Comparing the effectiveness of Logistic Regression versus Machine Learning algorithms in predicting consumer attrition

Luis Blanco

Northwestern University

luisblanco2014@u.northwestern.edu

lblanco.ixmbabw2011@alumni.ie.edu

https://www.linkedin.com/in/luis-blanco-618aa0

June 2022

Abstract

Over the last decade, Fortune 500 companies have invested an average of $1.28 Trillion[1] annually in marketing expenses, with prominent players like JP Morgan Chase investing $2.5bn[2] and GEICO $1.6bn[3] in 2020 alone. Customer retention strategies are a primary recipient of marketing investment, given that cost-effective interventions are possible and, often, successful at preventing attrition. Accurately predicting the likelihood of a particular customer to attrite is the cornerstone of customer management strategies, and the model's precision significantly impacts the strategies' economics. The default model for predicting churn has been Logistic Regression, given the binary nature of attrition. Still, logistic regression has fallen out of favor with the nascence of more sophisticated machine learning algorithms. This research contrasts both methods in a particular use case, using a holistic approach for model performance beyond accuracy. The most surprising finding is that, when considering metrics beyond accuracy, the retention strategy's use case, and the data set size and complexity, it is worth exploring Tree-based and non-Tree-based models as part of the modeling process.

Keywords: customer attrition, machine learning, Exploratory Data Analysis, Linear Regression, Support Vector Machine Classifier, Random Forest Classifier, Gradient Boosting Machine Classifier, Xtreme Gradient Boosting Machine Classifier, Light Gradient Boosting Machine Classifier, SMOTE, feature scaling , model accuracy, recall, model interpretability, model complexity, model scalability.

[1] The CMO Survey 27th Edition, August 2021, Deloitte/Duke Univeristy's Fuqua School of Business / AMA
[2] Statista, Marketing spending of U.S. banks 2020
[3] Ad Age Leading National Advertisers 2020 Fact Pack

1 Introduction and Overview

Given that the ability to effectively predict customer attrition is a core capability in the customer retention process, management and data practitioners would benefit from a robust framework to decide which model works best, given their circumstances. Model accuracy is the most important element in the decision-making process but not the only one. While most of the research focuses on model accuracy ((Theofilatos A 2019) and (Wei Chen 2017) are examples), Almaliki suggests: "After categorizing the problem and understanding the data, the next milestone is identifying the algorithms that are applicable and practical to implement in a reasonable time. Some of the elements affecting the choice of a model are:

The accuracy of the model.

The interpretability of the model.

The complexity of the model.

The scalability of the model.

How long does it take to build, train, and test the model?

How long does it take to make predictions using the model?

Does the model meet the business goal?" (Almaliki 2019)

Therefore, this research will look into more than just accuracy for model performance.

2 Background

2.1 Logistic Regression

Sperendei, in his "Understanding logistic regression analysis", gave a powerful yet straightforward explanation of Logistic Regression. According to Sperandei, Logistic regression is used to obtain odds ratio in the presence of more than one explanatory variable. The procedure is quite similar to multiple linear regression, with the exception that the response variable is binomial. The result is the impact of each variable on the odds ratio of the observed event of interest. A logistic regression will model the chance of an outcome based on individual characteristics. Because chance is a ratio, what will be actually modeled is the logarithm of the chance given by:

$$\log\left(\frac{\pi}{1-\pi}\right) = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \ldots \beta_m x_m$$

Equation 1 Logistic Regression formula (Sperandai 2014)

where π indicates the probability of an event (e.g., death in the previous example), and βi are the regression coefficients associated with the reference group and the xi explanatory variables. At this point, an important concept must be highlighted. The reference group, represented by β0, is constituted by those individuals presenting the reference level of each and every variable x1...m. (Sperandei 2014) Sperendei continues with a clear application example in the medical domain.

Karp laid the groundwork with "*Using Logistic Regression to Predict Customer Attrition*" back in 1998, into applications of Logistic Regression,. In this paper, Karp proposes a simple platform to manage customer churn in terms of prediction and recommendation on win-back strategies. This framework consists of three stages in the approach – churn detection, customer profiling

and recommendations for timely personalized retention actions. In the section "Churn Detection via Predictive Analytics", Karp mentioned that there are different regression models that can be leveraged depending on the context where predictive analysis need to done. Logistic regression can be used in B2C customer attrition context to build the predictive model [17]. The predictive scores generated through this model will represent the risk quotient for each customer towards attrition. (Karp 1998).

2.2 Theoretical background on Machine Learning Algorithms

In 1959, Stanford and Bell Lab's researcher Arthur Samuel defined Machine Learning as "Field of study that gives computers the ability to learn without being explicitly programmed". More explicitly, Tom Mitchell defined Machine Learning in 1997 as "a computer program is said to learn from experience E with respect to some task and some performance measure P, if its performance on T, as measured by P, improves with experience E".

