UNIVERSITÀ DI PISA

ARTIFICIAL INTELLIGENCE AND DATA ENGINEERING
project report for the examination
of
Data Mining and Machine Learning

Student:

Alessandro Diana

AA. 2019/2020

Contents

1 Application 1

1.1 Description 1

1.2 Requirement 1

1.2.1 Main actors 1

1.2.2 Functional requirements 1

1.2.3 Non-functional requirements 2

1.2.4 Use Case 2

1.2.5 Class analysis 2

1.2.6 Data model 2

1.3 System architecture 4

1.3.1 Client side 4

1.3.2 Server side 4

2 Machine Learning 5

2.1 Introduction 5

2.2 Dataset 5

2.3 Pre-processing 5

2.3.1 Data cleaning 5

2.3.2 Features reduction 6

2.3.3 Data transformation 7

2.4 Models evaluation and selection 8

2.5 Conclusion 14

3 Implementation 15

Bibliography 16

List of Figures

1 UML diagram of the use cases of the application. 2
2 UML diagram of the class analysis. 3
3 Example of user table. 3
4 Example of celestial_bodies table. 3
5 Rappresentation of system architecture. 4
6 Image that shows the distribution in the dataset. 7
7 Tab. of the result for J48 classifier. 9
8 Tab. conatining the results for JRIP and Naive Bayes classifiers. 10
9 Tab. of the result for K-NN classifier. 11
10 Tab. of the result for ensemble methods. 12
11 Tab. of an example of comparison between results of classifiers with and without methods for rebalancing the dataset. 13
12 Tab. which shows the result of t-test. 14

1 Application

1.1 Description

Astreo is an application whose main purpose is to help users in observing the spectral data of celestial bodies.

The user is provided with a large database containing spectral data of celestial bodies from observatories.
In addition, each user can save data of the objects they observe.

The application can independently classify new objects observed by users into three different classes: galaxy, quasar, star.

1.2 Requirement

1.2.1 Main actors

The application can be used by three different types of actors:

- Anonymous users: only able to register or log in;
- Registered users: they are the main users of the application. They can view spectral data of global celestial bodies (from observatories), save, display, delete data on the celestial bodies observed by them and have a classification of their objects;
- Administrator: has the role of inserting new celestial bodies, coming from observatories.

1.2.2 Functional requirements

- The application must allow a new user to register;
- The application must allow a registered user to log in and log out;
- The anonymous user must not have access to the functionality of the application, excluding registration;
- The registered user must be able to see the list of global celestial bodies;
- The registered user must be able to see the list of his/her own celestial bodies;
- The registered user must be able to manage his/her own saved celestial bodies, adding new ones or deleting existing ones;
- The registered user must be able to receive the classification of one of his objects, by specifying its parameters before it is added to his list of saved celestial bodies;
- The registered user must not have the possibility of modifying data on global celestial bodies;
- The administrator must be able to add and delete celestial bodies from the global list;

1.2.3 Non-functional requirements

- The application must use and have within it at least one machine learning algorithm.

1.2.4 Use Case

Figure 1 shows the use case of the application.

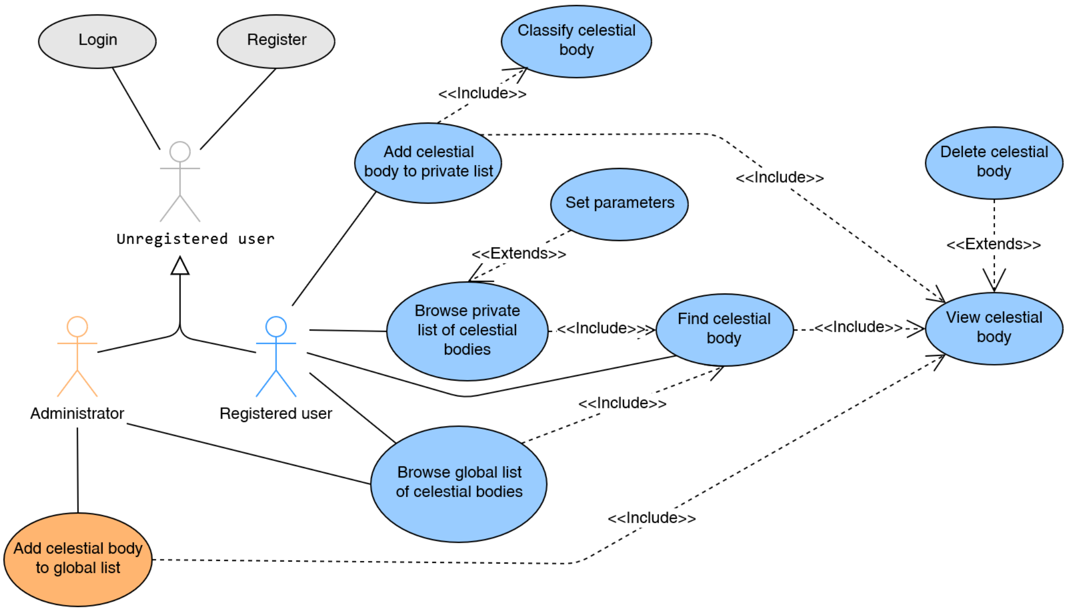


Figure 1: UML diagram of the use cases of the application.

1.2.5 Class analysis

Figure 2 shows the application entities and their relationships.

Users are characterised by a username and a password. A user can be a registered user or an administrator. Each registered user can be linked to none or more celestial bodies and each celestial body can be linked to none or more registered users.

Each celestial body is characterised by its coordinates, its spectral data, its class and the user who added it to the database.

1.2.6 Data model

I have decided to use a SQL Database to store the data, consisting of two figure 3-4.

- user: is the table representing the users in the database. Its attributes are: iduser, username, psw, admin. the password is encrypted with MD5. Admin indicates whether a user is an administrator or a registered user.

