Facultad
de
Ciencias

CLASIFICACIÓN DE FUENTES ASTRONÓMICAS MEDIANTE TÉCNICAS DE MACHINE LEARNING

(Classification of astronomical sources through Machine Learning techniques)

Trabajo de Fin de Grado
para acceder al

GRADO EN FÍSICA

Autor: Pablo Gómez Nicolás

Directora: Amalia Corral Ramos

Co-Director: Francisco Jesús Carrera Troyano

Octubre - 2019

Acknowledgements

First of all, I want to thank my director, Amalia Corral, for her help, confidence and guidance in this project. She is the one that has always guided me during the work, met me in her office and studied with me the results obtained. Thank you very much for your care and your patience: I have modified some meeting hours at the very last moment, and you have always forgiven me for my chaotic schedule. I want to thank, too, my co-director, Francisco Carrera, for helping me with the machine learning scripts and offering us always a second point of view, so useful and important in some points of the project.

Also, a special thanks to my family, who has supported me while I was doing and writing this report; and to Pablo Señas, who has submitted it in Santander while I was in Granada. Thank you for becoming the last link in this chain.

In my Maths end-of-degree project, I was very detailed when I wrote the acknowledgements page. As I thanked there the most important people during these last years, I just want to cite a quote from "The Little Prince", that I think that suits an astronomy work perfectly:

"All men have the stars, [...] but they are not the same things for different people. For some, who are travelers, the stars are guides. For others they are no more than little lights in the sky. For others, who are scholars, they are problems. [...] But all these stars are silent. You- you alone- will have the stars as no one else has them."

With this quote, I want to thank all the people that, during my life (in an academic and personal sense) have motivated me to see the world with different eyes, showing me how to see and deal with the problems as no one else does. I am not going to appoint anybody, but if you are reading this and you feel like you are one of these persons, I am eternally grateful for your influence in what I have become. Thank you.

Abstract/Resumen

Classification of astronomical sources through Machine Learning techniques

In this project, we have used unsupervised Machine Learning techniques in order to perform classificatory tasks of different kinds of astronomical sources. Specifically, we have taken photometric data from different catalogues relative to wavelengths within the near-infrared and mid-infrared regions.

After processing and filtering our initial data set in different ways, we have applied dimensionality reduction techniques to transform our problem data set. These transformed data have been used to apply different clustering methods, in order to distinguish different subgroups corresponding, in principle, to different kinds of radiating sources.

The data set which have been used to apply Machine Learning techniques is composed of more than 100000 sources. The automatic clustering methods have not been able to adapt to the topology of these data with reduced dimensionality. A manual definition of the groups produces instead a much closer match to different classes of astronomical sources.

Key words: Data Science, source classification, infrared, cross-correlation, unsupervised Machine Learning, dimensionality reduction, clustering.

Clasificación de fuentes astronómicas mediante técnicas de Machine Learning

En este proyecto, se han utilizado técnicas de Machine Learning no supervisado para realizar tareas clasificatorias de diferentes tipos de fuentes astronómicas. En concreto, se han tomado de diferentes catálogos datos fotométricos para longitudes de onda en las regiones del infrarrojo cercano e infrarrojo medio.

Tras procesar y filtrar de diversas formas nuestro conjunto inicial de datos, se ha procedido a aplicar técnicas de reducción de la dimensionalidad para transformar nuestro conjunto problema. A estos datos transformados se les ha aplicado diferentes técnicas de clustering, con el fin de distinguir subgrupos correspondientes, en principio, a diferentes tipos de fuentes de radiación.

El conjunto de datos al que se le han aplicado técnicas de Machine Learning tiene más de 100000 muestras. Los métodos de clustering no supervisado no han sido capaces de adaptarse a la topología de estos datos tras reducir su dimensionalidad. En cambio, una definición manual de los grupos produce una mayor correspondencia con diferentes tipos de fuentes astronómicas.

Palabras clave: Ciencia de Datos, clasificación de fuentes, infrarrojo, cross-correlación, Machine Learning no supervisado, reducción de dimensionalidad, clustering.

Table of contents

Chapter 1

Introduction

Astronomy is changing. This can be easily checked if we compare the way astronomers used to make and understand observations and how astronomers study the celestial sphere nowadays. The poetic image of a scientist using a rudimentary telescope to observe the moons of Jupiter has evolved into a research group working with a lot of computers in their respective offices.

What is the reason for this drastic change? The answer is easy: of course, the technological development. Nowadays, the technology available allows us to obtain extremely accurate astrometric and photometric data, even from faint sources that cannot be detected easily. Different technologies have also been developed, that can take data from different sky regions and save it automatically, without human supervision. These two advances have caused an enormous increase of the amount of astronomical data. There are many catalogues that contain information about millions of celestial sources, both astrometric and photometric data. These catalogues, far from stopping growing, are obtaining new data at a startling rate.

Therefore, some of the main features that astronomers are developing nowadays are ways of handling those huge amounts of data with relative ease. This trend, actually, is not only concerning Astronomy. We are living in the big data era: in a lot of sectors, like publicity and economy, it is necessary to save and deal with enormous data sets. Hence, astronomers do not walk alone in this path: these methods are developing fast enough in such a way that scientists can benefit from them.

In the other hand, some astronomical problems continue to be relevant. The one that concerns this work is a classification problem: how can we know which kind of celestial source we have detected? For some sources, this problem is easy to solve; however, in other cases it is not that easy, and we need to handle a lot of information about the source.

If we join these two problems, we obtain a more interesting problem: if we have a huge amount of astronomical data coming from different sources, is there a way to classify them automatically, without spending a large amount of time observing and studing each source individually? Different ways of solving this problem are being studied nowadays, but the answer to it seems to be in the area of Machine Learning. We will examine this area of data science later in this chapter.

The objective of this chapter is to explain the physical and mathematical insights under this work. Specifically, we will explain which data we will use to solve the classification problem, and where do these data come from; why we are using these data and not another data set; and which type of ML methods we are using and why. There is a guiding thread among all these answers: we want to avoid bias (i.e. perjuditial weights against or in favor of a previously set idea).

1.1 Astronomical sources

If we want to separate different astronomical sources depending on the light they emit, it is necessary to talk about the way we measure this information and the processes that underlie the energy

emission for each source. In this section, we will explain these topics. We will also explain in which region of the electromagnetic spectrum we are going to focus and why.

In photometry, the most used concept that describes how a radiating source emits energy is the magnitude. If we are working with a source that emits radiation with an observed energy flux density F (the units of F are $\mathrm{W} \cdot \mathrm{m}^{-2}$), the apparent magnitude of the source can be defined accurately as

$$m = -2.5 \log_{10} \frac{F}{F_0} \tag{1.1}$$

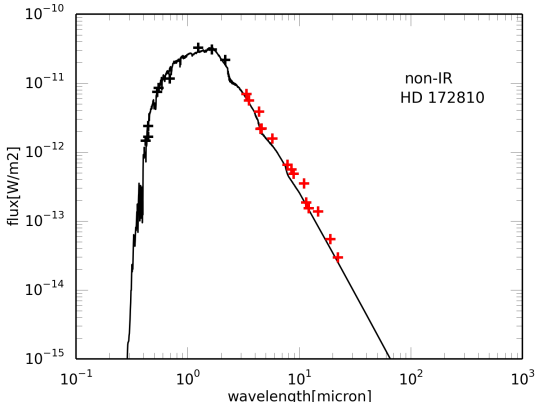
where F_0 is a reference flux that is selected as magnitude zero [1]. The "apparent" surname for the magnitude comes from the fact that it is related to the flux, which is proportional to L/d^2, where L is the luminosity of the source and d the distance to it. The "absolute" magnitude refers to the luminosity L, which is an intrinsic property of the source. The "apparent" surname also refers to the fact that the value depends on the reference flux F_0 chosen and on the instrument used to measure it: each instrument is more or less sensitive to the radiation with different wavelengths. The first fact is not important if we only want to compare magnitudes between them and it is not necessary to know the absolute flux density. The second fact can be exploited: the most accurate magnitude measurements can be obtained analysing only a small region of the electromagnetic spectrum. In order to do this, photometric bands are used. These bands allow only certain wavelengths to be detected.

If we study the flux density along the full electromagnetic spectrum, we would obtain the spectral energy distribution. The spectral energy distribution (SED) of a source that emits electromagnetic radiation is a graph of the energy emitted by the source as a function of the wavelength of the radiation. As the magnitudes encode flux information in a particular wavelength, these data can be used to know the SED of an astronomical source, adjusting the points to a curve given by a mathematical model.

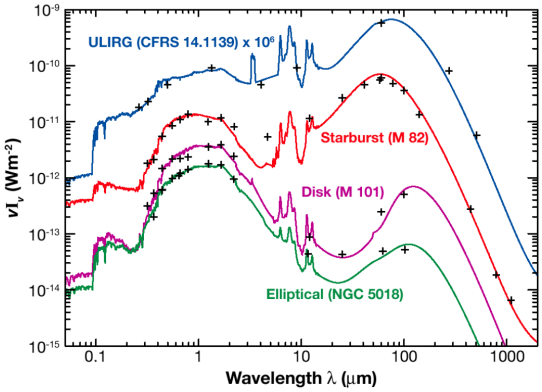
In Figure 1.1 we can see some examples of SEDs of different astronomical sources. These three examples are very different between them. This is because the mechanisms that produce radiation are different in each case. In the case of Figure 1.1a, we can visualize a typical SED for a star. Stars are optically thick all the way from the core up to the photosphere, where the light we see from them comes from. As the photosphere is in approximate thermal equilibrium with the radiation coming from the star's interior (and ultimately from the fusion reactions at its core), it emits nearly like a black body and its emission can be adjusted using a black body model [1].

In Figure 1.1b there are represented SEDs for different types of galaxy. The emission of a galaxy is given by the emission of the different objects that compose it. If we observe the figure, we can see that, for the three lower SEDs, there is a bump for $\lambda \sim 1\ \mu\mathrm{m}$ that comes from the radiation of normal stars, as the one represented in Figure 1.1a. For lower wavelengths ($\lambda \sim 0.1\ \mu\mathrm{m}$), there is another bump, coming from blue young stars. If we analyse the region where $\lambda \sim 10\ \mu\mathrm{m}$, there are some emission peaks that come from the emission of the dust particles in the galaxy. Finally, for higher wavelengths ($\lambda \sim 100\ \mu\mathrm{m}$) we can observe a thermal bump peaking. This thermal emission comes from the dust particles: the great amount of dust that is enshrouding the galaxy absorbs ultraviolet radiation emitted during the star formation and is re-emitted at infrared wavelengths. Depending on the galaxy we are analysing, we can find a greater or a lower bump. For example, in the figure we are studying we can see a starburst galaxy SED and an ULIRG SED. A starburst galaxy is a galaxy with an unusual high rate of forming stars (around 10^3 times greater than a common galaxy). An ULIRG (UltraLuminous InfraRed Galaxy), as its own name remarks, is a galaxy with unusual high emission at infrared wavelengths. It is a starburst galaxy with a very concentrated amount of dust, that produces a large amount of far-infrared radiation [5]. Therefore, this great amount of dust produces these bumps for far-infrared wavelengths.

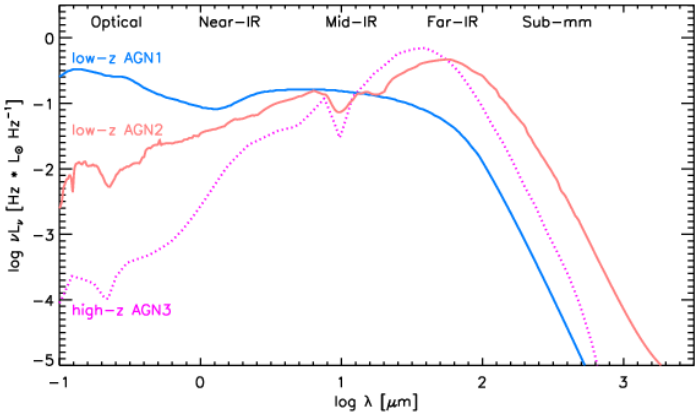
Finally, in Figure 1.1c we can see the SEDs of some AGNs. Some galaxies emit a lot of radiation from their nuclei: these regions are called Active Galactic Nuclei (AGN). The ultimate origin of this radiation is the liberation of potential energy from material falling into supermassive black holes (SMBH) at their centres. In unobscured AGN (AGN1, that corresponds to a quasar) this emerges



(a) Typical SED for a star. Extracted from [2].



(b) Typical SEDs for eliptical, disk, starburst and ULIRG galaxies. Extracted from [3].



(c) Typical SEDs for AGNs. The AGN 1 curve corresponds to a quasar. Extracted from [4].

FIGURE 1.1: Examples of SEDs for different astronomical structures

as a blue bump ($\lambda \sim$ 0.1-1 μm) from hot material near the SMBH and a second thermal bump ($\lambda \sim$ 1-100 μm) from cooler material further out. In obscured AGN (AGN2) the inner region is occulted from our direct line of sight and the emission is reddened and the cooler thermal bump is more prominent. Finally, some objects (AGN3) present even more absorption (redder spectra) and higher cooler thermal bumps [4].

Some sources may emit energy strongly in a specific region of the electromagnetic spectrum and be faint in other parts of it. Because of that, if we decide to study sources that are bright enough in a lot of different regions of the spectrum, we will probably ignore sources with interesting properties that do not emit in all the wavelengths analysed. As we said before in this chapter, we want to avoid bias. Because of that, we have decided to focus only on a limited region of the electromagnetic spectrum. Specifically, we have studied the near-infrared ($\lambda \sim$ 0.8-2.5 μm) and mid-infrared ($\lambda \sim$ 2.5-25 μm) regions. This part of the electromagnetic spectrum has been chosen because it has been found in previous works that this region is particularly good when used with this purpose. In the Figure 1.2 it can be seen how the differents celestial sources make groups when using some mid-infrared magnitudes to make colour indexes. For example, Carrasco et al. construct in [6] a quasar candidate catalogue, finding that the separation of quasars and stars performed when using optical and mid-infrared data is better than the one obtained using only optical data. In particular, for this project we have used the information provided by the 2MASS and WISE surveys. This will be discussed in detail in Chapter 2.

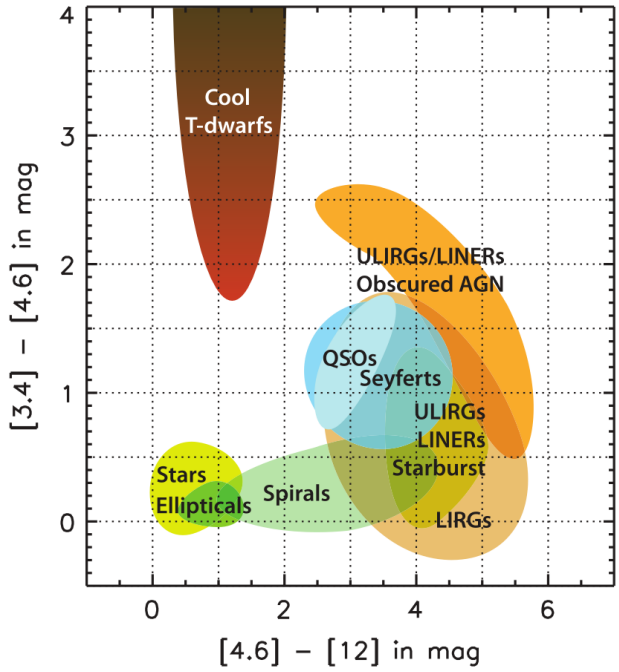


FIGURE 1.2: Separation of astronomical sources in groups using mid-infrared information. Figure extracted from [5].

One question may arise after this argumentation: why do we not analyse the lines of emission and absortion for each source? In other words, why do we not analyse their entire spectra? There are multiple reasons for not doing this. Obtaining the spectrum of a source is not easy: one instrument may analyse only the emission of a small region of the electromagnetic spectrum, and only for bright enough sources. We want to study sources that may be not that bright, and we want to analyse many of them, so taking spectra for all of them is not a viable option.

1.2 Automatic identification

Now, it is time to introduce the mathematical aspects of this project. When we are working with a low amount of data, the information encoded by them can be analysed manually, without needing sophisticated and refined scripts. However, in this work we are planning to work with more than 10^5 sources. Therefore, that approach will not be feasible: we will have to deal with data writing scripts in order to process them automatically. As we said at the beginning of this chapter, this kind of methods are becoming more common and necessary in Astronomy according to the technological progress. For example, in the future the Large Sypnotic Survey Telescope (LSST) will produce the deepest, widest image of the Universe, taking images using different wavelengths. This survey will produce around 15 Terabytes of data every night [7], so it is clear that those data cannot be handled as usual.

A field within data science that has become the trend in the last years is the Machine Learning. Machine Learning (ML) is the study of mathematical models created by a machine (i.e. a computer) to find patterns in a data set, in order to perform different tasks. Broadly speaking, ML methods take a training data set that is used to build the mathematical model. This mathematical model is later used to make predictions automatically by the computer [8]. This definition appears to fit perfectly in the current problems that astronomers are trying to solve nowadays, beyond trends.

Depending on the knowledge about the data set used to train the machine, ML methods can be classified into two groups: supervised ML and unsupervised ML. Supervised ML algorithms create

models that show relationships between some measurements and a target variable. In order to do this, a set consisted of selected examples is used. This set is usually called training set. After the machine creates the model using the training set, this model can be used to predict the target variable value for a different data set that do not have information about it. Somehow, their utility is similar to the traditional model fitting. However, the main difference lies in the fact that traditional model fitting uses a predefined model, while supervised ML methods create the model with the training data set. Some examples of supervised ML methods are Support Vector Machines, Random Forests and Artificial Neural Networks [9].

On the other hand, unsupervised ML algorithms create models using a data set with only measurements, without labels. Therefore, the machine tries to describe the data structure without the "additional information" given by the labels. Because of their nature, the output must be carefully analysed: each method have different variables that can change drastically the results obtained when they are modified. Some of the uses of this type of algorithms are finding clusters of data, reducing the dimensionality of the data space or finding anomalies in the data.

If we study the state of the art in the field of separating radiating sources using ML, we can find different studies. One example may be the Carrasco et al. article [6], cited before, where the authors create a catalogue of quasar candidates. In order to do this, they use Random Forests (a supervised ML method) using a training data set composed of stars and quasars, extracted from the SDSS (Sloan Digital Sky Survey) catalogue. The measurements they use to separate the two subsets are colour indices, obtained from optical, ultraviolet and mid-infrared magnitudes.

Another study that focuses in separating astronomical sources using ML is [10], by Kovács and Szapudi. They construct a training data set using sources that are in the SDSS catalogue and, taking magnitudes that analyse the near-infrared and mid-infrared from two different catalogues (the sames that we will use in this project, as we will explain in Chapter 2). The training data set is composed of stars and galaxies, and using Support Vector Machines (supervised ML) they find the best separation between the two groups.

As it can be seen, supervised ML is used in both examples with the objective of separating only two different groups. Furthermore, the training data set comes from a selection of the SDSS catalogue, that has optical information. Doing the separation that way produces bias: it will only work with sources that have a significant emission in the optical range. Our purpose is different and, somehow, more ambitious: we want to use unsupervised ML algorithms to separate different radiating sources, not only stars and galaxies.

1.3 Objectives

The objective of this project is to use unsupervised ML techniques in order to divide a problem data set, composed of near-infrared and mid-infrared magnitudes measurements, in an unespecified number of groups. These groups will be detected automatically by the computer and checked for correspondence to different types of astronomical sources. Specifically, we will use dimensionality reduction techniques to transform our initial data set and clustering models after these transformations. This is a new approach that involves a large amount of work because of the huge amount of data we pretend to use. We expect that using unsupervised ML techniques and using only a few infrared bands will reduce the bias that other methods have.

This thesis follows a classical structure. After this introductory chapter, in Chapter 2 we will explain the different methods and algorithms used to process and filter the data used. The work done can be divided in three important parts: processing and filtering the data, transforming it using dimensionality reduction techniques and finding clusters in the transformed data set. Because of this, the Chapter 2 is divided in three sections, each one concerning one of these aspects.

In Chapter 3, we will show the results obtained after applying the methods and algorithms to our initial data set. Similarly to Chapter 2, this chapter is divided in three sections that have the same name as the ones written in Chapter 2.

In Chapter 4, it can be found a discussion about the results obtained after applying all the methods explained. We will use a data set with labels (that is, a data set with classified sources according to their type) and will see if the models created are able to split this data set according to this classification. We will also try to split the clusters manually and compare this separation with the one performed by the computer.

Finally, in Chapter 5 we will end the thesis drawing conclusions about the work done, the results obtained and how it could be continued in the future.

There is also an appendix in this thesis (Appendix I) where there is a summary of all the scripts programmed and their function. The scripts can be found in the CD where this thesis has been recorded or in GitHub. The link is indicated in the appendix.

Chapter 2

Methodology

In this chapter, we are going to explain the methodology followed during the development of the project. This development has been extensive, and it can be divided in three overall parts. The first one focuses on the collection and the rendering of the data used. The second one concentrates on finding a new system of coordinates to reduce the dimensionality of the space where the original data live. Finally, in the last part, some clustering methods are applied to the data sets obtained in the second part of the project, in order to divide them into subgroups according to different criteria. In this chapter, the procedures concerning each of these parts are explained in detail.

2.1 Collecting and processing data

Before studying different ML techniques and their application to the problem under study, it is necessary to spend time talking about the data used and how it has been processed. Handling a huge amount of information and interpreting it is a crucial part before testing any ML method. Therefore, in this section, we will focus on this part of the project: all the downloaded data were processed and cleaned following different quality criteria to finally get a set with useful information, suitable to apply ML techniques.