In his book "Hands-On Machine Learning with Scikit-Learn & TensorFlow", Aureliene Geron categorizes Machine Learning algorithms in several ways, the most relevant regarding the involvement of human intervention when training the models. He identifies four major categories depending on the amount of supervision involved: supervised learning, unsupervised learning, semisupervised learning, and reinforcement learning. In supervised learning, the training data you feed to the algorithm includes the desired solutions, called labels. Logistic Regression, Support Vector Machines (SVMs), Decision Trees and Random Forests are all examples of Supervised Learning. (Géron 2017).

Regarding the application to our research's domain, Sahar F. Sabbeh in "Machine-Learning Techniques for Customer Retention: A Comparative Study" compares the accuracy of

machine learning algorithms across several categories. In particular, Sabbeh compares the following algorithms:

1) Logistic regression.

2) Decision tree–CART.

3) Bayes algorithm: Naïve Bayesian.

4) Support Vector Machine

5) Instance – based learning: k-nearest Neighbor.

6) Ensemble learning: Ada Boost, Stochastic Gradient Boost and Random Forest.

7) Artificial neural network: Multi-layer Perceptron.

8) Linear Discriminant Analysis

In the Conclusions, she states that the study tried to present a benchmark for the most widely used state of the arts for churn classification. The accuracy of the selected models was evaluated on a public dataset of customers in Telecom Company. Based on the findings of this study, ensemble–based learning techniques are recommended as both Random forest and Ad-boost models gave the best accuracy. (Sabbeh 2018)

3.- Dataset description and EDA

The dataset Chrun_Modeling is part of SuperDataScience's data repository for the Deep Learning A-Z™ course (Eremenko 2018). It consists of 10,000 rows and 14 columns. The columns contain the following variables: *RowNumber, CustomerId, Surname, CreditScore, Geography, Gender, Age, Tenure, Balance, NumOfProducts, HasCrCard, IsActiveMember, EstimateSalary, and Exited*. *Exited* is the target binary variable, with "1" for when the customer attrited during the review period and "0" for when it didn't. The features *RowNumber*, *CustomerID*, and *Surname* are case-identifying variables that don't add any value and thus are

eliminated from the dataset. Of the remaining ten independent variables, four are Nominal, three Continuous, and three Discrete. The numerical values' descriptive statistics are in the table below:

	CreditScore	Age	Tenure	Balance	NumOfProducts	HasCrCard	IsActiveMember	EstimatedSalary	Exited
count	10000.00	10000.00	10000.00	10000.00	10000.00	10000.00	10000.00	10000.00	10000.0
mean	650.53	38.92	5.01	76485.89	1.53	0.71	0.52	100090.24	0.2
std	96.65	10.49	2.89	62397.41	0.58	0.46	0.50	57510.49	0.4
min	350.00	18.00	0.00	0.00	1.00	0.00	0.00	11.58	0.0
25%	584.00	32.00	3.00	0.00	1.00	0.00	0.00	51002.11	0.0
50%	652.00	37.00	5.00	97198.54	1.00	1.00	1.00	100193.91	0.0
75%	718.00	44.00	7.00	127644.24	2.00	1.00	1.00	149388.25	0.0
max	850.00	92.00	10.00	250898.09	4.00	1.00	1.00	199992.48	1.0

Table 1 Dataset's Numerical Values Descriptive Statistics

It is worth noting that Table 1 shows a broad distribution in *CreditScore, Age, Balance* and *EstimatedSalary*, while the variable *NumOfProducts* is more tightly packed. For the binary variables *HasCrCard* and *IsActiveMember*, Table 1 offers little insight, and the variables are explored further in the upcoming section.

After creating a train set with 80% of the observations and a test set with the remaining 20%, we'll be looking more closely at the variables and their distribution.

As for the target variable *Exited*, 20.5% of the Train set attrited, while the remaining stayed.

As shown in Figure 1, the numerical variables *Age, CreditScore, Balance* and *EstimatedSalary* show different distributions. The *Age* variable doesn't have a normal distribution as it is highly skewed to the left and therefore resembles a Poisson distribution, similar to the *CreditScore* variable. The *Balance* variable has an interesting distribution, with the highest frequency value being "0", but the remaining values have a normal distribution.

