# Classifying astronomical sources with machine learning

Author: Jordi Sabatés de la Huerta.
*Facultat de Física, Universitat de Barcelona, Diagonal 645, 08028 Barcelona, Spain.*

Advisor: Jose Maria Solanes Majua & Maria Salamó Llorente

(Dated: January 19, 2022)

**Abstract:** More than 4 million astronomical sources extracted from the Sloan Digital Sky Survey catalog have been used to train a set of machine learning models, selected with a benchmarking program, in order to identify the best basic classifier of astronomical sources for future observations. We have also applied different filters to our dataset that modify its selection function, measuring the accuracy of the selected model to evaluate under which observational constraints this model performs better.

## I. INTRODUCTION

The universe is composed of various objects of different shape, size and color. In order to understand the universe we first need to classify the objects that make it up.

For centuries [1], our ancestors have been looking at the sky to understand what kinds of objects were in the universe. From the Egyptians to the present day, humanity has created thousands of different astronomical catalogs [2].

The goal of all of them is to collect observations (measurements) of astronomical objects made with one or more instruments and to combine them into a unique homogeneous description. This enables anybody interested in the study of a given class of sources to compare their properties on an equal basis. Now, with more extensive and higher quality catalogs, we can perform this study in a better way. This work focuses on the classification of astronomical light sources into the three fundamental classes of stars, galaxies and quasars.Specifically, we do this using Machine Learning (ML) techniques, a type of artificial intelligence that is very useful in cases like this where there are zillions of objects to classify.

The catalog we will use is the Sloan Digital Sky Survey (SDSS), a blind imaging and spectroscopic astronomical survey gathering data at optical wavelengths from a dedicated wide-angle telescope of 2.5 meters located at Apache Point Observatory in New Mexico [5]. This database, specifically its Data Release 16 (DR16) [4], contains a long list of measurements for hundreds of millions of astronomical objects that will be explained later in more detail. We will use the fact that astronomical sources of a given type or class share a set of common characteristics to train a machine learning model capable of making an automated classification of future survey targets.

## II. DATASET

The SDSS-DR16 dataset includes diferent types of information: imaging, photometric and spectroscopic, as well as cross-matches with observations from other large-scale surveys. We have focused this study on primary and secondary SDSS objects with reliable photometry and listed also in mid-IR AllWISE catalog.

The photometric data from the SDSS come in five optical broad-band filters: u($\lambda = 0.355\mu m$) and g($\lambda = 0.469\mu m$) and r($\lambda = 0.617\mu m$) and i($\lambda = 0.748\mu m$) and z($\lambda = 0.893\mu m$) [3] that are used to calculate up to 5 different measures of the total flux each object in our dataset: **psfMag**, measures the total flux determined by fitting a PSF point spread function model to the object; **modelMag**, calculates the magnitude from the best fit using a de Vaucouleurs function or an exponential one; **cModelMag**, composite magnitudes, calculates the total magnitude by fitting a linear combination of de Vaucouleurs + exponential profiles; **petroMag**, the Petrosian magnitude, a way of calculating the total magnitude by measuring a constant fraction of the total light, regardless of the distance to the object and that is unaffected by extinction; **modCorrected**, extinction-corrected modelMag.

For its part, the WISE dataset provides magnitudes in 4 mid-infrared bands: w1($\lambda = 3.4\mu m$), w2($\lambda = 4.6\mu m$), w3($\lambda = 12\mu m$) and w4($\lambda = 22\mu m$) [4].

The data have been extracted using SQL-queries and the CasJobs program of the SkyServer webpage of the SDSS. Our DR16 dataset includes 3648512 unique sources, for which we have spectroscopic, photometric and infrared information. These sources are classified by SDSS according to whether they are stars ("STAR"), galaxies ("GALAXY") or quasars ("QSO"). This classification was done by first observing whether the source was extended or not, classifying it as a galaxy or a star respectively [5]. Then, together with the redshift estimates of the sources, it was possible to differentiate between stars and quasars. Currently, for spectroscopic targets, classification is made by comparing different predefined models of spectra with their observed spectrum. [6].