The data used in this work come from a selection of sky regions widely studied in multiple wavelengths of the electromagnetic spectrum. We wanted to use only near-infrared and mid-infrared information, so it was decided to use the tables with photometric data provided by the 2MASS and WISE surveys. Using two different catalogues made it necessary to perform cross-correlations between them for unifying each source data. This technique and its theoretical basis will be explained in this section.

For the purpose of performing all these tasks, software such as Python, TOPCAT or STILTS has been used. The use made of this software will be explained when necessary. However, we will not go into details in these explanations because it is not the objective of this work; there are many manuals available with this purpose that attempt to explain the insights of the software used.

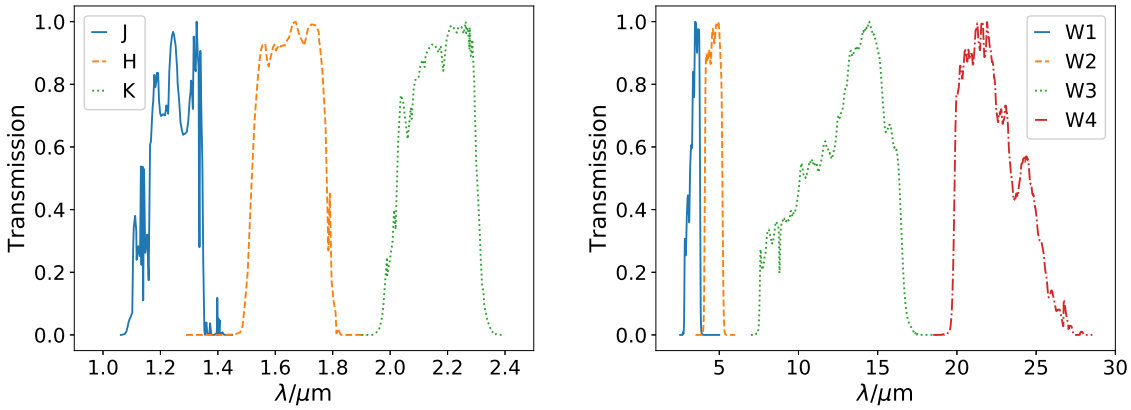
2.1.1 Catalogues used

In this project, we have worked with many data from multiple sources outside the Milky Way. The fields chosen avoid the Galactic plane because we are looking for extragalactic sources which are more easily detected outside the Galactic plane. Because of our interest in collecting data from objects equally distributed in the celestial sphere and using just their near-infrared and mid-infrared emission information, it was decided to use the data provided by the 2MASS and WISE surveys. The Two Micron All Sky Survey (2MASS) is a joint project of the University of Massachusetts and the Infrared Processing and Analysis Center/California Institute of Technology, funded by the National Aeronautics and Space Administration (NASA) and the National Science Foundation (NSF, USA)

[11]. Likewise, the Wide-field Infrared Survey Explorer (WISE) is a joint project of the University of California, Los Angeles, and the Jet Propulsion Laboratory/California Institute of Technology, also funded by the NASA [5].

The main reason for choosing these catalogues instead of other data sets is that the 2MASS and WISE surveys mapped more than the 99 % of the sky during their activity. At first sight this feature may appear unnecessary, but the decision was made to avoid bias: catalogues with more filters covering different wavelengths, and even previously cross-correlated data from more catalogues that analyse a small fraction of the celestial sphere, may focus only in one kind of source or in sources that emit light in more wavelengths (e.g. catalogues that store only sources that have emission in X-rays). Using those data may cause bias to arise. We want to work with a representative sample of the whole sky, just relying on the infrared data, and this can be carried out using these two catalogues.

The bands that the 2MASS survey analyses are the J band (centred at 1.25 μm), the H band (1.65 μm) and a slight modification of the K band (2.16 μm). In the same way, the WISE survey studies the $W1$, $W2$, $W3$ and $W4$ bands (centred at 3.35 μm, 4.60 μm, 11.6 μm and 22.1 μm respectively). The relative spectral response for these bands are represented in Figure 2.1.



(a) 2MASS (b) AllWISE

FIGURE 2.1: Relative spectral response for the 2MASS and AllWISE bands [12, 13]

The Two Micron All Sky Survey (2MASS) is a project that collected near-infrared data covering 99.998% of the celestial sphere. This survey produced a Point Source Catalog (PSC) with photometric and astrometric data for 470992970 sources [11]. This project was conceived as a ground-based survey. The atmosphere is transparent in a few windows in the near-infrared [1]. These windows correspond, approximately, to standard photometric bands, three of which have been chosen.

The Wide-field Infrared Survey (WISE) is a mission that studied over the 99% of the sky and collected mid-infrared data with higher sensitivity than previous infrared surveys (e.g. IRAS) [5]. This survey, combined with the NEOWISE phase, was used to create a catalogue with the information collected, known as the AllWISE Source Catalog. The number of objects within this catalogue was around 747000000 [14]. Unlike the 2MASS survey, WISE is a space mission. This is because the atmosphere is mostly opaque over the mid-infrared and hence these observations have to be carried out using a space mission.

The structure of the tables provided by 2MASS and AllWISE is complex and is deeply analysed in [12, 14]. Still, it is important to note that the sources have been selected following the recommended criteria in the supplements: for each source and each band there is a quality flag that merges the signal-noise and the uncertainty information. We will discuss these flags again in Chapter 3 when we comment the data cleaning achieved.

2.1.2 Data processing

After the description of the surveys and the catalogues used, we are in a position to explain the procedure followed to download and process the data of interest. To handle this data, some Python scripts were written. The purpose for each one of them is explained in depth in Appendix I.

The starting point for this work was a list of 2654 positions on the celestial sphere (for their selection, see below). These positions would be the centres of the analysed fields. To retrieve the records from 2MASS and AllWISE catalogues, we performed several Simple Cone Searches (SCS). This method describes a cone in the sky using a position that acts as a centre and an angular distance that acts as a radius. The angular distance was chosen to be always $15'$. The locations of these fields were not selected randomly: all the selected regions have been previously studied and analysed focusing on other wavelengths in other projects. One example is the ARCHES project (Astronomical Resource Cross-matching for High Energy Studies), a project that studied large samples of objects from the 3XMM X-ray catalogue. This project also developed a cross-correlation tool that has been used in this work and that will be discussed later in this chapter [15]. The angular distance was not chosen randomly either: this is the radius of the XMM-Newton field of view. Our input list of sky coordinates comes from the XMM-Newton (The European Space Agency's, ESA, X-ray Multi-Mirror Mission) fields selected by the ARCHES project by manually screening XMM-Newton observations, and taking into account the availability of information in other wavelengths. We could take advantage of this in case we wanted to expand the wavelength range covered in further studies, for example. To download all the tables, we made use of IRSA's SCS service (InfraRed Science Archive), a functionality that allows to get access to the data via the Linux terminal [16].

It is worthwhile to comment on the image formats and the programs used to handle the tables. The image formats handled during the progress of this work are ASCII (American Standard Code for Information Interchange) and FITS (Flexible Image Transport System). The first one, ASCII, has the advantage of being easily readable for small tables and accessible with almost every text editor available; the second one, FITS format, provides lighter files, something very important when handling thousands of tables and the headings of the tables are still very accessible. FITS format is widely used in astronomy because of its utility when handling information of an image in multiple wavelengths [17]. However, in this project the main benefit found is the one related to the sizes of the files.

The programming language used in this work is Python. This language offers a large variety of libraries with applications in numerical analysis, graphical representation, data mining, etc. The main libraries used in this project are NumPy, Matplotlib and Astropy [18, 19, 20]. Apart from Python, the main software used to analyse the studied data is TOPCAT and STILTS. TOPCAT is a graphical editor for tabular data, and it offers a wide variety of ways of analysing the data, joining tables, 2D and 3D plotting, creating subsets, etc. STILTS is the command-line version of TOPCAT. Both are written in pure Java, and are optimized for handling tables with millions of rows [21, 22]. Both TOPCAT and STILTS are compatible with ASCII and FITS formats. The graphical user interface of TOPCAT is very intuitive, but in some cases STILTS is more useful (e.g. performing tasks with thousands of tables), so TOPCAT has been mostly used to perform tasks with a few tables.

All the data downloaded has been processed using several Python scripts. However, the complete description of the test performed to filter the data is explained in Chapter 3.

2.1.3 Cross-correlation: ARCHES xMatch

After processing the raw data and some filtering, it was time to execute the cross-correlations between the 2MASS and AllWISE catalogues. However, this process is not as simple as it may look at first sight (or, at least, performing it in a fully coherent manner).

There are multiple methods and tools for the astronomers to perform cross-matches. A first example can be found in part of the software used previously: STILTS and TOPCAT offer a cross-matching tool between multiple tables, based only on the distance between counterparts in different catalogues [23, 24]. However, this tool does not handle any probabilistic information, nor about the counterparts, neither about the matches, so the outcome of the process lacks rigour.

Another example is the CDS xMatch service. This tool offers the same functionality as TOPCAT, but you can choose the option of using positional errors, in order to make use of statistical arguments in the cross-match [25]. Nevertheless, they do not provide the probability of real association after the matches, and they can give multiple counterparts for each source in one catalogue, so it is difficult to discern between the best match and the rest.

Some rigorous methods have been carried out recently. For example, Budavary and Szalay developed an algorithm using Bayesian statistics that performed cross-identifications between multiple catalogues [26]. This method is easy to expand if one wants to add photometric considerations to the probabilities, e.g., if we knew what kinds of sources we are handling we could use its SEDs to improve the matching probabilities obtained. However, this is not of our interest: we want to perform cross-identification based only on astrometric data and not in photometric information. Doing this way, we avoid biasing our data and allowing to cross-match different kinds of sources.

In this project, it has been decided to use the ARCHES xMatch service[1]. This tool was developed during the ARCHES project, and it can handle several catalogues at the same time to perform cross-matches. It uses an algorithm with a strong statistical basis, similar to the one described by Budavary and Szalay in [26], and after performing the cross-matches the program gives the probability of association. The description of the method and the algorithm is widely explained by Pineau et al. in [27]. Still, the basic scheme of the algorithm will be now briefly described to understand better the internal process of the software.

For performing the cross-identifications between two different catalogues a thorough statistical analysis is needed. The main framework for computing the probabilities of real association knowing the distance between the counterparts will be the Bayesian statistics. This will be revealed over the course of this subsection. Before using Bayes's theorem, there will be performed some classical hypothesis tests. Although we are going to focus on the cross-correlation of two different catalogues, the implemented method can be generalized to more than two catalogues without much difficulty. The general case is the one discussed by Pineau et al. in [27].

In order to work correctly, the program needs the positions to be accurate, i.e., there is not any offset between the positions in the two catalogues. We will also consider that the positional errors are trustworthy. It makes sense to accept these assumptions because astrometric information from 2MASS and AllWISE catalogues was calibrated using the Tycho-2 catalogue [28]. Last but not least, we will suppose that the local density of sources is constant. This last assumption will be important when performing the cross-matches in our data set.

The first step in the algorithm is finding sets of objects from both catalogues that are candidates to be detections of the same real source. In order to achieve this, the algorithm will perform several hypothesis tests. For each pair of detections from both catalogues, the null hypothesis will be H_0: "Both detections come from the same real object", and the alternative hypothesis H_1: "The detections come from different real sources". To know if we reject the null hypothesis, we have to choose a value for the type I error (that is, the probability of rejecting H_0 when it is true), α, calculate a Mahalanobis distance (i.e. a distance normalised to the combined positional error taking into account their correlations) for each detection, and make use of the χ^2_2 cumulative distribution function (all the details can be found in [27]). After those calculations, a threshold k will be found. Therefore, the candidate selection criterion will be "$x \leq k$", where x is the Mahalanobis distance obtained before.

[1] Accesible at `http://serendib.unistra.fr/ARCHESWebService/index.html`.

If the criterion is not satisfied, the null hypothesis H_0 will be rejected.

After performing the hypothesis test, the algorithm focuses on calculating the probability of associating two detections from the same real source for each candidate selected. To do this, it is necessary to use Bayesian statistics. This can be easily checked: if we call x to the Mahalanobis distance calculated in the hypothesis test and S to the positive result of the candidate selection criterion "$x \leq k$", we can calculate the probability that each hypothesis H_0, H_1 is true when knowing x and S. As one of both hypothesis has to be true for each pair, and they are disjoint, Bayes's theorem can be applied:

$$p(H_i|x,S) = \frac{p(H_i|S)p(x|H_i,S)}{p(H_0|S)p(x|H_0,S) + p(H_1|S)p(x|H_1,S)} \tag{2.1}$$

Each of the terms that appear in Eq. (2.1) is deeply analysed in [27]. Here, we will only write the results that will be used explicitly later. To study the priors $p(H_i|S)$, it can be calculated the mean surface area of all positional error ellipses in each catalogue, $\overline{\Omega}_j$ for $j = 1, 2$ referring to the catalogue, using the covariance matrices of the measurements. Writing Ω to refer to the area covered by the field analysed, when $\overline{\Omega}_1 + \overline{\Omega}_2 \ll \Omega$ the number of spurious associations $\hat{n}_{spur}$ can be estimated as

$$\hat{n}_{spur} = n_1 n_2 k^2 \frac{\overline{\Omega}_1 + \overline{\Omega}_2}{\Omega} \tag{2.2}$$

where n_1 and n_2 are the number of sources in each of the catalogues used. If we know the number of total associations after performing the test, n_T, an estimation of the priors can be made:

$$\begin{aligned} p(H_0|S) &= 1 - \frac{\hat{n}_{spur}}{n_T} \\ p(H_1|S) &= \frac{\hat{n}_{spur}}{n_T} \end{aligned} \tag{2.3}$$

The other terms that have to be studied are the ones with the form $p(x|H_i, S)$. In the case of two catalogues, the probability distribution for each $i = 0, 1$ is

$$\begin{aligned} p(x|H_0,S)dx &= \frac{1}{1-\alpha}\chi_2(x)dx \\ p(x|H_1,S)dx &= \frac{2}{k^2}xdx \end{aligned} \tag{2.4}$$

2.2 Dimensionality reduction techniques

It is important not to forget the main objective in this work: we want to separate the data into different groups using unsupervised ML. One possibility could be to use the raw data to find the clusters there. However, trying to find clusters using the raw data may be difficult: there may be noise that complicates the task, the geometry followed by the data set may be complex, etc. For this reason, in this section we seek to make the group separation more noticeable. To make this possible, we will use dimensionality reduction techniques. These methods fall within the area of unsupervised ML: the raw data are provided to the computer and it transforms the coordinates automatically, without using labels to differentiate the data. As its name suggests, the common scheme that all of these methods follow is transforming the initial space in one of lower dimension, trying to lose as little information as possible. This transformation is beneficial because using too many dimensions may arise in blurring the clusters and losing the data structure that we want to detect.

The main reference followed to select the dimensionality reduction methods and its explanation is [29]. Two different methods have been selected: Principal Component Analysis (PCA) and Locally Linear Embedding (LLE). In the first method a lineal transformation is performed, trying to find the

maximum variance directions. LLE is a Manifold Learning method. The models within this group try to find geometric properties in the data structure and describe them with a smooth manifold model that is embedded in a higher dimension space.

Apart from all the tools mentioned in the previous section, used to collect and process the data, here we have used the AstroML and Scikit-learn libraries in Python [30, 31]. Specifically, the Scikit-learn library contains all the ML methods that have been applied.

2.2.1 Principal Component Analysis (PCA)

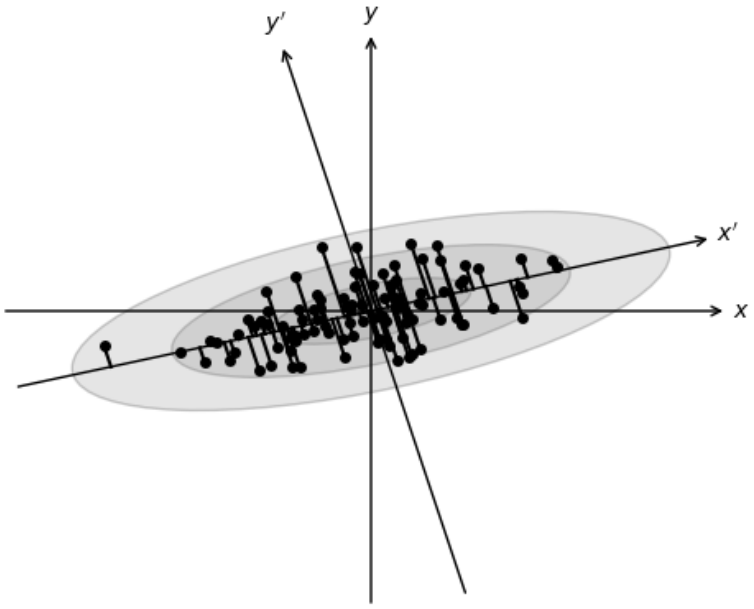


FIGURE 2.2: Example data set for PCA. Figure extracted from [30].

To understand our decision to use this technique, we can look at Figure 2.2. Here, we have a hypothetical data set that is distributed following a bivariate normal distribution. The two variables that come into play are not independent: as we can see in the figure, the y variable grows as x grows. However, if we rotated the axes, performing a change of coordinates from x and y to x' and y', the correlation would disappear. Furthermore, the x' axis would hold the maximum variance in the example data set. Making a rotation or a translation does not affect to the relative position between the points, so the data structure would not be lost.

This argumentation is the basis of the Principal Component Analysis (PCA) method. It consists in finding the directions of maximum variance for the problem data set. In order to achieve this, the coordinate space is changed using an orthogonal transformation. After this change of coordinates and ordering the new variables based on the variance amount that they explain, the components with the least variance associated may be ignored if the percentage of the total variance that they explain is minimal. In this case, those components may be associated only with noise.

The references consulted to explain the theoretical basis of this method are [29] and [32]. All the mathematical calculations have been redone during this work to prove rigorously, from a statistical and optimization perspective, that the directions of maximal variance are the ones that diagonalises the covariance matrix.

The starting point of this method is a set of d-dimensional measurements $\{X^{(i)}\}_{i=1}^{n}$. Each of these measurements can be written as a column vector $X^{(i)} = (X_1^{(i)}, X_2^{(i)}, \ldots, X_d^{(i)})^T$ for $i = 1, \ldots, n$,

that is, we are studying d one-dimensional observables. We have defined each $X^{(i)}$ as a column vector, so this set of measurements can be written as a matrix $X = \left(X^{(1)}, X^{(2)}, \ldots, X^{(n)}\right)$ of dimension $d \times n$.

To simplify the following mathematical procedure, it will be supposed that the measurements are centred, that is, the sample mean of each observable X_j is zero (if that is not the case, the data can easily be centred substracting to each measurement of each observable its sample mean). In this situation, it is easy to calculate the covariance matrix Σ_X: each entry Σ_{Xij} of the matrix ($1 \leq i, j \leq d$), when working with centred data, is given by

$$\Sigma_{Xij} = \frac{1}{n-1} \sum_{k=1}^{n} X_i^{(k)} X_j^{(k)} \tag{2.5}$$

When one analyses this expression carefully, it can be seen that the covariance matrix can be written as a matrix product using X and its transposed X^T:

$$\Sigma_X = \frac{1}{n-1} X X^T \tag{2.6}$$

The objective of PCA is to find an orthogonal transformation, represented by the $d \times d$ matrix P, that converts the measurements X into a different set of observables $Y = PX$. The main characteristic of Y is that each of its observables should maximize the variance. We will now see the peculiarity of this coordinate change. As P is an ortogonal transformation, Y will be composed of centred measurements, so Eq. (2.6) states that its covariance matrix Σ_Y can be written as

$$\Sigma_Y = \frac{1}{n-1} Y Y^T = \frac{1}{n-1} P X X^T P^T = P \Sigma_X P^T \tag{2.7}$$

Denoting the first row of P as p_1, our objective now is to maximize the variance of the observable Y_1. This variance is the entry in the first row and first column of Σ_Y, so we want to maximize the function $f(p_1^T) = p_1 \Sigma_X p_1^T$. We want P to be an orthogonal map ($PP^T = P^T P = I$), so there is a constraint for p_1 that can be written as $p_1 p_1^T = 1$. Now we have an optimization problem, and the Lagrangian function associated to it is

$$\mathcal{L}_1(p_1^T, \lambda_1) = p_1 \Sigma_X p_1^T - \lambda_1 (p_1 p_1^T - 1) \tag{2.8}$$

The gradient of the Lagrangian with respect to p_1^T must be equal to zero; this leads to

$$\nabla_{p_1^T} \mathcal{L}_1(p_1^T, \lambda_1) = 2\Sigma_X p_1^T - 2\lambda_1 p_1^T = 2(\Sigma_X - \lambda_1 I) p_1^T = 0 \tag{2.9}$$

Therefore, λ_1 is a root of the equation $\det(\Sigma_X - \lambda_1 I) = 0$, that is, λ_1 is an eigenvalue of Σ_X and p_1 its eigenvector. As we want to maximize $f(p_1^T) = p_1 \Sigma_X p_1^T$, λ_1 must be the largest eigenvalue of Σ_X.