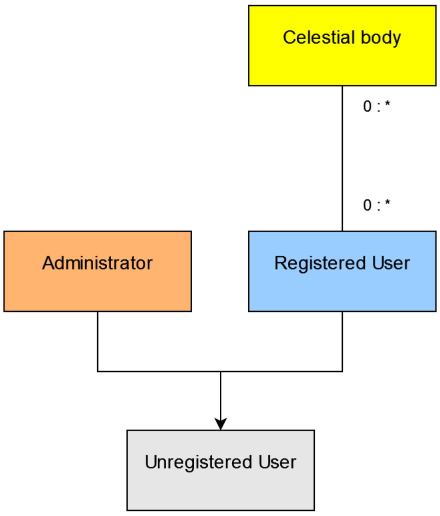
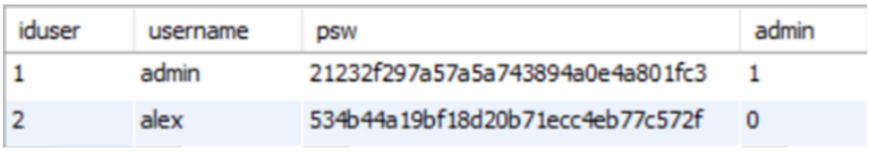


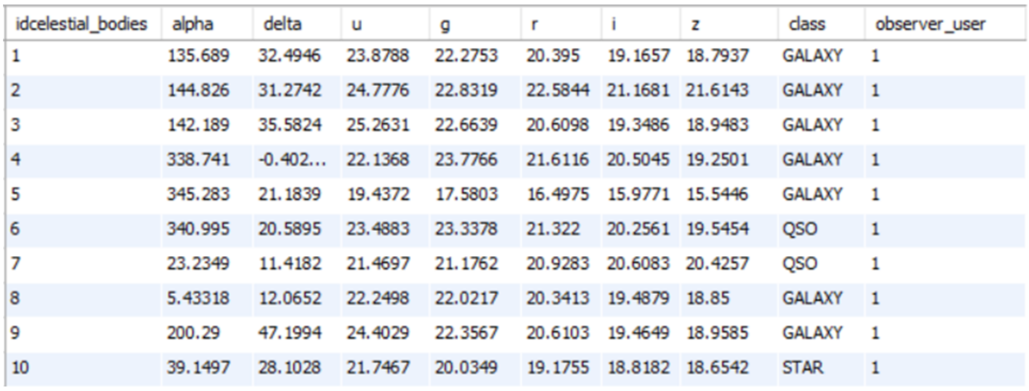
Figure 2: UML diagram of the class analysis.

- celestial_bodies: is the table rapresenting the celestial bodies in the database. Its attributes are: idcelestial_bodies, alpha, delta, u, g, r, i, z, class, observer_user. Each celestial body in database is rapresented by its coordinates, spectral data, class and id of the user who observed it.



iduser	username	psw	admin
1	admin	21232f297a57a5a743894a0e4a801fc3	1
2	alex	534b44a19bf18d20b71ecc4eb77c572f	0

Figure 3: Example of user table.



idcelestial_bodies	alpha	delta	u	g	r	i	z	class	observer_user
1	135.689	32.4946	23.8788	22.2753	20.395	19.1657	18.7937	GALAXY	1
2	144.826	31.2742	24.7776	22.8319	22.5844	21.1681	21.6143	GALAXY	1
3	142.189	35.5824	25.2631	22.6639	20.6098	19.3486	18.9483	GALAXY	1
4	338.741	-0.402...	22.1368	23.7766	21.6116	20.5045	19.2501	GALAXY	1
5	345.283	21.1839	19.4372	17.5803	16.4975	15.9771	15.5446	GALAXY	1
6	340.995	20.5895	23.4883	23.3378	21.322	20.2561	19.5454	QSO	1
7	23.2349	11.4182	21.4697	21.1762	20.9283	20.6083	20.4257	QSO	1
8	5.43318	12.0652	22.2498	22.0217	20.3413	19.4879	18.85	GALAXY	1
9	200.29	47.1994	24.4029	22.3567	20.6103	19.4649	18.9585	GALAXY	1
10	39.1497	28.1028	21.7467	20.0349	19.1755	18.8182	18.6542	STAR	1

Figure 4: Example of celestial_bodies table.

1.3 System architecture

The system is organised with a client-server architecture, as shown in figure 5.

1.3.1 Client side

The client side includes the GUI, the machine learning part, and the interface with the server. The GUI is implemented with python Tkinter. The machine learning part is realised using scikit learn in python and deals with creating, training and using the model for prediction. The interface to the MySQL server is developed in python and takes care of passing data and requests between the application and the database.

1.3.2 Server side

The server side consists of the SQL database made with MySQL and its purpose is to store the data of the celestial bodies and users.

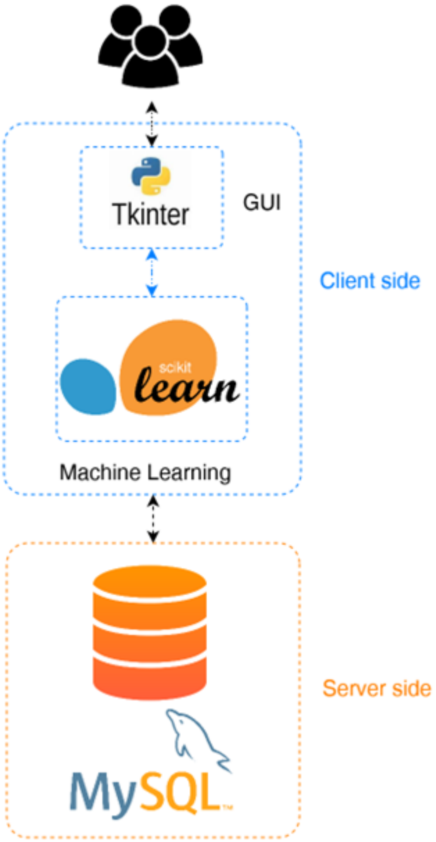


Figure 5: Rappresentation of system architecture.

2 Machine Learning

2.1 Introduction

For astronomy, the detection, observation and understanding of celestial bodies is of fundamental importance in order to understand the conformation and history of the universe. With the advancement of technology, hobbyists have been able to observe space better and better with tools that can be kept at home, such as telescopes that integrate equipment and software to enable images and analysis of the celestial vault that were previously only possible in research contexts.

An important method for understanding celestial bodies is the spectrographic analysis of the light coming from them. Spectrophotometry is the analysis of electromagnetic spectra that fall in the visible, from near ultraviolet to near-infrared. This can be done by amateurs by using special filters attached to telescopes.

The aim of my application is to provide support to astronomy fans who wish to observe the spectral characteristics of celestial bodies through a classification of them. The desired classification will discriminate the objects observed by users into three distinct classes: galaxies[1], stars[2], and quasars[3].

2.2 Dataset

The dataset used to train the classification model is a part of Release 17 of the Sloan Digital Sky Survey (SDSS)[4].
The SDSS is a survey whose aim is to map the universe in collaboration with more than 25 organisations in worldwide.

