Stellar Classification: Explore Features Engineering

Sara Cammarota, Gianmarco Fiorenza, Mattia Girolami, Luca Mazzucco, Ilaria Petrucci

December 27, 2022

1 Introduction

The aim of our work was to leverage data from Sloan Digital Sky Survey (SDSS) release 17, downloaded from Kaggle, to explore feature engineering while performing classification of quasars, stars and galaxies. We choose as classification models the ones explored in class that were best suited to this purpose, namely Multinomial Logistic Regression (MLR) and K-nearest Neighbours classifier (KNN).

2 Exploratory Data Analysis

2.1 Data Composition

The chosen dataset is composed by 100.000 observations of stellar objects, 17 features and the class which identifies each object to be either a star, galaxy or quasar. Many of the available features, however, were not directly related to the celestial objects characteristics but were inherent to the data collection procedure and we therefore chose to remove them and consider only the spectral features, which were more appropriate for classification purposes, taking into account in the end six features (emission in the ultraviolet, green, red, near infrared and infared filters and redshift). The dataset was split into three sets: training (70%) to train the model, test (15%), which we maintained untouched and validation (15%), used to choose the best model parameters.

2.2 Outlier Detection and Class Imbalance

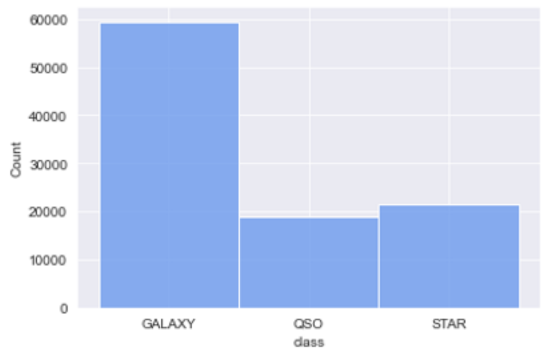


Figure 1: Barplot showing the class imbalance of the dataset. The number of galaxies is three times higher than that of the other two classes.

Exploring the dataset we discovered the presence of numerous outliers for each spectral feature and at the beginning we decided to maintain them since we supposed they explained the natural variability of data. As shown by Figure 2 they constitute around 10% of the data and for this reason it's worth to assess performance of models with or without them.

Then we focused on another arising issue: class imbalance. Indeed, as it possible to see from the barplot in Figure 1 it's evident that the dataset is skewed towards galaxies, which turn out to be present almost twice as much as quasars and stars.

