Churn Classifier

Team members:

- Jonathan Derks
- Francesco Sala
- Cristina Salvatori
- Riccardo Vicinanza

Introduction

This study aims at finding a good predictive model of customer attrition from the StayWithMe (SWM) Bank services. The task is the correct classification of current and future customers as attriting customers and thus it constitutes a supervised, labeled, batch and model-based machine learning task. The study gathers meaningful insights on the main predictors of customer loss available in the churn dataset provided by the StayWithMe Bank.

Methods

2.1 Features

In order to explore the dataset features we ran some preliminary analysis including descriptive statistics, plots (barplots, histograms, kde curves), outlier detection, chi-squared analysis, ANOVA test and correlation analysis with Pearson R. We also tried improving our results by manipulating the dataset variables and computing one new feature called Amount per Transaction. The outcomes of this analysis are discussed in the first section of our results.

2.1.1 Dataset Structure

After loading the dataset in our Jupyter workspace we used the .head() and .info() methods to get a first quick grasp of the dataset composition.

2.1.2 Categorical Variables: value_counts, percentage count plots, cross tables

The value_counts() method was employed to find out which categories exist for each categorical feature and the count per category. We also employed count plots from seaborn to better visualize the percentages of data points in each category for each variable.
We computed the cross tables with the pandas .crosstab() method between each categorical variable and the target Attrition Flag. We also calculated the ratio between Attrited and Existing customers for each category to inspect any unbalanced distribution of the two flags across categories belonging to the same features.

2.1.3 Numerical Variables: histograms, kde, cross tables

By calling the hist() method from matplotlib on the whole dataset we plotted a histogram for each numerical attribute and gained a quick view of the type of data we were dealing with. We decided to replot a few histograms to further explore a few outliers and graphical artifacts that caught our attention.
In order to visualize how Attrited and Existing customers distribute for each attribute, we overplotted two kde curves for each numerical variable (one for Attrited and the other for Existing customers).

2.1.4 Test for Dependency

Chi_Squared Test of Independence
In order to build a good predictive model and decide/determine which variables to include/should be included in the model, we tested the degree of dependence among individual variables with our target variable “Attrition Flag”.

We used Pearson's Chi-Squared Test and Likelihood Chi-Squared Test to test for the dependence between “Attrition_Flag” and the others categorical variables in the dataset. In these tests, the null hypothesis holds that the variables are independent of each other, and the alternative hypothesis states that the variables are dependent on each other. We used a function in the python package "bioinfokit.analys" to perform these tests. The output of the function provides Degrees of Freedom and p-values:

We can reject the null hypothesis when the Pvalue is less than 0.05 and say that there is a statistically significant association between the variables considered. If the p-values exceeds 0.05, we accept the null hypothesis that the variables are independent

In order to perform the Chi_Squared Test of Independence we used cross tables between our target variable “Attrition_Flag ” and each of that categorical variables ("Gender", "Education_Level", "Marital_Status", "Card_Category" and "Income_Category")

Anova Test and Boxplots
We performed the Anova test to analyze the relationship between the categorical target variable and the individual quantitative variables of the dataset. The ANOVA inspects whether the mean of the quantitative character varies as the qualitative characteristics change. We used a function from the python package "statsmodel” that runs a F test. If the test has a significant p-value, it means that at least one of the averages is different from the others. As the quantitative and qualitative characteristics of a variable vary, the two variables can be considered strongly related. As a way to visualize and better understand the relationship between “Attrition_Flag” and quantitative variables, we also plotted boxplots.

Correlation Analysis: Pearson R
To investigate the presence of possible correlations between all the attributes in the dataset we plotted a heatmap from seaborn of the correlation matrix computed on both on the *train set* and on the full set. Because of the nature of categorical variables we had to factorize them before being able to plot the matrix. We were particularly interested in spotting any meaningful correlation between the predictive features and the target variable. Hence we isolated and sorted the correlation coefficients between the Attrition_Flag all the features in the dataset.

2.1.5 Features Selection
Combining the results obtained from the statistical analysis we selected from the original dataset a few variables that we considered more significant to build a good predictive model. We decided to assemble three different dataset:

- a OR_DATA dataset containing all the variables in the original dataset (including the “Unknown” rows).
- a DATA_FULL dataset containing all the variables presented in the original dataset (not including the “Unknown” rows).

- a DATA_BEST dataset of intermediate size containing only ten variables that proved significant across the multiple analyses conducted (kde, correlation, chi-squared, etc.): "Attrition_Flag", "Gender", "Income_Category", "Total_Relationship_Count", "Months_Inactive_12_mon", "Contacts_Count_12_mon", "Credit_Limit", "Total_Trans_Amt", "Total_Trans_Ct", "Avg_Utilization_Ratio".
- a DATA_MINI dataset including only Total_Trans_Ct and Total_Trans_Amt.

After assembling the datasets we also removed the “Unknown” data points for each categorical variable retained in the dataset (except for OR_DATA which preserves these rows). We performed this operation after creating the datasets in order to keep as many data points as possible. For example in the data_best dataset the only categorical variable presenting “Unknown” values is Income_Category thus we only removed the 1112 rows containing unknown values for Income_Category. If we computed this operation on the full dataset we would have also lost all the datapoints containing “Unknown” for the Marital_Status and Education_Level, whether or not we would have actually retained those two variables in the final dataset.

2.2 Algorithms, training overview, design choices

2.2.1 Train-Test Splits

On the three datasets obtained we implemented two different train-test split methods:

- The traditional train-test split provided by scikit-learn model selection module.
- A Stratified train-test split based on the Attrition_Flag that aims at retaining the proportion of Attrited and Existing customers of the full dataset also in the train and test sets. We checked that the proportions between Attrited and Existing customers were maintained in the stratified train and test sets.

As we already mentioned before we repeated both the plain and the stratified split for three different test_sizes: 20%, 25%, 30%.

*Due to time constraints we were not able to implement Synthetic Minority Oversampling Technique(SMOTE) to handle the imbalanced dataset, synthesize new data points for minority class and oversample the Attrition_Flag. We succeeded in running the SMOTE on the dataset but we were unable to feed it properly to our machine learning algorithms.

2.2.1 Data Preparation for ML Algorithms

For both the traditional split and the stratified split we ran the following operations in order to prepare the data for Machine Learning Algorithms:

- We separated the train_set predictors from the train set target value (labels)
- We split the train set of predictors in numerical and categorical predictors
- We ran a Transformation Pipeline which Scaled the numerical attributes with Standard Scaling and encoded the Categorical columns with OneHot Encoding. For both operations we imported the methods from scikit-learn model selection module.

2.2.3 Machine Learning Models

According to the nature of the data and of the task we selected four main machine learning models to feed the data with:

- Logistic Regression

- Support Vector Machine (SVM) - capable of performing linear or nonlinear classification and particularly well suited for classification of complex small or medium-sized datasets
- Decision Tree for classification
- Random Forest for classification

2.2.4 Metrics: Accuracy, Confusion Matrix, Cross validation

We fed the four ML models with the full train dataset, the “data_best" train set and the minimalistic train dataset. We compared the four indicators of accuracy, sensitivity, precision and specificity computed from the confusion matrix. As aforementioned we repeated this training session with three different "test sizes": 0.2, 0.25. 0.3. We also obtained three cross validation scores for each algorithm and each test size. The goal of cross validation was to gather the additional information of mean and standard deviation of the accuracy scores even though in classification tasks the confusion matrix proves more reliable than accuracy in measuring performance. To perform such operations we imported the confusion_matrix, precision_score, recall_score, f1_score, cross_val_score methods from the scikit learn module metrics.

