Final Assignment Supervised Learning: Classification

Main objective

Understanding when users are about to **churn from a company** is useful to take actions to try to retain them. **Credit card** service is an important part of the bank business, that can be more profitable when customers are more loyal. In this framework, a bank has asked for our services in order to help them to predict when their customers are about to dispense without the company services. So, **the main goal in this project will be to create a model that can predict the customer churn** as better as possible.

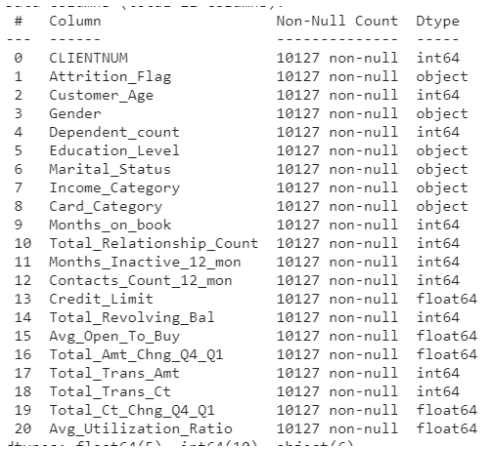
Although, if it is not our main objective to figure out which reasons could be the ones that lead to churn, it could be also possible to gather this information during this process.

The **business** will **benefit** from this study by **anticipating** when a customer is going to leave the service. Thus, the bank could design some actions to keep the customer inside. Also, this process will give us some insights about the reasons so this would help to design the actions to be taken.

Brief description of the data

The bank provides us with a file containing some personal information about the customers and about their account activity and also information if finally, a customer churned or not.

The data set **consists** of **10127 rows** and **21 columns**. Also, from the 21 columns we know that **15 are numeric columns** and the **other 6 are categorical** ones. The column "**CLIENTNUM**" represent the customer **id**, so after checking the **unique values** for this column we can say that **each row represent a different customer**. Furthermore, the column "**Attrition_Flag**" contain information about **churn**. So, it will be our **target variable** with the 84% not churn and the 16% churn, so we have an **unbalanced data set** which could introduce some errors on our model when predicting churn. Continuing with the description of our data we can also say that for each column there are 10127 **non-null values** so we will not need to deal with missing values later. In the following image (**figure 1**) we show the columns with their non-null values and types:



#	Column	Non-Null Count	Dtype
0	CLIENTNUM	10127 non-null	int64
1	Attrition_Flag	10127 non-null	object
2	Customer_Age	10127 non-null	int64
3	Gender	10127 non-null	object
4	Dependent_count	10127 non-null	int64
5	Education_Level	10127 non-null	object
6	Marital_Status	10127 non-null	object
7	Income_Category	10127 non-null	object
8	Card_Category	10127 non-null	object
9	Months_on_book	10127 non-null	int64
10	Total_Relationship_Count	10127 non-null	int64
11	Months_Inactive_12_mon	10127 non-null	int64
12	Contacts_Count_12_mon	10127 non-null	int64
13	Credit_Limit	10127 non-null	float64
14	Total_Revolving_Bal	10127 non-null	int64
15	Avg_Open_To_Buy	10127 non-null	float64
16	Total_Amt_Chng_Q4_Q1	10127 non-null	float64
17	Total_Trans_Amt	10127 non-null	int64
18	Total_Trans_Ct	10127 non-null	int64
19	Total_Ct_Chng_Q4_Q1	10127 non-null	float64
20	Avg_Utilization_Ratio	10127 non-null	float64

Figure 1. Initial columns, number of non-null values and variable type.

After having a look on the main statistics of the data set, we can say that features have different ranges and some of them seem to be skewed, so actions on the data set should be taken to correct the data set before training the models.

Summary of EDA. Cleaning and feature engineering

The **first** action will be to **drop the column "CLIENTNUM"** since, as we could expect, the different values are equal to the total number of rows so **no useful information** to the model is contained on it.

