Bank Marketing Campaign Data Analysis

Thesis for the exam of Data Spaces

Academic Year 2020/21

Student: Marigo Tommaso

INDEX

INTRODUCTION

In this work the Bank Marketing Dataset was analyzed, which can be downloaded free of charge from the UCI Machine Learning Repository:

https://archive.ics.uci.edu/ml/datasets/bank+marketing

The dataset collects information from customers of a Portuguese bank between May 2008 and November 2010 during a marketing campaign conducted by the bank, carried out by telephone on its customers. The bank has proposed to its customers a subscription for a term deposit.

The goal of the study was to provide an accurate classifier that would allow the bank to contact those customers most likely to subscribe.

The work was conducted with *Python 3.8* with the help of the following libraries:

- *Pandas* and *Numpy* for data structures.
- *Imblearn* for rebalancing algorithm.
- *Sklearn* for classification algorithms.
- *Matplotlib* and *Seaborn* for the realization of the graphs.

DATASET ANALYSIS

The dataset is composed of 41188 records of user data and 21 variables for each of them, collected during their choice of subscription.

Customer Information:
1. *age*: age of client (numeric).
2. *job*: type of job (categorical).
3. *marital*: marital status (categorical).
4. *education*: level of education (categorical).
5. *default*: whether the customer has a default credit or not (binary).
6. *housing*: whether the customer has a housing loan or not (binary).
7. *loan*: whether the customer has a loan or not (binary).

Information on the last contact during the marketing campaign:
8. *contact*: contact communication type (categorical).
9. *month*: last contact month of year (categorical).
10. *day_of_week*: last contact day of the week (categorical).
11. *duration*: last contact duration, in seconds (numeric).

Other attributes:
12. *campaign*: number of contacts performed during this campaign and for this client (numeric).
13. *pdays*: number of days that passed by after the client was last contacted from a previous campaign (numeric, 999 means client was not previously contacted).
14. *previous*: number of contacts performed before this campaign and for this client (numeric).
15. *outcome*: outcome of the previous marketing campaign (categorical).

Social and economic context attributes:
16. *emp.var.rate*: employment variation rate - quarterly indicator (numeric).
17. *cons.price.idx*: consumer price index - monthly indicator (numeric).
18. *cons.conf.idx*: consumer confidence index - monthly indicator (numeric).
19. *euribor3m*: euribor 3 month rate - daily indicator (numeric).
20. *nr.employed*: number of employees - quarterly indicator (numeric).

Output variable (desired target):
21. *y*: has the client subscribed a term deposit? (binary: "yes", "no").

This is a highly unbalanced dataset. Only 4640 said yes, the remaining 36548 refused the term deposit.

Preliminarily I chose to ignore missing values.

In addition, the following attributes were removed from the analysis:
• *month* and *day_of_week*: redundant for classification.
• *duration*: as suggested by UCI, this is a value known only after the call.
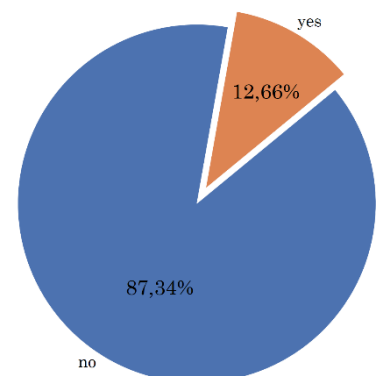
Let's analyze these variables in detail.

Figure 1: Dataset samples

2.1 NUMERIC VARIABLES

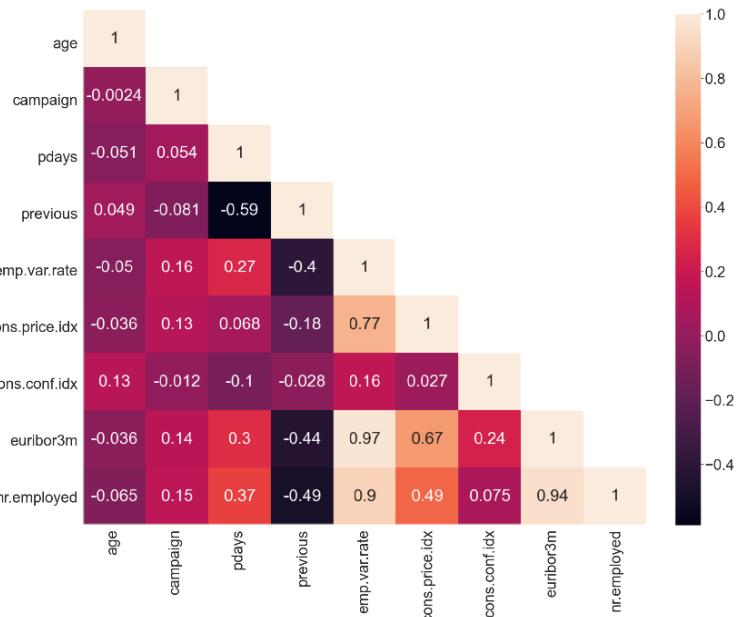Let's look at the correlation matrix with the Pearson index:

Figure 2: Correlation Matrix

The variables that represent the social and economic context are generally the most correlated, especially *emp_var_rate*, *nr_employed* and *euribor3m*. In addition, we can see also a strong anti-correlation between these variables and *previous* attribute.

Correlations can be observed graphically using scatter plots:
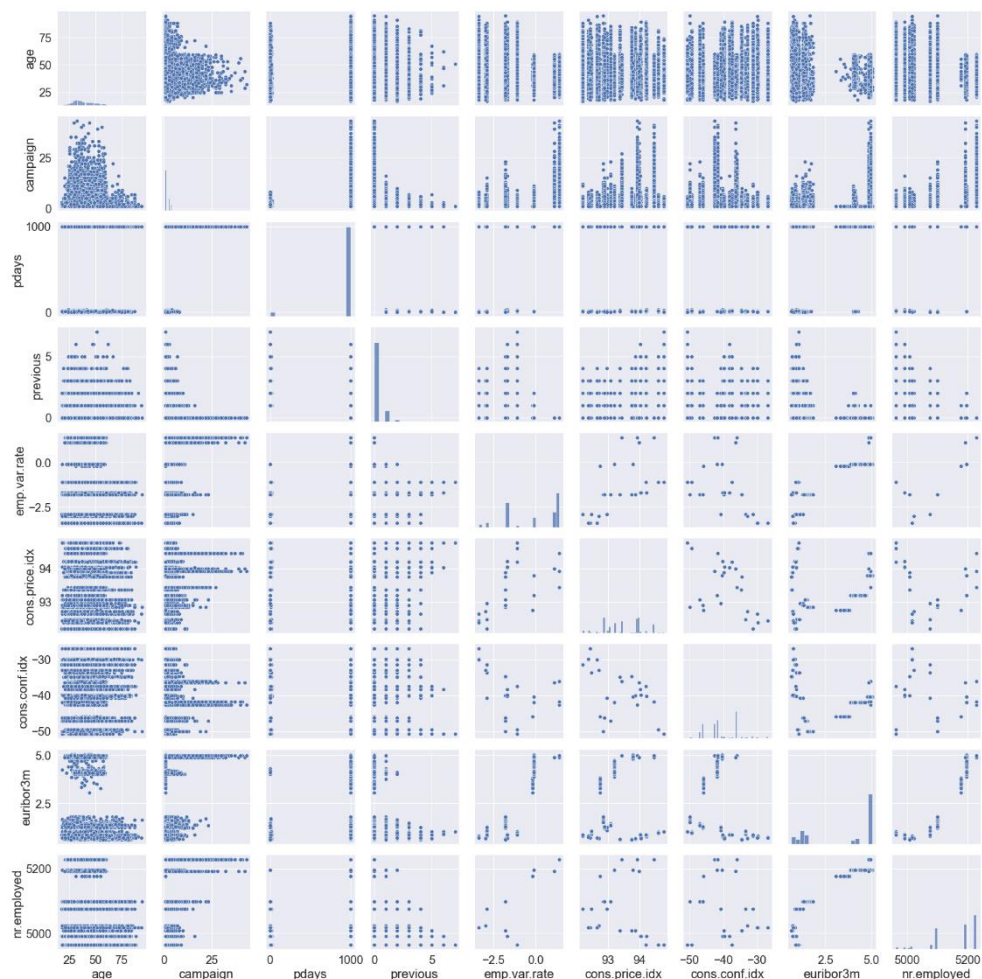

Figure 3: Scatter Plots

For each numeric variable we will observe and comment on the following things:

- The Boxplots.
- A table of measures i.e. minimum, 1st quartile, median, mean, standard deviation, 3st quartile and maximum.
- A graph showing the distribution in the dataset using the Kernel Density Estimation.

