Universitat Oberta de Catalunya (UOC)

Máster Universitario en Ciencia de Datos (*Data Science*)

TRABAJO FINAL DE MÁSTER

Área: 2

Feasibility study of a spectra-based classification of the Gaia Photometric Alerts

---

Autor: Mario Martínez García

Tutor: Laura Ruiz Dern

Profesor: Jordi Casas Roma

Barcelona, January 3, 2021

Créditos/Copyright

FICHA DEL TRABAJO FINAL

Título del trabajo:	*Feasibility study of a spectra-based classification of the Gaia Photometric Alerts*
Nombre del autor:	Mario Martínez García
Nombre del colaborador/a docente:	Laura Ruiz Dern
Nombre del PRA:	Jordi Casas Roma
Fecha de entrega (mm/aaaa):	01/2021
Titulación o programa:	Máster en ciencia de datos
Área del Trabajo Final:	Data Science in Astrophysics
Idioma del trabajo:	Inglés
Palabras clave	"Gaia", "Photometry", "Classification", "Machine Learning"
GitHub	Link to the project

Acknowledgements

I acknowledge ESA Gaia, DPAC and the Photometric Science Alerts Team for their contributions to the scientific world that have enabled me to carry out this project.

Special mention for my "partners in crime" in this last stage, Germán and Pablo.

To Julie, for helping me in what I have needed.

To Laura, for your kind attention and support.

Finally, thanks to my father, my mother and my sister for their unquestionable support all these years. My achievements are yours.

Let's continue.

Abstract

The photometric alerts obtained by the Gaia satellite are collected when a significant change from a constant magnitude is detected. This alert is recorded to be later studied and classified, in other words, to know what has caused it (variable star, microlensing effects, transits...). The project focuses on the alerts that have been published and classified in order to study the feasibility of automating the process of classifying these alerts.

After selecting the alerts with the greatest representation, a web scraping process is carried out where the photometric spectra of each of the alerts participating in the study are obtained. Once we obtain a dataset formed by the photometric spectra and the classification of the alert, various supervised machine learning techniques are implemented. Given the large volume of data we work with, a balanced random subset of 4000 elements is selected to obtain the best hyperparameters and evaluate the performance of the following classifiers: Decision Tree, Support Vector Machines, Random Forests and Gradient Boosting Classifier. This process is repeated on the complete dataset using the hyperparameters obtained in the subset. Finally, the performance of different models for an Artificial Neural Network is built and evaluated.

The best model obtained is the Gradient Boosting Classifier which, with a maximum depth of 7 nodes, 200 estimators and a learning rate of 0.1, gives an accuracy of 66.8%. Although the results are not excessively good, we can affirm that the classification of Gaia photometric alerts according to their spectra is feasible.

Keywords: "Gaia", "Photometry", "Classification", "Machine Learning", "Web Scraping".

Resumen

Las alertas fotométricas obtenidas por el satélite Gaia son recopiladas cuando se detecta un cambio significativo en una magnitud constante. Esta alerta se registra para ser posteriormente estudiada y poder clasificar la misma, es decir, conocer a qué ha sido debida (estrella variable, efectos de microlensing, tránsitos...). El proyecto se centra en las alertas que se han publicado y clasificado con el objetivo de estudiar la viabilidad de automatizar el proceso de clasificación de dichas alertas.

Tras seleccionar las alertas con mayor representación, se lleva a cabo un proceso de web scraping donde se obtienen los espectros fotométricos de cada una de las alertas que participan en el estudio. Una vez conseguimos una base de datos formada por los espectros fotométricos y la clasificación de la alerta, se implementan diversas técnicas de machine learning. Dado el gran volumen de datos con el que se trabaja, se selecciona un subconjunto aleatorio de 4000 elementos balanceado para obtener los mejores hiperpáremetros y evaluar el rendimiento sobre los siguientes clasificadores: Decision Tree, Support Vector Machines, Random Forests and Gradient Boosting Clasifier. Este proceso se repite sobre el conjunto completo de datos utilizando los hiperparámetros obtenidos en el subconjunto. Por último, se construyen y evaluan los rendimientos de diversos modelos para una Red Neuronal Artificial.

El mejor modelo obtenido es el Gradient Boosting Classifier que con una profundidad máxima de 7 nodos, 200 estimadores y un learning rate de 0.1, proporciona una precisión del 66.8%. Aunque los resultados no son excesivamente buenos, podemos afirmar que es viable la clasificación de las alertas fotométricas de Gaia en función de sus espectros.

Palabras clave: “Gaia”, “Photometry”, “Classification”, “Machine Learning”, “Web Scraping”.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Overview of the problem

This project focused on data science within the astrophysical context will use the light spectra patterns of the alerts obtained by the Gaia satellite in order to study the feasibility of classifying them automatically according to their origin (variable star, supernova, deflagration, gravitational phenomena, etc.)

Gaia is a space satellite that was sent into space in December 2013 from the European Spaceport in French Guiana with the aim of creating a three-dimensional map of our galaxy, the Milky Way, as well as revealing the composition, formation and evolution of this galaxy. The satellite is made up of two optical telescopes which, with the help of other instruments, enables us to accurately determine the position of the stars and divide their light into a spectrum for analysis. During its journey around space, the spacecraft turns slowly sweeping the two telescopes over the whole celestial sphere (Age15).

Thanks to this European Space Agency mission, we will be able to discover if we are in one of the spiral arms, if we are in a calm area and also we will be able to know how we are moving inside its structure.

The Gaia satellite has a rotational mechanism that allows it to explore the same work areas several times throughout its mission. If it detects any changes in magnitude within the same area, an alert will be registered and catalogued. Each of these alerts is available in the online public catalogue "Gaia Photometric Science Alert" (IoA15), where we can find a table with all the registered entries.

The difficulty lies in the information that can be collected and could be useful for a correct approach for classification, i.e. the types of alerts that can be identified or the size of the dataset among other problems that we will be solving and dealing with in detail.

1.2 Justification

The alert detection service does not incorporate any system to detect its origin automatically. This process is carried out manually when possible and the object in question can be identified. Hence, the alert classification process is not immediate.

The aim of this project is to provide a solution in order to speed up the classification process and facilitate the subsequent study of certain objects by the scientific community. The study belongs to a wider project in which some brushstrokes have begun to be made for specific events, but it is interesting to generalize to any type of event that can be detected.

1.3 Motivation

From my point of view, I consider this project as a tool which enables me to immerse myself completely in the world of data science, understand first-hand the concepts developed during the Master as well as some complementary ones.

Personally, with this work I was looking for the possibility of applying web scraping and machine learning techniques. The project offered by Laura Ruiz met these two conditions and was also linked to astrophysics, a branch in which I am quite motivated to work. It is exciting to be able to develop my work on this major project and I hope to provide my own small contribution to the Gaia satellite observations.

Furthermore, during the development of the work we would have the support of the *Science Alerts Gaia* team with whom we could fine-tune the objective of the project or even study similar proposals as long as it is in accordance with the alerts.

It should be noted that this project is undertaken with the aim of becoming a basis for future research in the same direction

1.4 Objectives

Main objective

- Feasibility of carrying out the classification of alerts through machine learning techniques

Secondary objectives

- Data selection
- Data pre-processing (Data cleaning and analysis)

- Extraction of the information by means of web scraping
- Preparation and processing of the extracted information
- Application of different machine learning techniques
- Application of Artificial Neural Networks
- Selection of the best methods of classification
- Validation of result
- Conclusion on the feasibility of classification

1.5 Methodology

In order to establish an order in the elaboration of the work, we use the CRISP-DM methodology which is capable of covering all the phases of the project as well as relationships between different tasks. This model comprises six major phases.

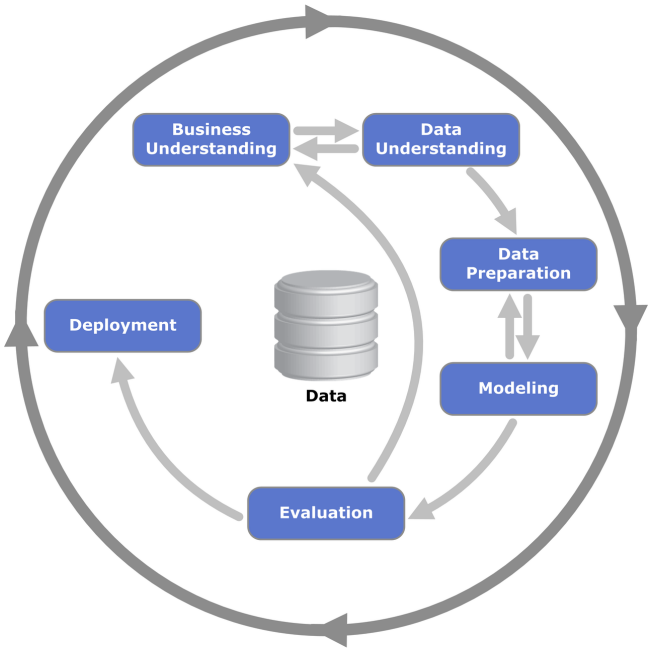


Figure 1.1: Relationship between the different CRISP-DM phases. Image by Kenneth Jense, used under CC BY 3.0

1. **Business Understanding**. In this phase we understand the objective of the work and draw up a preliminary plan to achieve the objectives.

2. **Data Understanding**. During this stage we tackle the interpretation of the initial dataset and the discovery of new subsets that may be enriching.

3. **Data Preparation**. This phase covers all the necessary activities to build the final dataset.

4. **Modelling**. At this stage, we focus on implementing different techniques that allow us to classify the alerts of our model.

5. **Evaluation**. In a final check, the results of the model are conscientiously evaluated.

6. **Publication**. The information obtained throughout the work is compiled and presented in a coherent manner.

It should be noted that we will make use of Python Language to work with the data collected, implement the accurate web scraping techniques, and set different machine learning models.

1.6 Planning

As established for each of the deliveries defined by the UOC, we draw up the following planning:

Phase	Task	Start	End	Duration
Phase 1	Definition and planning	16/09/2020	27/09/2020	11 days
Phase 2	State of the art	28/09/2020	18/10/2020	21 days
Phase 3	Design and implementation	19/10/2020	20/12/2020	63 days
Phase 4	Report Writing	21/12/2020	03/01/2021	13 days
Phase 5	Presentation	04/01/2021	10/01/2021	7 days

Table 1.1: Planning.

This table can be translated to a Gantt chart for a better visualization of the duration of each task.

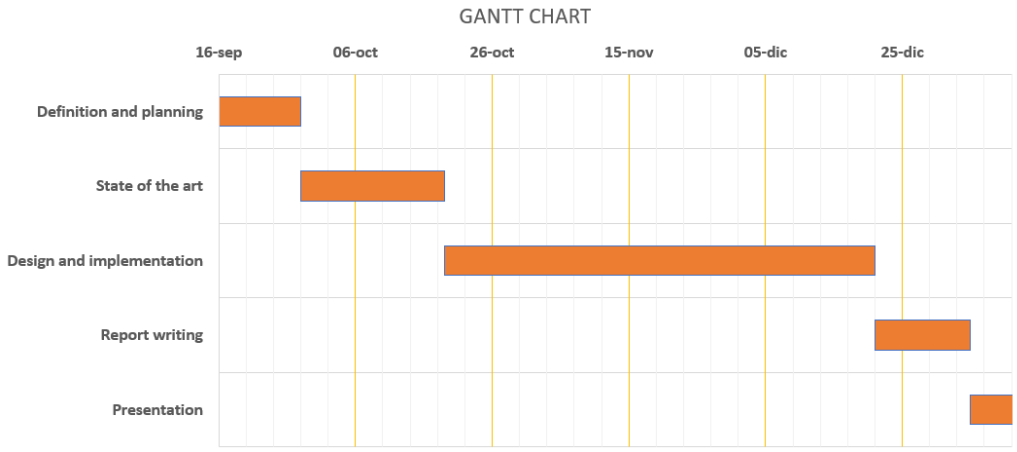


Figure 1.2: Gantt chart. Planning stages.

As we can see, the design and implementation stage corresponds to the bulk of the course where the secondary objectives are carried out to guide us towards fulfilling the main objective of the project. In phases, we find the following tasks:

Phase 1. Definition and planning.

- Determine the subject matter of the final work.
- Describe concisely what the project consists of.
- Explain the work motivation and justification.
- Define the goals.
- Carry out a plan.

Phase 2. State of the art.

- Justify with scientific evidence the project chosen.
- Search for the appropriate bibliography for the project.
- Refine the partial objectives defined in the previous activity.
- Identify the correct methodology and techniques.
- Organize and write the information coherently following the established scientific formalities.

Phase 3. Design and implementation

- Carry out the secondary objectives that lead us to the completion of the main objective following the established methodology.

Phase 4. Report writing.

- Documentation and justification of the project.
- Organize and write the information coherently following the established scientific formalities.

Phase 5. Presentation.

- Make a video-presentation to summarize the content of the project.

Chapter 2

State of the art

2.1 The Gaia Mission

As mentioned above, the main aim of Gaia is to measure the three-dimensional spatial and the three-dimensional velocity distribution of stars and to determine their astrophysical properties, such as effective temperature and surface gravity, in order to map and understand the formation, structure, and the past and future evolution of our Galaxy.

The scientific goals of the design reference mission rely on astrometry, supported by its photometric and spectroscopic surveys. The space environment and design of Gaia allows us to collect good photometric and spectroscopic results.

2.1.1 Gaia scientific goals

Below we summarize the most important contributions of a non-exhaustive list of scientific topics expected to be covered by Gaia (PdBB+16).

1. *Structure, dynamics, and evolution of the Galaxy.*

   Gaia was built with the purpose of answering the questions about the formation and evolution of the Galaxy through the analysis of the distribution and kinematics of the luminous and dark mass in the Galaxy. Although Gaia will only capture 1% of the stars in the Milky Way, it will consist of more than 1000 million stars which will be enough to carry out deep scientific researches.

2. *Star formation history of the Galaxy.*

   With the help of some physical properties that Gaia can obtain, like distance and metallicities, it will be possible to get absolute luminosities or even individual ages. By combining

the structure and dynamics of the stars with the information from the physical properties, it is possible to deduce the star formation histories of the stellar population in the Milky Way.

3. *Stellar physics and evolution.*

   The parallaxes will help us to achieve high-quality colour-magnitude diagrams and to make significant progress in stellar astrophysics. Furthermore, the symbiosis between Gaia astrometry and photometry will also contribute significantly to star formation studies.

4. *Stellar variability and distance scale.*

   The astrometric measurements will provide a census of variable stars with tens of millions of new variables, including rare objects. When Gaia detects unexpected photometric changes in transient objects, it captures them and alerts the community about subsequent observations.

5. *Binaries and multiple stars.*

   This mission will resolve many binaries and all instruments in Gaia will enable us to improve our understanding of multiple stars. The large number of objects that Gaia provides, gives us information about mass distribution and orbital eccentricities among binaries.

6. *Exoplanets.*

   Although the exoplanet research has been at its most dynamic in the past two decades it is expected that this mission will increase our knowledge about them providing relevant astrophysical parameters which have never been found before.

7. *Solar system.*

   The movement of solar system objects with respect to the stars smears their images and makes them less point-like. However, if this smearing is modest, Gaia will still detect the object. Asteroids are the most relevant solar system object group for Gaia because they remain typically point-like and have brightness in the dynamical range of Gaia. Gaia astrometry and photometry will provide a census of orbital parameters and taxonomy in a single, homogeneous photometric system.

8. *The local group.*

   The spatial resolution of Gaia allows us to resolve and observe the brightest individual stars. While for the faintest dwarf galaxies only a few dozen of the brightest stars are

observed, this number increases to thousands and millions of stars in Andromeda and the Large Magellanic Cloud, respectively. In this field, the mission will focus on dynamical interactions, transverse-velocity determination or stellar motions which could reveal the impact of dark matter among other physical processes.

9. *Unresolved galaxies, quasars, and the reference frame.*

   Gaia will provide a homogeneous, magnitude-limited sample of unresolved galaxies of which the most valuable measurements are the photometric observations.

   Although the intrinsic properties of quasars can be studied, they can also be used in comparisons of optical and radio reference frames.

10. *Fundamental physics.*

    The large amount of measurements that Gaia collects can be used in fundamental-physics experiments. There is a wide range of possibilities that physics can address such as testing theories or determining some important parameters.

2.1.2 The development and the scientific breakthroughs of Gaia

The Gaia Satellite took off on 19 December 2013 and the data collection started on 25 July 2014. A first batch of results, called Gaia DR1 was released on 15 September 2016 with only 14 months of data processed. In other words, Gaia DR1 is based on observations collected between 25 July 2014 and 16 September 2015. Although this first release based on just over one year of data was far from touching the core objective of the mission, it contained the largest position catalogue collection for 1.14 billion stars, a set of 2200 quasars and variable stars among others discoveries.