Having a look into the **numeric columns** we would like to have a look into the feature distributions, **studying how skewed they are to understand if there are any feature that need a logarithmic transformation or not**. Furthermore, we are interested to **find out if there are any correlation between the features to eliminate any possible duplicity** that can lead to wrong predictions. Also, we should try to **identify outliers** that could be a problem for our model.

Regarding to the **categorical features** we need to **encode** them in order to be used by the models. So, we will apply a proper transformation for each one.

Since we will train some different models and some of them could be sensible to the scale of the features, we **standardize** our data.

During the Exploratory data analysis, **we find**:

1) Having a look into the **categorical** features we found that only **Attrition_Flag** and **Gender** are **binary**. So, we encode both to the following values: for Attrition_Flag **not churn** will be **0** and **churn** will be **1**, and for Gender **M** will be **0** and **F 1**. The other categorical features, **Education_Level, Marital_Status, Income_Category and Card_Category** were encoding **as ordinals**. Also, **Education_Level** was **grouped into 4 categories** meaning the educational levels and the **Income_Category into 6 categories**. Having the following values:

- Education_Level: unknown **0**, (Uneducated) **1**, (Graduate, High School) **2**, (College, Post-Graduate, Doctorate) **3**

- Marital_Status: Unknown **0**, Single **1**, Married **2**, Divorced **3**

- Income_Category: unknown **0**, less than 40k$ **1**, 40k-60k$ **2**, 60k-80k$ **3**, 80k-120k$ **4** and +120k$ **5**

- Card_Category: blue **0**, silver **1**, gold **2**, platinum **3**

2) Regarding the **numeric** features. First, we study which of them could be **ordinals** ones. We found that **Customer_Age** and **Months_on_book** should be ordinals. With the following values:

- Customer_Age:0-35 **0**, 35-45 **1**, 45-55 **2**, 55-65 **3**, 65-99 **4**

- Months_on_book: 0-20 **0**, 20-30 **1**, 30-40 **2**, 40-or more **3**

3) We found some **skewed** features, see **figure 2**, and we **apply a logarithmic transformation** on them. These features are: **Credit_Limit, Avg_Open_To_Buy, Total_Trans_Ct, Total_Trans_Amt, Total_Ct_Chng_Q4_Q1 and Total_Amt_Chng_Q4_Q1**.

4) Then we study the **correlation**, see **figure 3**. For this data set we found some highly correlated features. To eliminate duplicity and avoid errors on our model we **drop** the following features: **Avg_Open_To_Buy, Avg_Utilization_Ratio, Total_Trans_Ct, Total_Ct_Chng_Q4_Q1** and **Months_on_book_group**. The pairs were:

- Customer_age_group-Months_on_book_group

- Total_Revolving_Bal-Avg_Open_To_Buy

- Total_Revolving_Bal-Avg_Utilization_Ratio

- Avg_Open_To_Buy-Avg_Utilization_Ratio

- Total_Trans_Amt-Total_Trans_Ct

- Total_Amt_Chng_Q4_Q1-Total_Ct_Chng_Q4_Q1

5) Regarding the **outliers**, we found some on the next features: **Customer_age_group, Months_Inactive_12_mon, Contacts_Count_12_mon,** see **figure 4**. So, after **dropping the outliers**, our data sets consist of **9179 rows.** After removing these rows, we still have the **same proportion between classes**.

6) Then, we **scale** our data using **MinMax** scikit-learn method.

7) And finally, we perform a **stratified split of our data set into a train and test set**. Like this, we **keep the original proportion of the classes in the target value in the train and in the test data sets**.

We present also in **figure 5** a visual summary of **the numeric feature distributions and how are related between them after cleaning the data.** The colour code indicates how are the data distributed in **relation with the target variable**, being the colour **blue for not churn** and the **orange for churn**.