Age

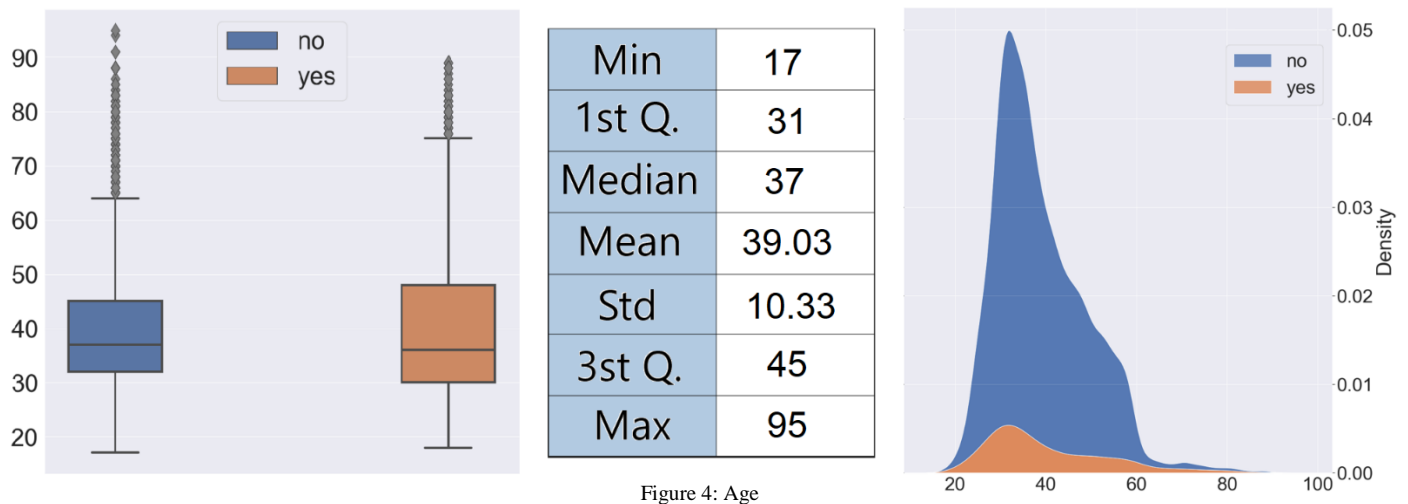| Min	17
1st Q.	31
Median	37
Mean	39.03
Std	10.33
3st Q.	45
Max	95

Figure 4: Age

In an age range from a low of 17 to a high of 95, we have the data slightly unbalanced to the left, with an average of around 39. The body of both boxplots pretty much cover the same values so this variable is not a very good indicator.
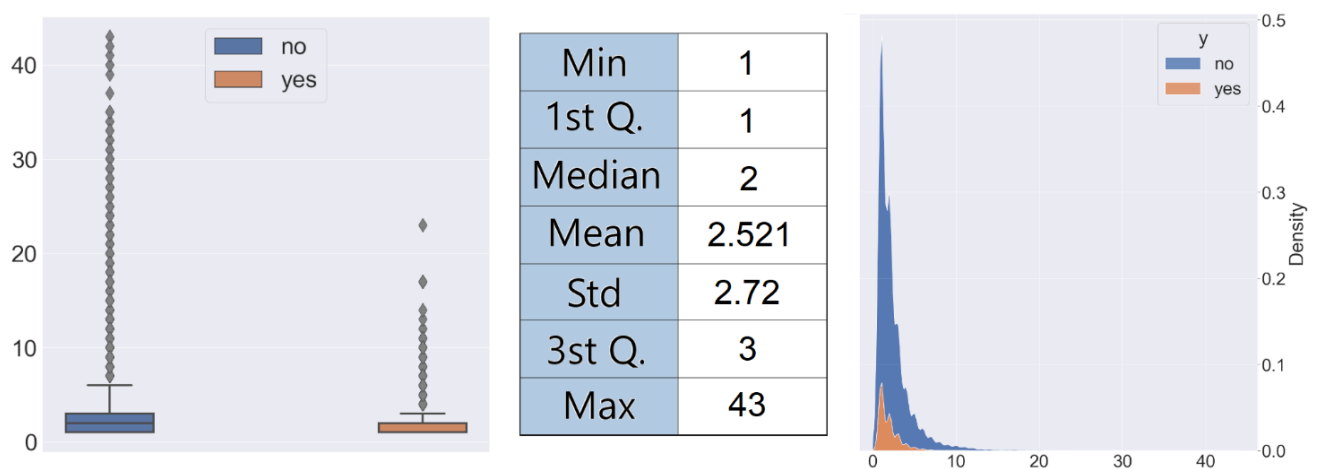
Campaign

Min	1
1st Q.	1
Median	2
Mean	2.521
Std	2.72
3st Q.	3
Max	43

Figure 5: Campaign

As expected, most of the values are between 0 and 3 with an important presence of outliers.

Previous Days

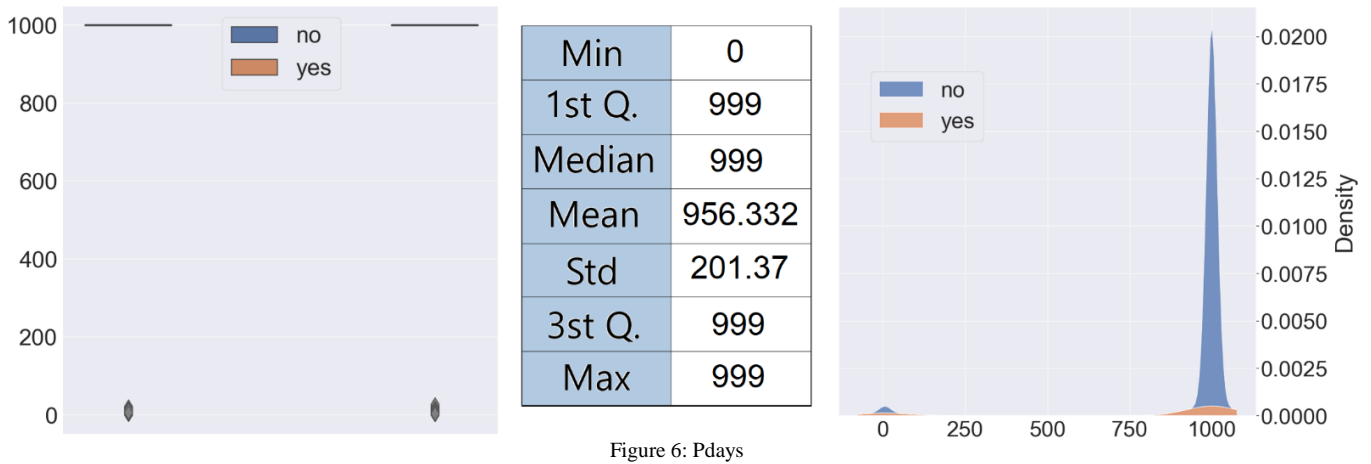|	
Min	0
1st Q.	999
Median	999
Mean	956.332
Std	201.37
3st Q.	999
Max	999

Figure 6: Pdays

Most customers had never been contacted before (*pdays* = 999). We could have done an analysis by taking into consideration only the customers previously contacted at least once, but (as we also see from the graphs) in the context of the classification the latter would be considered only as noise.

Previous

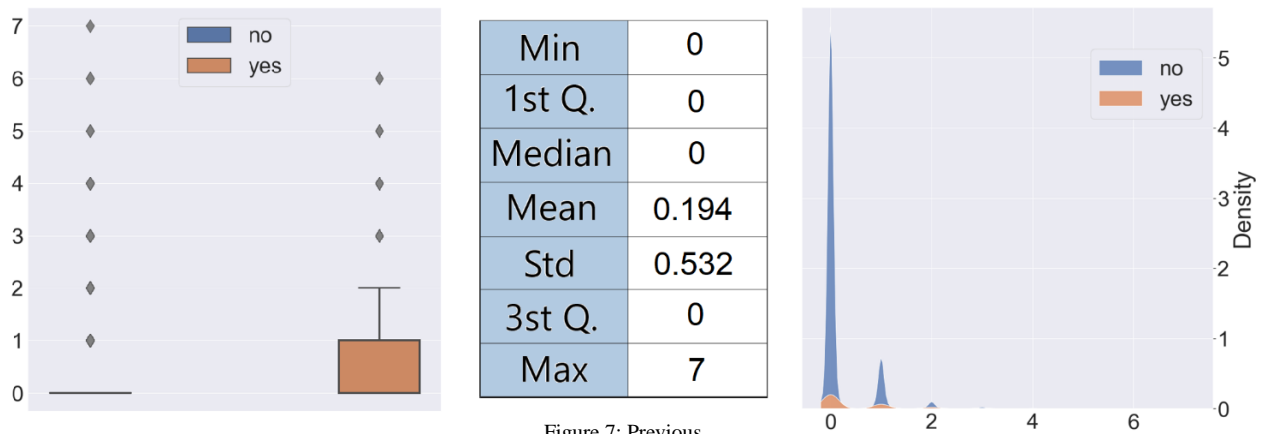|	
Min	0
1st Q.	0
Median	0
Mean	0.194
Std	0.532
3st Q.	0
Max	7

Figure 7: Previous

If we look at the boxplots, we notice that, although most customers have never been contacted before, there is a good chance that customers contacted at least once in the past will respond positively to the term deposit proposal.

Employment Variation Rate

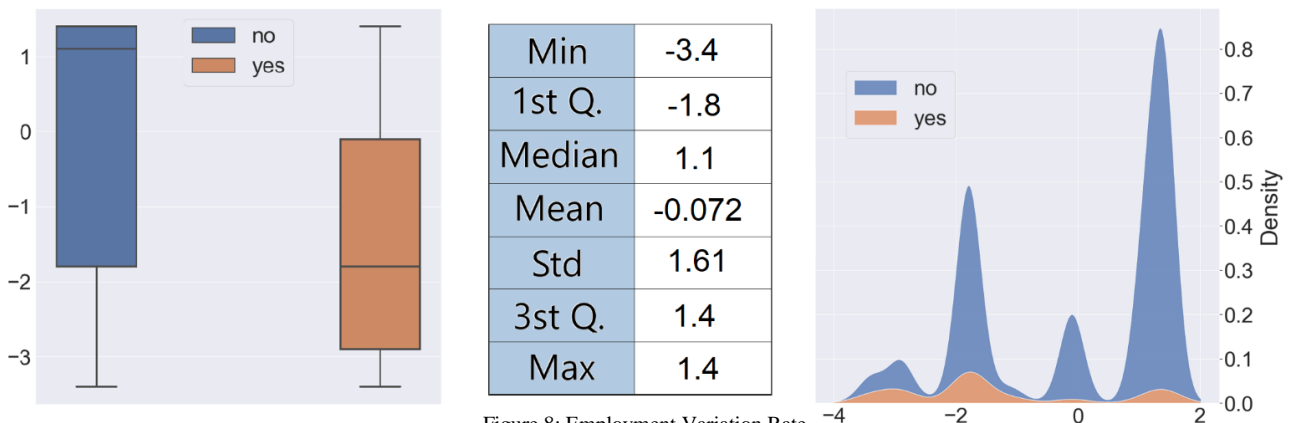|	
Min	-3.4
1st Q.	-1.8
Median	1.1
Mean	-0.072
Std	1.61
3st Q.	1.4
Max	1.4

Figure 8: Employment Variation Rate

The variability is quite small. We observe from the boxplots that the medians are significantly different, in particular with an increasingly negative *emp.var.rate* the customer tends to accept the proposal.

