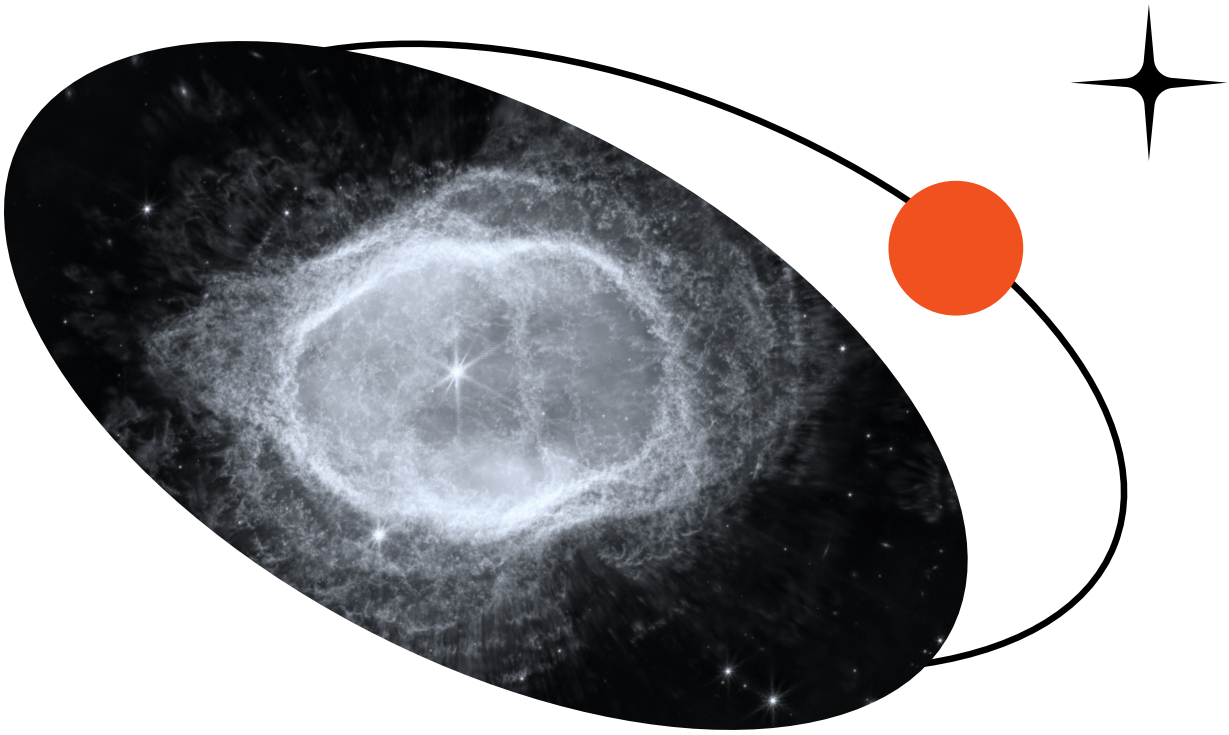
SAPIENZA
UNIVERSITÀ DI ROMA

Stellar Classification.

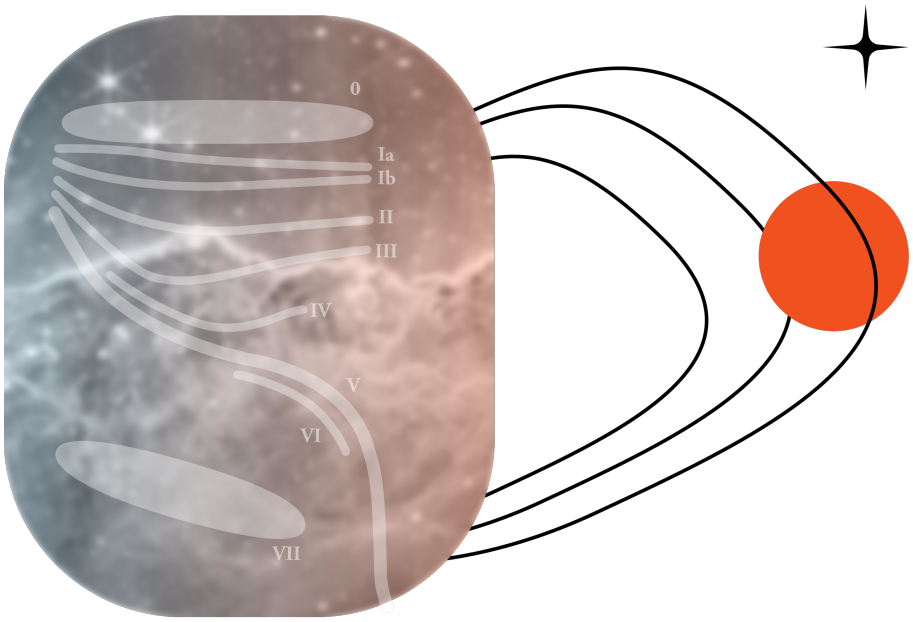
AI Lab project report.



AI Lab: Computer Vision and NLP
Prof. Daniele Pannone
Team: Giovanni Milone, Christian Gennarelli
Applied Computer Science and Artificial Intelligence, 2023

Introduction - The problem.

One of the most fundamental problems in astronomy is stellar classification. When a space agency carries out observations in the cosmos, each observed celestial body is stored in a database, which contains its own spectral characteristics. The Star Classification task is focused over categorizing and classifying stars based on these characteristics. This help astronomers extract more information about stars, such as composition, temperature, density, evolutionary stage and magnetic field.
The most widely used system for star classification is the *Morgan–Keenan (MK) system*, also known as the *Harvard spectral classification system*. This system classifies stars based on their spectral lines, which are absorption or emission features observed in the star's spectrum.



The size of datasets of aerospace agency observations can range from hundreds of thousands to billions of samples. This feature makes it necessary to automate the classification process. In particular, it is possible to generalize this problem to that of Multi-class Classification, which lends itself to being solved efficiently through machine learning techniques. Implementing these techniques ensures efficiency and scalability in the classification of large-volume star datasets.

Dataset Overview.

The data consists of 100,000 observations of space taken by the *SDSS* (Sloan Digital Sky Survey). Every data point is described by 17 feature columns and 1 class column which identifies it to be either a *star*, *galaxy*, or *quasar*.
Here is an overview of the dataset features:

1. `obj_ID:` Object Identifier, the unique value that identifies the object in the image catalog used by the CAS
2. `alpha:` Right Ascension angle (at J2000 epoch)
3. `delta:` Declination angle (at J2000 epoch)
4. `u:` Ultraviolet filter in the photometric system
5. `g:` Green filter in the photometric system
6. `r:` Red filter in the photometric system
7. `i:` Near Infrared filter in the photometric system
8. `z:` Infrared filter in the photometric system
9. `run_ID:` Run Number used to identify the specific scan
10. `rerun_ID:` Rerun Number to specify how the image was processed
11. `cam_col:` Camera column to identify the scanline within the run
12. `field_ID:` Field number to identify each field
13. `spec_obj_ID:` Unique ID used for optical spectroscopic objects (this means that 2 different observations with the same spec_obj_ID must share the output class)
14. `redshift:` redshift value based on the increase in wavelength
15. `plate:` plate ID, identifies each plate in SDSS
16. `MJD:` Modified Julian Date, used to indicate when a given piece of SDSS data was taken
17. `fiber_ID:` fiber ID that identifies the fiber that pointed the light at the focal plane in each observation

The model - Random Forest.