In more than 20 years of work, it has succeeded in creating the largest 3D map of the visible Universe ever, thanks to the detailed analysis of millions of objects, including spectral data for more than 930,000 galaxies and 120,000 quasars [5].

The dataset used can be found on Kaggle at the link: https://www.kaggle.com/fedesoriano/stellar-classification-dataset-sdss17#
Its volume is approximately 17 MB.

The dataset used includes 100,000 observations of celestial bodies and for each one there is information on its spectral characteristics, technical observation data, and the class to which it belongs.
Each celestial body is classified as a star, galaxy or quasar.

2.3 Pre-processing

2.3.1 Data cleaning

In this dataset, there aren't objects with missing values or duplicates.
There is only one obvious case of noise which is represented by an object that has values for the u,g, and z filters of -9999.

It is not a plausible value moreover the dataset has on average values between 10 and 30. Furthermore, these values are all exactly equal in the u,g, and z fields, which suggests that the SDSS adopted a method that replaces and identifies missing values or errors in the measurements.

This value can be substituted by using value replacement techniques such as:

- fill the u,g, and z fields with the average value for that feature in the DB;
- fill the u,g, and z fields with the average value for that feature of the objects in the DB belonging to the same class.

These techniques are not guaranteed to find a correct value for the object. For this reason, together with the fact that it was only 1 item over 100,000 and three of eight values are to be replaced, I have chosen to discard it.

2.3.2 Features reduction

As mentioned in the paragraph above, there are features in the chosen dataset related to the method of observing celestial bodies. More precisely, features related to the telescope used, the software analysis, and the data storage method used by the SDSS.

To better understand this data, I will briefly explain the method used for observation. To simplify, observation and analysis are carried out using various instruments: telescopes, spectrographs, a focal plane system, and cameras.

The spectra of many objects are taken simultaneously. This is possible because the spectrographs are connected by fibre optic cables to an aluminium plate on the focal plane of the telescope.
Each plate corresponds to a specific portion of the sky and is pre-drilled with holes corresponding to the positions of the objects in that area, which means that each area needs its own plate. Each hole on each plate corresponds to an object in the sky. Optical fibres inserted in each hole carry light from the focal plane to the spectrograph pseudoslit.

The nine features related to the observation system are:

- obj_ID: Object Identifier, the unique value that identifies the object in the archive server (CAS);
- run_ID: Run Number used to identify the specific scan;
- rereun_ID: Rerun Number to specify how the image was processed;
- cam_col: Camera column to identify the scanline within the run;
- field_ID: Field number to identify each field;
- spec_obj_ID: Unique ID used for optical spectroscopic objects;
- plate: plate ID, identifies each plate in SDSS;
- MJD: Modified Julian Date, indicate when a data was taken;

- fiber_ID: fiber ID that identifies the fiber that pointed the light at the focal plane in each observation.

These nine features are related to the observation system and equipment used and are not related to the nature of the celestial bodies (galaxies, stars or quasars).
In addition, any correlation of these features with the classes could only have worsened the performance of the classifier which only has to classify with spectral data. These features are related to the SDSS observation system and not to the observed object, so they are useless to classify the observed object itself.
For these reasons, I have decided to consider them not useful for the learning phase and to remove them.

Another feature that has been removed, is the measure of the redshift.
Redshift [6] is the phenomenon for which electromagnetic radiation emitted by a moving object, with respect to an observer, will present a longer wavelength on its arrival than its emission.
Its opposite is the Blueshift and both are examples of the Doppler effect.

In astronomy, this data is very important and is influenced by many complex factors. It can be used to calculate displacement speeds, to understand if an object is approaching or moving away, to understand the distance and age of an observed object and for many other purposes.
Due to its very complex nature, the observation and calculation of the redshift cannot be done except with elaborate techniques and knowledge that are not accessible by users at home.

Therefore, I have decided to remove the Redshift.

2.3.3 Data transformation

After data cleaning and features reduction, the dataset appears like in the figure 6.
The dataset is neither balanced nor strongly unbalanced.

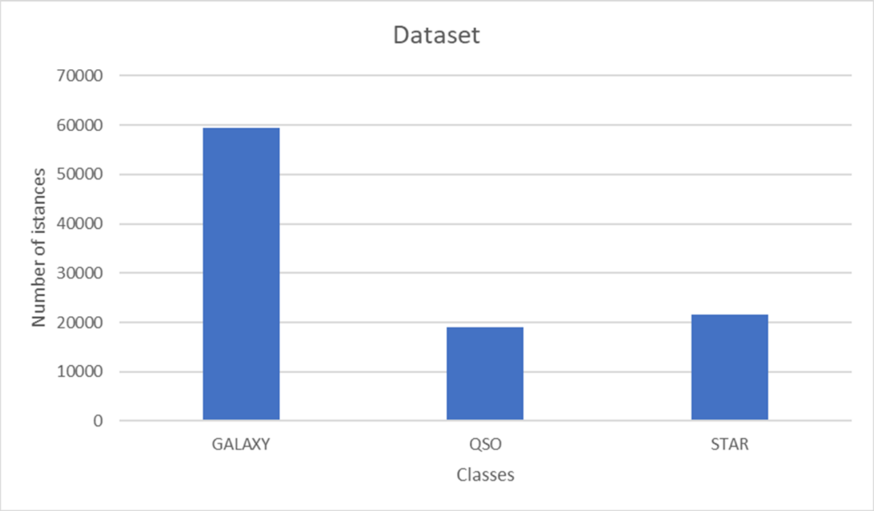


Figure 6: Image that shows the distribution in the dataset.

The dataset is composed by 99999 objects and each of them is described by these eight features:

- alpha: Right Ascension angle [7];
- delta: Declination angle [8];
- u: Ultraviolet filter in the photometric system;
- g: Green filter in the photometric system;
- r: Red filter in the photometric system;
- I: Near Infrared filter in the photometric system;
- z: Infrared filter in the photometric system;
- class: object class (galaxy, star, or quasar object).

The first two attributes indicate the position of the object in the celestial sphere. The value of the attributes u, g, r, i, and z represent the spectral data of the object in the ugriz photometric system.

A further normalisation step was made for the K-NN algorithm, to give equal weight to each attribute when calculating the Euclidean distance. The z-score normalisation was chosen since it is less affected by outliers.

A further step of discretization was made for the naive Bayes algorithm, to facilitate the calculation of probabilities for prediction. The Fayyad & Irani's discretization algorithm was chosen, which is a type of supervised algorithm that generates a better discretization according with the classes.