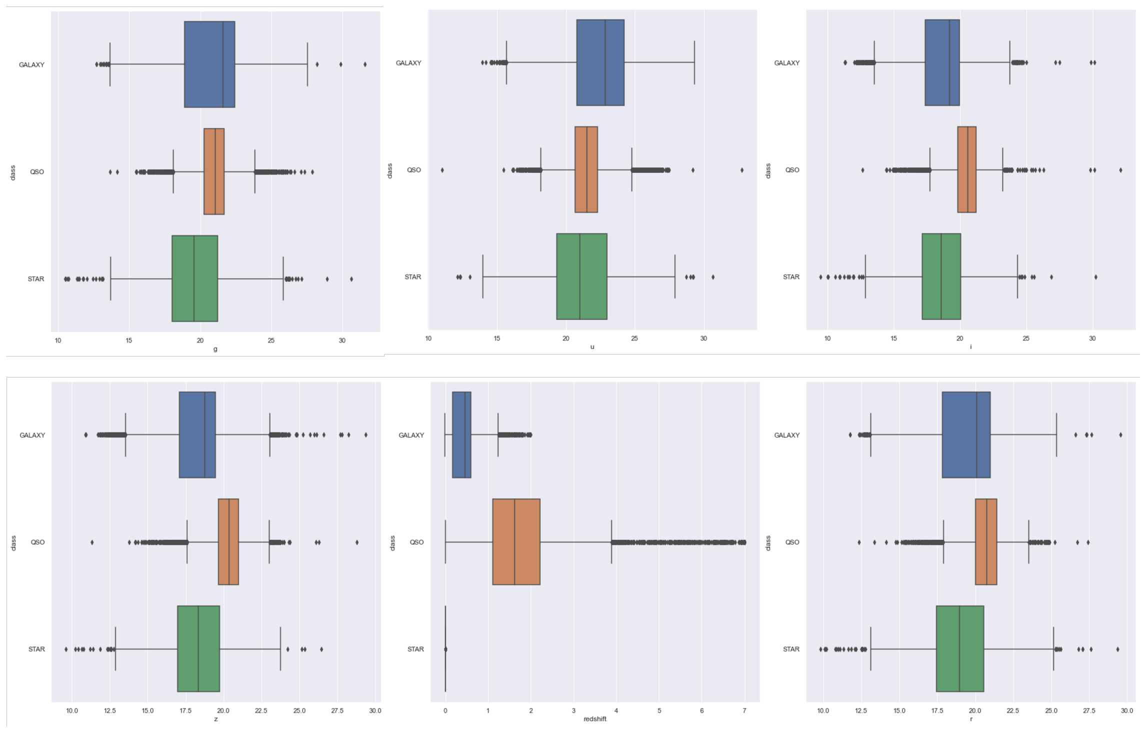


Figure 2: Box plots for all the spectral features of the dataset, divided by class.

We focused on class imbalance of the training set since it could affect machine learning models. Indeed what could happen with an imbalanced dataset is that the model could overclassify the most frequent class (in our case galaxies), leading to a general overfitting. Moreover in this situation the accuracy will not be a proper metric to evaluate the performance of the model because it will not be uniform over the classes, therefore other metrics should be employed. To overcome limits, we decided to use Area under Receiver Operating Characteristic (AUROC) and balanced accuracy from Scikit-learn library, that is an average of the recalls (the ratio between true positive and the sum of true positives and false negatives) computed for each class:

$$Recall = \frac{TP}{TP + FN} \tag{1}$$

In order to minimize the effects of unbalancing, oversampling was chosen. Oversampling involves randomly selecting examples from the minority classes (in our case stars and quasars), with replacement, and adding them to the training set; this obviously leads to an increase in the size of the training set, while leaving validation and testing unaffected.

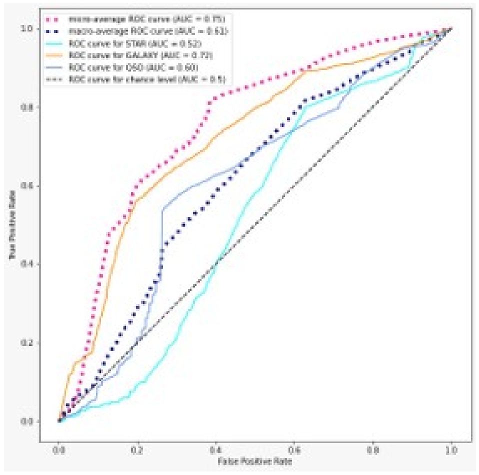
2.3 Multinomial Logistic Regression

At this point we moved on to the development of the first chosen model, namely Multinomial Logistic Regression (MLR) and the evaluation of its performance, in the presence and absence of oversampling. As mentioned, we relied on AUROC. In particular, we calculated through the scikit-learn library various ROC types. We computed One-vs-the-Rest curves which consists in computing a ROC curve per each single class considering the other two as the negative class. Then we moved to average ROC curves, choosing Micro-average ROC which is particularly suitable in case of skewed datasets, since it aggregates the contributions from all the classes to obtain the average metrics as follows:

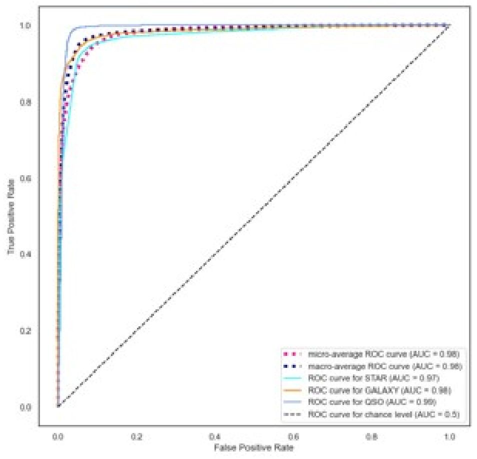
$$TPR = \frac{\sum_c TP_c}{\sum_c (TP_c + FN_c)}$$

$$FPR = \frac{\sum_c FP_c}{\sum_c (FP_c + TN_c)}$$

Moreover we computed the Macro-average ROC that requires computing the metric independently for each class and then taking the average over them, hence treating all classes equally a priori. We have to take into account that Micro-averaged ROC is dominated by the more frequent class (galaxies) while macro-averaged alternative better reflects the statistics of the less frequent classes (stars and quasars), and then is more appropriate when performance on all the classes is needed as in our case. With these more reliable metrics we were able to conclude that a clear increase in performance was observed as a result of oversampling, as evident from all types of curve.



(a) AUROC of MLR before oversampling.



(b) AUROC of MLR after oversampling.

Figure 3: AUROC for MLR performance.

We then focused on the previously detected outliers and decided to test the performance of MLR by removing them from the training set. We observed a generic maintenance in the performance of Multinomial Logistic Regression in both AUROC and balanced accuracy for test and validation set. This result seems unusual since the presence of outliers tends to invalidate the model and therefore their removal should lead to an increase in performance. However, in this case the outliers do not result from errors in the data collection but are part of the natural variability of the data. This means that the measurements themselves have a huge range of values that they can assume for different celestial objects and this could explain the obtained result.

3 Features Engineering

At this point, we wanted to investigate the contribution of the features to the performance of the chosen models.

3.1 Correlation Analysis

First, we analysed the correlation between features by creating a correlation matrix showing the most significantly correlated features with a heat map (Figure 4). We noticed that there was a very high correlation especially between the emission features. To avoid multi-collinearity we did not opt for features removal but instead focused on dimensionality reduction.

3.2 Principal Component Analysis

As a method of dimensionality reduction, we chose principal component analysis, carried out after an appropriate normalisation of all features (StandardScaler from scikit-learn library). From the PCA, we noted the presence of a first component that alone explained 70% of the variance, and we assumed that this component tended to summarise the contribution of redshift, which alone (removing all other features) resulted in a rather higher accuracy than we expected for the MLR model. However, we chose to include more components, including the second that explained 15.6% of the variance and the third (7.3%). Figure 5 shows the datapoints plotted with respect to the first three principal components.

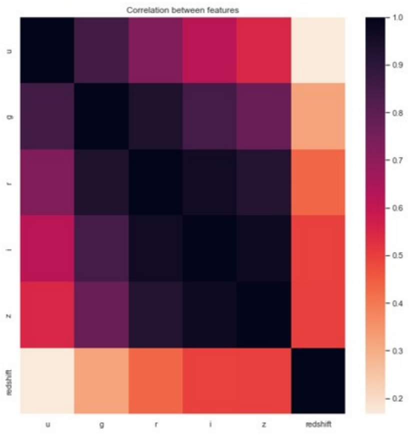


Figure 4: Correlation matrix of the spectral features.