The machine learning model chosen in this instance is *Random Forest*: an ensemble learning method widely used for both classification and regression tasks. This algorithm combines the output of multiple *Decision Trees* in order to make more ponder predictions.

A Decision Tree is a flowchart-like structured supervised learning model where each internal node represents a feature or attribute, each branch represents a decision rule, and each leaf node represents an outcome or prediction. A decision tree works by recursively partitioning the feature space based on the statistical properties of the training data, in order to create homogeneous subsets of data at each node of the tree, leading to effective predictions. For the data to be partitioned at each level, it is required to choose a criteria to perform the splitting. Generally the two most widely used criteria are:

- *Gini Impurity:* the probability of incorrectly classifying a randomly chosen element in a set.

$$Gini(X) = 1 - \sum_{x \in X} p(x)^2$$

- *Entropy:* average amount of information required to identify the class of a randomly chosen element from the set.

$$H(X) = -\sum_{x \in X} p(x) \log_2(p(x))$$

Entropy was used as a partitioning parameter in the project.

At each step the split which minimizes the impurity is performed. Once the tree is constructed, predictions are made by traversing the tree from the root to a leaf node based on the feature values of the input data.

What Random Forest does, is building n decision trees over n random subsets of the training data, also by selecting random subsets of features at each node of the *i*-th decision tree in order to reduce correlation between trees and increasing variability in the model itself. Then, once the trees are trained, the most frequent class predictions will be those returned as output by the ensemble.

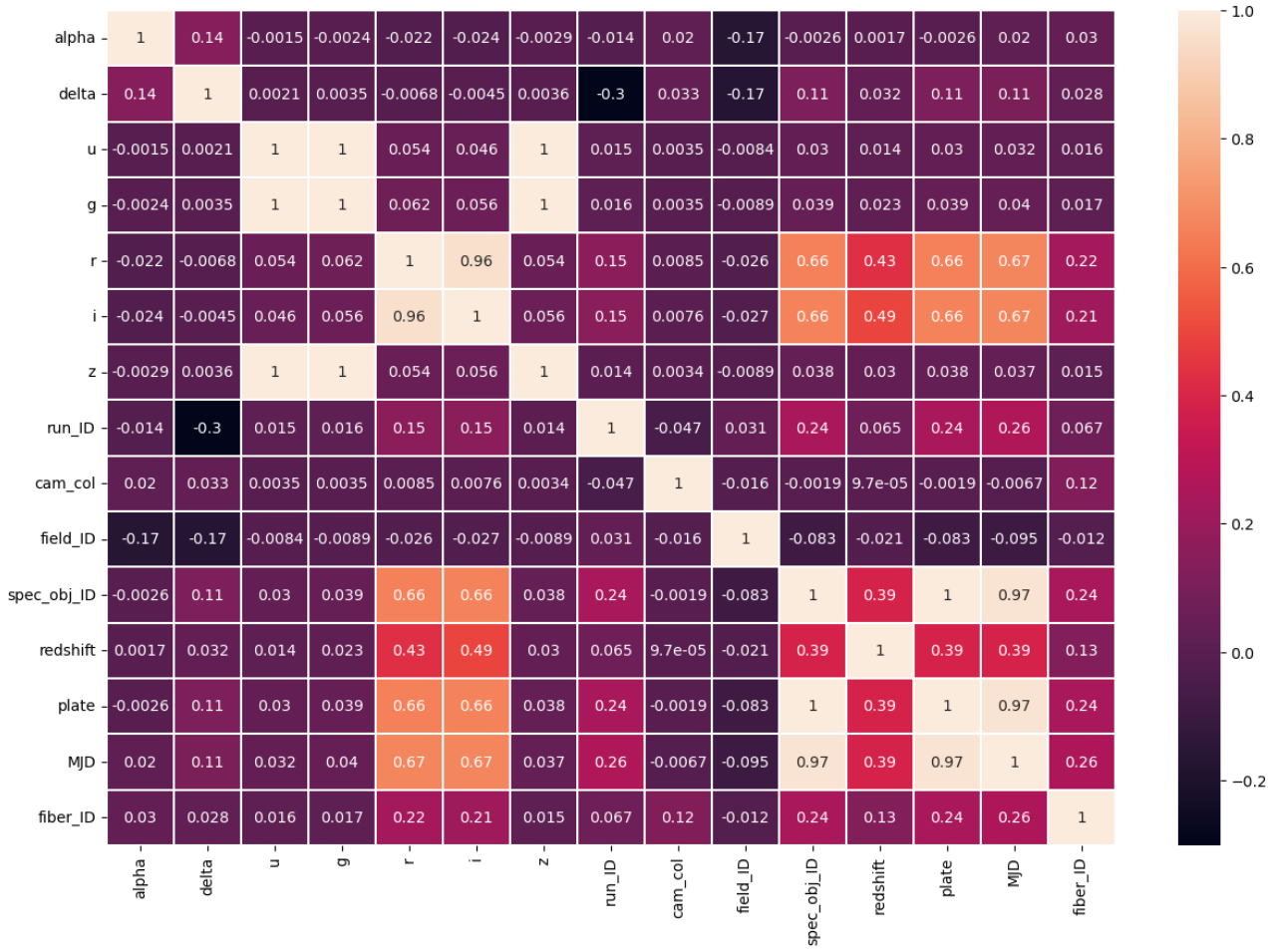
The challenge in building this model is choosing the correct number of hyperparameters, which will be addressed later in this report.

Preprocessing.