To train a ML model we need first to clean the dataset, removing invalid and doubtful data. This has been done by performing the following operations: (1) we have ensured that there are no duplicate entries, or objects with infinite magnitudes (NaN); (2) we have also eliminated sources with invalid apparent magnitudes (those with negative or deliberately high values, like 9999); (3) we have kept only objects with a value of zWarning, a parameter provided by the dataset, equal to 0 (indicates no error) and 16 (indicates a source with noise but does not necessarily mean an error); (4) finally, we have removed data with a PSF $r$-band magnitude greater than 24 to avoid what appears to be a failure in the data processing (see Fig. 1). In total we have eliminated 209,888 sources with errors. With all this we have a clean dataset of 3,438,624 celestial objects of which 2,374,624 are galaxies, 517,101 quasars and 546,899 stars (see Fig. 1), the abundance ratios being [4.59, 1.0, 1.05], respectively.

From this cleaned dataset, 80% will be used for training and validating the model, while the remaining 20% will be used later to test the accuracy of our ML methods (Sec. V).

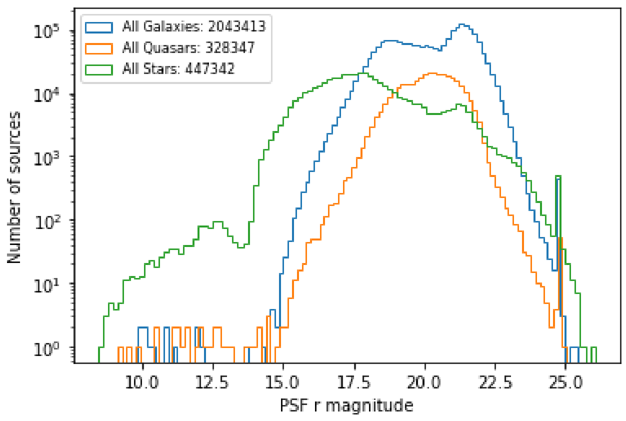

**FIG. 1:** Distribution of the different classes of sources according to the PSF magnitude in the $r$ band. Note the anomalous peak near 25 mag that may result from a glich in the photometric pipeline and that has led us to eliminate from our sample objets with psfMag > 24.

## III. MODELLING

ML methods use a training dataset to determine which functions are best suited to obtain the expected classification results, both in terms of accuracy and process execution time. In this work, we will compare a large set of different classifiers after doing some previous preprocessing of the clean data.

### A. Types of Algorithms

Most of the classifiers that we are going to compare are based on the following commonly used supervised classification methods:

- **Decision trees**: Based on a tree structure. It uses a set of "if-then" style rules that are mutually exclusive and exhaustive for classification. It is a top-down algorithm. The attributes at the top of the tree have the greatest impact on the ranking. The number of branches we choose will depend on the number of objects in the training set.
- **Naive Bayes**: Based on Bayes theorem, assuming that the features are conditionally independent. That is, each of the features of an object will influence the final classification of that object.

$$P(c|X) = P(x_1|c) \times P(x_2|c) \times ... \times P(x_n|c) \times P(c) \quad (1)$$

  - $P(c|x)$ is the posterior probability of *class* (target) given *predictor (attribute)*
  - $P(c)$ is the previous *class* probability
  - $P(x|c)$ is the given *class* probability of the *predictor*
  - $P(x)$ is the past *predictor* probability

  It is an algorithm to be taken into account for very large datasets, like ours, since the training time is linear, it does not scale with the size of the dataset.
- **K-nearest Neighbor (KNN)**: Based on classifying the data by looking for the "most similar" points by proximity learned in the training stage. With this algorithm we estimate the density function $(F(x/Yj))$ of the variables x for each class Yj. The algorithm stores the characteristic vectors and class labels of the training examples. To make the predictions, the vectors are represented in the characteristic space for the elements we want to predict, and the distance between stored vectors and the new vector is calculated. The k closest examples are selected. The new vector (belonging to the object to predict) is classified with the class that is most repeated in the selected vectors.

### B. Pre-Processing

Before one can benchmark the classification algorithms, it is necessary to prepare the clean dataset subjecting it to a preprocessing that consists on the following three steps:

1. **Sampling**: There is the option to choose a subset of data from our dataset to train our model. This must be done very carefully so as not to hide information from the algorithm.

2. **Encoding**: Classification algorithms only work with numerical data (some even only with integers!). In order to pass our variables to the model we will use the **labelEncoder**. This encoder assigns a number to each distinct category. We have only one categorical variable, our ”label”. The astronomical source classification. So the function of this encoder will be as follows:

$$[GALAXY, STAR, QSO] \rightarrow [0, 1, 2]$$

3. **Scaling**: In this process all the variables are transformed within the same range. That is to say, all the magnitudes have the same measurement scale. We will use the so-called **MinMaxScaler**, which transforms all quantities between 0 and 1.