Consumer Price Index

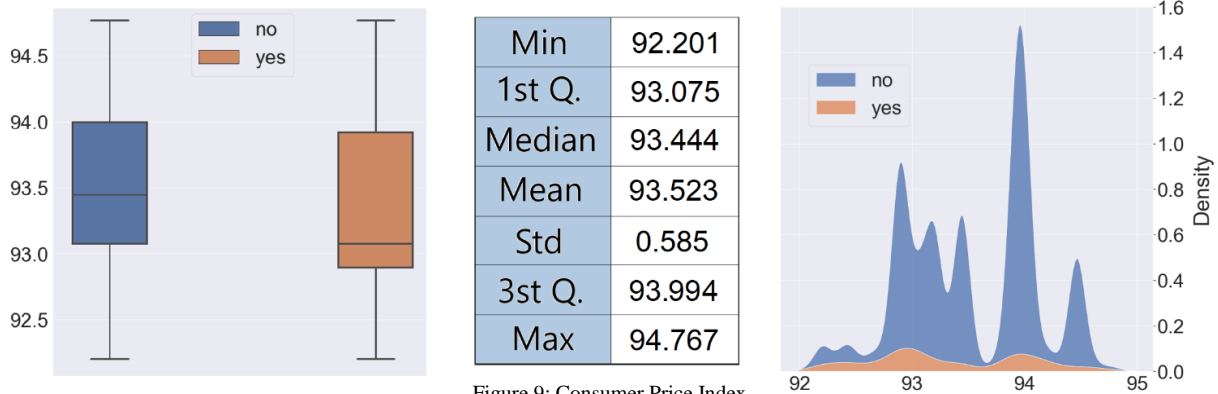|	
Min	92.201
1st Q.	93.075
Median	93.444
Mean	93.523
Std	0.585
3st Q.	93.994
Max	94.767

Figure 9: Consumer Price Index

We have the bodies of the boxplots practically on the same range but also in this case the two medians are slightly different. To confirm the fact that the lower this index is, the more the customer tends to accept the term deposit in this case as well. Let us remember that these variables that describe the socio-economic context are correlated with each other (Figure 2).

Consumer Confidence Index

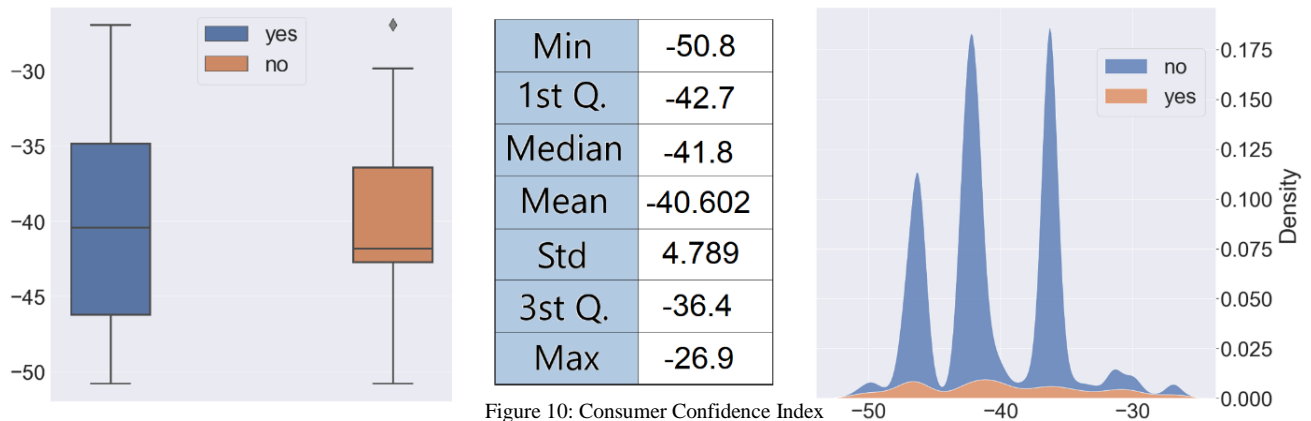|	
Min	-50.8
1st Q.	-42.7
Median	-41.8
Mean	-40.602
Std	4.789
3st Q.	-36.4
Max	-26.9

Figure 10: Consumer Confidence Index

The variability in this case is large. The body of the orange boxplot is tighter and contained in a range that goes from about -43 to -34. Also in this case (observing the medians) we can see that, with a lower index, customers tend to answer "yes" to the term deposit.

Euribor

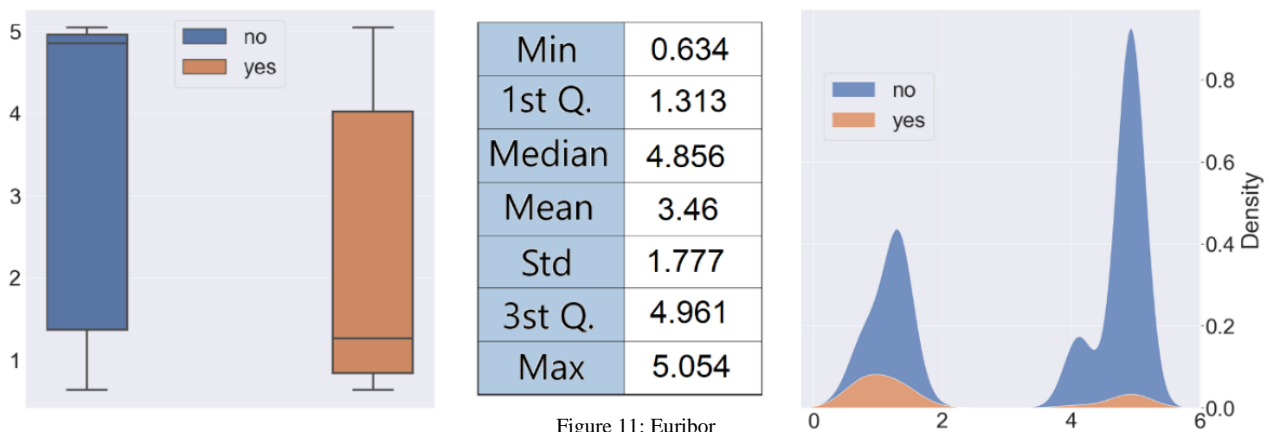|	
Min	0.634
1st Q.	1.313
Median	4.856
Mean	3.46
Std	1.777
3st Q.	4.961
Max	5.054

Figure 11: Euribor

The *euribor* is the variable apparently among the most significant in the context of classification. The median of the blue boxplot is very close to the maximum, while the median of the other boxplot is very close to the minimum.
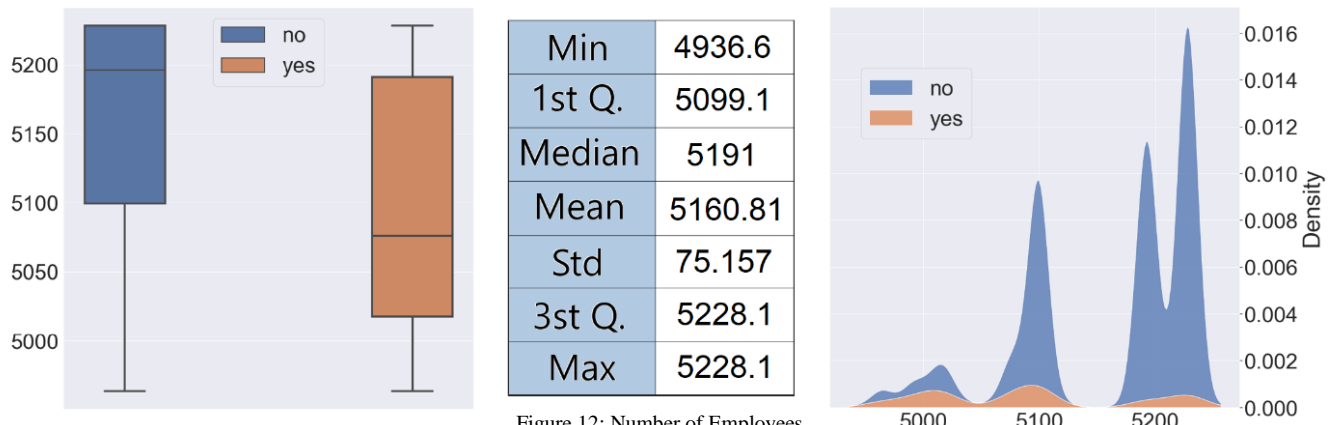
Number of Employees

Min	4936.6
1st Q.	5099.1
Median	5191
Mean	5160.81
Std	75.157
3st Q.	5228.1
Max	5228.1

Figure 12: Number of Employees

This variable also confirms what we have previously said.