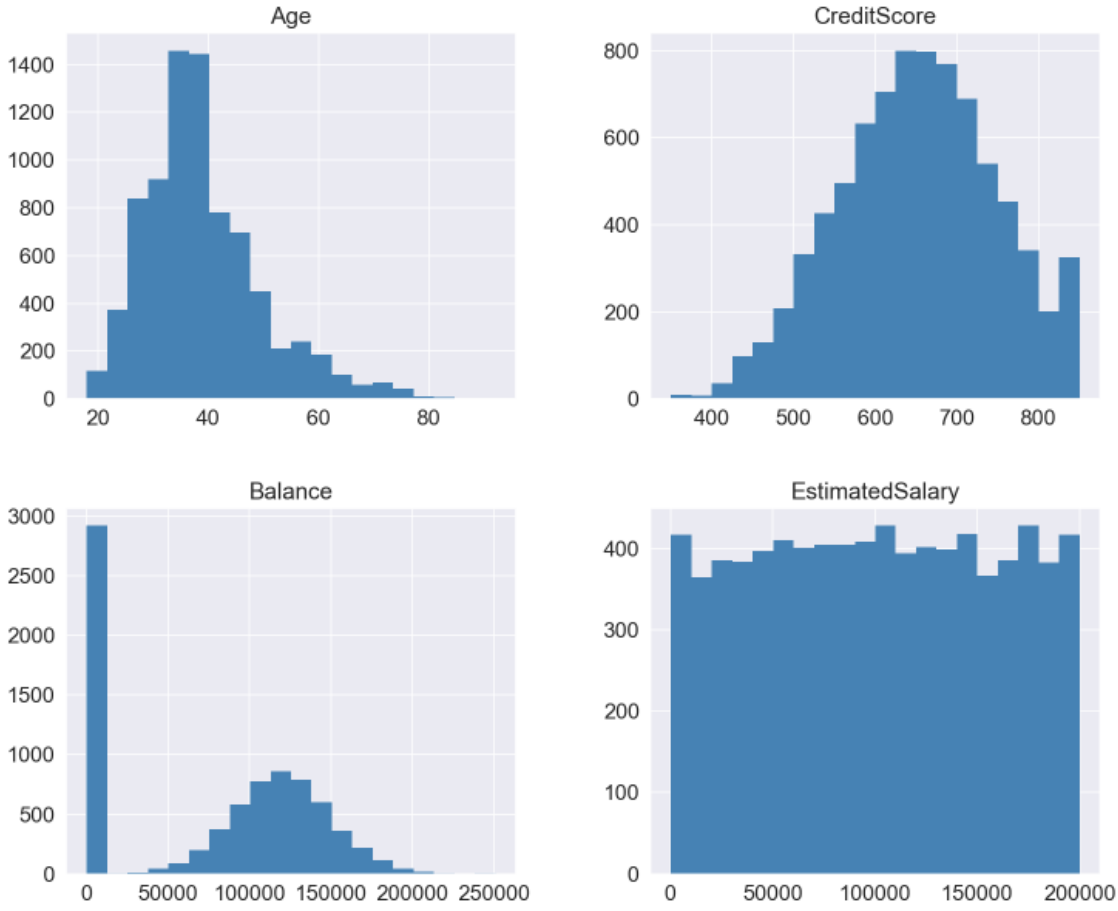

Figure 1 Numerical variables

As Figure 2 shows, none of the variable mixes have a linear correlation greater than 0.03, which bodes well for not having a collinearity issue when fitting the respective models.