The derivation for each of the next rows of P, p_i for $i = 2, \ldots, d$, is analogous to the reasoning followed for p_1 adding the constraints that the principal components must be uncorrelated, i.e., $p_j \Sigma_X p_i^T = 0$ for $j < i$. Our objective is the same: maximize $f(p_i^T) = p_i \Sigma_X p_i^T$ with the constraints $p_i p_i^T = 1$ and the new ones. In this situation, writing $\delta = (\delta_1, \delta_2, \ldots, \delta_{i-1})^T$ Eq. (2.8) is transformed to

$$\mathcal{L}_i(p_i^T, \lambda_i, \delta) = p_i \Sigma_X p_i^T - \lambda_i (p_i p_i^T - 1) - \sum_{j=1}^{i-1} \delta_j p_j \Sigma_X p_i^T \tag{2.10}$$

Calculating again the gradient with respect to p_i^T leads us to

$$\begin{aligned}\nabla_{p_i^T}\mathcal{L}_i(p_i^T,\lambda_i,\delta) &= 2\Sigma_X p_i^T - 2\lambda_i p_i^T - \sum_{j=1}^{i-1}\delta_j\left(p_j\Sigma_X\right)^T = 2\Sigma_X p_i^T - 2\lambda_i p_i^T - \sum_{j=1}^{i-1}\delta_j\Sigma_X p_j^T \\ &= 2\left(\Sigma_X - \lambda_i I\right)p_i^T - \sum_{j=1}^{i-1}\delta_j\lambda_j p_j^T = 0\end{aligned} \tag{2.11}$$

If we also ask the principal components p_j to be linearly independent, Eq. (2.11) is only true when $\delta_j\lambda_j = 0$ for $j < i$, that is, $\delta = 0$ (as a vector). As in the case of p_1, this equation is true when λ_i is an eigenvalue of Σ_X and p_i its eigenvector. We can choose the largest eigenvalue for λ_i that has not been chosen yet.

In a nutshell: the PCA method consists in diagonalising the covariance matrix for a set of measurements. We can order the eigenvalues from largest to lowest and their respective eigenvectors. This eigenvectors are the ones that transform the initial measurements X into $Y = PX$. This diagonalisation can always be performed in a covariance matrix because it is positive semi-definite. Also, as it is a symmetric matrix, its eigenvectors will be orthogonal, that is, $p_i p_j^T = 0$ for $i \neq j$. Taking into account this property along with the restrictions applied before, that means that $p_i p_j^T = \delta_{ij}$ where δ_{ij} denotes the Krönecker delta.

After computing the matrix P, it is easy to transform X in Y and vice versa. Assuming that the original data are not centred, and writing $\mu = (\mu_1, \mu_2, \ldots, \mu_d)^T$ to refer to the sample mean of the data set, each measurement $X^{(k)}$ is transformed to its principal components using $Y = P(X - \mu)$ or, transforming explicitly each observable $Y_i^{(k)}$,

$$Y_i^{(k)} = \sum_{j=1}^{d} p_{ij}(X_j^{(k)} - \mu_j) \tag{2.12}$$

Since the matrix P is orthogonal ($P^T = P^{-1}$), it is easy to reconstruct the measurements $X^{(k)}$ using its principal components:

$$X_j^{(k)} = \mu_j + \sum_{i=1}^{d} p_{ij} Y_i^{(k)} \tag{2.13}$$

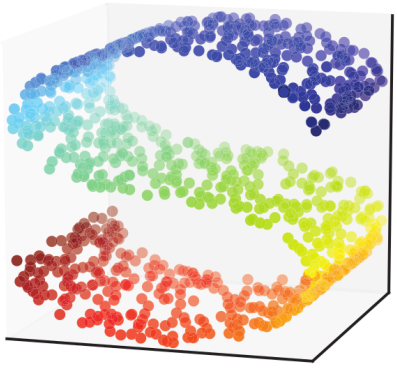
The PCA model is used to reduce the dimensionality of a set of measurements. In order to do this, the variance of each principal component must be analysed. The method most frequently used to choose the optimal number of principal components consists in analysing the cumulative variance of these components. The sum of all the eigenvalues λ_i is the total variance encoded in the eigenvectors. These eigenvalues are ordered from highest to lowest variance. We can choose a threshold $\alpha \in (0, 1]$ that represents the portion of variance that we want to take in account when choosing the components of interest. After choosing α, it is easy to find a criteria to choose the number of principal components to use: taking in account that the variance values are ordered, we will use D components from the d available if

$$\frac{\sum_{i=1}^{D}\lambda_i}{\sum_{i=1}^{d}\lambda_i} \geq \alpha \qquad \text{and} \qquad \frac{\sum_{i=1}^{D-1}\lambda_i}{\sum_{i=1}^{d}\lambda_i} < \alpha \tag{2.14}$$

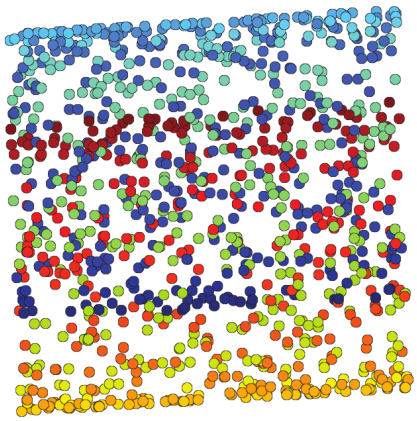
2.2.2 Manifold learning: Locally-linear embedding (LLE)

PCA is a powerful and easily interpretable method that can reduce the number of dimensions used to analyse data using orthogonal transformations. However, sometimes the data set is not that easy

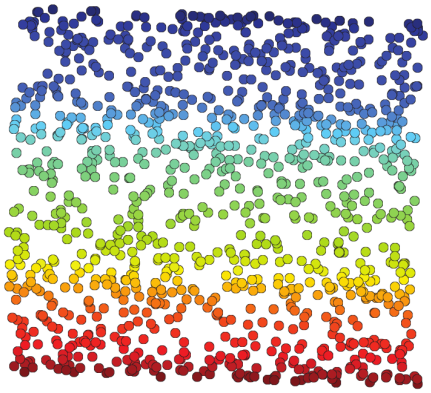
to transform, and linear coordinate changes are not enough. A classical example to show this is the S-curve data set. This set of points is shown in Figure 2.3a. In this case, the points are embedded in a three-dimensional space, but they have a smooth manifold structure that has a shape of an S letter. This geometric structure is a two-dimensional manifold embedded in a three-dimensional space. Using local coordinates, two coordinates should be enough to describe the data. However, if we used PCA in order to reduce the dimensionality, we obtain the results shown in Figure 2.3b: the points are mixed and the geometric structure is lost.



(a) S-curve data set



(b) PCA projection



(c) LLE projection

FIGURE 2.3: Example data set for LLE. Figure extracted from [30].

Because of this reason, it is interesting to use Manifold Learning techniques. These models perform non-lineal transformations for each set of coordinates. In this case, Figure 2.3c shows the performance of one of these methods acting over the S-curve data set. As it can be seen, the S-shape of the manifold is unwrapped and projected into a two-dimensional space.

The method that has been chosen to use in this project is the Locally Linear Embedding (LLE). This method makes use of local geometric information to explain the data structure. The main references consulted to explain its theoretical basis are [29] and [33]. In this case, the calculations are not as explicit as in the case of PCA method. The reason of this is that it is not necessary to derive as many mathematical expressions: it is enough to understand two similar expressions.

The LLE method is based on the relationship of each data point with its nearest neighbours. Its objective is to detect geometric structures based in local information, that is to say, the information

perceived within the vicinity of each point. The number of neighbours used to obtain information will be denoted as k.

The starting point for this method, as in the PCA algorithm, is a set of n d-dimensional measurements $\{X^{(i)}\}_{i=1}^{n}$. Each of these measurements have information for d different observables. This algorithm has two distinct parts. First of all, we seek a $n \times n$ weight matrix W such that its entries W_{ij} minimize the reconstruction error given by

$$\epsilon_1(W) = \sum_{i=1}^{n} \left| X^{(i)} - \sum_{j=1}^{n} W_{ij} X^{(j)} \right|^2 \tag{2.15}$$

Of course, W is not allowed to be any matrix (in that case, taking $W = I$ the identity matrix would trivially solve this problem). There are some important restrictions. First of all, $W_{ii} = 0$ for each $i = 1, 2, \ldots, n$. Second of all, for each point $X^{(i)}$ the matrix will only take in account the information given by the k nearest neighbours. This means that $W_{ij} = 0$ if $X^{(j)}$ is not one of the k nearest neighbours of $X^{(i)}$. Lastly, for each i the sum of the weights must be equal to one, that is, $\sum_{j=1}^{n} W_{ij} = 1$. The key restriction is the one that forces $W_{ij} = 0$ except for the nearest neighbours. Doing it that way, the matrix W will encode the local information for each point and, as a result, the geometric information of the original data set. This information will be invariant under translations, symmetries, rotations, etc.

The second part of the LLE method is similar to the first one, but it differs subtly from it. At this point, we will suppose that the data lie in a smooth manifold of dimension $D < d$. In this case, a set of D local coordinates should be enough to locate a point within the manifold. As the matrix W reflects the geometric properties of the original data set, this matrix should be enough to reconstruct the D-dimensional information (that is, the local coordinates) for each point in the original data set. Our objective is to find a set of D-dimensional vectors $\{Y^{(i)}\}_{i=1}^{n}$ that are obtained using the original data set $\{X^{(i)}\}_{i=1}^{n}$. These vectors must encode the local geometry of the manifold. The way to find them is minimizing the embedding cost function

$$\epsilon_2(Y) = \sum_{i=1}^{n} \left| Y^{(i)} - \sum_{j=1}^{n} W_{ij} Y^{(j)} \right|^2 \tag{2.16}$$

It can be seen that Eq. (2.16) is very similar to Eq. (2.15). However, the main difference is that ϵ_2 depends on the vectors $Y^{(i)}$ while ϵ_1 depends on the matrix W.

In summary, the LLE method main parts consist on minimizing two functions using a least squares method. There are two questions that have not been solved in this explanation, but are crucial: how do we choose the number of neighbours k to obtain optimal results? What about the dimension of the manifold D? These questions are not easy to answer. The way that is usually followed to choose a value for k and D is the empirical one: the algorithm is executed for several values of k and the results are compared with each other. Graphically, it can be seen that, for small values of k, the geometric structure of the manifold is lost. Usually, an optimal range of values for k can be found. The values within this range provide similar results when executing the algorithm, so one of them can be chosen to act as the number of neighbours. The value for D has to be lower than the coordinate space where the manifold is embedded, and it is usually chosen to be 1, 2 or 3.

2.3 Clustering

In this context, the word "clustering" refers to finding data structure and distinguish groups of points, called clusters, where the concentration increases. From a mathematical point of view, we

want to create a disjoint partition from a data set. This partition will satisfy some requirements. Of course, when the data set is not complex, the clusters can be found manually. However, when the data set contains too much data or the space where the points live is high-dimensional, the partition is not trivial. That is why we want this division to be found automatically by a computer using ML techniques. In the case of unsupervised ML, the division in subgroups will be made automatically, without giving prior information about the points within the data set.

Before studying the methods that will be applied, it is necessary to explain what we mean by "distance between two points". For a mathematician, defining a distance is not something trivial. If we take a topology book (for example, [34]) and search a definition of distance over a set X, it is defined as a function $d\colon X \times X \to \mathbb{R}$ that satisfies the following conditions for all $x, y, z \in X$:

- $d(x, y) \geq 0$, where $d(x, y) = 0 \iff x = y$
- $d(x, y) = d(y, x)$
- $d(x, z) \leq d(x, y) + d(y, z)$

For each set X, several functions can be defined and act as distances. Depending on the requirements in each situation, the best distance for each case can be chosen. In Physics, the most used distance when we are working in a real n-dimensional vectorial space $\mathbb{R}^n$ is the Euclidean distance: for two vectors $x, y \in \mathbb{R}^n$ with $x = (x_1, \ldots, x_n)$ and $y = (y_1, \ldots, y_n)$, the Euclidean distance d_2 between them is defined as

$$d_2(x, y) = \sqrt{\sum_{i=1}^{n} (x_i - y_i)^2} \tag{2.17}$$

As it has been said in the previous section, the clustering methods that are in this section will be applied to the data sets obtained with the dimensionality reduction models. The main references followed to select and understand the clustering methods that we have used are [29] and [9]. To explain the theoretical basis of the method, both [29] and [35] have been consulted. We have decided to use three different methods. The first one, called K-Means, seeks a partition that is performed based on minimizing a function. This function depends on the distance between the points within the data set and other points that act as the centres of the subgroups in the partition. The second method, Mean shift, makes use of a kernel density estimation of the data used. The last method is called Gaussian Mixture Model (GMM), and differs subtly from the other two methods: this is a distribution-based clustering method. These three models will be explained from a mathematical point of view in order to fully understand the differences between the methods followed by each model.

The tools used in this section are the same as in the previous one (Python and its libraries, TOPCAT and STILTS). Also, the mathematical insights of each method are explained in order to understand the key parts of each algorithm.

2.3.1 K-Means

The K-Means method is one of the simplests methods to do clustering. If we have a data set with n d-dimensional measurements, $\{X^{(i)}\}_{i=1}^{n}$, our objective is to find a disjoint partition of these data. Equivalently, this problem can be thought in an easier way: if we call $I = \{1, 2, \ldots, n\}$ to the indices set, our objective is to find a disjoint partition of I, $\{I_1, I_2, \ldots, I_K\}$, where the number K is the number of clusters to find, and it has to be chosen by the user. These sets have to satisfy $I_j \cap I_k = \emptyset$ for $j \neq k$ and $I_1 \cup I_2 \cup \ldots \cup I_n = I$. Each set I_j is equivalent to one of the clusters found when splitting the original set.

If we call $\mu_j = \frac{1}{N_j}\sum_{i\in I_j} X^{(i)}$ to the mean position in each of the subgroups, where N_j is the number of elements within each subgroup, the objective of the K-Means method is to minimize the sum of the squares distances from each measurement to the mean of their subgroup. Writing it mathematically, it can be seen as minimizing the following quantity:

$$\sum_{j=1}^{K}\sum_{i\in I_j} d_2(X^{(i)},\mu_j)^2 \tag{2.18}$$

In this case, we are using the Euclidean distance d_2 but any distance can be used.

There is a subtle detail that may not be noticed: the amount that has to be minimized depends on the position of the means. The means, in turn, depend on the elements within each subgroup. However, if we want to minimize the sum above, each point must be assigned to the subgroup whose mean is closest to the point. This looks like a snake biting its tail. To solve this problem, the initial positions of the means are randomly chosen, taking K points from the data set. After this choice, two steps are repeated until the amount in Eq. (2.18) does not change significantly. The first step is assigning each point in the data set to the subset that has its mean closest to the point. The second step is updating the mean of each subgroup using the new points within the subgroups. This method converges locally, but it does not necessarily converge globally. In order to find a global minimum, the method can be run several times, choosing different initial positions for the means. The partition with a lower value for the amount in Eq. (2.18) will be chosen.

There are several methods for choosing the optimal number of clusters, K. However, the most frequently used is a heuristic one, similar to the criterion used to choose the PCA number of useful components, known as the elbow method. This method consists in calculating the amount in Eq. (2.18) for different values of K and plotting it as a function of K. The optimal value for K will be the one where the graph has a pronounced angle. This will be better understood in Chapter 3, when we see those plots.

2.3.2 Mean shift

The Mean shift method differs from the K-Means method in its working philosophy. The previous method divided the original data set into disjoint subsets based on minimizing a function. This time, the algorithm concerning this subsection transforms each point in the problem data set using an iterative process. The points from the original data set are iteratively transformed, moving in the direction where the point density increases. In other words, this method seeks the density peaks in the data set.

In order to estimate the probability distribution function in the space where the points live, it is necessary to perform a kernel density estimate. This estimation is made overlapping different density functions centred in each point in the data set. The kernel function, named K, can take different forms and the performance of the estimate will be different depending on the choice. The kernel function that is used by the Python algorithm is a flat function [35]. This function (ignoring the normalizing constant), when centred on a point x_0, is defined for each x as

$$K_{x_0}(x) = \begin{cases} 1 & \text{when} \quad d_2(x_0,x) \leq h \\ 0 & \text{when} \quad d_2(x_0,x) > h \end{cases} \tag{2.19}$$

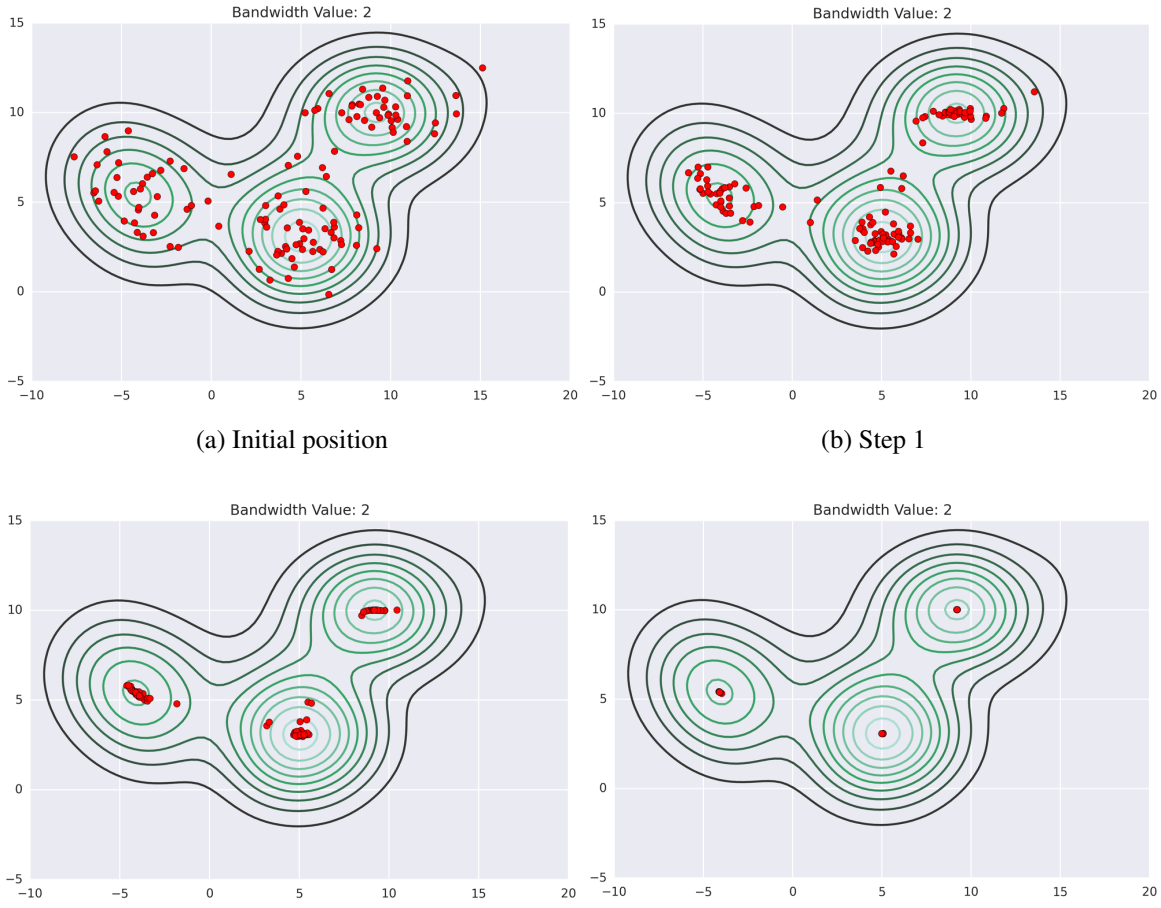
where h is called the bandwidth, and it is an argument that can be given by the user of the algorithm or estimated automatically with a different method that will not be described here. As it was said for the K-Means method, another function different from the Euclidean distance can be used.

The starting point for this algorithm is, as in the K-Means case, a set of n d-dimensional measurements $\{X^{(i)}\}_{i=1}^{n}$. These points will be modified in order to find the cluster centres, moving in the

direction where the gradient of the density increases more rapidly. The iterative process is defined as follows: if we call $X_t^{(i)}$ to the position of the ith measurement during iteration t, we want to update the position of this point to $X_{t+1}^{(i)}$. In order to do this, we write K_j to denote the kernel function centred in the position $X^{(j)}$. In this situation, the new position will be computed as follows:

$$X_{t+1}^{(i)} = \frac{\sum_{j=1}^{n} K_j(X_t^{(i)}) X^{(j)}}{\sum_{j=1}^{n} K_j(X_t^{(i)})} \tag{2.20}$$

The difference $X_{t+1}^{(i)} - X_t^{(i)}$ is usually called mean-shift. From the Eqs. (2.19) and (2.20) can be deduced that, when using a flat function as kernel, the updating of $X_t^{(i)}$ turns out to be a mean of the points in the original data set such that $d_2(X_t^{(i)}, X^{(j)}) \leq h$. Repeating this step several times, the points will find the centroids of the clusters. These centroids will be some local maxima in the density distribution estimated.



(a) Initial position (b) Step 1

(c) Step 2 (d) Step 3

FIGURE 2.4: Mean shift transformation for a set of points. The algorithm detects three centroids. Figure extracted from [36].

In Figure 2.4 we show how a set of points is transformed iteratively, in order to find the centroids of the clusters. In this figure, it can be seen that, unlike the K-Means method, in the Mean shift algorithm, it is not possible to choose the number of clusters you want to detect. The number of clusters detected will depend on the bandwidth h chosen. Also, the algorithm is guaranteed to converge, and it will stop when the centroids of the clusters stop moving significantly.

2.3.3 Gaussian Mixture Model (GMM)

Both clustering methods studied until now, K-Means and Mean shift, have focused on finding the position of the centroids, using the information given by the measurements. Each one has used a different approximation to the solution of the proposed problem. However, a different philosophy underlies the method concerning this subsection: we are going to fit the data set using just a few density functions that act as kernels. These methods are known as mixture models. The most used kernels are Gaussian functions. This method is known as Gaussian Mixture Model (GMM).