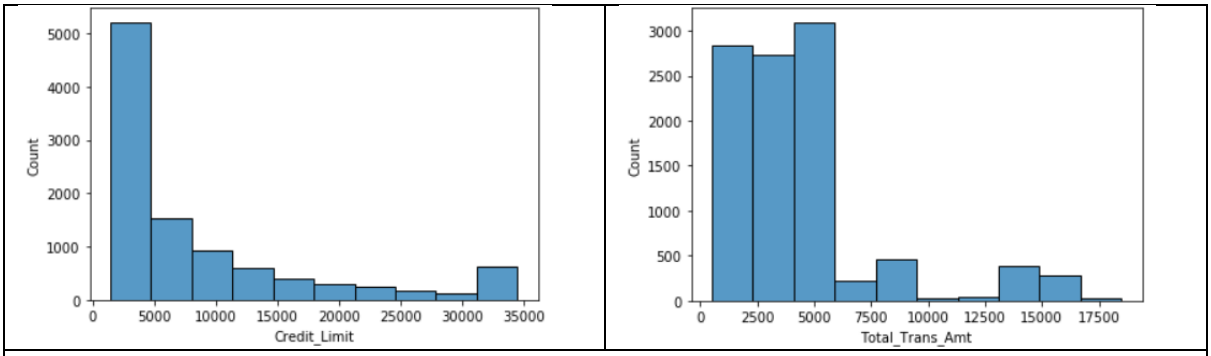


Figure 2. Histograms for columns Credit limit and total transaction amount shows the characteristic tails for skewed features.

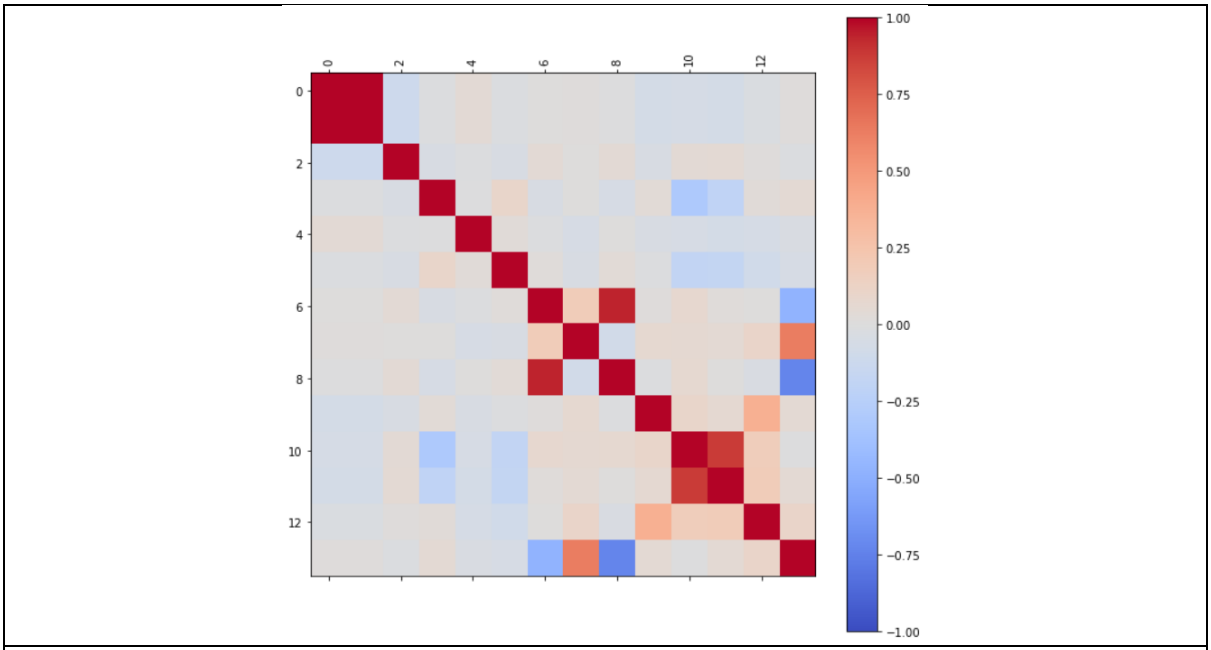


Figure 3. Correlation matrix between numeric features.