The second release contains the data collected between 25 July 2014 and 23 May 2016, spanning a period of 22 months of data collection. Almost two years later, on 25 April 2018, Gaia DR2 was published. Gaia DR2 contains astrometry, broad-band photometry, radial velocities, variable star classifications, the characterization of the corresponding light curves, and astrophysical parameter estimates for a total of 1.6 billion sources (Mig19). The following list of items shows some of the most important discoveries to come from the Gaia data so far:

- A panoramic map of the stellar streams of the Milky Way based upon astrometric and photometric measurements from the Gaia DR2 catalogue (MIM18).

- A sample of highly-probable members of the longest cold stream in the Milky Way, GD-1 (PWB18).

- Astrophysicists have debated the origin of Type Ia supernovas for decades. Thanks to Gaia DR2 has been found strong evidence for a theory dubbed the "dynamically driven double-degenerate double-detonation" scenario (SBG+18).

- Thanks to Gaia we know that the lost galaxy, called Gaia-Enceladus, was consumed by the Milky Way and we also know that the Milky Way is in the process of consuming the Magellanic Clouds (HBK+18).

- The Sagitario dwarf galaxy has been an important factor in the build-up of the Milky Way disc stellar mass (RLGBC20).

The next data release, Gaia Data Release 3, is already being prepared and it will be split into two releases. The early release called Gaia Early Data Release (Gaia EDR3) is available from 3 December 2020. The full Gaia Data Release 3 (Gaia DR3) is planned for the first half of 2022 (Age20).

2.2 The Gaia scanning law

The satellite sweeps the sky with two identical, three-mirror anastigmatic telescopes which are separated by the basic angle (106.5°). At its operating point, the second Lagrange point of the Sun-Earth-Moon system, Gaia makes use of the scanning law which enables them to scan the sky using uniform revolving scanning. The main properties of the nominal scanning law are the following (PdBB+16):

- The spin period of the satellite is exactly 6 hours.

- There is a fixed spin rate of 60 arcsec s^{-1} around the spacecraft spin axis (z)

- The solar-aspect angle between the Sun and the instrument z-axis (45°), is fixed to ensure maximum parallax sensitivity.

- The speed of the precession is as small as possible to limit the across-scan smearing of images. The precession period is 63 days.

These movements can be seen in Fig. 2.1.

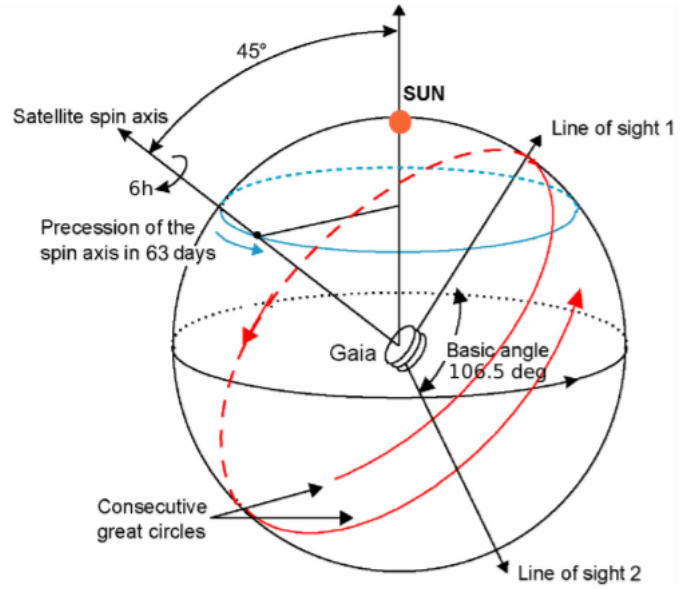


Figure 2.1: Representation of the Gaia scanning law (HV12).

In spite of the fact that the scanning law returns maximum sky-coverage uniformity, the number of times that an object is observed depends on its position in the sky, in particular on its ecliptic latitude.

We illustrate this process in Fig. 2.2, where we can see a simulation of a map with the estimated number of times that Gaia looks at each location in the sky.

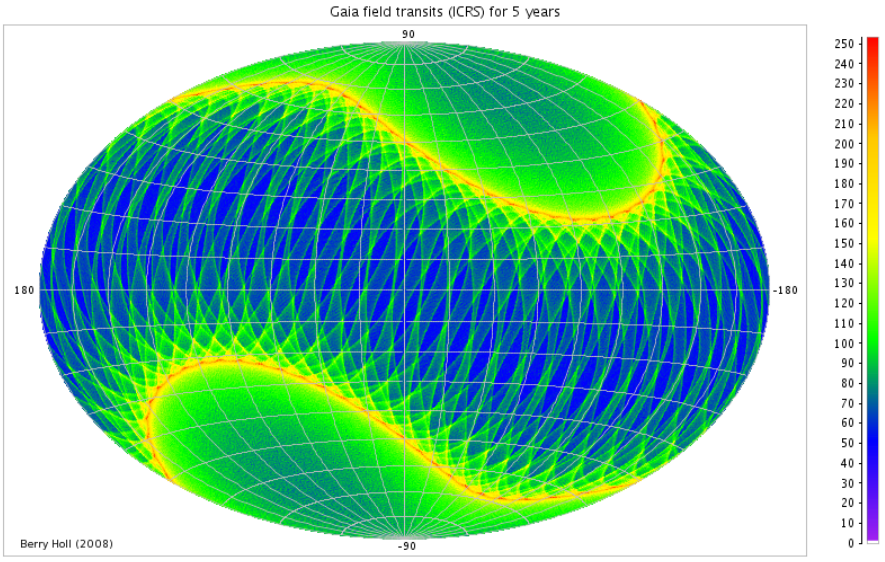


Figure 2.2: Simulation of the number of transits that Gaia observes during the first 5 years of the mission vs. sky position given in equatorial coordinates (HV12).

On the one hand, this scanning law guarantees that on average 70 transits per source will be observed along the mission. On the other hand, for sources close to the knots of the scanning law (ecliptic latitude 45°) up to 200 transits will be recorded by Gaia as can be seen in Fig. 2.2.

2.3 Gaia Focal Plane

Both telescopes have a focal plane assembly which is a cornerstone in the development of the mission. The focal plane has five main functions (PdBB+16):

1. Metrology
2. Object detection in the sky mapper.
3. Astrometry in the astrometric field.
4. Low-resolution spectro-photometry using blue and red photometers.
5. Spectroscopy using the radial-velocity spectrometer

The focal plane is illustrated in Fig. 2.3 and carries 106 charge-couple device (CCD) detectors which feature 7 CCD rows and 17 CCD strips. It is divided into four parts: Sky Mapper, Astrometric Field, Blue Photometer, Red Photometer and Radial Velocity Spectrometer.

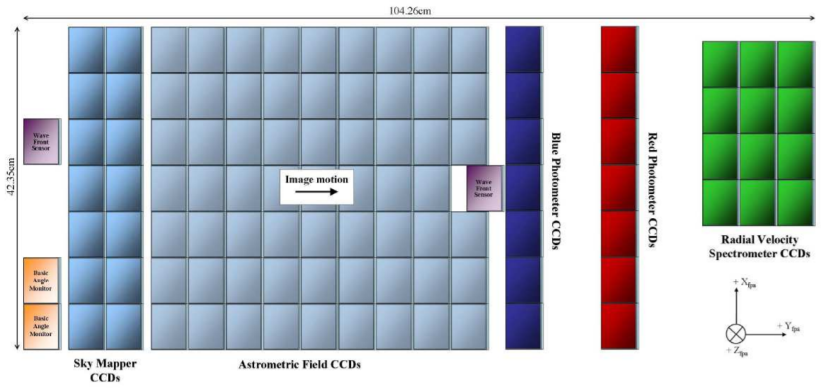


Figure 2.3: Gaia focal plane (JGC+10).

Each telescope captures the light passing through the focal plane and collects the information extracted from it as we can see in Fig. 2.4.

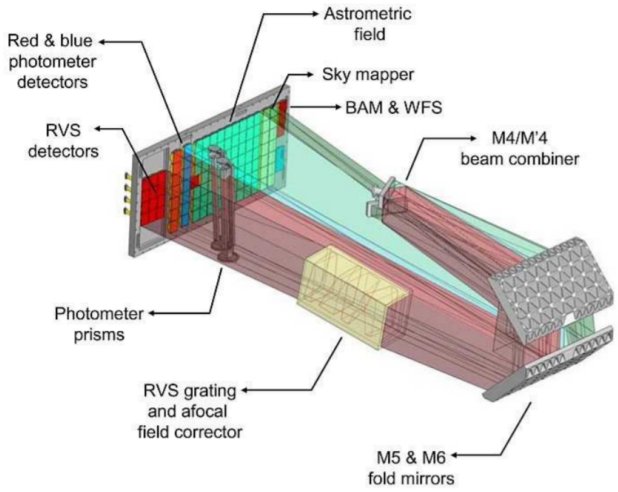


Figure 2.4: Light from the two Gaia telescopes is dispersed in wavelength (JGC+10).

Gaia Focal Plane supplies sixty-two CCDs to the Astrometric Field, fourteen CCDs for the BP and RP spectra (seven CCDs recorded in strips for each one) and twelve CCDs in the RVS instrument. The mirrors are coated with "Ag" and are the same for all instruments, while the coatings of the prisms act as low-pass and high-pass bands for BP/RP.

2.3.1 Gaia photometry

This section aims to provide a brief characterization of the Gaia passband: white light G, blue G_{BP}, red G_{RP} and G_{RVS}.

The Gaia photometry is obtained for every source by means of two low-resolution dispersion optics. One disperser, called BP (Blue Photometer) , operates in the wavelength range 330-680 nm. The other disperser, called RP (Red Photometer), covers the wavelength range 640-1050 nm.

The aims of the BP/RP photometric instrument are to measure the spectral energy distribution and to classify the sources by deriving the astrophysical characteristics (effective temperature, gravity and chemical composition) among other things (JGC+10). Furthermore, the results obtained by the BP/RP photometric instrument will be relevant in this project to classify the alerts.

In addition to these G_{BP} and G_{RP} passband, we also have a G and a G_{RVS} passband. G-magnitudes monitor the measurements of unfiltered (white) light in the Astrometric field (AF) of the focal plane from about 350 to 1000 nm. G_{RVS}-magnitudes correspond to the radial velocity instrument which disperse the light in the range 847-874 nm (region of de CaII triplet) and the integrated flux of the resulting spectrum can be seen as measured with a photometric

narrow band. In table 2.1 the values of the mean wavelength and the widths of these passbands are shown.

Band	G	G_{BP}	G_{RP}	G_{RVS}
$\lambda_0(nm)$	673	532	797	860
$\Delta\lambda(nm)$	440	253	296	28

Table 2.1: Central wavelength and FWHM (Full Width at Half Maximum) for the Gaia passband (JGC+10).

Below we can see a graph of the normalized energy in function of the wavelengths for each one of the Gaia passbands. As we can see, each passband is drawn in its wavelength range.

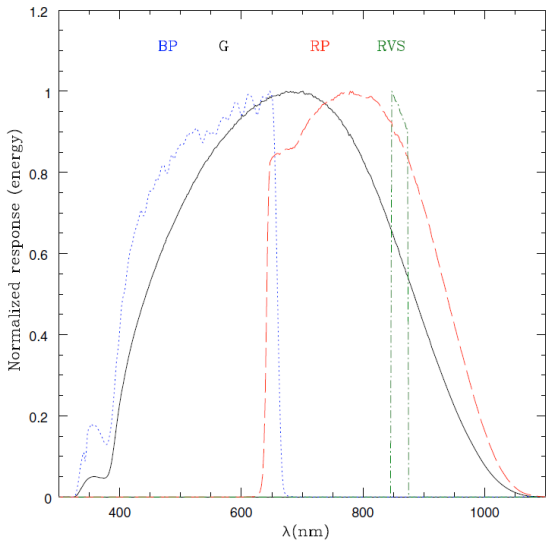


Figure 2.5: Gaia G (solid black line), GBP (dotted blue line), GRP (dashed red line) and GRVS (dot-dashed green line) normalized passbands (JGC+10).

The passbands shown in Fig. 2.5, are derived by the convolution of the response curves of the optics and the quantum efficiency (QE) curves of the CCDs.

2.4 Gaia Alerts

Throughout this project, we will handle the data of the photometric science alerts from Gaia which we can find on the website Gaia Alerts-University of Cambridge (IoA15). It should be noted that the first alerts of the mission were obtained on 30 August 2014 and to date Gaia is still collecting alerts.

The data gathered during the scanning of the sky follows a process until the alerts are detected and registered. First of all, the data gathered by the satellite is downlinked to the ground every day during an 8h window of visibility. It is then transferred to DPAC nodes in Germany and Spain where it is pre-processed during the Initial Data Treatment (IDT) process. As soon as this process finishes, the data is sent to Cambridge and analysed by the alerting system or also called "AlertPipe".

The anomaly detection system within the "AlertPipe" depends on the crossmatch information from IDT such that sources not matched with known objects are flagged as "new". All objects must pass a detection threshold and they are checked against possible asteroid positional coincidence. All surviving candidates for new transients pass to the next stage, the classification stage.

In the classification stage, a preliminary classification and filtering of detected candidates for alerts is carried out. Gaia monitors and detects variability with millimag precision down to V=15 mag and about 0.01 mag precision down to V = 20mag. As the satellite consists of two telescopes separated by 106.5 minutes, most of the Gaia observations may come in pairs. In addition to have a double check on the possible transient candidate there is a light curve classifier which can exploit the flux-gradient as an indicator of object type. Moreover, the BP/RP spectra as other features are available to aid in classification. It should be noted that the filtering and classification of the transient events are supplemented by contextual information obtained from available archival catalogues (WHB+12).

Name	Unique name of the alert.
TNS	IAU[1] Transient Name Server identifier of the Gaia alert.
Observed	Time of observation of the event that triggered the alert, in TCB [2].
RA (deg.)	Right ascension of the alerting source, in degrees, in the IRCS[3] frame.
Dec. (deg.)	Declination of the alerting source, in degrees, in the IRCS frame.
Magnitude	Magnitude, in Gaia's G band of the alerting source at the time of the alert.
Historic mag.	Mean, historic magnitude of the alerting source, in Gaia's G band before the alert.
Historic scatter	Observed variation of magnitude (standard deviation of measurements) of the alerting source, in Gaia's G band.
Class	Type of transient event.
Published	Time of publication, in UTC [4].
Comment	Comment on the type of alert detected.
RVS	Checked if the alert has any RVS (Radio Velocity Spectrometer) spectrum available.

Table 2.2: Columns of the Gaia Photometric Alerts (IoA15). [1]*International Astronomical Union.* [2]*Barycentric Coordinate Time.* [3]*International Celestial Reference System.* [4]*Coordinated Universal Time.*

On the Gaia Alerts-University of Cambridge website we can find all the gaia photometric alerts raised to date. Each alert provides coordinates of an event, a light curve, BP/RP spectrum and the results of the cross-match and classification analysis. The dataset is formed by the columns described in Table 2.2.

2.5 Previous studies

Although no studies on the classification with the Gaia Photometric Science Alerts have been done to date, we can find some articles about Gaia which inspire us to investigate the problem of classification.

The research undertaken by Nadejda Blagorodnova et al. (BKW+14) presents an algorithm to classify the nearby transient objects detected by the Gaia satellite. As inputs to perform the classification, the algorithm used the BP/RP photometers on board the satellite and according to the Gaia G-magnitude (From 15mag to 20 mag with a one step) the data was divided. As a classification method they took the Bayesian approach and each of the G values described was studied separately. One of the main results found in this article is that for magnitudes brighter than 19 in Gaia G-magnitude, around 75% of the transients will be robustly classified.

Machine learning techniques used for the classification are widely present in several studies related to Gaia DR2. One of them uses the Random Forest algorithm to classify whether the input object belongs to a galaxy or a star (ME+19). Another bases its study on the identification of Young Stellar Objects ("YSO") candidates in the Gaia DR2 x AllWISE catalogue with machine learning methods (BLW18). This article does not use a single machine learning method but experiments with various classifiers to evaluate their performance and select the best one for their needs. The methods used are the Support Vector Machines, the k-Nearest Neighbours, the Naive Bayes, Neural Networks and Random Forests which gives the best results. According to the authors, the probabilistic catalogue that they made, can also be useful to identify "YSOs" among future Gaia alerts.