2.4 Models evaluation and selection

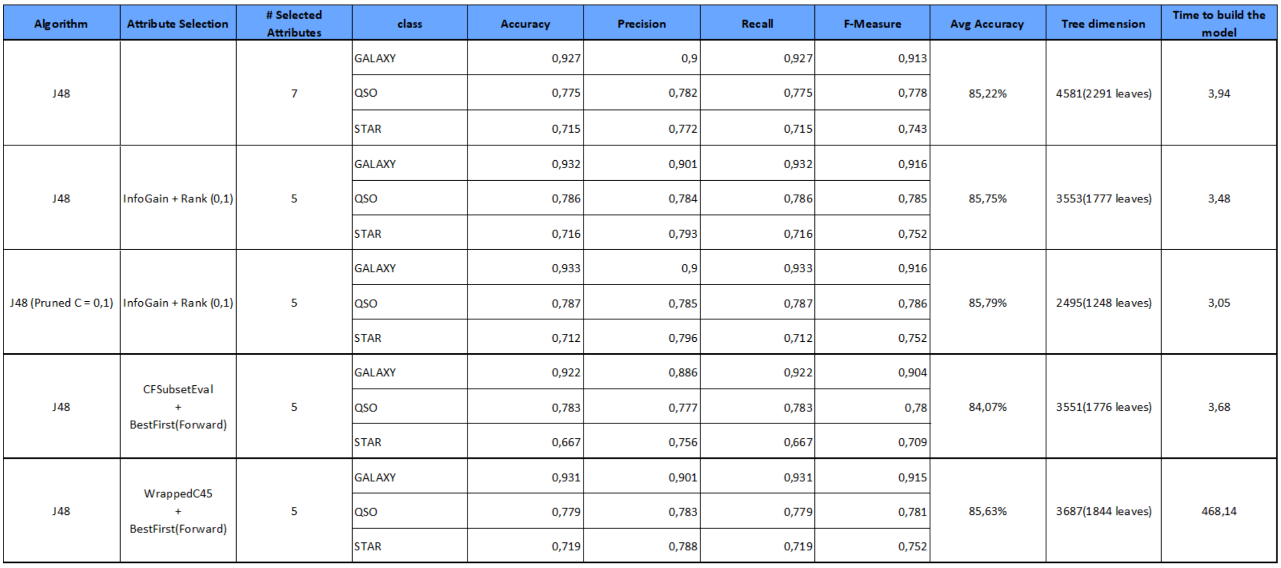
The problem we have to solve is a multiclass classifier, more precisely with three classes. To solve this problem, we have to build and train a classifier model.
To find the model that gives the best results, several classifiers were tried out and their performances were evaluated on our dataset.

In the first part of the analysis, several classifiers were tried with various attribute selection algorithms using 10-fold cross-validation. The models tested are J48 (an implementation of the C4.5 tree algorithm) 7, JRIP (a propositional rule learner) 8, Naive Bayes 8, iBk (K-nearest neighbours classifier) 9 and ensemble methods.

Each classifier was tested without attribute selection and with different attribute selection algorithms.
The main algorithms for selecting attributes used are:

- InfoGainAttributeEval: evaluates the value of an attribute by measuring the information gain with respect to the class;
- CFSubsetEval: evaluates the value of a subset of attributes by considering the individual predictive ability of each feature along with the degree of redundancy between them;

- CorrelationAttributeEval: measures the correlation between attributes and class.



Algorithm	Attribute Selection	# Selected Attributes	class	Accuracy	Precision	Recall	F-Measure	Avg Accuracy	Tree dimension	Time to build the model
J48		7	GALAXY	0,927	0,9	0,927	0,913	85,22%	4581(2291 leaves)	3,94
			QSO	0,775	0,782	0,775	0,778			
			STAR	0,715	0,772	0,715	0,743			
J48	InfoGain + Rank (0,1)	5	GALAXY	0,932	0,901	0,932	0,916	85,75%	3553(1777 leaves)	3,48
			QSO	0,786	0,784	0,786	0,785			
			STAR	0,716	0,793	0,716	0,752			
J48 (Pruned C = 0,1)	InfoGain + Rank (0,1)	5	GALAXY	0,933	0,9	0,933	0,916	85,79%	2495(1248 leaves)	3,05
			QSO	0,787	0,785	0,787	0,786			
			STAR	0,712	0,796	0,712	0,752			
J48	CFSubsetEval + BestFirst(Forward)	5	GALAXY	0,922	0,886	0,922	0,904	84,07%	3551(1776 leaves)	3,68
			QSO	0,783	0,777	0,783	0,78			
			STAR	0,667	0,756	0,667	0,709			
J48	WrappedC45 + BestFirst(Forward)	5	GALAXY	0,931	0,901	0,931	0,915	85,63%	3687(1844 leaves)	468,14
			QSO	0,779	0,783	0,779	0,781			
			STAR	0,719	0,788	0,719	0,752			

Figure 7: Tab. of the result for J48 classifier.

Figure 9 shows that five selected attributes proved to be a good number for reducing the dimensionality of the features and selecting the most significant ones for most of the models. Decreasing the number of selected features even further would have worsened performance, as we would have lost important information for correct classification.

From the experimental results, considering five selected features, it can see that:

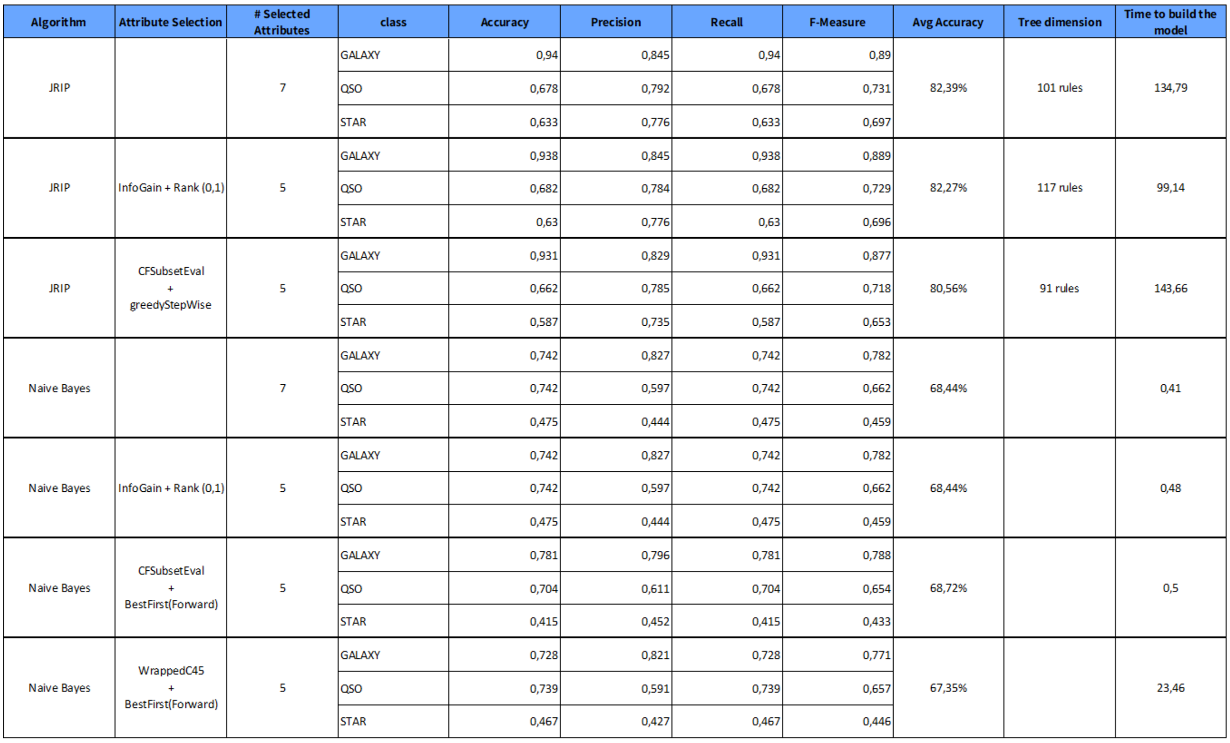
- between InfoGain and CorrelationAttributeEval only the order changed;
- between CFSubsetEval and InfoGain (or CorrelationAttributeEval) both the order and the selected attributes changed.

The various models were compared by looking:

- number of attributes selected;
- Accuracy: (TP + TN)/(TP + TN + FP + FN);
- Precision: TP/(TP +FP);
- Recall: TP/P ;
- F-measure: 2*P*R/(P + R) ;
- dimension of tree (in the tree classifier);
- number of rules (in the rules classifier);
- time to build the model.

Figure 7 shows that the J48 algorithm performed very well. There are no major differences in results between the different configurations tested.

In figure 8 we can see the results of JRIP and Naive Bayes. In both algorithms we can see how the features selection has not led to better results, indeed in many cases, it has worsened them.
My explanation for this behavior is that all the information given by the features is needed for these algorithms to work properly. Therefore, by eliminating the features considered less important, you are not going to give a boost to the classification performance because you are not going to remove superfluous and perhaps deleterious information but you are going to remove information useful for correct classification.



Algorithm	Attribute Selection	# Selected Attributes	class	Accuracy	Precision	Recall	F-Measure	Avg Accuracy	Tree dimension	Time to build the model
JRIP		7	GALAXY	0,94	0,845	0,94	0,89	82,39%	101 rules	134,79
			QSO	0,678	0,792	0,678	0,731			
			STAR	0,633	0,776	0,633	0,697			
JRIP	InfoGain + Rank (0,1)	5	GALAXY	0,938	0,845	0,938	0,889	82,27%	117 rules	99,14
			QSO	0,682	0,784	0,682	0,729			
			STAR	0,63	0,776	0,63	0,696			
JRIP	CFSubsetEval + greedyStepWise	5	GALAXY	0,931	0,829	0,931	0,877	80,56%	91 rules	143,66
			QSO	0,662	0,785	0,662	0,718			
			STAR	0,587	0,735	0,587	0,653			
Naive Bayes		7	GALAXY	0,742	0,827	0,742	0,782	68,44%		0,41
			QSO	0,742	0,597	0,742	0,662			
			STAR	0,475	0,444	0,475	0,459			
Naive Bayes	InfoGain + Rank (0,1)	5	GALAXY	0,742	0,827	0,742	0,782	68,44%		0,48
			QSO	0,742	0,597	0,742	0,662			
			STAR	0,475	0,444	0,475	0,459			
Naive Bayes	CFSubsetEval + BestFirst(Forward)	5	GALAXY	0,781	0,796	0,781	0,788	68,72%		0,5
			QSO	0,704	0,611	0,704	0,654			
			STAR	0,415	0,452	0,415	0,433			
Naive Bayes	WrappedC45 + BestFirst(Forward)	5	GALAXY	0,728	0,821	0,728	0,771	67,35%		23,46
			QSO	0,739	0,591	0,739	0,657			
			STAR	0,467	0,427	0,467	0,446			

Figure 8: Tab. conatining the results for JRIP and Naive Bayes classifiers.

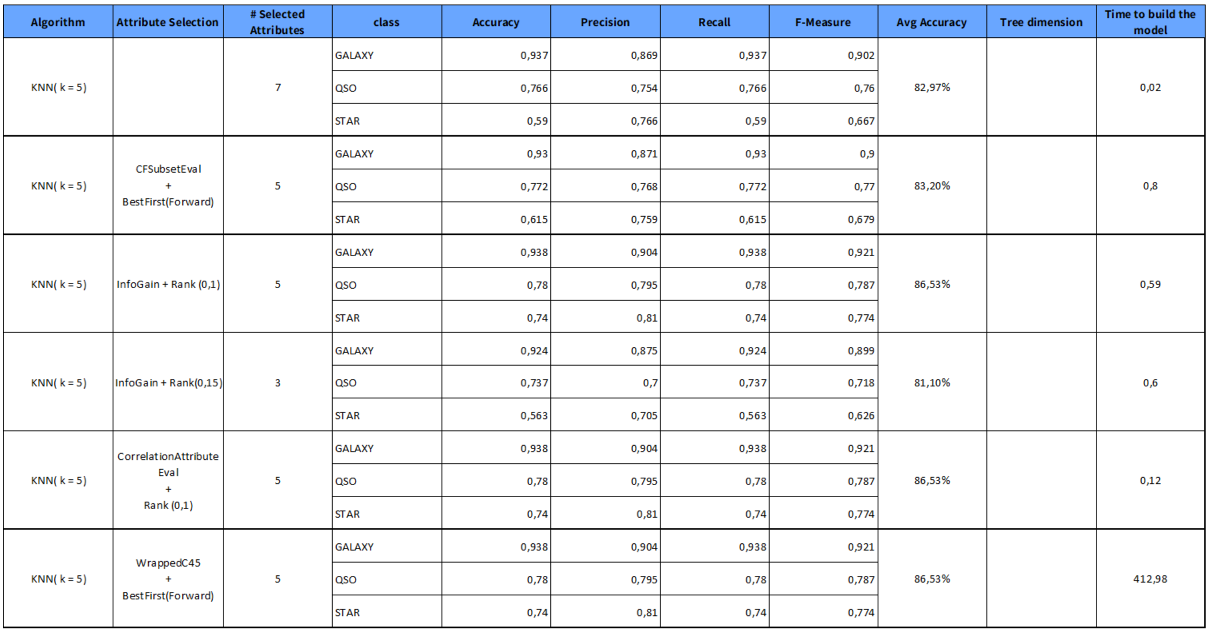
In general, we can see that JRIP has good results, even if lower than those obtained with the J48, while Naive Bayes has the worst results especially inherent in the star class.