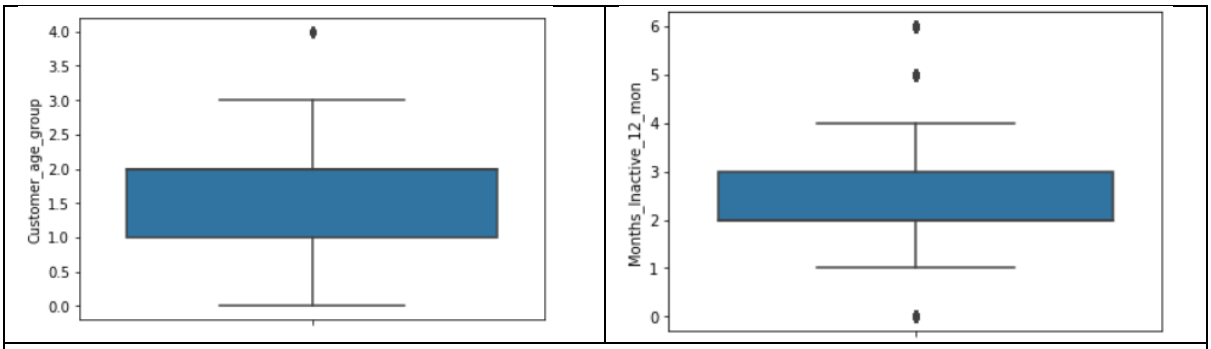


Figure 4. Boxplot for columns Customer Age and Months Inactive showing the outliers.