In addition to the machine learning techniques used for classification processes, regressions have also been carried out. One of these studies focuses on the application of machine learning algorithms to obtain a regression of stellar effective temperatures in the second release of Gaia (Gaia DR2) (BLB+19). This study used two colours of the photometric bands of Gaia as inputs to calculate the effective temperature. The Support Vector Machines, Gaussian Processes and Random Forests were the algorithms under study. Although the three models had a similar performance, Random Forest algorithm was chosen because of its fast learning curve. Moreover, cross-validation was applied to test regression performance. The same author who wrote the last

two articles cited, "Yu Bai", wrote another article where he applies machine learning algorithms to carry out a regression of extinction in the second release of Gaia. As in the previous article, Random Forest was chosen to be the regression algorithm. (BLWW20).

On the one hand, according to the studies made on Gaia DR2, we observe that the most used machine learning algorithm is Random Forest. On the other hand, the article described by Nadejda Blagorodnova et al. (BKW+14) uses the values of the BP/RP photometers that we had planned to use in our study from the beginning as inputs. Hence, this article could help us to treat the inputs in the correct way. As we study the feasibility of classification, several machine learning models will be used to be able to contrast their behaviour on our data, including the Random Forest classifier.

2.6 Classification Methods

The dataset provided by the University of Cambridge should be treated before starting with the models. In this section, we will describe the set of supervised machine learning techniques that, according to previous studies, we decided to consider in this work.

2.6.1 Decision Tree

Before embarking on the Random Forest algorithm described in the previous studies, we decided to study the performance of the simple classifier that forms it, the Decision Tree.

Decision Trees are one of the most studied models, and not specifically because of their predictive capacity but because of their high explicative capacity (JG17). This data mining model focuses on subdividing the dataset into smaller regions until the entire input space is partitioned into disjointed regions that only contain elements of the same class. A Decision Tree is formed by the root node, internal nodes and leaf nodes.

- Root node. This is the upper element that determines in which direction a new piece of data goes.
- Internal nodes. They decide into which subregion each piece of information reaching that node moves.
- Leaf nodes. They represent the deepest regions and are labelled according to a class.

Furthermore, one of the most important hyperparameters of this model is the "partition criteria" that determines which leaf node is chosen to be partitioned. One of the ways of measuring it is by using "entropy", which evaluates the degree of disorder in the distribution of

elements. It is also possible to use the "Gini index" defined as the probability of a class being on a given leaf. Therefore, due to the simplicity of its construction and interpretation we will evaluate the performance of this classifier in future sections.

2.6.2 Support Vector Machines

Support Vector Machines are capable of solving linear and non-linear problems. This model is considered to be one of the most powerful algorithms in pattern recognition.

The Support Vector Machines are focused on the search for an optimal separator hyperplane which maximizes the margin of the training data (JG17). In other words, these models seek to maximize the margin between the points belonging to two groups which are to be classified.

This classifier deals only with linear problems but relies on kernel functions to transform non-linear problems in original space "X" into a linear problem in transformed space "F". The types of kernel functions available are as follows:

- **Linear kernel**.

$$k(u, v) = u^T \cdot v \tag{2.1}$$

  where u, v $\in$ X

- **Polinomial kernel**.

$$k(u, v) = (\gamma u^T \cdot v + b)^p \tag{2.2}$$

  where u, v $\in$ X, $\gamma > 0$, b ≥ 0 and p > 0.

- **Radial kernel**.

$$k(u, v) = e^{-\gamma \|u-v\|^2} \tag{2.3}$$

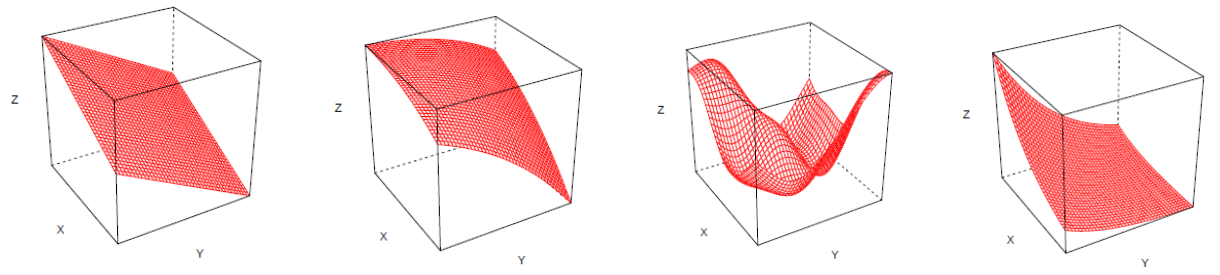
  where u, v $\in$ X and $\gamma > 0$.

- **Sigmoidal kernel**.

$$k(u, v) = tanh(\gamma u^T \cdot v + b) \tag{2.4}$$

  where u, v $\in$ X, $\gamma > 0$, b ≥ 0.

Although this type of classifier is strictly binary, there are two strategies that can help us deal with multiclass problems like the current one. One of them is the *one-versus-all* (OvA) strategy which trains as many models as the classes we have by evaluating each class individually against the rest of the elements. On the other hand, the *one-versus-one* (OvO) strategy will train a binary classifier for each different pair of classes, so, it will train a total of $N \cdot (N-1)/2$ classifiers. (Gé19)



(a) Linear Kernel (b) Polinomial kernel (c) Radial kernel (d) Sigmoidal kernel

Figure 2.6: Example of the different forms in the transformed space that we can manage with the kernel functions (JG17).

Owing to the efficiency which characterizes the algorithm, the wide use in classification tasks and the high performance it presents when dealing with many dimensions, we consider the Support Vector Machines a model to include in our study.

2.6.3 Random Forest

Random Forest is a type of ensemble method that combines multiple machine learning models to create more powerful models. This classifier is mainly characterized by the idea of working with a set of Decision Trees that have been created through a random process.

The main idea behind Random Forest is that each tree might do a relatively good job of predicting, but will likely overfit on part of the data. If we build several trees we will be able to calculate the average of these trees and reduce the amount of overfitting (ACM16). This process will allow us to measure the relative importance of each variable (JG17) and obtain a more general vision of what a single tree brings to the model.

It should be noted that, as mentioned in section 2.5, G. Marton et al. (ME+19) used Random Forest algorithm as a classifier and the article wrote by Yu Bai et al. (BLW18) on which a comparison of different models was made, the one with the best performance was also Random Forest. Hence, we will see in the following chapters if this algorithm is able to provide us with the best performance as it did in these articles.

2.6.4 Gradient Boosting Classifier

The Gradient Boosting Classifier is another ensemble method which combines multiple Decision Trees with the aim of creating a more powerful model. In contrast to the Random Forest approach, gradient boosting works by building trees in a serial manner, where each tree tries to correct the mistakes of the previous ones.

By default, there is no randomness in Gradient Boosting Machines. Instead, a strong pre-pruning is used. The main idea behind the Gradient Boosting is the combination of shallow trees which makes the model smaller in terms of memory and faster predictions. Moreover, each tree will only provide good predictions on part of the data, so more and more trees are added to iteratively improve the performance.

One of the most important parameters to take into account in this model is the "learning_rate" which controls how strongly each tree tries to correct the mistakes of the previous ones. Furthermore, another important parameter allows you to tune the number of trees consider under study. (ACM16)

On the whole, the Gradient Boosting Machines are slightly more sensitive to parameter settings than Random Forest, but can provide a better accuracy if the parameter are set correctly.

2.6.5 Artificial Neural Network

Artificial Neural Networks are a machine learning model inspired by the network of biological neurons found in our brain. These models are characterized by their versatility, scalability and their ability to deal with a large number of highly complex problems.

The construction of an Artificial Neural Network is carried out by a combination of neurons grouped in layers. The first input layer will take the raw data and the last layer will return the class on which it is classified. In addition, the neural network will be regulated by the following hyperparameters:

- **Epochs**. Number of times that the entire dataset is passed through the network.
- **Batch Size**. Number of samples that will be passed through the network.
- **Loss Function**. This function is a mathematical way of measuring how wrong your predictions are. Furthermore, loss function is used in the process of optimization to find the best model weights for your data (Alg18a). Some of the most popular loss functions are: Mean Squared Error (MSE), Likelihood Loss or Cross Entropy Loss.
- **Optimizer**. The Optimizer acts in conjunction with the Loss function by updating the model weights according to the response of the Loss function (Alg18b). Some Optimizers are: "Adam", "Adamax", "Nadam" and "RMSprop".
- **Activation function**. This is added into an Artificial Neural Network in order to help the network learn complex patterns in the data. This function takes in the output signal

from the previous cell and converts it into a form that can be taken as input to the next cell (Jai18). Some activations functions are: “tanh”, “softsign”, “selu”, “elu” and “ relu”.

- **Learning rate**. The learning rate is a hyperparameter which controls how much to change the model in response to the estimated error each time the model weights are updated. It is difficult to choose an appropriate learning rate value: a very small value leads to a long and tedious training process and a high value to a fast and unstable learning (Bro19).

The capacity to learn and deal with high dimensional problems makes Artificial Neural Networks a model which we will consider in the following sections.

Chapter 3

Design and development aspects

The design and development of this project was carried out using Python programming language and the Jupyter Notebook application. All the documents as well as the codes and datasets can be found in Appendix A: Documents.

3.1 Obtaining data from photometric spectra

The Gaia Photometric Science Alerts website gives us a complete table (*alerts22102020_920.csv*) in CSV format where all the alerts raised to date can be seen. However, this table does not contain the values of the photometric spectra. Hence, we carried out a web scraping process to obtain the photometric spectra that would be the basis for the construction of the future models.

Before implementing web scraping techniques, a cleaning and pre-processing of the data provided by the website was performed in order to optimize this process and work only with the relevant data.

3.1.1 Pre-processing

We worked with the data downloaded on 22 October 2020 at 9:20 a.m. The downloaded data has the same form as described in table 2.2. Using Python programming language and the Jupyter Notebook application, the filtering, preparation and data cleaning processes were carried out.

Data preparation and cleaning

1. Unification of the names of each feature by removing unnecessary spaces and characters.
2. Unification of the "Comment" column items which have equal meaning.

3. Creation of a new column called "Feature" where the class feature of the alert appears as long as it exists. When no class is available, this gap is filled with the "Comment" feature.

4. Creation of a new column called "Class_comment" which complements the "Feature" column. In this column, the name "Class" appears when the "Feature" comes from the class column and the name "Comment" when it comes from the comment column.

Data filtering

The initial database contained a total of 14060 elements but not all classes had a relevant representation. Therefore, the data was filtered to determine the frequency of appearance of each of the "features" of the dataset. All those classes or comments that had less than 50 elements were discarded from the study (this dataset was called "less50.csv"). The table 3.1 shows the classes left after filtering.

Feature	Frequency
SN Ia	1368
QSO	634
CV	459
SN II	419
Blue Hostless Transient	280
Apparently Blue Hostless Transient	183
YSO	135
Apparently Hostless Transient	107
SN IIn	87
SN IIP	83
AGN	78
BL Lac	51
TOTAL	**3884**

Table 3.1: Frequency of appearance of each of the "features".

As we can see, a total of 3884 entries were obtained from which the photometric spectra recorded in each of them have to be extracted. The extraction of the spectra will be explained in the following section using web scraping techniques.

3.1.2 Web Scraping

Before starting with web scraping, we will explain in more detail what exactly we were looking for. As an introduction to this section, it is necessary to clearly distinguish the differences between alert and detection:

- **Alert**. When the satellite detects a significant change in magnitude or a new object (black circle on figure 3.1). When this happens, an object entry is created in the Gaia Alerts catalogue (e.g. "Gaia20evz"). That is, after the cleaning and filtering we obtained a dataset with 3884 rows ("less50.csv") where each row is an entry like "Gaia20evz".
- **Detections**. Once the alert is registered, previous and future detections of the object obtained by the satellite will automatically be added to the registered entry (blue circles on figure 3.1). The number of detections may vary when moving to a different entry.

According to Simon Hodgkin, member of the Cambridge Astronomical Survey Unit, each entry of the Gaia alert catalogue, such as "Gaia20evz", is only published once and in it we will find multiple detections and only one alert.

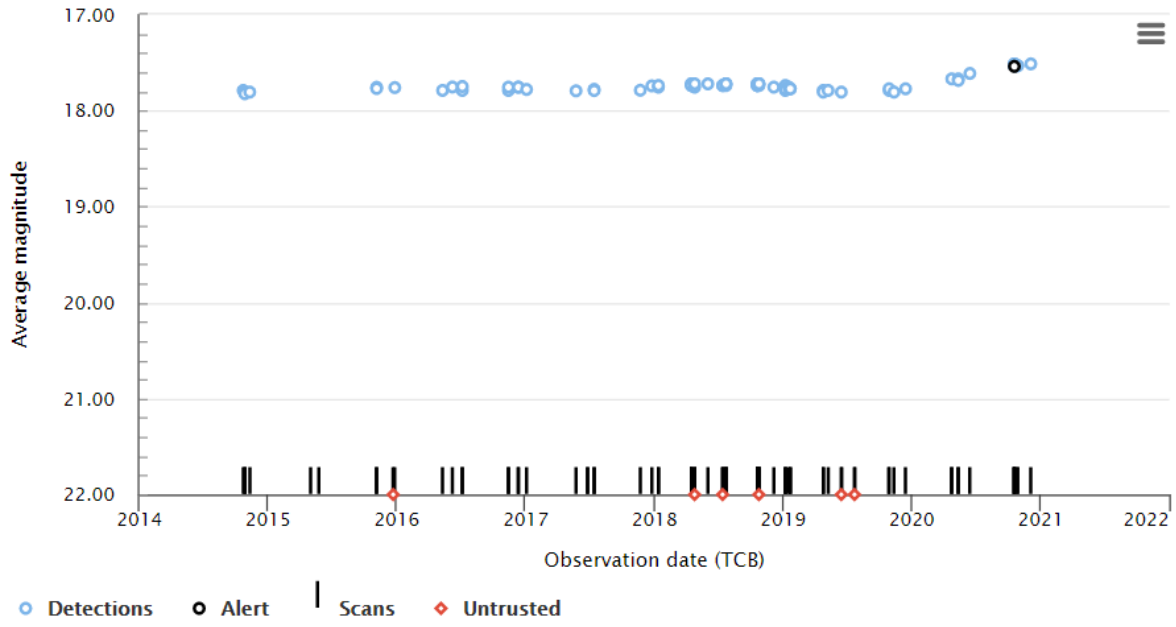


Figure 3.1: Average magnitude vs. Observation date collected by "Gaia20evz". The blue dots represent the detections and the black one the alert collected by the entry "Gaia20evz".

Most importantly, each of these detections and alerts which constitute an entry in the Gaia Alerts catalogue contains a photometric spectra (Figure 3.2) that provides the necessary values for the classification methods. While performing the web scraping, not only did we extract the photometric spectra related to a certain alert detection (black circle), but also the whole set of detections of the given alert (blue circles), in order to get more information about the type of alert we were working with.

Using web scraping techniques, we obtained the elements that describe the RP and BP spectra of a given alert or detection. Each of the RP and BP spectra are formed of 60 values represented as graph 3.2 . These graphs can be obtained for each of the alerts and detections of every entry of the Gaia Alerts catalogue.

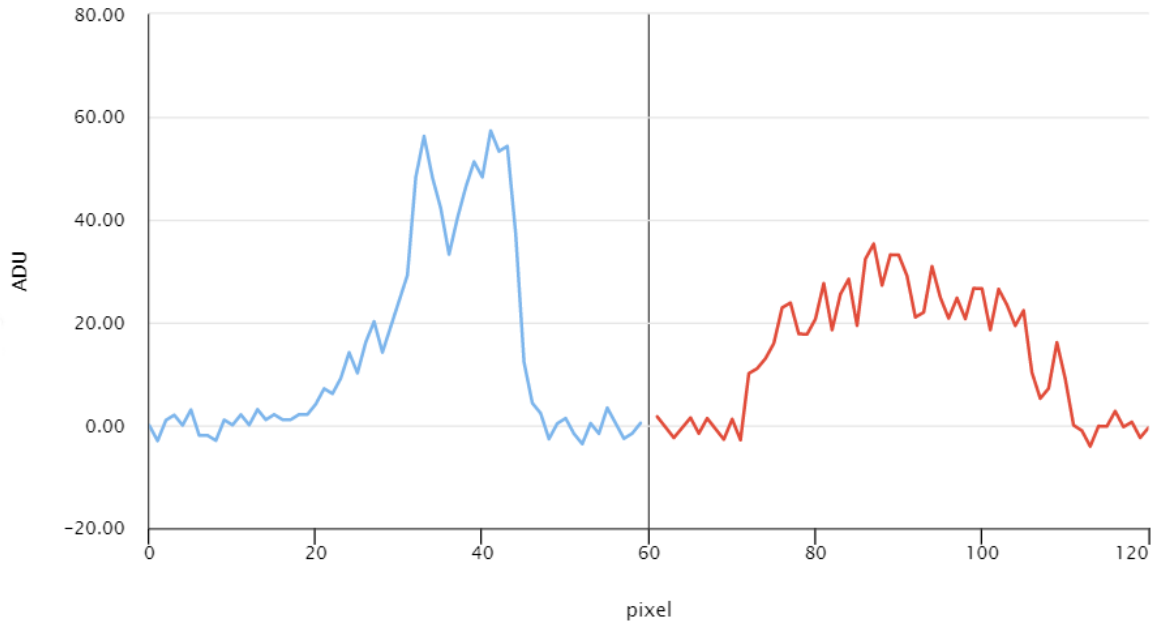


Figure 3.2: BP and RP graphics. Analog Digital Units (ADU) vs. Pixel.

