It also showed us that considering the evolution of solar data in the forecasting process, we could achieve good results. Next, we present the experiment performed to analyze the parallel version of SS, which was developed to optimize the original sequential version.

6.1.3 SeMiner - third experiment: SS parallel optimizations

The third experiment with SeMiner were performed in two phases. It aimed to compare the parallel version of the SS algorithm against its sequential implementation.

The first phase of this experiment was performed using the implementation based on pure C language, and a three years time series of X-ray intensity data (2013 to 2015). The *current window*, *jump* and *future window* were set to 288 instances, while *step* was set to one instance. As the sample rate is 5 min, it is equivalent to say that *Current Window*, *Jump* and *Future Window* are set to one day. Thus, we intended to forecast one day in advance. The running times of interest were measured using the time command of C and CUDA, and the start/end of the time command involved just the function that executed the pure C algorithm. The pure C algorithm was run three times, and its mean execution time was calculated.

The second phase of experiments was performed with the same configuration of the previous one, except that the CUDA kernel implementation was used. In this experiment phase, the time command of C and CUDA was also used, but the start/end was concerned with the parallelized function. Therefore, it measured the execution time of the parallelized SS. The kernel function was run three times and its mean execution time was calculated.

A summary of this experiment, the mean execution time and the processed data size are shown in Table 6.4.

Table 6.4: SS CUDA implementation: Executions and Results

Setup						**Results**	
SeMiner Implementation	**Data volume**	**Current Window**	**Jump**	**Future Window**	**Step**	**Mean execution time (sec)**	**Size of output file**
pure C	3 years (2013-2015)	288	288	288	1	208,5	1,09GB
With CUDA	3 years (2013-2015)	288	288	288	1	47,8	1,09GB

This experiment was performed on a computer with the following hardware configuration:

- Processor: Intel i5;
- RAM: 8 GB;
- Graphic's card: Geforce 960X
  - Number of CUDA cores (GPUs): 1024
  - Memory: 2 GB
- Operating System: Windows 10;
- Application running during the test: Eclipse IDE

The execution speedup obtained by using CUDA for the implementation of SS is presented in Equation (6.1):

$$\frac{time_{\text{pureC}}}{time_{\text{CUDA}}} = \frac{208.4562}{47.8299} = 4.36 \qquad (6.1)$$

The absolute execution time, in both sequential and parallel versions of SS was relatively small for the project objectives. This reflects the efficiency of the algorithm. Even so, the parallelized version implemented using CUDA showed to be 4.36 times faster than the sequential version, what indicates that the parallel version should be adopted. Next section, we present the experiments performed to validate the proposed multi-class and multi-label method: ECID.

6.2 Experiments using ECID

In this section, the experiments performed to test and validate ECID are described. In order to evaluate the proposed method, the experiments implemented the following methodology:

- Selection of the solar features used in the forecasting process. For this purpose, a feature selection method was run to rank the features and choose the ones to take part in the experiments;
- Selection of the classification methods used as base inducers of ECID;
- Selection of the SS parameters in order to include the aims of the domain specialist;
- Preparation of the training and testing dataset used in the experiments;
- Evaluation of ECID through the quality metrics used to validate the hypothesis;
- Comparison of ECID results with SeMiner;
- Comparison of ECID with solar flare forecasting methods of the literature.

All the experiments were performed in a notebook with the following configuration: 8 GB of memory RAM, 1 TB of hard disk and *Intel Core i5* processor of 2.2 GHz. The operating system was Linux Ubuntu 16.04. The data was stored and managed in the Postgres DataBase Management System, and ECID was developed using the programming languages C and Java 8.

Due to the multi-class and multi-label characteristic of our problem, the original binary metrics may not fit to our domain. Next section, we describe the issues we faced when considering traditional metrics to our domain. Also, we show multi-class and multi-label metrics, which are more suitable for this domain.

6.2.1 Metrics in solar flare forecasting

As mentioned in Section 1.4.1, special attention must be taken about metrics original designed to analyze binary classification. Problems that require multi-class and multi-label classification should not use these metrics directly, because it may incur in misinterpretation of the real capabilities or limitations of the method. Here, we will call binary metrics the ones that were originally designed to validate binary classification problems.

In literature, we found some customizations applied in binary metrics in order to use them in multi-class domains. Usually, it is found metrics of *multi-class precision(c), multi-class recall(c) and multi-class F-Measure(c)* calculated for each class. However, this approach turns difficult to compare multi-class methods, so that it is also found aggregated metrics like *multi-class precision, multi-class recall and multi-class F-Measure* that assess the overall performance of the method (BRANCO; TORGO; RIBEIRO, 2016). For example, *multi-class recall* (*true positive rate*), *multi-class precision* and *multi-class F-Measure* calculated for an individual class *c* are defined as follows (BRANCO; TORGO; RIBEIRO, 2016):

$$\textit{multi-class recall(c)} = \frac{TP_c}{TP_c + FN_c} \tag{6.2}$$

$$\textit{multi-class precision(c)} = \frac{TP_c}{TP_c + FP_c} \tag{6.3}$$

$$\textit{multi-class F-Measure(c)} = \frac{2 \times \textit{multi-class precision(c)} \times \textit{multi-class recall(c)}}{\textit{multi-class precision(c) + multi-class recall(c)}} \tag{6.4}$$

The *aggregated* metrics are:

- *overall multi-class recall* is the mean of the *multi-class recall* for all classes;
- *overall multi-class precision* is the mean of the *multi-class precision* for all classes;
- *overall multi-class F-Measure* is the mean of the *multi-class F-Measure* for all classes.

For multi-class domain, this approach would produce interesting metrics for performance comparison purposes. However, they are not enough when dealing with a multi-label method.

Since ECID is multi-class and also multi-label, both metrics, individual and aggregated, should be adapted to work when the forecasting contains more than one class (multi-label). The *multi-label recall(c)* for individual classes in multi-label domain may use a equation similar to Equation 6.2, because the calculation of TP remains the same.

$$\textit{multi-label recall(c)} = \textit{multi-class recall(c)} \tag{6.5}$$

But, the *multi-label precision(c)* may be modified, as we define next. In this case, for each class c, the *multi-label precision(c)* is the fraction of corrected predictions in all predictions in which c belongs to the result.

$$\textit{multi-label precision(c)} = \frac{TP_c + TN_c}{TP_c + FP_c + TN_c} \tag{6.6}$$

The *precision* adaptation for multi-label classification is specially important when the forecasting falls in the boundary of two classes. For example, suppose the forecasting obtained was M X, but the actual event is given by M. If we consider class c as X in the calculation of the precision, it would have a False Positive. However, a part of the forecasting matched the actual event. Then, actually, the label X took part in the correct forecasting. This is what we want to measure. In a multi-label approach, it is important to calculate the full or partial hit. This is more meaningful that just analyzing each class isolated in a multi-label domain.

Another important metric is the *multi-label error* of a class c, which is the relation of false negatives and the total number of observations in a certain test set.

$$\textit{multi-label error(c)} = \frac{FN_c}{TP_c + TN_c + FP_c + FN_c} \tag{6.7}$$

It assess the number of prediction errors for a given class c among all possible observations found in the testing dataset. This metric is related with the cost of a false negative for a given class.

The overall multi-label metrics are:

- *overall multi-label recall* is the mean of the *multi-label recall* for all classes;
- *overall multi-label precision* is the mean of the *multi-label precision* for all classes;
- *overall multi-label F-Measure* is the mean of the *multi-label F-Measure* for all classes.

Additionally to the metrics obtained from the mean of the individual classes, we found in the literature aggregated metrics for the multi-label domain (GODBOLE; SARAWAGI, 2004) calculated using the results of each instance. These ones differ from the metrics derived from individual metrics, because they consider the intersection from the forecasting and actual events for each instance. This leads to the exact count of matches considering the multi-label characteristic of the method. The *overall aggregated multi-label precision*, *overall aggregated multi-label recall* and *overall aggregated multi-label F-Measure* are calculated as follows.

$$\textit{overall aggregated multi-label precision} = \frac{|Actual \bigcap Forecast|}{|Forecast|} \tag{6.8}$$

$$\textit{overall aggregated multi-label recall} = \frac{|Actual \bigcap Forecast|}{|Actual|} \tag{6.9}$$

$$\textit{overall agg ml F-Measure} = \frac{2 \times \textit{overall agg ml precision} \times \textit{overall agg ml recall}}{\textit{overall agg ml precision} + \textit{overall agg ml recall}} \tag{6.10}$$

Where:

- $|Actual|$ is the number of the actual classes;
- $|Forecast|$ is the number of the classes predicted;
- *overall agg ml F-Measure* is the *overall aggregated multi-label F-Measure*;
- *overall agg ml recall* is the *overall aggregated multi-label recall*;
- *overall agg ml precision* is the *overall aggregated multi-label precision*.

These aggregated metrics assesses the overall performance of a multi-label forecasting method.