The GMM method objective is to fit a problem data set $\{X^{(i)}\}_{i=1}^{n}$ using a finite number of Gaussian kernels. If we call K to the number of Gaussians used to fit the model, the form that the density function takes in this situation is

$$\rho(X) = n \sum_{j=1}^{K} C_j \mathcal{N}(\mu_j, \Sigma_j) \tag{2.21}$$

Therefore, each of these Gaussians will be characterized by its mean μ_j, its covariance matrix Σ_j and the constant α_j. A constraint can be added to the constants α_j in order to normalize the probability distribution function. The number or kernels to use, K, is a parameter that is given by the user. Therefore, we should adjust $3K-1$ free parameters, that is, find their maximum likelihood estimators. The way to do this is using an expectation-maximization algorithm. Simply stated, this algorithm is an iterative method that starts with random values for the estimators. After this, two steps are repeated until convergence (the convergence is guaranteed for this method). The first step consists in computing, for each point, the probability of being generated by the probability function. Then, the parameters are tweaked in order to maximize the likelihood of the data [35]. After this fit, each measurement is assigned to the kernel that has a higher probability.

As it was said before, the number of Gaussians used to fit the data to the model is chosen before running the algorithm. A natural question, analogue to the one asked for the K-Means method, arises: which value of K is the optimal for our data set? The answer to this question comes with two different values: the Akaike Information Criterion (AIC) and the Bayesian Information Criterion (BIC). These concepts quantify how well a model represents the experimental data, taking into account the complexity of the model to avoid overfitting. The Python object offered by sk-learn to perform GMM has two methods that compute AIC and BIC, so the mathematical definition is not relevant in this case: the most important thing that we should know is that the model with a lower value for AIC and BIC offers usually better results.

Chapter 3

Results

In this chapter, we show the tests performed in order to clean the initial data and the results obtained after performing the different ML algorithms. As in Chapter 2, this chapter is divided in three sections. In the first part, we will show the methods followed to filter the initial data and the results obtained after the cross-correlation. In the second section, we will represent the data in the new systems of coordinates and we will decide which coordinates are useful and which ones we can ignore. In the last section, we will show the division in subgroups of the transformed data, both for the PCA and LLE transformations. The divisions will be done using the three methods explained in Chapter 2.

3.1 Collecting and processing data

The positions of the centres of the regions analysed are represented in Figure 3.1. In this figure, the rounded strip of the sky with no centres correspond to the Galactic plane. It can be verified that there are isolated centres and others that form a bigger cluster. This fact will be important in the implementation of the problem, as it can be seen in Appendix I.

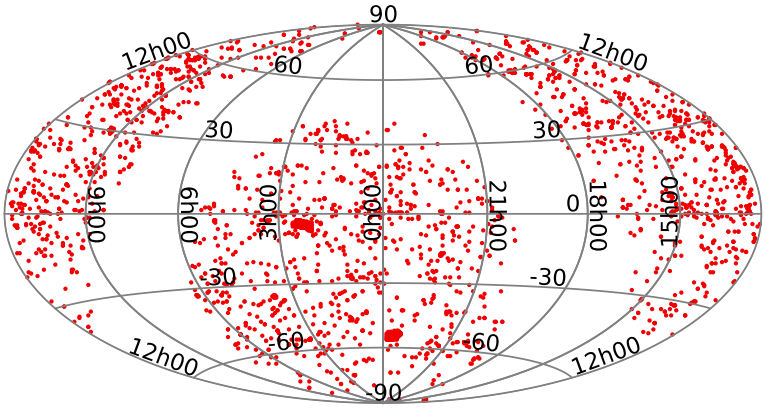
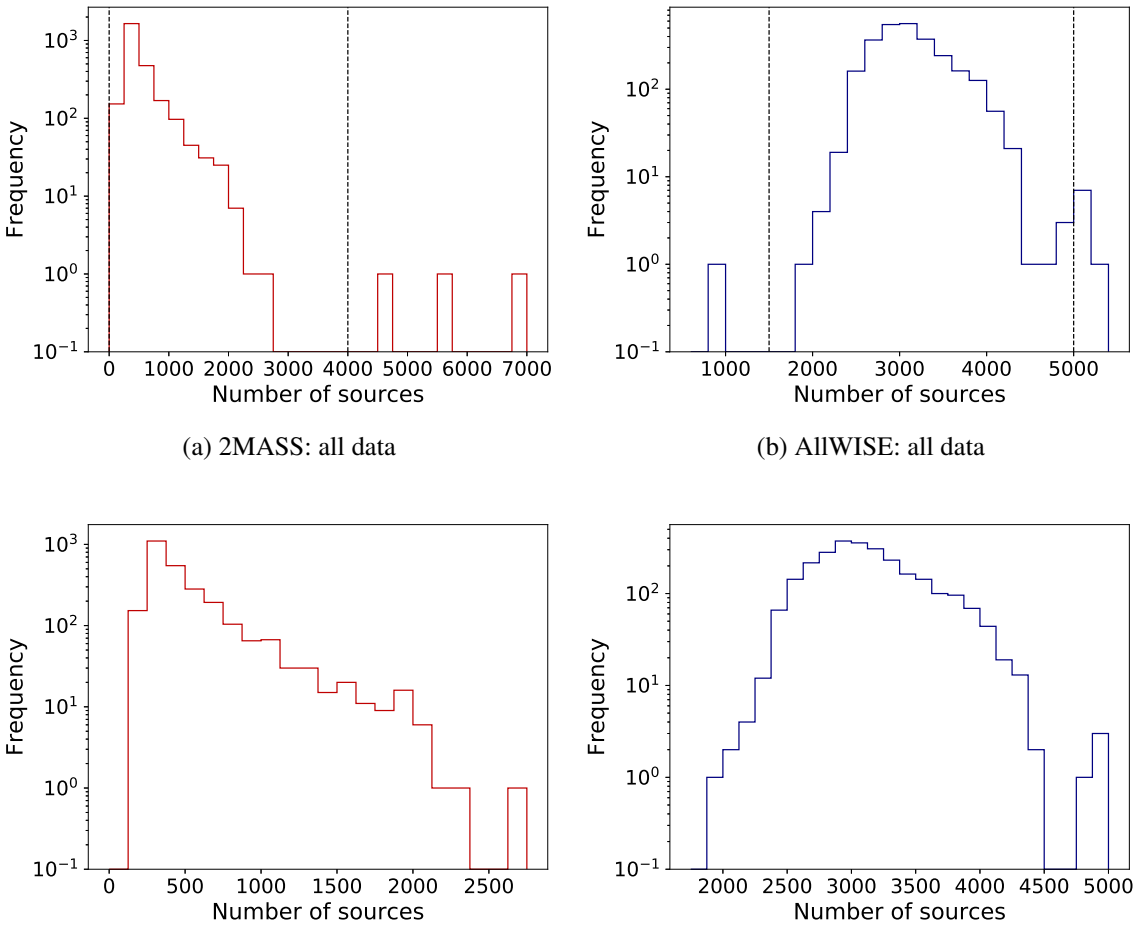


FIGURE 3.1: Positions of the SCS centres on the celestial sphere in equatorial coordinates

3.1.1 Data processing

After downloading the tables via the IRSA's SCS query, it was checked if there were any 2MASS or AllWISE table with an unusual number of sources. This could indicate a problem with the corresponding observations, and if this is the case, those fields need to be filtered out. To test this, we plotted the first two histograms in the Figure 3.2 (the frequency axis is in logarithmic scale). We can

see in the Figures 3.2a and 3.2b that, in the case of the 2MASS tables, there are three fields that have an unusually high number of sources; and for the AllWISE tables some fields are too far from the main part of the distribution. Some of these tables may correspond to deep fields (sky images with long exposure time) or to regions with very bright sources that saturate the image. Indeed, the field corresponding to the AllWISE table with less than 1000 sources detected was a sky region with a very bright source blocking the emission of the rest of the objects. Therefore, in order to avoid these tables and clean the data, it was decided to remove these fields from the original data set. We can see the clean histograms in the Figures 3.2c and 3.2d. After doing this, the number of fields handled decreased from 2654 to 2643.



(a) 2MASS: all data

(b) AllWISE: all data

(c) 2MASS: selected data

(d) AllWISE: selected data

FIGURE 3.2: Histograms plotting the number of sources in each field. The top ones take account of all the tables, the bottom ones plot the subset of fields that fulfil the chosen filtering criteria.

This cleaning was not the only one performed. After it, we made a second one according to the source density in each field for 2MASS and AllWISE tables and the correlation between these two densities. We expect the 2MASS and AllWISE densitites to be correlated. In the cases they are not, again, this indicates a possible problem with those observations. Before this filtering, it was necessary to see which fields overlapped. To do this, we grouped the centres of the fields based on the distance between them: if two points were located within a distance of $30'$, the fields associated with these centres overlapped. After grouping the fields using this distance argument, we calculated the sky area for each group and joined the 2MASS and AllWISE tables using STILTS. It is worth noting that some of the rows of the new tables were repeated because, due to the overlapping, a source could be at the same time in two of the initial tables. Therefore, it was necessary to erase these repeated rows.

Although it was clear what to do, it was necessary to write more than one script to achieve it. Those scripts are referenced in Appendix I.

Once we computed the number of sources in each sky region and its area, it was easy to obtain the 2MASS and AllWISE density of sources for each field and ascertain that they were correlated. Figure 3.3 shows this relation between both densities for almost all fields: in general, AllWISE density grows as 2MASS density increases. Still, some of the sky regions are separate from the cluster. That is the case of the five regions with the lowest AllWISE density, marked in the Figure 3.3. These five fields were inspected visually and, in each one, there was a very bright source that saturated the image, causing that many sources were not detected in AllWISE catalogue. The image of one of these fields and the effect produced in AllWISE catalogue is shown in Figure 3.4: there are some parts of the image where there is not any source detected by the WISE survey. Because of this effect, these five fields were discarded.

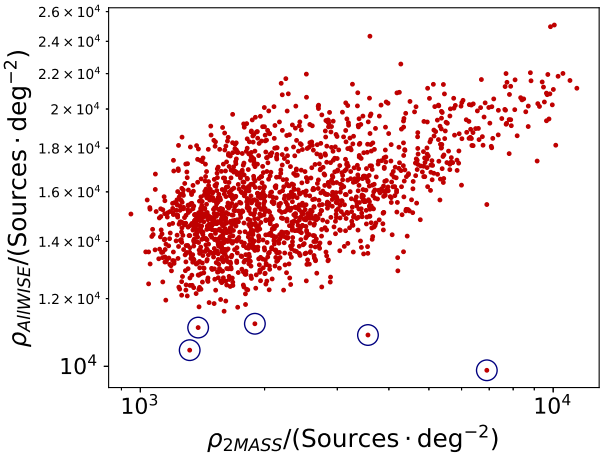


FIGURE 3.3: Plot that shows the correlation between 2MASS and AllWISE densities

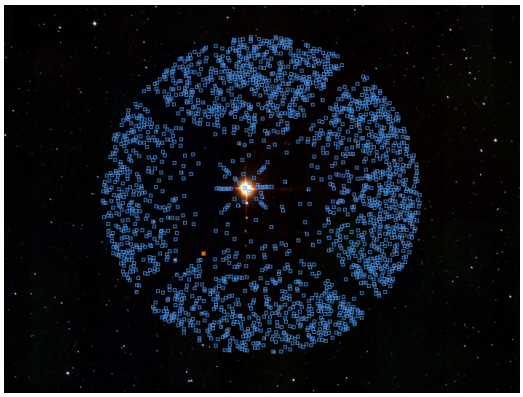


FIGURE 3.4: AllWISE sources detected in one of the problematic fields

After all this processing, there were two important data sets: one, corresponding to 2MASS survey, consisted of 982125 different sources; the other one, from the AllWISE catalogue, had 6305563 different rows of data. These data were obtained from 1677 different regions of the sky with different sky areas.

3.1.2 Cross-correlation

As a consequence of the theoretical basis of the algorithm described before, all the information used was divided in three big groups. This division was made because of the initial assumption that the source density was approximately constant for the data used. Because of this, all the available data were divided into the three groups shown in the Figure 3.5. The division was made according to the 2MASS density value, and the limits of the groups shown in the figure are located at 2200 and 4000 in 2MASS density. This division was also useful in terms of optimizing the task: the ARCHES xMatch service was not able to work with too heavy files and the time of execution had to be less than 10 minutes, so it was recommended to slice the work in small parts. Because of this reason, the columns with unimportant information for our project were also removed, in order to lighten the files. The scripts created to execute the cross-correlation were made according to the information given in [37]. Each data set's area was, from least dense to most dense in 2MASS catalogue, $199.42\ \text{deg}^2$, $140.46\ \text{deg}^2$ and $50.028\ \text{deg}^2$. The value chosen for the error type I was $\alpha = 0.9973$.

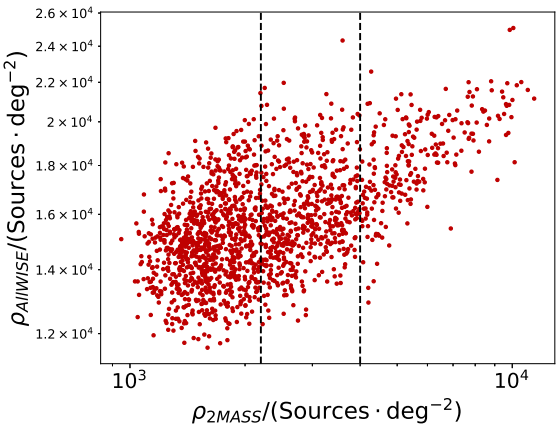
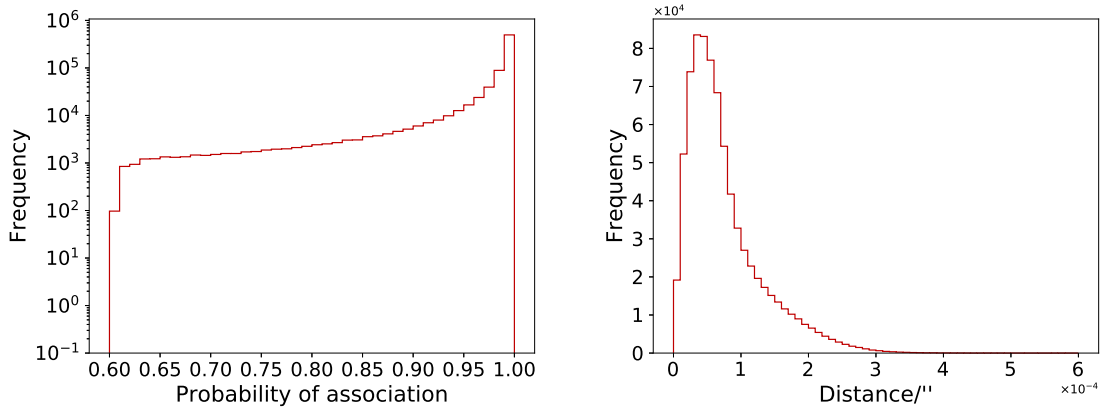


FIGURE 3.5: Division in groups of the used data

3.1.3 Results and tests

After dividing the original data set in three different parts and executing the cross-correlation algorithm, we had three tables with information about the matches obtained. First, we checked if there were any multiple matches in the output (i.e. any detection matched with two or more detections in the other catalogue). This multiple matches appear because there may be two detections from one catalogue that are too close to one detection in the other catalogue. In these situations, we analysed the probability of real association of each match, keeping the one with a higher probability. We found 21 of these multiples matches. The typical probability difference between the match chosen and the first one discarded was around 0.15, although there were some higher probability differences (around 0.3) or lower differences (0.005). After this quick cleaning, we analysed the information provided by the different columns in the table.

After performing the cross-correlation, we had 772353 matches available to work with. At this point, we wanted to eliminate the associations that looked like spurious matches. To do this, before the last filtering and working with the photometrical data, we analysed the probabilities $p(H_0|x, S)$ (Eq. (2.1)), that is, the probabilities of matching two rows coming from the same real source. The histogram with the distribution of this probability is shown in the Figure 3.6a. Our objective at this point was to estimate the number of spurious associations using Eq. (2.2) and removing $\hat{n}_{spur}$ matches from the data set. We do not know which associations are the spurious one, so our criterion was to delete the ones with the lowest probability of real association. Using this equation we obtained

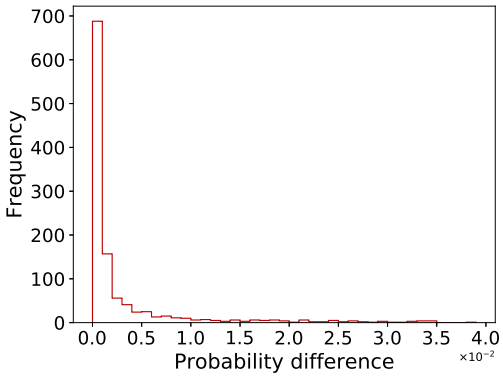


(a) Probability of association histogram

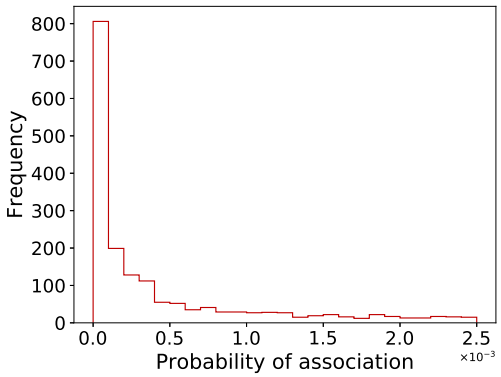
(b) Distance histogram

FIGURE 3.6: Relevant histograms to analyse the quality of the performance of the ARCHES xMatch algorithm

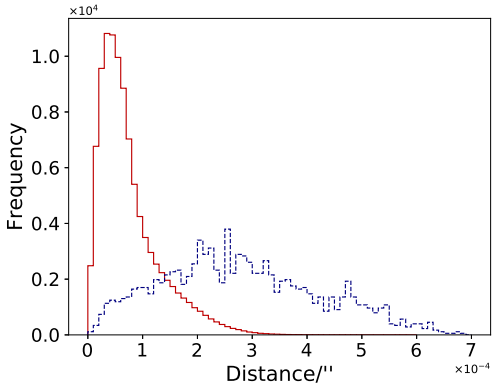
$\hat{n}_{spur} \approx 10^4$, what implies that $p(H_1|S) \approx 0.013$. Before removing the 10000 rows with the lowest probability of real association, we spent time analysing the available histograms. The information that can be extracted from Figure 3.6a is curious: as it can be seen, there is a cut for probabilities smaller than 0.60. To this fact, we can add the shape of the histogram shown in the Figure 3.6b: there, we can see the distribution of the distances between the positions in 2MASS and AllWISE catalogues for the cross-matches found. The shape of this histogram is close to a χ_2 distribution. As we can see in Eq. (2.4), the χ_2 distribution is the one followed by $p(x|H_0, S)$. These two occurrences together suggest that all the multiple matches (the 21 commented before) have been removed.



(a) Probability difference histogram



(b) Probability histogram of spurious associations



(c) Distances histogram of real and spurious matches. The spurious is the dashed one.

FIGURE 3.7: Relevant histograms to analyse the different tests performed

We can also estimate $\hat{n}_{spur}$ without using Eq. (2.2). To achieve this, two tests were performed. The first one wanted to check whether the matches obtained were dependent or not on the area of the sky field analysed. To do this, we took four of the initial fields obtained from the SCS with a radius of $15'$ and performed the cross-correlation using the 2MASS and AllWISE tables for these fields separately. After obtaining the results, the matches obtained were compared with the ones in the bigger groups. When using directly the SCS tables, the algorithm identified 1133 detections from both catalogues; on the other side, with the big tables there were found 1134 matches. The 1133 matches were contained in the 1134, that is, the associations were the same in both cases. This test shows that the performance of the algorithm does not strongly depend on the area of the sky field analysed. If we compare the probabilities, we can see that there are differences between both matches (small differences, but they are still there). These differences can be explained using the Eqs. (2.1) and (2.3): as we can see, the probability $p(H_i|x, S)$ depends on the priors $p(H_i|S)$, which need the area of the field Ω to be computed. Because of this reason the probabilities are not the same, but the matches coincide. The Figure 3.7a shows the histogram for the difference between the probability

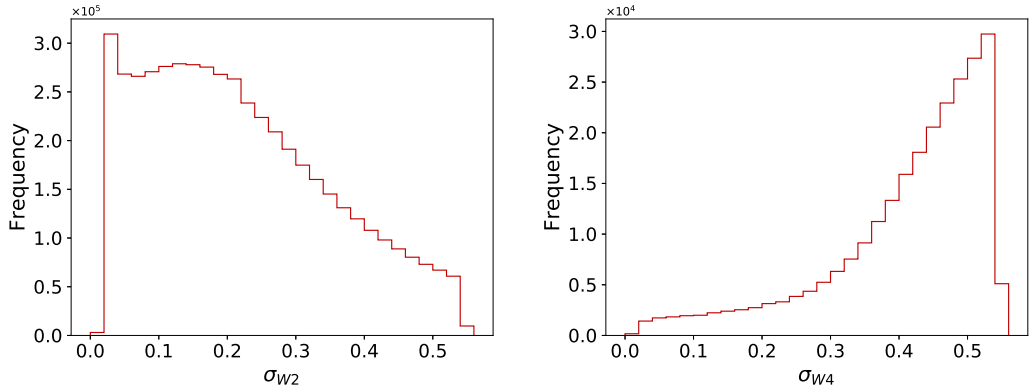
for the little groups and the big groups. As we can easily check, the probabilities obtained for the matches using the small fields are all larger than the ones obtained using a huge field. The difference, however, is not significant (less than 0.5%).