2.2 CATEGORICAL VARIABLES

Let's now take a look at categorical variables. For each of them we will see the distribution of values and the success rate for each value.
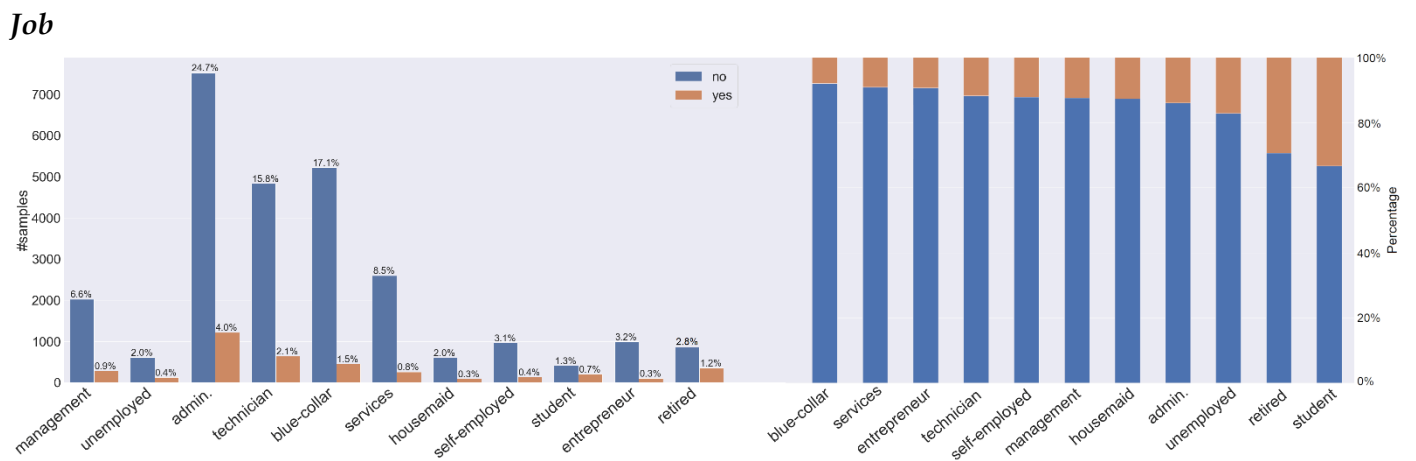
Job

Figure 13: Job

It is noted that the professions with the highest percentage of subscriptions are Administrative Staff and Technical Specialists. We also note that students and retirees have a high proportion of "yes".
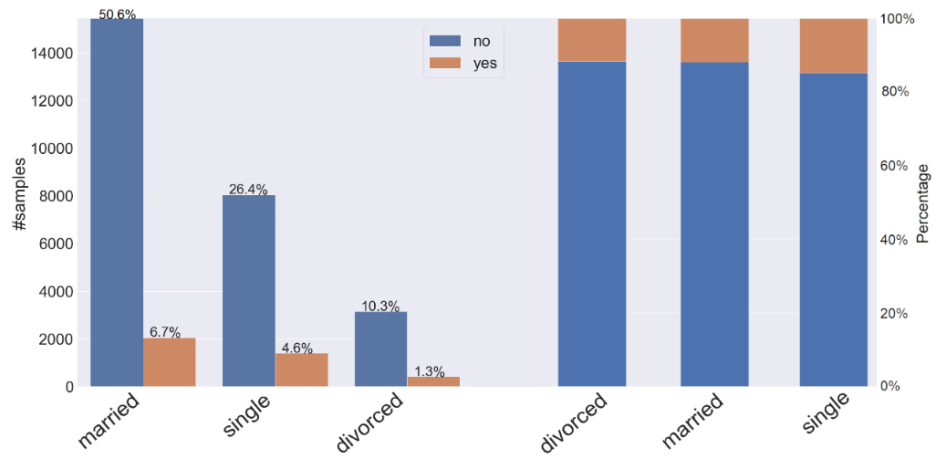
Marital

Figure 14: Marital

Married people, of all those who said "yes", are more in terms of numbers. In proportion, however, single people seem to be more favorable to subscription.
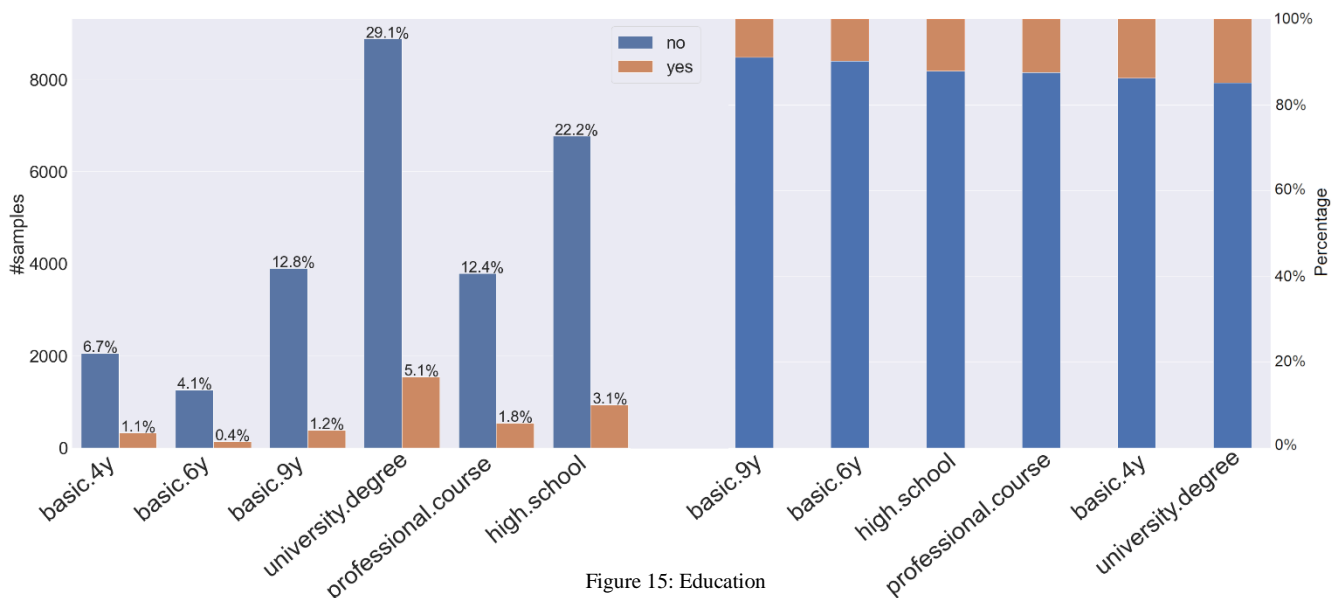
Education

Figure 15: Education

People with a university degree appear to be the most favorable to subscription, both in percentage and proportion.
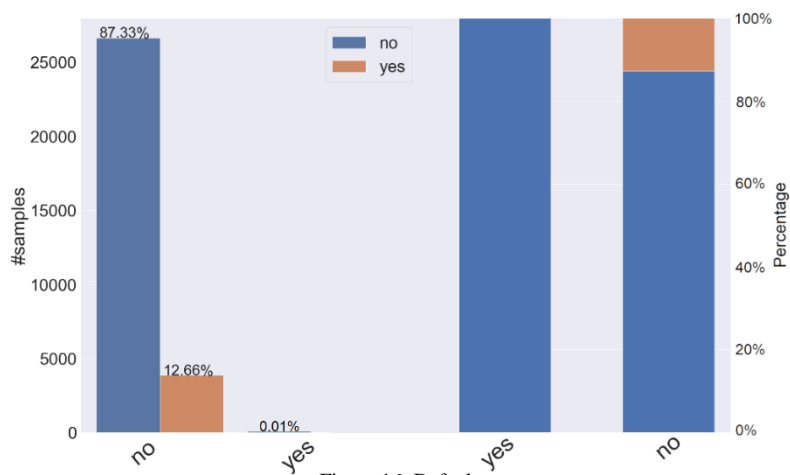
Default

Figure 16: Default

In the whole dataset, there are only 3 people who have a default credit. This variable is so unbalanced that we could consider to eliminate it.
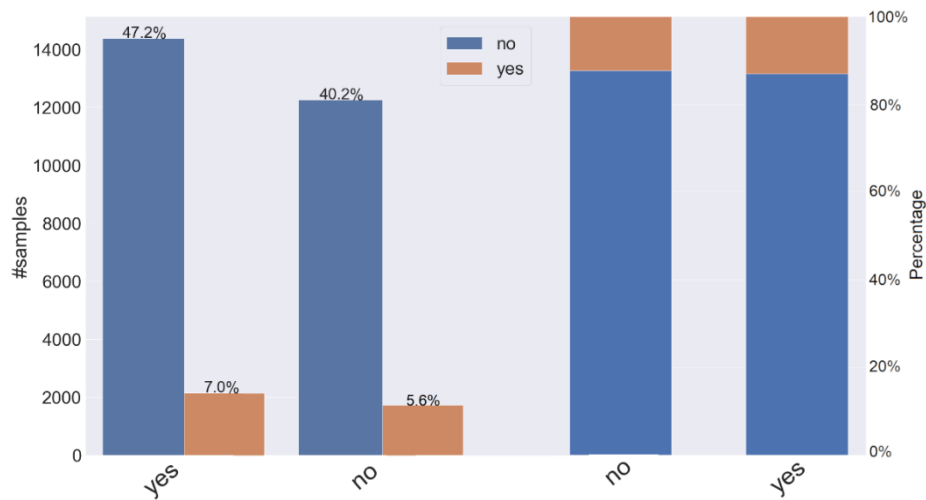
Housing

Figure 17: Housing

In this case we observe an almost total balance. Home ownership does not greatly affect marketing company performance.
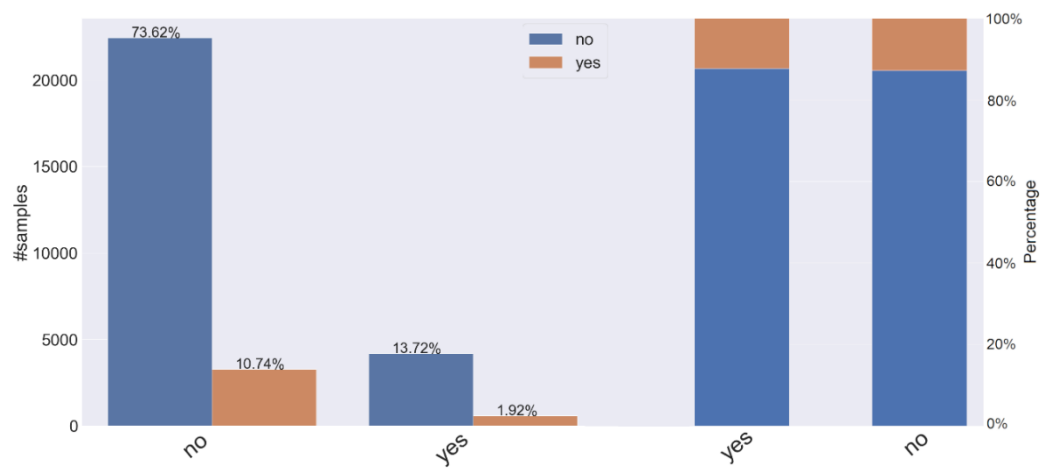
Loan

