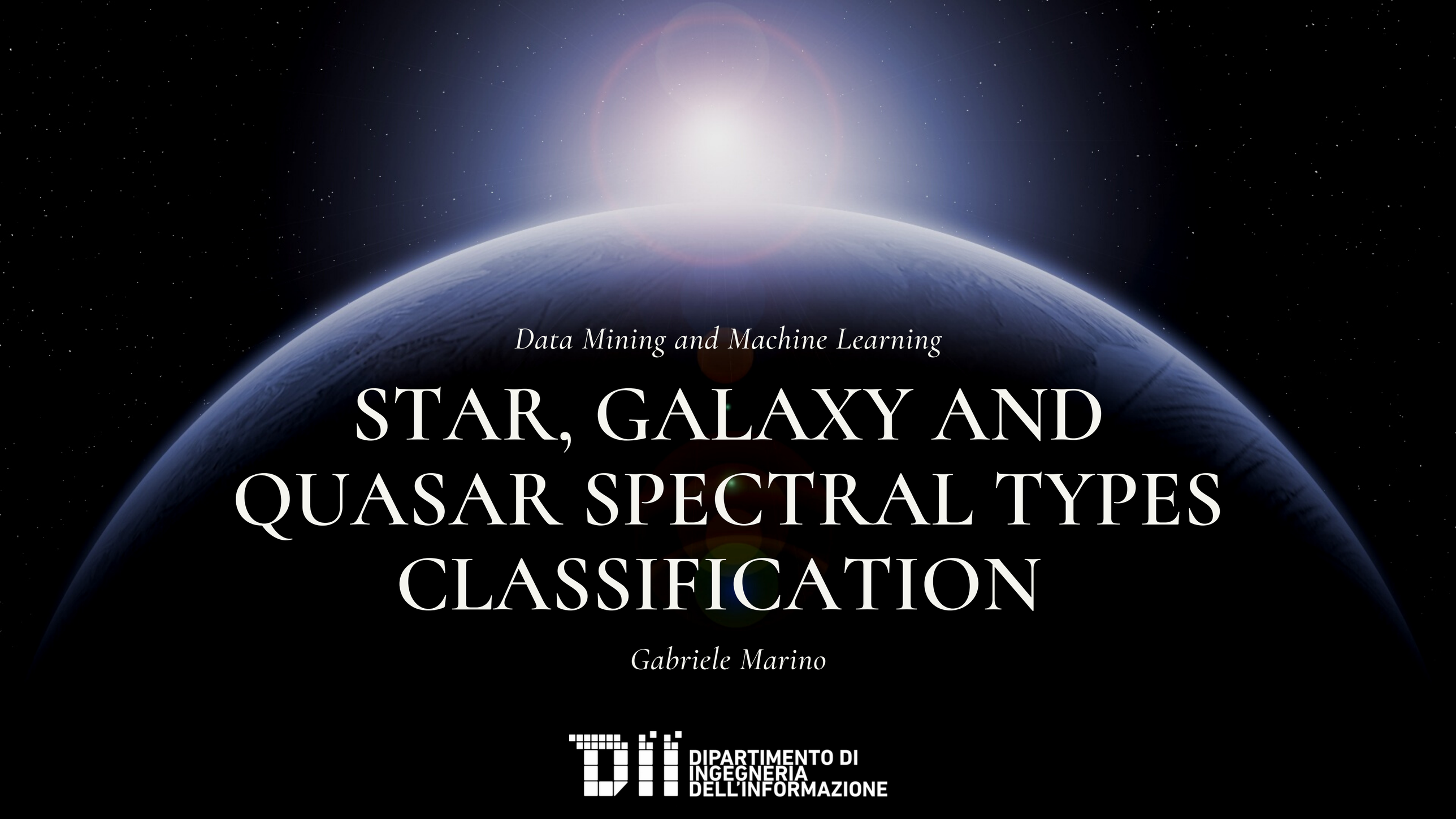


Data Mining and Machine Learning

STAR, GALAXY AND QUASAR SPECTRAL TYPES CLASSIFICATION

Gabriele Marino

DIPARTIMENTO DI INGEGNERIA DELL'INFORMAZIONE

— The Topic

Brief introduction

The classification of galaxies, quasars and stars is one of the most fundamental in astronomy. The first cataloging of stars and their distribution across the sky led to the understanding that they make up our galaxy and, following the distinction that Andromeda was a separate galaxy to our own

Dataset

Context

The data consists of 100.000 observations of space taken by the SDSS (Sloan Digital Sky Survey). Every observation is described by 17 features and 1 class-labeled which identifies each object as a star, galaxy or quasar.

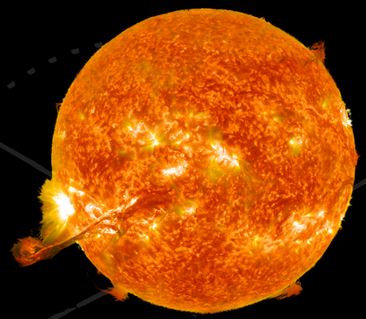
Attributes

- **objid** = Object Identifier, the unique value that identifies the object in the image catalog used by the CAS
- **ra** = Right Ascension angle (at J2000 epoch)
- **dec** = Declination angle (at J2000 epoch)
- **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
- **run** = Run Number used to identify the specific scan
- **rereun** = Rerun Number to specify how the image was processed
- **camcol** = Camera column to identify the scanline within the run
- **field** = Field number to identify each field
- **specobjid** = Unique ID used for optical spectroscopic objects
- **redshift** = redshift value based on the increase in wavelength
- **plate** = plate ID, identifies each plate in SDSS
- **mjd** = Modified Julian Date, used to indicate when a given piece of SDSS data was taken
- **fiberid** = fiber ID that identifies the fiber that pointed the light at the focal plane in each observation

Labeled Class

The labeled class has this kind of distribution among the three values:

Galaxies: 59%



Stars: 22%

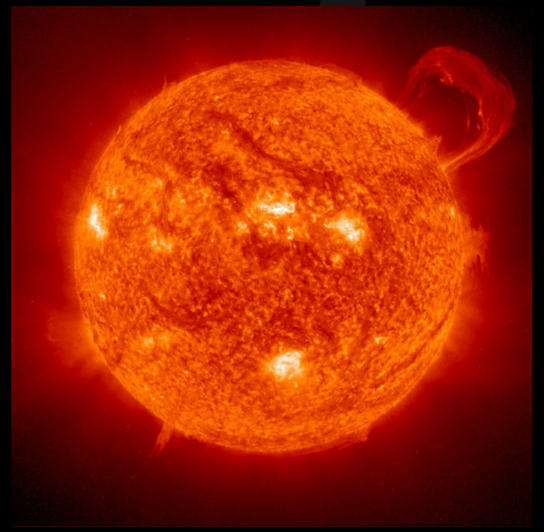
Quasars: 19%

I needed to balance the dataset

WorkFlow before Classification

- Initial Inspection
- Encoding Labeled Class
- Features Selection
- Outlier Detection
- Rebalance

DISCOVER MORE



Initial inspection

01

There is no NaN or NULL value

02

There is no duplicate rows

Encoding Labeled Class

I decided to convert the nominal labeled class in a numerical one, because some method gave error with string.