6.2.2 Solar Data used in the experiments and feature selection

These experiments used solar data recorded in the period from 01-May-2010 to 26-Dec-2017. The first step of them followed the procedure described in 5.2.1. The Preliminary

Database (PD) was produced as presented in Section 5.2.1. The database obtained has the format of the example shown in Table 5.7. The magnetic features used to compose PD were based on Bobra e Couvidat (2015) work: *R_VALUE*, *USFLUX, TOTUSJZ, TOTUSJH, ABSNJZH, SAVNCPP, MEANPOT, TOTPOT, SHRGT45*. PD was built with these magnetic features together with the X-ray time series and the list of events obtained in the mentioned period. The resulting PD had the following features:

PD = (T_REC, $\mathrm{RVALUE_{MIN}}$, $\mathrm{RVALUE_{MAX}}$, $\mathrm{RVALUE_{AVG}}$, $\mathrm{USFLUX_{MIN}}$, $\mathrm{USFLUX_{MAX}}$, $\mathrm{USFLUX_{AVG}}$, $\mathrm{TOTUSJZ_{MIN}}$, $\mathrm{TOTUSJZ_{MAX}}$, $\mathrm{TOTUSJZ_{AVG}}$, $\mathrm{TOTUSJH_{MIN}}$, $\mathrm{TOTUSJH_{MAX}}$, $\mathrm{TOTUSJH_{AVG}}$, $\mathrm{ABSNJZH_{MIN}}$, $\mathrm{ABSNJZH_{MAX}}$, $\mathrm{ABSNJZH_{AVG}}$, $\mathrm{SAVNCPP_{MIN}}$, $\mathrm{SAVNCPP_{MAX}}$, $\mathrm{SAVNCPP_{AVG}}$, $\mathrm{MEANPOT_{MIN}}$, $\mathrm{MEANPOT_{MAX}}$, $\mathrm{MEANPOT_{AVG}}$, $\mathrm{TOTPOT_{MIN}}$, $\mathrm{TOTPOT_{MAX}}$, $\mathrm{TOTPOT_{AVG}}$, $\mathrm{SHRGT45_{MIN}}$, $\mathrm{SHRGT45_{MAX}}$, $\mathrm{SHRGT45_{AVG}}$, $\mathrm{XR_{1_{MIN}}}$, $\mathrm{XR_{1_{MAX}}}$, $\mathrm{XR_{1_{AVG}}}$, $\mathrm{XR_{2_{MIN}}}$, $\mathrm{XR_{2_{MAX}}}$, $\mathrm{XR_{2_{AVG}}}$,...., $\mathrm{XR_{12_{MIN}}}$, $\mathrm{XR_{12_{MAX}}}$, $\mathrm{XR_{12_{AVG}}}$, $\mathrm{CLASS_{MAX}}$, $\mathrm{CLASS_{MIN}}$)

PD has a total of 335521 instances composed of 319926 instances classified as AB, 13498 as C, 1933 as M, and 164 as X. This database serves as input to feature selection methods to rank the magnetic features used in the experiments. As we aim to forecast the maximum solar flare occurred in a given day, it was used the attributes with the maximum values named $\{\mathrm{MagneticFeature}\}_{\mathrm{MAX}}$.

The employed feature selection methods and a brief description of them are listed in the following (HALL et al., 2009):

- CfsSubsetEval: *Evaluates the worth of a subset of attributes by considering the individual predictive ability of each feature along with the degree of redundancy between them..*
- ClassifierAttributeEval: *‘‘Evaluates attribute subsets on training data.*
- CorrelationAttributeEval: *Evaluates the worth of an attribute by measuring the correlation (Pearson’s) between it and the class.*
- GainRatioAttributeEval: *Evaluates the worth of an attribute by measuring the gain ratio concerning the class.*
- InfoGainAttributeEval: *Evaluates the worth of an attribute by measuring the information gain concerning the class.*
- PrincipalComponents: *Performs a principal components analysis and transformation of the data.*

- SymmetricalUncertAttributeEval: *Evaluates the worth of an attribute by measuring the symmetrical uncertainty for the class.*

Table 6.5: Top-4 attributes ranked by the feature selection methods

Feature Selection Method	Rank of attributes
CfsSubsetEval	XR_{MAX}
ClassifierAttributeEval	XR_{MAX}, $TOTUSJZ_{MAX}$, $USFLUX_{MAX}$, $SHRGT45_{MAX}$
CorrelationAttributeEval	$USFLUX_{MAX}$, $TOTPOT_{MAX}$, XR_{MAX}, $RVALUE_{MAX}$
GainRatioAttributeEval	XR_{MAX}, $USFLUX_{MAX}$, $RVALUE_{MAX}$, $TOTPOT_{MAX}$
InfoGainAttributeEval	XR_{MAX}, $RVALUE_{MAX}$, $USFLUX_{MAX}$, $TOTPOT_{MAX}$
PrincipalComponents	$RVALUE_{MAX}$, $USFLUX_{MAX}$, $TOTUSJZ_{MAX}$, $TOTUSJH_{MAX}$
SymmetricalUncertAttributeEval	XR_{MAX}, $USFLUX_{MAX}$, $RVALUE_{MAX}$, $TOTPOT_{MAX}$

Table 6.5 shows the rank obtained in the execution of each feature selection method. The feature $\mathrm{XR}_{\mathrm{MAX}}$, $\mathrm{RVALUE}_{\mathrm{MAX}}$, and $\mathrm{USFLUX}_{\mathrm{MAX}}$ appear in the majority of the ranks. Thus, we selected these three features to apply in ECID.

6.2.3 Description of ECID's experiments

It was performed 12 experiments aiming to test and validate ECID. The experiments were grouped into three phases. The first phase intended to search for the most significant(s) feature(s) for the solar flare forecasting method. The second phase aimed to perform more tests in ECID with the selected feature in the previous phase. Finally, the third phase aimed to tune ECID up. In this section, we call the *multi-label precision(c)* simply as *precision*.

Phase-1: Most significant features for ECID

The three features selected were combined in this phase to verify the effects of employing them in ECID. For this purpose, five experiments were performed in this phase. All the experiments used IBK as base inducers, the balancing method was random undersampling, and the training and testing datasets were composed as 70% and 30% of the original SD, respectively. SS was configured by setting the *current window, jump and future window* as one day, meaning that it was used data values of two days before the event, and the forecasting horizon was set to one day.

- Experiment-1.1 used SD with just the XR_{MAX} feature as ECID input;

- Experiment-1.2 used SD with just XR_{MAX} and $USFLUX_{MAX}$ features as ECID input;
- Experiment-1.3 used SD with just XR_{MAX} and $RVALUE_{MAX}$ features as ECID input;
- Experiment-1.4 used SD with just $RVALUE_{MAX}$ feature as ECID input;
- Experiment-1.5 used SD with just $USFLUX_{MAX}$ feature as ECID input;

This experiment was analyzed using the multi-label metrics presented in Section 6.2.1. As the results obtained are multi-class and multi-label, we took some cautions to calculate the metrics:

1. First, it was calculated individual metrics for each class (AB, C, M, and X) by considering Positive for a specific class, and Negative as all the remaining classes;
2. The overall metrics were obtained to compare the experiments performed with related works.

Tables 6.6 and 6.7 shows the results obtained for each class. The mean of the results of Experiment-1.1 produced the best results and its individual results are presented in bold.

Table 6.6: Experiments 1.1-1.3 - Metric Details

	Experiment-1.1: XR_{MAX}				**Experiment-1.2: XR_{MAX} + $USFLUX_{MAX}$**				**Experiment-1.3: XR_{MAX} + $RVALUE_{MAX}$**			
	AB	***C***	***M***	***X***	***AB***	***C***	***M***	***X***	***AB***	***C***	***M***	***X***
multi-label recall	**71,1**	**90,9**	**83,6**	**92,3**	77,9	79,4	82,1	61,5	83,0	82,5	73,7	86,7
multi-label precision	**92,7**	**85,4**	**84,0**	**82,2**	86,3	86,5	80,7	79,5	87,3	88,4	84,7	81,9
multi-label F-Measure	**80,4**	**88,1**	**83,8**	**87,0**	81,9	82,8	81,4	69,4	85,1	85,3	78,8	84,2
multi-label error	**11,5**	**4,0**	**2,3**	**0,2**	8,8	9,0	2,5	1,0	6,6	7,7	3,8	0,4

Table 6.7: Experiments 1.4-1.5 - Metric Details

	Experiment-1.4: $RVALUE_{MAX}$				**Experiment-1.5: $USFLUX_{MAX}$**			
	AB	***C***	***M***	***X***	***AB***	***C***	***M***	***X***
multi-label recall	83,0	79,1	75,0	73,3	75,3	78,9	76,1	76,9
multi-label precision	85,5	88,0	82,8	80,5	85,4	84,8	79,7	77,5
multi-label F-Measure	84,2	83,3	78,7	76,7	80,0	81,8	77,9	77,2
multi-label error	6,6	9,2	3,6	0,8	9,8	9,2	3,3	0,6

The individual results of Experiment 1.1 shows interesting results specially for the rarest classes. The individual results achieved for class-X reached a *multi-label recall* equal 92.3%, a *multi-label precision* equal 82.2%, a *multi-label F-Measure* equal 81.9% and a *multi-label error* of 0.2%, what indicates very few occurrences of false negatives.

Figure 6.5 shows the results obtained in the experiments of Phase-1.

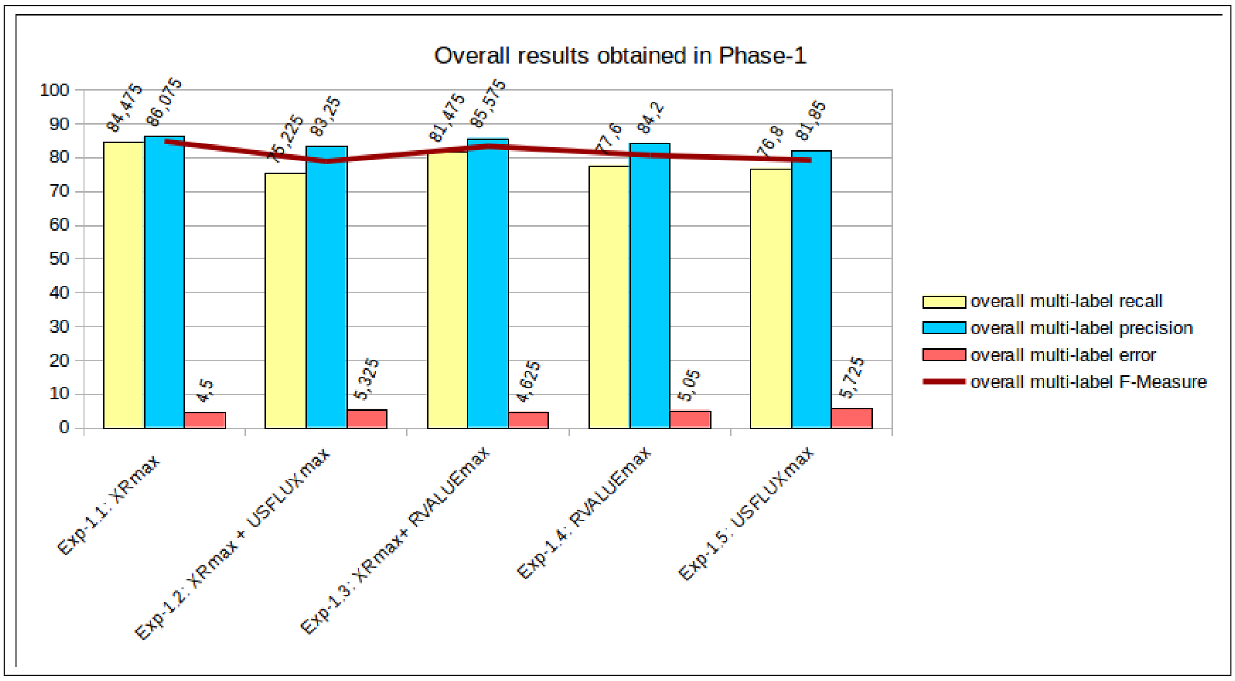


Figure 6.5: ECID: Results obtained by the experiments performed in Phase-1

Experiment 1.1 achieved the highest results: *overall multi-label recall* = 84.5%; *overall multi-label precision* = 86.1%; *overall multi-label error* = 4.5%. We can observe that the results of all experiments were in the range from 70% to 80%, so that the inclusion of magnetic features with X-ray did not improve the final results.