To implement web scraping techniques, we worked with the Python library *"BeautifulSoup"*. Thanks to this library we were able to work on the source code of the website where the values of the spectra contained in an alert or detection were found. We carried out a sweep of all the entries present in the dataset obtained after filtering. As we said before, each of these entries presents a different number of spectra. Hence, we first detected the number of spectra that each entry contained and by means of a loop we extracted the values of the BP and RP spectra.

For each entry, we generated as many rows as the number of spectra it contained. Thus, we obtained a dataset with all the spectra (one per row) to be considered in the study. This dataset was called "Spectra dataset", it was saved as .csv document (Appendix A: Documents) and it has the following attributes:

id	Unique name of the alert.
order	Order of the spectra on a given alert.
bp	Values as a list of the Blue Photometer.
rp	Values as a list of the Red Photometer.
a_d	If the spectra corresponds to a detection it have the letter "D". If the spectra corresponds to an alert it have the letter "A".
feature	It show us the class or comment of a spectra.

Table 3.2: Columns of the "Spectra dataset".

It should be noted that of the 3884 entries that were submitted to the web scraping process to extract their spectra, 11 had no spectral values, therefore they were removed of the "Spectra dataset" (Appendix B: Information about datasets).

The extraction process was very slow and expensive given the large amount of data we were working with. The duration depends on the characteristics of the computer used. In our case, after three hours of execution we obtained the 101406 spectra that were involved in the creation of the models.

3.2 Pre-processing of the "Spectra dataset"

Once we obtained the spectra, we processed this dataset ("Spectra dataset") to be used as input for the models.

3.2.1 Data preparation

When the dataset was reloaded, the columns "bp" and "rp" appeared as strings instead of a list of numbers as they should. Using a loop through each element of the dataset we converted these string elements into a list of numbers.

3.2.2 Data filtering

As we were dealing with a large amount of data, it was important to know the number of spectra available for each class in order to make conclusions based on it.

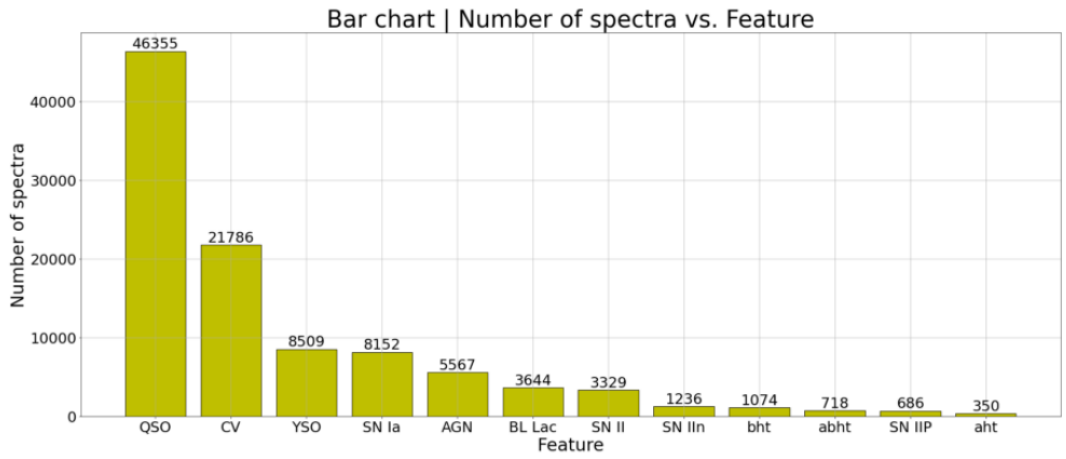


Figure 3.3: Number of spectra vs. feature.

The graph 3.3 shows how the number of spectra obtained for the different classes is very unequal. Trying to maximize the number of classes under study and avoiding those with a lower representation, a limit was established related to the number of spectra. This limit was set at 3000 spectra. All those classes that had more than 3000 spectra were included in the study and those that had less than 3000 spectra were grouped together to form a new class that was called "Other". Hence, the final dataset had 8 different classes: "QSO", "CV", "YSO", "SN Ia", "AGN", "BL Lac", "SN II" and "Other". The description of the classes, the frequency and the number of spectra of the final elements are compiled in Appendix B: Information about datasets.

3.2.3 Data transformation

With an eye towards applying machine learning models, we saw how the distribution of the "bp" and "rp" columns in the "Spectra dataset" was not appropriate. With lists of numbers we were not able to work with these models. Hence, each item on the list was transformed into a new column in the database. That is, if these columns (both "bp" and "rp") contained lists of 60 elements, then 60 columns corresponding to the "bp" list and another 60 corresponding to the "rp" list were added to the dataset. The resulting dataset was called "Spectra Columns dataset" and has the following shape:

id	**order**	**bp**	**rp**	**a_d**	**feature**	**bp0**	...	**bp59**	**rp0**	...	**rp59**
"Gaia20ewl"	0.0	[0.65, ...]	[2.47, ...]	A	SN Ia	0.65	...	0.75	2.47	...	1.33

Table 3.3: Structure of the "Spectra Columns dataset".

As we can see in table 3.3 we increased the "Spectra dataset" by 120 columns.

3.2.4 Standardization

The standardization process allows us to obtain data with average zero and unit variance. This process was carried out on the "bp" and "rp" columns treated separately. For a given feature x the standard score is given by

$$Z = \frac{x - \mu}{\sigma} \tag{3.1}$$

where μ is the average and σ the standard deviation. Through standardization our data is much less affected by outliers and allows us to obtain the best performance from the machine learning models. (Gé19)

3.3 Machine Learning models

In this section we will focus on explaining the different machine learning models implemented with the main objective of seeking maximum accuracy in spectra classification.

For the implementation of all models we will use the "Spectra Columns dataset" and we will divide the section into four parts: Subset of 4000 spectra, Complete Dataset, Artificial Neural Network and Project workflow.

3.3.1 Subset of 4000 spectra

In this first stage, we looked for the model's best values of hyperparameters within a reasonable time. The "Spectra Columns dataset" was too large to perform a deep search for the best values of hyperparameters because the algorithms execution times would have been disproportionate. Therefore, we took a subset of 4000 random and class-balanced spectra. In this way, we had 500 different elements for each of the 8 classes that constitute the "Spectra Columns dataset".

This subset is composed of 121 columns: 120 correspond to the values of the "bp" and "rp" spectra, and the remainder correspond to the "feature" column with the class of the spectra. A new division of the subset dedicated 80 % to the training of the models and 20 % to the test. Furthermore, a 5-fold cross validation was applied to the models.

The search for the best values of the hyperparameters was carried out using the command *GridSearchCV()* (Provided by the *scikit-learn* Python library) with a cross validation of four folds. The supervised learning methods used in this study were: Decision Tree, Support Vector Machines, Random Forest and Gradient Boosting Classifier.

3.3.1.1 Decision Tree

By means of the *GridSearchCV()* the Decision Tree (DT) carried out a search to find the best values for the hyperparameters: "max_depth", "criterion", "min_split" and "max_features". In table 3.4 we can see a description of these hyperparameters and the values under consideration.

Hyperparameter	Description	Values under consideration
max_depth	The maximum depth of the tree. If it is "None", nodes are expanded until all leaves are pure or until all leaves contain less than "min_samples_split" samples.	"None" and from 1 to 10 (Step 1)
criterion	The function to measure the quality of a split.	"Entropy" or "Gini"
min_samples_split	The minimum number of samples required to be at a leaf node.	From 5 to 256 (Step 25)
max_features	The number of features to consider when looking for the best split.	"sqrt"→$\sqrt{Number\ of\ features}$ "log2"→$\log_2(Number\ of\ features)$

Table 3.4: Hyperparameters of the Decision Tree.

3.3.1.2 Support Vector Machine

Likewise, the Support Vector Machine (SVM) also carried out a search to find the best values for the hyperparameters: "kernel", "C" and "gamma". In table 3.5 a description of these hyperparameters and the values under consideration is shown.

Hyperparameter	Description	Values under consideration
kernel	Specifies the kernel type to be used in the algorithm.	"linear", "poly", "rbf", "sigmoid"
C	Regularization parameter.	0.01, 0.1, 1, 10, 50, 100
gamma	Kernel coefficient for 'rbf', 'poly' and 'sigmoid'.	0.001, 0.01, 0.1, 1, 10

Table 3.5: Hyperparameters of the Support Vector Machine.

Furthermore, in order to deal with the multiclass problem on this classifier, the *one-versus-all* (OvA) strategy was used.

3.3.1.3 Random Forest

On Random Forest (RF), the search for the best values was carried out on the following hyperparameters: "max_depth", "criterion", "n_estimators" and "max_features". In table 3.6 we can see a description of these hyperparameters and the values under consideration.

Hyperparameter	Description	Values under consideration
max_depth	The maximum depth of the tree. If it is "None", nodes are expanded until all leaves are pure or until all leaves contain less than "min_samples_split" samples.	"None" and from 1 to 10 (Step 1)
n_estimators	The number of trees in the forest.	From 10 to 400 (Step 50)
criterion	The function to measure the quality of a split.	"Entropy" or "Gini"
max_features	The number of features to consider when looking for the best split.	"sqrt"→$\sqrt{Number\ of\ features}$ "log2"→$\log_2(Number\ of\ features)$

Table 3.6: Hyperparameters of the Random Forest.

3.3.1.4 Gradient Boosting Classifier

The Gradient Boosting Classifier (GBC) carried out a search to find the best values for the hyperparameters: "max_depth", "n_estimators" and "learning_rate". In table 3.7 a description of these hyperparameters and the values under consideration is shown.

Hyperparameter	Description	Values under consideration
max_depth	The maximum depth of the tree. If it is "None", nodes are expanded until all leaves are pure or until all leaves contain less than "min_samples_split" samples.	"None" and from 1 to 10 (Step 1)
n_estimators	The number of boosting stages to perform.	10, 25, 50, 100, 200
learning_rate	Learning rate of the classifier.	0.01, 0.1, 1

Table 3.7: Hyperparameters of the Gradient Boosting Classifier.

3.3.2 Complete Dataset

In this section, the subset of 4000 random spectra provided us with the best hyperparameters for each of the classifiers under study and the complete dataset ("Spectra Columns dataset") with its 101406 spectra was used. In the same way as in the previous section 3.3.1, we selected 121 columns of which 120 corresponded to the photometric spectra and the remainder to the class of the spectra. We took 80% of the data as a training set that was used to fit the different machine learning models and 20% as a test set. Moreover, we also applied a stratified 4-folds cross-validator where the folds are made by preserving the percentage of samples for each class.

On the one hand, taking into account the imbalance of the classes we added the hyperparameter *class_weight = " balanced"* on the DT, SVM and RF models. The *"balanced"* mode uses

the "feature" values to automatically adjust weights inversely proportional to the frequency of the classes in the input data. Furthermore, the Random Forest Classifier was also treated with the *"balanced_subsample"* mode which is the same as *"balanced"* except that weights are computed based on the bootstrap sample for every tree grown.

On the other hand, the Gradient Boosting Classifier deals with class imbalance by constructing successive training sets based on incorrectly classified examples, so this model doesn't have any hyperparameter related to the balance of the classes (Wan18). Although it usually outperforms Random Forest on imbalanced dataset, we will see if the same thing happens with our data.

3.3.3 Artificial Neural Network

We carried out the construction of an Artificial Neural Network (ANN) with the aim of improving the performance obtained by the previous models. To build the model, we undertook a search for the best architecture (number of neurons and layers to be used) and certain hyperparameters such as the optimizer, the batch size, the epochs, the activation function and the learning rate.

First, as in section 3.3.1, we took a subset of 4000 random elements where each of the (eight) classes had the same representation, i.e. 500 elements. After this, we took the 120 columns that gave shape to the spectra and the column relative to the class. In order to work with the neural network, each of the classes were transformed by means of One-Hot-Encoding and represented by an eight element vector containing seven zeros and a one in the position corresponding to its class (Appendix B: Information about datasets). For example:

$$\text{“}QSO\text{”} \longrightarrow [0, 0, 0, 0, 1, 0, 0, 0] \tag{3.2}$$

As in the past models, we took 80% of the data as a training set and 20% of the data as a test set. It should be noted that from the 80% of the data in the training subset, which represented 3200 spectra, we took 500 spectra as a validation set. This left 2700 spectra for the training set (67.5%), 500 spectra for the validation set (12.5%) and 800 spectra for the test set (20%).

To find the best neural network we proposed 8 different architectures where the number of layers and neurons in the network varies. These can be seen in table 3.8:

Hidden Layers	Architecture
3	[120, 92, 64, 36, 8]
3	[120, 90, 60, 20, 8]
3	[120, 150, 100, 50, 8]
3	[120, 140, 70, 30, 8]
4	[120, 96, 74, 52, 30, 8]
4	[120, 100, 80, 40, 20, 8]
4	[120, 150, 100, 50, 20, 8]
4	[120, 140, 100, 75, 30, 8]

Table 3.8: Architectures ANN.

A painstaking search for the best values of the hyperparameters was carried out in order to achieve an optimized solution. In the following list, we compile the hyperparameters under study: optimizer, batch size, epochs, activation and the learning rate.

Hyperparemeter	Values under consideration
Epochs	50, 100
Batch Size	12, 32, 64
Optimizer	“Adam”, “Adamax”, “Nadam”, “RMSprop”
Activation function	“tanh”, “softsign”, “selu”, “elu”, “ relu”
Learning Rate	0.01, 0.1, 1

Table 3.9: Hyperparameters ANN.

By means of a loop we went over the different architectures and hyperparameters described. Each of the models was trained and the accuracy of each one was computed. According to the results obtained from the best models, new searches with new architectures or hyperparameter values would be carried out.

Once the best models were obtained, we evaluated the Artificial Neural Network on the complete dataset. We took 90% of the 101406 spectra as the training set and 10% as the test set. In addition, as we had done for the subset, we used 10,000 spectra from the training set as the validation set. So finally the data was distributed as follows: 80.14% as a training set, 9.86% as a validation set and 10% as a test set. If the results with the complete dataset were not suitable, new searches with new architectures or hyperparameter values would be carried out.

In order to solve the problem of unbalanced classes, we used the python library *sklearn.utils* which, by means of the command *class_weight*, allowed us to calculate the weight of each one of the classes under study.