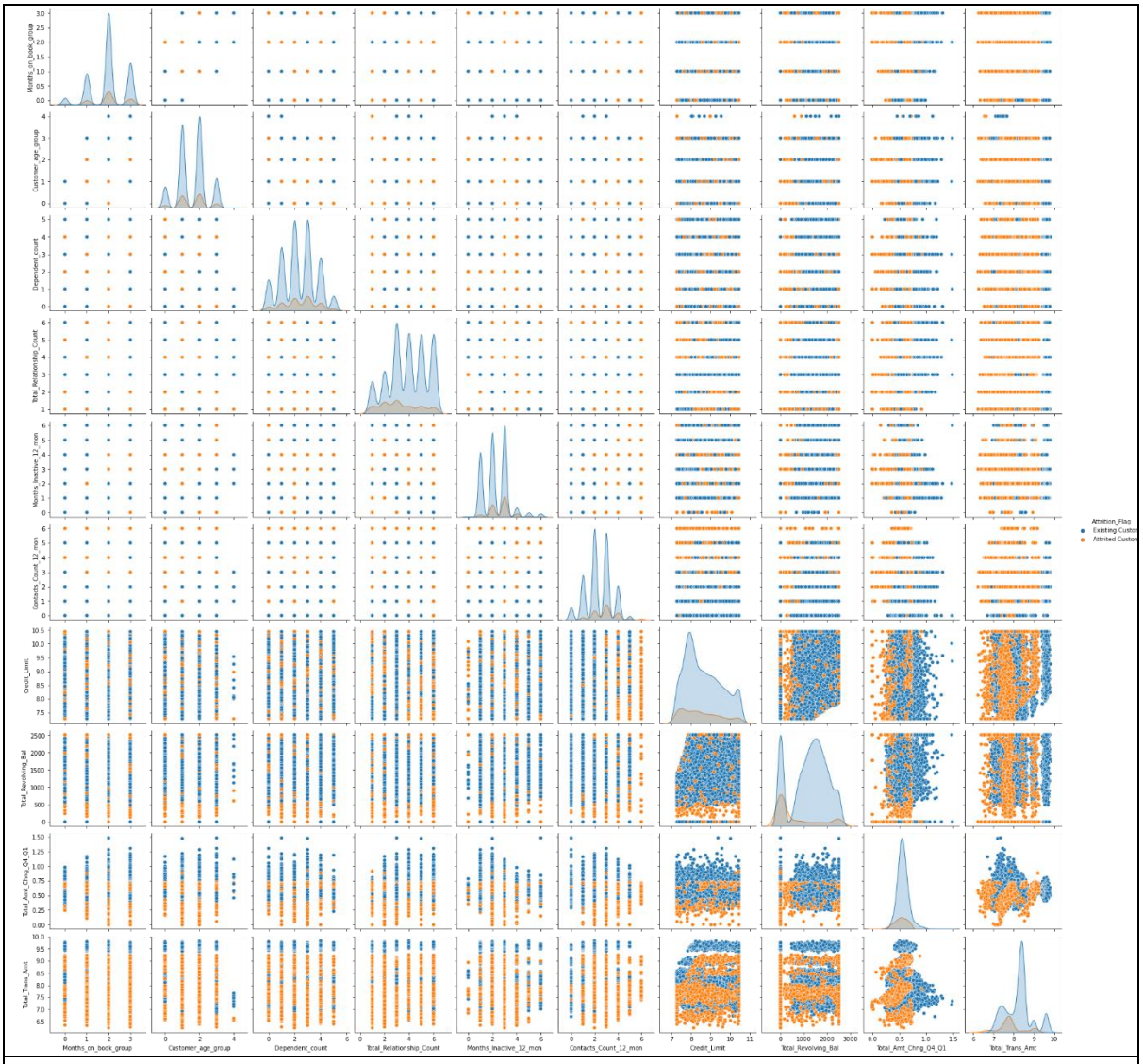


Figure 5. Pairplot after cleaning the data with the numeric features. Orange is the distribution for the rows “churn” and blue for the “not churn”.

Summary of training at least three different classifier models

Our main objective is to **predict when a customer is going to churn or not**. So, we need to resolve a **binary classification problem**. For this purpose, we first **study** which classification **algorithm** could give us the **best** result on this **prediction**. First, we will train a **logistic regression** model, which is going to be our **baseline**. Then, we will train other two different model, a **KNN model and a Decision Tree model**.

Since the bank is basically interested on predicting when a customer is about to churn, we will use the **confusion matrix to compare between the different classification models**. Thus, we will **visualize easier which of them has a greater True Negative value**.

Also, we would like to **show the effect of working with unbalanced data.** We will use the logistic regression model to show this difference.

All the models were trained and tested with the **same data set**.

Logistic Regression

As we said before we want to show the effect of working with unbalance data. After cleaning our data set the proportion between the two target classes where 84% not churn and 16% churn. To keep the same ration for train and test sets we uses the method StratifiedShuffleSplit from the scikit-learn library. That means that we have **6439 rows for the train set** and **2740 for the test one**, being in the former only **434 values for the minority class**.

If we keep the **data set unbalance the model would has** a very low accuracy for the minority class.

For the logistic regression model in this case as we can see in **figure 6**, we only predict 134 of the minority class **correctly (the 31%)**.