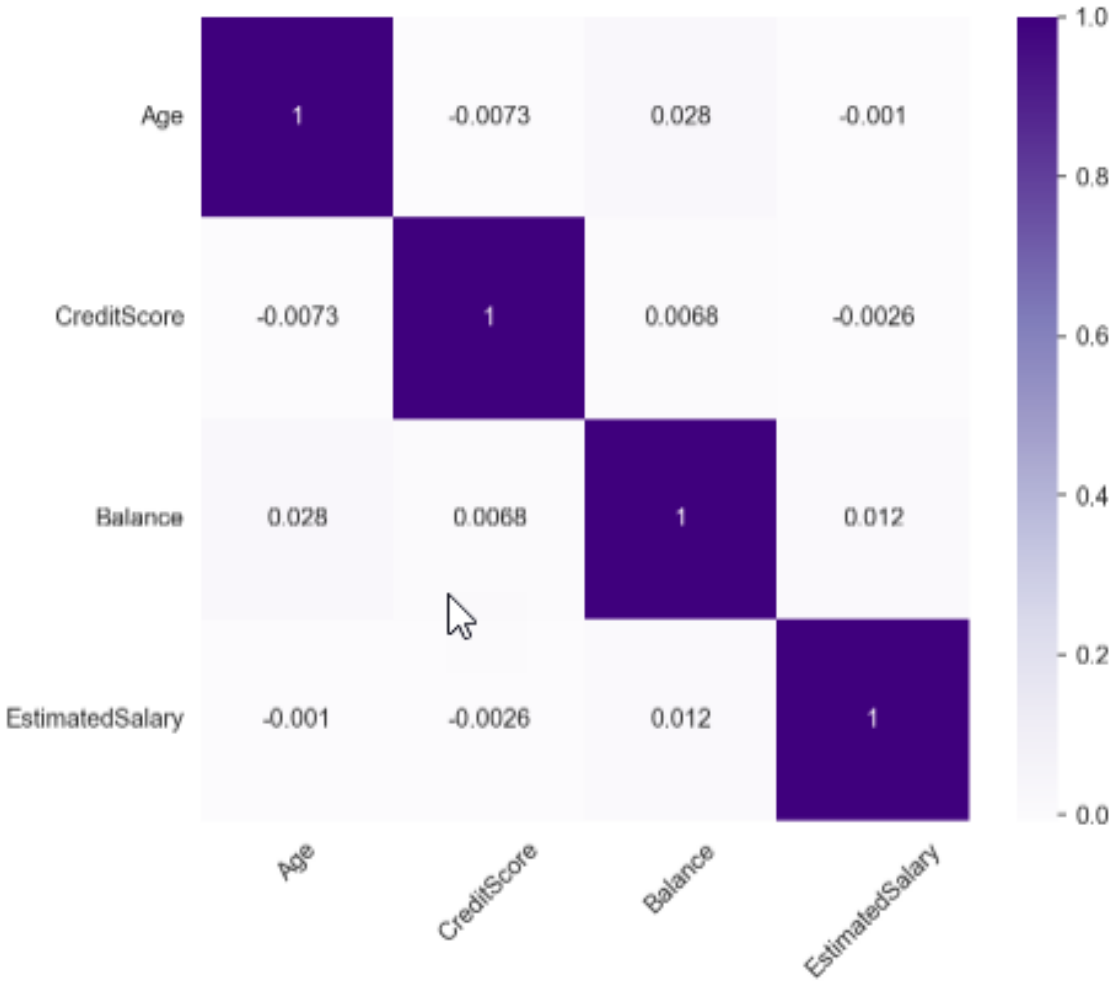

Figure 2 Correlation heat map between numerical variables

Looking at the relationship of these variables with the target variable, we can observe the following:

As observed in Figure 3, for the variable *Age*: the distribution of those retained is very different from the ones attriting, which is translated into Age being a variable that will help discriminate between the two.

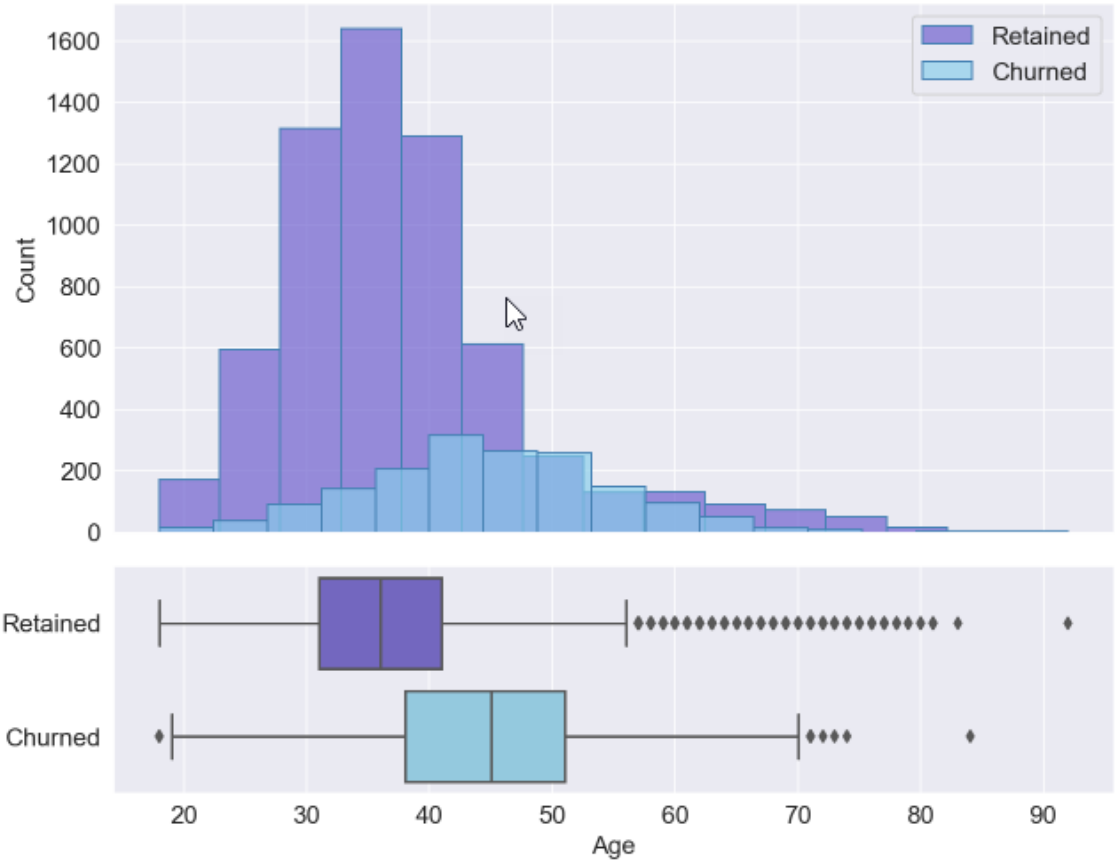

Figure 3 Age variable distribution: Retained and Churned

As shown in Figure 4, for *CreditScore*, we can see that both Retained and Churned have very similar distributions:

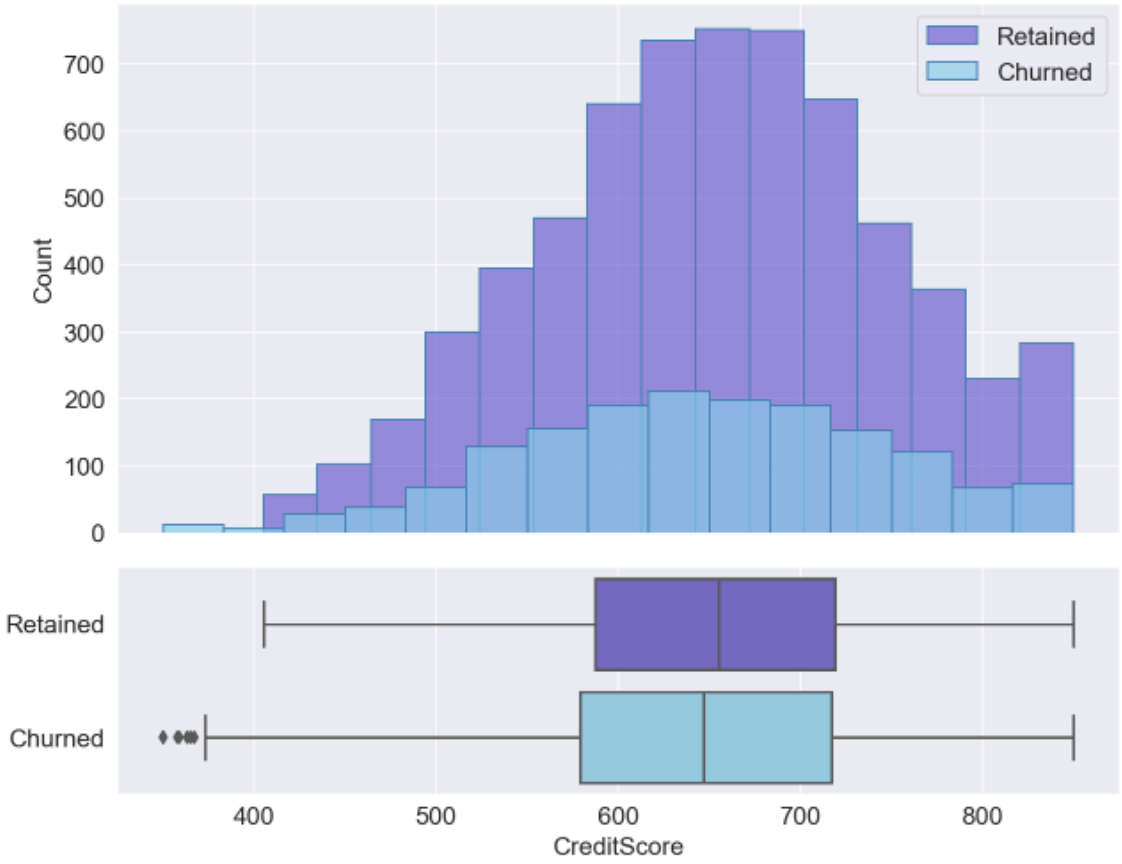

Figure 4 CreditScore variable distribution: Retained and Churned

As shown in Figure 5, *Balance* displays an interesting pattern: while distributions are similar between Retained and Churned, a higher proportion of 0 balance in Retained versus Churned is a counterintuitive finding.