The second test performed consisted in taking the AllWISE data from the third of the three big groups and biasing it, shifting one of its angular coordinates (declination) by a fixed amount. This value was chosen to be $30''$. With this large shift, we ensure that all the matches are spurious and the source densities are not modified. Our initial idea was to use three times the mean value of the standard deviation in the declination, but this number was approximately $1''$, a very small value for our purpose. This biased data set was cross-correlated with the original 2MASS data set and the matches were analysed. The results obtained can be seen in the histograms shown in the Figures 3.7b and 3.7c. This time, the number of matches is 1765, a very little number in comparison with the 221035 real matches in this group: the proportion is, approximately, less than 1% of all matches (without taking into account the probabilities of association) could be spurious. Furthermore, the values for the probability of real association, shown in the Figure 3.7b, are extremely small compared with the probabilities of the real associations. Finally, Figure 3.7c shows the distances between counterparts in the real case and in the biased case. When comparing both cases, it can be seen that the distances are larger than in the later case. They are also more dispersed.

After these tests, we can see that the estimation done using Eq. (2.2) is higher than the one obtained empirically: the result of the last one was $n_{spur} \approx 6000$. We have decided to be cautious and take the maximum value for n_{spur} from the two calculated, that is, there will be removed around 10000 associations from the obtained matches. This decision coincides with keeping the matches with a probability $p(H_0|x, S) \geq 0.69133$. After doing this, the number of matches decreased to 762352.

3.1.4 Data filtering

The data cleaning performed until now used only the information given by the cross-match. The filtering that concerns now focused in the photometric data and their quality. We analysed the photometric data of the 762352 detections and the ones with the worst quality were removed from the data set. First, both 2MASS and AllWISE catalogues had a column denoted *use_src*. As its name suggests, this value indicates which detections are optimal to work with. The data used in this work were the one marked as optimal by the *use_src* column.



(a) σ_{W2} histogram (b) σ_{W4} histogram

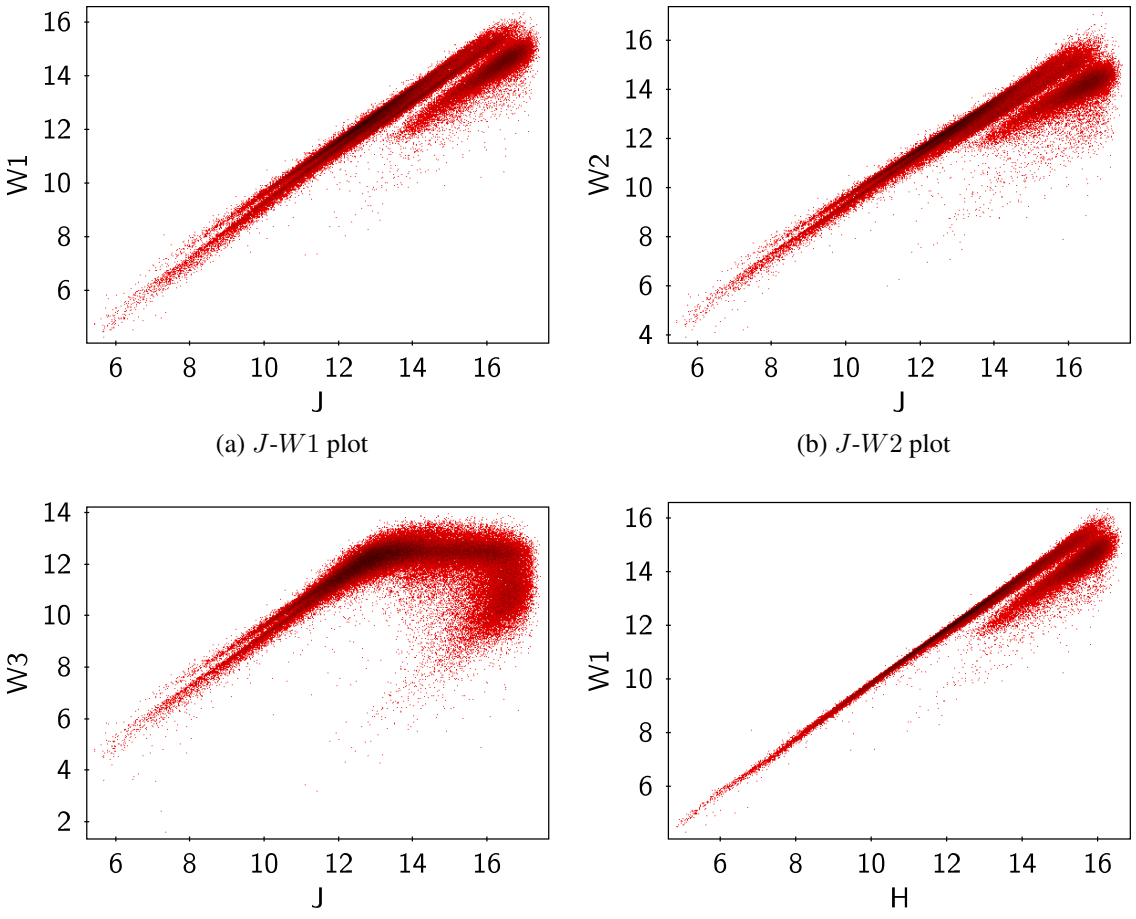
FIGURE 3.8: Histograms with the errors for the filters $W2$ and $W4$

At this point, we had to select the best photometric data to work with. We could take seven different filters for each source: the J, H and K bands from the 2MASS catalogue and $W1$, $W2$, $W3$ and $W4$ bands from AllWISE. We could have worked with the whole set of bands; however, it was decided that the $W4$ would not be used. The reason for not using it is that only the 4% of the

original AllWISE data had information about the $W4$ error σ_{W4}. Furthermore, the rows with that information had an excessively big error. A large photometric error is usually associated with some other problem with the photometry of the source that would make W4 unreliable. In the Figure 3.8 it can be seen a comparison between the errors in $W2$ and $W4$: the sources that have available data for the $W4$ band have associated big values for σ_{W4} in comparison with their σ_{W2} values.

The final selection of the data was done using the photometric quality flags provided by the catalogues. These flags contain information about the error of each photometric measurement and its signal-to-noise ratio. Each filter had a flag associated. The explanatory documents for each catalogue [12, 14] recommend using the maximum quality filter, named A. However, if we used this criterion, the number of detections that would be available to work with would reduce to 18967, a small data set that could be not useful to perform unsupervised ML techniques. Because of this reason, we decided to be less conservative and take photometric data with quality flags A, B and C. This implies that the photometric detections for J, H and K bands would have a signal-to-noise ratio greater than 5 and an error lower than 0.21714; on the other hand, the $W1$, $W2$ and $W3$ data would have a signal-to-noise ratio greater than 2. Choosing these flags, the number of data used in the ML tests turned out to be 126573.

3.2 Dimensionality reduction techniques



(a) J-$W1$ plot (b) J-$W2$ plot

(c) J-$W3$ plot (d) H-$W1$ plot

FIGURE 3.9: 2D representations of the raw data.

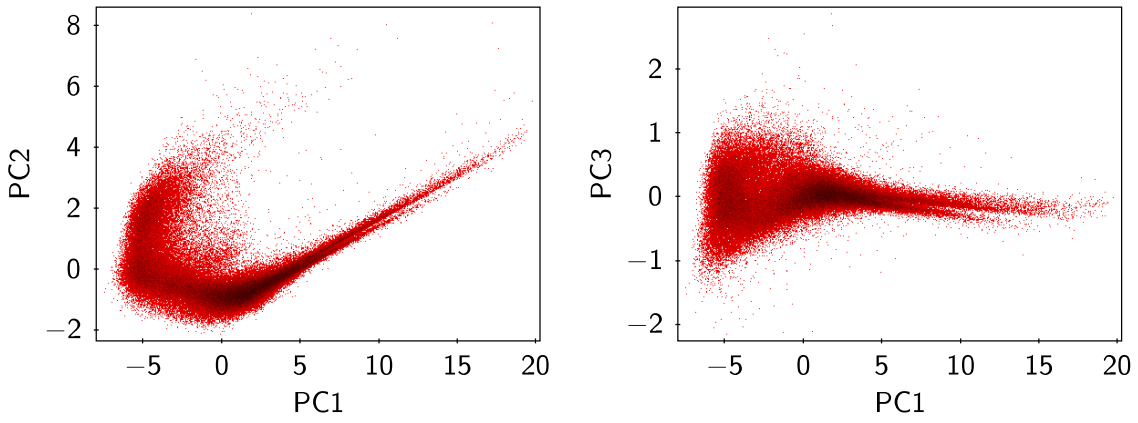
As we said in Chapter 2, some data structure can be distinguished using the raw data. For example, in Figure 3.9 some plots have been represented. In this figure, the regions with more density of points

are obscured, using a colour gradient to indicate the local density. It can be seen there that, in general, we can distinguish a dense line (actually, this line have two tails towards bottom-left and another one towards top-right, but they are so close that in some diagrams are difficult to tell apart) and another group, more or less separated from the line depending on the magnitudes used. However, although some structure can be seen at a glance, it is preferable to work with the transformed data using the dimensionality reduction techniques. In the transformed parameter space, we expect to be able to separate different groups better. These groups in turn, should correspond to different classes of astronomical sources (see Chapter 4).

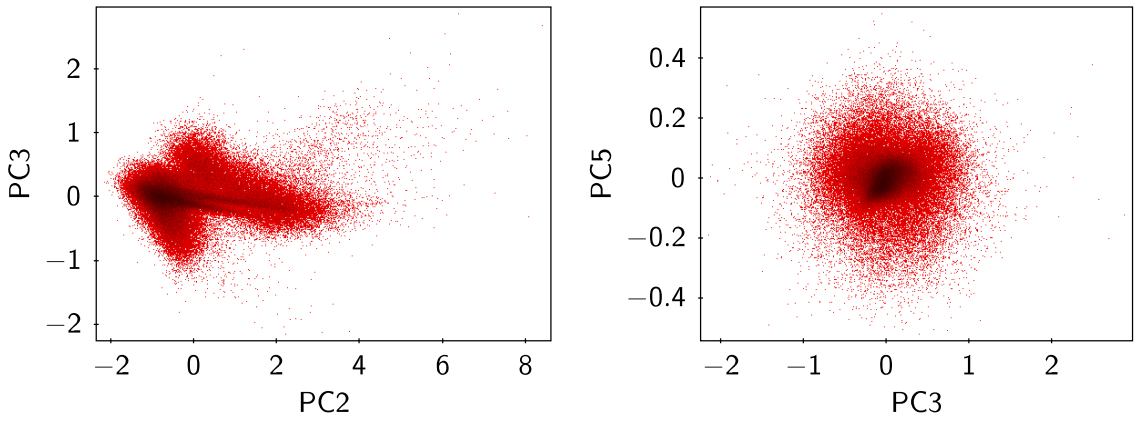
3.2.1 Principal Component Analysis (PCA)

After creating the PCA model, the analysed bands were transformed to the principal components. The resulting values for the components in Eqs. (2.12) and (2.13) were:

$$\mu = \begin{pmatrix} 13.7598 \\ 13.2307 \\ 13.0052 \\ 12.7933 \\ 12.7388 \\ 11.6540 \end{pmatrix} \quad P = \begin{pmatrix} -0.4944 & -0.4722 & -0.4403 & -0.4085 & -0.3844 & -0.1555 \\ 0.2897 & 0.2007 & 0.0692 & -0.1114 & -0.2156 & -0.9012 \\ 0.4766 & 0.2507 & -0.0238 & -0.4278 & -0.6025 & 0.4042 \\ 0.6006 & -0.3234 & -0.6606 & 0.0780 & 0.3034 & -0.0118 \\ 0.2723 & -0.6746 & 0.5990 & -0.3100 & 0.1254 & -0.0084 \\ 0.0981 & -0.3378 & 0.0750 & 0.7318 & -0.5788 & 0.0101 \end{pmatrix}$$



(a) $PC1$-$PC2$ plot (b) $PC1$-$PC3$ plot

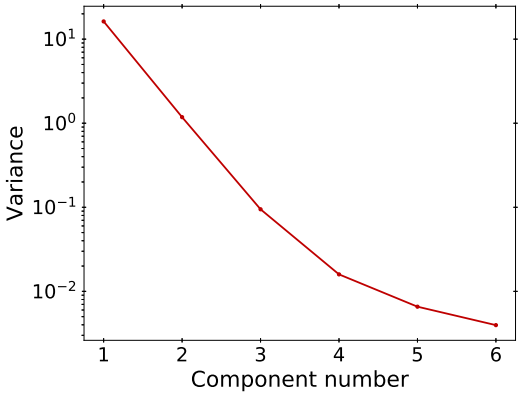


(c) $PC2$-$PC3$ plot (d) H-$W1$ plot

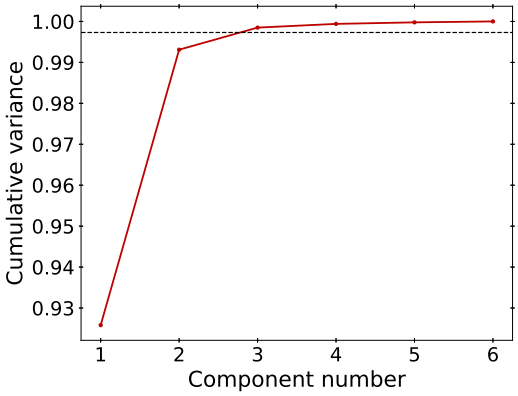
FIGURE 3.10: 2D representations of the PCA data

Some of the 2D plots with the principal components obtained are shown in Figure 3.10. There, it can be seen that some principal components provide interesting information: for example, in Figure

3.10c it can be seen some groups well defined. On the other hand, some components do not provide any useful information: for example, in Figure 3.10d no data structure can be distinguished. Because of that reason, it is necessary to choose the components that explain the maximum amount of variance possible.



(a) Variance explained by each component



(b) Cumulative percentage of variance explained by each component. The black line corresponds to the α value chosen.

FIGURE 3.11: Variance histograms for the PCA model created

Using the notation in the Eq. (2.14), the value chosen for α was 0.9973. As it can be seen in the Figure 3.11a (the ordinate axis is in logarithmic scale), the variance for each principal component decreases rapidly. Using the Figure 3.11b, it is easy to check that, for the α value chosen, the first three principal components explain more than the 99.73% of the variance associated with them. Therefore, the other three components ($PC4$-$PC6$) were discarded.

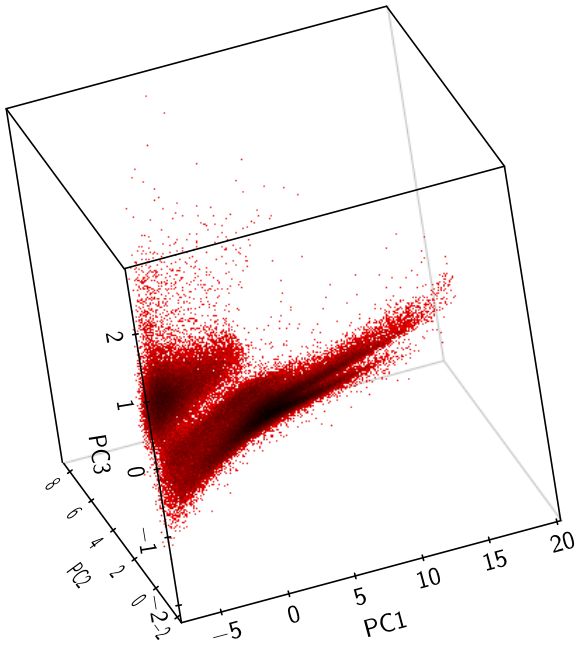


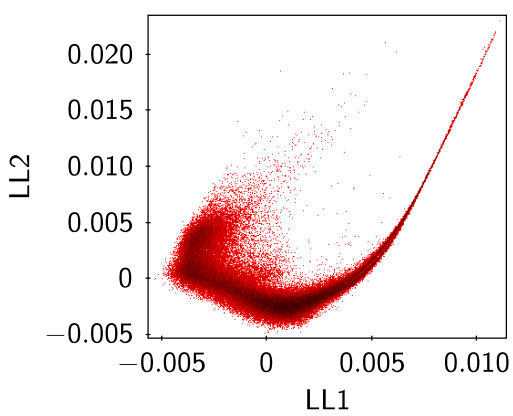
FIGURE 3.12: 3D representation for the chosen components

In Figure 3.10 we have represented 2D plots. However, as we have kept three principal components, it would be more useful a 3D plot with all the information. For this reason, in Figure 3.12 it

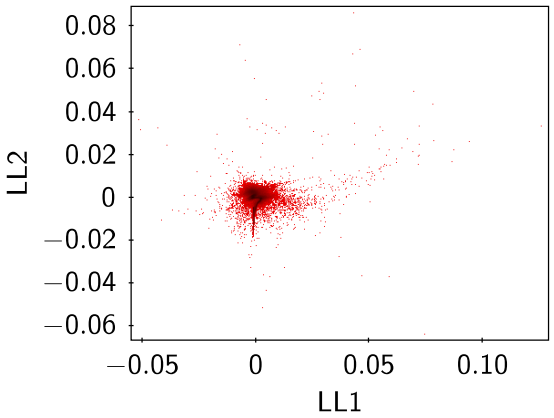
has been represented a 3D plot with all the components of interest. At first glance, it looks like there are at least four different clusters of data: a rounded one at the top of the figure, and three separate planes below it. We will discuss this in more detail in Chapter 4.

3.2.2 Locally-linear embedding (LLE)

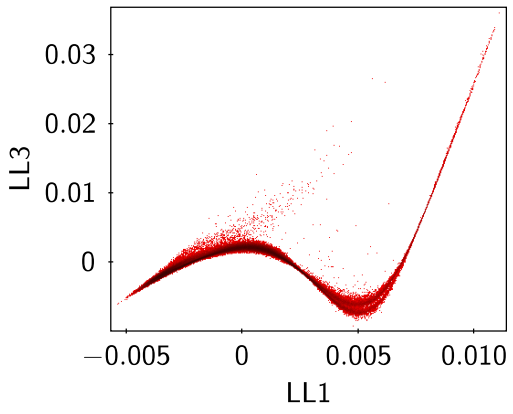
The LLE model was created using six dimensions too. The first point to comment is the number of neighbours k used to create the model. As it can be seen in Figure 3.13b, for $k = 10$ the points does not describe any coherent geometric structure. However, for $k = 50$ this structure can be seen, as can be checked in the rest of panels of Figure 3.13. After some testing, the number of neighbours was finally chosen to be $k = 50$. We decided to use the same number of coordinates as in the PCA case, that is, the first three coordinates, discarding $LL4$, $LL5$ and $LL6$. The 2D plots are represented in Figure 3.13. However, as in the case of the PCA model, a 3D plot can be made with the first three components. The three-dimensional plot for these three components is shown in the Figure 3.14. Roughly, it can be seen a narrow and dense line that contains a lot of points; and a wavy surface, less dense than the line. The structure distinguished will be analysed in more detail in Chapter 4.



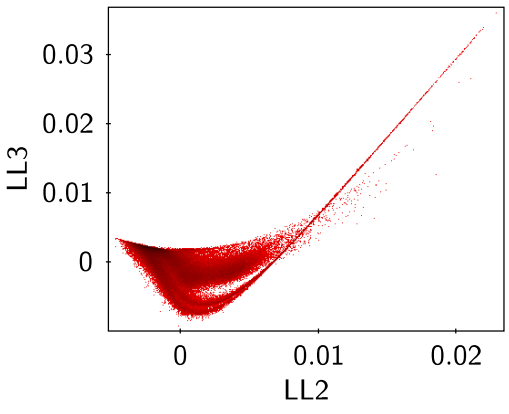
(a) $LL1$-$LL2$ plot for $k = 50$



(b) $LL1$-$LL2$ plot for $k = 10$



(c) $LL1$-$LL3$ plot for $k = 50$



(d) $LL2$-$LL3$ plot for $k = 50$

FIGURE 3.13: 2D representations of the LLE data

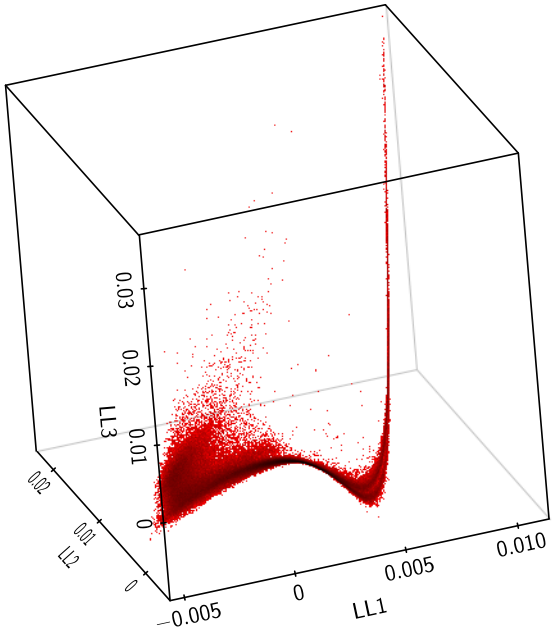


FIGURE 3.14: 3D representation for the chosen coordinates

3.3 Clustering

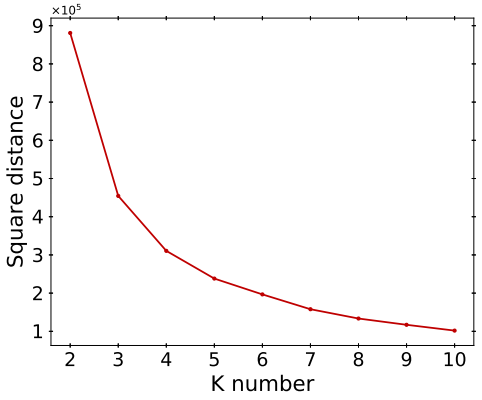
Now, it is time to apply clustering techniques to the problem data. Each technique will be applied to the PCA and LLE transformed data set, and the results will be widely discussed in Chapter 4. However, we will comment briefly the shapes of the subsets obtained with each algorithm. As we have said in the previous section, we will use only three components in these clustering methods, both for PCA and LLE data.