3.4 Project workflow

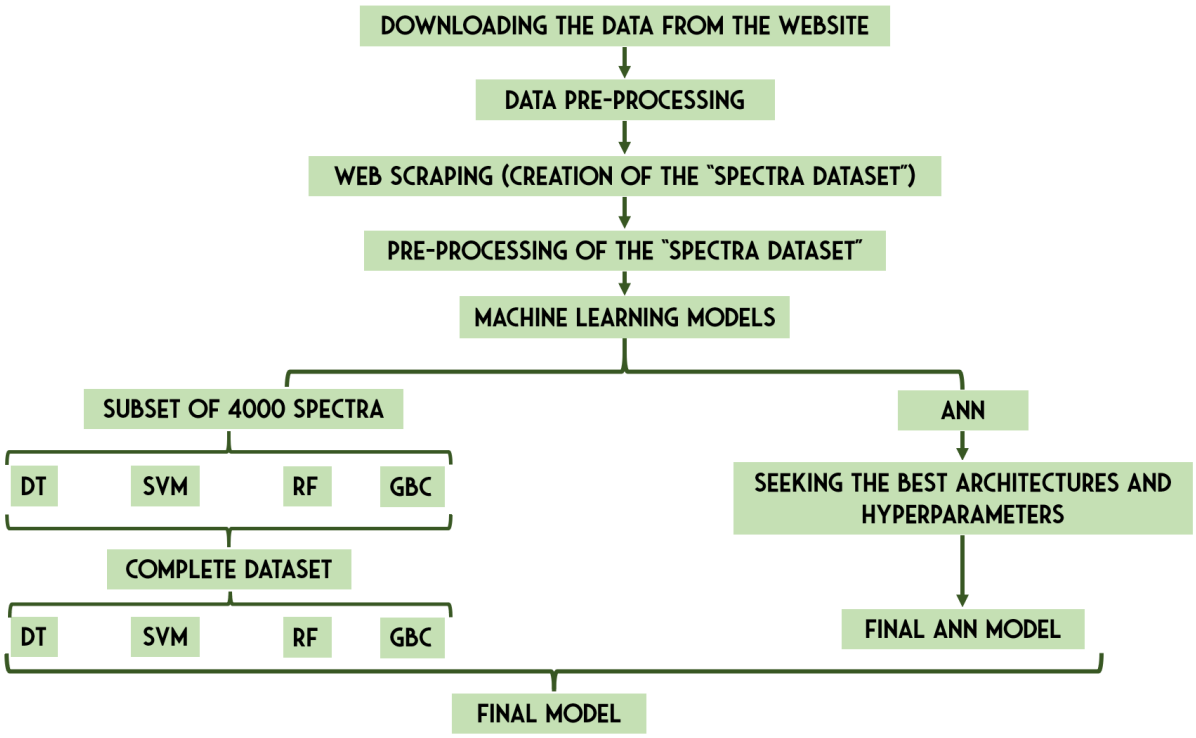


Figure 3.4: Workflow for the classification model of the Gaia Photometric Alerts.

Chapter 4

Results

In this chapter we present the different results obtained and the performances achieved by the models, establishing a rigorous comparison between the different models implemented. Although we use some metrics (Precision, Recall, F1-Score and Accuracy) to evaluate the performance of the models (Appendix C: Metrics), the metric that we have chosen in order to select the best model is "accuracy".

4.1 Results Subset of 4000 spectra

The best values of the hyperparameters obtained on each classifier, the confusion matrix and the metrics that will allow us to evaluate the performance of the model, will be exposed.

4.1.1 Decision Tree

The best values for the hyperparameters of the Decision Tree Classifier (DT) obtained by the command *GridSearchCV()* are the following:

max_depth	**criterion**	**min_samples_split**	**max_features**
None	"gini"	55	"sqrt"

Table 4.1: Best values for the hyperparameters of the Decision Tree Classifier.

Evaluating this model on the subset of 4000 spectra, a mean cross validation over the 5 sets of 0.335 is obtained. In addition, the accuracy of the model is 32.8%. The following confusion matrix 4.1 allows us to see the relationship between predicted and real values.

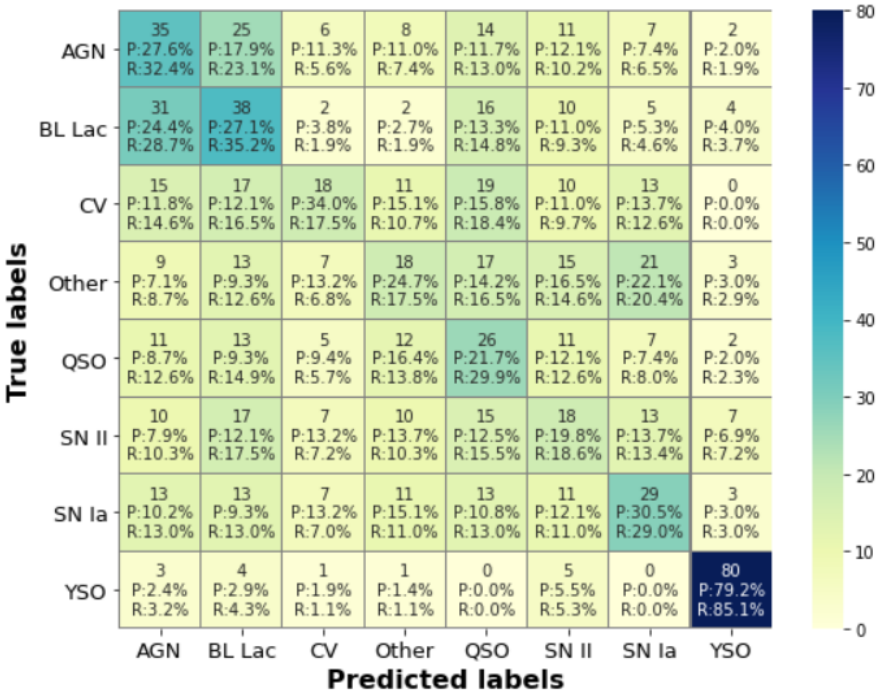


Figure 4.1: Decision Tree. Confusion Matrix. Subset of 4000 spectra. The X-axis represents the class type predicted by the classifier and the Y-axis represents the true type. "P" represents the precision and "R" the recall of each element.

It should be noted that on the test set, each class is represented by approximately 100 spectra. As we can see, the class that best classifies the information is "YSO", and of the 94 predictions made about this class 80 are correct, which is a recall of 85.1%. Furthermore, a high precision of 79.2% is obtained for "YSO". In spite of the results obtained for this element, the remainder of classes do not achieve good performances. Other metrics that can help us to understand the model better appear in the following table 4.2.

Decision Tree Classifier Results				
5-fold (Accuracy)	**Precision**	**Recall**	**F1 score**	**Accuracy**
0.335	0.331	0.331	0.325	0.328

Table 4.2: Decision Tree Classifier Results. Subset of 4000 spectra.

This classifier achieves discouraging results. As we can expect from the confusion matrix, the precision and the recall on the whole model are really low. In the same line we have the F1 score and accuracy metrics.

4.1.2 Support Vector Machine

The best values for the hyperparameters of the Support Vector Machine (SVM) are the following:

kernel	C	gamma
"rbf" (Radial)	100	0.1

Table 4.3: Best values for the hyperparameters of the Support Vector Machine.

Evaluating this model on the subset of 4000 spectra, a mean cross validation over the 5 sets of 0.403 is obtained and an accuracy of 40.0%. The following confusion matrix 4.2 allows us to see the relationship between predicted and real values.

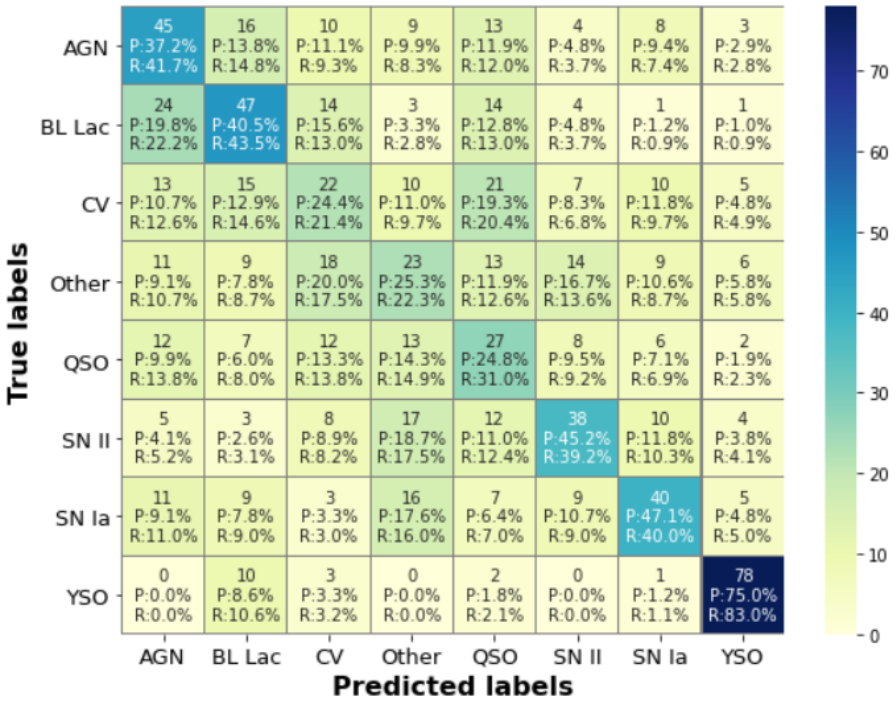


Figure 4.2: Support Vector Machine. Confusion Matrix. Subset of 4000 spectra. The X-axis represents the class type predicted by the classifier and the Y-axis represents the true type. "P" represents the precision and "R" the recall of each element.

The Support Vector Machine also manages to clearly classify the "YSO" class (recall 83%, precision 75%). Although these values are slightly lower than in the Decision Tree, this model

increases the performance of many classes. The recall of all classes except the "YSO" class and the precision of "SN Ia", "SN II", "BL Lac" and "AGN" classes increases with respect to Decision Trees. Despite this increase, the percentages remain low. Other metrics that can help us to understand the model better appear in the following table 4.4.

Support Vector Machine Results				
5-fold (Accuracy)	**Precision**	**Recall**	**F1 score**	**Accuracy**
0.403	0.399	0.403	0.399	0.400

Table 4.4: Support Vector Machine Results. Subset of 4000 spectra.

The model manages to slightly improve all the metrics of the Decision Tree Classifier. However, the performance obtained by the Support Vector Machine is still deficient. We will see if over the complete dataset the performance rises.

4.1.3 Random Forest

The best values for the hyperparameters of the Random Forest Classifier (RF) are the following:

max_depth	**criterion**	**nestimators**	**max_features**
None	"gini"	400	"sqrt"

Table 4.5: Best values for the hyperparameters of the Random Forest Classifier.

Random Forest is composed of a set of Decision Trees. Hence, if we compare the values of the hyperparameters obtained for the Decision Tree 4.1 with those obtained for the Random Forest 4.5, we can see that the hyperparameters which coincide in both tables have the same values.

Evaluating this model on the subset of 4000 spectra, a mean cross validation over the 5 sets of 0.517 is obtained and an accuracy of 53.5%. The following confusion matrix 4.3 allows us to see the relationship between predicted and real values.

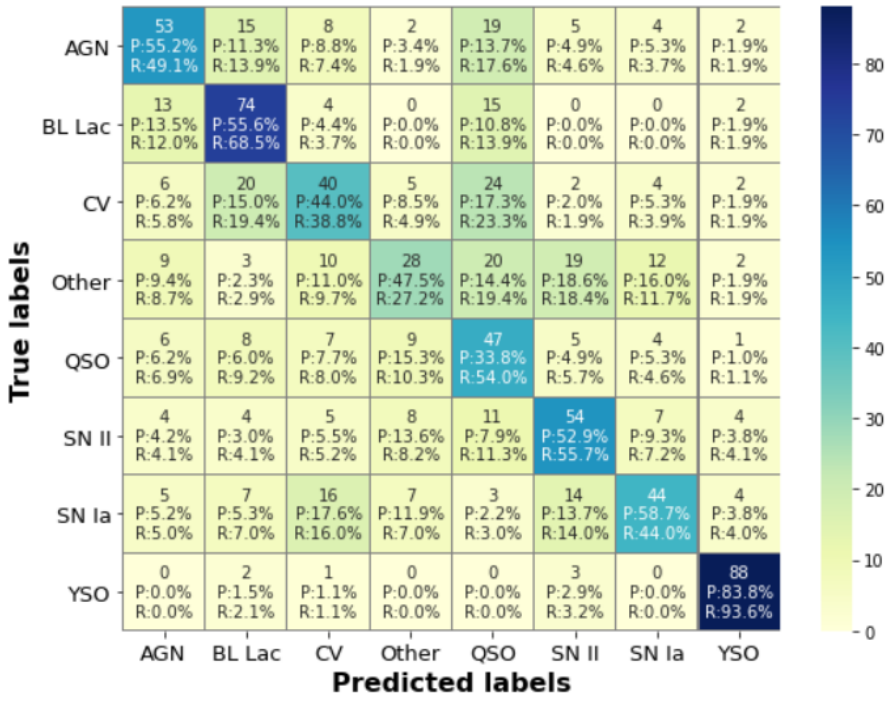


Figure 4.3: Random Forest. Confusion Matrix. Subset of 4000 spectra. The X-axis represents the class type predicted by the classifier and the Y-axis represents the true type. "P" represents the precision and "R" the recall of each element.

In this classifier, we can appreciate a remarkable improvement over the confusion matrix. The classes with the highest predictive capacity (recall) are "YSO" with 93.6% and "BL Lac" with 68.5%. Moreover, all the classes manage to improve their percentages in the precision and recall metrics. Other metrics that can help us to understand the model better appear in table 4.6.

Random Forest Results				
5-fold (Accuracy)	**Precision**	**Recall**	**F1 score**	**Accuracy**
0.517	0.539	0.539	0.530	0.535

Table 4.6: Random Forest Results. Subset of 4000 spectra.

As expected, this set of Decision Trees improves the results obtained by a single tree and also by the Support Vector Machines. In spite of the results increase in all metrics, the performance of the classifier remains low.

4.1.4 Gradient Boosting Classifier

The best values for the hyperparameters of the Gradient Boosting Classifier (GBC) are the following:

max_depth	n_estimators	learning_rate
7	200	0.1

Table 4.7: Best values for the hyperparameters of the Gradient Boosting Classifier.

Evaluating this model on the subset of 4000 spectra, a mean cross validation over the 5 sets of 0.472 is obtained and an accuracy of 48.9%. The following confusion matrix 4.4 allows us to see the relationship between predicted and real values.

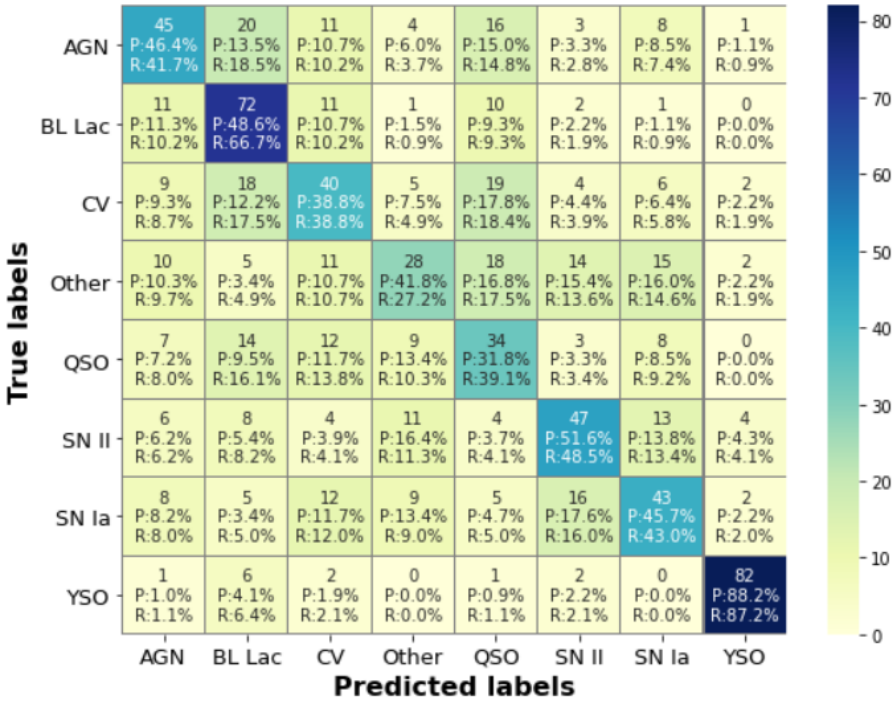


Figure 4.4: Gradient Boosting Classifier. Confusion Matrix. Subset of 4000 spectra. The X-axis represents the class type predicted by the classifier and the Y-axis represents the true type. "P" represents the precision and "R" the recall of each element.

Although the confusion matrix of this classifier is quite similar to the confusion matrix of the Random Forest, it has slightly worse results. The best percentages for the recall obtained in

this confusion matrix corresponds to the "YSO" and "BL Lac" classes. Furthermore, the best precision also corresponds to the "YSO" class. Other metrics that can help us to understand the model better appear in table 4.8.

Gradient Boosting Classifier Results				
5-fold (Accuracy)	**Precision**	**Recall**	**F1 score**	**Accuracy**
0.472	0.491	0.490	0.486	0.489

Table 4.8: Gradient Boosting Classifier Results. Subset of 4000 spectra

The Gradient Boosting Classifier is unable to establish clear patterns to identify the type of spectrum that it is working with. The results obtained are lower than the Random Forest, although they exceed the Decision Tree and the Support Vector Machine.

4.2 Complete Dataset

Taking the best values of the hyperparameters obtained over the Subset of 4000 random spectra, we will implement the classifiers again but using the complete dataset. We hereby expose the metrics that will allow us to evaluate the performance of the classifiers.