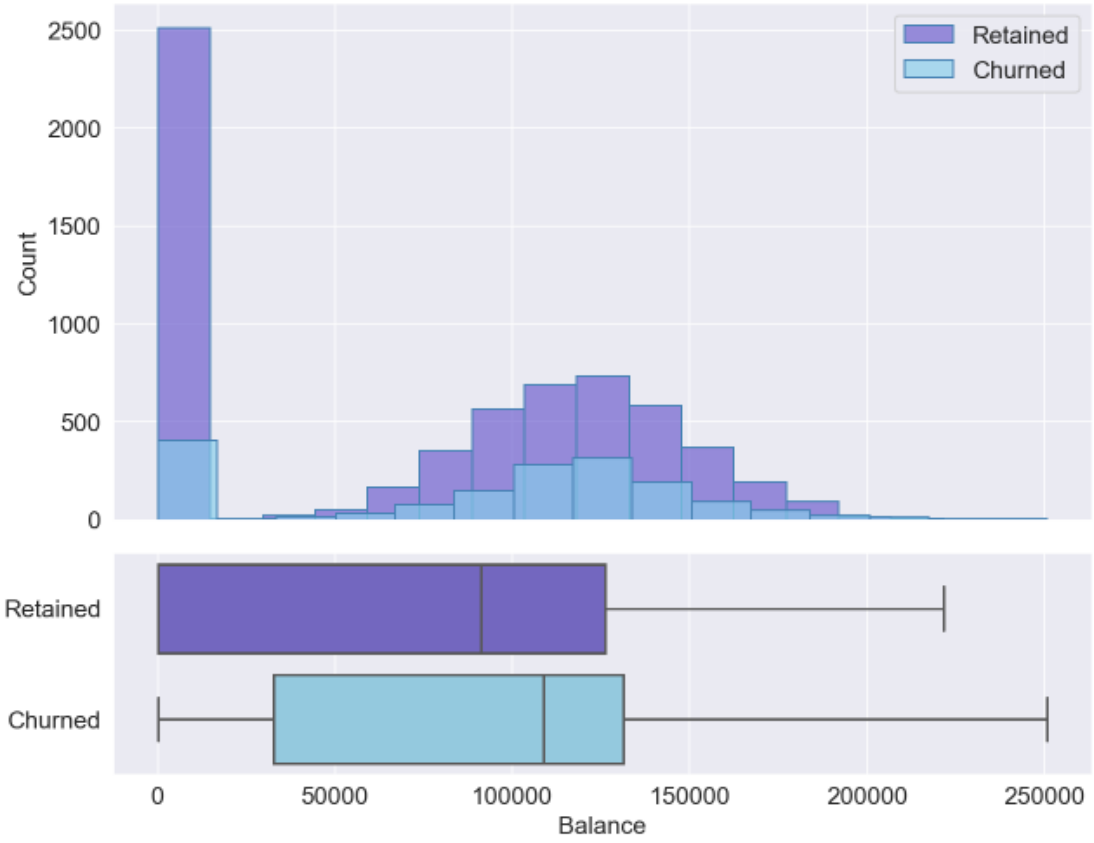

Figure 5 Balance variable distribution: Retained and Churned

As shown in Figure 6, *EstimatedSalary* shows a very similar distribution between variables.

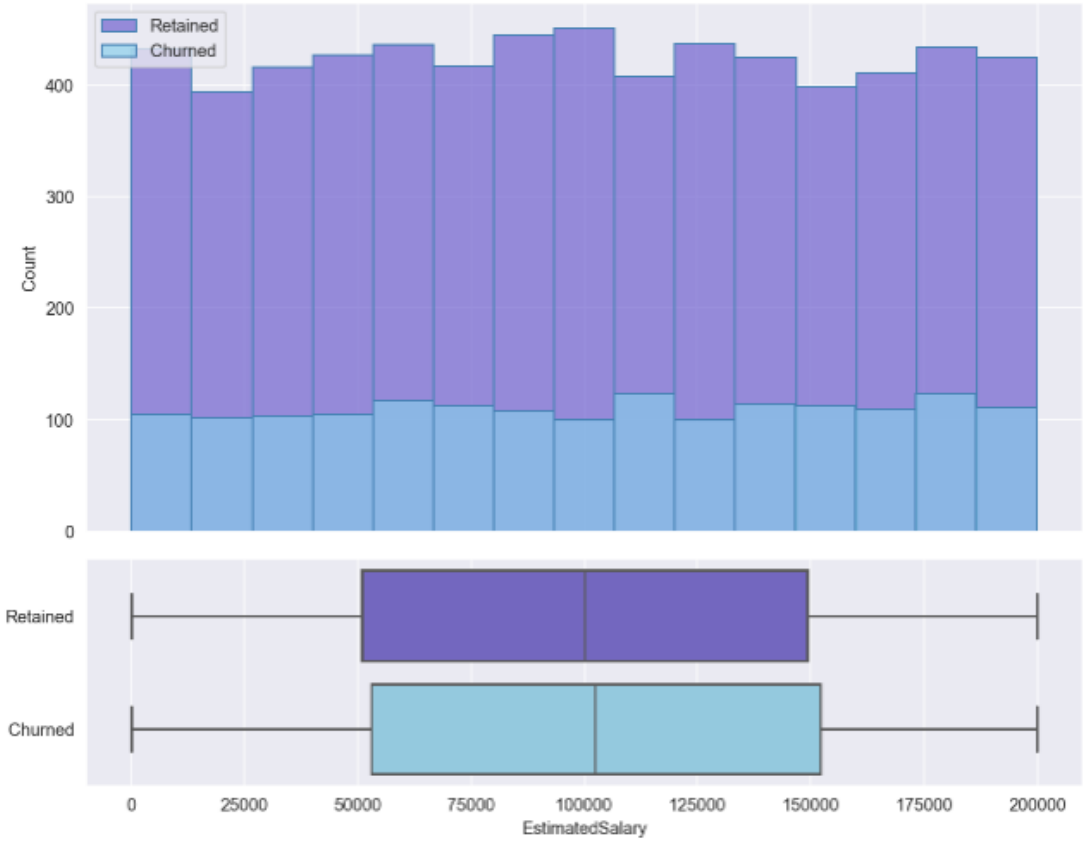

Figure 6 EstimatedSalary variable distribution: Retained and Churned

Continuing with the EDA for Categorical variables related to the target variable Retained/Churned, we can observe the following:

As shown in Figure 7, *Geography* is a relevant variable: while Frances has the highest frequency, Germany has the highest churn rate, a potential good predictor of attrition.