2.2.5 Precision-Recall curve, ROC curve

We also plotted the Precision-Recall curve and the ROC curve. Our goal was to visualize how well our final Random Forest classifier would manage the trade-off between Precision and Recall. The receiver operating characteristic (ROC) curve is a common tool used with binary classifiers. It is very similar to the precision/recall curve but instead of plotting precision versus recall, it plots the true positive rate (recall) against the false positive rate (FPR). We imported from sklearn.metrics both the precision_recall_curve, roc_curve methods.

2.2.6 Model Fine Tuning

In order to fine tune our model’s hyperparameters and their combinations we employed the RandomizedSearchCV from the model selection module of scikit-learn. We also used the best_estimator_ in output to refit the model and check the performance metrics.

2.2.7 Decision Boundary plots

To further visualize our classifiers performance we decided to plot decision boundaries for the two variables of Tot_Trans_Count over the Tot_Trans_Amt for both the training and the test set. We repeated this procedure for both the Decision Tree and the Random Forest classifier. Our goal was also to spot any overfitting trend of the models. We employed the ListedColorMap method from matplotlib.colors.

2.3 Environment

To install all packages and their dependencies we ran the following commands via terminal bash. We used virtualenv to create and manage the environment in the project directory

Create a project dir
$ export ML_PATH="$HOME/ml_project" && mkdir -p $ML_PATH

Make sure the latest pip is installed and install virtualenv
$ python3 -m pip install --user -U pip

$ python3 -m pip install --user -U virtualenv

Navigate to project directory, create the environment and activate it
$ cd $ML_PATH
$ virtualenv ml_env
$ source ml_env/bin/activate

Install all packages and their dependencies
$ python3 -m pip install -U jupyter matplotlib seaborn numpy pandas scipy scikit-learn bioinfokit statsmodels imblearn

Add virtualenv as jupyter kernel and open jupyter
$ python3 -m ipykernel install --user --name=python3
$ jupyter notebook

Interrupt connection with Jupyter Server and deactivate the environment
$ ctrl + c # y
$ deactivate

Code description

Please find the code description attached in the two PDF files:

- data_analysis.pdf
- ML_algorithms.pdf

Results

Preliminary analysis
The results of the exploratory analysis on the features helped us select the most promising categorical and numerical features in the dataset.

Dataset Structure
The dataset includes 17 attributes and 10,127 instances:

- The target variable is the “Attrition_Flag” which is binary and categorizes customers as Existing or Attriting.
- Eleven attributes are numerical (9 of type: int64 and 2 of type: float64): "Customer_Age", "Dependent_count", "Months_on_book", "Total_Relationship_Count", "Months_Inactive_12_mon", "Contacts_Count_12_mon", "Credit_Limit"(float), "Total_Trans_Amt", "Total_Trans_Ct", "Avg_Utilization_Ratio"(float).
- Six attributes are of type object: 'Attrition_Flag', 'Gender', 'Education_Level', 'Marital_Status', 'Income_Category', 'Card_Category' . They could hold any kind of Python object but since we loaded this data from a CSV file, we know that they must be text attributes. Inspection of the top rows revealed that the values were repetitive thus confirming that the attributes were categorical.

```
In [9]: # Quick vars description
        churn.info()
```

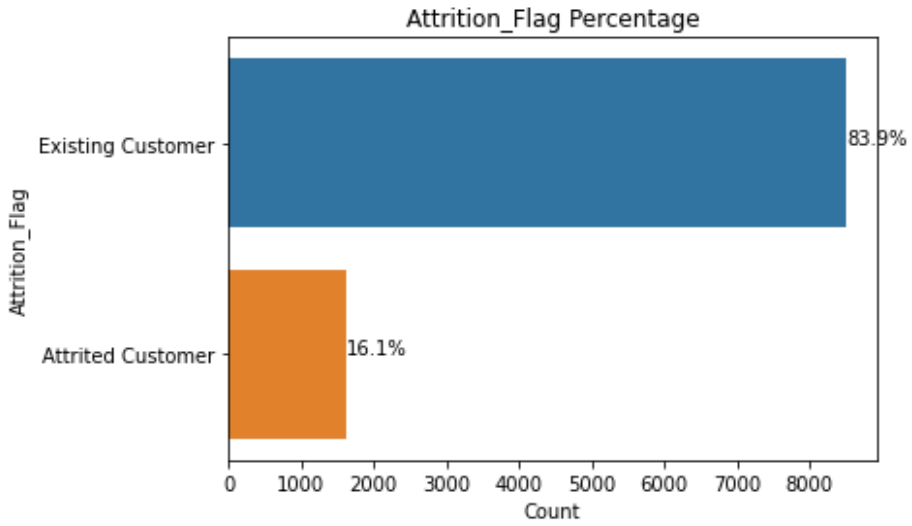
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10127 entries, 0 to 10126
Data columns (total 18 columns):
 #   Column                    Non-Null Count  Dtype
---  ------                    --------------  -----
 0   Unnamed: 0                10127 non-null  int64
 1   CLIENTNUM                 10127 non-null  int64
 2   Attrition_Flag            10127 non-null  object
 3   Customer_Age              10127 non-null  int64
 4   Gender                    10127 non-null  object
 5   Dependent_count           10127 non-null  int64
 6   Education_Level           10127 non-null  object
 7   Marital_Status            10127 non-null  object
 8   Income_Category           10127 non-null  object
 9   Card_Category             10127 non-null  object
 10  Months_on_book            10127 non-null  int64
 11  Total_Relationship_Count  10127 non-null  int64
 12  Months_Inactive_12_mon    10127 non-null  int64
 13  Contacts_Count_12_mon     10127 non-null  int64
 14  Credit_Limit              10127 non-null  float64
 15  Total_Trans_Amt           10127 non-null  int64
 16  Total_Trans_Ct            10127 non-null  int64
 17  Avg_Utilization_Ratio     10127 non-null  float64
dtypes: float64(2), int64(10), object(6)
memory usage: 1.4+ MB
```

We noticed that all the variables show the same number of non-null values (10,127). Nevertheless we also discovered that three categorical variables include an “Unknown” category. We will further discuss this matter later on in this paper.

A variable Unnamed:0 was probably generated from the loading of data in the workspace. It just looked like an index thus we decided to remove that artifact column. We also decided to temporarily keep the CLIENTNUM unique identifier just in case it might come handy in splitting the train and test sets by ID or performing other operations requiring a unique customer identifier.

Categorical Variables:
From the value_counts() method, barplots and cross tables we gained a few insights on the categorical features. We discovered that:

Most of the dataset is constituted by existing customers (83.9%) more than of attrited customers (16.1%). For this reason we also performed stratified sampling to split train and test respecting this proportion of attrited customers.

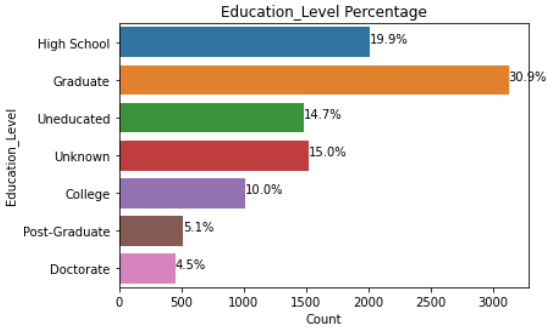


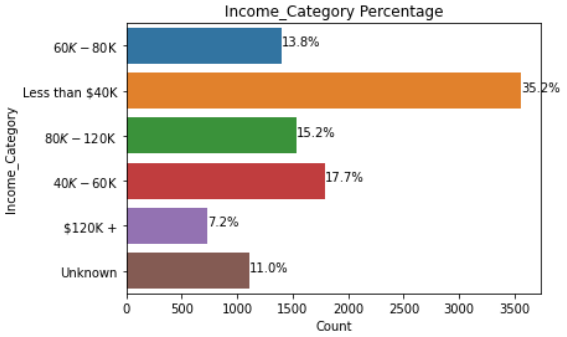
Marital_Status, Income_Category, Education_Level present an “Unknown” class. As we were unable to replace these values with a median or mean, we considered either replacing it with NaN values or to delete all the rows containing Unknown values before feeding the data to any machine learning model. After assembling the final datasets we opted for deleting the rows of the retained categorical variables.