4.2.1 Decision Tree

After evaluating the Decision Tree (DT) on the complete dataset, an accuracy of 35.3% and a mean cross validation over 4 sets of 0.350 are obtained by the classifier. The following confusion matrix 4.5 allows us to see the relationship between predicted and real values.

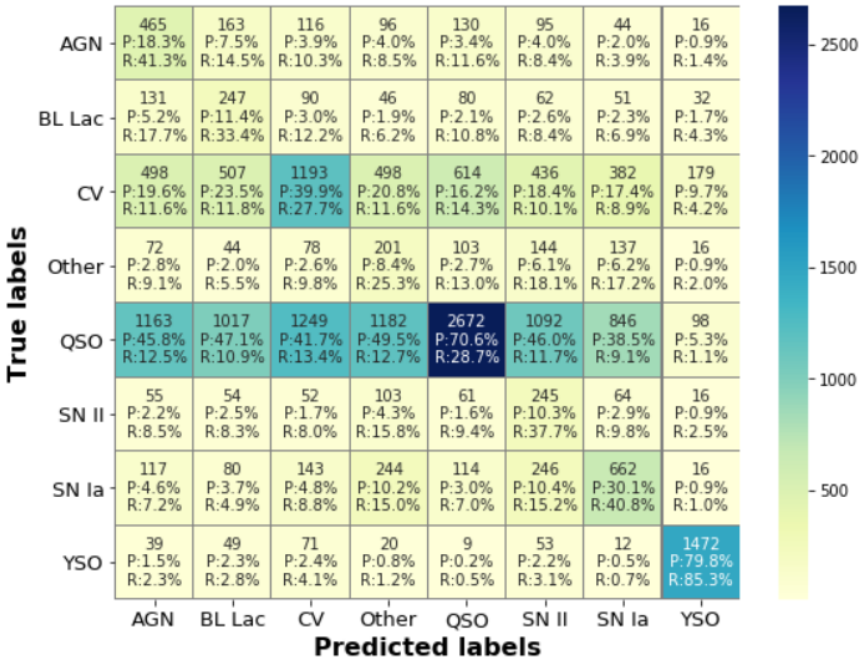


Figure 4.5: Decision Tree. Confusion Matrix. Complete Dataset. The X-axis represents the class type predicted by the classifier and the Y-axis represents the true type. "P" represents the precision and "R" the recall of each element.

In spite of the class imbalance is rectified, on this confusion matrix we do not have the same number of classes on the test set and this structure will be repeated on the confusion matrices obtained in the complete dataset. Hence, we must focus on the precision and accuracy of the matrix to observe the behaviour of the model.

As we observe on the matrix, a large amount of the predictions adopt the value of the class that has more representation, the "QSO". The class detected with greater efficiency is "YSO" which has values of precision and recall slightly higher than those obtained in the subset. Other metrics that can help us to understand the model better appear in table 4.9.

Decision Tree Results				
Stratified 4-fold (Accuracy)	**Precision**	**Recall**	**F1 score**	**Accuracy**
0.350	0.336	0.400	0.327	0.353

Table 4.9: Decision Tree Results. Complete Dataset.

The results obtained by the complete dataset are deficient for the Decision Tree. The performance slightly increases with respect to the treatment of this same classifier on the subset of 4000 spectra but does not exceed the other models evaluated on that subset.

4.2.2 Support Vector Machine

The Support Vector Machine (SVM) evaluated on the complete dataset, provides an accuracy of 50.6%. In addition, a mean cross validation over 4 sets of 0.495 is obtained by the classifier. The following confusion matrix 4.6 allows us to see the relationship between predicted and real values.

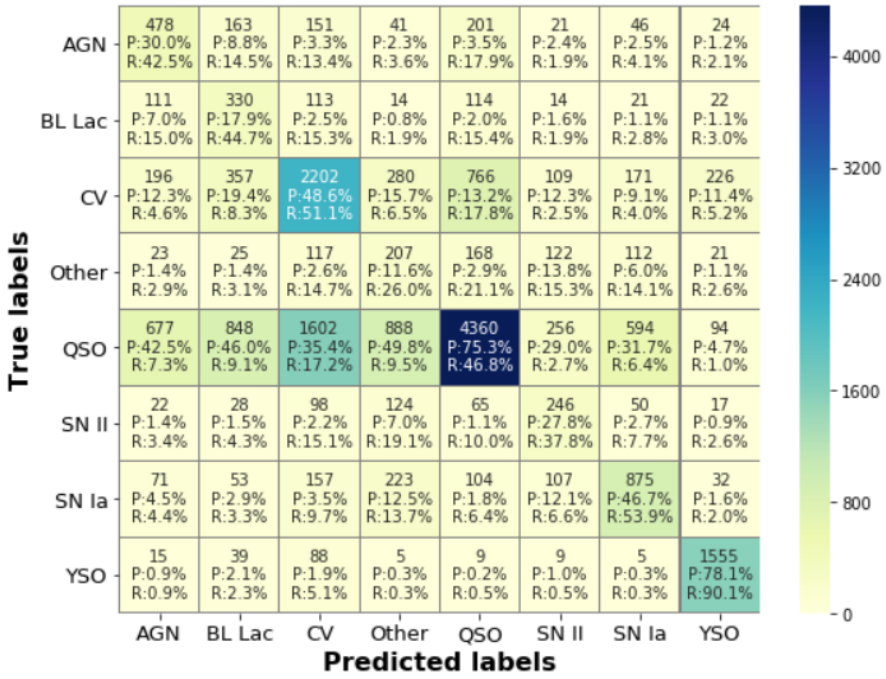


Figure 4.6: Support Vector Machine. Confusion Matrix. Complete Dataset. The X-axis represents the class type predicted by the classifier and the Y-axis represents the true type. "P" represents the precision and "R" the recall of each element.

The incorrect predictions about "QSOs" made in the Decision Tree Classifier seem to have been corrected in the Support Vector Machine. The "YSO" class, with the highest precision and recall, continues to shows great detectability. Additionally, we can also appreciate acceptable performance for the "SN Ia", "CV" and "QSO" classes. The latter, given its 75.3% precision, stands out for its ability to be 'QSO' when it is detected as such. Other metrics can be found in table 4.10 below.

Support Vector Machine Results				
Stratified 4-fold (Accuracy)	**Precision**	**Recall**	**F1 score**	**Accuracy**
0.495	0.420	0.491	0.438	0.506

Table 4.10: Support Vector Machine Results. Complete Dataset.

On the one hand, the results obtained by the complete dataset improve with respect to the subset of data. On the other hand, the accuracy is similar to the Random Forest Classifier obtained on the subset of 4000 random spectra. Despite the remarkable improvements, the performance remains poor.

4.2.3 Random Forest

class_weight = "balanced"

Evaluating the Random Forest Classifier (RF) over the complete dataset with *class_weight = "balanced"*, we obtain an accuracy of 64.0% and a mean cross validation over 4 sets of 0.630. The following confusion matrix 4.7 allows us to see the relationship between predicted and real values.

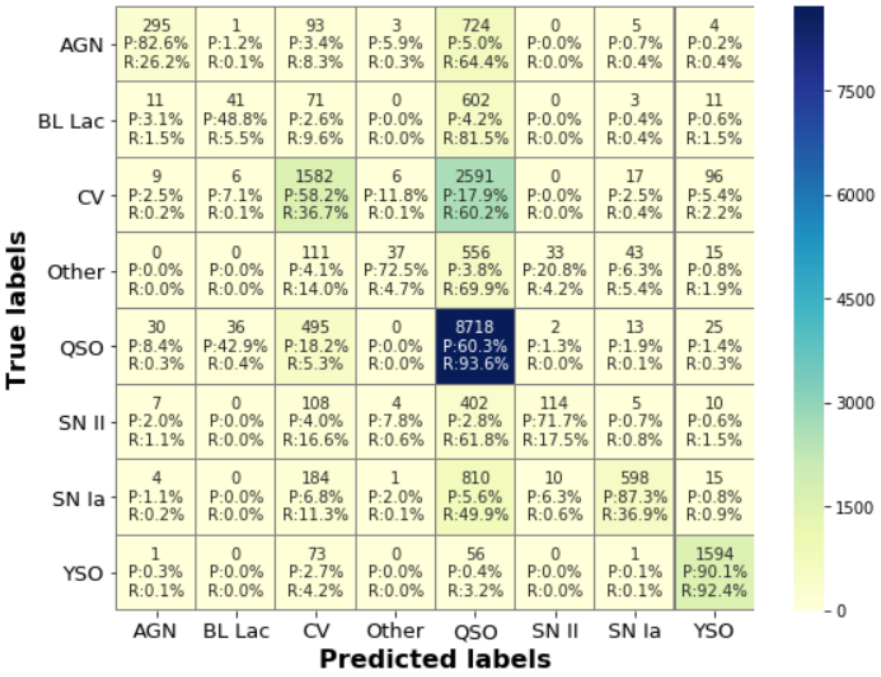


Figure 4.7: Random Forest ("balanced"). Confusion Matrix. Complete Dataset. The X-axis represents the class type predicted by the classifier and the Y-axis represents the true type. "P" represents the precision and "R" the recall of each element.

When this model predicts that the class is a "QSO", it has difficulties in asserting that it is really a "QSO" because it gets confused with the other classes. This decrease in the precision of the "QSO" class causes a decrease in the recall in almost all classes in the model and a considerable increase in the precision. The "YSO" class is the only one that is not affected by the precision decrease of the "QSO" class with a precision and recall higher than 90%. Other metrics can be found in table 4.11 below.

Random Forest Results ("Balanced")				
Stratified 4-fold (Accuracy)	**Precision**	**Recall**	**F1 score**	**Accuracy**
0.630	0.714	0.392	0.435	0.640

Table 4.11: Random Forest Results ("balanced"). Complete Dataset.

class_weight = "balanced_subsample"

Using *class_weight = "balanced_subsample"*, an accuracy of 64.1% and a mean cross validation over 4 sets of 0.631 are obtained. The following confusion matrix 4.8 allows us to see the relationship between predicted and real values.

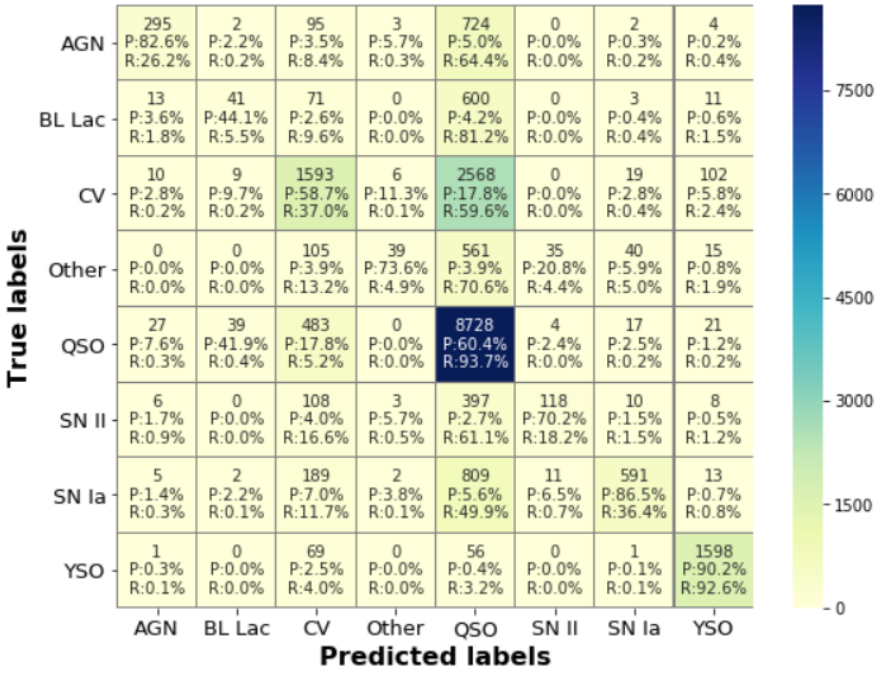


Figure 4.8: Random Forest ("balanced_subsample"). Confusion Matrix. Complete Dataset. The X-axis represents the class type predicted by the classifier and the Y-axis represents the true type. "P" represents the precision and "R" the recall of each element.

This confusion matrix created for the *"balanced_subsample"* model is scarcely different from the *"balanced"* model. The structure and distribution of the data is identical. Other metrics can be found in table 4.12 below.

Random Forest Results ("Balanced Subsample")				
Stratified 4-fold (Accuracy)	**Precision**	**Recall**	**F1 score**	**Accuracy**
0.631	0.708	0.393	0.437	0.641

Table 4.12: Random Forest Results ("balanced_subsample"). Complete Dataset.

As we can see, the metrics obtained for both models are almost identical. It should be noted that the model has a better precision but a low recall. This is an indication that the model is not able to detect the class very well, but when it detects the class, it is quite reliable (Na819). Furthermore, the values of the metrics obtained for the complete dataset improve with respect to the subset of data. This classifier has shown the best performance so far.

4.2.4 Gradient Boosting Classifier

Applying the Gradient Boosting Classifier (GBC) over the whole dataset, we obtain an accuracy of 66.8% and a mean cross validation over 4 sets of 0.659. The following confusion matrix 4.9 allows us to see the relationship between predicted and actual values.

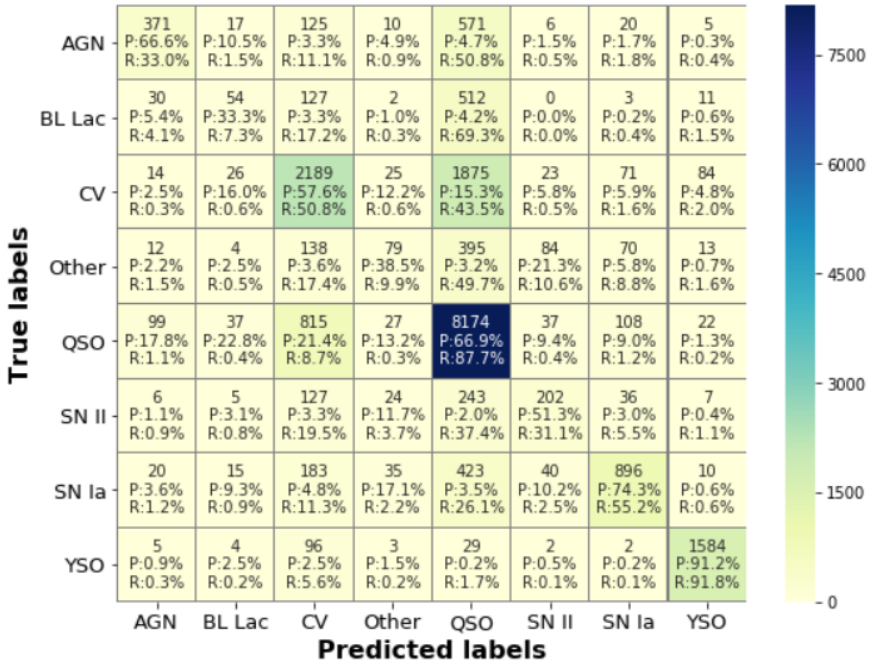


Figure 4.9: Gradient Boosting Classifier. Confusion Matrix. Complete Dataset. The X-axis represents the class type predicted by the classifier and the Y-axis represents the true type. "P" represents the precision and "R" the recall of each element.

In comparison with the confusion matrix obtained for Random Forest, we observed an increase in recall in almost all classes where "CV", "QSO", "SN Ia" and "YSO" stand out. Although the precision decreases slightly, good percentages are obtained for "AGN", "CV", "QSO", "SN Ia" and "YSO". The performance of the "YSO" class is at its best with the Gradient Boosting Classifier. Other metrics can be found in the table 4.13 below.

Gradient Boosting Classifier Results				
Stratified 4-fold (Accuracy)	**Precision**	**Recall**	**F1 score**	**Accuracy**
0.659	0.600	0.459	0.494	0.668

Table 4.13: Gradient Boosting Classifier Results. Complete Dataset.

This model manages to drastically improve the results obtained in the subset and to become the best classification model obtained so far.

4.3 Artificial Neural Network

After carrying out an exhaustive analysis of the best architecture (Table 3.8) for the Artificial Neural Network (ANN) and the best values for the hyperparameters (Table 3.9) over the subset, the best three models obtained are shown in table 4.14.

Nº	Architecture	Learn. Rate	Batch Size	Epochs	Optimizer	Activation	Loss	Accuracy
1	[120, 96, 74, 52, 30, 8]	0.01	64	50	"Adamax"	"selu"	1.900	0.461
2	[120, 92, 64, 36, 8]	0.01	32	50	"Adamax"	"softsign"	1.763	0.456
3	[120, 90, 60, 20, 8]	0.01	64	100	"Adamax"	"elu"	2.519	0.450

Table 4.14: Best three models ANN. Subset.