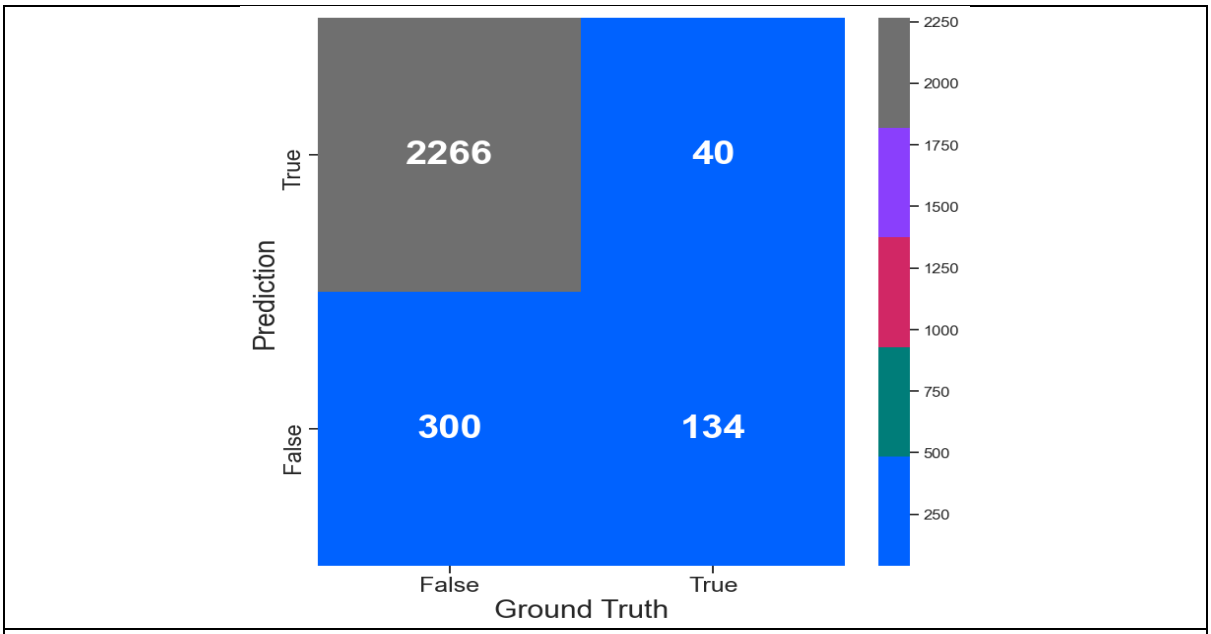


Figure 6. Confusion matrix for logistic regression trained with unbalanced data.

To balance our data, we perform a **random oversampling of the minority class** as not to lose data. The method used is the RandomOverSampler from imblearn library. We perform the oversampling **on the train set**. So, after training the logistic regression model with the **balanced data** and predicting with the test set, we have **311 TN** values, **which** represents the **71.6%** as we can see in **figure 7**.

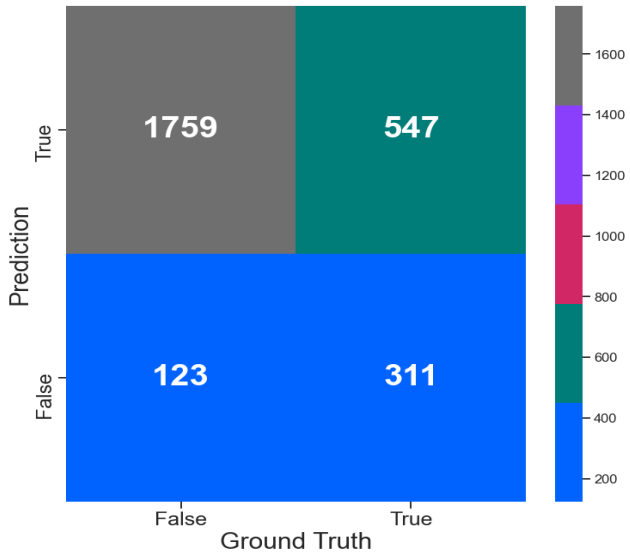


Figure 7. Confusion matrix for logistic regression model trained with the balanced data.

KNN model

From now on, we continue the study with the balanced data. After training our baseline model, the logistic regression, we train with the same data set a **K-Nearest Neighbors** model. In this case, **first** we need to set the proper **k-value**. We compare the different KNN models using the k versus the Error ratio. The k where this curve reaches the plateau is the best one. As we can see in **figure 8**, the selected **k value will be 11**.