```
Graduate         3128
High School      2013
Unknown          1519
Uneducated       1487
College          1013
Post-Graduate     516
Doctorate         451
Name: Education_Level, dtype: int64


Married     4687
Single      3943
Unknown      749
Divorced     748
Name: Marital_Status, dtype: int64


Less than $40K    3561
$40K - $60K       1790
$80K - $120K      1535
$60K - $80K       1402
Unknown           1112
$120K +            727
Name: Income_Category, dtype: int64
```

All the variables do not display the categories in an ordinal manner. This proves relevant especially for Income_Category, Card_Category, Education_Level which could have been

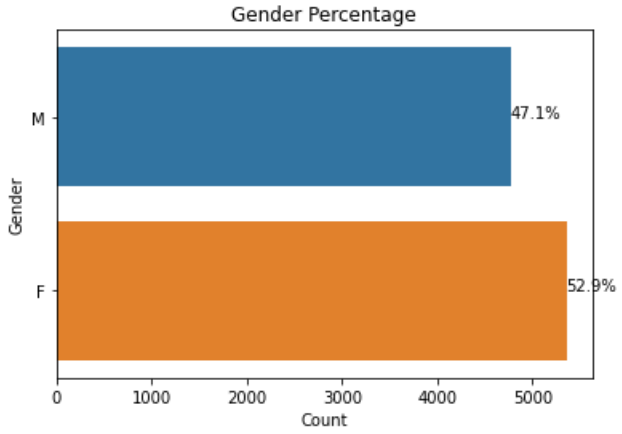
treated as ordinal variables or in the case of Income_Category as continuous (probably preprocessed with pd.cut()). We do not want our models to learn from the category order.





There are more or less the same number of Males and Females customers in the set. With a slight prevalence of females by 589 points.

```
F    5358
M    4769
Name: Gender, dtype: int64
```

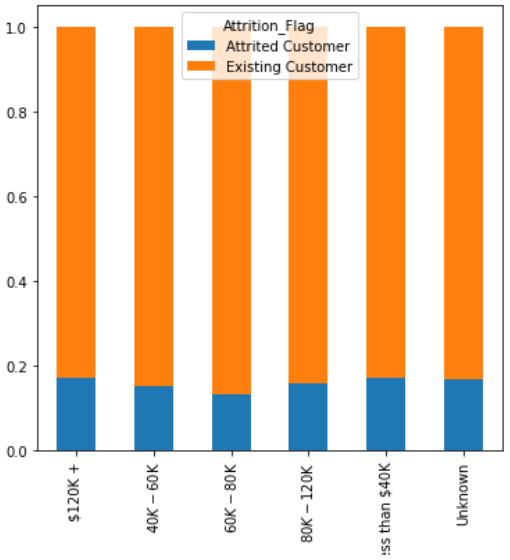


The majority of customers are graduates, married, with less than 40K of income and have a blue card. The aforementioned features are not necessarily associated.

From the cross tables analysis we concluded that the most promising categorical variables in distinguishing Attrition rate might be Gender, Income_Cat and Card_Cat.

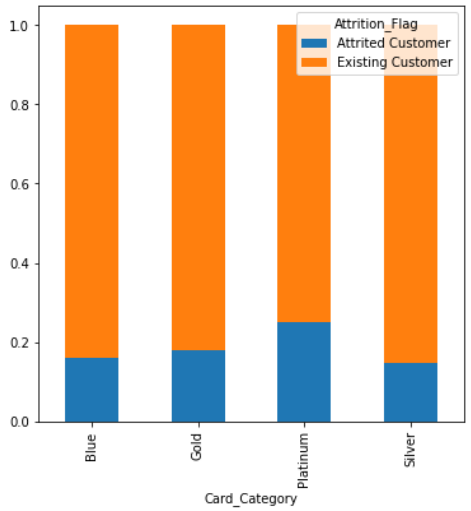
Attrition_Flag	Attrited Customer	Existing Customer
Income_Category		
$120K +	126	601
$40K - $60K	271	1519
$60K - $80K	189	1213
$80K - $120K	242	1293
Less than $40K	612	2949
Unknown	187	925

```
Income_Category
$120K +           0.209651
$40K - $60K       0.178407
$60K - $80K       0.155812
$80K - $120K      0.187162
Less than $40K    0.207528
Unknown           0.202162
dtype: float64
```

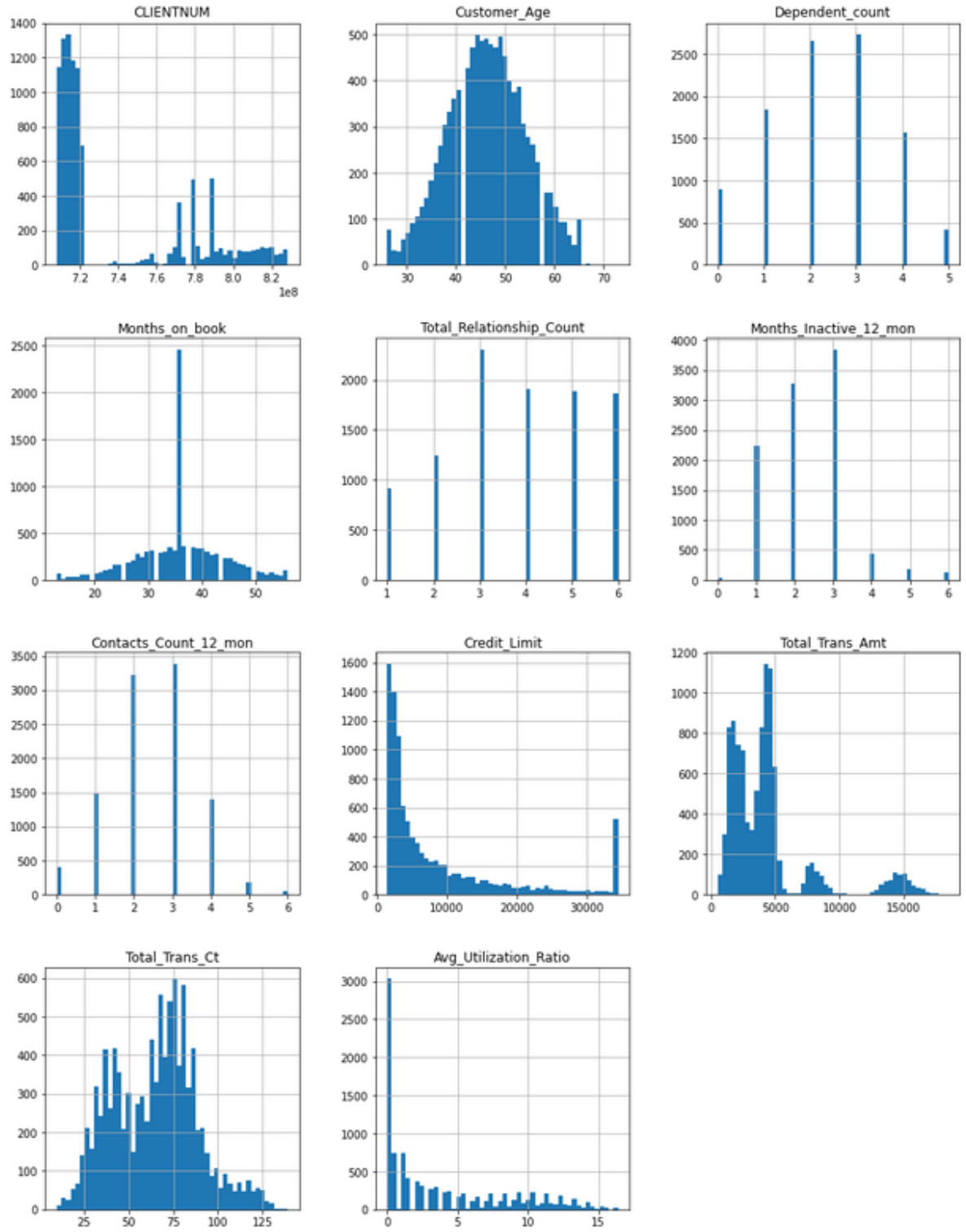


Attrition_Flag	Attrited Customer	Existing Customer
Card_Category		
Blue	1519	7917
Gold	21	95
Platinum	5	15
Silver	82	473