As we can see, the optimizer and the learning rate are the same for the first three models. Furthermore, only two values alternate on the epochs column, 50 and 100. The same happens for the batch size with 32 and 64 samples. The results provided by the subset gave us models with a low accuracy.

As the results were not favourable, we again carried out a search to obtain the best hyperparameter values over two new architectures where the first hidden layer is much larger than the input layer. These new architectures were proposed to check if the model needs an increase of neurons in the second layer in order to make a good classification. The best models for these two architectures are shown in table 4.15.

Nº	Architecture	Learn. Rate	Batch Size	Epochs	Optimizer	Activation	Loss	Accuracy
4	[120, 200, 100, 50, 20, 8]	0.01	64	50	"Adamax"	"elu"	1.956	0.435
5	[120, 256, 150, 75, 30, 8]	0.01	32	50	"Adamax"	"selu"	4.077	0.431

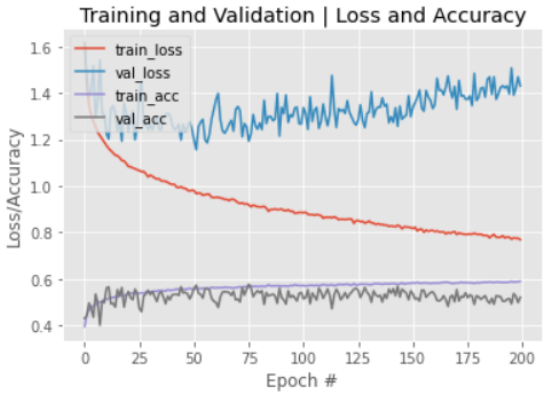
Table 4.15: Second Search. Best two models ANN. Subset.

Although these architectures did not surpass the first ones, we studied all models shown in the complete dataset to see if the accuracy improved with respect to the previous models.

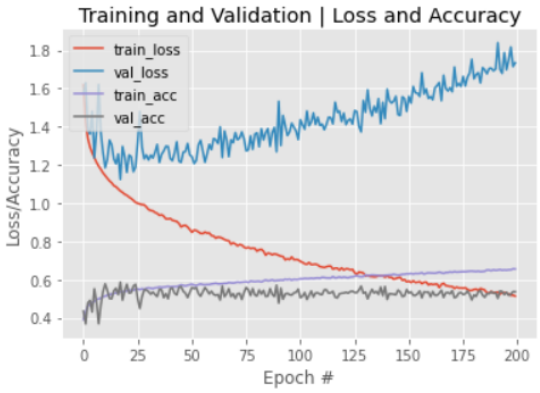
	Subset		Complete Dataset	
Nº	**Loss**	**Accuracy**	**Loss**	**Accuracy**
1	1.900	0.461	1.263	0.510
2	1.763	0.456	1.229	0.522
3	2.519	0.450	1.238	0.540
4	1.956	0.435	1.256	0.546
5	4.077	0.431	1.375	0.493

Table 4.16: Results ANN. Complete Dataset.

From the results in table 4.16, we can see that the models Nº3 three and Nº4 are the best because they have the highest accuracy. Despite the results, we increased the number of epochs of the two best models to 200 and observed their graphs to see if there was still room for improvement.



(a) Model Nº3



(b) Model Nº4

Figure 4.10: Evolution of the accuracy and the loss of the training and validation sets according to the epochs.

Accuracy did not tend to improve and overfitting could be seen on the two different models because of train_loss << val_loss, especially in model Nº4. Although the graphs of models Nº2 and Nº1 were not represented, they follow the same trend as Nº3.

On model Nº3 results did not improve beyond 50-100 epochs so we applied the technique of *early stopping* on models Nº1, Nº2 and Nº3. In spite of the excessive overfitting suffered by model Nº4, we also applied this technique to it. In this way, the models used as many epochs as necessary while avoiding overfitting. Furthermore, on this algorithm we tuned the patience, that is the number of epochs with no improvement after which training will be stopped. By applying this technique the performance was slightly improved.

	Subset		Complete Dataset		Early Stopping			
Nº	**Loss**	**Accuracy**	**Loss**	**Accuracy**	**Loss**	**Accuracy**	**Epochs**	**Patience**
1	1.900	0.461	1.263	0.510	1.207	0.551	43	25
2	1.763	0.456	1.229	0.522	1.206	0.558	58	40
3	2.519	0.450	1.238	0.540	1.259	0.542	87	35
4	1.956	0.435	1.256	0.546	1.235	0.543	33	15

Table 4.17: Results ANN. Subset. Complete Dataset. Early Stopping.

The accuracy of the models when applying the *early stopping* technique increased, except in model Nº4 as it is shown in table 4.17.

In order to try to solve the overfitting in another way, we tried to reduce the complexity of the network with architectures which had fewer neurons and layers. We again carried out a search to obtain the best hyperparameter values over new architectures. The best models appear in table 4.18.

Nº	**Architecture**	**Learn. Rate**	**Batch Size**	**Epochs**	**Optimizer**	**Activation**	**Loss**	**Accuracy**
6	[120, 32, 16, 8]	0.01	64	100	"Adamax"	"selu"	2.008	0.445
7	[120, 60, 30, 8]	0.01	32	100	"Adamax"	"selu"	2.957	0.434

Table 4.18: Third Search. Best two models ANN. Subset.

These models were evaluated on the complete dataset to see if accuracy and losses improved.

	Subset		Complete Dataset	
Nº	**Loss**	**Accuracy**	**Loss**	**Accuracy**
6	2.008	0.445	1.272	0.552
7	2.957	0.434	1.213	0.586

Table 4.19: Results ANN. Third Search. Complete Dataset.

Model Nº7 achieved quite acceptable results. Below are the graphs of the evolution of the accuracy and the loss of the training and validation sets.

(a) Model N°6 (b) Model N°7

Figure 4.11: Evolution of the accuracy and the loss of the training and validation sets according to the epochs.

The three best models obtained are compiled in table 4.20 to decide the best model capable of classifying the satellite alerts.

	Subset		Complete Dataset		Early Stopping			
N°	**Loss**	**Accuracy**	**Loss**	**Accuracy**	**Loss**	**Accuracy**	**Epochs**	**Patience**
2	1.763	0.456	1.229	0.522	1.206	**0.558**	58	40
6	2.008	0.445	1.272	**0.552**	-	-	-	-
7	2.957	0.434	1.213	**0.586**	-	-	-	-

Table 4.20: Final Results ANN.

Although all three models have a similar accuracy the element with the highest value is model N°7. We also find little difference for the losses but it is model N°2 the one with the best performance. Despite the fact that the results are very similar, we establish model N°7 as the best Artificial Neural Network. The properties of this ANN are:

- **Architecture**.
  - *Input layer*. The input layer consists of the 120 points that constitute the spectrum and the "selu" activation function.
  - *Hidden layers*. We have a total of 2 hidden layers [60, 30] whose neurons decrease until they approach the output layer. Each of these layers have 'selu" as its activation function and the "Categorical Cross Entropy" as its loss function.
  - *Output layer*. The output layer has as many neurons as classes have our dataset, in other words, eight.

- **Optimizer**. Adamax.
- **Learning Rate**. 0.01
- **Epochs**. 100 times that the entire dataset is passed through the network.
- **Batch Size**. 32 samples that are passed through the network at each epoch.

The confusion matrix 4.12 allows us to see the relationship between predicted and real values. The labels are explained in Appendix B: Information about datasets.

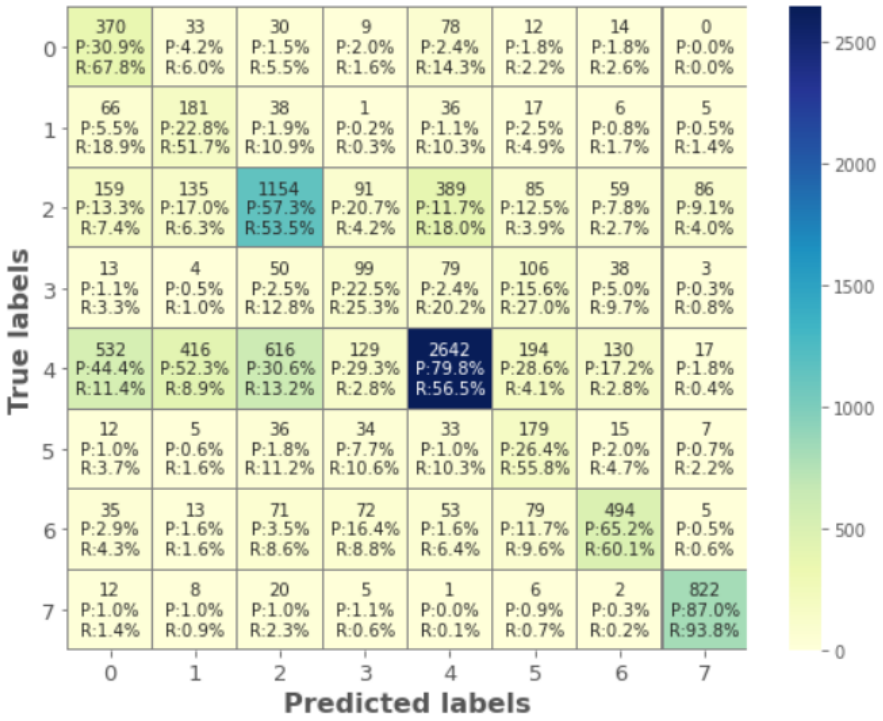


Figure 4.12: Artificial Neural Network. Confusion Matrix. Complete Dataset. The X-axis represents the class type predicted by the classifier and the Y-axis represents the true type. "P" represents the precision and "R" the recall of each element.

The confusion matrix of the Artificial Neural Network shows excellent percentages for the class "YSO" (7). Although the "QSO" (5) class obtains 79.8% of precision, the rest of the class precision percentages are lower than 65%. The same happens with the recall whose best percentage apart from "YSO" (7) is 67.8%. Other metrics can be found in table 4.21.

Artificial Neural Network			
Precision	**Recall**	**F1 score**	**Accuracy**
0.490	0.581	0.510	0.586

Table 4.21: Artificial Neural Network Results. Complete Dataset.

4.4 Comparison of models

The performance of the models obtained, which we can find in table 4.22, is not very high with an accuracy ranging from 32% to 67% approximately. When dealing with a balanced data subset we see how the Decision Tree Classifier is unable to correctly classify the alerts. However, when we consider a set of Decision Trees things change. The Random Forest Classifier achieves the best performance for the subset with an accuracy of 53.5%. Behind the Random Forest, we find the Gradient Boosting Classifier with 48.9% and the Support Vector Machines with 40.0% which do not classify clearly.

By processing these models over the entire dataset, we obtained a significant improvement on all metrics. The Decision Tree slightly improves its performance to an accuracy of 35.3%. Moreover, on the Support Vector Machines we can see a clear improvement capable of reaching up to 50.6% accuracy but still not exceeding the accuracy obtained by the Random Forest in the subset.

Despite the effort made to optimize the configuration of the Artificial Neural Network, it does not manage to be the best model. However, the ANN surpasses the performance obtained for the Decision Tree and the Support Vector Machines with an accuracy of 58.6%.

Finally, the models that have given us the best performance are the Random Forest and the Gradient Boosting Classifier. The Random Forest, with an accuracy of 64.0% (*"Balanced"*) and 64.1% (*"Balanced Subsample"*), has as a main characteristic, the high precision and low recall. And above the Random Forest, the Gradient Boosting Classifier has a more balanced precision and recall and the best accuracy of all models under study with 66.8%.

Model	Cross Val.	Precision	Recall	F1 score	Accuracy
DT (Subset)	0.335	0.331	0.331	0.325	0.328
SVM (Subset)	0.403	0.399	0.403	0.399	0.400
RF (Subset)	0.517	0.539	0.539	0.530	0.535
GBC (Subset)	0.472	0.491	0.490	0.486	0.489
DT	0.350	0.336	0.400	0.327	0.353
SVM	0.495	0.420	0.491	0.438	0.506
RF ("Balanced")	0.630	0.714	0.392	0.435	0.640
RF ("Balanced Subsample")	0.631	0.708	0.393	0.437	0.641
GBC	0.659	0.600	0.459	0.494	0.668
ANN	-	0.490	0.581	0.510	0.586

Table 4.22: Metrics. Comparison of models.

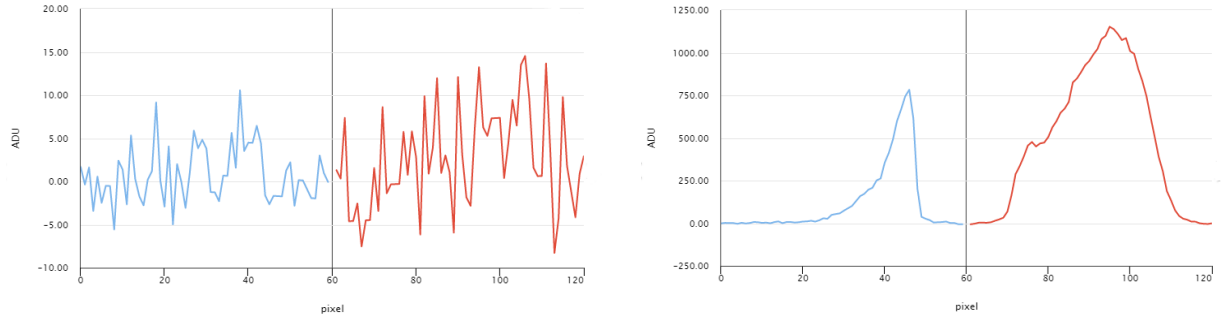
4.5 The reasons for the outstanding performance of the "YSO" class

The high precision and recall values obtained for the "YSO" class is impressive. Moreover, this fact is present in all the models tested. These values are shown in table 4.23.

YSO			
Model	**Precision**	**Recall**	**F1 score**
DT (Subset)	0.792	0.851	0.820
SVM (Subset)	0.750	0.830	0.788
RF (Subset)	0.838	0.936	0.884
GBC (Subset)	0.882	0.872	0.877
DT	0.798	0.853	0.826
SVM	0.781	0.901	0.837
RF ("Balanced")	0.901	0.924	0.912
RF ("Balanced Subsample")	0.902	0.926	0.914
GBC	0.912	0.918	0.914
ANN	0.870	0.938	0.903

Table 4.23: Precision, Recall and F1-score obtained for the "YSO" class. F1-score is calculated using equation 6.

The best performance is achieved by the RF (*"Balanced Subsample"*) and the GBC with a F1 Score of 0.914. And the worst is achieved by the SVM classifier evaluated on the subset with a not bad F1-Score of 0.788. The figure 4.13 shows a comparison between a random spectra from "YSO" class and one from "CV" class.

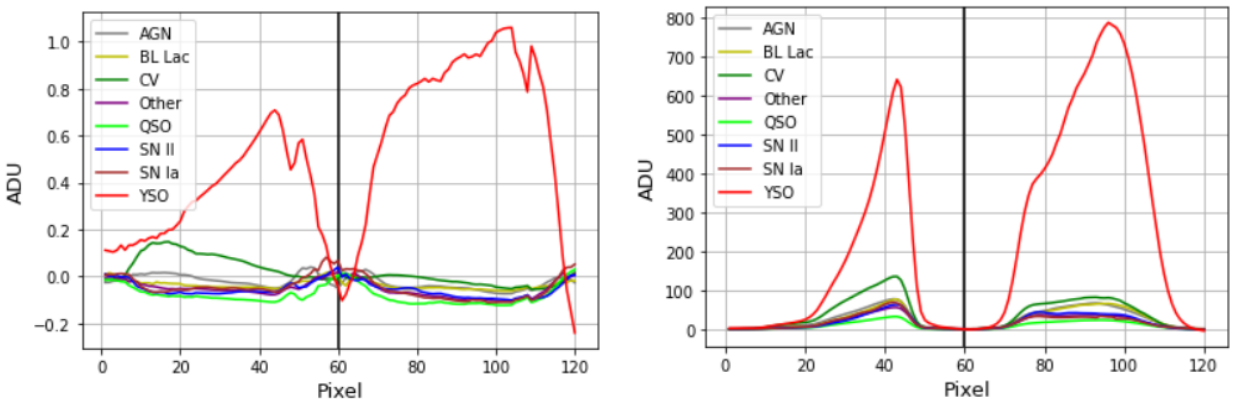


(a) Entry: Gaia20eja. Class: "CV"

(b) Entry: Gaia20azw. Class:"YSO"

Figure 4.13: BP and RP graphics comparing "YSO" and "CV" classes. Analog Digital Units (ADU) vs. Pixel.