In figure 9 we can see the results of K-NN. This algorithm has good performances, which are improved by the features selection (except the case in which only 3 features are taken).

Ensemble methods were also tested, which are techniques to improve the results of the classifiers.
The techniques tested are:

- Bagging: averaging the prediction over a set of classifiers;

- AdaBoost: weighted vote with a set of classifiers;
- RandomForest: several decision tree classifiers are created, each one performs a prediction and the most popular class is returned.



Algorithm	Attribute Selection	# Selected Attributes	class	Accuracy	Precision	Recall	F-Measure	Avg Accuracy	Tree dimension	Time to build the model
KNN(k = 5)		7	GALAXY	0,937	0,869	0,937	0,902	82,97%		0,02
			QSO	0,766	0,754	0,766	0,76			
			STAR	0,59	0,766	0,59	0,667			
KNN(k = 5)	CFSubsetEval + BestFirst(Forward)	5	GALAXY	0,93	0,871	0,93	0,9	83,20%		0,8
			QSO	0,772	0,768	0,772	0,77			
			STAR	0,615	0,759	0,615	0,679			
KNN(k = 5)	InfoGain + Rank (0,1)	5	GALAXY	0,938	0,904	0,938	0,921	86,53%		0,59
			QSO	0,78	0,795	0,78	0,787			
			STAR	0,74	0,81	0,74	0,774			
KNN(k = 5)	InfoGain + Rank(0,15)	3	GALAXY	0,924	0,875	0,924	0,899	81,10%		0,6
			QSO	0,737	0,7	0,737	0,718			
			STAR	0,563	0,705	0,563	0,626			
KNN(k = 5)	CorrelationAttribute Eval + Rank (0,1)	5	GALAXY	0,938	0,904	0,938	0,921	86,53%		0,12
			QSO	0,78	0,795	0,78	0,787			
			STAR	0,74	0,81	0,74	0,774			
KNN(k = 5)	WrappedC45 + BestFirst(Forward)	5	GALAXY	0,938	0,904	0,938	0,921	86,53%		412,98
			QSO	0,78	0,795	0,78	0,787			
			STAR	0,74	0,81	0,74	0,774			

Figure 9: Tab. of the result for K-NN classifier.

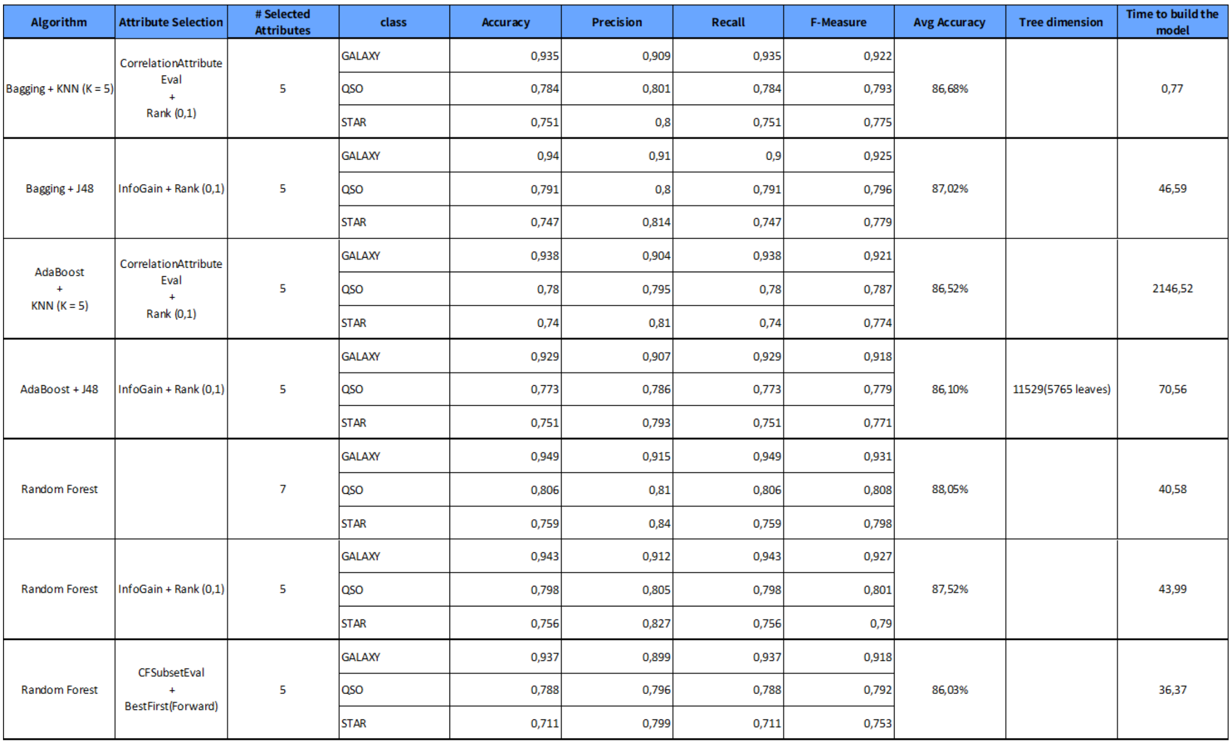
The ensemble methods were done on the algorithms that showed the best results: J48 and K-NN.
For the J48 I have chosen the configuration that uses InfoGain and that has more pruning. This configuration was chosen not only for the best results, but because it had a smaller tree and a shorter model build time.

For the K-NN there are three configurations that have the exact same results. These tree configurations are obtained by different feature selection algorithms: InfoGain, CorrelationAttributeEval, and WrappedC45.
I chose the configuration with CorrelationAttributeEval because it has a much shorter model build time than the others.

As the figure 10 shows, both the Bagging and AdaBoost have led to very small improvements, and in one case small decreases.