```
Card_Category
Blue        0.191866
Gold        0.221053
Platinum    0.333333
Silver      0.173362
dtype: float64
```

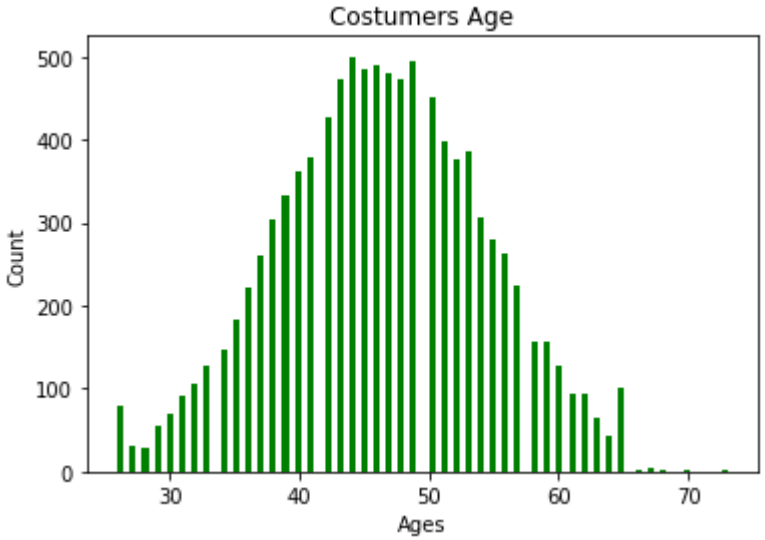


Numerical Variables:
We gained a few meaningful insights from the histograms:



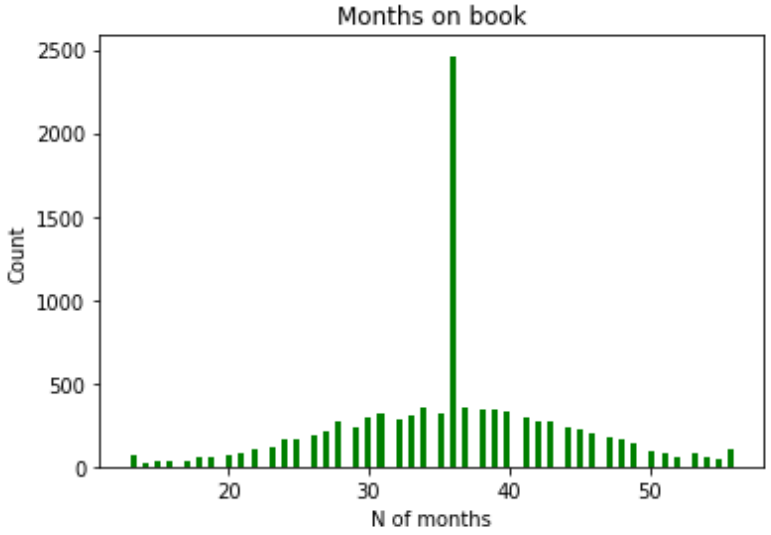
The customer age variable seemed capped around 42 and 58 years. It seems normally distributed with mean around 45 years of age. We replotted Customer_Age separately and counted the missing values. We discovered that the age ranges from 26 to 73 years and that

the only missing ages in the dataset were 69, 71, 72 proving that the cap we noticed was just a graphical artifact generated by the hist method.



Months_on_book showed a substantial outlier around 36 months counting 430 customers, although for the rest it follows a normal distribution trend. We hypothesized that maybe the StayWithMe (SWM) Bank offers a good 3-years plan or promotion and that the bank is able to retain most of the customers on book for this time span. Hence it seemed premature to exclude this outlier from the dataset, considering also the substantial data loss we would have had to face.

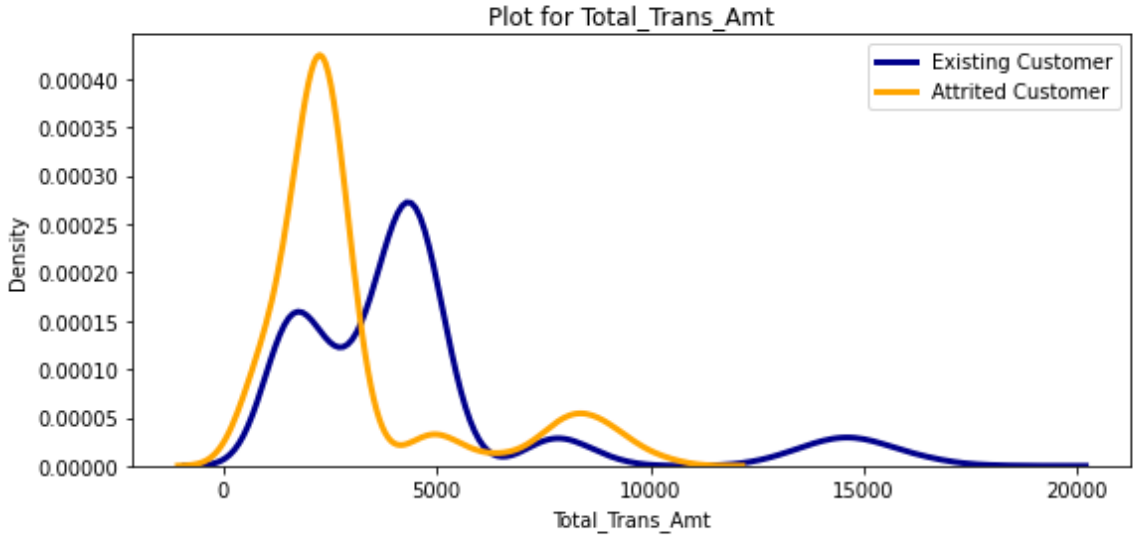
Credit_limit seems to present an outlier as well and its distribution is tail-heavy on the right. The outlier on the right end of the graph may be generated from the maximum threshold available for credit limit. Thus we retained this outlier too and moved on with our analysis.

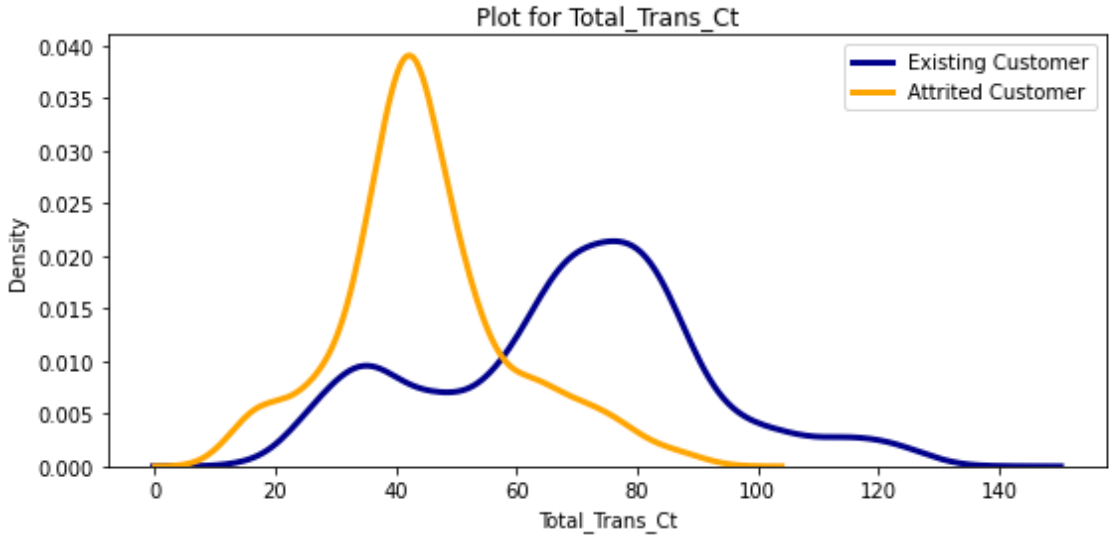


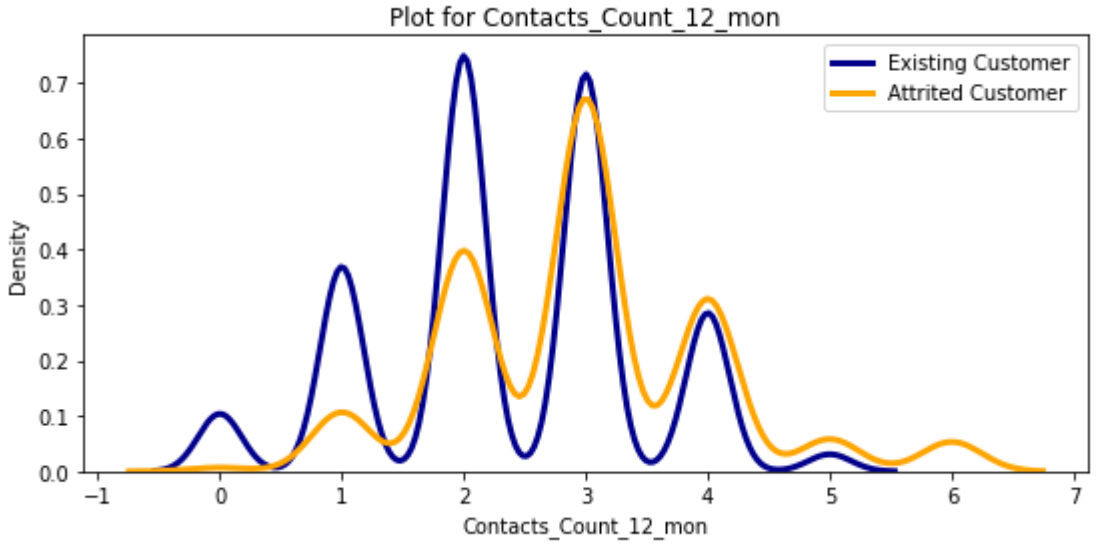
The dependent count ranges from 0 to 5 with a prevalence of 2-3 dependent people per customer. The total transaction amount in the last 12 months looks to present an irregular distribution, suggesting multiple trends, with rare transactions exceeding 15k. Credit_limit and the Avg_Utilization_Ratio seem to follow the same trend with a heavy right tail. The total_transaction_count seems more or less normally distributed as well. The total number of products held by the customer arrives at a maximum of 6. Most people do not stay inactive for more than 3 months a year and very few customers stay active all year long.

Kde curves

The kde curves for attrited and existing customers overlap or show the same trend for Customer_Age, Dependent_Count, Months_on_Book, Credit_Limit suggesting that these variables may not distinguish well our target variable categories. While the Months_Inactive_12_mon, the Average_Utilization_Ratio and the Total_Relationship_Count weakly differ in the two kde distributions. The Total_Trans_Amt, the Total_Trans_Ct and the Contacts_Count_12_mon show significantly different curves between Attrited and Existing customers at first glance. Thus the intuition that the total transaction count and amount should significantly differ between customers who attrit and customers who stay was confirmed by the kde plots.

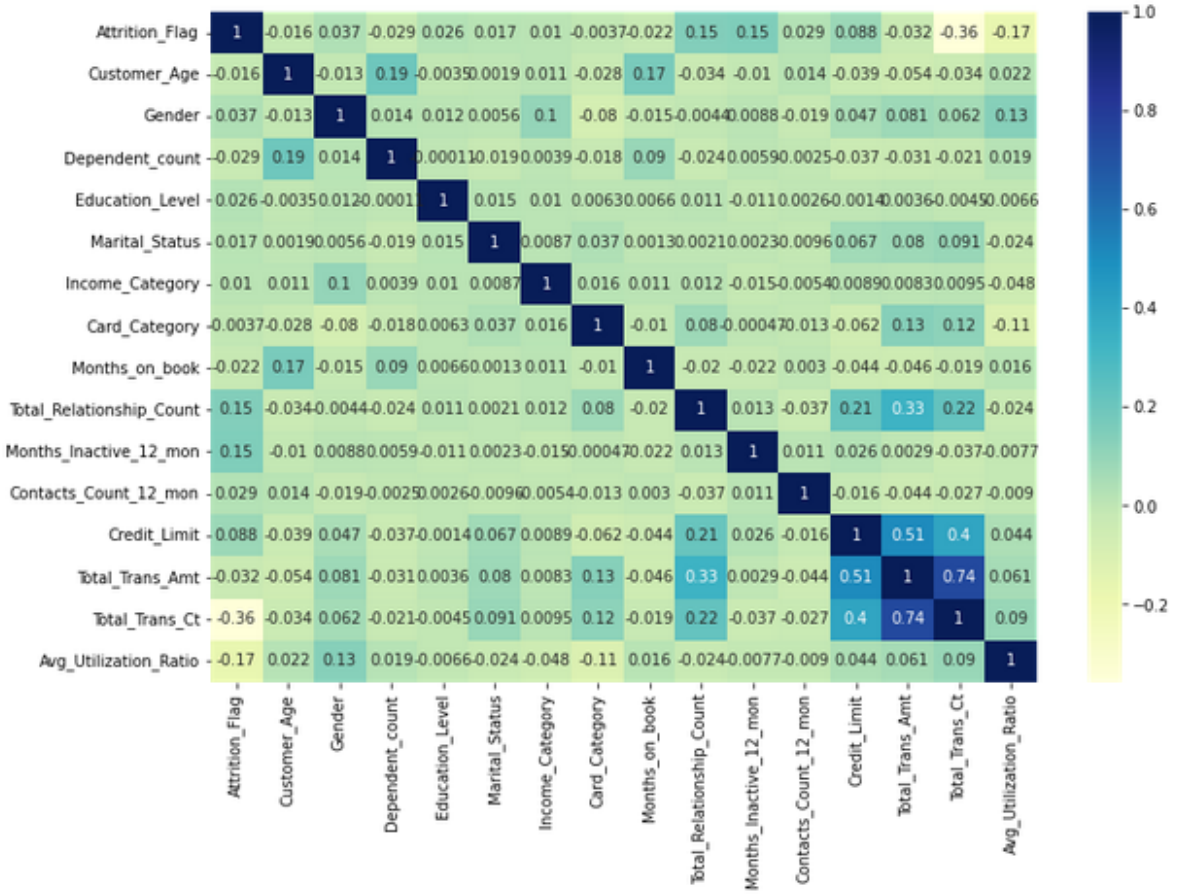






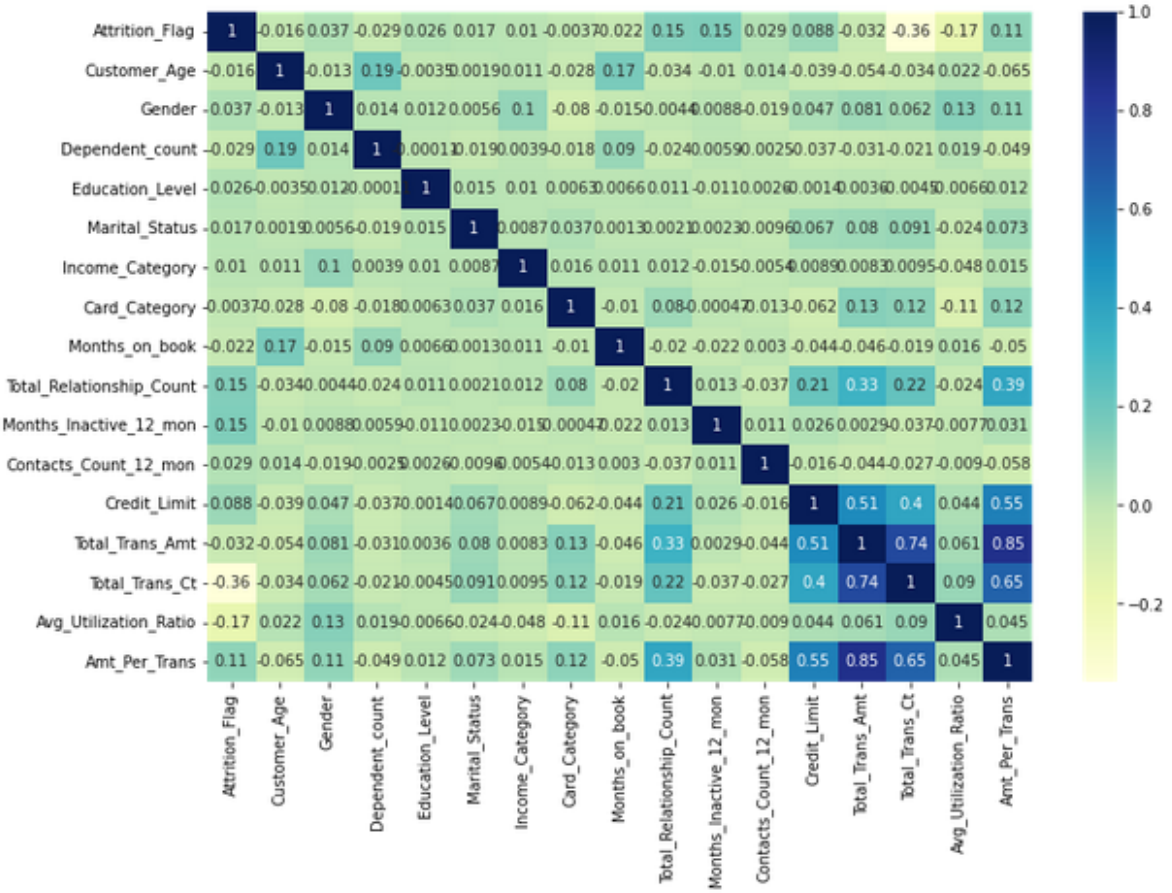
Correlation Analysis

We plotted the correlation matrix on both the full dataset and the train set. In the train set for both the plain and the stratified split the features looked very uncorrelated. While better results were obtained with the full dataset.