GALAXY became 0

STAR became 1

QSO became 2

Splitting in Train and Test sets

Before made every kind of transformation on the DataSet I splitted it in **Train** and **Test** sets.
I putted the 30% of the total into the Test Set and the other part in the Training Set.

Features Selection

Not all the features are really useful for this classification.

- For instance the attribute ***rerun*** is constant for all the rows, whereas the attribute ***objid*** varies.
- I used the class ***SelectKBest*** with ***f_regression*** as a score function, and the ranking that I found is the follow:

		Score
1	redshift	28655.95
2	z	9269.76
3	u	6208.25
4	i	3823.40
5	plate	3402.13
6	specobjid	3402.12
7	mjd	3109.64
8	r	1659.22
9	g	305.13

		Score
10	dec	118.45
11	field	100.51
12	fiberid	77.34
13	camcol	14.04
14	objid	0.42
15	run	0.42
16	ra	0.09
17	rerun	0.00

Features Selection

I choose to select the first 9 feature in the ranking

1. redshift
2. z
3. u
4. i
5. plate
6. specobjid
7. mjd
8. r
9. g

Outlier Detection

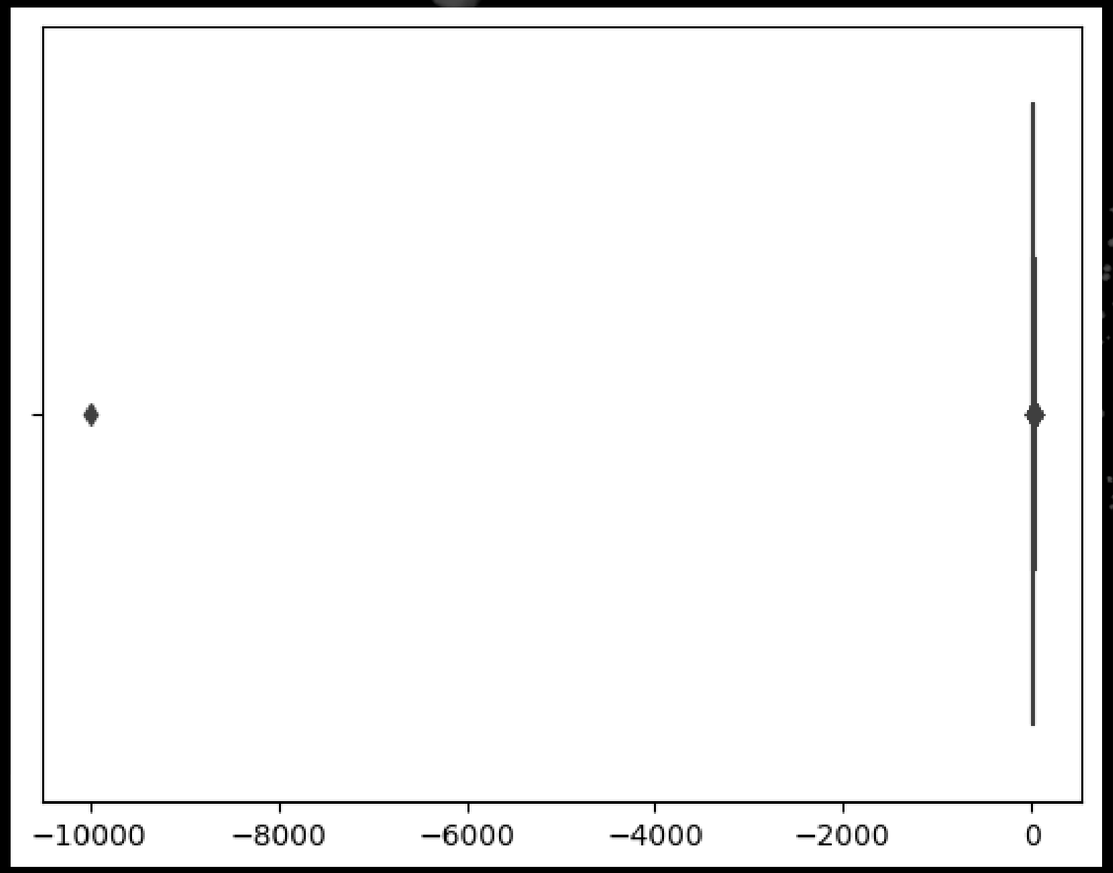
I performed the LOF algorithm for the outlier detection analysis.
And, looking at the BoxPlot, the following characteristics are the ones that have the worst outliers.

For understand the best values of the two parameters of **LOF** I used the class **GridSearchCV**, of **sklearn**.
This class need as a parameters a list of values of **n_neighbors** and **contamination**.
The best value that I found are:

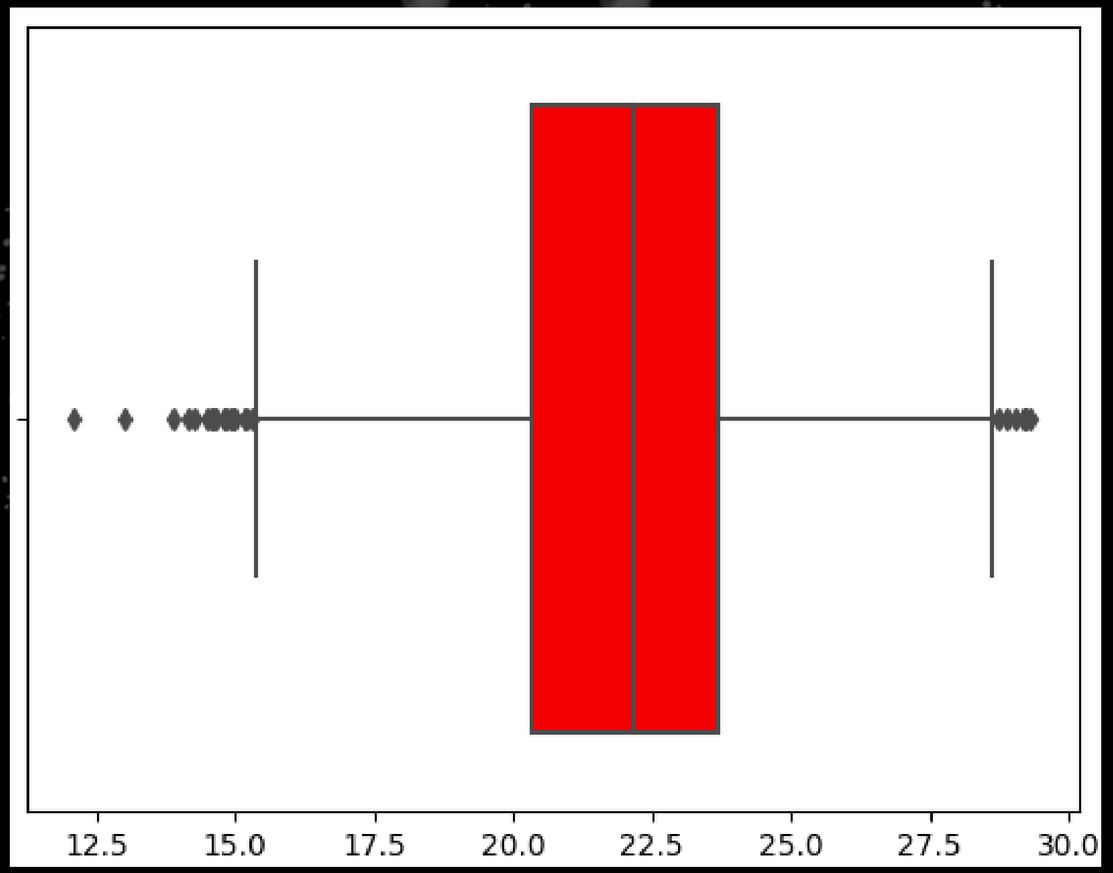
N_NEIGHBORS = 1
CONTAMINATION = 0.1

Outlier Detection

BoxPlot for u



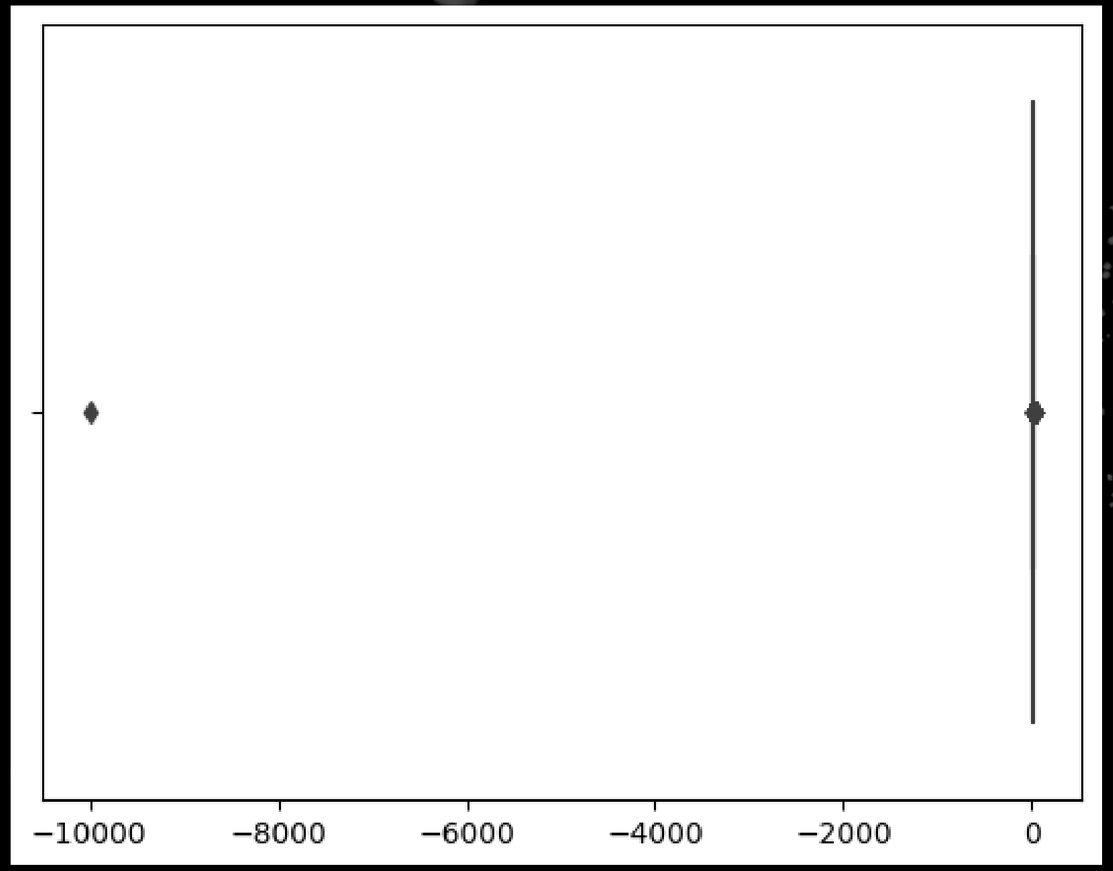
Before Outlier Detection



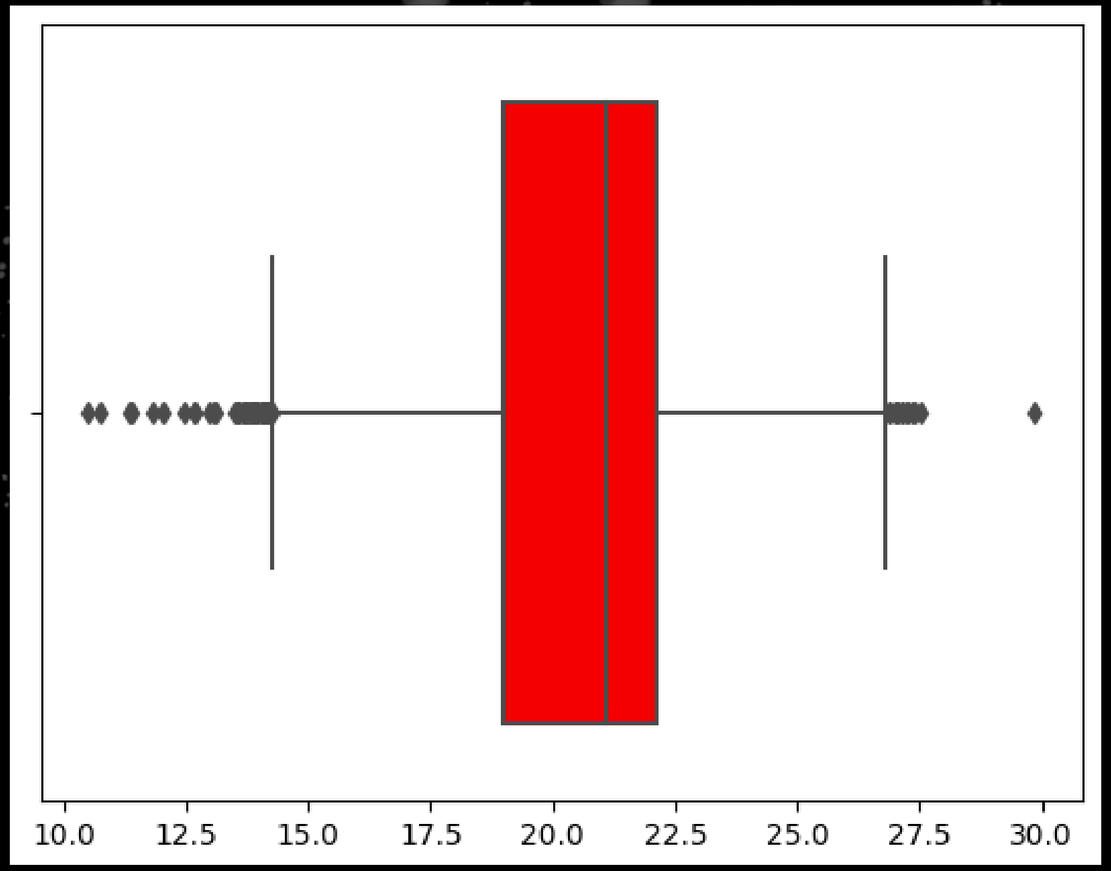
After Outlier Detection

Outlier Detection

BoxPlot for g



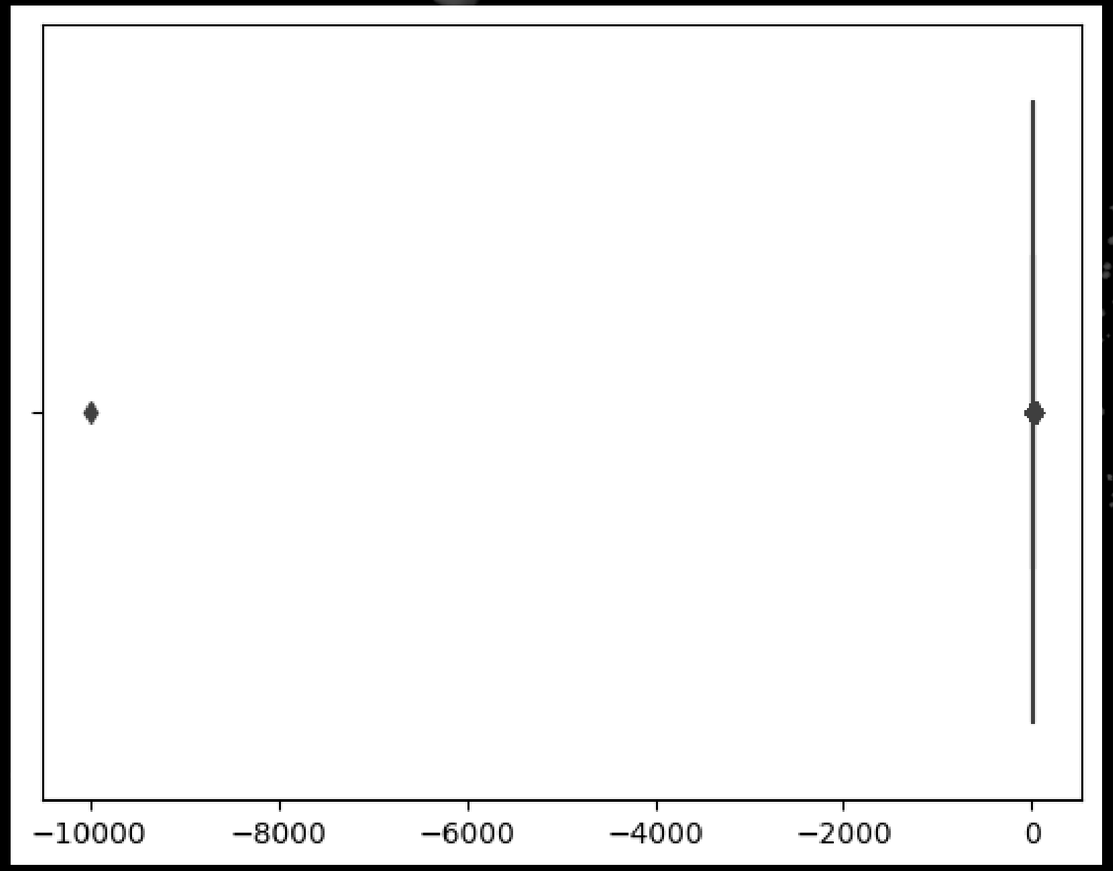