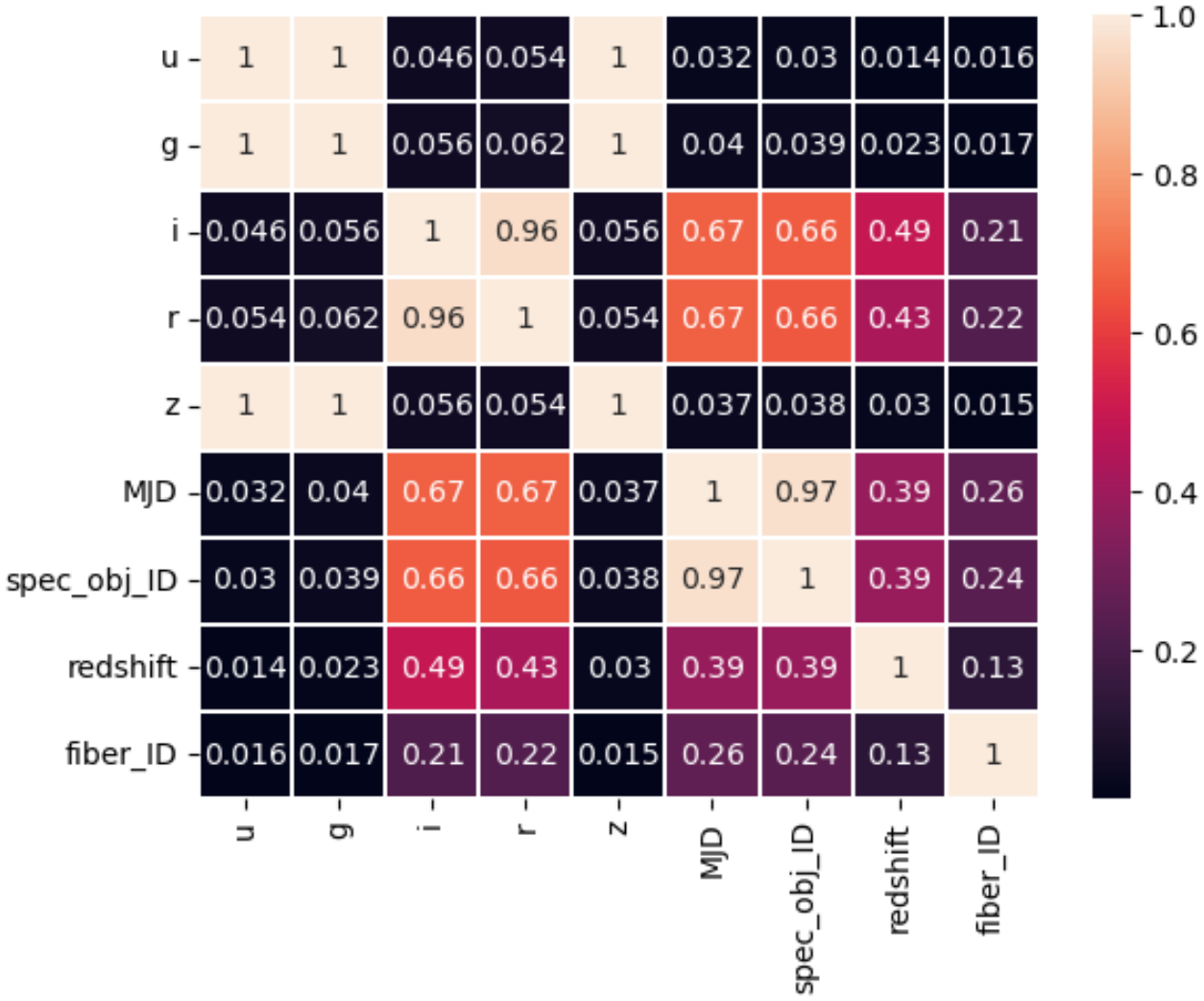
Before bringing data into the model, it's necessary to make sure they are properly processed to be usable and meaningful. The first step consists in encoding the `class` target variable with integer labels. The encoding in the project is the following:

- GALAXY = 0
- QSO = 1
- STAR = 2

Subsequently it is possible to proceed with the creation of the *correlation matrix* to study the degree of linear correlation between all variables. This should provide an overview of which of the variables we can expect to be more inferential for the model.



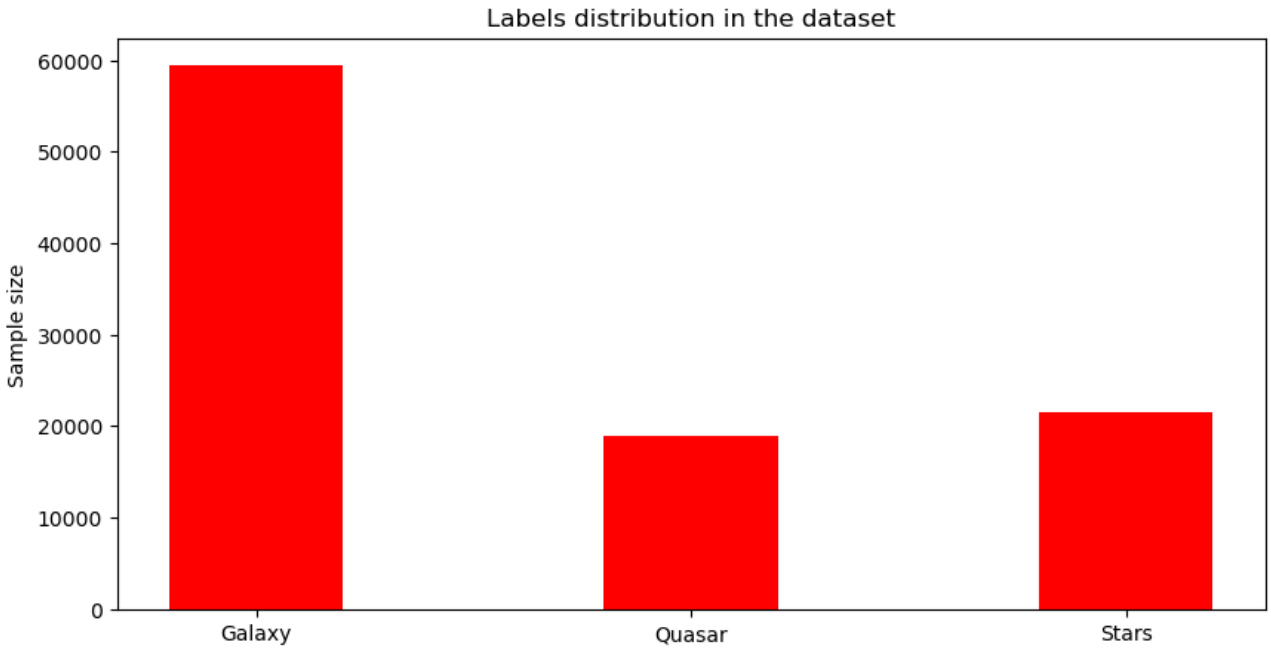
Before removing the most linearly irrelevant features, it is important to verify that there is no *multicollinearity* occurring among the variables as it could cause multiple problems for the model. The *Variance Inflation Factor (VIF)* could be helpful to tackle this by assessing how much the variance of the estimated regression coefficients is inflated due to multicollinearity. By computing VIF we find out that the `plate` variable has the highest value (2.09e+09) which corresponds to a severe degree of multicollinearity, therefore it is necessary to remove it.
After performing the removal of the undesirable features, we are left with 9 features.



It will be interesting to note later on how the removal of features will affect the performance of the model.
Then we proceed by *scaling the dataset* (i.e. standardize the data with z-score transformation) in order to reduce the gap between orders of magnitude for each feature to make them comparable.

Tackling Imbalanced Learning - Hybrid Sampling.

The current problem could be categorized as Imbalanced Learning, as a considerable gap between the frequency of the samples for each class of the dataset is evident. This problem could lead to an imbalance of classifications in favor of the largest class.