3.3.1 *K*-Means

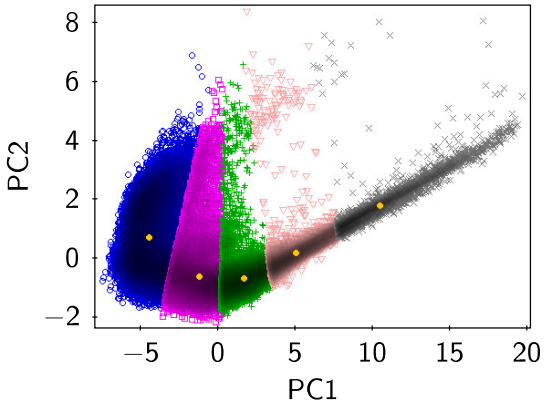
In this section, we will show the results obtained after aplying K-Means method to the PCA and LLE data sets. First of all, we discuss which number of subsets, K, is the optimal to split each of the problem data sets. As we described in Chapter 2, we will use the "elbow method", that is, we will plot the value given by Eq. (2.18) as a function of the number K selected. These plots can be seen in Figure 3.15a for the PCA data and in Figure 3.17a for LLE. It can be seen that, in the case of PCA data, the angle in the plot emerges for $K = 5$; on the other hand, for LLE data the angle can be seen for $K = 4$. Because of that reason, the plots represented in this section correspond to these values.

For each data set, we made the two-dimensional representations of the components used, as well as the three-dimensional representation for the first three components. In each of the two-dimensional representations the centres of the clusters have also been represented as yellow points. The 2D plots shown in Figures 3.15 and 3.17 have been represented using different colours and marks for each subset; however, to ease the data visualisation, in Figures 3.16 and 3.18 the marks are all the same, and the only way to difference the subgroups is the different colours between them.

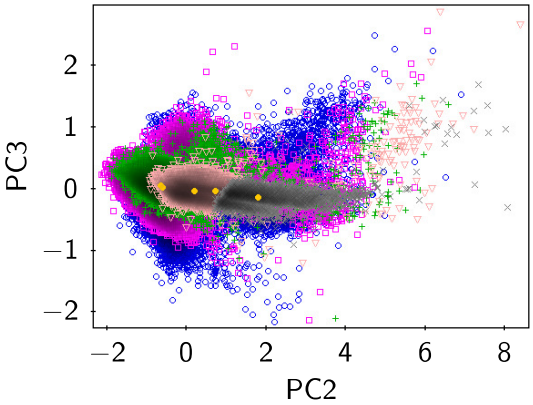
At first glance, it looks like the K-Means method has not distinguished as subgroups the data structure that can be seen without applying clustering methods. Still, the discussion will be extended in Chapter 4.



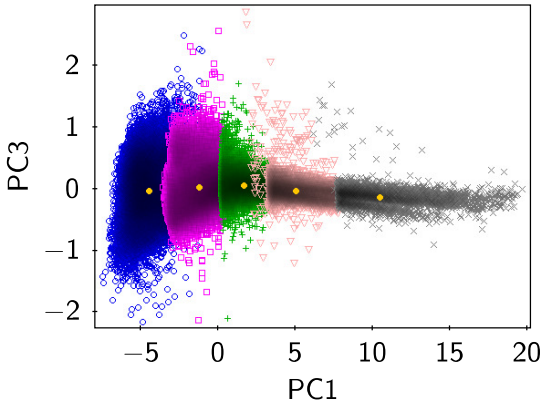
(a) PCA square error for each value of K



(b) $PC1$-$PC2$ plot for K-Means



(c) $PC2$-$PC3$ plot for K-Means



(d) $PC1$-$PC3$ plot for K-Means

FIGURE 3.15: 2D representations of the PCA data after aplying K-Means algorithm

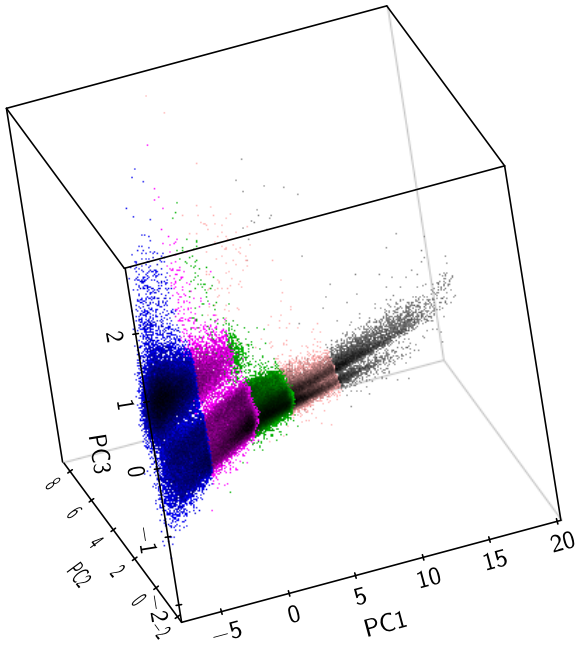
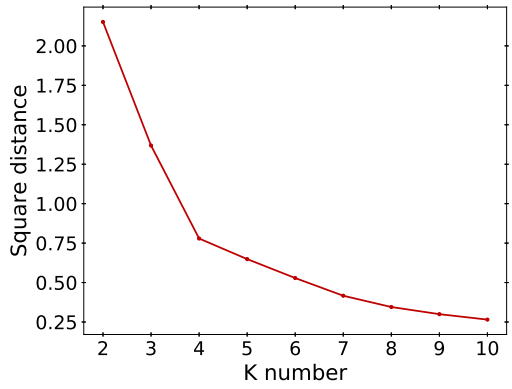
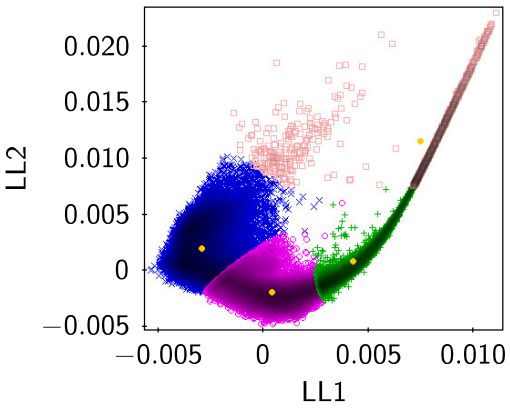


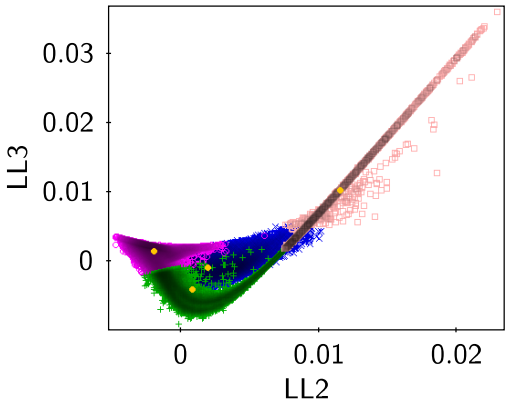
FIGURE 3.16: 3D representation of the PCA data after aplying K-Means algorithm



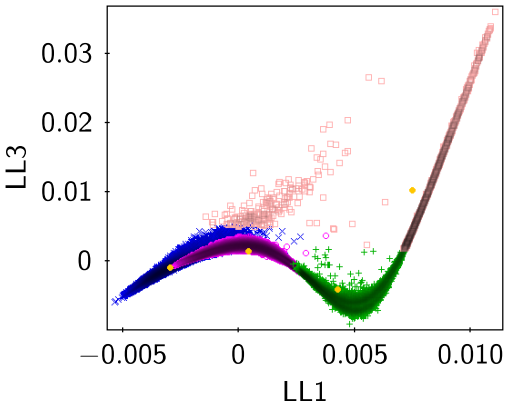
(a) LLE square error for each value of K



(b) $LL1$-$LL2$ plot for K-Means



(c) $LL2$-$LL3$ plot for K-Means



(d) $LL1$-$LL3$ plot for K-Means

FIGURE 3.17: 2D representations of the LLE data after aplying K-Means algorithm

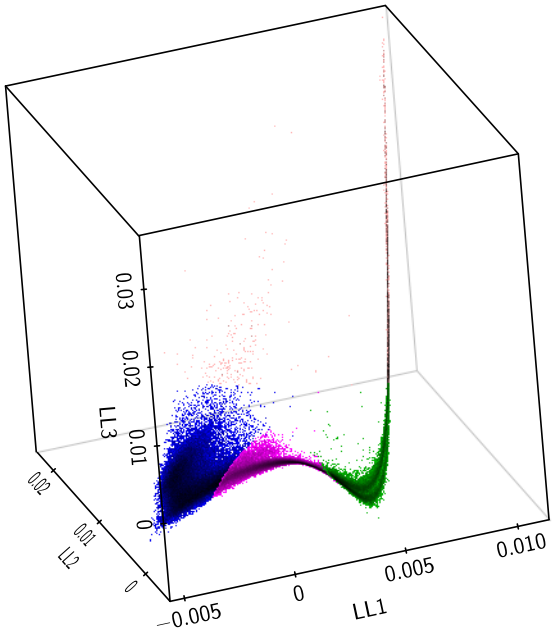


FIGURE 3.18: 3D representation of the LLE data after aplying K-Means algorithm

3.3.2 Mean shift

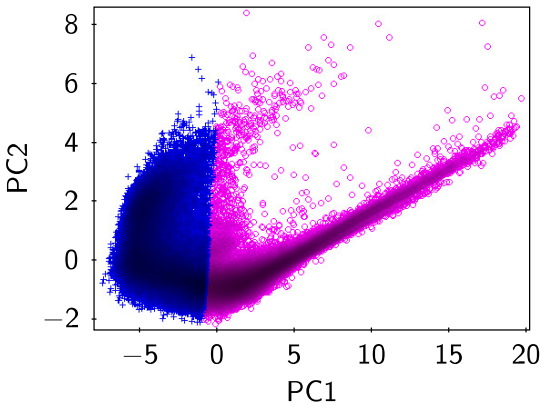
The results obtained using Mean shift method are shown in this section. In the case of PCA, the program estimates that a value of $h = 2.8$ is the optimal bandwidth to work with. However, it can be seen in Figure 3.19a that, when we use that value to create the model, the algorithm finds only two different clusters. This number of subgroups is very small, so we decided to choose a lower value for the bandwidth. Specifically, it was chosen the value $h = 2.0$. This value caused the algorithm to find four different clusters. This value looks like a more appropiate number. This will be discused more thoroughly in Chapter 4. In the case of LLE, with the estimated bandwidth $h = 0.004$ the model was able to find four different clusters.

As in the case of K-Means, in Figures 3.19 and 3.21 we have represented all the two-dimensional plots that can be done with the coordinates used to adjust the models. On the other hand, in Figures 3.20 and 3.22 we can see the three-dimensional representations. The results obtained will be discussed in Chapter 4; however, at a first glance, it looks like the results obtained are similar to the ones provided by the K-Means method.

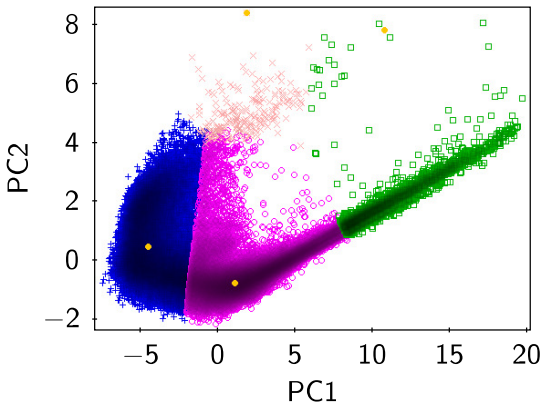
3.3.3 Gaussian Mixture Model (GMM)

As for the K-means and Mean shift methods, we present here the results of the GMM method over the PCA and LLE transformed data separately. In order to know the number of Gaussians that are optimal to fit the PCA and LLE data sets, it is necessary to analyse the Figures 3.23a and 3.25a. In the figure associated to the PCA data set, it can be seen that both AIC and BIC decrease as the number of Gaussians K increases. Also, they take similar values for each K. Theorically, lower values of AIC and BIC indicate that the model fit is better. Therefore, it looks like we should use 10 Gaussians to fit the PCA data set. However, if we do this, we are taking the risk of overfitting our data. Because of that, it has been decided to use the elbow method to choose the optimal number of Gaussian components. If we analyse Figure 3.23a, it can be seen that, for $K = 5$, there is a pronounced angle. For that reason, the number of Gaussian components used to fit the PCA was $K = 5$. Likewise, for the LLE data set we decided to use $K = 4$ Gaussian components, even if the AIC and BIC values for $K > 4$ are lower.

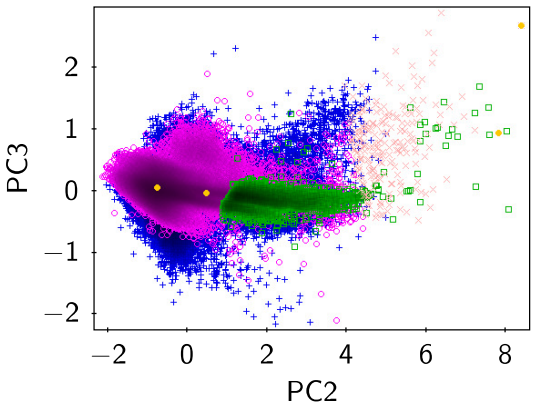
In Figures 3.23 and 3.25 we have plotted all the possible two-dimensional representations of the three first coordinates, both for PCA and LLE data sets. The three-dimensional representations are shown in Figures 3.24 and 3.26. Roughly, it looks like, in the PCA case, the separation performed makes more sense than the K-Means and Mean shift. However, in the LLE case, this method apparently does not work as well as for PCA. The complete discussion can be found in Chapter 4.



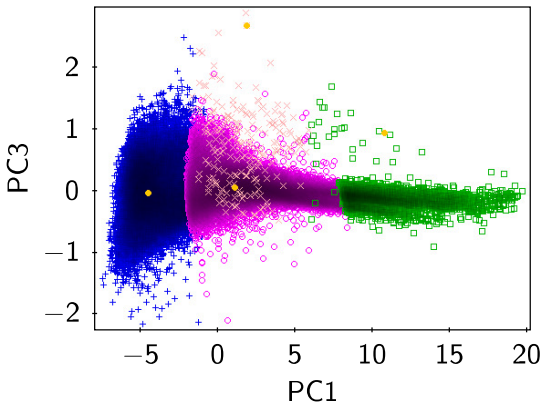
(a) $PC1$-$PC2$ plot for Mean shift using $h = 2.8$



(b) $PC1$-$PC2$ plot for Mean shift using $h = 2.0$



(c) $PC2$-$PC3$ plot for Mean shift using $h = 2.0$



(d) $PC1$-$PC3$ plot for Mean shift using $h = 2.0$

FIGURE 3.19: 2D representations of the PCA data after aplying Mean shift algorithm

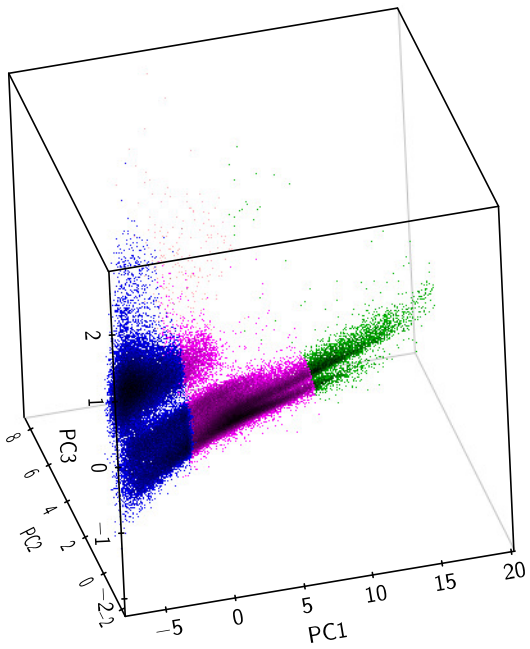
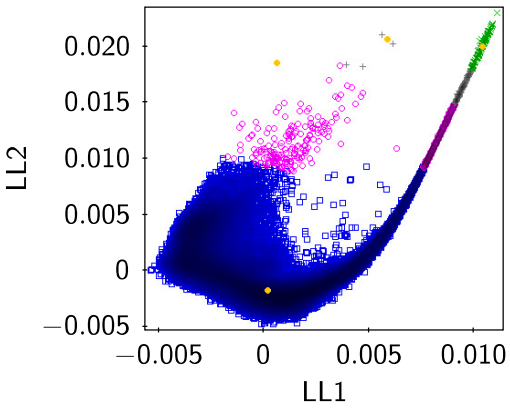
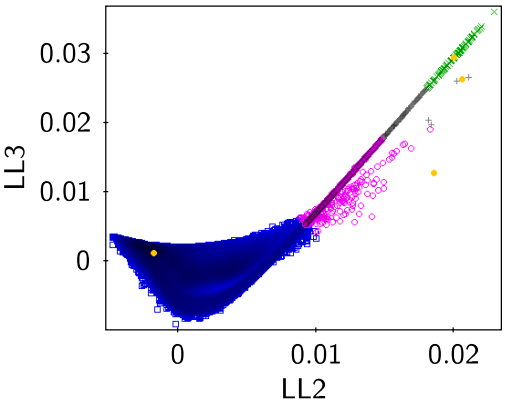


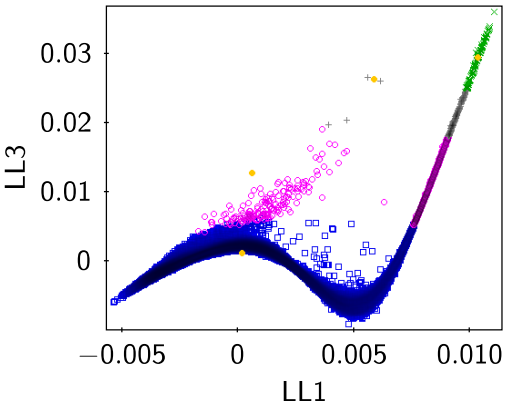
FIGURE 3.20: 3D representation of the PCA data after aplying Mean shift algorithm



(a) $LL1$-$LL2$ plot for Mean shift



(b) $LL2$-$LL3$ plot for Mean shift



(c) $LL1$-$LL3$ plot for Mean shift

FIGURE 3.21: 2D representations of the LLE data after aplying Mean shift algorithm

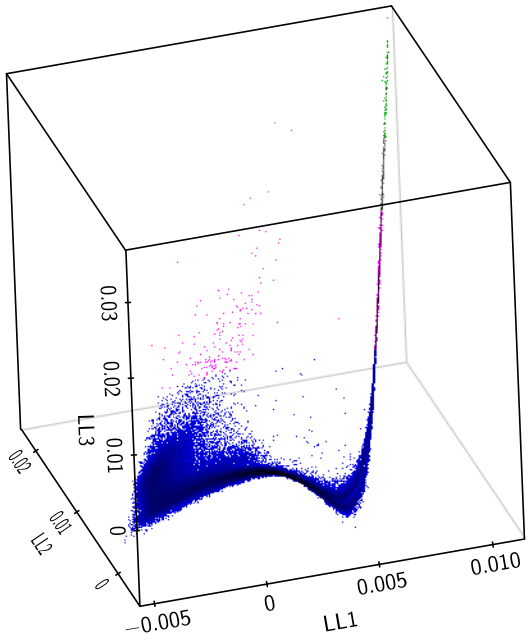
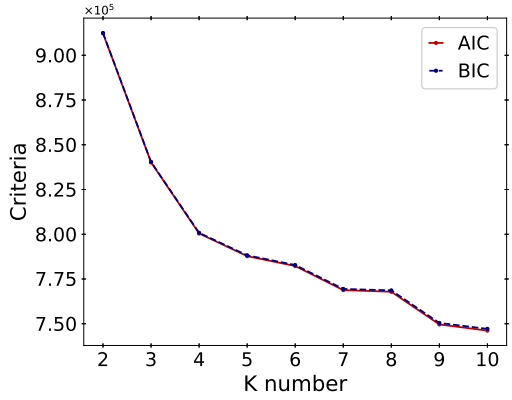
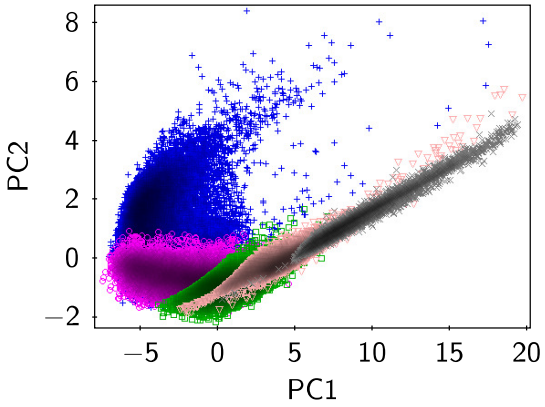