## IV. BENCHMARKING

There are different scores that can be used to compare classifiers. Here, we will deal with two of the most relevant, namely, accuracy and training time.

The TABLE I lists the classifiers benchmarked with the values of accuracy ordered from highest to lowest.

The most important factor in choosing a classifier is by far the accuracy. However, we can see that some classifiers have identical accuracy values. In this case, one has to pay attention to the training time. Note that the classifier that takes the longest time is **kNN brute**, which uses a brute force algorithm. In comparison, the **kNN Tree** which uses a decision tree algorithm, gives the same accuracy but is about 70 times faster!

Among the top five algoritms the ”HistGradBoost” classifier offers the best training time. However, given that the relatively modest size of our dataset, we have given full preference to the accuracy, so our preferred choice is the Random Forest (RF) Entropy, with an accuracy of 0.982 and a training time of 584 seconds.

The RF estimator use bagging (picking a sample of n < N observations rather than all of them) and random subspace method (picking a sample of k ≤ K features rather than all of them, called attribute bagging) to grow a tree. They work very well with large datasets like ours.

But before a RF classifier can be applied to a dataset one must determine what the optimal number of estimators (decision trees) given the the number of observations available from the dataset. To find out which is the best value for us, we have created a program that measures the accuracy of the RF classifier for three different ways of choosing the subset of features:

- *sqrt* uses a random subset considering $\sqrt{nfeatures}$ where nfeatures is the number of features in our dataset.
- *log2* uses $log_2(nfeatures)$ features for each split.
- *none* algorithm uses all features instead of choosing a random subset.

| Classifier | Accuracy | Training-Time (s) |
|---|---|---|
| RF entropy | 0.982 | 584.009 |
| RF gini | 0.982 | 948.446 |
| ExtraTrees | 0.981 | 89.944 |
| Bagging | 0.980 | 115.177 |
| HistGradBoost | 0.980 | 37.903 |
| kNN ball tree | 0.979 | 5933.081 |
| kNN kd tree | 0.979 | 510.750 |
| kNN brute | 0.979 | 36311.472 |
| GradientBoosting | 0.972 | 2443.474 |
| SGD L1 | 0.971 | 3.981 |
| Calibrated-CV | 0.970 | 167.692 |
| LibLinear SVC L2 | 0.969 | 16.546 |
| DecisionTree | 0.968 | 70.771 |
| LibLinear SVC L1 | 0.966 | 403.401 |
| Ridge CV | 0.965 | 2.961 |
| Ridge Auto | 0.965 | 1.309 |
| Passive-Aggressive | 0.964 | 2.462 |
| SGD Elastic-Net | 0.964 | 3.664 |
| SGD L2 | 0.963 | 2.393 |
| Perceptron | 0.957 | 2.217 |
| AdaBoost | 0.930 | 203.204 |
| GaussianNB | 0.746 | 1.957 |
| MultinomialNB | 0.691 | 1.004 |
| BernoulliNB | 0.691 | 1.007 |
| NearestCentroid | 0.659 | 0.887 |

**TABLE I:** Accuracy and training time values for each classifier tested. All of them have been extracted from the Scikit-Learn python package. [7]

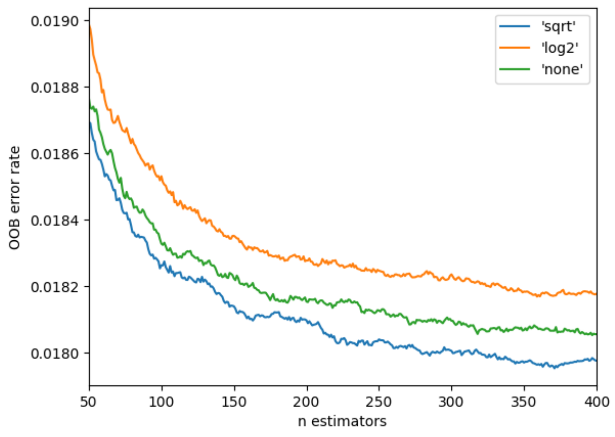

**FIG. 2:** OOB error rate vs. the number of estimators in our random forest classifier.

Fig. 2 shows on the Y axis the out-of-bag (OOB) error, giving the average of the error for each of the observations calculated from the predictions using a different number of trees (n estimators). The RFs are trained using bootstrap aggregation, where each new tree is trained from a bootstrap sample of the observations. The three curves shown correspond to different ways of choosing the subset of features of the classifier, i.e. the size of the random subsets of features when spliting a node. As the number of estimators grows, all curves tend asymptotically to a minimum value of the OOB error shape. This implies, in principle, that the higher the number of estimators, the better the model (assuming the model has no overfitting). However, since more estimators also means more CPU consumption and one can see from Fig. 2 that above 300 estimators all three curves tend to stabilize, we have set the optimal number of trees to 371, for which we obtain the best error value for the sqrt algorithm.