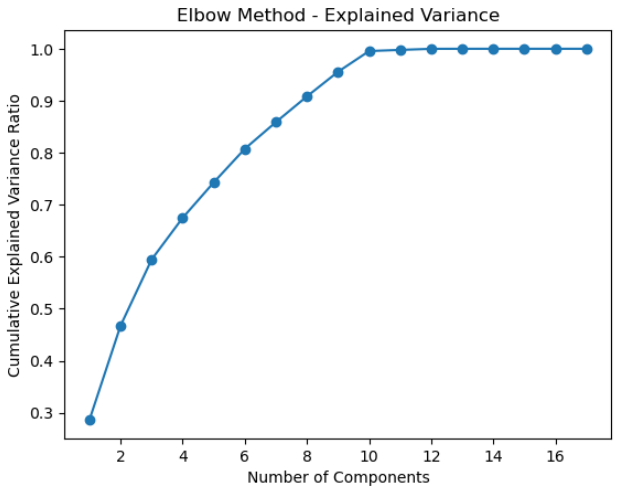
To overcome the gap, *oversampling* or *undersampling* can be applied on the data. However, since the gap is very large, using only oversampling would carry the risk of overfitting, while using only undersampling would translate into deletion of relevant samples. The solution lies in the use of *Hybrid sampling*, which involves simultaneously implementing both methods discussed. The oversampling will be implemented through *SMOTE (Synthetic Minority Over-sampling Technique)*, which works by creating synthetic samples for the minority class by interpolating between neighboring minority class samples. On the other side, the undersampling implementation is Tomek, which removes samples from the majority class that are close to the decision boundary between the minority and majority classes.

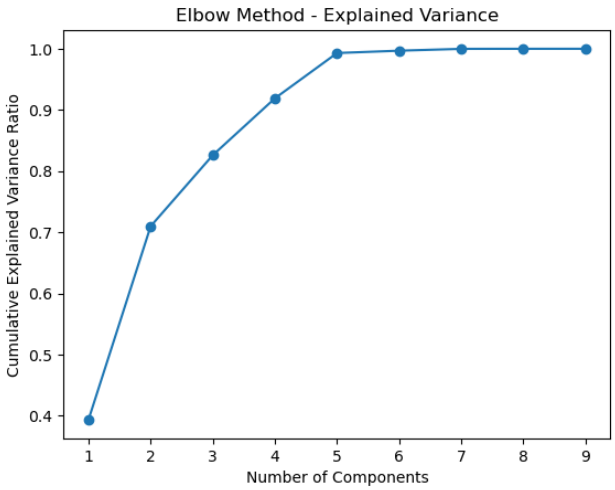
Once the data is balanced, it is possible to split the data between *training set* and *test set*, with 75% and 25% percentage distribution, respectively. In this instance, there is no need to generate a validation set as well, as the random forest classifier does not expect its use.

PCA.

The last step to consider before moving on to work on the model, is using *PCA (Principal Component Analysis)*. As a statical dimensionality reduction tool, it works by finding the principal components, which are linear combinations of the original variables that capture the most significant variations in the data, in order to project the data onto a lower-dimensional space that captures the most important patterns and relationships.

In this case it is critical to choose enough components to explain the largest possible amount of variance in the data. This can be achieved through the *elbow method:* a heuristic technique used to determine the optimal number of components in a dataset. It consists in finding the “elbow” point in a plot of explained variance against number of components. The plots were carried out both on the full 17-features dataset, and on the reduced 9-features one. In the first case a sharp “elbow“ is revealed in correspondence with the use of 10 components, while in the second case it occurs again with 5 components.





The tests on the model will also be aimed at observing how its performance varies based on the application of the PCA in different preprocessing scenarios.

Hyperparameters tuning - Grid Search.

Once the data has been preprocessed, the last step is to choose the hyperparameters for Random Forest. In particular, the parameters we are interested in are `max_depth` (maximum number of splits of the decision tree models within the ensemble) and `n_estimators` (number of decision trees to be included in the ensemble). In this phase it is important to choose carefully, as too narrow or too large depth values could lead to underfitting or overfitting. As for the number of trees in the ensemble, usually the more the better. However, there is a point of diminishing returns, where adding more trees may not significantly improve performance or may increase computational cost.
It is possible to verify how the choice of values for hyperparameters affects the efficiency of the model through grid search. It consists in evaluating the efficiency of each possible combination of multiple ranges of hyperparameters values, in this case there are two. In each of the tests the grid search was performed with `max_depth` values ranging from 1 to 6 and `n_estimators` ranging from 1 to 21.

Tests.

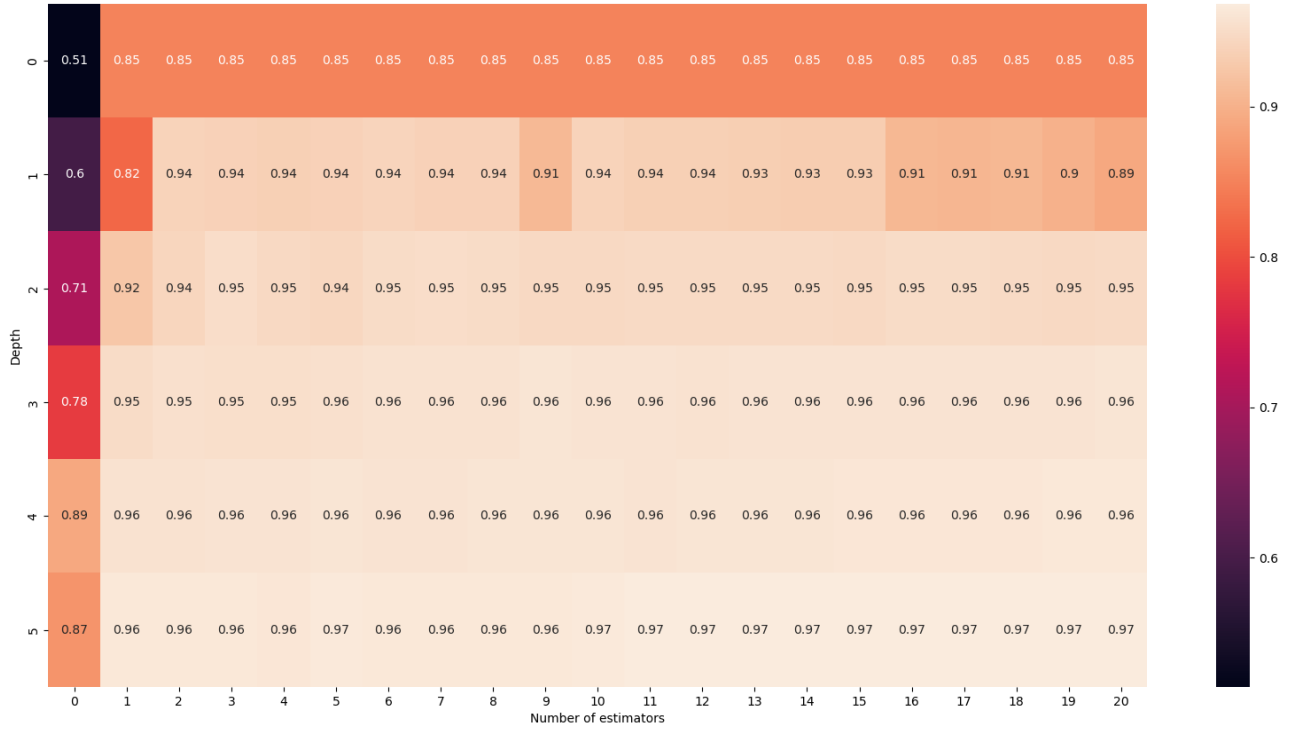
It is important to clarify that the primary objective of the tests is to verify how the performance of the model varies in relation to the type of preprocessing and in relation to the choice of using Random Forest rather than a single decision tree. The tests performed are organized according to the following specifications:

- Random Forest on 9 features hybrid sampled dataset.
- Single Decision Tree on 9 features hybrid sampled dataset.
- Random Forest on 9 features imbalanced dataset.
- Random Forest on 17 features hybrid sampled dataset.
- Random Forest on 17 features hybrid sampled dataset, with 10 components PCA.
- Random Forest on 9 features hybrid sampled dataset, with 5 components PCA.