Figure 18: Loan

The dataset contains many more customers who have a loan. Also, in this case we observe an almost equal proportion in both cases.
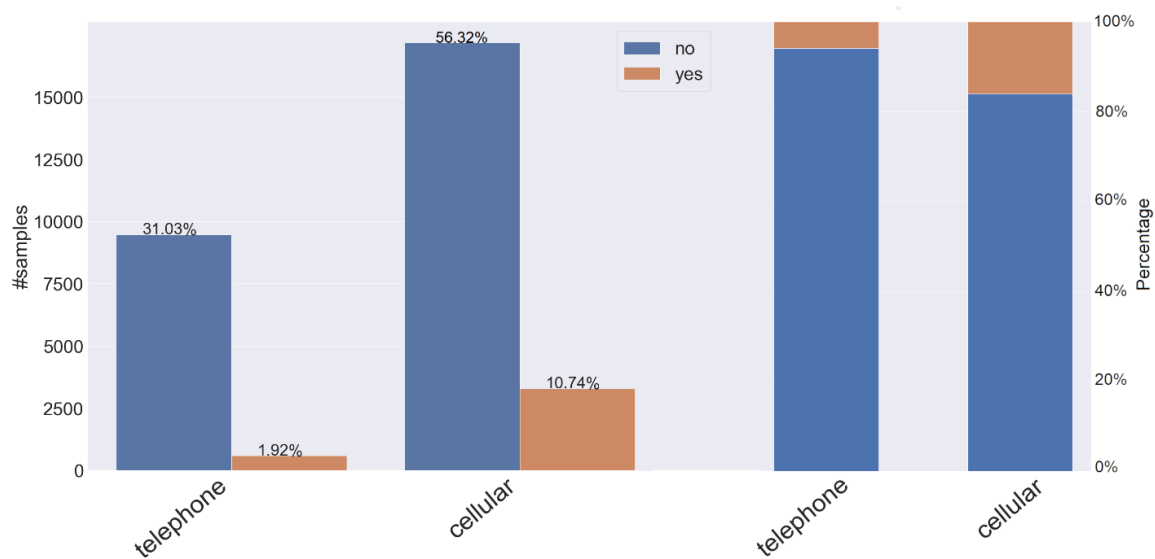
Contact

Figure 19: Contact

The best communication channel seems to be cellular.
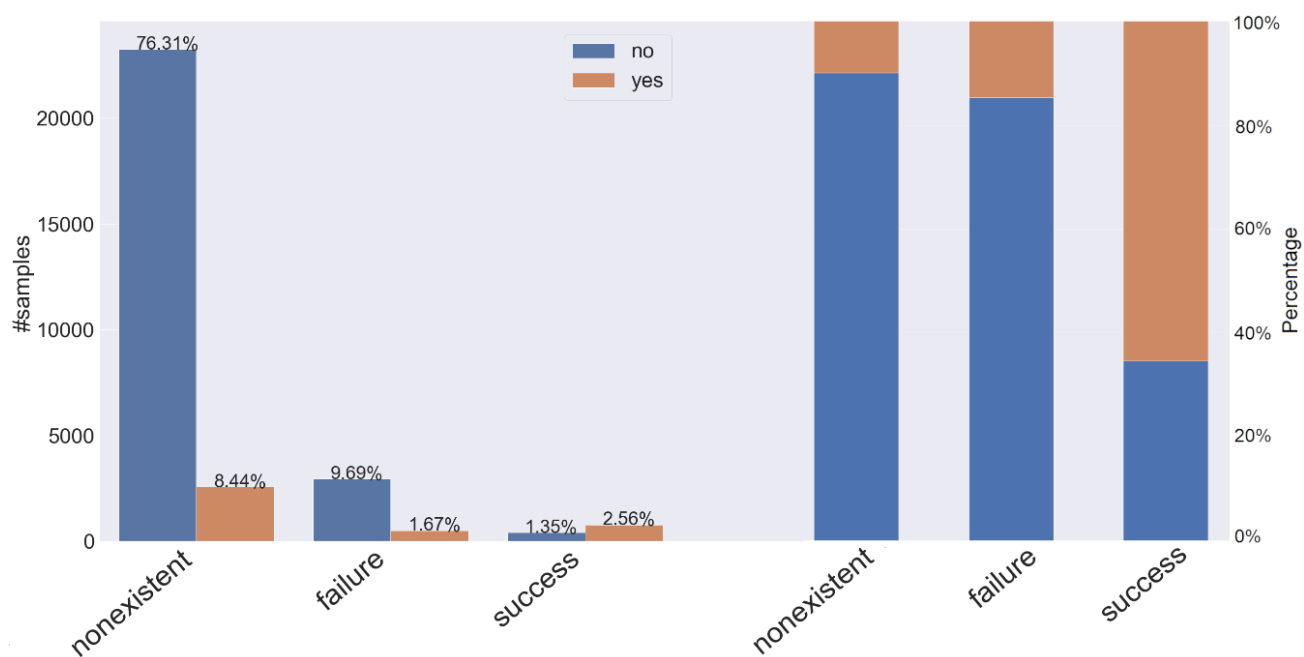
Previous Outcome

Figure 20: Previous Outcome

Most of the customers who gave a positive response to the previous marketing campaign also responded positively to this.

2.3 DATA PREPROCESSING

I have chosen not to make any changes to the numeric variables.

As regards the categorical variables, the following operations were carried out:

- Elimination of the *default* variable.

- Elimination of rows with "unknown" values .

- The values of the binary variables *housing*, *loan* and *contact* have been encoded in numbers with the LabelEncoder class of the sklearn library.

loan	contact		loan	contact
no	telephone		0	0
yes	cellular		1	1
no	cellular		0	1
no	telephone		0	0

Figure 21: LabelEncoder

- For the remaining variables (*job*, *marital*, *education* and *poutcome*) the dataset does not suggest a sensible sorting criterion for this classification problem, so it was decided to encode them with the One Hot Coding method of the sklearn library. This method adds an attribute for each variable domain value we want to encode.

marital		marital_divorced	marital_married	marital_single
married		0	1	0
married		0	1	0
single		0	0	1
divorced		1	0	0

Figure 22: OneHotEncoder

In summary, the dataset will have the following characteristics:

- 38245 records.

- No missing values.

- 37 variables: 9 numeric and 27 binary.

CLASSIFICATION MODELS

In this section we will look at the behavior of various classification algorithms with our problem. In the end, we will compare them and choose the best one based on the results.

For our analysis, the following models were chosen:

- Logistic Regression
- K-Nearest Neightbors
- Decision Tree
- Random Forest
- Support Vector Machine

Preliminarily, the dataset will be mixed several times and divided into *training set* and *test set* with a ratio of 9:1. That is, the *test set* will make up 10% of the dataset.

Before the training phase for each algorithm will be carried out a search for the best hyperparameters through a K-Fold Cross Validation. During this tuning phase the *validation set* will make up 15% of the *training set*.

The models have always been trained with a balanced dataset using the SMOTE algorithm of the *imblearn* library. This algorithm oversamples the data by choosing a sample and a neighbor (among K neighbors), from the minority class, and generating a synthetic sample between these two.

Thanks to this technique the algorithm will give the same importance to both classes and (in general) we expect that it will tend to decrease the number of false negatives, at the expense of precision and accuracy.

However, after a rapid test with default parameters, I have seen that SMOTE increases f1-score in general for any of the chosen algorithms:

	without SMOTE	with SMOTE
Logistic Regression	0,284	0,380
KNN	0,325	0,370
Decision Tree	0,317	0,346
Random Forest	0,220	0,325
SVM	0,320	0,358

Figure 23: F1-score with SMOTE

After the tuning phase, the accuracy and the f1-score of the model will be calculated using the *test set*.

Finally, some considerations will be made by analyzing some graphs that is the ROC Curve, the Confusion Matrix and the Learning Curve.

3.1 LOGISTIC REGRESSION

This model is a regressor and as such, it tries to approximate the trend of the data with a function. The function is a logistic function that maps real numbers into values between 0 and 1, which we will interpret as the probability that a data belongs to a class or not:

$$P(Y = 1|x) = \frac{e^{\beta_0 + \beta x^T}}{1 + e^{\beta_0 + \beta x^T}}$$

Logistic Function

To determine the best weights β, a tuning phase will be performed on the following sets of hyperparameters:

- Penalty in *[l1 , l2]:* the type of penalty in the cost function.
- C in *[100, 10, 1.0, 0.1, 0.01]:* inverse of regularization strength, smaller values specify stronger regularization.

Result

The K Fold Cross Validation suggested penalty equal to *"l1"* and C equal to 10.
We therefore see the result on the test set through the ROC curve and the confusion matrix...
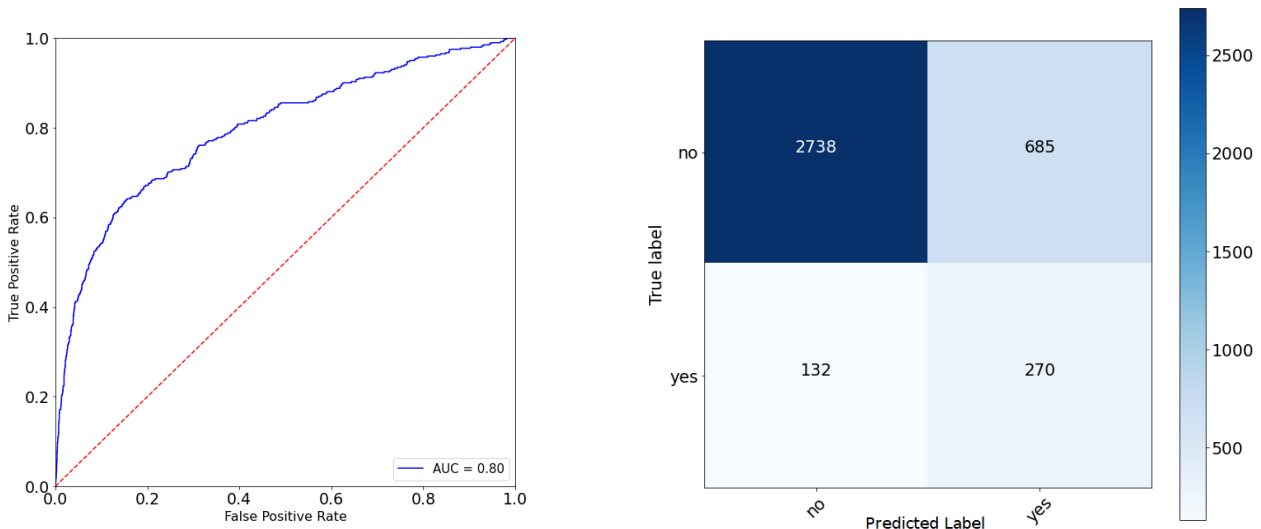

Figure 24: Roc Curve and Confusion Matrix of Logistic Regression

... so, we get an accuracy value of 0.79.