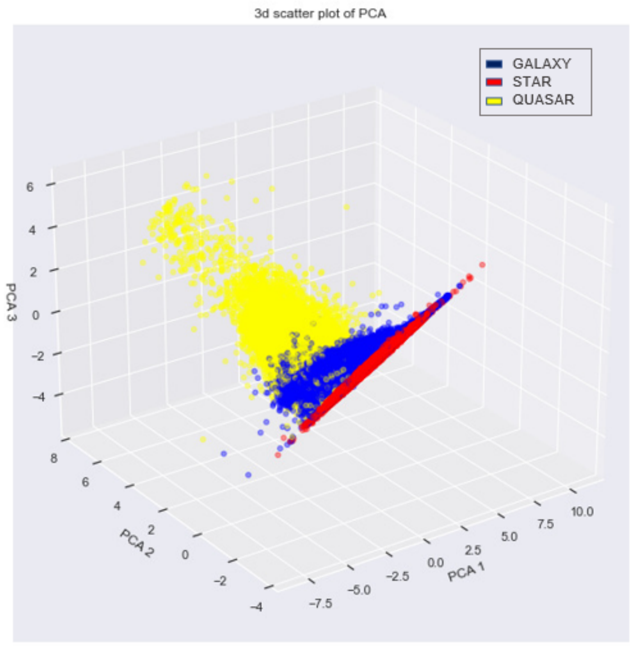


Figure 5: The three classes plotted with respect to the first three principal components. The classes are almost completely separated.

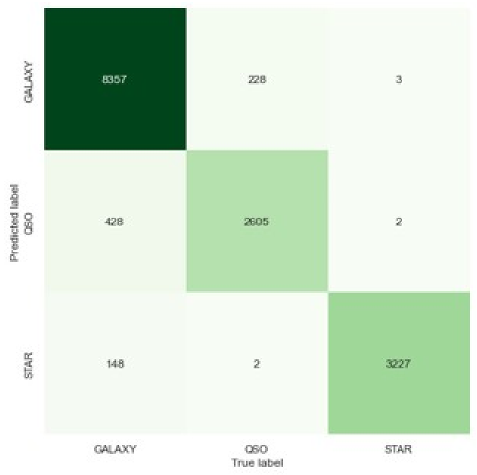
3.3 Multinomial Logistic Regression model

In order to obtain a balanced procedure, we oversampled the dataset with the three main components as well, and trained and tested the previously used Multinomial Logistic Regression model on both the original dataset and the latter. At this point, we computed the confusion matrix for both cases (Figure 6a and Figure 6b) and calculated the balanced accuracy.

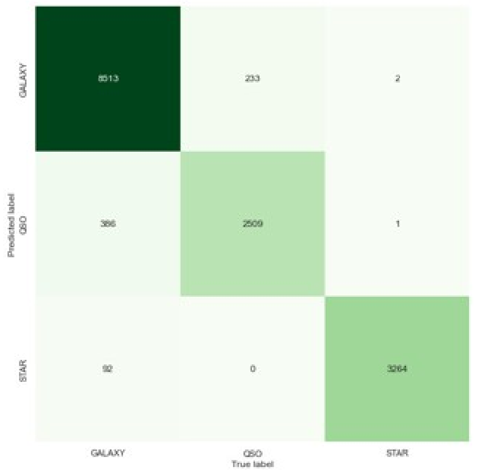
In this way we were able to observe a substantial maintenance of the model's performance with the use of the principal components, a procedure that is very convenient since by making a dimensional reduction it allows us to considerably reduce the computational training time.

3.4 K-Nearest Neighbours Classifier model

We then moved to the KNN to test its performance and observe if there were any changes when using the original features and principal components. We used the validation set to assess which was the best value of neighbours (k), i.e. the one with the highest accuracy. We then calculated accuracy and



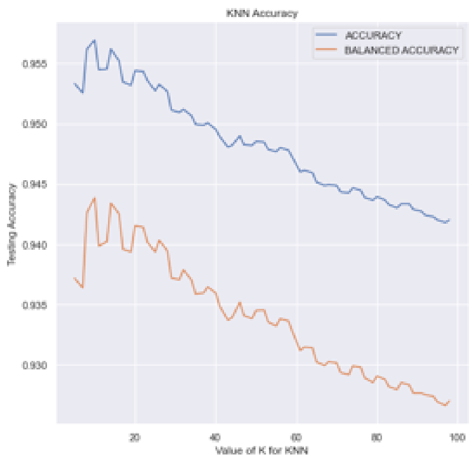
(a) Confusion matrix after MLR on whole dataset.