Additionally, Table 6.8 shows that the *overall aggregated multi-label recall* obtained in Experiment 1.1 was 92%, the *overall aggregated multi-label precision*, 58%, and the *overall aggregated multi-label F-Measure* achieved 82%.

Table 6.8: Overall aggregated metrics - Experiment 1.1

	Experiment 1.1
overall aggregated multi-label recall	92%
overall aggregated multi-label precision	58%
overall aggregated multi-label F-measure	82%

The medium *precision* is compensated by the low *overall multi-label error*, since in solar flare domain a false negative is more costly than a false positive.

Thus, we decided to concentrate our efforts in using only the X-ray time series in the next phase of the experiments.

6.2.3.1 Phase-2: Baseline ECID Setup

This phase aimed to validate the method by building another training and testing dataset from the original SD using XR_{MAX}. Additionally, we also compared ECID and SeMiner results. We applied the same training and testing datasets and performed multi-class forecasting for SeMiner, and multi-class/multi-label forecasting in ECID. This comparison was done to analyze the gain obtained by ECID over SeMiner, once the first handles the imbalanced dataset, while the second is not able to deal with this issue.

For this purpose, the experiments in this phase were configured as follows:

- It was used the same solar dataset composed of the time series of X-ray intensity emitted by Sun and the solar flare report from the period 2010 to 2017, due to the results obtained in Phase-1;
- It was used 70% of the dataset for training and 30% for testing to validate the method. More specifically, the training dataset was composed of 67678 instances (50% of class AB, 35% of class C, 13% of class M, and 2% of class X).

In this phase, it was performed six experiments identified as *Experiment 2.1 to Experiment 2.6*.

For experiments 2.1 to 2.3, we employed SeMiner varying the classification method: *Experiment 2.1* used IBK; *Experiment 2.2* employed SVM; and *Experiment 3* used J48. The implementation of these methods was obtained from Weka (HALL et al., 2009).

For experiments 2.4 to 2.6, we employed the proposed ensemble method ECID. The stratified under-sampling performed in the Step 4(a) of ECID (see Figure 5.1) is shown in Figure 6.6.

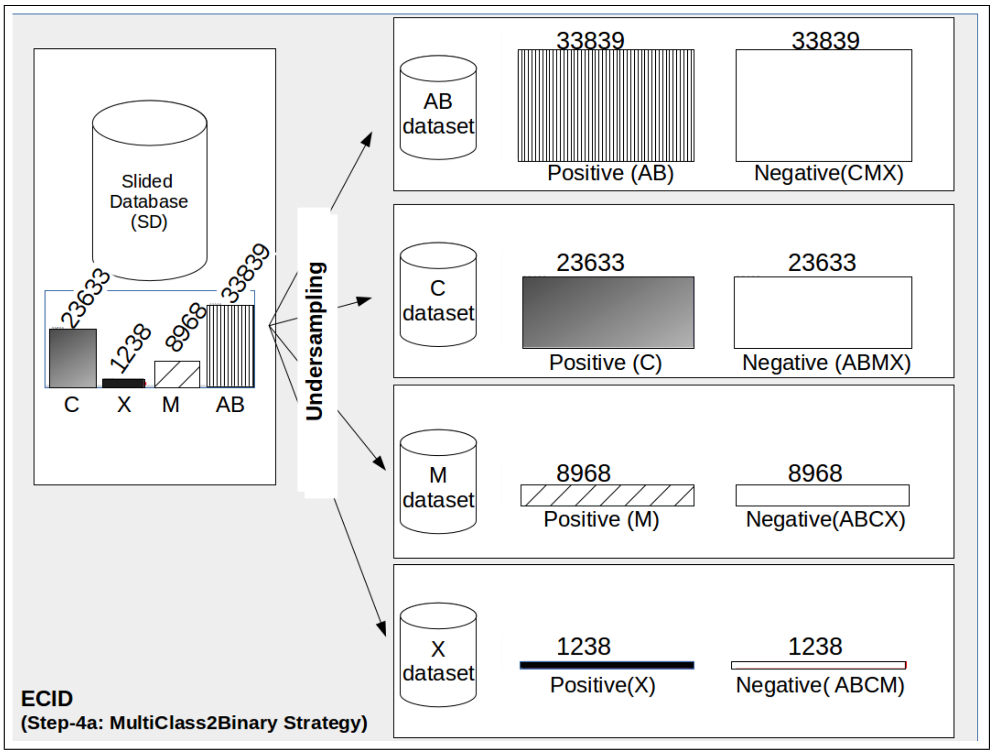


Figure 6.6: Example of the application of the MultiClass2Binary strategy into SD

The training dataset contained 33839 tuples labeled as class A or B (called AB), 23633 labeled as C, 8969 as M and 1238 as class X. The method randomly under-samples the original dataset according to each class. For example, the balanced dataset-AB is composed of 33839 tuples labeled A or B of the original dataset, these tuples are re-labeled as Positive(AB), the other 33839 tuples are composed by a subsample of the original tuples labeled as classes C, M and X, which is also re-labeled as Negative(CMX). The experiments 2.4-2.6 used this strategy for under-sampling, but they vary the base inducers in each experiment: in *Experiment 2.4* used IBK; in *Experiment 2.5* employed SVM; and, in *Experiment 2.6* employed J48.

All the experiments used the same training and testing dataset.

The analysis of the experiments of this phase was performed using the individual and the overall metrics described in Section 6.2.1. SeMiner was originally validated as a binary forecasting. But, as it is prepared to use any traditional classifier, including multi-class classification methods, we used these methods to provide a multi-class forecasting. Thus, the metrics used to validate SeMiner in this phase was the individual and overall multi-class ones. In the other hand, ECID is a multi-class and multi-label forecasting, thus we used individual and overall multi-label metrics in this phase.

Table 6.9 shows the results obtained by SeMiner using different classifications methods. And, Table 6.10 shows the results obtained by ECID varying the base inducers.

As observed in Table 6.9, the classifiers alone produced relatively good results for predictions of classes AB and C (which are the majority classes), and inferior results for classes M and X (the minority and most important classes). The highest results for classes AB and C were 72.8% of *multi-class recall* and 64.1% of *precision* in Experiment-2.2 (SVM), and 76% of *multi-class recall* and 56.6% of *multi-class precision* for class C in Experiment-2.1 (IBK). In the other hand, it reached 34.2% of *multi-class recall* and 12.4% of *multi-class precision* for class M, and 73.3% of *multi-class recall* and 10.8% of *multi-class precision* for class X. Although a good *multi-class recall* was obtained for class X, an inferior *multi-class precision* drastically decreased the level of reliability of the forecasting model using pure SeMiner (which did not handle imbalanced datasets). The *multi-class F-Measure* measure corroborate these results, which were reasonable for classes AB and C, and very poor for classes M and X.

Table 6.9: Experiments 2.1-2.3 - Metric Details obtained from SeMiner usage

	Experiment-2.1: IBK				**Experiment-2.2: SVM**				**Experiment-2.3: J48**			
	AB	***C***	***M***	***X***	***AB***	***C***	***M***	***X***	***AB***	***C***	***M***	***X***
multi-class recall	0,635	0,760	0,197	0	0,728	0,649	0,131	0,266	0,309	0,286	0,342	0,733
multi-class precision	0,708	0,566	0,468	0	0,641	0,565	0,434	0,8	0,914	0,482	0,124	0,108
multi-class F-Measure	0,670	0,649	0,277	0	0,681	0,604	0,202	0,4	0,462	0,359	0,182	0,189
multi-class error	0,102	0,256	0,032	0,000	0,158	0,220	0,024	0,002	0,012	0,136	0,345	0,169

Table 6.10 shows the results of Experiments 2.1-2.6, which used our proposed method ECID. The results obtained for the minority, but most important classes were higher than the previous experiment using SeMiner. The **best results** were achieved in Experiment 2.6 using J48 as the base inducers of ECID: 86.6% of *multi-label recall* and 98.5% of *multi-label precision*, and an *multi-label F-Measure* of 92.2% for class X. Also, the predictions of class M resulted in 100% of *multi-label recall* and 97.7% of *multi-label precision*. Predictions of classes AB and C achieved results of more than 95% from both *multi-label recall and precision*. We can observe that the *multi-label error* achieved low values to classes M and X: 0% and 0,8%, respectively.

Table 6.10: Experiments 2.4-2.6 - Metric Details obtained from ECID usage

	Experiment-2.4: IBK				**Experiment-2.5: SVM**				**Experiment-2.6: J48**			
	AB	***C***	***M***	***X***	***AB***	***C***	***M***	***X***	***AB***	***C***	***M***	***X***
multi-label recall	0,900	1	0,916	0,933	0,892	0,910	0,875	0,933	**0,966**	**0,98**	**1**	**0,866**
multi-label precision	0,988	0,943	0,950	0,946	0,948	0,933	0,911	0,904	**0,992**	**0,977**	**0,977**	**0,985**
multi-label F-Measure	0,942	0,970	0,933	0,940	0,919	0,922	0,893	0,918	**0,979**	**0,988**	**0,988**	**0,922**
multi-label error	0,046	0,0	0,008	0,004	0,05	0,034	0,011	0,004	**0,015**	**0,004**	**0,0**	**0,008**

Figure 6.7 shows the improvement of the overall metrics from the first three experiments

(using SeMiner) and the last three ones (using ECID). In average, the *overall multi-class/multi-label recall* had an increase of more than 120%. The *overall multi-class/multi-label precision* had an increase of 56.0% and the resulting *overall multi-class/multi-label error* is eight times lower than the one obtained through SeMiner. The improvement of the overall metrics is mainly due to the capability of ECID to handle imbalanced datasets and produce multi-class and multi-label forecasting.