RandomForest is the method with the best results of all those tried.

From the tables seen until now, not only the average accuracies but also the precision, recall, and F-measure values for each of the classes were analyzed.
This was done to get a better idea of the behavior of the classifier for each individual class as the dataset is not perfectly balanced.



Algorithm	Attribute Selection	# Selected Attributes	class	Accuracy	Precision	Recall	F-Measure	Avg Accuracy	Tree dimension	Time to build the model
Bagging + KNN (K = 5)	CorrelationAttribute Eval + Rank (0,1)	5	GALAXY	0,935	0,909	0,935	0,922	86,68%		0,77
			QSO	0,784	0,801	0,784	0,793			
			STAR	0,751	0,8	0,751	0,775			
Bagging + J48	InfoGain + Rank (0,1)	5	GALAXY	0,94	0,91	0,9	0,925	87,02%		46,59
			QSO	0,791	0,8	0,791	0,796			
			STAR	0,747	0,814	0,747	0,779			
AdaBoost + KNN (K = 5)	CorrelationAttribute Eval + Rank (0,1)	5	GALAXY	0,938	0,904	0,938	0,921	86,52%		2146,52
			QSO	0,78	0,795	0,78	0,787			
			STAR	0,74	0,81	0,74	0,774			
AdaBoost + J48	InfoGain + Rank (0,1)	5	GALAXY	0,929	0,907	0,929	0,918	86,10%	11529(5765 leaves)	70,56
			QSO	0,773	0,786	0,773	0,779			
			STAR	0,751	0,793	0,751	0,771			
Random Forest		7	GALAXY	0,949	0,915	0,949	0,931	88,05%		40,58
			QSO	0,806	0,81	0,806	0,808			
			STAR	0,759	0,84	0,759	0,798			
Random Forest	InfoGain + Rank (0,1)	5	GALAXY	0,943	0,912	0,943	0,927	87,52%		43,99
			QSO	0,798	0,805	0,798	0,801			
			STAR	0,756	0,827	0,756	0,79			
Random Forest	CFSubsetEval + BestFirst(Forward)	5	GALAXY	0,937	0,899	0,937	0,918	86,03%		36,37
			QSO	0,788	0,796	0,788	0,792			
			STAR	0,711	0,799	0,711	0,753			

Figure 10: Tab. of the result for ensemble methods.

Furthermore, it can be observed that the various models classify objects belonging to the predominant class (galaxy) better and the minority classes (star and quasar) worse.

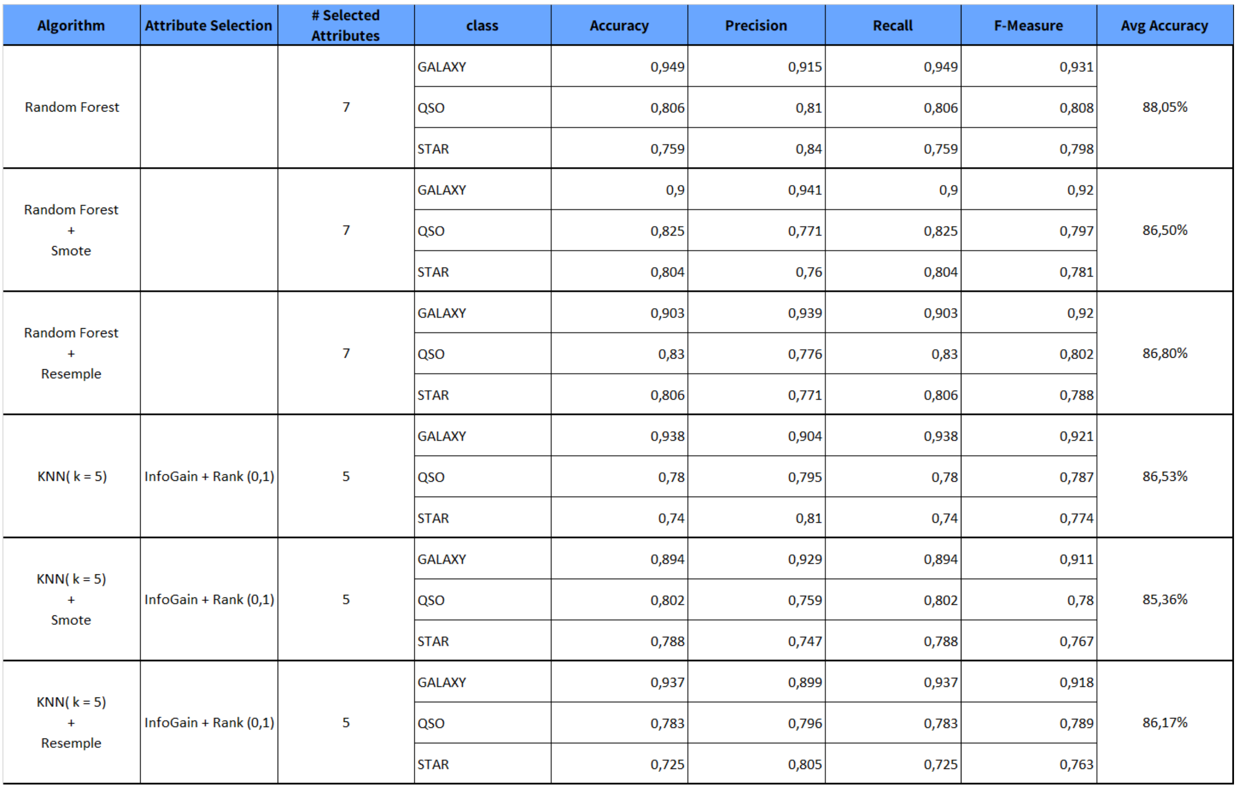
Thus, I have tried to apply the best models on the dataset after having made it perfectly balanced using the smote and resemble technique.
Smote is a technique of over-sampling which, having chosen a value for k, inserts a synthetic object along the line segments joining any or all of the k minority class nearest neighbors.

With both balancing techniques, classifiers with worse results were obtained.
In particular, as shown in figure 11, a slight improvement was obtained in the classification of the minority classes at the cost of greater deterioration in the performance of the Galaxy class.

It can be noticed that the order of the best predicted classes has remained unchanged, the objects belonging to the galaxy class remain the best predicted while the star ones remain the worst.

An explanation for this is that the difficulty in classifying the minority classes, especially in the star class, is not only due to the imperfect balance of the dataset but to an objective greater difficulty of classification. Another explanation could be that the rebalancing techniques are not effective enough.