As you see all the variables proved pretty uncorrelated with attrition (close to 0) nevertheless the most correlated were: Months_Inactive_12_mon, Total_Relationship_Count, Total_Trans_Ct, Total_Trans_Amt and Avg_Utilization_Ratio.

We also tried generating a new attribute called Amt_Per_Trans which computes the ratio between Total_Trans_Amt and Total_Trans_Count . Then we recomputed the correlation matrix to inspect whether this new feature would correlate more with Attrition_Flag by combining Total_Trans_Amt and Total_Trans_Count. However the correlation of the new attribute with the Attrition was still weak (r=0.110520)



Finally we recomputed the correlation matrix after deleting all the rows containing “Unknown” values. The r scores improved a bit but not substantially.

Chi-squared

Analysis of our chi-squared test reveals that all but two of our categoricals are very loosely dependent on our target variable. The three exceptions to this are the Gender, Income Category and Education_Level variables. The two aforementioned variables have p-values < 0.05 meaning they are statistically significant, therefore permitting us to reject the null hypotheses of independence, thereby establishing that there is a relationship with Attrition_Flag.

Gender

Chi-squared test for independence

Test	Df	Chi-square	P-value
Pearson	1	13.8656	0.000196358
Log-likelihood	1	13.9168	0.000191082

Income_Category

Chi-squared test for independence

Test	Df	Chi-square	P-value
Pearson	4	12.3681	0.014814
Log-likelihood	4	12.5808	0.0135166

Education_Level

Chi-squared test for independence

Test	Df	Chi-square	P-value
Pearson	5	11.7724	0.0380432
Log-likelihood	5	11.082	0.0497782

As some of our categorical variables presented "unknown" data, we tried to perform the chi-square test by removing all the lines that presented an "unknown" value for the variable under consideration, from the original dataset, to see if there was any change. The variables containing “unknown” values are: Income_Category, Marital_Status and Education_Level.

Income_Category

Chi-squared test for independence

Test	Df	Chi-square	P-value
Pearson	4	12.3119	0.0151767
Log-likelihood	4	12.4891	0.0140618

Education_Level

Chi-squared test for independence

Test	Df	Chi-square	P-value
Pearson	5	11.3492	0.0448789
Log-likelihood	5	10.7865	0.0557817

From the results of the new tests, we noted that the Marital Status variable is confirmed to not be dependent on our dependent variable, while the Income_Category variable is confirmed to be dependent on Attrition_Flag. For the Education Level variable, we notice a worsening, in fact the p-value is less significant than before.

ANOVA Test

The ANOVA Test reveals which quantitative variables are most susceptible to variance in the target variable, indicating a causal relationship with the target variable. The quantitative variables that are most associated with Attrition_Flag are:

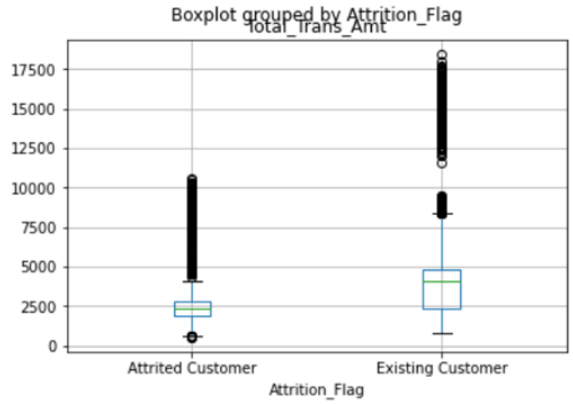
1. Total_Relationship_Count
2. Months_Inactive_12_mon
3. Contacts_Count_12_mon
4. Credit_Limit
5. Total_Trans_Amt
6. Total_Trans_Ct
7. Avg_Utilization_Ratio

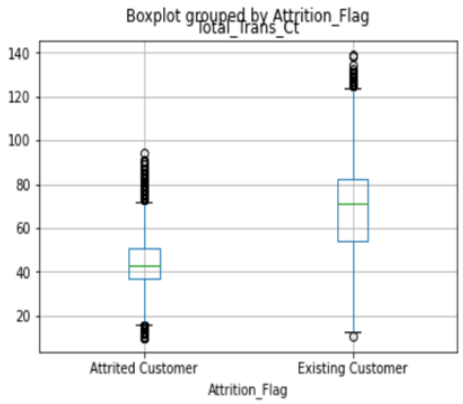
Of the seven variables listed above, two appear far more dependent on the target variable relative to the rest given the results demonstrated both in the ANOVA Test and the related boxplots:

1. Total_Trans_Amt

2. Total_Trans_Ct