Before Outlier Detection



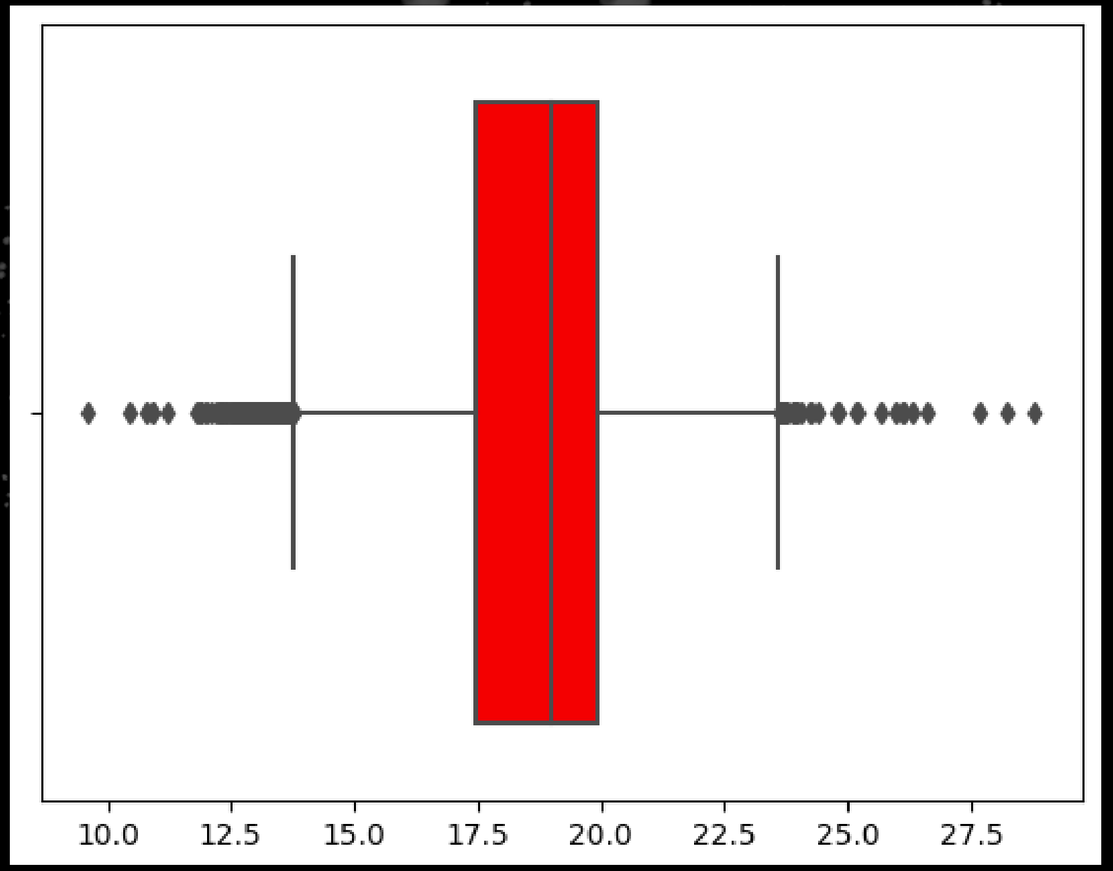
After Outlier Detection

Outlier Detection

BoxPlot for z



Before Outlier Detection



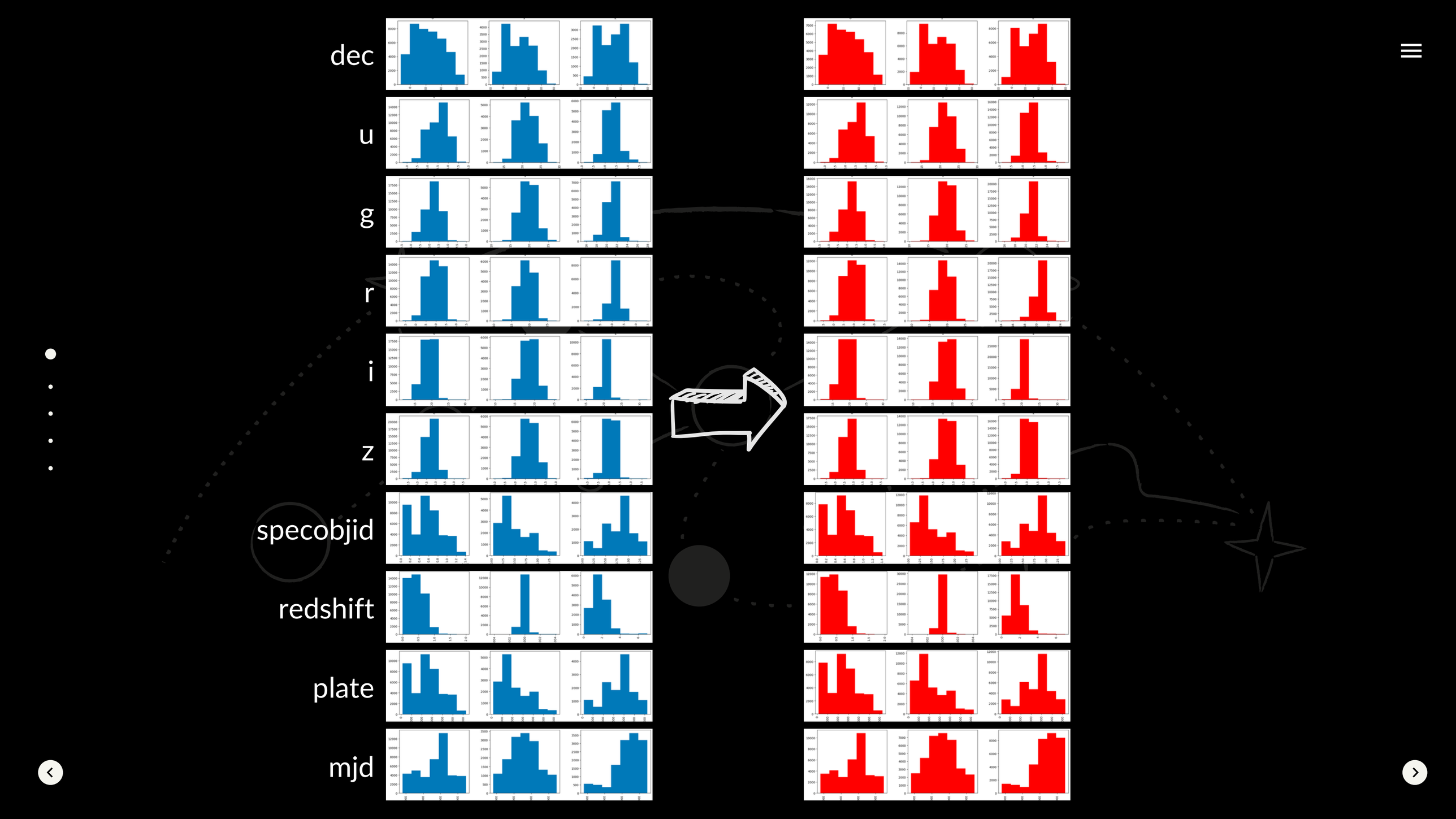
After Outlier Detection

Rebalance

To balance all the labeled classes I used the ***SMOTE*** class from the ***imblearn.over_sampling*** library.
In this way I brought the two minority classes to the same size as the majority class with **41219** examples

```
# using oversampling with SMOTE to deal with imbalanced data
sm = SMOTE(random_state=42)
X_train_after_balancing, y_train_after_balancing = sm.fit_resample(X_train, y_train) # type: ignore
```