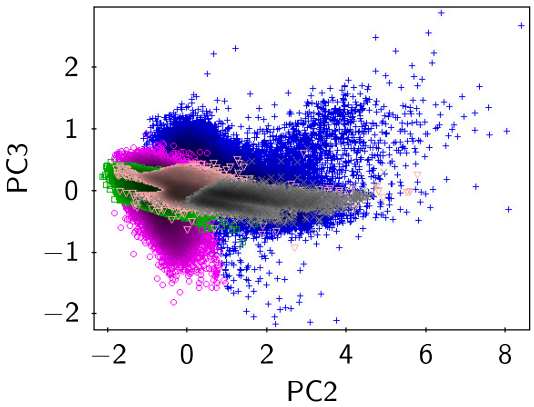
FIGURE 3.22: 3D representation of the LLE data after aplying Mean shift algorithm



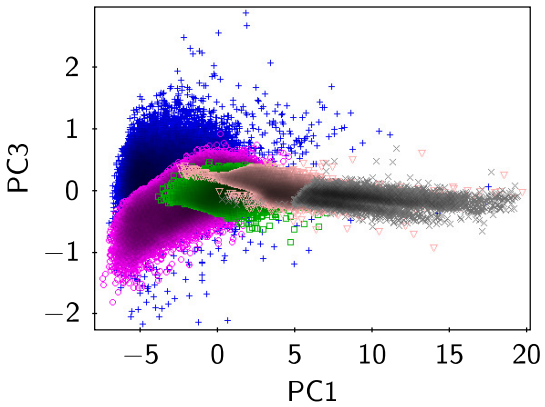
(a) AIC and BIC for PCA data set



(b) $PC1$-$PC2$ plot for GMM



(c) $PC2$-$PC3$ plot for GMM



(d) $PC1$-$PC3$ plot for GMM

FIGURE 3.23: 2D representations of the PCA data after aplying GMM algorithm

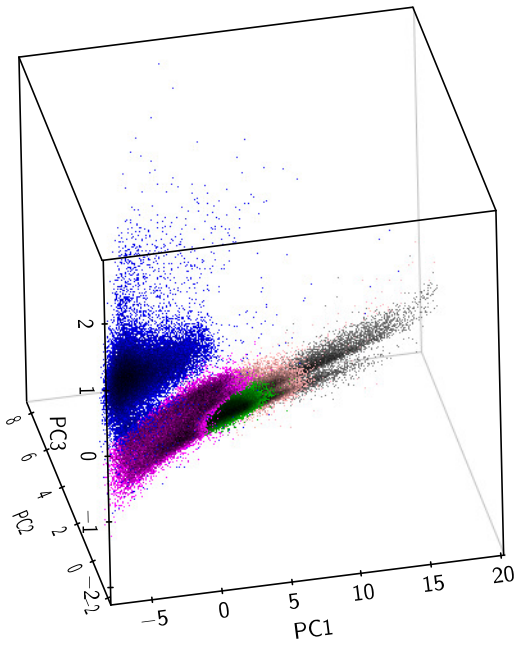
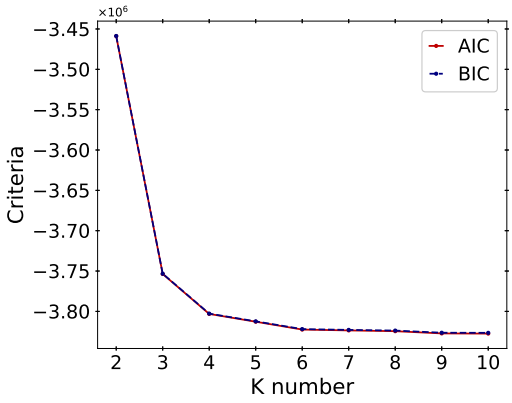
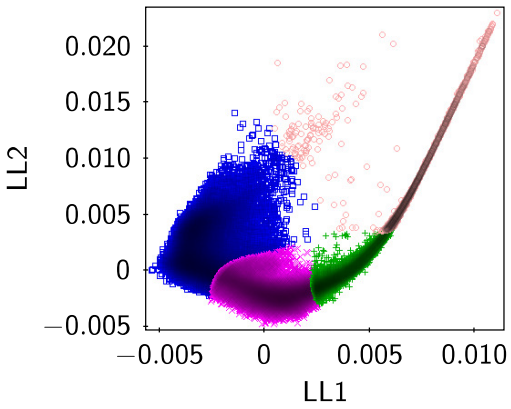


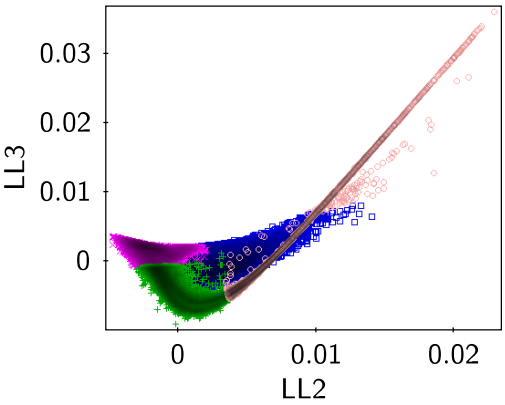
FIGURE 3.24: 3D representation of the PCA data after aplying GMM algorithm



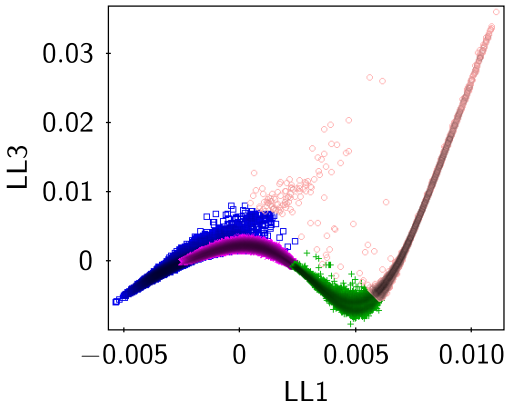
(a) AIC and BIC for LLE data set



(b) $LL1$-$LL2$ plot for GMM



(c) $LL2$-$LL3$ plot for GMM



(d) $LL1$-$LL3$ plot for GMM

FIGURE 3.25: 2D representations of the LLE data after aplying GMM algorithm

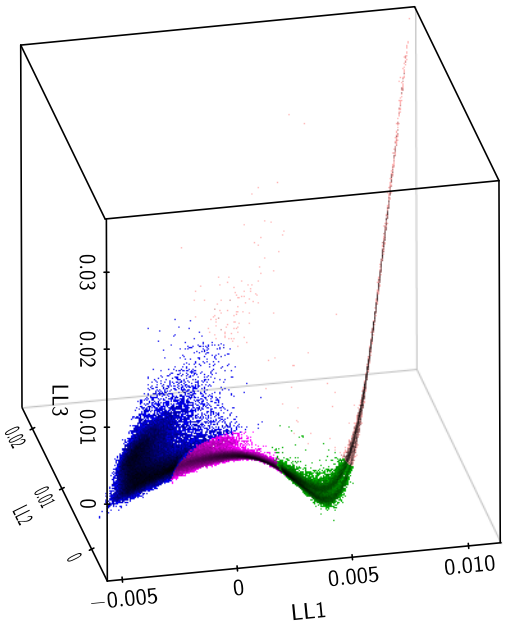


FIGURE 3.26: 3D representation of the LLE data after aplying GMM algorithm

Chapter 4

Discussion

After showing the results, in this chapter we discuss them and their quality. We will justify some decisions made in the course of the data processing and filtering. We will also discuss the performance of each ML algorithm and verify if the subgroups found with the clustering methods correspond to different types of astronomical sources. In some cases, this verification will be done using a visual clasification; in others, we will use a verification data set with labelled sources.

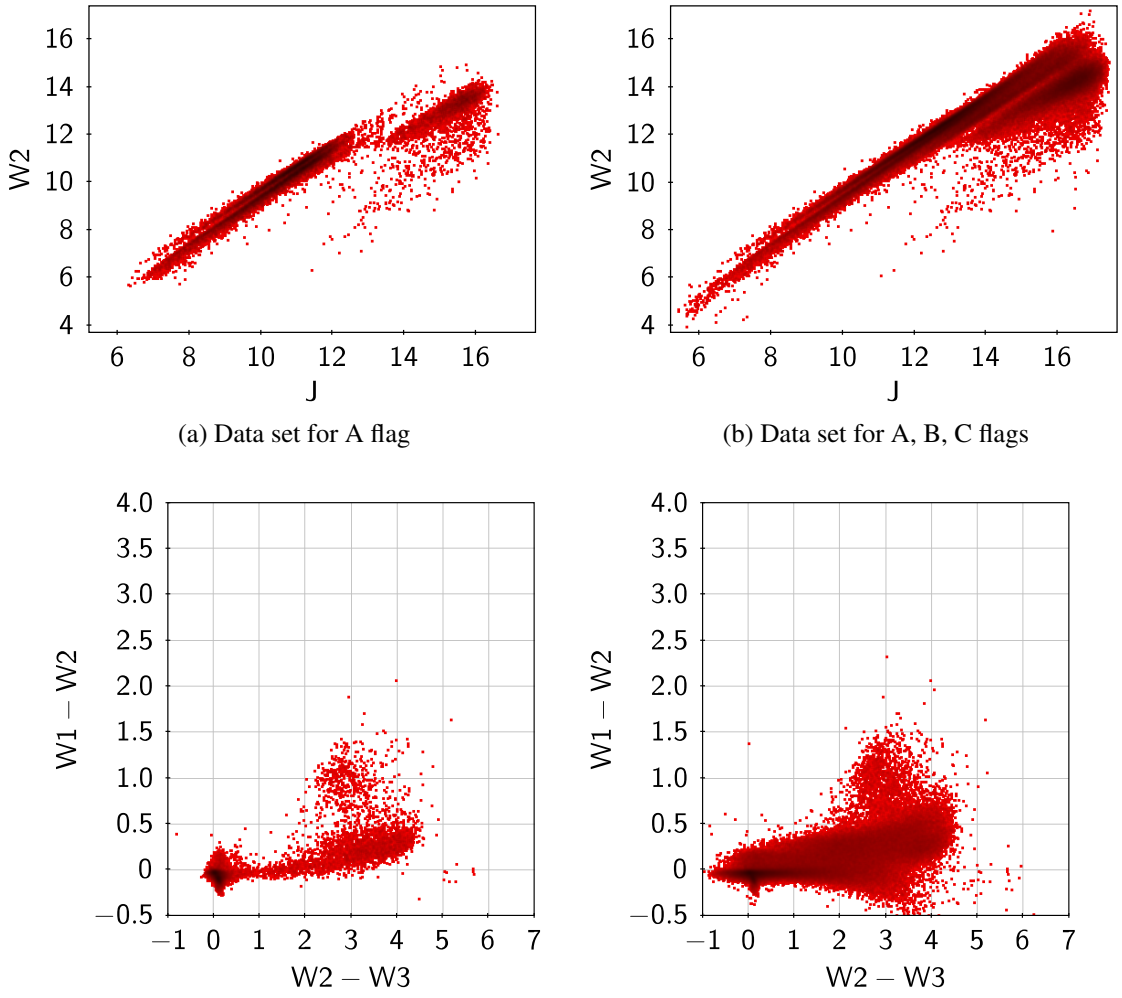
4.1 Data quality effects

One interesting point to discuss is something we said at the end of Section 3.1: why is the data set with 18967 sources not enough to find clusters, while the one with 126573 may provide better results? We do not have a numeric answer to this question, but a visual one. In Figure 4.1 we can see a two-dimensional representation of the J and $W2$ magnitudes for both data sets. In the Figure 4.1a, corresponding to the "most filtered" data, we can see that there are easily separable groups, so using it should be enough. However, if we compare it with the Figure 4.1b, corresponding to the "less filtered" data (the one used in this project), we can see that the sizes of the clusters are larger and we even loose a group (the one for $J > 13$ and $W2 > 12$) if we filter with the A flag our data set. To check that these additional sources are not just noise, we have represented the colour indices using the AllWISE magnitudes. These plots are shown in Figures 4.1c and 4.1d. With the help of Figure 1.2, we can check that the new points are not noise, but real sources. Because of that, we decided to be more permissive and use the data with A, B, and C quality flags in this project.

4.2 Performance of the dimensionality reduction methods

In respect of the dimensionality reduction algorithms, we can discuss the performance of each algorithm. At first sight, it looks like the PCA algorithm provides results that are easier to interpret than the ones obtained by the LLE method. This happens because PCA produces a linear transformation, something that a priori is easier to visualize than the non-linear transformation obtained with the LLE model. We can even distinguish some groups in Figure 3.12: a big cluster at the top, and three flat groups under it. In the case of Figure 3.14 we can see a dense line with a lot of sources, but the groups are not separated as in the PCA case.

Moreover, the execution times for each algorithm are very different: in the case of the PCA algorithm, we need less than a minute to obtain the transformed data set, while it takes more than two hours to obtain the LLE data set (it is important to mark that we have used a common laptop to run the scripts). This important difference in the execution times comes from the fact that, in the PCA case, we are diagonalizing a 6×6 matrix, while in the LLE algorithm we are finding two matrices, both with the size of the initial data set. That is a huge difference in terms of computation. We must



(a) Data set for A flag

(b) Data set for A, B, C flags

(c) Colour indices of the data set for A flag

(d) Colour indices of the data set for A, B, C flags

FIGURE 4.1: Data sets obtained when restricting more or less the quality flags

also note that our choice of the number of neighbours k used to adjust the LLE model affects to the execution time. This is because it has a direct impact in the sparsity of one of the matrices computed. Because of that reason we have chosen $k = 50$: it is a number big enough to distinguish some data structure, but not too big to have execution times of several hours.

One question may arise at this point: why we have chosen PCA and LLE methods and not others to perform dimensionality reduction? The reason of choosing PCA is simple: it is the main linear method to transform a data set, and it is easy to decide how many components you want to use. Furthermore, its simplicity makes it easy to understand how it works. In the case of LLE embedding, the answer is not that simple because there are more manifold learning methods that are not complicated. Originally, we thought that we could apply one more manifold learning model, called IsoMap. However, this method caused problems because of the size of the data set used. Apparently, it could not handle such a large data set. As we had already executed a manifold learning method, we decided to keep the LLE results and ignore the IsoMap method.

4.3 Performance of the clustering methods

Now it is time to analyse the performance of each clustering method. First, the execution time of the K-Means and GMM methods are very similar, taking a few seconds to create each model.

However, the Mean shift method takes more time in executing the algorithm, more than two hours if we calculate the optimal bandwidth with the Python method.

In terms of the effectiveness of the methods, if we analyse first the PCA data set, we can see that the K-Means and Mean shift methods ignore the data structure that can be seen with the naked eye. The limits of the groups are transverse to the groups, as it can be seen in Figures 3.16 and 3.20. Beyond the number of clusters found in each model, the shapes of the groups are similar in both cases. Considering the time execution for each one of the methods, K-Means method looks like a more plausible approach to this problem. The analysis of the LLE data set is similar: especially for the Mean shift method it can be seen that the clustering is not very effective, obtaining a huge group and another three with a limited number of sources. The sizes of the groups found in the LLE data set using K-Means method are more balanced. However, based on the fact that the geometry of the group searched by the method does not match with the groups that we can see visually, the result does not look optimal.

4.3.1 Validation sample

The method that, apparently, provides better results in both cases is the GMM model. As before, this argumentation is based in the groups that we can visualize a priori. In order to prove the effectiveness of the method, we have selected a validation set from the SDSS catalogue [38] with labelled sources and their magnitudes of interest. The classification of these sources is based on spectroscopy. This validation set is composed of sources with A quality flags and with the 2MASS and AllWISE magnitudes studied. We have transformed these magnitudes to the PCA and LLE spaces and labelled them using the subgroups obtained with the clustering methods.

As the verification data set had a lot of subgroups for each type of radiating source, we decided to represent the colour indices shown in Figure 1.2 and, with the help of the figure, group these subgroups in larger subgroups. This representation, shown in Figure 4.2 (the clusters are not shown in this panel), also serves to check that the verification set occupies the same space as the problem data set. In this figure, the verification sources (blue crosses) have been located over the problem data set (red circles). The final spectroscopic classification was made with five types of sources: stars, galaxies, starbursts, AGNs/Seyferts and quasars. The incidence of each source group in the clusters found by the GMM method using the PCA data is summarized in Table 4.1. One thing to remark is that there is not any cool T-dwarf in the data set (according to the separation shown in Figure 1.2). Nor are ULIRGs/LINERs/obscured AGN, as we can check in the same figure.

TABLE 4.1: Table that encodes the performance of the GMM algorithm over the PCA model using the verification data set. For each type of object we have indicated the number of sources within the data set and how many are in each cluster, as well as the percentages.

Source	Total	Magenta group	Green group	Blue group	Grey group	Pink group
Star	2542	275 (10.82%)	63 (2.48%)	753 (29.62%)	507 (19.94%)	944 (37.14%)
Galaxy	61042	48 (0.08%)	1 (0.00%)	60974 (99.89%)	16 (0.03%)	3 (0.00%)
Starburst	5355	0 (0.00%)	0 (0.00%)	5355 (100.00%)	0 (0.00%)	0 (0.00%)
AGN	6302	4 (0.06%)	0 (0.00%)	6298 (99.94%)	0 (0.00%)	0 (0.00%)
Quasar	4244	1 (0.02%)	1 (0.02%)	4211 (99.22%)	19 (0.45%)	12 (0.28%)
TOTAL	79485	328 (0.41%)	65 (0.08%)	77591 (97.62%)	542 (0.68%)	959 (1.21%)

One first thing to comment about the information shown in Table 4.1 is the low number of stars that our verification data set has, in comparison with the other type of sources. One remarkable thing that can be seen in this table is that the stars are not correctly split: they are spread in the five clusters. Apart from that, another striking thing that is important to highlight is that most of the remaining

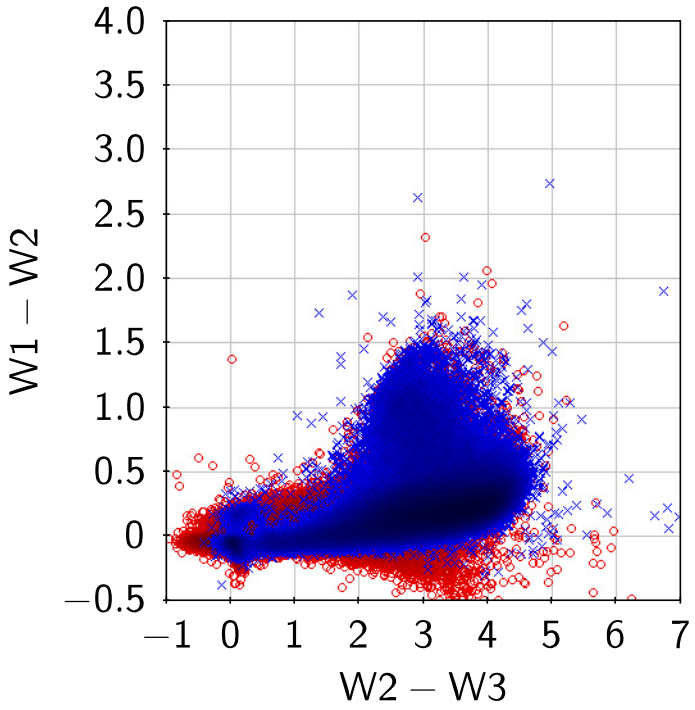


FIGURE 4.2: Colour indices representation for the validation set (blue crosses), represented over the problem set (red circles)

sources fall within the blue group. Moreover, all the clusters except the blue one are composed practically only of stars. This may indicate two things: that our validation data set is somehow biased, or that our problem data set does not have as many galaxies (not only "normal" galaxies, but all types of them) as we would like. These two hypotheses are supported by the three-dimensional representation shown in Figure 4.3. In this figure, we have represented in blue the sources within the blue group, while the rest of sources have been coloured in red. If we analyse the blue group, it can be seen that it does not follow a Gaussian distribution. Actually, the shape of the group looks like two transverse Gaussians, one vertical and other horizontal. Therefore, if we had more sources of that kind (starburst, quasar, etc.) in our problem data set, we may obtain more accurate results. In the figure it can also be seen that we have fewer sources corresponding to the dense and narrow groups that can be seen in Figure 3.12.

TABLE 4.2: Table that encodes the performance of the GMM algorithm over the LLE model using the verification data set. For each type of object we have indicated the number of sources within the data set and how many are in each cluster, as well as the percentages.