```
Total_Trans_Amt
                     sum_sq       df           F        PR(>F)
Attrition_Flag  3.321765e+09      1.0  296.227714  1.857439e-65
Residual        1.135372e+11  10125.0         NaN           NaN


Total_Trans_Ct
                     sum_sq       df           F  PR(>F)
Attrition_Flag  7.695721e+05      1.0  1620.121692     0.0
Residual        4.809465e+06  10125.0          NaN     NaN
```





These analyses indicate that the transaction values and the number of transactions carried out by "attrited customers" fall within a range of values that is on average lower than the transaction values and the number of transactions counted for "existing customers."

To better evaluate the influence that the weight of the variables of the starting dataset has on the Attrition Flag variable, we decided to run different machine learning models on 3 different types of datasets and to compare the results. The datasets we have considered are the following:

1. A dataset containing all variables except for CLIENTNUM called “data_full”

2. A dataset containing the dependent variables obtained from our statistical analysis called “data_best”: Total_Relationship_Count, Months_Inactive_12_mon,

Contacts_Count_12_mon, Credit_Limit, Total_Trans_Amt, Total_Trans_Ctflowchart, Avg_Utilization_Ratio, Income_Category and Gender

3. A dataset called “data_mini” containing the two variable Total_Trans_Amt and Total_Trans_Ct

We analyzed the datasets using Logistic Regression, Support Vector Machine, Decision Tree and Random Forest and compared the different metrics obtained from the confusion matrix. We ran both “stratified” and “non stratified” models and then compared the results for each.The Stratified train-test split based on the Attrition_Flag that aims at retaining the proportion of Attrited and Existing customers of the dataset also in the train and test sets. Through these models, we were able to obtain a prediction of the Attrition_Flag variable with respect to the variables contained in each dataset. Using these percentages as the test size: 20%, 25% and 30%

For “data_full” these are the best results obtained:

```
DATA_FULL

LOGISTIC
stratified
test_size=0.2
Accuracy: 88.75 Precision: 90.62 Sensitivity: 96.59 Specificity: 47.69
CrossVal scores: [0.88374676 0.88481481 0.89111111] Mean e std: 0.887   0.003
Prec and recall: 0.906  0.966   F1 score: 0.935

SVM
stratified
test_size=0.2
Accuracy: 88.94 Precision: 90.47 Sensitivity: 97.06 Specificity: 46.46
CrossVal scores: [0.88670863 0.88703704 0.8962963 ] Mean e std: 0.89    0.004
Prec and recall: 0.905  0.971   F1 score: 0.936


DECISION TREE
stratified
test_size=0.3
Accuracy: 92.63 Precision: 95.75 Sensitivity: 95.45 Specificity: 77.87
CrossVal scores: [0.92001693 0.92636479 0.91617273] Mean e std: 0.921   0.004
Prec and recall: 0.958  0.955   F1 score: 0.956


RANDOM FOREST
stratified
test_size=0.3
Accuracy: 93.75 Precision: 94.26 Sensitivity: 98.55 Specificity: 68.65
CrossVal scores: [0.93821413 0.93609818 0.93607113] Mean e std: 0.937   0.001
Prec and recall: 0.943  0.985   F1 score: 0.964
```

For “data_best” these are the best results obtained:

```
DATA_BEST

LOGISTIC
strat
test_size=0.2
Accuracy: 88.2 Precision: 90.21 Sensitivity: 96.41 Specificity: 45.23
CrossVal scores: [0.87819326 0.87407407 0.88814815] Mean e std: 0.88    0.006
Prec and recall: 0.902  0.964   F1 score: 0.932

SVM
strat
test_size=0.2
Accuracy: 88.6 Precision: 90.16 Sensitivity: 97.0 Specificity: 44.62
CrossVal scores: [0.88411699 0.88037037 0.89111111] Mean e std: 0.885   0.004
Prec and recall: 0.902  0.97    F1 score: 0.935

DECISION TREE
strat
test_size=0.3
Accuracy: 92.3 Precision: 95.27 Sensitivity: 95.57 Specificity: 75.2
CrossVal scores: [0.91959374 0.92678798 0.91532599] Mean e std: 0.921   0.005
Prec and recall: 0.953  0.956   F1 score: 0.954

RANDOM FOREST
strat
test_size=0.2
Accuracy: 94.72 Precision: 95.28 Sensitivity: 98.59 Specificity: 74.46
CrossVal scores: [0.93891151 0.94185185 0.94037037] Mean e std: 0.94    0.001
Prec and recall: 0.953  0.986   F1 score: 0.969
```

For “data_mini” these are the best results obtained:

```
DATA_MINI

LOGISTIC
stratified
test_size=0.25
Accuracy: 81.95 Precision: 84.95 Sensitivity: 95.39 Specificity: 11.79
CrossVal scores: [0.82661927 0.81951027 0.81864876] Mean e std: 0.822   0.004
Prec and recall: 0.85   0.954   F1 score: 0.899

SVM
stratified
test_size=0.2
Accuracy: 83.96 Precision: 83.96 Sensitivity: 100.0 Specificity: 0.0
CrossVal scores: [0.83931877 0.83925926 0.83925926] Mean e std: 0.839   0.0
Prec and recall: 0.84   1.0     F1 score: 0.913

DECISION TREE
stratified
test_size=0.3
Accuracy: 87.63 Precision: 92.83 Sensitivity: 92.4 Specificity: 62.7
CrossVal scores: [0.87431231 0.88066018 0.87891617] Mean e std: 0.878   0.003
Prec and recall: 0.928  0.924   F1 score: 0.926