All features remained with the same trend the same after balancing

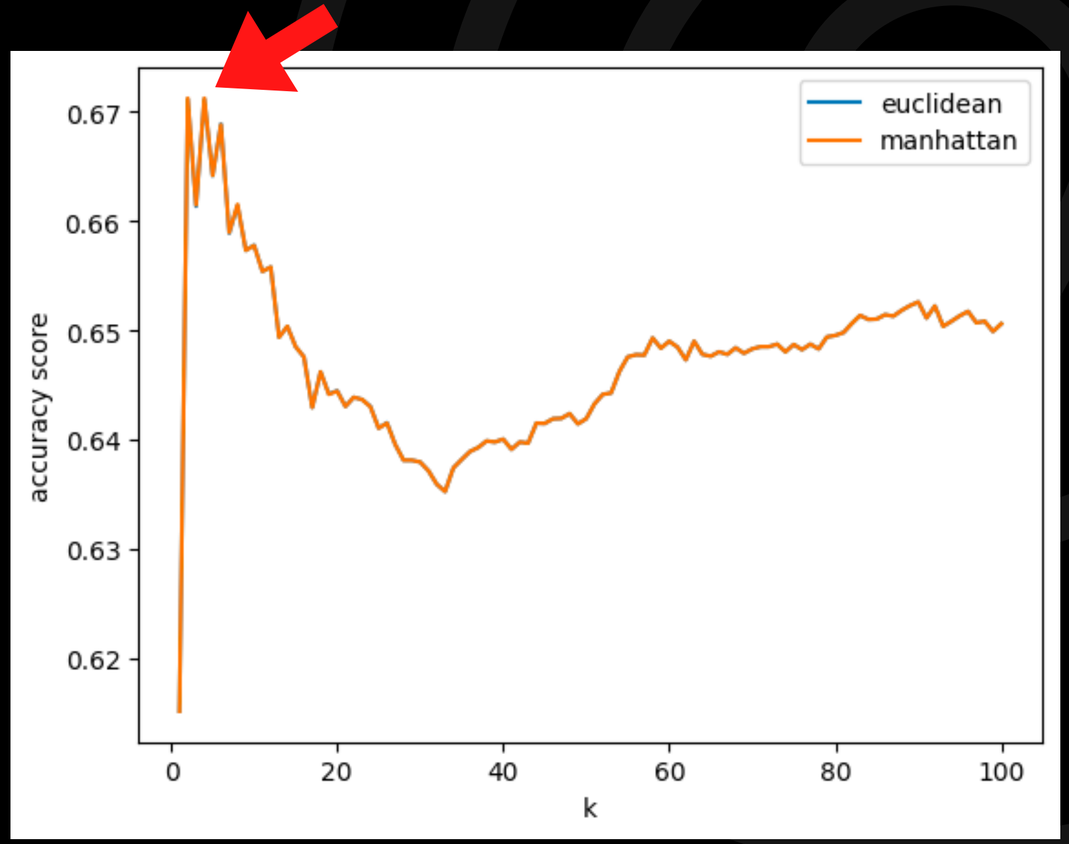

dec
u
g
r
i
z
specobjid
redshift
plate
mjd

Classification

K-Nearest Neighbors, Decision Tree, Random Forest, Bayesian Classifier

K-Nearest Neighbors

I tried to perform KNN with different k, for understand which is the best.
I decided to use the euclidean and the manhattan distance, for make a comparison



Euclidean Distance

- K = 3
- Accuracy = 0.67210

Manhattan Distance

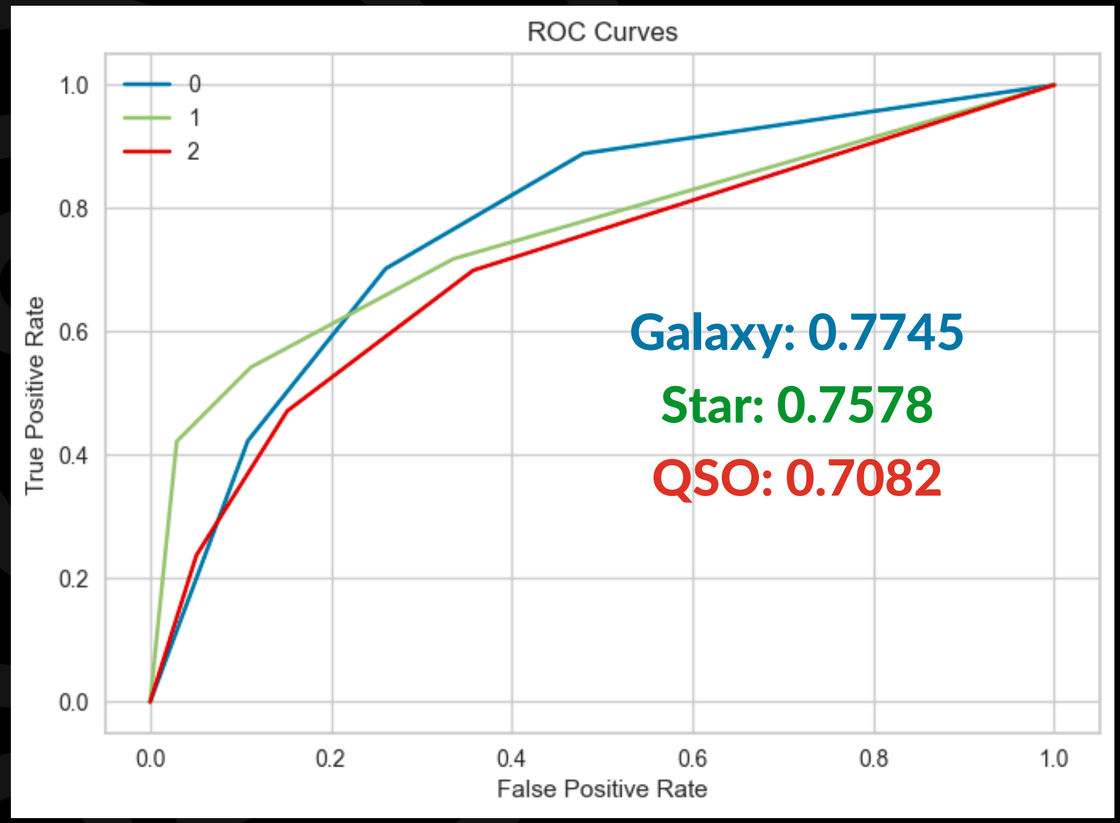
- K = 3
- Accuracy = 0.67213

K-Nearest Neighbors

Cross-validation accuracy scores:

0.7019
0.6940
0.6857
0.6930
0.6958
0.6948
0.6973
0.6922
0.6917
0.6965

Mean score: 0.69429



ROC Curves

K-Nearest Neighbors

Confusion Matrix

	GALAXY	STAR	QSO
GALAXY	13519	1745	2545
STAR	1848	3553	1149
QSO	2113	868	2660

	Precision	Recall	F1-Score
Galaxy	0.77	0.76	0.77
Star	0.58	0.54	0.56
SQO	0.42	0.47	0.44

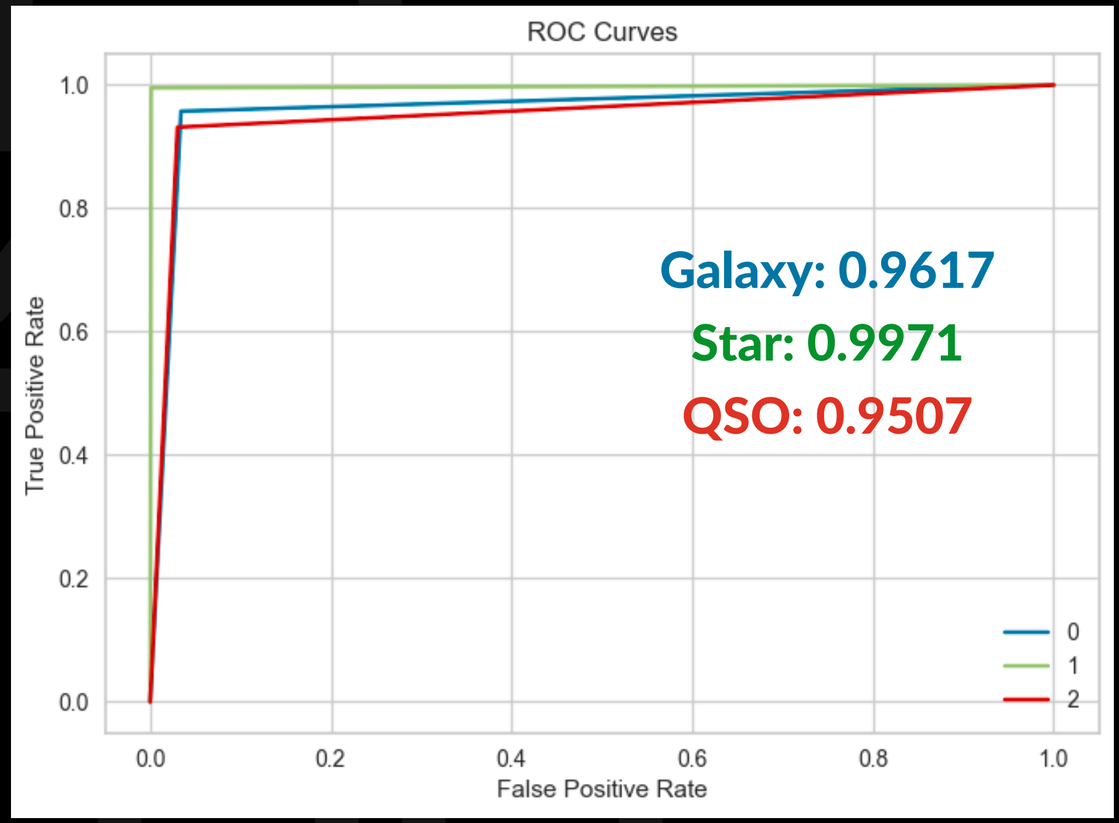
	Accuracy
Galaxy	0.72
Star	0.81
SQO	0.78

Decision Tree

Cross-validation accuracy scores:

0.9672
0.9627
0.9646
0.9680
0.9648
0.9641
0.9626
0.9650
0.9659
0.9630

Mean score: 0.96478



ROC Curves

Decision Tree

Confusion Matrix

	GALAXY	STAR	QSO
GALAXY	17044	24	741
STAR	19	6531	0
QSO	378	0	5263

	Precision	Recall	F1-Score
Galaxy	0.98	0.96	0.97
Star	1.00	1.00	1.00
SQO	0.88	0.93	0.90