Additionally, Table 6.11 shows that the *overall aggregated multi-label recall* obtained in Experiment 2.6 was similar to Experiment 1.1 reaching 92%, the *overall aggregated multi-label precision*, 57%, and the *overall aggregated multi-label F-Measure* achieved 82%.

Table 6.11: Overall aggregated metrics - Experiment 2.6

	Experiment 2.6
overall aggregated multi-label recall	92%
overall aggregated multi-label precision	57%
overall aggregated multi-label F-measure	82%

The medium *precision* was again compensated by the low *overall multi-label error*. These results show that even testing with a different training and testing dataset compared with Phase-1 (though the dataset still was collected from 2010 to 2017), ECID proved to reach consistent results.

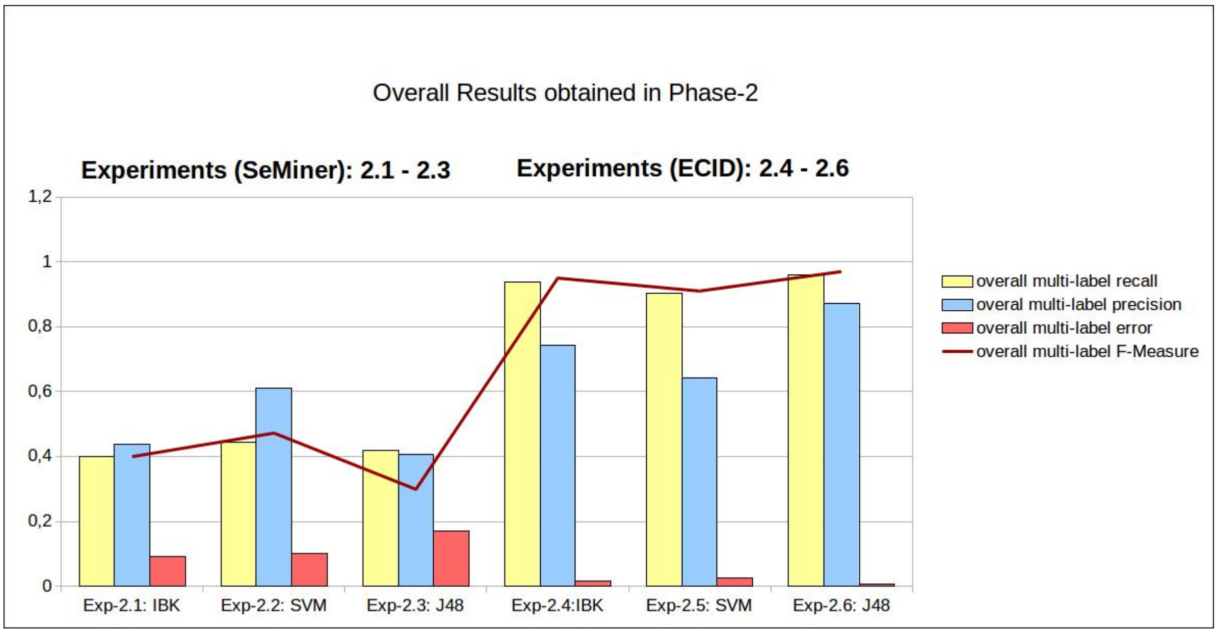


Figure 6.7: ECID: Results obtained by the experiments performed in Phase-2

The forecasting results produced by ECID is multi-label. Thus, an analysis of the combination of labels in the multi-label result was made. The method produced the distribution of classes shown in Table 6.12 for the best result obtained (using J48).

Table 6.12: Percentage of multi-label result of Experiment-2.6

Forecasting Result	%
AB	11,1
ABC	8,8
ABCM	52,5
CMX	27,6

The forecasting result of Experiment 2.6 is well concentrated in two sets of labels: ABCM and CMX. Though the results obtained are good, there is a chance to improve it by spreading the forecasting results in more diverse multi-labels. For this purpose, an adaptation was performed in the aggregation method of ECID. Thus, Phase 3 of the experiments aimed to test ECID by including in the aggregation method one more decision step. It was included a classification method, SVM, to analyze the voting count of the original aggregation method and perform the final forecasting.

6.2.3.2 Phase 3: Optimizing ECID Forecasting Results

In the aggregation method shown in Figure 5.4, it was initially used parameters *p1, p2 and p3* as described in Subsection 5.2.2. However, as discussed in Phase-2, the forecasting results achieved highly concentrated forecasting classes. For this purpose, we added to the original aggregation method, a new step to analyze the voting list through SVM. The result of this improvement had distributed the forecasting labels, and increased some important metrics as we show next. The configuration of Experiment 3.1 was set with the same values of Experiment-2.6, using the same training and testing datasets as well as the J48 classification method as base inducer. The only difference was the inclusion of SVM in the aggregation method, as previously mentioned. Table 6.13 shows the results obtained in this experiment.

Table 6.13: Experiments 3.1 of ECID - Results of the inclusion of SVM in the Aggregation Method

	AB	***C***	***M***	***X***	***overall multi-label metrics***
multi-label recall	92,6	99,8	91,7	93,3	94,4
multi-label precision	97,4	89,4	70,9	77,8	83,9
multi-label F-Measure	94,9	94,3	80	84,8	88,5
multi-label error	11,5	28,7	30,7	22,2	23,3

Considering class-X results, the *multi-label recall* increased from 86.6% to 93.3% in comparison of Experiment-2.6 (Table 6.10). While the *multi-label precision* had decreased from 98.5% to 77.8%, and the *multi-label error* had increased from 1% to 22.2%. The decrease of the *multi-label precision* was due to the increase of the spreading of the forecasting results. The *overall multi-label recall* achieved 94.4%, the *overall multi-label precision* 83.9%, the *overall multi-label F-Measure* 88.5% and the *overall multi-label error* was 23.3%. Thus, all metrics remained below the expected thresholds indicated by the domain specialist.

Table 6.14 shows the label's distribution resulted to the new forecasting model.

Table 6.14: Percentage of multi-label result of Experiment-3.1 (Aggregation method added with SVM)

Forecasting Result	%
AB	13,0
ABC	41,4
CM	33,7
CMX	4,6
MX	7,3

The forecasting produced was distributed among the following results: *AB, ABC, CM, CMX and MX*, where the majority of the results was *ABC and CM* (75,1%) and a few quantity of forecasting results were *AB, CMX and MX* (24.9%). This result means that the use of SVM in the aggregation method allowed to split the forecasting result, because in Phase-2 the best results contained the majority of the predictions composed of at least three classes.

As conclusion of Experiment-3.1, the inclusion of the SVM in the aggregation method increased the distribution of the forecasting results while kept the values of the metrics at interesting levels and according to the needs of the domain specialist.

Finally, it was performed the Experiment 3.2, which analyzed the forecasting in adjacent classes. For this purpose, it was used the same configuration of Experiment 3.1. In Experiment 3.1, ECID produced the *forecasting results* for each instance of the testing dataset and the metrics were calculated by comparing the *forecasting results* with the *actual solar flares*. The *actual solar flare* of this experiment was given according to the event occurred in each instant of the original X-ray time series used.

However, in Experiment 3.2, we aimed to analyze the behavior of ECID in adjacent classes. Thus, it was considered the sub-levels of each main class in order to assign the *adjacent classes* as the *new actual solar flares* in the testing dataset. For this purpose, the adjacent classes were determined as presented in Table 6.15.

Table 6.15: Mapping *Actual Class* to *Adjacent Classes (“New” actual class)*

Actual solar flare	Adjacent Classes (*“New” actual class*)
B1.0 to B6.9	B
B7.0 to B9.9	B,C
C1.0 to C6.9	C
C7.0 to C9.9	C,M
M1.0 to M6.9	M
M7.0 to M9.9	M,X
X1.0 and higher	X

In summary, if a certain solar flare occurred in a sub-level of more than 7.0, it was considered that the *adjacent classes* belongs to that class and the immediate highest one. For example, if a solar flare of class C7.0 occurred, the *adjacent classes* considered in the training dataset was CM. Thus, we reassigned the *actual classes* of the testing dataset according to Table 6.15. Now, the metrics were calculated comparing the *forecasting results* against the *adjacent classes (the new Actual* classification. Table 6.16 shows the results obtained in Experiment-3.2.

Table 6.16: Experiments 3.2 of ECID - Results of the analysis of adjacent classes together with the inclusion of SVM in the Aggregation Method

	AB	***C***	***M***	***X***	***overall multi-label metrics***
multi-label recall	91,6	97,2	100,0	76,5	91,3
multi-label precision	100,0	96,5	95,0	98,0	97,3
multi-label F-Measure	98,2	95,5	89,1	74,0	89,2
multi-label error	4,9	1,5	0,0	3,1	2,4

We observe in Table 6.16 that the use of adjacent classes, decreased the *overall multi-label recall* in comparison with Experiment 3.1 (Table 6.13), but the *overall multi-label F-Measure* remained almost similar. However, the most interesting result were a significant increase in the *overall multi-label precision*, and a significant decrease in the *overall multi-label error* compared to Experiment 3.1. Thus, by handling the adjacent class issue using Ensemble of Classifiers, ECID was able to decrease the *overall multi-label error*.

Additionally, Table 6.17 shows that the *overall aggregated multi-label recall* obtained in Experiment 3.2 was 76%, the *overall aggregated multi-label precision*, 72%, and the *overall aggregated multi-label F-Measure* achieved 75%.