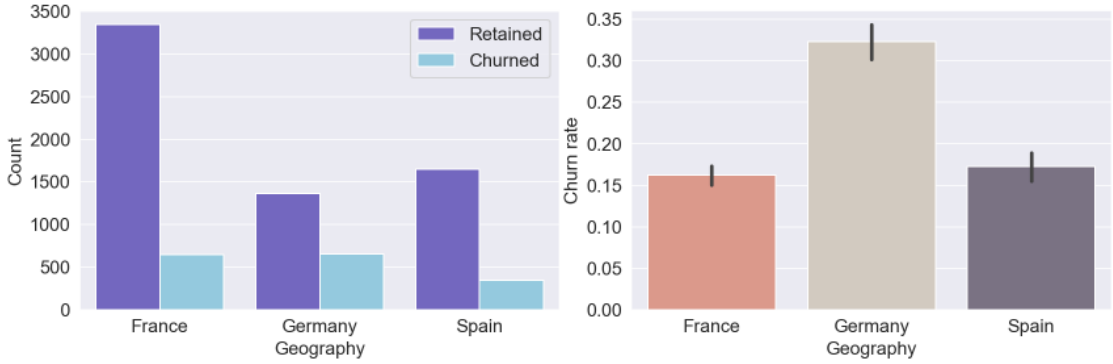

Figure 7 Geography by Retained vs. Churn and Churn Rate

As shown in Figure 8 for *Gender*, females show a higher churn rate than males.

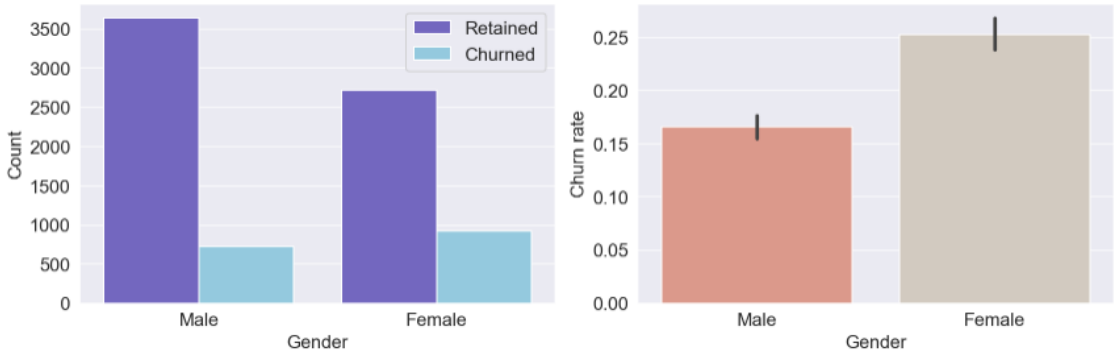

Figure 8 Gender by Retained vs. Churn and Churn Rate

The variable *Tenure* shows no clear distinction between the ten different classes, as shown in Figure 9.

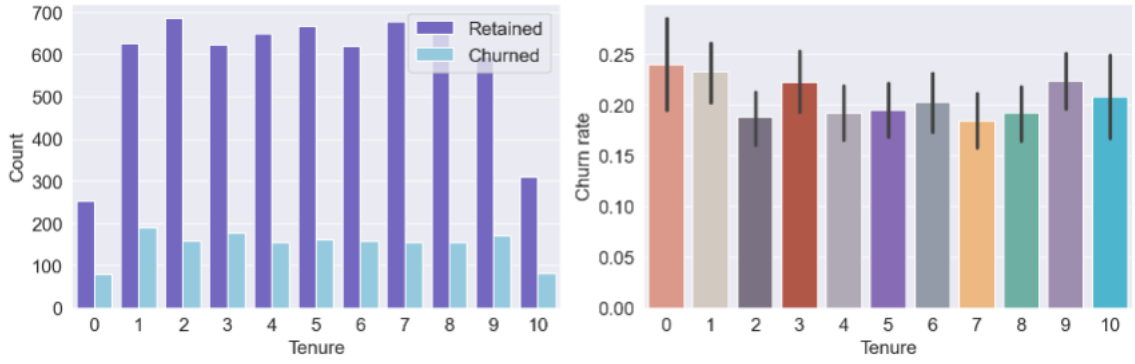

Figure 9 Tenure by Retained vs. Churn and Churn Rate

As Figure 10 shows, the variable *NumofProducts* for two products has a churn rate significantly lower than for 1 product, but counterintuitively, churn rates are high in 3 and 4 products.