Figure 24 shows the learning curves on the training set and on the test set. We see a gradual decrease and increase of the curves in training score and test score respectively, remaining approximately constant after 25.000 training samples. The gap between the two pairs of values stabilizes around 0.3 points, so the model seems to overfill the training data.
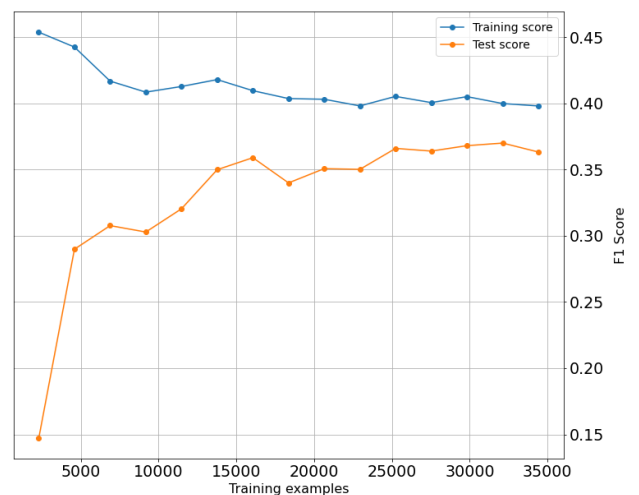

Figure 25: Learning Curve of Logistic Regression

3.2 K-NEAREST NEIGHBOR

The KNN algorithm is one of the simplest classifiers in Machine Learning. The classification is done by checking the closest K data, obviously the choice of K is crucial in evaluating the model. The big advantage is that you don't have a real training phase because the training set is itself the model. But the classification in general turns out to be rather slow compared to the other classifiers, especially if the dataset is complex and K is very large.

The only set of hyperparameters used for the tuning phase are the values of K:

- K in *[1, 4, 7, 13, … ,1876,1951,2029, … ,7204, 7351]:* the number of nearest closest neighbors. Since the training set is very large, I decided to choose a range of values that grows exponentially.

Result

The following graph describes the variation in accuracy on the validation set as a function of K:
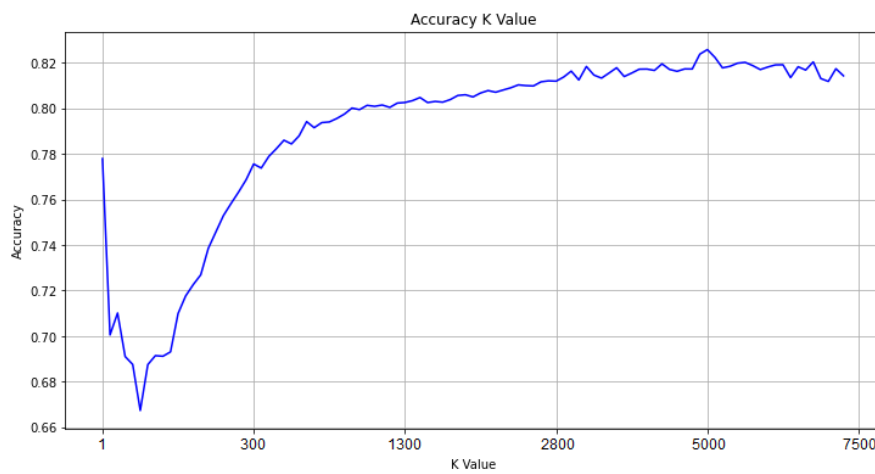

Figure 26: Tuning Phase of KNN

We observe that the model performs best for very large K values, with a maximum in 4921.

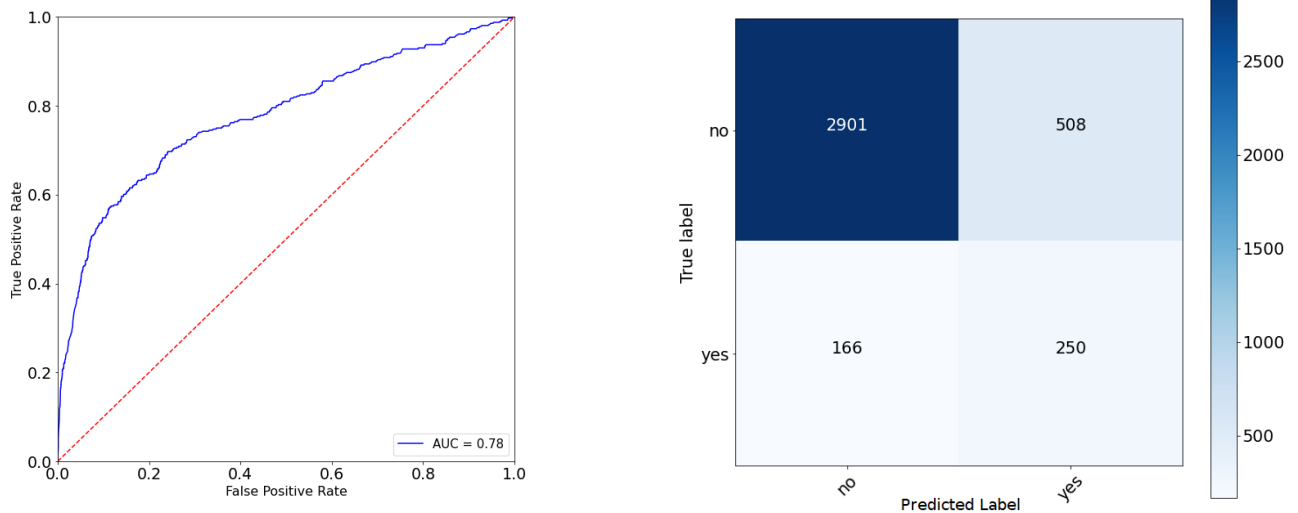So, this value has been used to evaluate the KNN on the test set:

Figure 27: Roc Curve and Confusion Matrix of KNN

The accuracy value obtained is 0.82.

The learning curves have the same trend. The training set curve records a slightly higher score value, as was expected.

It is interesting to note the vertical growth of the two curves up to a score value of 0.47 between the values of 12.000 and 18.000 of training examples, and then decreases to around a value of 0.41.

Also in this case we can observe a slight overfitting of the model when the dataset exceeds size 30.000, with the two curves that gradually increase their gap.
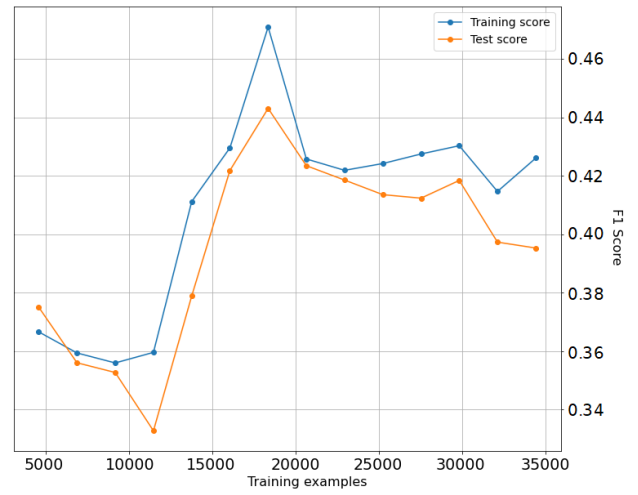

Figure 28: Learning Curve of KNN

3.3 DECISION TREE

A decision tree is a graph where the intermediate nodes form a set of IF-THEN rules that bring the data to be classified up to a leaf node, or its final classification. The rules are simply conditions on the characteristics of the data. The model training phase consists in building these rules through the training set. The rules are built at each level of the tree by doing a top-down gready search on the characteristics of the dataset using what is a measure of impurity, which essentially establishes how much a characteristic is able to better divide the dataset.

The hyperparameter sets for the tuning phase are as follows:

- Criterion in *["gini", "entropy"]:* type of impurity measurement. Given a characteristic and an associated rule, the gini index for that characteristic is:

$$G(x,r) = 1 - [P(yes|x,r)]^2 - [P(no|x,r)]^2$$

The pair (x,r) generates a subset of the dataset, which if it is balanced in the class, the Gini index will be high.
Entropy is an alternative measure:

$$H(x,r) = -P(yes|x,r) * log_2[P(yes|x,r)] - P(no|x,r) * log_2[P(no|x,r)]$$

A little more onerous computationally, due to the logarithm.