Table 6.17: Overall aggregated metrics - Experiment 3.2

	Experiment 3.2
overall aggregated multi-label recall	76%
overall aggregated multi-label precision	72%
overall aggregated multi-label F-measure	75%

Hence, considering the adjacent classes, the *overall aggregated recall* decreased, but it still remained in the domain specialist needs. And, the more important, the *overall aggregated precision and F-Measure* increased substantially compared with Phases 1 and 2.

The results obtained in Phase-3 shows that ECID handles the forecasting in adjacent classes in a satisfactory manner.

6.3 Overall Analysis of Experimental Results

According to Section 1.4.1, the following metrics and correspondent thresholds were set to validate the hypothesis:

1. Minimum True Positive Rate for each class: 70%;
2. Maximum Classification Error for each class: 30%;
3. Forecast horizon: one day;

These goals were achieved by this thesis work, because:

1. The minimum *multi-label recall* obtained in Experiment 3.2 reached 76.5% for class X. This result is greater than the threshold established, 70%;

2. The maximum *multi-label error* were lower than 30%, reaching a maximum of 5.0% for class AB in Experiment 3.2;

3. The forecast horizon used in all the experiments was one day as initially defined;

Comparing ECID with the main solar flare forecasting methods found in literature, it is possible to comment that:

In literature, Solar flare forecasting methods usually aggregate more than one class as the Positive and Negative turning it a binary classification model. Instead of this, our approach provides the contribution to give multi-class and multi-label forecasting. Additionally, as the astrophysicists do not fully understand the physic model that guides these phenomena, each work considers a different set of solar features as input. Though works use datasets with different sizes and features, we present the results for each method in Table 6.18 in order to give an overview of the results obtained in each condition. Additionally, the comparison of our work to the ones found in literature is difficult, because ours is the only one which provides multi-label forecasting. Taking those facts into account, we can cite some significant works:

- In Nishizuka et al. (2017), it was developed a solar flare forecasting method that labels: (1) $classes = X$ as Positive, in a first experiment, and (2) $classes \geq M$ as Positive, in the last set of experiment;

- In Bobra e Couvidat (2015), Li e Zhu (2013), Yu et al. (2010), it was developed a solar flare forecasting method that labels $classes \geq M$ as Positive;

- In Yuan et al. (2010), it was developed a solar flare forecasting method that labels $classes \geq X$ as Positive according to an equation of flare importance;

- In Ahmed et al. (2013), it was developed a solar flare forecasting method that labels $classes \geq C$ as Positive according to a machine learning method;

- In Colak e Qahwaji (2009), it was developed a solar flare forecasting method that labels $classes \geq C$ as Positive according to a machine learning method;

Table 6.18 presents the results obtained by the *Nishizuka, Bobra, Li, Ahmed, Yu, Yuan, and Colak* forecasting methods and our proposed method ECID:

Table 6.18: Experimental results of the solar flare forecasting methods

	TPR	TNR	ACC	FPR	Precision	TSS	F-Measure
SeMiner (>=C)	0,71	0,79	0,73	0,20	0,83	-	0,75
ECID (overall metrics)	0,91	0,96	0,95	0,04	0,97*	0,84	0,87
ECID (=X)	0,77	0,98	0,95	0,02	0,98*	0,60	0,73
Nishikawa (=X)	0,90	0,99	0,99	0,0003	0,89	0,91	0,89
Nishikawa (>=M)	0,91	0,99	0,99	0,002	0,92	0,91	0,92
Bobra (>=M)	0,71	0,98	0,97	-	0,80	0,70	0,75
Li (>=M)	0,73	0,78	0,77	-	-	-	-
Ahmed (>=C)	0,67	0,99	0,97	0,006	-	-	-
Yu (>=M)	0,85	0,87	-	-	-	-	-
Yuan (= X)	0,83	0,73	0,74	-	-	-	-
Colak (C,M,X)	0,81	-	-	0.30	-	-	-
Colak (=X)	0,98	-	-	0.97	-	-	-

A direct comparison was possible for SeMiner with Ahmed et al. (2013) work, because he established, like us, the Positive class as any event greater than or equal to C. The values informed in the table for SeMiner's experiment is related to *Test 6* of the second experiment.

The comparison of ECID was performed in a general way, because, as far as our knowledge, it was not found any multi-label solar flare forecasting method in the literature. Then, for this purpose, it was used the *overall metrics* calculated from the *individual multi-label metrics*. Additionally, it was also calculated the following metrics described in Section 2.5 to comparison purposes: *accuracy, false positive rate, and TSS*. The values informed in the table for ECID were obtained in *Experiment 3.2*. In the other hand, it was not possible to obtain all the calculated metrics for all the works. For example, the *error* like calculated in our work was not found in any other, though we believe it is of utmost relevance. In this context, ECID was directly compared to Colak's work, because it predicts individual classes, while it was possible to make a partial comparison with some of the other works, once these may individually forecast a specific class.

As shown in Table 6.18, the first method developed in this thesis, SeMiner, achieved results similar to AHMED et al., which also performed the forecasting of solar flares from class C and above. The main difference is that we used only the X-ray time series in the forecasting process unlike the compared work, which had used a set of magnetic features. Also, we used training and testing data from different years. Differently, AHMED et al. used a random selection to split the training and testing datasets. These randomized approaches may influence the results, because close instances may be divided in both datasets. We achieved a *TPR* of 71%, while

AHMED et al. achieved a *TPR* of 67%. SeMiner achieved a lower *TNR*.

Comparing *ECID (=X)* with *Nishikawa (=X)*: ECID achieved *multi-label precision* and *multi-label F-Measure* of 60%, 98% and 87%, respectively, close to *Nishikawa's* related metrics. The *multi-label recall (TPR)* was 19% lower than *Nishikawa's*.

Comparing *ECID(overall metrics)* with *Nishikawa* ($\geq X$): ECID results are slightly lower than *Nishikawa*, but our method also predicted flares of classes AB, C and M in a multi-label fashion, instead of *Nishikawa*, which performed a binary classification. Another comment regarding *Nishikawa* work is that it used cross-validation for validation. This approach, in this domain, tends to perform a biased validation, because it usually classifies correctly due to the separation approach of the instances in training and testing datasets.

Comparing *ECID(overall metrics)* with *Colaq (C,M,X)* : ECID achieved results of the *overall metrics* very close to COLAK; QAHWAJI, but the available metrics of this work was limited to the *TNR* and *FPR*. The main difference between ECID and his work is that our approach implements a multi-label forecasting method, unlike theirs.

Comparing *ECID(=X)* with *Colaq (=X)* : ECID achieved results of the *overall recall (TPR)* 27% lower than COLAK; QAHWAJI. However, our method achieved a *multi-label FPR* for class X 77% lower than *Colak's* work. It means that ECID committed much less false alarms than compared work. Also, the main difference between ECID and his work is that our approach implements a multi-label forecasting method, unlike theirs.

Comparing *overall metrics* with the other works: considering the *overall recall*, our work overcomes all the others, while the *overall precision* is very close to the other works. The work of BOBRA; COUVIDAT was entirely reproduced, and we found out that the authors removed the instances classified as C in either the training and testing datasets. It was done probably because the instances of class C confound the classifier in distinguishing the classes M and X. Thus, removing it may result in a model that best predicts instances of classes M and X. However, in a real forecasting method, it is not possible to remove the C instances. Thus, we can conclude that the results of this work are particular to the training and testing dataset employed in that method. As we divided the original dataset in 70% for training and 30% for testing ensuring that each dataset had different days and considering all classes, our results are more generalizable that these related works. The works of AHMED et al., YU et al. and YUAN et al. does not implement a multi-class approach, differently from ours. Finally, the multi-label approach implemented in ECID has the advantage to support the domain specialist in his task, using his/her knowledge.

Concluding, we handled the following open issues (described in Section 1.1.4) in the solar flare forecasting problem:

1. Imbalanced dataset;
2. Differentiation of solar flares that are in adjacent classes, and multi-class/multi-label forecasting;
3. Data evolution;
4. Most significant time intervals to consider the data for the forecasting process;
5. Knowledge of domain specialists within the method;

Issues 1 and 2 were validated considering the results achieved in Experiment 3.2. The experiments were tested in an actual imbalanced dataset, and the results increased as we added adjacent classes in the analysis of the final forecasting. Another indication of our successful handling of the imbalanced dataset is that as we can see in Figure 6.7, there was a significant increase in the *overall metrics* comparing SeMiner (which does not handle the imbalanced issue) and ECID (which deals with this issue). Additionally, the threshold established by the domain specialist for each class was achieved according to Table 6.16.

Issues 3, 4 and 5 were validated in SeMiner experiments in Figure 6.2 and corroborated by the insertion of SS algorithm inside ECID method. As SS foundation is based on the analysis of the time series historical evolution, the results showed that our assumption to consider that was correct. Also SS allowed to be configured with simple, efficient and necessary domain specialist knowledge. This approach was also shown correct, because the most significant interval to be considered for data gathering for the forecasting process was two days, according to the experiments of SeMiner presented in Figure 6.4.

Thus, we believe that the proposed approach is a valid and useful contribution to the area of solar flare forecasting, aiding the domain specialist in his/her daily task of constructing space weather bulletins containing the solar flare predictions.

IV CONCLUSIONS

Chapter 7
CONCLUSIONS

Nowadays, solar flare forecasting is an application domain very relevant due to the high intensity impacts on Earth's electronic devices. This thesis dealt with the open issues related to the solar flare forecasting domain. For this purpose, it was also necessary to improve the state of the art of computational approaches. As presented in Chapter 3, according to the review of the literature, it was not found a computational method that handled all the open issues related to solar flare forecasting.