## V. FEATURE SELECTION

With our classifier fully set, the next step is to select which features of our dataset are the most relevant and discard those that only add noise to the outcome. We have implemented two feature selection methods: one manual and one automated.

For the automated one we have implemented a small python script with a recursive variable elimination method. The idea is very straightforward: we train and evaluate the model with an increasing number of features each time. This will give us the score for each subset of features used. So, we will keep the one with the highest score.

We find out that, for our dataset, adding features to the model increases the score we obtain, up to a critical value, which in our case is 16 features, among which there are: CModelMag magnitudes, PSF, ModelMag and WISE magnitudes.From this point on, the score slowly drops. This is due to the fact that the model uses features that do not contribute value at the time of the classification, they only hinder the model and contribute noise.

For the manual selector we will check the scores obtained by the model when training it with different groups of magnitudes present in the dataset that we will choose conscientiously. We will analyze the different groups: [modcorrected, mod, petroMag, cmod, psf, and each one with Wise Magnitudes].

In Fig. 3 we obtain a higher score using the PSF magnitude together with the Wise Magnitudes. We also see that for all the magnitudes we will obtain a better score if we put them together with the Wise magnitudes. We should not be surprised, because our dataset is very large and these magnitudes give us other relevant information independent of the other magnitudes. That is, the information provided by the Wise magnitudes does not overlap with the photometric magnitudes. They are independent of each other.

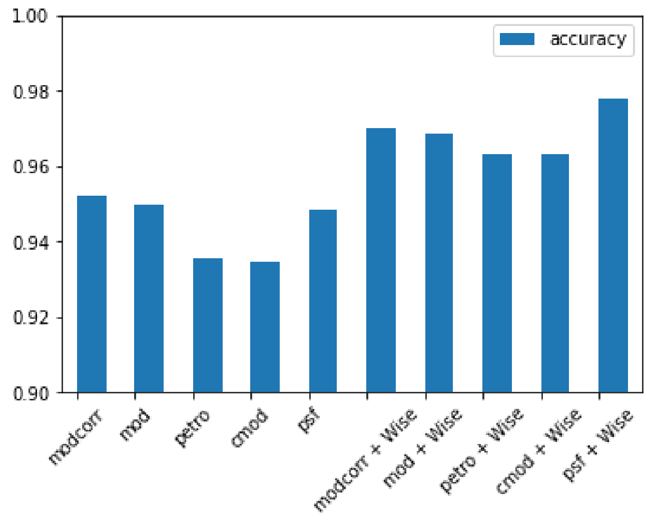

**FIG. 3:** Histogram of the accuracy values obtained by training our random forest classifier for each of the groups.

## VI. FILTERS

In this section we will apply some filters to the dataset to see what changes they imply in the accuracy of the model.

### A. Redshift

The astronomical redshift, z, is an extrinsic feature of astronomical sources that provides information on their radial velocity (if stars) or on their comoving distance from us (if galaxies and quasars). For the latter, a high value of z may also require to take into account evolutionary and cosmological corrections for some of the observables. In this section, we proceed to determine how the quality of the classification varies as a function of the value of this parameter.

Fig. 4 shows that the classification for low redshift sources is much better and more accurate (the model score is higher) than for high redshift sources. This indicates that the model can better learn and correlate the magnitudes and therefore provide an improved classification. When we increase the redshift, we include ”biased” (uncorrected) observations that add noise to the model. To get rid of this bias and provide correct information we would have to take into account the absolute magnitude of the objects, not the apparent one.