At first glance we can see how the ADU of the "YSO" graph acquire values much higher than those obtained by the "CV" class. Despite this clear difference, we can not draw relevant conclusions by comparing only two spectra. Hence, in order to see the trend of the spectra associated to each of the classes involved in the study, the average of each of the 120 elements that compose the spectra was taken for each class. That is to say, for a given class, we took the average for each of the columns ("bp_i" and "rp_i") of the "Spectra Columns dataset". The results are shown for both standardized and non-standardized data.



(a) Standardized data

(b) Non-standardized data

Figure 4.14: Trend of the spectra associated to each of the classes involved in the study. Analog Digital Units (ADU) vs. Pixel.

As a result of the graphs obtained, a significant difference between the "YSO" class and the rest of the classes can be appreciated. This difference can be considered as the main cause of the good performance of the models. In other words, the distinction of the "YSO" class from the other classes is simple for classifiers due to values difference. However, the other classes are in a similar range of values and therefore their distinction is more complicated.

Chapter 5

Conclusions

The project started with a small astrophysical introduction to put into context the data we were going to use for the construction of the models and the importance of the of the Gaia satellite's mission. Once the context was understood, one of the most important stages was carried out: the extraction of data by means of web scraping techniques. In spite of the slowness in the extraction of the code, the photometric spectra of each of the alerts involved in the study were obtained with total success.

As soon as the necessary data for the training of the models was obtained and after a previous pre-processing, the implementation of the classifiers took place. As we have seen, the subset only served to obtain the best values for the hyperparameters. The low performance of these models means that they are not decisive when choosing the final classifier. Although all the classifiers manage to improve their performance over the complete dataset, the best model is the Gradient Boosting Classifier with an accuracy of 66.8%. Furthermore, the impressive performance presented by the "YSO" class in all models due to the significant difference with the rest of the classes must be highlighted. It is also worth mentioning the large amount of time dedicated to obtain an optimized configuration of the Artificial Neural Network: varying the number of neurons, layers, epochs, learning rate and batch size as well as implementing different optimizers and activation functions. Despite the efforts put into building the network, the performance obtained only place it as the third best model.

At the beginning of this work we were faced with a complex classification problem due to the non-linearity of the problem to be traced. In other words, on certain occasions we had to go backwards in order to take two steps forward. The type of classification that we carried out based on the graphs of the photometric spectra where we involved eight different classes is complicated even for the human eye, so the search for a good classifier is, at least, tedious. Despite this, we can say that after a general test on different types of classifiers, it is feasible

to carry out models that allow spectra to be classified automatically and with acceptable performance. In this way, we highlight the help that the implementation of machine learning techniques can provide in the process of classifying the alert, which together with specialized professionals who can contrast the opinion of the algorithm would drastically reduce the time needed to classify the alerts.

Chapter 6

Future Steps

As a more detailed classification study is feasible, a large number of points that can be covered in the future should be addressed. Firstly, we must start by considering more data that would give consistency to the classes. And secondly, a new pre-processing should be carried out to see if any of the discarded classes have come to the fore.

Furthermore, we could also consider the implementation of a model where each general entry only takes into account the evolution of a certain number of detections immediately before the alert is triggered, the alert itself and a number of other detections immediately after the alert. An example, in this case, would be to try two detections before, the alert and two detections after. This is only an option and the number of input and output elements could be varied in order to optimize the results.

It should be noted that in future steps we would recommend knowing the type of alerts we will be working with. On the one hand, we have certain objects such as variable stars or quasars whose spectra are more periodic, and possibly, considering an evolution of the detections would help in the classification. On the other hand, for phenomena such as a flare or a supernova explosion it would be interesting to take only the main alert. This reduction of elements in the study would be beneficial because these phenomena are more sporadic or instantaneous.

In addition to a specific treatment of the classes, we would recommend the introduction of new techniques that have not been able to be addressed in this work such as the Recurrent Neural Network or even the implementation of Cascading Classifiers on those models which provide better performance.

As a validation method, once a relatively "reliable" model obtained, all those alerts with a comment such as "possible YSO" should be extracted and classified with the trained model. In this way, the behaviour of the classifier towards these possible candidates for a given alert could be studied.

In conclusion, the continuation of this work is feasible and there are a large number of options that deserve to be studied carefully.

Bibliography

[ACM16] Sarah Guido Andreas C. Mueller. *Introduction to Machine Learning with Python: A Guide for Data Scientists*. O'Reilly Media, 2016.

[Age15] European Space Agency. Gaia overview, 2015. URL: https://www.esa.int/Science_Exploration/Space_Science/Gaia/Gaia_overview.

[Age20] European Space Agency. Gaia early data release 3 (gaia edr3), 2020. URL: https://www.cosmos.esa.int/web/gaia/earlydr3.

[Alg18a] Algorithmia. Introduction to loss functions, 2018. URL: https://algorithmia.com/blog/introduction-to-loss-functions.

[Alg18b] Algorithmia. Introduction to optimizers, 2018. URL: https://algorithmia.com/blog/introduction-to-optimizers.

[BKW+14] Nadejda Blagorodnova, Sergey E. Koposov, Łukasz Wyrzykowski, Mike Irwin, and Nicholas A. Walton. gs-tec: the gaia spectrophotometry transient events classifier. *Monthly Notices of the Royal Astronomical Society*, 442(1):327–342, Jun 2014. URL: http://dx.doi.org/10.1093/mnras/stu837, doi:10.1093/mnras/stu837.

[BLB+19] Yu Bai, JiFeng Liu, ZhongRui Bai, Song Wang, and DongWei Fan. Machine-learning regression of stellar effective temperatures in the second gaia data release. *The Astronomical Journal*, 158(2):93, Aug 2019. URL: http://dx.doi.org/10.3847/1538-3881/ab3048, doi:10.3847/1538-3881/ab3048.

[BLW18] Yu Bai, Ji-Feng Liu, and Song Wang. Machine learning classification of gaia data release 2. *Research in Astronomy and Astrophysics*, 18(10):118, Oct 2018. URL: http://dx.doi.org/10.1088/1674-4527/18/10/118, doi:10.1088/1674-4527/18/10/118.

[BLWW20] Yu Bai, JiFeng Liu, YiLun Wang, and Song Wang. Machine-learning regression of extinction in the second gaia data release. *The Astronomical Journal*, 159(3):84, Feb 2020. URL: http://dx.doi.org/10.3847/1538-3881/ab63d5, doi:10.3847/1538-3881/ab63d5.

[Bro19] Jason Brownlee. Understand the impact of learning rate on neural network performance, 2019. URL: https://machinelearningmastery.com/understand-the-dynamics-of-learning-rate-on-deep-learning-neural-networks/

[Gé19] Aurélien Géron. *Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems*. O'Reilly Media, 2nd edition, 2019.

[HBK+18] Amina Helmi, Carine Babusiaux, Helmer H. Koppelman, Davide Massari, Jovan Veljanoski, and Anthony G. A. Brown. The merger that led to the formation of the milky way's inner stellar halo and thick disk. *Nature*, 563(7729):85–88, Oct 2018. URL: http://dx.doi.org/10.1038/s41586-018-0625-x, doi:10.1038/s41586-018-0625-x.

[HV12] C. Fabricius et al. H. Voss, C. Jordi. Exoplanetary transits as seen by gaia. *Highlights of spanish astrophysics VII,Proceedings of the X Scientific Meeting of the Spanish Astronomical Society*, July 2012.

[IoA15] UK Institute of Astronomy, University of Cambridge. Gaia photometric science alerts, 2015. URL: http://gsaweb.ast.cam.ac.uk/alerts/home.

[Jai18] Vandit Jain. Everything you need to know about "activation functions" in deep learning models, 2018. URL: https://towardsdatascience.com/everything-you-need-to-know-about-activation-functions-in-deep-learning-mo

[JG17] J. Minguillón R. Caihuelas J. Gironés, J. Casas. *Minería de datos. Modelos y algoritmos*. Editorial UOC, Barcelona, 2017.

[JGC+10] C. Jordi, M. Gebran, J. M. Carrasco, J. de Bruijne, H. Voss, C. Fabricius, J. Knude, A. Vallenari, R. Kohley, and A. Mora. Gaia broad band photometry. *Astronomy Astrophysics*, 523:A48, Nov 2010. URL: http://dx.doi.org/10.1051/0004-6361/201015441, doi:10.1051/0004-6361/201015441.

[Mig19] F. Mignard. The gaia mission and significance, 2019. arXiv:1906.09022.

[MIM18] Khyati Malhan, Rodrigo A Ibata, and Nicolas F Martin. Ghostly tributaries to the milky way: charting the halo's stellar streams with the gaia dr2 catalogue. *Monthly Notices of the Royal Astronomical Society*, 481(3):3442–3455, Sep 2018. URL: http://dx.doi.org/10.1093/mnras/sty2474, doi:10.1093/mnras/sty2474.

[ME+19] G Marton, P Ábrahám, E Szegedi-Elek, J Varga, M Kun, Á Kóspál, E Varga-Verebélyi, S Hodgkin, L Szabados, R Beck, and et al. Identification of young stellar object candidates in the gaia dr2 x allwise catalogue with machine learning methods. *Monthly Notices of the Royal Astronomical Society*, 487(2):2522–2537, May 2019. URL: http://dx.doi.org/10.1093/mnras/stz1301, doi:10.1093/mnras/stz1301.

[Na819] Na8. Clasificación con datos desbalanceados, 2019. URL: https://www.aprendemachinelearning.com/clasificacion-con-datos-desbalanceados/#:~:text=Alta%20precision%20y%20bajo%20recall,logra%20clasificar%20la%20clase%20correctamente.

[PdBB+16] T. Prusti, J. H. J. de Bruijne, A. G. A. Brown, A. Vallenari, C. Babusiaux, C. A. L. Bailer-Jones, U. Bastian, M. Biermann, D. W. Evans, and et al. The gaia mission. *Astronomy Astrophysics*, 595:A1, Nov 2016. URL: http://dx.doi.org/10.1051/0004-6361/201629272, doi:10.1051/0004-6361/201629272.

[PWB18] Adrian M. Price-Whelan and Ana Bonaca. Off the beaten path: Gaia reveals gd-1 stars outside of the main stream. *The Astrophysical Journal*, 863(2):L20, Aug 2018. URL: http://dx.doi.org/10.3847/2041-8213/aad7b5, doi:10.3847/2041-8213/aad7b5.

[RLGBC20] Tomás Ruiz-Lara, Carme Gallart, Edouard J. Bernard, and Santi Cassisi. The recurrent impact of the sagittarius dwarf on the milky way star formation history, 2020. arXiv:2003.12577.

[SBG+18] Ken J. Shen, Douglas Boubert, Boris T. Gänsicke, Saurabh W. Jha, Jennifer E. Andrews, Laura Chomiuk, Ryan J. Foley, Morgan Fraser, Mariusz Gromadzki, James Guillochon, and et al. Three hypervelocity white dwarfs in gaia dr2: Evidence for dynamically driven double-degenerate double-detonation type ia supernovae. *The Astrophysical Journal*, 865(1):15, Sep 2018. URL: http://dx.doi.org/10.3847/1538-4357/aad55b, doi:10.3847/1538-4357/aad55b.

[Wan18] Zichen Wang. Practical tips for class imbalance in binary

classification, 2018. URL: https://towardsdatascience.com/practical-tips-for-class-imbalance-in-binary-classification-6ee29bcdb8a7.

[WHB+12] Lukasz Wyrzykowski, Simon Hodgkin, Nadejda Blogorodnova, Sergey Koposov, and Ross Burgon. Photometric science alerts from gaia, 2012. arXiv:1210.5007.

[Zhe15] Alice Zheng. *Evaluating machine learning models : a beginner's guide to key concepts and pitfalls*. O'Reilly Media, first edition. edition, 2015.

Appendices

Appendix A: Documents

Documents attached to the dissertation. These documents can be found in GitHub.

CSV Documents	
Name	**Description**
alerts22102020_920.csv	Web page data
less50.csv	Used in section 3.1.1
Spectrums.csv	"Spectra Dataset"
Spectrums_columns.csv	"Spectra Columns Dataset"
Subset_14_12_2020.csv	Subset used in section 3.3.1
SubsetANN_14_12_2020.csv	Subset used in section 3.3.3
Lay_results_26122020.csv	Results of the table 4.14

Table 6.1: CSV Documents. "Spectrums.csv" and "Spectrums_columns.csv" are located in Zenodo.

Code Documents	
Name	**Description**
1_Load_Scrap_Gaia.ipynb	Compiles what is described in section 3.1
2_Data_Pre_Processing.ipynb	Compiles what is described in section 3.2
3_Subset_4000_GS.ipynb	Compiles what is described in section 3.3.1
4_Complete_dataset.ipynb	Compiles what is described in section 3.3.2
5_ANN.ipynb	Compiles what is described in section 3.3.3
6_YSO.ipynb	Compiles what is described in section 4.5
utils_gaia.py	Functions used in the project

Table 6.2: Code Documents.

Appendix B: Information about datasets

The following eleven entries will not appear in the "Spectra Dataset".

Name
Gaia20eqo
Gaia20dwa
Gaia19fkd
Gaia19erx
Gaia19emu
Gaia19ddq
Gaia19ckq
Gaia18cxm
Gaia18buw
Gaia18bto
Gaia16adk

Table 6.3: Entries without spectra.

The frequency and the number of spectra for each feature that we can find in the "Spectra Dataset" have been collected in table 6.4.

Feature	Frequency	Number of Spectra
QSO	633	46355
CV	457	21786
YSO	134	8509
SN Ia	1364	8152
AGN	78	5567
Other	737	4064
BL Lac	51	3644
SN II	419	3329
TOTAL	**3873**	**101406**

Table 6.4: Features, frequency and number of spectra.

In table 6.5 a description of the eight classes used and the different ways to label them is compiled.

Feature	Description	One-Hot-Encoding	Number Label
AGN	Active galactic nucleus	[1,0,0,0,0,0,0,0]	0
BL Lac	Type of active galactic nucleus	[0,1,0,0,0,0,0,0]	1
CV	Cataclysmic variable stars	[0,0,1,0,0,0,0,0]	2
Other	Other feature	[0,0,0,1,0,0,0,0]	3
QSO	Quasar	[0,0,0,0,1,0,0,0]	4
SN II	Supernova II	[0,0,0,0,0,1,0,0]	5
SN Ia	Supernova Ia	[0,0,0,0,0,0,1,0]	6
YSO	Young Stellar Object	[0,0,0,0,0,0,0,1]	7

Table 6.5: Description of the features and possible labels.

Appendix C: Metrics

In this appendix the different metrics used in the dissertation will be explained. Firstly, we will look at the confusion matrix that gives us a matrix capable of fully describing the performance of the model. This matrix has the following form.

		Predictions		
		Positive	Negative	Total
Real values	Positive	TP	FN	$TP + FN$
	Negative	FP	TN	$TN + FP$
	Total	$TP + FP$	$TN + FN$	N

Table 6.6: Binary confusion matrix structure.

Precision - Recall

The **precision** shows the ratio between the predictions that are correct and the total number of positive predictions made by the classifier.

$$Precision = \frac{Number\ of\ correct\ positive\ predictions}{Number\ of\ positive\ predictions\ made} = \frac{TP}{TP + FP} \tag{6.1}$$

The precision is intuitively the ability of the classifier not to label as positive a sample that is negative.

The **recall** shows the ratio between the predictions that are correct and the total number of positive examples made by the classifier.

$$Recall = \frac{Number\ of\ correct\ positive\ predictions}{Number\ of\ positive\ examples} = \frac{TP}{TP + FN} \tag{6.2}$$

The recall is intuitively the ability of the classifier to find all the positive samples.

F1-Score

The value F1 is used to combine precision and recall measurements in a single value with equal importance. This value facilitates the handling of these two values simultaneously in order to be able to compare different models.

$$F1 = 2 \cdot \frac{precision \cdot recall}{precision + recall} \tag{6.3}$$

Accuracy

Accuracy simply measures how often the classifier makes the correct prediction. It is the ratio between the number of correct predictions and the total number of predictions. (Zhe15)

$$Accuracy = \frac{Correct\ predictions}{Total\ number\ of\ predictions\ made} \tag{6.4}$$

For multi-class problems, as in this dissertation, we have to establish a hyperparameter to help us obtain these metrics. Python, through the *scikit-learn* library, provides us with the hyperparameter *average*. In our case, it receives the *"micro"* value by means of which we are able to calculate the metrics globally.