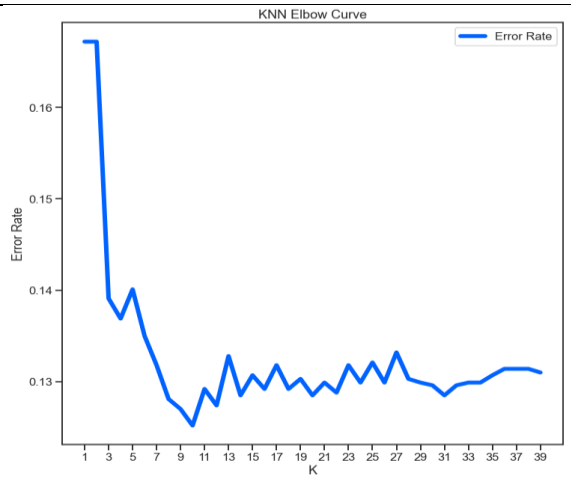


Figure 8. Error rate vs K, for a KNN model.

For this K, the KNN model has **313 TN**, which represents the 72% as we can see in **figure 9**.

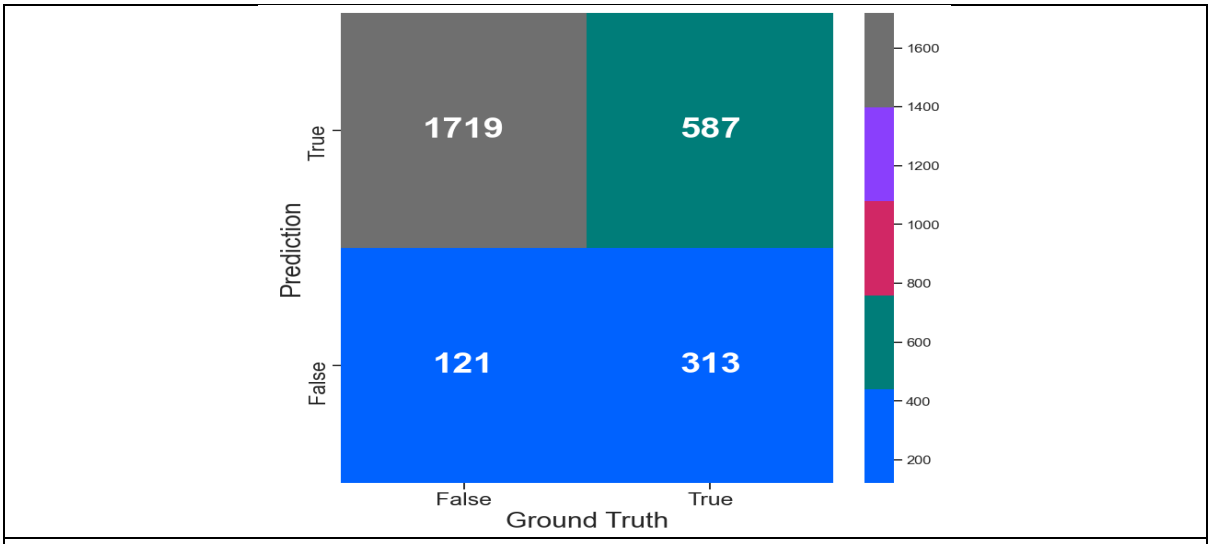


Figure 9. Confusion matrix for the KNN model.

Decision Tree

We continue with a Decision Tree model. In this case, we have two hyperparameters, the **max depth** and the **max features**. So, to find the best possible model we will play with these parameters and compare them using **cross-validation**. After this process, we found that the **max depth is 17** and the **max features is 11**. For these values we found that **323 TN** was found, representing the **75%**, as we can see in **figure 10**.