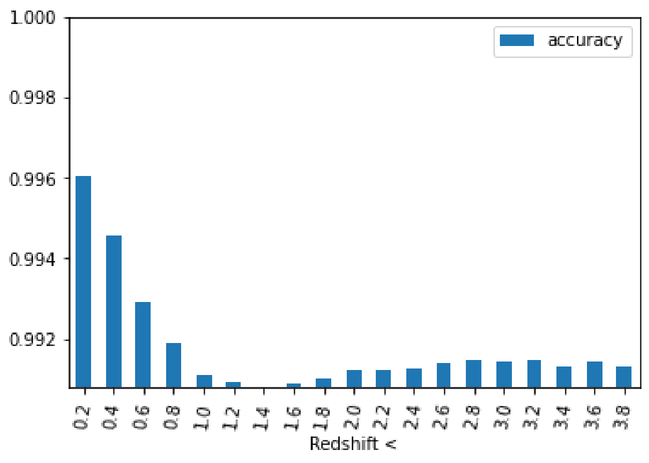

**FIG. 4:** Values of the accuracy obtained by training our RFC with subsets of sources below a given redshift.

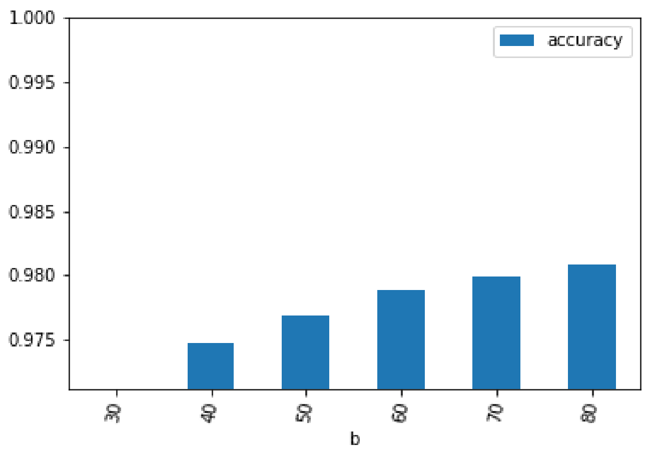

**FIG. 5:** Values of the accuracy obtained by training our RFC with the subsets of sources below a given b.

### B. Magnitude

This filter impose an upper value for the PSF magnitude in the r-band. We observe that for low and high magnitudes we obtain worse scores than for magnitudes around 16, where the accuracy peaks.

It is worth noting that in this case something different from the case of the redshift filter happens. We obtain low accuracy at low magnitudes due to the lack of data in the subset. We lack data to complement the model. On the other hand, we obtain a similar result for the redshift filter at high magnitudes. The model is worse for these cases. That is, when we increase the sources with higher magnitude we introduce noise in the model.

### C. Galactic Latitude

With this filter we will train the RFC with subsets of sources split by galactic latitude (symbol b), one of the two coordinates of a celestial coordinate system centered in the Sun which measures the angle of a source northward of the galactic equator as viewed from our star.

As we can see in Fig. 5, the farther away the values of b from the galactic equator, the better the accuracy. This is because high latitudes are less affected by the extinction from the dust in the galactic disk.

## VII. CONCLUSIONS

We used a dataset with more than 4 million astronomical sources extracted from the SDSS-DR16 to train a series of ML models to classify astronomical sources in order to find the best suited for this type of problem when using only photometric data. After cleaning our dataset of incomplete and/or doubtful data, the best model has proven to be a Random Forest classifier. We have also applied several filters to our clean dataset to determine the role played by different extrinsic and intrinsic observables on the accuracy of the classification. We have found that the optimal classifier achieves better scores for low redshift values and for PSF magnitudes around 16 mag (r-band). The accuracy also improves when one selects sources farther away from the plane of the Galaxy, which is understandable since these sources are less affected by galactic dust.

**Acknowledgments**

I would like to thank my advisors JM and Maria for his advices and corrections to improve my thesis. Thanks to my grandmother, my father and my aunt Carmen for trusting me during this stage. Thanks also to Rosana for being with me throughout this journey.

[1] Historical context of catalogs and big data *Science Direct Astronomical Catalogs* `www.sciencedirect.com/topics/physics-and-astronomy/astronomical-catalogs`

[2] More than 2000 astronomical catalogs *List of Astronomical Catalogs* `en.wikipedia.org/wiki/List_of_astronomical_catalogues`

[3] SDSS Filters *Wavelength of Filters bands* `www.sdss.org/instruments/camera/`

[4] The Astronomical Journal, Volume 140, Issue 6, pp. 1868-1881 (2010) *The Wide-field Infrared Survey Explorer (WISE): Mission Description and Initial On-orbit Performance*

[5] SDSS Object types *Morphology & Classification* `www.sdss.org/dr12/algorithms/classify/`

[6] Bolton et al *Spectral Classification and Redshift Measurement for the SDSS-III Baryon Oscillation Spectroscopic Survey* (2012)

[7] Scikit-Learn *Python package* `scikit-learn.org/`