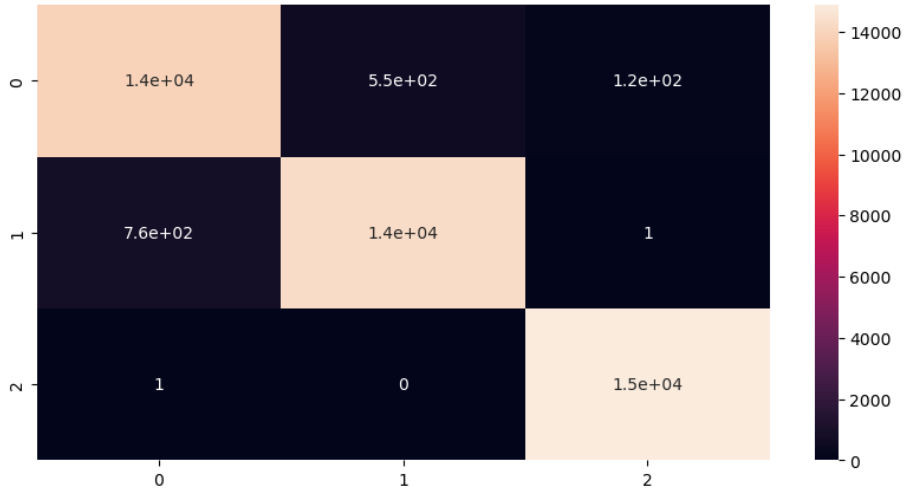
Therefore it is clear that the following tests are aimed at measuring the performance of techniques such as Hybrid Sampling, PCA and feature space reduction with respect to the problem in question and the model used.

Furthermore, the Random Forest on 9 features hybrid sampled dataset test will be considered as the main benchmark for the others.

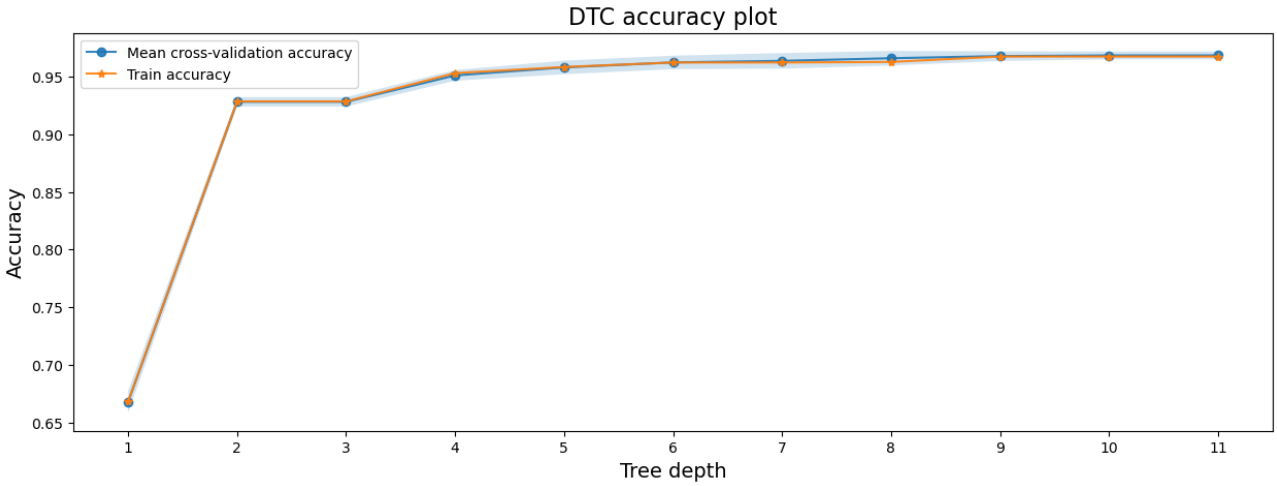
→ RFC, 9 features, Hybrid Sampling.



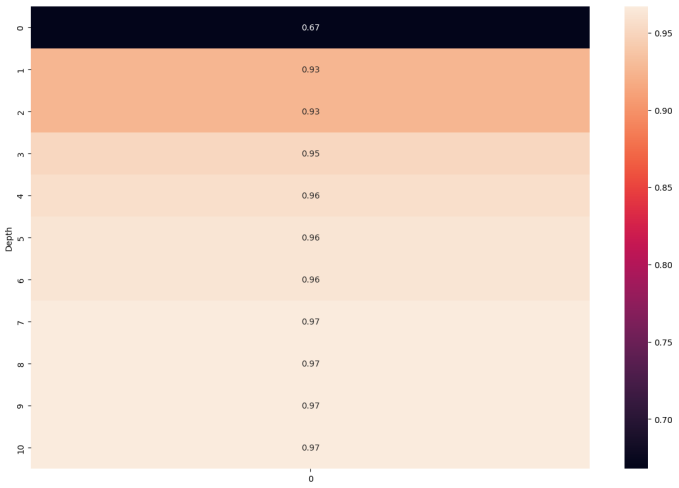
The highest accuracy score is obtained at `max_depth = 6`. By setting `n_estimators = 21`, the following confusion matrix is obtained.

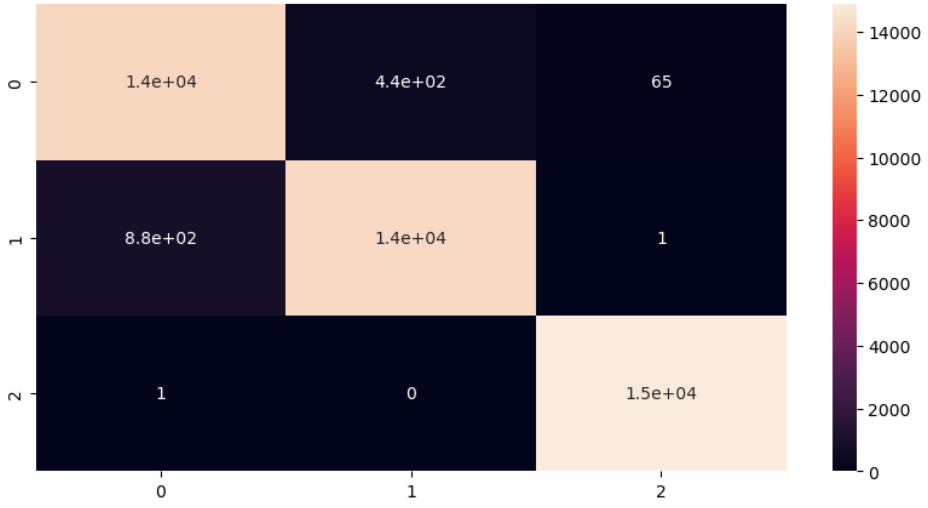


→ DT, 9 features, Hybrid Sampling.