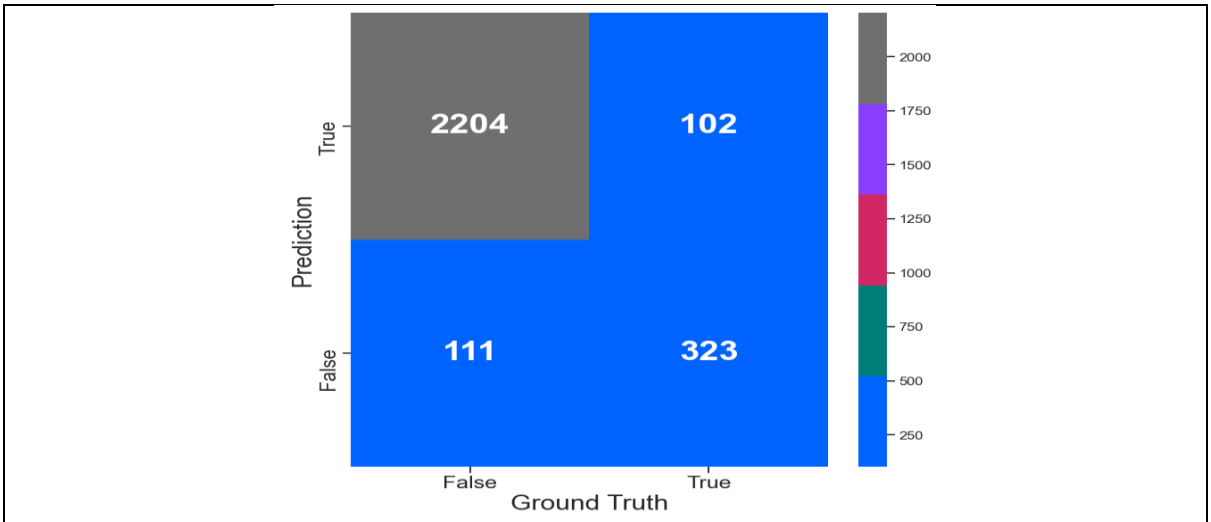


Figure 10. Confusion matrix for the Decision Tree model.

Our recommendation to solve this problem is using the **Decision Tree** which have the **greatest success in predicting the minority class**, the one we are interested on, for all the trained models. Also, for the Decision Tree we have the largest ration for the TP, the majority class. The Decision Tree model, if we need it, can also give us insights about the most important features by inspecting the nodes of the Tree.

Summary key findings

During the logistic regression model, we found out that working with unbalanced data are terrible wrong. The predictions were overfitted to the majority class having a bad behaviour for the minority ones, which was the objective of our study. So, to **balance the data is a critical step**.

Since, the minority class is the one that the business is interested we observe how was the prediction for this class over all the model. We found out that the **Decision Tree was the one performing better** with a 75% of good predictions. Also, for the decision tree model the majority class has a very good performance of almost 97% of good predictions. The result for the minority class could **not** be **satisfactory enough**, so further investigation should be done.

A second insight, although we are not in principle interesting on, are the importance of the features. We can derive this information also from the decision tree, but in this case once we have work with the logistic regression model by having a look on the coefficients for the model, we can advance that the most important features are Total_Revolving_Bal, Total_Amt_Chng_Q4_Q1,'Total_Trans_Amt and Months_Inactive_12_mon.

Next Steps

Three major actions can be taken in order to achieve better results. **First**, we could perform some **feature engineering by applying some polynomial features**. These new features could gain importance on the prediction so it is worthy to have a look on it.

A **second** action, and more critical, is to **study further models**. Since the Decision Tree seems to be in a good way, we should try some variations on it performing actions that could **increase the randomness and so lower the variance and thus improving the result**. We could try **Random Forest** model or other ensemble model such as **Adaptive Boosting** which is more robust in overfitting.

Finally, we could expend more time trying to find a **better balance** of the data by trying different sampling methods such us SMOTE or a combination of oversampling and udersampling like the **combination** of **SMOTE and Tomek's Link**.