The variations of the two measures as a function of $P(yes|x,r)$ are described in Figure 28. The goal of the research is to minimize these measures and then build rules that create subsets of the dataset as unbalanced as possible.
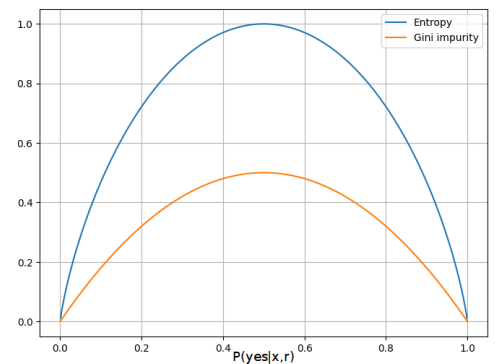

Figure 29: Gini and Entropy

- Max_depth in *[15, 20, 25, 30, 35, 40, 45, 50, 60, 65, 70, 75]:* the maximum depth of the tree.
- Min_samples_splits in *[4, 6, 8, 10, 12, 14, 16, 18, 20]:* the minimum number of samples in a node to be considered for further splitting.

17

Result

The tuning phase returned as a result "gini" as a criterion, a maximum depth of 15 and a minimum number of samples per node 6. The result on the test set is the following:
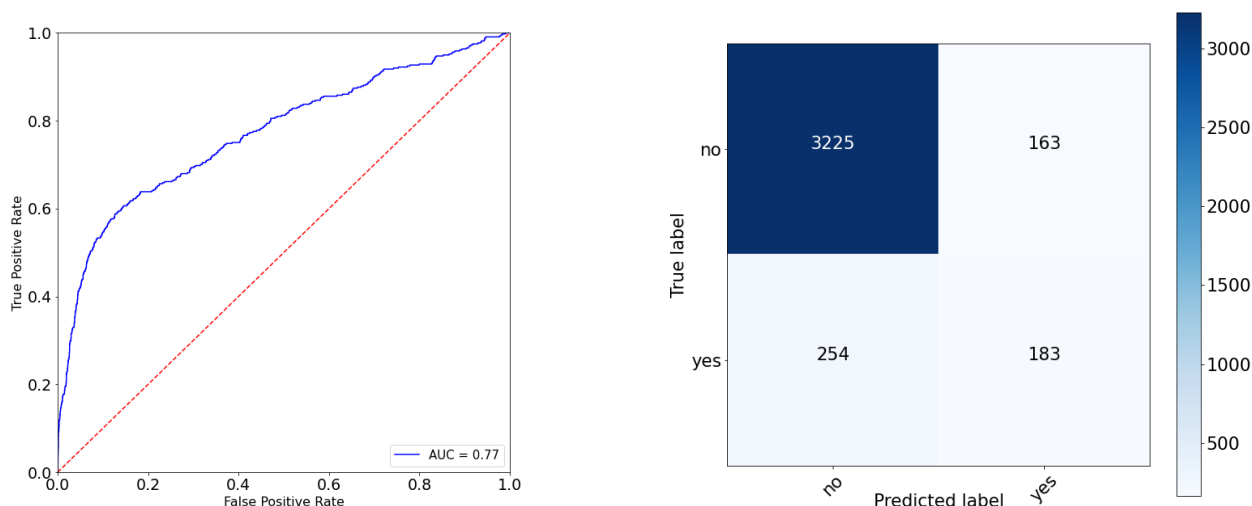

Figure 30: Roc Curve and Confusion Matrix of Decision Tree
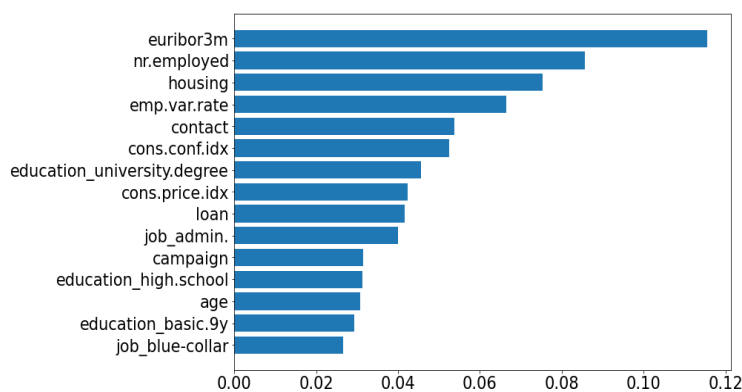
We get a good accuracy value of 0.89.

Figure 31: Importance of Features for Decision Tree

On the left we see the fifteen most important characteristics according to the decision tree, contained in the *feature_importances_* attribute of the classifier. These scores are calculated by weighing the Gini Index with the probability of crossing the node of this characteristic.

The most important characteristics are those that represent the socio-economic context, which are very correlated with each other.

The learning curves of the Decision Tree show how this model is sensitive to overfitting, despite the fact that the maximum depth is relatively small compared to the number of variables in the dataset.
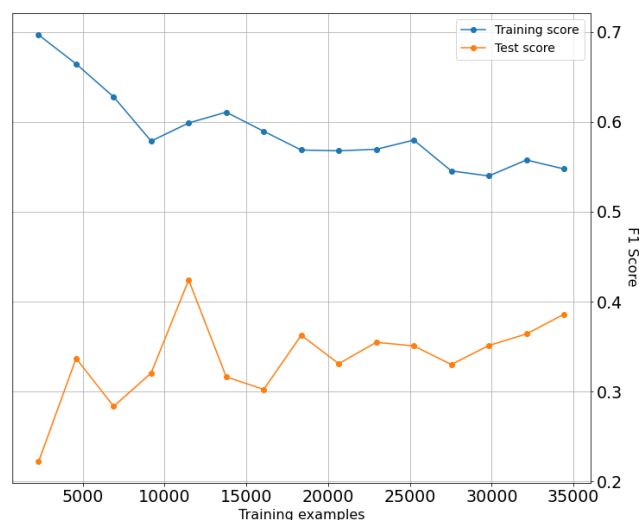

Figure 32: Learning Curve of Decision Tree

18

3.4 RANDOM FOREST

Decision trees are often inefficient and sensitive to dataset variations because of their simplicity. The Random Forest algorithm uses a set of decision trees, trained on subsets of the dataset (created by extraction with replacement), and averages them. This technique generally allows to obtain a more precise accuracy value and to reduce the overfitting of decision trees.

During the construction of the trees, each split is evaluated on a subset of features, more precisely on 6 features (the square root of the number of features) chosen randomly.

Hyperparameter sets are pretty much the same as Decision Trees with a few additions:

- N_estimators in *[50, 100, 150, … , 850, 900]:* numbers of trees.
- Criterion in *["gini", "entropy"].*
- Max_depth in *[15, 20, 25, 30, 35, 40, 45, 50, 60, 65, 70, 75]:* the maximum depth of the trees.
- Min_samples_splits in *[4, 6, 8, 10, 12, 14, 16, 18, 20]:* the minimum number of samples in a node to be considered for further splitting.

Result

The hyperparameters obtained are entropy criterion, maximum tree depth equal to 10, minimum number of splits equal to 2 and a number of trees equal to 400. These are the results on the test set:
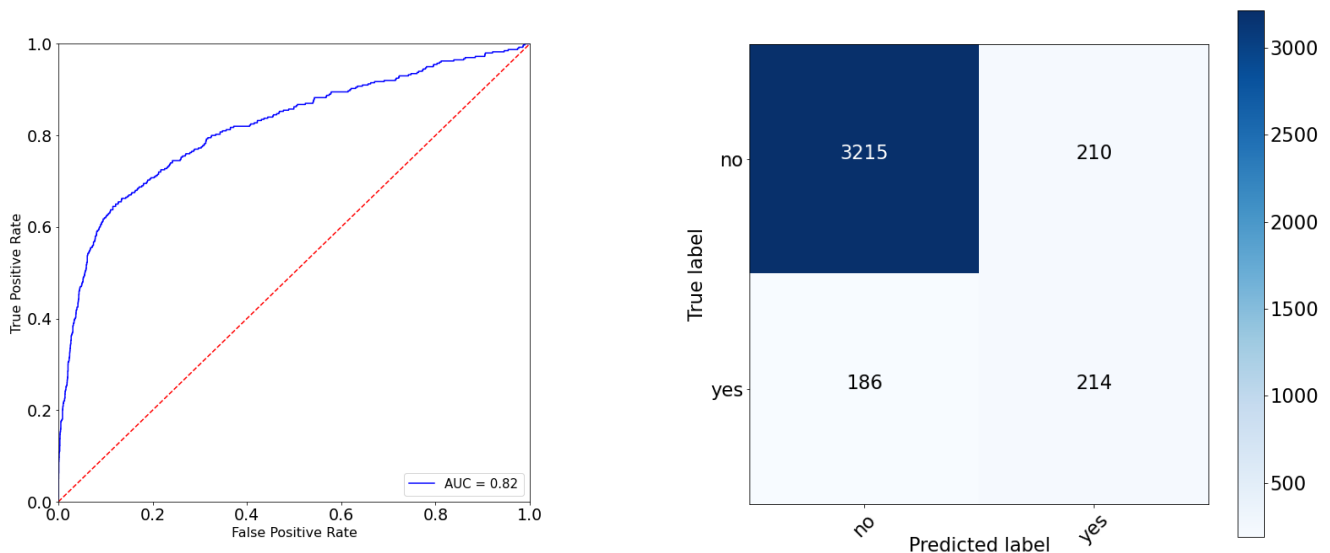

Figure 33: Roc Curve and Confusion Matrix of Random Forest
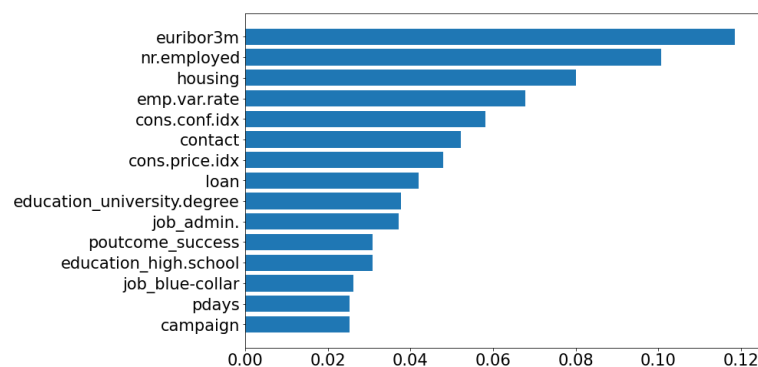
In this case, the accuracy is 0.90.