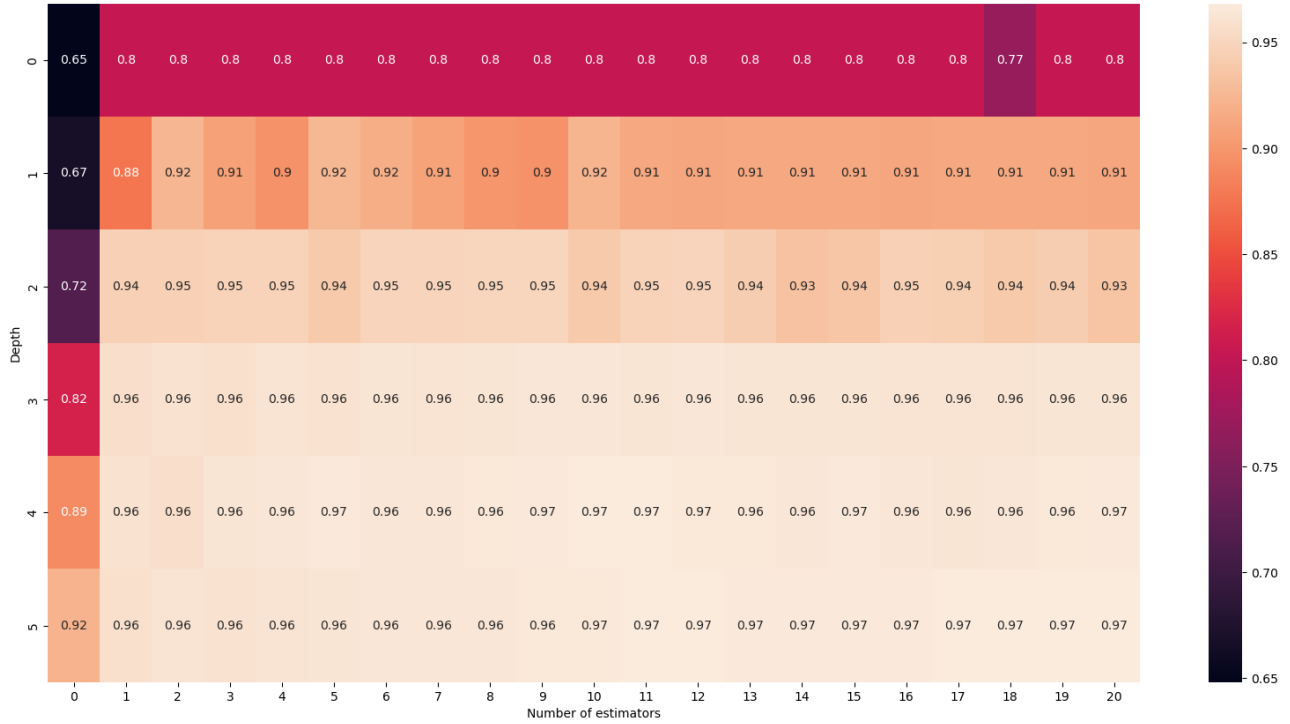


In this single Decision Tree test, *K-folds* was used as a cross-validation method. In practice this means that during the tests the decision tree was trained and evaluated on different subsets of the dataset (10 subsets in this case). From the tests it is evident that it is essentially useless to use values `> 9` for `max_depth`. From `max_depth = 8` on, a level of accuracy comparable to that of RFC is reached.

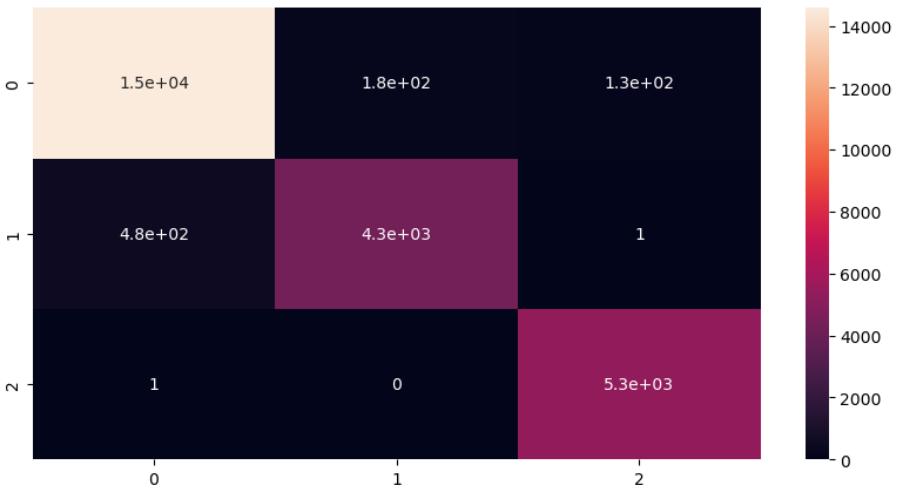




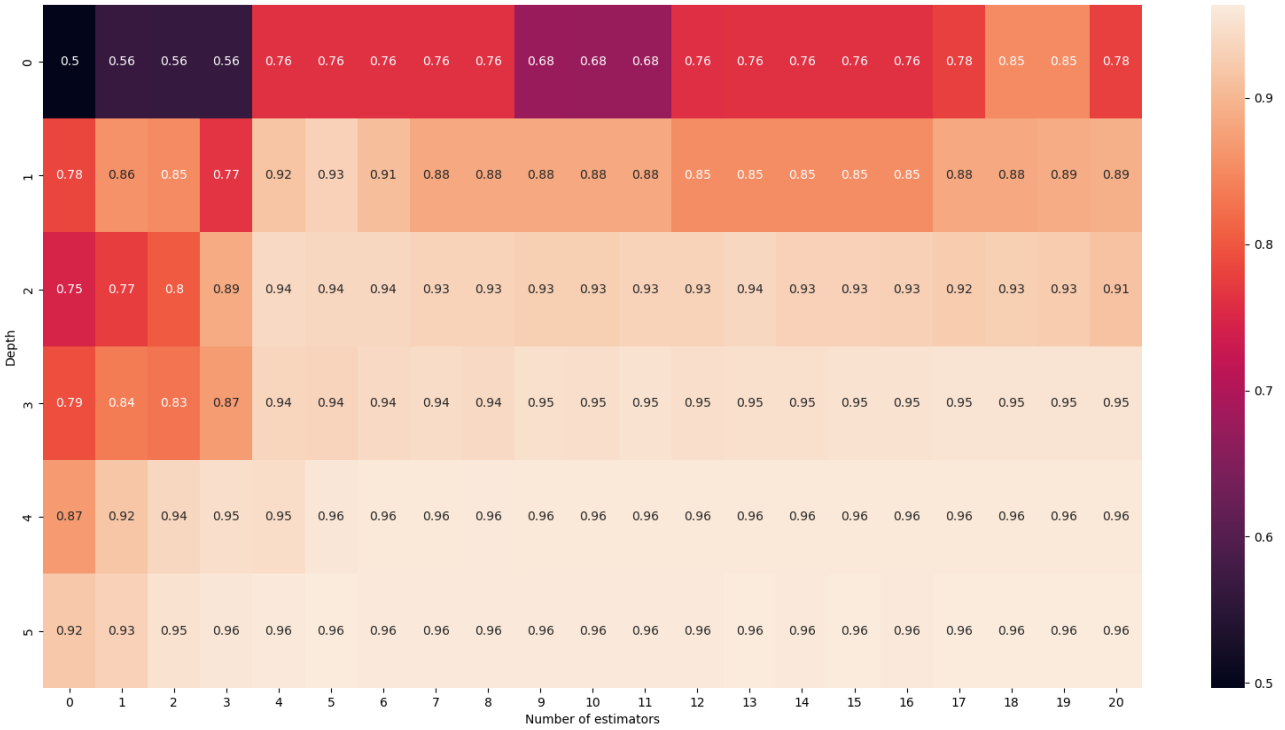
→ RFC, 9 features, imbalanced dataset.