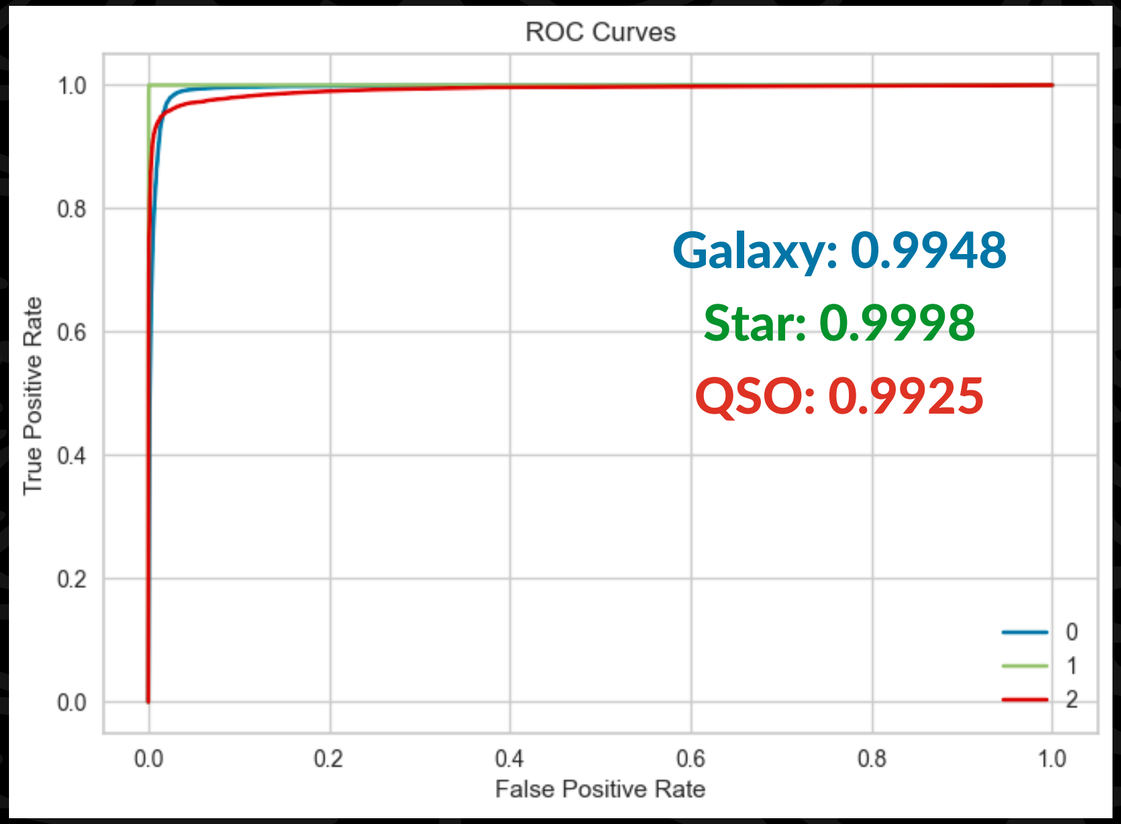
	Accuracy
Galaxy	0.96
Star	1.00
SQO	0.96

Random Forest

Cross-validation accuracy scores:

0.9803
0.9777
0.9772
0.9797
0.9777
0.9787
0.9776
0.9813
0.9778
0.9790

Mean score: 0.9787



ROC Curves

Random Forest

Confusion Matrix

	GALAXY	STAR	QSO
GALAXY	17450	27	332
STAR	4	6546	0
QSO	300	1	5340

	Precision	Recall	F1-Score
Galaxy	0.98	0.98	0.98
Star	1.00	1.00	1.00
SQO	0.94	0.95	0.94

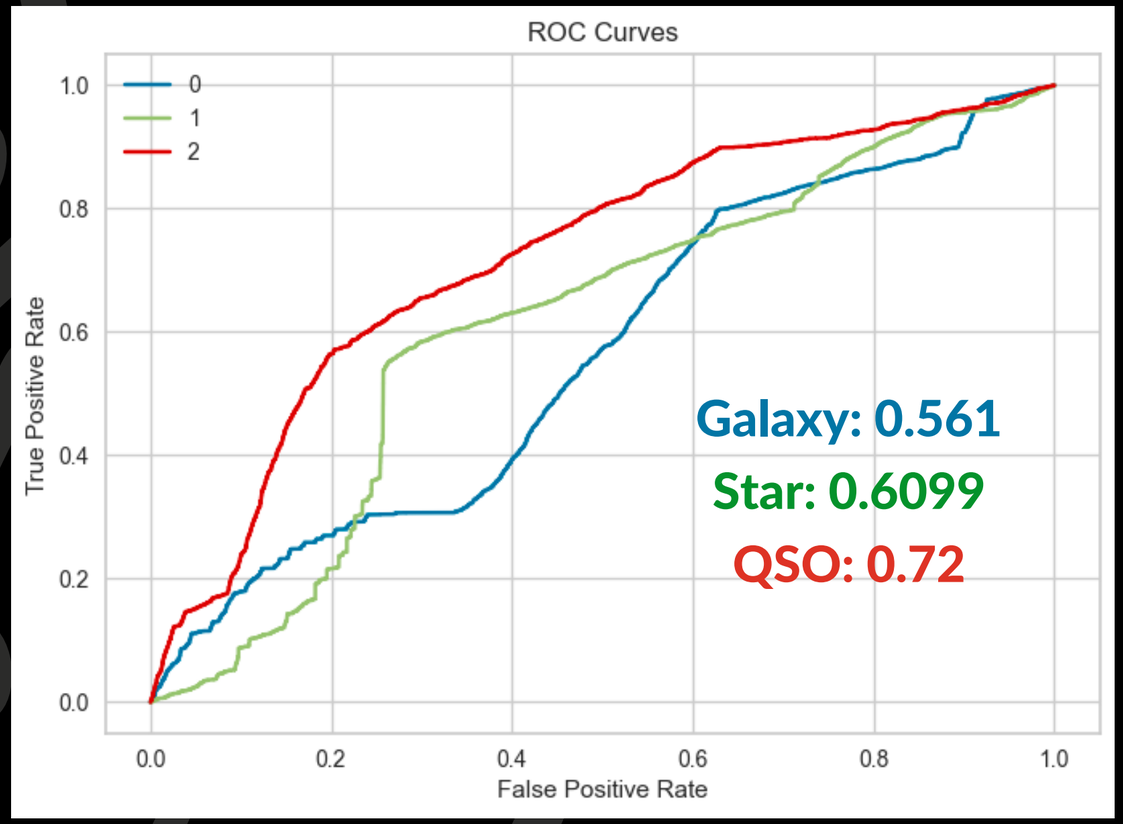
	Accuracy
Galaxy	0.98
Star	1.00
SQO	0.98

Bayesian Classifier

Cross-validation accuracy scores:

0.6038
0.6052
0.6014
0.6006
0.5990
0.6008
0.6053
0.6004
0.5985
0.6024

Mean score: 0.60174



ROC Curves

Bayesian Classifier

Confusion Matrix

	GALAXY	STAR	QSO
GALAXY	851	10043	6915
STAR	191	4498	1861
QSO	211	1533	3897

	Precision	Recall	F1-Score
Galaxy	0.68	0.05	0.09
Star	0.28	0.69	0.40
SQO	0.31	0.69	0.43

	Accuracy
Galaxy	0.42
Star	0.55
SQO	0.65

Conclusion

The conclusion is that the best classifier for this problem is Random Forest, because they has the best metrics in terms of Precision, Recall, Accuracy and Cross-Validation scores. And if we looking at the ROC curve, Random Forest have the greatest value.

User Application

I made a simple user application. I exported the model via the **Pickle** library, which allows you to save the model in a file