We showed in this thesis how our method solved a series of open issues considering the main related works. So, in order to tackle these issues, we developed two methods: SeMiner and ECID. These issues and how they were tackled are summarized next:

1. Imbalanced dataset: this issue was tackled by ECID, which performs a random stratified undersampling of the original dataset to produce four balanced datasets. These are submitted to an EC that builds four forecasting models, one for each solar flare class, and returns a voting count list. This list guides a decision rule to produce the final multi-label forecasting. It was shown in the Chapter 6 that the method produced interesting results for each solar flare class;

2. Data evolution: this issue was handled by employing the SeMiner method through the SS algorithm. This algorithm takes into account the historical evolution of a time series by transforming a sub-series in sequences using a sliding-window approach. The results obtained in SeMiner experiments were accurate in the forecasting of solar flares higher than or equal to C. ECID also used an extension of the original SS algorithm, so that its good results also corroborate the successful usage of the historical evolution of solar data within the forecasting method;

3. Difficulty to distinguish adjacent classes: this issue was tackled in ECID through the

usage of a multi-label approach. The multi-label forecasting given by ECID proved to be a valid way to support the domain specialist to distinguish adjacent classes. Experiment 3.2 of ECID shows the increase of the *overall aggregated multi-label metrics* when considering the adjacent classes in the method's analysis;

4. Incorporation of domain specialists knowledge within the method: SeMiner is a parameterized method. It is configured with information regarding the length of a sequence (*current window* parameter of SS), the advance of the forecasting (*jump* parameter), and the forecasting horizon (*future window*). Tests performed in SeMiner showed that the appropriate setting of these parameters had improved the forecasting results as shown in the second SeMiner's experiment (Section 6.1.2).

5. Selection of the most significant period: this issue was handled in SeMiner by using a feature selection method together with SS. This combination resulted in the selection of the intervals that contained the most significant data used to distinguish solar flares. Figure 6.4 shows that the usage of data recorded two days before a solar flare had accurately distinguished the events.

The first method developed (SeMiner) aimed to deal with data evolution, the inclusion of specialist knowledge in the forecasting process and the analysis of the most significant periods of data to be considered in the process. ECID provided a multi-class and multi-label forecasting. It handled more than one solar feature, dealt with imbalanced datasets and handled the adjacent classes issue.

Additionally, ECID consisted a new computational technique that provides a multi-class and multi-label forecasting using *n* time series through balancing techniques and a modified ensemble of classifier providing a computational solution for all requirements of the solar flare forecasting domain.

Next, the most relevant contributions of our work are described.

7.1 Most relevant contributions

SeMiner deals with a crucial step in the process of forecasting solar flares by using knowledge from domain specialists: the preprocessing step. Another distinguishing characteristic compared to other forecasting methods is that the historical evolution of solar time series was considered to produce the forecasting model. In addition, SeMiner also identifies the most significant time intervals to be considered in the forecasting method. It was concluded that, in the scope of our experiments, the best time interval was within two days for current window and comprised both, the initial and end periods of the first day, and the last 16 hours of the second day. Finally, performance optimization was proposed and implemented for SeMiner. The core module of SeMiner, the SS algorithm, was parallelized using CUDA, targeting parallel execution in GPUs. The strategy showed to be 4.36 times faster than its pure C sequential version. As the experiments showed, the pre-processing spent a low time to be concluded due to the cleaning task performed in the original dataset. Also, the SS algorithm has complexity of order n, so that it turns its execution scalable. Thus, the parallel implementation prepares SS to be used for large amount of data, even though the experiments and its characteristics did not showed this need at this moment.

ECID method provides a tool to support the specialist decision in the solar flare forecasting task. It provides multi-class and multi-labeled forecasting. This strategy differentiates from other forecasting methods because its results indicate the solar flare classes most probably to occur in a forecasting horizon. This result can support the domain specialist in situations when it is difficult to distinguish between two adjacent classes. ECID employs a stratified random sampling for the training of base inducers, strengthen their sensitivity to one individual class. Using a modified bootstrap approach, its aggregation method combines the inducers results enabling multi-class and multi-label forecasting.

The SeMiner and ECID foundations provided relevant contributions for the enhancement of the state of the art of the solar flare forecasting domain as well as it improved the state of the art of computational approaches. Mainly, to the techniques that deals with multi-class and multi-label forecasting through the use of time series classification with imbalanced time series.

The most relevant contributions of our work were:

- Contributions in the state of the art of solar flare forecasting methods:
  - SeMiner method showed that the inclusion of data evolution and the specialist knowledge and aims, instead of using snapshots of solar features in an inflexible method proved to be a better approach in the forecasting process;
  - ECID method showed that the use of ensemble of classifiers and the inclusion of techniques to handle imbalanced datasets is a good strategy due to the high imbalanced dataset and the multi-label requirements of this application domain;
  - Dataset construction: we assembled a flexible dataset of two solar time series of different sample rates used in our method and, which may be used in extensions of our work. The main advantage of this approach is that it is easily configurable due to

the equalization strategy used. As time series are joined by equalizing their sample rates, new time series may be easily added in the forecasting method;

  – An incremental experimental evaluation of several solar features and configurations, which gives a clue regarding the most significant features to be used and the most significant periods of data to be considered in the forecasting process;
  – Development of an optimized version of SS. The sequential version of SS algorithm has computational complexity of *O(n)*. Thus, parallelization techniques are not critical to its execution. However, we developed a parallel version of SS that reduces in four times the execution time. Consequently, as solar data increases along the time, the algorithm is prepared to run in an optimized manner.

- ECID was developed to provide a multi-class and multi-label forecasting using n time series through balancing techniques and a modified ensemble of classifier. Consequently, its contributions in the state of the art of computational are:
  – ECID improved the current Bootstrap Aggregating method (Bagging) providing means to the binary inducers to handle datasets with different sizes. This was possible due to the strategy used in the aggregation method of giving more voting power to the least most voted;
  – It was also designed a new efficient strategy of spreading the multi-label results by analyzing the voting table through the SVM classifier.

7.2 Scientific Publications resulting from this thesis

During the doctoral period, three articles were published:

- Díscola Junior, S. L., Cecatto, J. R., Xavier Ribeiro, M., Merino Fernandes, M. (2019). Handling imbalanced time series through Ensemble of Classifiers: a Multiclass approach for solar flare forecasting. In 16th International Conference on Information Technology : New Generations - Advances in Intelligent Systems & Computing Series. Las Vegas, Nevada, USA: Springer US.
- Díscola Junior, S. L., Cecatto, J. R., Merino Fernandes, M., Xavier Ribeiro, M. (2018). SeMiner: A Flexible Sequence Miner Method to Forecast Solar Time Series. Information, 9(1), 8.

- Discola.Jr., S. L., Cecatto, J. R., Fernandes, M. M., & Ribeiro, M. X. (2017). An Optimized Data Mining Method to Support Solar Flare Forecast. In 14th International Conference on Information Technology : New Generations - Advances in Intelligent Systems & Computing Series. Las Vegas,Nevada, USA: Springer US.

7.3 Future Works

The main proposal for future works are:

- Improve ECID aiming to use different solar features as input for different base inducers in the proposed Ensemble of Classifiers;
- Improve the aggregation method of ECID to consider the information of the forecasting day. This was a suggestion of the specialist, because according to his/her experience, solar flare forecasting methods should also verify recent data to increase its predictability;
- Develop a functional software using ECID method to actually support the domain specialist in his/her daily work.

REFERENCES

Abd Elrahman, S. M.; ABRAHAM, A. Class imbalance problem using a hybrid ensemble approach. *International Journal of Hybrid Intelligent Systems*, v. 12, n. 4, p. 219–227, 2015. ISSN 14485869.

AGRAWAL, R.; FALOUTSOS, C.; SWAMI, A. Efficient similarity search in sequence databases. In: . [S.l.]: Springer, Berlin, Heidelberg, 1993. p. 69–84.

AHA, D. W.; KIBLER, D.; ALBERT, M. K. Instance-based learning algorithms. *Machine Learning*, v. 6, n. 1, p. 37–66, 1991.

AHMED, O. W. et al. Solar flare prediction using advanced feature extraction, machine learning, and feature selection. *Solar Physics*, v. 283, n. 1, p. 157–175, nov 2013.

Alonso González, C. J.; GONZÁLEZ, C. J. A.; DIEZ, J. J. R. BOOSTING INTERVAL-BASED LITERALS: VARIABLE LENGTH AND EARLY CLASSIFICATION. *Series in Machine Perception and Artificial Intelligence*, p. 149–171, 2002.

BAGNALL, A.; JANACEK, G. A Run Length Transformation for Discriminating Between Auto Regressive Time Series. *Journal of Classification*, Springer US, v. 31, n. 2, p. 154–178, jul 2014. ISSN 0176-4268.

BAGNALL, A. et al. The great time series classification bake off: a review and experimental evaluation of recent algorithmic advances. *Data Mining and Knowledge Discovery*, Springer US, v. 31, n. 3, p. 606–660, may 2017. ISSN 1384-5810.

BALL, N. M.; BRUNNER, R. J. DATA MINING AND MACHINE LEARNING IN ASTRONOMY. *International Journal of Modern Physics D*, v. 19, n. 07, p. 1049–1106, jul 2010. ISSN 0218-2718.

BARNES, G. et al. Probabilistic forecasting of solar flares from vector magnetogram data. *Space Weather*, v. 5, n. 9, sep 2007. ISSN 15427390.

BASU, S. et al. Specification of the occurrence of equatorial ionospheric scintillations during the main phase of large magnetic storms within solar cycle 23. *Radio Science*, v. 45, n. 5, p. 1–15, 2010.

BATISTA, G. E. A. P. A. et al. CID: an efficient complexity-invariant distance for time series. *Data Mining and Knowledge Discovery*, Kluwer Academic Publishers, v. 28, n. 3, p. 634–669, may 2014. ISSN 1384-5810.

BATISTA, G. E. A. P. A.; PRATI, R. C.; MONARD, M. C. A study of the behavior of several methods for balancing machine learning training data. *ACM SIGKDD Explorations Newsletter*, ACM, v. 6, n. 1, p. 20, jun 2004. ISSN 19310145.

BAYDOGAN, M. G.; RUNGER, G. Time series representation and similarity based on local autopatterns. *Data Mining and Knowledge Discovery*, Springer US, v. 30, n. 2, p. 476–509, mar 2016. ISSN 1384-5810.

BAYDOGAN, M. G.; RUNGER, G.; TUV, E. A Bag-of-Features Framework to Classify Time Series. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, v. 35, n. 11, p. 2796–2802, nov 2013. ISSN 0162-8828.