RANDOM FOREST
stratified
test_size=0.3
Accuracy: 88.78 Precision: 93.1 Sensitivity: 93.57 Specificity: 63.73
CrossVal scores: [0.89081676 0.89293271 0.88738357] Mean e std: 0.89    0.002
Prec and recall: 0.931  0.936   F1 score: 0.933
```

Analysis of these results reveals that the indicators of the confusion matrix assume higher values for all the stratified datasets considered when we run the Decision Tree and Random Forest with a test size of 30%. We have grouped these results in the following table:

	DECISION TREE						**RANDOM FOREST**					
	T.Size	**Strat**	**Acc**	**Prec**	**Sens**	**Spec**	**T.Size**	**Strat**	**Acc**	**Prec**	**Sen**	**Spec**
data_full	0.3	yes	93.63	95.75	95.45	77.87	0.3	yes	**93.75**	**94.26**	**98.53**	**68.65**
data_best	0.3	yes	92.3	95.27	95.57	75.2	0.3	yes	**94.67**	**95.61**	**98.16**	**76.43**
data_mini	0.3	yes	87.63	92.83	92.4	62.7	0.3	yes	88.78	93.1	93.57	63.73

Generally speaking, we noticed that passing from a dataset with many variables to a dataset with a small number of variables, the values of the indicators obtained from the models undergo a reduction. This is probably because the variables capable of predicting whether a

customer is retained or is lost are many and it is difficult to accurately predict with only a few simple variables, especially if it is based only on historical bank data regarding the customer. For confirmation statistical analysis carried out, we can note that in the Random Forest model with a test size, switching from the dataset containing all the variables to the dataset with the “best” variables considered most dependent on our target variable, leads to an increase in the levels of accuracy, precision and specificity. We also note that passing from the original dataset to the dataset containing the two variables Total_Trans_Amt and Total_Trans_Ct therefore by removing 13 variables from the model, we do not have drastic or significant reductions in our indicators.

Below you can also compare the performance measures of the Decision Tree and Random forest classifiers obtained on the data_best after deleting the “Unknown” values of Income_Category. The results are slightly worse if compared with the same classifiers obtained on the dataset retaining the “Unknown”. Nevertheless we hypothesized that either the classifiers are actually learning from the class “Unknown” or more probably the improvements are due to the data loss of more than 1k points (30%) from the dataset. We would not like our classifiers to learn from the unknown class but Income_Category is not strong for predictive power and the removal operation does not justify the consistent data loss.

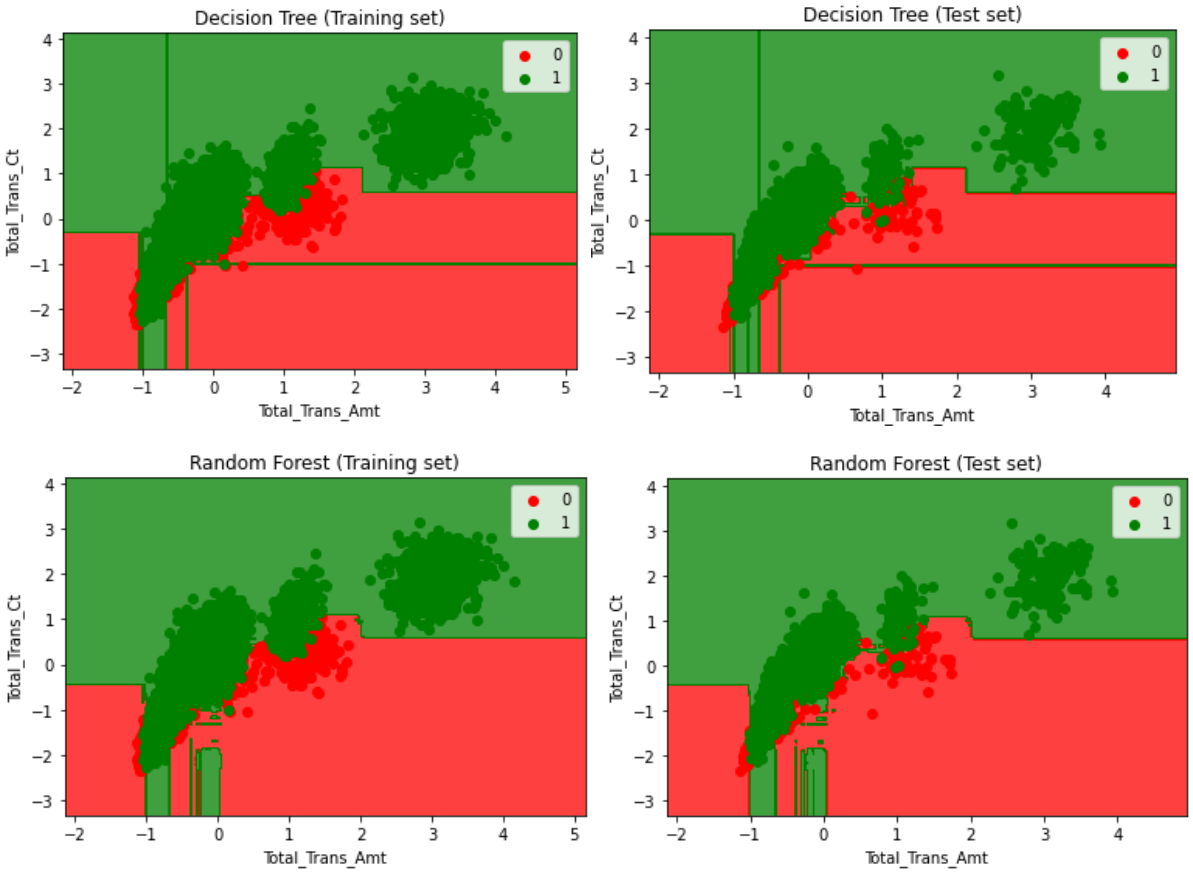
Decision Tree stratified for data_best without unknown

```
test_size=0.3
Accuracy: 91.46 Precision: 94.94 Sensitivity: 94.9 Specificity: 73.38
CrossVal scores: [0.91302281 0.91107941 0.91821208] Mean e std: 0.914
0.003
Prec and recall: 0.949  0.949   F1 score: 0.949
```

Random Forest stratified

```
test_size=0.3
Accuracy: 94.2 Precision: 94.98 Sensitivity: 98.28 Specificity: 72.69
CrossVal scores: [0.93726236 0.94008559 0.93770804] Mean e std: 0.938
0.001
Prec and recall: 0.95   0.983   F1 score: 0.966
```

The decision boundary helped further visualize our classifiers performance. We plot it for the two variables of Tot_Trans_Count over the Tot_Trans_Amt for both the training and the test set of the Decision Tree (test_size: 0.2, plain split) and the Random Forest classifier (test_size: 0.2, plain split) estimated by the RandomizedGridSearchCV. One conclusion we can draw is that as the total_trans_ct grows the classifier predicts that the customer is existing while the reserve trend is true for attrited. We can clearly see from these figures how the classifier is probably overfitting the data and that the number of attrited customers makes it very hard to spot meaningful trends by sight.



Conclusions

According to the performance metrics our study obtained better classifiers on the stratified train-test sets as they resemble the proportions found in the original datasets and in reality between attrited and existing customers.
The best models were the Decision Tree and the Random Forest which probably handle classification tasks better and suit our dataset better.
With the data_best we niched down the variables and still were able to obtain good results. Hence the data_best proved a good mid size dataset capable of exploiting the most predictive variables without sacrificing meaningful information. The data_best thus confirmed the preliminary analysis that we ran on the dataset before performing machine learning training.
We hypothesized that by removing the Unknown values from the datasets we would have obtained more significant results. Our metrics and the ANOVA test show that the metrics actually slightly worsened. That leads us to the conclusion that either the classifiers are learning from the Unknown class or the data loss due to the removal is too consistent and sacrifices too much information.
The transactions count and amount proved the most predictive for the models but this is clearly an artifact that generates logical issues. The classifiers are learning that the most predictive features are actually those that describe attrition but do not explain attrition. It would be difficult for a real bank to predict the attrited and existing customers from a datum they would gather only after the customer left. It would be interesting to rerun our entire models on a dataset excluding these two variables.