Since there are no particular constraints on a minimum accuracy threshold for the various classes and for the reasons explained above, I have decided to use the classifiers obtained from the non-rebalanced dataset, which have better overall performance.



Algorithm	Attribute Selection	# Selected Attributes	class	Accuracy	Precision	Recall	F-Measure	Avg Accuracy
Random Forest		7	GALAXY	0,949	0,915	0,949	0,931	88,05%
			QSO	0,806	0,81	0,806	0,808	
			STAR	0,759	0,84	0,759	0,798	
Random Forest + Smote		7	GALAXY	0,9	0,941	0,9	0,92	86,50%
			QSO	0,825	0,771	0,825	0,797	
			STAR	0,804	0,76	0,804	0,781	
Random Forest + Resemple		7	GALAXY	0,903	0,939	0,903	0,92	86,80%
			QSO	0,83	0,776	0,83	0,802	
			STAR	0,806	0,771	0,806	0,788	
KNN(k = 5)	InfoGain + Rank (0,1)	5	GALAXY	0,938	0,904	0,938	0,921	86,53%
			QSO	0,78	0,795	0,78	0,787	
			STAR	0,74	0,81	0,74	0,774	
KNN(k = 5) + Smote	InfoGain + Rank (0,1)	5	GALAXY	0,894	0,929	0,894	0,911	85,36%
			QSO	0,802	0,759	0,802	0,78	
			STAR	0,788	0,747	0,788	0,767	
KNN(k = 5) + Resemple	InfoGain + Rank (0,1)	5	GALAXY	0,937	0,899	0,937	0,918	86,17%
			QSO	0,783	0,796	0,783	0,789	
			STAR	0,725	0,805	0,725	0,763	

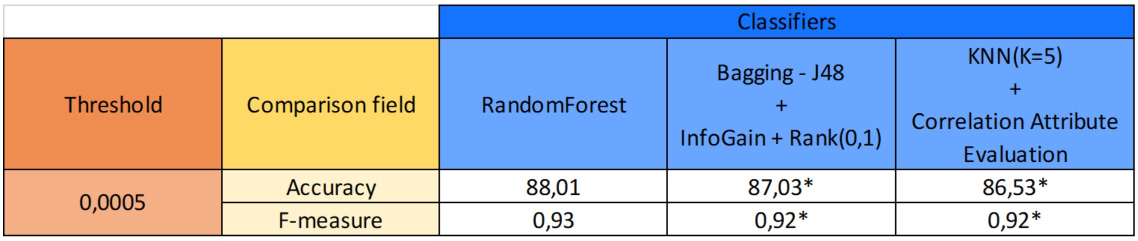
Figure 11: Tab. of an example of comparison between results of classifiers with and without methods for rebalancing the dataset.

To determine if one model was better than the others with statistical significance, the models with the highest performance were chosen to perform the t-test with pairwise comparison for each 10-fold cross-validation round, iterated five times.

The top three algorithms chosen are: Random Forest, bagging with J48 + InfoGain, and K-NN + Correlation Attribute Evaluation.
K-NN with bagging is better than K-NN without bagging. I have chosen the latter because the performances between the two are very similar, with a difference of 0.15 in average accuracy, and it is simpler and faster.

The t-test was done with different threshold values between 0.05 and 0.005. The results obtained with the different threshold values are equal. Figure 12 shows the results with the lowest threshold value.

The t-tests were performed comparing both the average accuracy and the f-measure, given the non-balancing of the dataset. In both measures, the test has shown that Random Forest is statistically better than the others examined.



		Classifiers		
Threshold	Comparison field	RandomForest	Bagging - J48 + InfoGain + Rank(0,1)	KNN(K=5) + Correlation Attribute Evaluation
0,0005	Accuracy	88,01	87,03*	86,53*
	F-measure	0,93	0,92*	0,92*

Figure 12: Tab. which shows the result of t-test.

At the end of this phase of analysis, I have decided to adopt Random Forest without features selection as the classifier.

2.5 Conclusion

Several steps were necessary to obtain the classifier for our application.

On the dataset, data cleansing, feature reduction, and for some algorithms also normalisation and discretization were carried out.

To find the best one, several algorithms were tried: J48, JRIP, Naive Bayes, K-NN, Bagging, AdaBoost, and Random Forest with different attribute selection algorithms.

Statistical tests were also carried out to see if there was a better algorithm even with statistical evidence. In the end, the Random Forest algorithm was the best, achieving an average accuracy of 88%.

3 Implementation

The application code is divided into three files:

- AstreoGUI: this file contains the GUI realised by Python Tkinter. It is the main file and manages the interface with the user and all the operations required for the correct functioning of the application. It contains a Classifier object, which enables the machine learning functionality, and a DB_connect object, which manages the interactions with the server;

- Classifier: in this file, there is the Classifier class, realised in Python. This class implements the RandomForest classifier. It performs all the necessary preprocessing operations on the dataset and model training. Through the 'predict' method you can receive the classification of an object passed as a parameter;

- DB_connect: this file contains the class DB_connect realised in Python. This class is responsible for interfacing with the MySQL database. It contains methods to open and close the connection; make all necessary requests to the application; and pass data between the application and the database.

The complete application code is available on GitHub[9].

Bibliography

[1] *Link to the wikipedia page on galaxy*, `https://en.wikipedia.org/wiki/Galaxy`, Accessed: 2023-01-10.

[2] *Link to the wikipedia page on stars*, `https://en.wikipedia.org/wiki/Star`, Accessed: 2023-01-10.

[3] *Link to the wikipedia page on quasar*, `https://en.wikipedia.org/wiki/Quasar`, Accessed: 2023-01-10.

[4] *Link to sdss data release 17*, `https://www.sdss4.org/dr17/`, Accessed: 2023-01-10.

[5] *Link to the wikipedia page on sdss*, `https://en.wikipedia.org/wiki/Sloan_Digital_Sky_Survey`, Accessed: 2023-01-10.

[6] *Link to the wikipedia page on redshift*, `https://en.wikipedia.org/wiki/Redshift`, Accessed: 2023-01-10.

[7] *Link to the wikipedia page on right ascension*, `https://en.wikipedia.org/wiki/Right_ascension`, Accessed: 2023-01-10.

[8] *Link to the wikipedia page on declination angle*, `https://en.wikipedia.org/wiki/Declination`, Accessed: 2023-01-10.

[9] *Project link on github*, `https://github.com/Arzazrel/ProjectDM22.git`, Accessed: 2023-01-15.