BESNARD, P.; HANKS, S. *Uncertainty in artificial intelligence : proceedings of the Eleventh Conference (1995) : August 18-20, 1995*. [S.l.]: Morgan Kaufmann Publishers, 1995. 591 p. ISBN 1558603859.

BOBRA, M. G.; COUVIDAT, S. Solar flare prediction using sdo /hmi vector magnetic field data with a machine-learning algorithm. *The Astrophysical Journal*, v. 798, n. 2, p. 135, jan 2015.

BOSTROM, A.; BAGNALL, A. Binary Shapelet Transform for Multiclass Time Series Classification. In: . [S.l.]: Springer, Cham, 2015. p. 257–269.

BRANCO, P.; TORGO, L.; RIBEIRO, R. P. A Survey of Predictive Modeling on Imbalanced Domains. *ACM Computing Surveys*, ACM, v. 49, n. 2, p. 1–50, aug 2016. ISSN 03600300.

BREIMAN, L. Bagging predictors. *Machine Learning*, Kluwer Academic Publishers, v. 24, n. 2, p. 123–140, aug 1996. ISSN 0885-6125.

BROCKWELL, P.; DAVIS, R. *Introduction to Time Series and Forecasting*. New York: Springer-Verlag New York, 2002. 437 p. ISSN 00401706. ISBN 0387953515.

CAO, H. et al. SPO: Structure Preserving Oversampling for Imbalanced Time Series Classification. In: *2011 IEEE 11th International Conference on Data Mining*. [S.l.]: IEEE, 2011. p. 1008–1013. ISBN 978-1-4577-2075-8.

CARRINGTON, R. C. Description of a Singular Appearance seen in the Sun on September 1, 1859. *Monthly Notices of the Royal Astronomical Society*, Oxford University Press, v. 20, n. 1, p. 13–15, nov 1859. ISSN 0035-8711.

CHAWLA, N. V. et al. SMOTE: Synthetic Minority Over-sampling Technique. *J. Artif. Int. Res.*, AI Access Foundation, USA, v. 16, n. 1, p. 321–357, 2002. ISSN 1076-9757.

CHEN, S.; GOO, Y.-J. J.; SHEN, Z.-D. A hybrid approach of stepwise regression, logistic regression, support vector machine, and decision tree for forecasting fraudulent financial statements. *The Scientific World Journal*, Hindawi Publishing Corporation, v. 2014, p. 1–9, 2014.

COLAK, T.; QAHWAJI, R. Automated Solar Activity Prediction: A hybrid computer platform using machine learning and solar imaging for automated prediction of solar flares. *Space Weather*, v. 7, n. 6, p. n/a–n/a, jun 2009. ISSN 15427390.

CORPORATION_NVIDIA. *CUDA C PROGRAMMING GUIDE*. 2017. Disponível em: ⟨http://docs.nvidia.com/pdf/CUDA{\\_}C{\\_}Programming{\\_}⟩.

CORTES, C.; VAPNIK, V. *Machine Learning, Support Vector Networks*. Boston: Kluwer Academic Publishers, 1995. 273–297 p.

DHURJAD, R. K.; BANAIT, S. S. Imbalanced Time Series Data Classification Using Oversampling Technique. *International Journal of Electronics, Communication and Soft Computing Science and Engineering*, n. Special Issue of IJECSCSE (International Conference on "Emerging Trends in Computer Engineering,Science and Information Technology"- 2015), p. 75–80, 2015.

DING, C.; HE, X. Cluster Aggregate Inequality and Multi-level Hierarchical Clustering. In: . [S.l.]: Springer, Berlin, Heidelberg, 2005. p. 71–83.

DONGXIAO, N.; TIANNAN, M.; BINGYI, L. Power load forecasting by wavelet least squares support vector machine with improved fruit fly optimization algorithm. *Journal of Combinatorial Optimization*, Springer US, v. 1, p. 1–22, 2016.

DRIEL-GESZTELYI LIDIA; GREEN, L. M. van. Evolution of Active Regions. *Living Reviews in Solar Physics*, v. 12, n. 1, p. 1, 2015. ISSN 1614-4961.

DU, P. et al. Multiple Classifier System for Remote Sensing Image Classification: A Review. *Sensors*, Molecular Diversity Preservation International, v. 12, n. 4, p. 4764–4792, apr 2012. ISSN 1424-8220.

ESLING, P.; AGON, C. Time-series data mining. *ACM Computing Surveys*, ACM, v. 45, n. 1, p. 1–34, nov 2012. ISSN 03600300.

FERNANDEZ, A. et al. SMOTE for Learning from Imbalanced Data: Progress and Challenges, Marking the 15-year Anniversary. *Journal of Artificial Intelligence Research*, v. 61, p. 863–905, apr 2018. ISSN 1076-9757.

FERNÁNDEZ, A. et al. An insight into imbalanced Big Data classification: outcomes and challenges. *Complex & Intelligent Systems*, Springer Berlin Heidelberg, v. 3, n. 2, p. 105–120, jun 2017. ISSN 2199-4536.

FREUND, Y.; SCHAPIRE, R. E. A Decision-Theoretic Generalization of On-Line Learning and an Application to Boosting. *Journal of Computer and System Sciences*, v. 55, n. 1, p. 119–139, aug 1997. ISSN 00220000.

GALAR, M. et al. A Review on Ensembles for the Class Imbalance Problem: Bagging-, Boosting-, and Hybrid-Based Approaches. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)*, v. 42, n. 4, p. 463–484, jul 2012. ISSN 1094-6977.

GALLAGHER, P. T.; MOON, Y.-J.; WANG, H. Active-region monitoring and flare forecasting i. data processing and first results. *Solar Physics*, Kluwer Academic Publishers, v. 209, n. 1, p. 171–183, 2002.

GODBOLE, S.; SARAWAGI, S. Discriminative Methods for Multi-labeled Classification. In: DAI, H.; SRIKANT, R.; ZHANG, C. (Ed.). *Advances in Knowledge Discovery and Data Mining*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2004. p. 22–30.

HALL, M. et al. The weka data mining software: An update. *SIGKDD Explorations*, v. 11, n. 1, p. 10–18, 2009.

HART, P.; P. The condensed nearest neighbor rule (Corresp.). *IEEE Transactions on Information Theory*, IEEE Press, v. 14, n. 3, p. 515–516, may 1968. ISSN 0018-9448.

HE, G. et al. Early classification on multivariate time series. *Neurocomputing*, Elsevier, v. 149, p. 777–787, feb 2015. ISSN 0925-2312.

HE, G. et al. An ensemble of shapelet-based classifiers on inter-class and intra-class imbalanced multivariate time series at the early stage. *Soft Computing*, Springer Berlin Heidelberg, p. 1–18, jun 2018. ISSN 1432-7643.

HOLMAN, G. D. The Mysterious Origins of Solar Flares. *Scientific American*, v. 294, n. 4, p. 38–45, apr 2006. ISSN 0036-8733.

HOLTE, R. C. Very simple classification rules perform well on most commonly used datasets. *Machine Learning*, Kluwer Academic Publishers-Plenum Publishers, v. 11, n. 1, p. 63–90, 1993.

HUANG, X.; WANG, H.; DAI, X. Influences of misprediction costs on solar flare prediction. *Science China Physics, Mechanics and Astronomy*, SP Science China Press, v. 55, n. 10, p. 1956–1962, oct 2012. ISSN 1674-7348.

HUANG, X. et al. Short-term solar flare prediction using predictor teams. *Solar Physics*, v. 263, n. 1-2, p. 175–184, apr 2010.

JEONG, Y.-S.; JEONG, M. K.; OMITAOMU, O. A. Weighted dynamic time warping for time series classification. *Pattern Recognition*, Elsevier Science Inc., v. 44, n. 9, p. 2231–2240, sep 2011. ISSN 00313203.

JI, C. et al. A Shapelet Selection Algorithm for Time Series Classification: New Directions. *Procedia Computer Science*, Elsevier, v. 129, p. 461–467, jan 2018. ISSN 1877-0509.

JOSHI, J. C.; SRIVASTAVA, S. A hidden markov model for avalanche forecasting on chowkibal?tangdhar road axis in indian himalayas. *Journal of Earth System Science*, Springer India, v. 123, n. 8, p. 1771–1779, dec 2014.

KATE, R. J. Using dynamic time warping distances as features for improved time series classification. *Data Mining and Knowledge Discovery*, Springer US, v. 30, n. 2, p. 283–312, mar 2016. ISSN 1384-5810.

KIRA, K.; RENDELL, L. A. A practical approach to feature selection. *Proceedings of the ninth international workshop on Machine learning*, Morgan Kaufmann Publishers Inc., p. 249–256, 1992.

KRAWCZYK, B. *Learning from imbalanced data: open challenges and future directions*. 2016.

KRAWCZYK, B. et al. Ensemble learning for data stream analysis: A survey. *Information Fusion*, v. 37, p. 132–156, 2017. ISSN 1566-2535.

LEE, K. et al. Solar Flare Occurrence Rate and Probability in Terms of the Sunspot Classification Supplemented with Sunspot Area and Its Changes. *Solar Physics*, v. 281, n. 2, p. 639–650, 2012. ISSN 1573-093X.

LI, R.; ZHU, J. Solar flare forecasting based on sequential sunspot data. *Research in Astronomy and Astrophysics*, v. 13, n. 9, p. 1118–1126, 2013.

LIANG, G. An Effective Method for Imbalanced Time Series Classification: Hybrid Sampling. In: . [S.l.]: Springer, Cham, 2013. p. 374–385.

LIANG, G.; ZHANG, C. A Comparative Study of Sampling Methods and Algorithms for Imbalanced Time Series Classification. In: *Proceedings of the 25th Australasian joint conference on Advances in Artificial Intelligence*. [S.l.]: Springer-Verlag, 2012. p. 637–648.

LIN, J.; KHADE, R.; LI, Y. Rotation-invariant similarity in time series using bag-of-patterns representation. *Journal of Intelligent Information Systems*, Springer US, v. 39, n. 2, p. 287–315, oct 2012. ISSN 0925-9902.

LONGADGE, R.; DONGRE, S. Class imbalance problem in data mining review. v. 2, 05 2013.

MAIMON, O.; ROKACH, L. *Data Mining and Knowledge Discovery Handbook.* [S.l.]: Springer US, 2010. ISBN 9780387098227.