Source	Total	Magenta group	Green group	Blue group	Pink group
Star	2542	399 (15.70%)	1126 (44.30%)	649 (25.53%)	368 (14.48%)
Galaxy	61042	1545 (2.53%)	9 (0.01%)	59458 (97.41%)	30 (0.05%)
Starburst	5355	0 (0.00%)	0 (0.00%)	5336 (99.65%)	19 (0.35%)
AGN	6302	68 (1.08%)	0 (0.00%)	6205 (98.46%)	29 (0.46%)
Quasar	4244	1 (0.02%)	18 (0.42%)	4121 (97.10%)	104 (2.45%)
TOTAL	79485	2013 (0.41%)	1153 (0.08%)	75769 (97.62%)	550 (0.68%)

In the case of the LLE data set, the information is written in Table 4.2. Prior to the table analysis, we can imagine that the GMM method will be less effective than in the PCA case: as we can see in the Figure 3.26, the $LL1$-$LL3$ coordinates practically follow a two-dimensional manifold structure in the three-dimensional space. Because of this, modelling this data set with a finite number of Gaussians

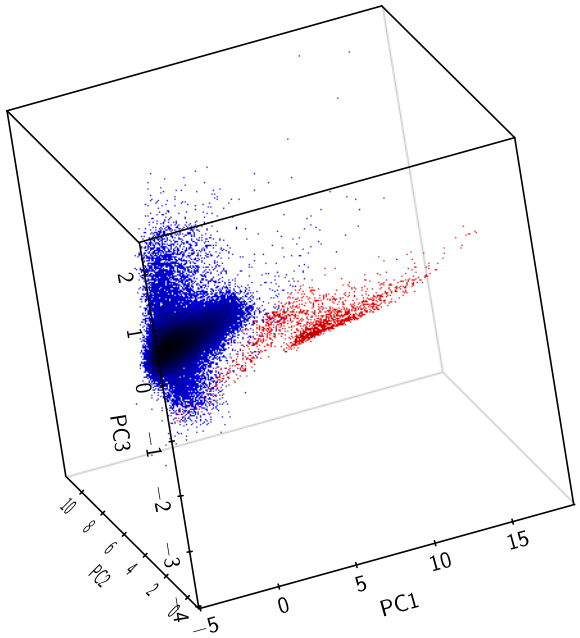


FIGURE 4.3: 3D representation of the PCA data obtained using the validation data set. The blue cluster corresponding to the GMM model is marked in the plot.

will not be very effective. As it can be seen in Table 4.2, the performance of the clustering method is very similar as in the PCA case: the stars are not clearly split, and the rest of the sources can be mostly found in the blue cluster.

4.3.2 Manual clustering

It looks like the clustering methods have not been as effective as we expected. This may be produced because of the initial source or because of the chosen methods. The topology of the problem data set appears to be more complicated than some ellipsoids overlapping. Because of that, we have decided to separate the groups manually, using the naked eye and the data set represented in Figure 3.12. The result is shown in Figure 4.4. The groups have been separated in the most intuitive way possible.

After this three-dimensional representation, we generated the indices plot corresponding to Figure 1.2. This representation is shown in Figure 4.5. As it can be seen, the three lower groups (green, magenta and pink colours) correspond, according to the reference plot, to stars and common galaxies. Moreover, the blue group corresponds to more than one kind of source. This manual separations works well in order to separate the stars from other sources, according to the WISE plot shown in Figure 1.2.

This representation supports the idea that our initial data set should have more sources in order to find more data structure. We would probably double the size of our problem data set in order to improve the results. Specifically, we would need some starburst galaxies, LIRGs, ULIRGs, etc. With this increase, the PCA and LLE methods may transform in a different way our initial data set and the clustering methods may be able to differentiate the clusters more clearly. However, we must remark again that the topology of the data set may just be complicated, and the GMM does not work as expected because of this complicated geometry. Finally, we can see that almost all the sources above the blue group (grey cluster) may be Seyfert galaxies or quasars.

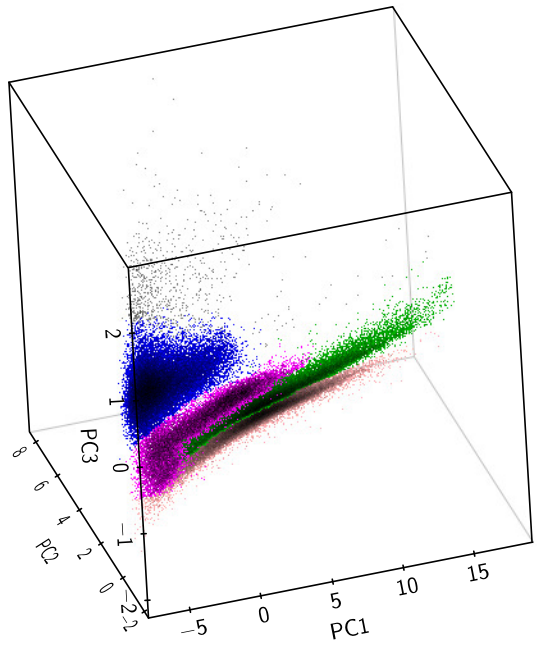


FIGURE 4.4: 3D representation of the manual clustering using the PCA data set

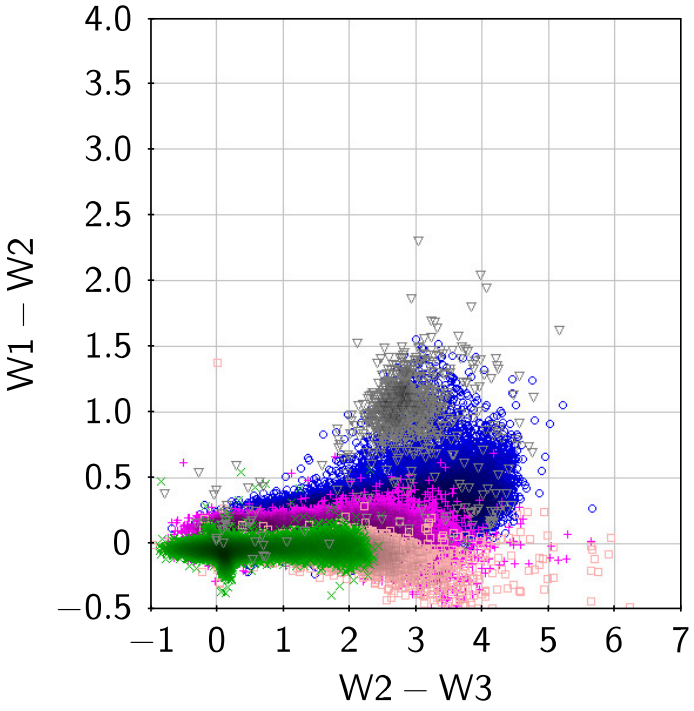


FIGURE 4.5: Colour indices representation for the new groups, corresponding to the plot shown in Figure 1.2

Chapter 5

Conclusions and future work

To conclude this thesis, we explain in this chapter the conclusions reached after all the work realized. We will comment what we have learned with this work, which goals we have reached and which ones not, and how we could continue this work in a future. Clearly, as we must close this project in some point, there are some open questions that need to be asked and, with more work, they could be answered.

5.1 Conclusions

In this project, we have taken a problem data set with a huge amount of sources and we have filtered it in order to run different unsupervised ML methods. We have learned how to deal with a database and different techniques of processing and filtering data using scripts, without the need of analysing each source individually. We have also used five different ML methods, two of them related to dimensionality reduction and the other three to clustering. We have seen that each method follow a different scheme and the main ideas under each algorithm have different natures. In some cases the mathematical insights are easy to follow, and in other cases the methods are more difficult to understand. However, a common point in the used methods is that all of them have a solid mathematical basis. This project has served to learn a lot about a field of Data Science that is getting the attention of experts in different subjects.

Related to the dimensionality reduction methods, it has been proved that are very useful in order to interpret a data set more easily. In this case, we have been able to visualize the information given by six magnitudes in a three-dimensional space (using different approachs according to the model used), reducing by half the number of coordinates. These new coordinates can be easily represented in a three-dimensional plot. Furthermore, they have made more clear the separation between groups.

On the other hand, it is clear that the clustering methods used in this project have not been effective. This fact has made clear that each clustering model has a better or worse performance according to the shape of our data sample. It looks like the K-Means and Mean shift methods are not the most suitable algorithms to use in this project. The GMM appears to provide better results, but it is still inadequate. These results may be better with a larger data set. However, it looks like, in this situation, another clustering-searching method better adapted to the geometry of the groups found would have provided better results. In any case, the methods used are a good point to start the analysis of the problem data set: the logical procedure is to start with the simplest methods and continue studying the most complex ones. Therefore, with the clustering methods we have learned that unsupervised ML may provide unexpected results, but using them is an unbiased and objective way to study a problem data set.

After the discussion of the results in Chapter 4, we can see that if we increased our problem data set or used another clustering methods we could obtain better results. However, this is not something dissapointing. On the contrary, it is hopeful: considering that other studies have shown that this

magnitudes are enough to classificate some kinds of sources, and that the results may be better with a larger data set, this means that the near-infrared and mid-infrared magnitudes used in this project appear promising in order to increase our knowledge in the classification problem.

Of course, one of the learned thing with this project is how to programme a lot of different tasks using Python and the Linux terminal. Not only Astronomy, but almost all the fields into which Physics is divided need to know how to script a list of tasks that you need to do. In this sense, the experience adquired with this project is invaluable.

Finally, one of the things that have been checked in this project (and probably the most important one) is the pressing need for using data science knowledge in Astronomy. We have handled a huge volume of data, and it looks like that even that huge amount of data has not been enough to distinguish some kinds of sources. Therefore, it could be said that the first sentence in Chapter 1 is not entirely true: Astronomy is not changing, Astronomy has changed.

5.2 Future work

As we have said before, this work can be continued in different ways. One of these ways is to work with a larger data set. Probably, doubling the size of it would provide different results. Another way to continue the project, and probably the most suitable one, would be testing other clustering methods. If we study the different algorithms that are available to perform clustering, we can find a wide variety of them. Our aim would be to choose algorithms that do not depend on the shape of the clusters. A good point of start to test new clustering algorithms is the DBSCAN method. DBSCAN model (Density-Based Spatial Clustering of Applications with Noise) is, as the Mean shift method, an algorithm based in density. However, there are some important differences between both methods. DBSCAN model interprets clusters as regions with high density, separated with low density zones. Roughly speaking, this method takes a radius as a parameter and search regions where there are a minimum number of points that are distanced within that radius [39]. The figures analysed with the PCA and LLE data sets suggest that this method may be appropiated. Another interesting method could be Spectral Clustering. This method creates a graph using the similarity between diferent points in the data set. This graph is then embedded in a lower dimension space and other clustering algorithms are applied [35].

Last, a different approach can be done. As we said in Section 2.3, we can use another distance in the different clustering methods, and not only the Euclidean distance. In [9], Baron presents an algorithm that uses Random Forests to obtain what she calls a General Similarity Measure. Broadly speaking, the key step of this method is creating a synthetic data set that keeps the probability distribution for each variable measured in our original data set. These two data sets are labeled as class 1 and class 2 and the Random Forest is trained to differenciate these two groups. A similarity measurement can be defined using these Random Forests and the classification that they perform. The algorithm to obtain these measurements is available in GitHub [1], but some modifications should be done in order to perform clustering with the new distances.

[1] `https://github.com/dalya/WeirdestGalaxies`

Bibliography

[1] H. Karttunen et al. *Fundamental Astronomy*. 5th ed. Berlin Heidelberg New York: Springer, 2007.

[2] I. Gezer et al. "The WISE View of RV Tauri Stars". In: *Monthly Notices of the Royal Astronomical Society* 453 (Nov. 2015), pp. 133–146.

[3] G. Lagache, J. L. Puget, and H. Dole. "Dusty Infrared Galaxies: Sources of the Cosmic Infrared Background". In: *Annual Review of Astronomy and Astrophysics* 43 (Sept. 2005), pp. 727–768.

[4] Z. Y Cai et al. "A Hybrid Model for the Evolution of Galaxies and Active Galactic Nuclei in the Infrared". In: *The Astrophysical Journal* 768.21 (May 2013), p. 24.

[5] E. L. Wright et al. "The Wide-field Infrared Survey Explorer (WISE): Mission Description and Initial On-orbit Performance". In: *The Astronomical Journal* 140.6 (Dec. 2010), pp. 1868–1881.

[6] D. Carrasco et al. "Photometric Classification of Quasars from RCS-2 Using Random Forest". In: *Astronomy & Astrophysics* 584 (Dec. 2015), A44.

[7] LSST. *Large Sypnotic Survey Telescope*. [Online; accessed: 2019-10-07]. 2019. URL: https://www.lsst.org/.

[8] Wikipedia, the free encyclopedia. *Machine Learning*. [Accessed: 2019-10-07]. 2019. URL: https://en.wikipedia.org/wiki/Machine_learning.

[9] D. Baron. "Machine Learning in Astronomy: a Practical Overview". In: (2019). arXiv: 1904.07248.

[10] A. Kovács and I. Szapudi. "Star-Galaxy Separation Strategies for WISE-2MASS All-Sky Infrared Galaxy Catalogue". In: *Monthly Notices of the Royal Astronomical Society* 448 (Mar. 2015), pp. 1305–1313.

[11] M. F. Skrutskie et al. "The Two Micron All Sky Survey (2MASS)". In: *The Astronomical Journal* 131.2 (Feb. 2006), pp. 1163–1183.

[12] R. M. Cutri et al. *Explanatory Supplement to the 2MASS All Sky Data Release and Extended Mission Products*. [Online; accessed: 2019-08-12]. 2006. URL: http://old.ipac.caltech.edu/2mass/releases/allsky/doc/explsup.html.

[13] R. M. Cutri et al. *Explanatory Supplement to the WISE All-Sky Data Release Products*. [Online; accessed: 2019-08-12]. 2015. URL: http://wise2.ipac.caltech.edu/docs/release/allsky/expsup/index.html.

[14] R. M. Cutri et al. *Explanatory Supplement to the AllWISE Data Release Products*. [Online; accessed: 2019-08-12]. 2014. URL: http://wise2.ipac.caltech.edu/docs/release/allwise/expsup/index.html.

[15] ARCHES. *ARCHES - Astronomical Resource Cross-matching for High Energy Studies*. [Online; accessed: 2019-08-15]. 2013. URL: http://www.arches-fp7.eu/arches/.

[16] IRSA. *VO Simple Cone Search*. [Online; accessed: 2019-08-15]. 2018. URL: http://irsa.ipac.caltech.edu/docs/vo_scs.html.

[17] Wikipedia, the free encyclopedia. *FITS*. [Accessed: 2019-08-15]. 2019. URL: http://en.wikipedia.org/wiki/FITS.

[18] T. Oliphant. *NumPy: A Guide to NumPy*. USA: Trelgol Publishing. [Online; accessed: 2019-09-10]. 2006. URL: http://www.numpy.org.

[19] J. D. Hunter. "Matplotlib: a 2D Graphics Environment". In: *Computing in Science & Engineering* 9.3 (June 2007), pp. 90–95.

[20] Astropy Collaboration et al. “Astropy: A community Python package for astronomy”. In: *Astronomy & Astrophysics* 558 (Oct. 2013), A33.

[21] M. B. Taylor. “TOPCAT & STIL: Starlink Table/VOTable Processing Software”. In: *Astronomical Data Analysis Software and Systems XIV* 347 (Dec. 2005), p. 29.

[22] M. B. Taylor. “STILTS - A Package for Command-Line Processing of Tabular Data”. In: *Astronomical Data Analysis Software and Systems XV* 351 (July 2006), p. 666.

[23] M. B. Taylor. *TOPCAT - Tool for OPerations on Catalogues And Tables*. [Online; accessed: 2019-08-19]. 2019. URL: http://www.star.bris.ac.uk/~mbt/topcat/sun253/sun253.html.

[24] M. B. Taylor. *STILTS - Starlink Tables Infrastructure Library Tool Set*. [Online; accessed: 2019-08-19]. 2019. URL: http://www.star.bris.ac.uk/~mbt/stilts/sun256/sun256.html.

[25] T. Boch, F. X. Pineau, and S. Derriere. *CDS xMatch Service Documentation*. [Online; accessed: 2019-08-21]. 2016. URL: http://cdsxmatch.u-strasbg.fr/xmatch/doc/index.html.

[26] T. Budavári and A. S. Szalay. “Probabilistic Cross-identification of Astronomical Sources”. In: *The Astrophysical Journal* 679.1 (May 2008), pp. 301–309.

[27] F. X. Pineau et al. “Probabilistic Multi-catalogue Positional Cross-match”. In: *Astronomy & Astrophysics* 597 (Jan. 2017), A89.

[28] E. Høg et al. “The Tycho-2 Catalogue of the 2.5 Million Brightest Stars”. In: *Astronomy & Astrophysics* 355 (Mar. 2000), pp. L27–L30.

[29] Ž. Ivecić et al. *Statistics, Data Mining, and Machine Learning in Astronomy. A practical Python guide for the analysis of survey data*. Princeton and Oxford: Princeton University Press, 2014.

[30] J. T. Vanderplas et al. “Introduction to AstroML: Machine Learning for Astrophysics”. In: *Conference on Intelligent Data Understanding (CIDU)*. Oct. 2012, pp. 47–54.

[31] F. Pedregosa et al. “Scikit-learn: Machine Learning in Python”. In: *Journal of Machine Learning Research* 12 (Oct. 2011), pp. 2825–2830.

[32] I. T. Jolliffe. *Principal Component Analysis*. 2nd ed. Berlin Heidelberg New York: Springer, 2002.

[33] S. T. Roweis and L. K. Saul. “Nonlinear Dimensionality Reduction by Locally Linear Embedding”. In: *Science* 290.5500 (Dec. 2000), pp. 2323–2326.

[34] J. R. Munkres. *Topology*. Upper Saddle River: Prentice Hall, 2000.

[35] Scikit learn developers. *Scikit-learn User Guide*. [Online; accessed: 2019-09-25]. 2019. URL: https://scikit-learn.org/stable/_downloads/scikit-learn-docs.pdf.

[36] M. Nedrich. *Mean Shift Clustering*. [Online; accessed: 2019-09-27]. 2015. URL: https://spin.atomicobject.com/2015/05/26/mean-shift-clustering/.

[37] F. X. Pineau. *ARCHES WP4 Software Documentation*. [Online; accessed: 2019-08-28]. 2015. URL: http://serendib.unistra.fr/ARCHESWebService/XMatch_soft_doc.pdf.

[38] M. R. Blanton et al. “Sloan Digital Sky Survey IV: Mapping the Milky Way, Nearby Galaxies, and the Distant Universe”. In: *The Astronomical Journal* 154.1 (July 2017), p. 35.

[39] G. Seif. *The 5 Clustering Algorithms Data Scientists Need to Know*. [Online; accessed: 2019-10-07]. 2018. URL: https://towardsdatascience.com/the-5-clustering-algorithms-data-scientists-need-to-know-a36d136ef68.

Appendix I

Implementation of the problem

In this appendix we summarise the programming aspects of the project. We will list the main scripts and files needed to run these programs. These files can be found adjoint to this project: In the case the reader cannot access to the CD where this project has been recorded, we have uploaded the files to GitHub[1], so it is possible to access these information using the URL in the footnote.

The description that we will give here will not be detailed; to know the details about each method, we recommend to read the heading of each script. There, we give a detailed description of the input and output of each method.

TABLE I.1: Summary of the different scripts and programs used to process and filter the initial data set. For each script we include a brief description of their function.

Name	Description
`CoordinateList.txt`	File, in ASCII format, that contains the position (right ascension and declination) of the search region centers.
`DownloadTables.py`	Script that downloads tables from 2MASS and AllWISE catalogs using IRSA's Simple Cone Search. The center of the cones are given by the user. The program also searchs for equal rows.
`NumberOfSources.py`	Script that takes the downloaded tables and analyses the number of sources found in each of the cones. You can also choose an upper and lower threshold for the number of sources in 2MASS and AllWISE regions. The program informs about the number of the fields that surpass the thresholds.
`CoordinateList.fits`	File, in FITS format, that contains the position (right ascension and declination) of the search region centers. It also contains in the first column (FOV) the number of the row.
`JoiningFields.py`	Script that takes a table with positions and joins the tables that overlap. The script is complex, and each part perform a different task. Each part should be executed following the logical order.
`get_area.pl`	PERL script used to compute the area of a sky region.
`MOCorder2pixel.topcat`	Table with the sizes of pixels, needed to compute the are of a sky region.
`ArchesScript.txt`	Script that performs the cross-correlation between two dataset from different catalogues. This script must be run by ARCHES xMatch service. Before running it, the files must be uploaded to this service.
`stilts.jar`	STILTS archive.
`stilts`	Script that executes STILTS.

[1] `https://github.com/pgn95/physics-thesis`

In Table I.1 we summarise the files that we used to test our problem data set and filter it. Some of these files have been written; other files are used as an input; and other files are auxiliary and are used in other original scripts. Concerning this part of the work, there have been written more than 750 source code lines.

TABLE I.2: Summary of the different scripts and programs used to perform ML techniques. For each script we include a brief description of their function.

Name	Description
`ExtractTable.py`	Script that creates the table with the crosscorrelated objects from 2MASS and AllWISE catalogs. In order to do that, it takes the three tables obtained from xMatch and uses the information contained in them.
`DimensionalityReduction.py`	Script that takes a data set with photometric information and applies different machine learning techniques. The data set must be in the directory of work. These techniques apply dimensionality reduction. It also transforms the data with labels.
`Clustering.py`	Script that takes a data set with photometric information and applies different machine learning techniques. The data set must be in the directory of work. These techniques apply clustering. It also applies the techniques to the labeled data set.
`sdsstmasswisegood_corralra.fit`	Verification data set.

Finally, we show in Table I.2 the descriptions of the scripts related to the studied ML techniques. The three programmed Python scripts were written almost 1000 new source code lines. In total, in this project almost 1750 code lines have been written.

Before ending, in the GitHub repository it has been created the directories necessary to execute without problems the three Python methods in Table I.2. The parameters are the ones indicated in the names of some folders (we use a probability threshold of 0.69133 and the ABC flags for both 2MASS and AllWISE).