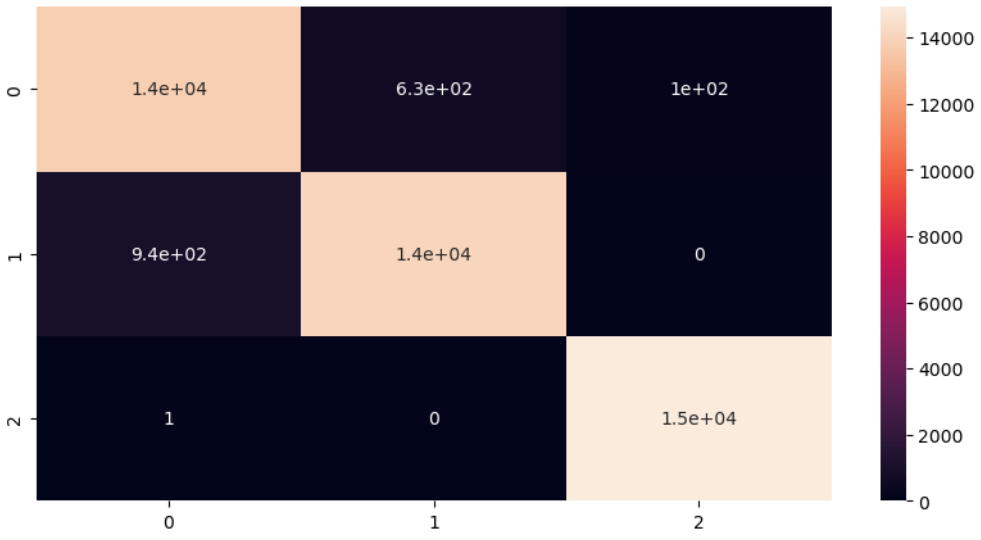
An interesting fact relating to this test in particular is that the approximate degree of entropy of the unbalanced dataset is 1.379, which compared to the average entropy of 1.585 of the other tests where hybrid sampling was used, appears to be the lowest value obtained throughout the experiment. This is due to the use of SMOTE-Tomek, which when applied allows to achieve a uniform distribution between lables. However, the aim of decision tree is to have in each region of the space defined by the splits, a distribution of labels as far as possible from the uniform one. Here is the confusion matrix for `max_depth = 6`, `n_estimators = 21`.



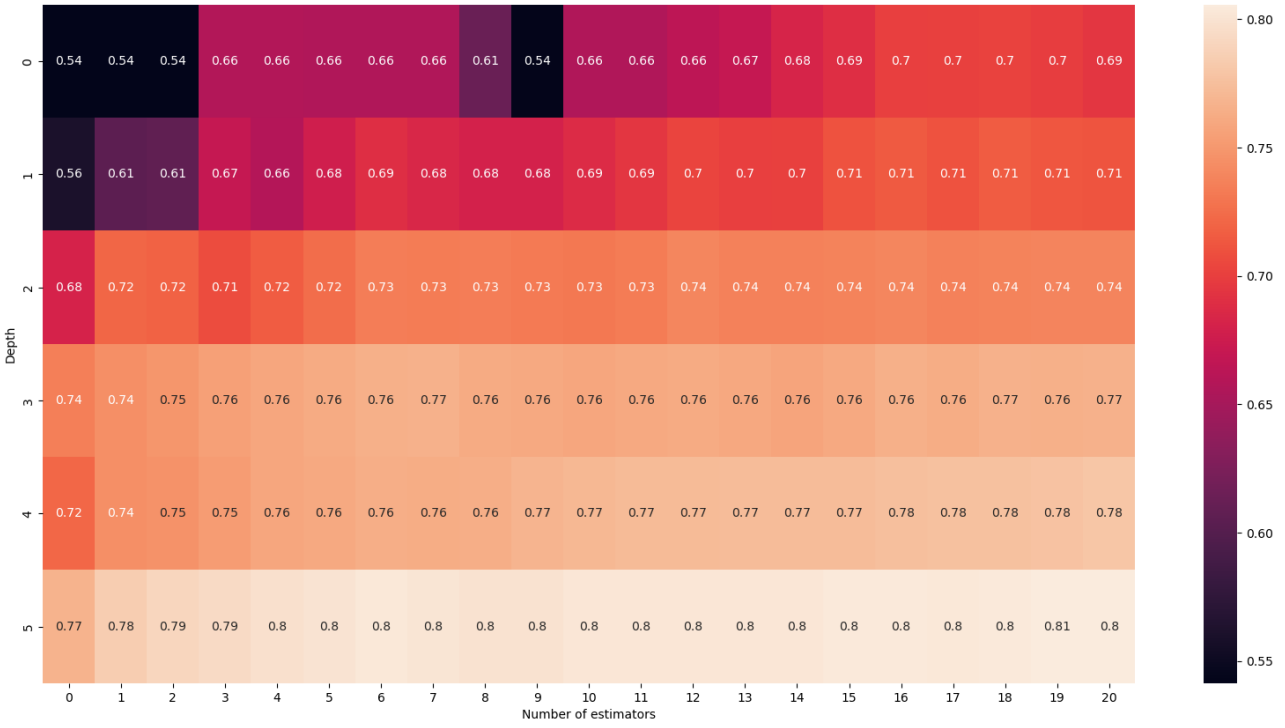
→ RFC, 17 features, Hybrid Sampling.



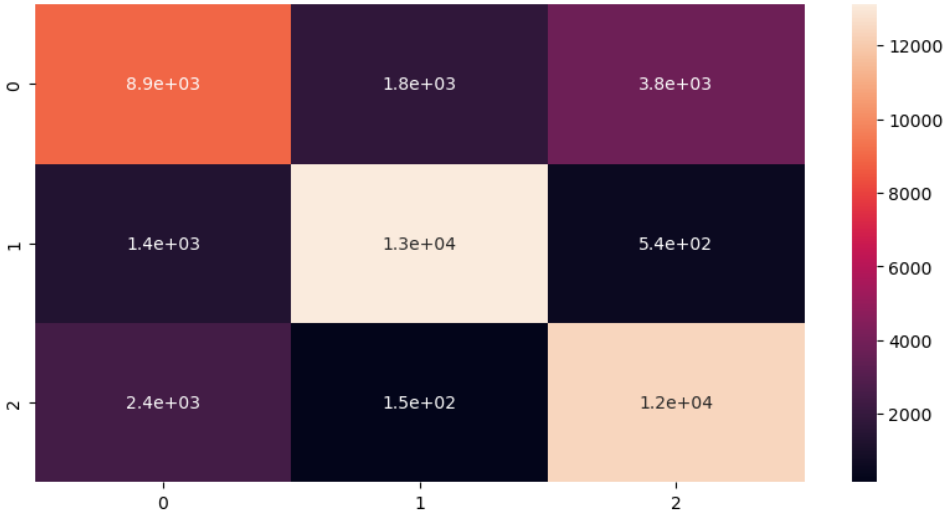
In this case it is immediately visible how with lower `max_depth` values, the model could behave significantly less accurately than the one trained on 9 features. This may be due to the presence of multicollinearity among the variables. Here is the confusion matrix for `max_depth = 6`, `n_estimators = 21`.