MAKRIDAKIS, S. Time series prediction: Forecasting the future and understanding the past. *International Journal of Forecasting*, v. 10, p. 463–466, 1994.

MARTEAU, P.-F. Time Warp Edit Distance with Stiffness Adjustment for Time Series Matching. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, v. 31, n. 2, p. 306–318, feb 2009. ISSN 0162-8828.

MCINTOSH, P. S. The classification of sunspot groups. *Solar Physics*, Kluwer Academic Publishers, v. 125, n. 2, p. 251–267, 1990. ISSN 00380938.

MEKANIK, F. et al. Multiple regression and artificial neural network for long-term rainfall forecasting using large scale climate modes. *Journal of Hydrology*, Elsevier B.V., v. 503, p. 11–21, 2013.

MENDES-MOREIRA, J. et al. Ensemble approaches for regression. *ACM Computing Surveys*, ACM, v. 45, n. 1, p. 1–40, nov 2012. ISSN 03600300.

NISHIZUKA, N. et al. Solar Flare Prediction Model with Three Machine-learning Algorithms using Ultraviolet Brightening and Vector Magnetograms. *The Astrophysical Journal*, IOP Publishing, v. 835, n. 2, p. 156, jan 2017. ISSN 1538-4357.

NOAAEVENTREPORT. *NOAA-SOLAR AND GEOPHYSICAL EVENT REPORTS*. 2019. Disponível em: ⟨https://www.swpc.noaa.gov/products/solar-and-geophysical-event-reports⟩.

OLIVEIRA, M.; TORGO, L. Ensembles for Time Series Forecasting. In: PHUNG, D.; LI, H. (Ed.). *Proceedings of the Sixth Asian Conference on Machine Learning*. Nha Trang City, Vietnam: PMLR, 2015. (Proceedings of Machine Learning Research, v. 39), p. 360–370.

PEDRO, H. T. C.; COIMBRA, C. F. M. Nearest-neighbor methodology for prediction of intra-hour global horizontal and direct normal irradiances. *Renewable Energy*, Elsevier Ltd, v. 80, p. 770–782, 2015.

PRATI, R.; BATISTA, G.; MONARD, M.-C. Data mining with imbalanced class distributions: Concepts and methods. In: *Paper presented at the IICAI*. [S.l.: s.n.], 2009. p. 359–376.

PRATI, R. C.; BATISTA, G. E. A. P. A.; MONARD, M. C. A Study with Class Imbalance and Random Sampling for a Decision Tree Learning System. In: *Artificial Intelligence in Theory and Practice II*. Boston, MA: Springer US, 2008. p. 131–140.

PRIEST, E.; FORBES, T. The magnetic nature of solar flares. *The Astronomy and Astrophysics Review*, Springer-Verlag, v. 10, n. 4, p. 313–377, mar 2002. ISSN 0935-4956.

QAHWAJI, R.; COLAK, T. Automatic short-term solar flare prediction using machine learning and sunspot associations. *Solar Physics*, v. 241, n. 1, p. 195–211, feb 2007.

QUINLAN, J. R.; ROSS, J. *C4.5 : programs for machine learning*. [S.l.]: Morgan Kaufmann Publishers, 1993. 302 p.

RABOONIK, A. et al. PREDICTION OF SOLAR FLARES USING UNIQUE SIGNATURES OF MAGNETIC FIELD IMAGES. *The Astrophysical Journal*, IOP Publishing, v. 834, n. 1, p. 11, dec 2016. ISSN 1538-4357.

RAKTHANMANON, T. et al. Addressing Big Data Time Series: Mining Trillions of Time Series Subsequences Under Dynamic Time Warping. *ACM Transactions on Knowledge Discovery from Data (TKDD)*, ACM, v. 7, n. 3, p. 10, 2013. ISSN 1556-4681.

RAKTHANMANON, T.; KEOGH, E. Fast Shapelets: A Scalable Algorithm for Discovering Time Series Shapelets. In: *Proceedings of the 2013 SIAM International Conference on Data Mining*. Philadelphia, PA: Society for Industrial and Applied Mathematics, 2013. p. 668–676. ISBN 978-1-61197-262-7.

RÄTSCH, G.; ONODA, T.; MÜLLER, K.-R. Soft Margins for AdaBoost. *Machine Learning*, Kluwer Academic Publishers, v. 42, n. 3, p. 287–320, 2001. ISSN 08856125.

RIBEIRO, M. X. et al. Mining statistical association rules to select the most relevant medical image features. In: *Mining Complex Data*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2009. p. 113–131.

RODRÍGUEZ, J. J.; ALONSO, C. J.; MAESTRO, J. A. Support vector machines of interval-based features for time series classification. *Knowledge-Based Systems*, Elsevier, v. 18, n. 4-5, p. 171–178, aug 2005. ISSN 0950-7051.

SAGI, O.; ROKACH, L. Ensemble learning: A survey. *Wiley Interdisciplinary Reviews: Data Mining and Knowledge Discovery*, Wiley-Blackwell, v. 8, n. 4, p. e1249, jul 2018. ISSN 19424787.

SAKOE, H.; CHIBA, S. Dynamic programming algorithm optimization for spoken word recognition. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, v. 26, n. 1, p. 43–49, feb 1978. ISSN 0096-3518.

SCHÄFER, P. The BOSS is concerned with time series classification in the presence of noise. *Data Mining and Knowledge Discovery*, Springer US, v. 29, n. 6, p. 1505–1530, nov 2015. ISSN 1384-5810.

SCHAPIRE, R. E.; SINGER, Y. Improved Boosting Algorithms Using Confidence-rated Predictions. *Machine Learning*, Kluwer Academic Publishers, v. 37, n. 3, p. 297–336, 1999. ISSN 08856125.

SCHOLKOPF, B.; BURGES, C. J. C.; SMOLA, A. J. *Advances in kernel methods : support vector learning*. [S.l.]: MIT Press, 1999. 376 p. ISBN 0262194163.

SENIN, P.; MALINCHIK, S. SAX-VSM: Interpretable Time Series Classification Using SAX and Vector Space Model. In: *2013 IEEE 13th International Conference on Data Mining*. [S.l.]: IEEE, 2013. p. 1175–1180. ISBN 978-0-7695-5108-1.

Senzhang Wang et al. Applying adaptive over-sampling technique based on data density and cost-sensitive SVM to imbalanced learning. In: *The 2012 International Joint Conference on Neural Networks (IJCNN)*. [S.l.]: IEEE, 2012. p. 1–8. ISBN 978-1-4673-1490-9.

SHARMA, S. et al. Machine learning techniques for data mining: A survey. In: *2013 IEEE International Conference on Computational Intelligence and Computing Research*. [S.l.]: IEEE, 2013. p. 1–6. ISBN 978-1-4799-1597-2.

SILVA, L. A. da; PERES, S. M.; BOSCHARIOLI, C. *Introdução à Mineração de Dados - Com Aplicação em R*. 1. ed. Rio de Janeiro: Elsevier, 2016. 277 p.

TANDBERG-HANSSEN, E.; EMSLIE, A. G. *The Physics of Solar Flares*. 1st. ed. New York: Cambridge University Press, 2009. 288 p. ISBN 9780521115520.

TSURUTANI, B. T. et al. A brief review of solar flare effects on the ionosphere. *Radio Science*, v. 44, n. 1, p. 0–0, feb 2009.

WACKLER, T. *National Space Weather Strategy Notices. Federal Register*. 2015. 24296–24297 p.

WANG, H. et al. Solar flare forecasting model supported with artificial neural network techniques. *Advances in Space Research*, v. 42, n. 9, p. 1464–1468, nov 2008.

WAZLAWICK, R. S. *Metodologia de pesquisa para ciência da computação*. Rio de Janeiro: Elsevier, 2009. 124 p. ISBN 978-85-352-3522-7.

WEBSITENOAA. *GOES X-ray Flux — NOAA website*. 2017. Disponível em: ⟨http://www.swpc.noaa.gov/products/goes-x-ray-flux⟩.

WOŹNIAK, M.; GRAÑA, M.; CORCHADO, E. A survey of multiple classifier systems as hybrid systems. *Information Fusion*, Elsevier Science Publishers B. V., v. 16, p. 3–17, mar 2014. ISSN 15662535.

XI, X. et al. Fast time series classification using numerosity reduction. In: *Proceedings of the 23rd international conference on Machine learning - ICML '06*. New York, New York, USA: ACM Press, 2006. p. 1033–1040. ISBN 1595933832.

XING, Z.; PEI, J.; YU, P. S. Early classification on time series. *Knowledge and Information Systems*, Springer-Verlag, v. 31, n. 1, p. 105–127, apr 2012. ISSN 0219-1377.

YADAV, N.; YADAV, A.; KUMAR, M. History of Neural Networks. In: *An Introduction to Neural Network Methods for Differential Equations*. [S.l.]: Springer Netherlands, 2015. cap. Chapter 2, p. 13–15.

YE, L.; KEOGH, E. Time series shapelets: a novel technique that allows accurate, interpretable and fast classification. *Data Mining and Knowledge Discovery*, Springer US, v. 22, n. 1-2, p. 149–182, jan 2011. ISSN 1384-5810.

YU, D. et al. Short-term solar flare prediction using multiresolution predictors. *The Astrophysical Journal*, v. 709, n. 1, p. 321–326, 2010.

YU, D. et al. Short-term solar flare prediction using a sequential supervised learning method. *Solphys*, v. 255, p. 91–105, 2009.

YUAN, Y. et al. Automated flare forecasting using a statistical learning technique. *Research in Astronomy and Astrophysics*, v. 10, n. 8, p. 785–796, aug 2010. ISSN 1674-4527.

ZAVVARI, A. et al. Solar flare m class prediction using artificial intelligence techniques. *Journal of Theoretical and Applied Information Technology*, v. 74, n. 1, 2015.

ZHANG, X.; LIU, J.; WANG, Q. Image feature extraction for solar flare prediction. In: *2011 4th International Congress on Image and Signal Processing*. Shanghai, China: IEEE, 2011. p. 910–914. ISBN 978-1-4244-9306-7.