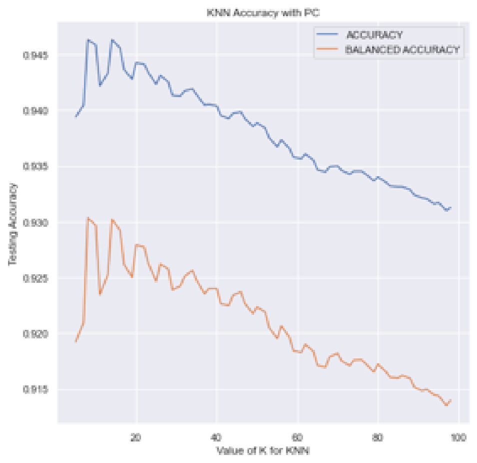
(b) Confusion matrix after MLR on the first three principal components.

Figure 6: Confusion matrices for MLR, before and after PCA.

the balanced accuracy mentioned earlier, as the value of k varied from 1 to 100. We saw that both metrics were particularly low in the case of value of k multiples of three (multiples of the number of classes) and therefore excluded them from the range under consideration.



(a) Accuracy and Balanced Accuracy for KNN on the whole dataset



(b) Accuracy and Balanced Accuracy for KNN on the first three principal components

Figure 7: Accuracy and Balanced Accuracy metrics for KNN

We choose the best k as 10 in the case of the original dataset (Figure 7a) and $k = 8$ for the three principal components (Figure 7b) and in conclusion we observed that the performance in the original dataset was, as expected, slightly higher than that calculated with the 3 main components. Overall, considering the reduction in computational time for both the choice of the best $k - value$ and the training phase, the performance on the reduced dataset is more than satisfactory. Here again, we tested the performance of the model following the removal of outliers in the training set and in an oversampling condition and observed similar, but more pronounced behaviour than in the case of Multinomial Logistic Regression, meaning that there was a slight performance reduction. We therefore also came to the same conclusions here, assuming that the removal of outliers is not appropriate in this case.

4 Conclusions

In general, we can draw several conclusions from our analysis. First, it emerges that it is critical to deal with any class imbalances as they can result in a sharp drop in model performance, and that in our case oversampling was extremely helpful in improving it.

When investigating the outliers, we discovered that we must proceed cautiously in removing them as it is necessary to understand the cause of their presence in order to decide how to deal with them. As observed in this case, we were in the rarer situation where outliers are part of the inherent variability of the data and therefore their removal does not cause an increase in performance as expected.

We then noted the importance of using appropriate metrics in the presence of skewed datasets since we can be misled by very high values that indicate high model performance but actually hide a situation of overfitting, which should be avoided.

We then looked at dimensionality reduction and found that the performance levels of the KNN and Multinomial Logistic Regression are essentially unaffected when using Principal Component Analysis and taking into account a sufficient number of components that explain most of the variance. This can be very useful since, as mentioned, working on a reduced dataset brings numerous benefits in terms of computational time, leading to a net reduction of it.

Finally we observed that in the case of this dataset, with this type of features, the best performing algorithm was the Multinomial Logistic Regression, which overall had higher metrics and thus higher classification ability. Nevertheless, K-Nearest Neighbours also had a high performance being adequate for this type of classification analysis with three classes.

5 Group Organization

Basically, for this project we all worked together on the planning and execution phase. In particular, we took together the decisions on features engineering and then divided up the models to be tested (Gianmarco Fiorenza, Mattia Girolami and Luca Mazzucco dealt with most part of Multinomial Logistic Regression, while Sara Cammarota and Ilaria Petrucci dealt with most part of KNN). Ilaria Petrucci and Sara Cammarota took care of the final report.

6 References

The SDSS data we used is available here, while a benchmark notebook that we leveraged for the features engineering part is available here.