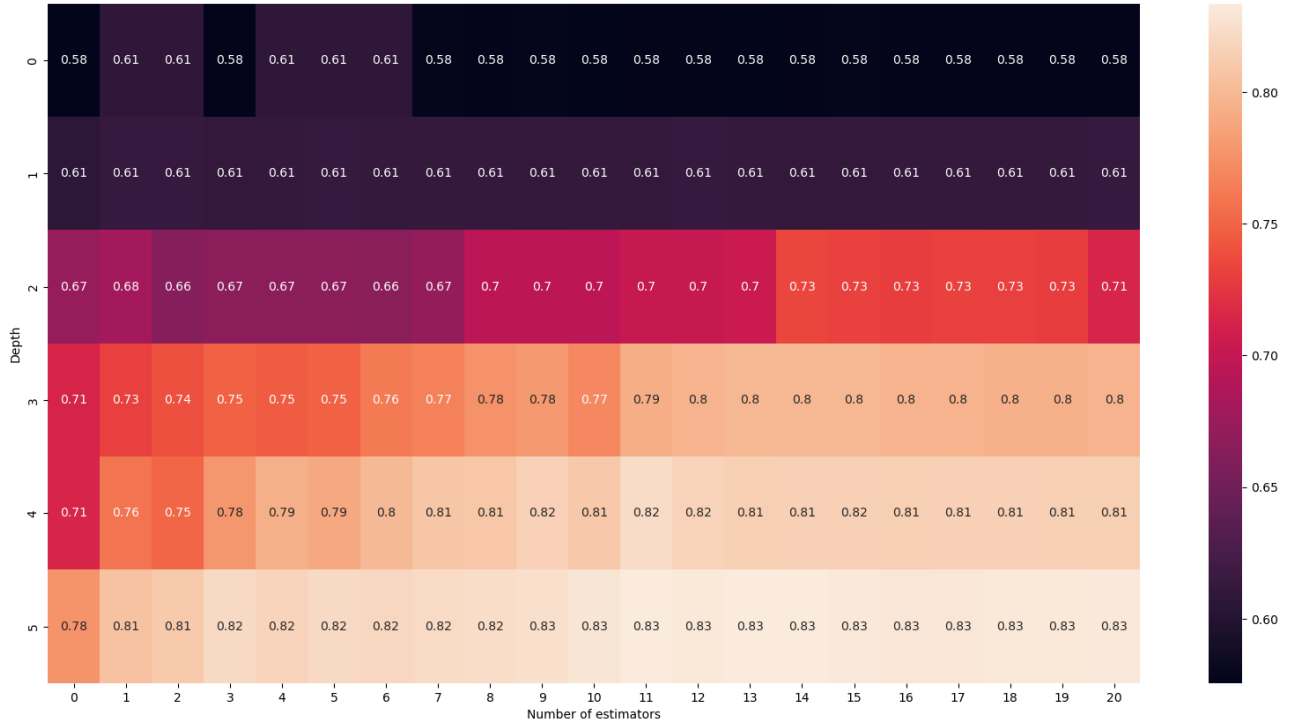
→ RFC, 17 features, Hybrid Sampling, PCA.



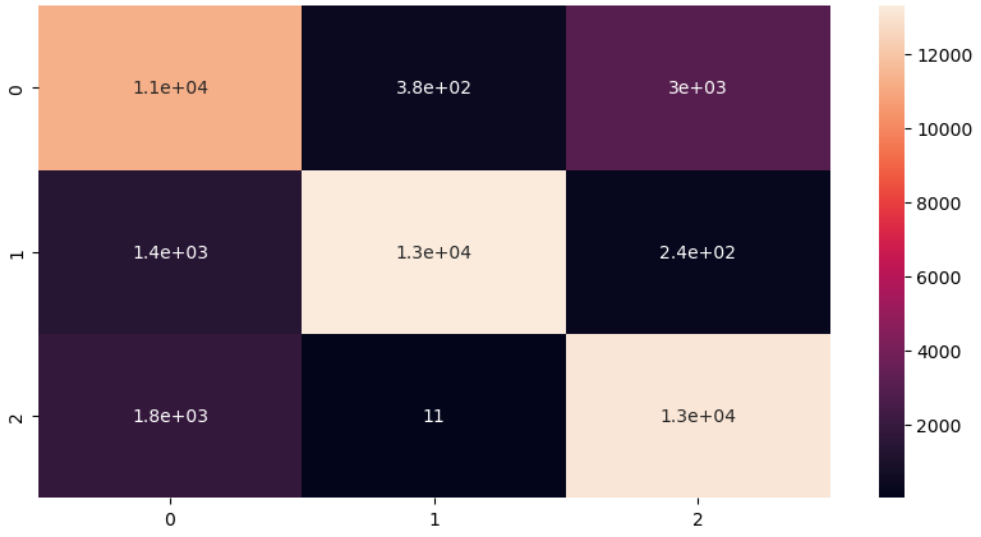
In this test, the task of carrying out all the dimensionality reduction was entirely assigned to the PCA. As previously specified, 10 components have been used in this text. it is evident how the results are qualitatively inferior to all the previous tests. Here is the confusion matrix for `max_depth = 6`, `n_estimators = 21`.



→ RFC, 9 features, Hybrid Sampling, PCA.



In this last test we want to verify how far it is considered safe to reduce the dimension of the dataset. Following what is indicated in the elbow test, the 9 features have been reduced to only 5 components. In this case it is also evident how the excessive dimensional reduction has made the model less accurate on average compared to all the previous tests. However, by keeping the same hyperparameters values (i.e. `max_depth = 6`, `n_estimators = 21`) the outcome of the confusion matrix confirms the model to be more reliable than the same based on 10 components.



In conclusion.

The highest degree of accuracy achieved in the tests was 0.98. The first test was found to be the most accurate by taking into consideration the average of the results of all the hyperparameter combinations tested. In the model described by the confusion matrix, the restricted 9-feature space was 0.2% more accurate than the 17-feature one.

The implementation of a single Decision Tree also worked surprisingly well compared to RFC. However, it can be expected that in general the RFC may perform better in the unseen data realm, given its more weighted structural nature.

The imbalanced dataset, which by default should facilitate the partitioning of the feature space within the model, proved to be equally reliable. It is clearly the test in which the least amount of false positives was produced, as it is also the one with the smallest dataset.

So far, PCA tests have yielded the worst results. However, it cannot be denied that the problem may be due to the choice of the number of components. The purpose of the experimentation was to verify its performance in relation to the values highlighted in the elbow test, which turned out to be very evident, as can be seen from the graphs. However, exists the possibility that the choice of a different number of components could have positively altered the performance of the model.