Figure 34: Importance of Features for Random Forest

Also in this case, the most important characteristics according to the model are those of the socio-economic context.

The variables *cons.conf.idx* and *cons.price.idx* are considered even more important than the Decision Tree ranking.

By plotting the learning curves, we can observe a large score gap when the number of training examples is relatively low. After 18.000, the two curves remain quite stable with a difference of about 0.05, so the model does a slight overfitting of the training data. However, the training score curve does not have a high score, probably due to a very low depth of the tree in relation to the number of variables. Compared to the Decision Tree, however, this gap is quite small.
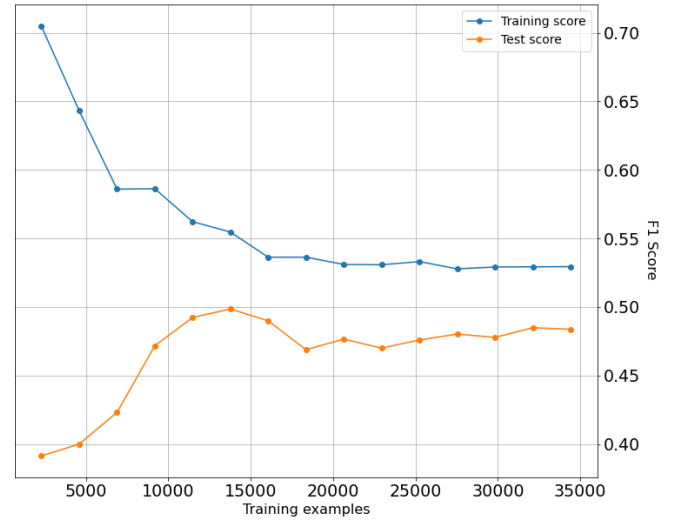

Figure 35: Learning Curve of Random Forest

3.5 SUPPORT VECTOR MACHINE

The Support Vector Machine (SVM) technique is based on the concept of hyperplane: the aim is to build a class of learning machines that build a hyperplane that separates the points of the classes in the best possible way. If the data is not linearly separable, a method called the **Kernel method** is used to map the datas into a space of higher dimension, to make it possible linear separation.

An explicit mapping of the entire dataset would be very expensive, using a Kernel function that has

$$K(x, y) = <f(x), f(y)>$$

the following form:

Basically it is a function that provides a proximity score between two oversized vectors through a function $f()$.

The Kernel function used for this analysis is the RBF:

$$K(x, y) = e^{-\gamma(||x-y||^2)}$$

The set of hyperparameters for the tuning phase are as follows:

- C in *[0.01, 0.1, 1, 10, 100, 1000]*: inverse of regularization strength, smaller values specify stronger regularization.
- Gamma in *[0.0001, 0.001, 0.01, 0.1]*: gamma value in Kernel function.

20

Result

The hyperparameters obtained are entropy C equal to 0.1 and Gamma equal to 0.001.

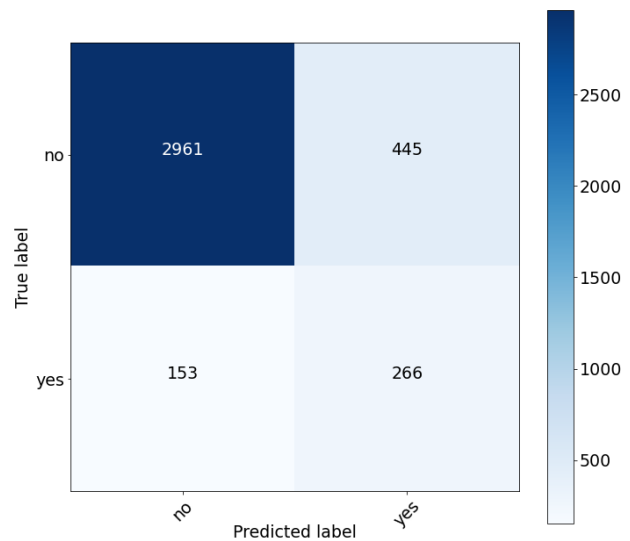Now let's see the results on the test set:
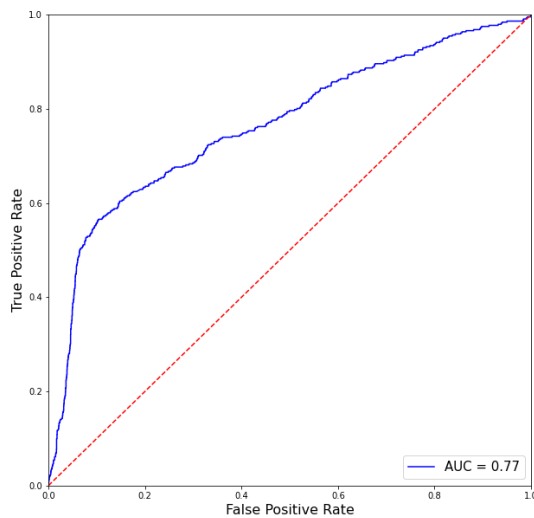
Figure 36: Roc Curve and Confusion Matrix of SVM

The accuracy value is 0.84.

Both learning curves have practically the same trend. Only when the training set is smaller than 10.000, is there a bad classification of the model on the test set. When the dimension is equal to 13.500 the model returned the same score value, both in the test set and in the training set.

The training score curve is generally increasing, with values always slightly higher than the test score curve. So also in this case we record an overfitting of the training data.

Figure 37: Learning Curve of SVM

COMPARISON & CONCLUSIONS
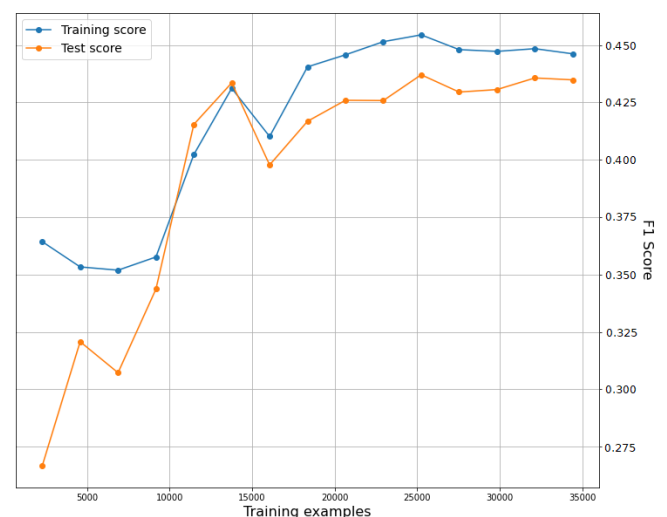
In this final part, I will summarize the results of the models. The following table shows the *precision*, *recall*, and *f1-score* metrics for each of the two classes. I remember that *precision* is the ratio between the number of correct classifications and the total of classifications, *recall* is the ratio between the number of correct classifications and the total of data belonging to that class, while the *f1-score* is a metric that contains both *precision* and *recall*.

The table also shows the accuracy and the AUC value:

	Logistic Regression	KNN	Decision Tree	Random Forest	SVM
Precision "no"	0,95	0,95	0,93	0,95	0,95
Precision "yes"	0,28	0,33	0,53	0,50	0,37
Recall "no"	0,80	0,85	0,95	0,94	0,87
Recall "yes"	0,67	0,60	0,42	0,54	0,63
F1-score "no"	0,87	0,90	0,94	0,94	0,91
F1-score "yes"	0,40	0,43	0,47	0,52	0,47
Accuracy	0,79	0,82	0,89	0,90	0,84
AUC	0,80	0,78	0,77	0,82	0,77

Figure 38: Comparison

We immediately observe that the performance values of all models are much higher in dealing with negative data than positive ones. Unfortunately, this is due to the strong imbalance of the dataset.

We can say with great probability that Random Forest is the algorithm that best manages the dataset. This model provided the highest Accuracy, AUC and F1-Score values. Decision Tree also has quite similar performance.

We note how, in this test, the Logistic Regression recorded a significantly higher recall value than the others, with a rather low precision value. So, it seems that it tends to reduce the number of false negatives.

However, in general none of the algorithms used can be exploited with too much reliability. In fact, we see that the F1-Score values are rather low.

Conclusions and recommendations

In summary, this work had as its objective the construction of a classification model that would allow the bank to subscribe as many term deposits as possible with the fewest possible phone calls. The dataset is a mixture of numeric and categorical variables, which I chose to encode with the One Hot Encoding technique. Furthermore, given the strong imbalance of the dataset, a preliminary test suggested us to apply the SMOTE algorithm to obtain more sincere accuracy values.

The Random Forest is the approach we suggest to the bank, which also suggests (as in the preliminary analysis of the dataset) paying attention to socio-economic indices and therefore making phone calls based on these parameters. So, one strategy can be to increase the number of phone calls when these rates are low.

FINAL REMARKS

In conclusion, I wanted to report a couple of observations made during the preprocessing of the dataset and the evaluation of the algorithms:

- As I said on page 14, the SMOTE algorithm in the preliminary tests has improved the f1-Score and the recall of all the models, this however greatly penalizing precision. The precision value using the unbalanced dataset in fact increases by about 5 points for Random Forest and Decision Tree, 10 points for SVM and by about 30/40 point for Logistic Regression and KNN. So, if you prefer to reduce the costs and times of phone calls and have more chances of subscribing (even if you have fewer customers) then this could be an alternative way.

- The "*duration*" variable has been removed from the analysis since it is a data not known before the phone call. This choice was also suggested by the notes present in the dataset itself. While carrying out some tests, I noticed how much this variable is extremely influential for the purposes of classification. In fact, keeping this variable in the dataset allowed all models to increase all metrics by 10/15 points and obtain an average AUC value of 0.96. Both Decision Tree and Random Forest have considered this variable as the most important.
An alternative analysis could perhaps also have taken this variable into consideration and suggested that the bank study the structure of the phone calls, to regulate their duration in an optimal way.