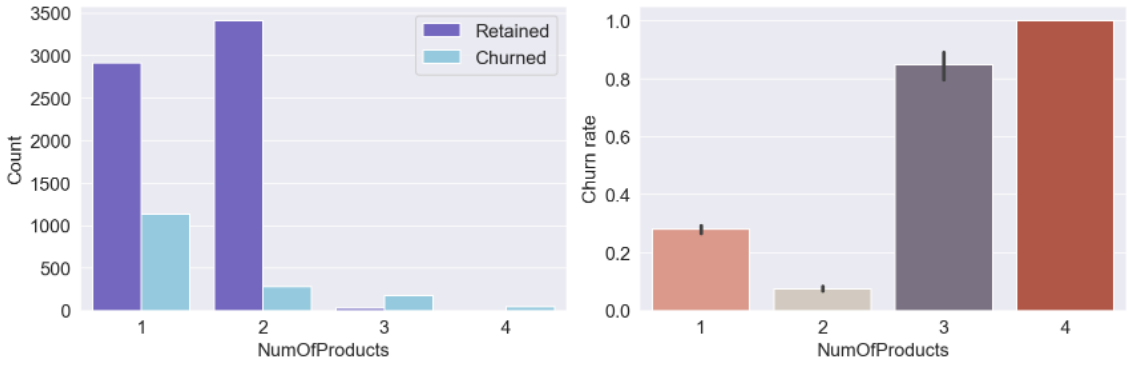

Figure 10 NumOfProducts by Retained vs. Churn and Churn Rate

Surprisingly, the *HasCrCard* variable shows no difference in the churn rate between these two segments, as can be observed in Figure 11.

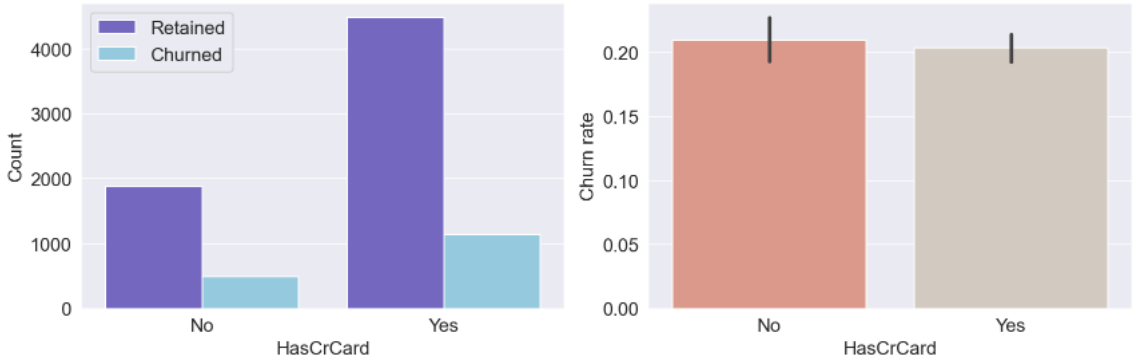

Figure 11 HasCrCard by Retained vs. Churn and Churn Rate

IsActiveMember: As Figure 12 shows, Active Members have a lower attrition rate than non-members.

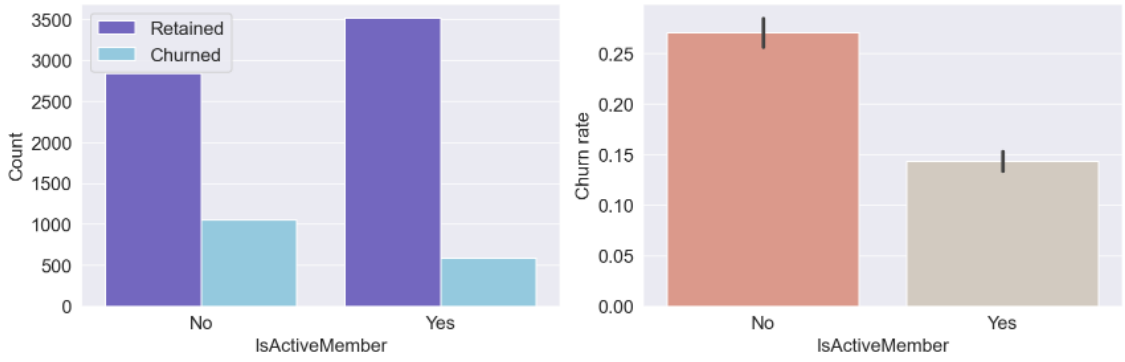

Figure 12 IsActiveMember by Retained vs. Churn and Churn Rate

4.- Research design and Modeling Methods

The first experiment runs the following models on the entire 10,000 observation set: Logistic Regression, Support Vector Machine Classifier, Random Forest Classifier, Gradient Boosting Machine Classifier, Xtreme Gradient Boosting Machine Classifier, and Light Gradient Boosting Machine Classifier. The experiment uses the recall metric (True Positive / True Positive + False negative) to measure accuracy, which attempts to solve the following question: What proportion of actual positives did the model identify correctly? (Developers 2020).

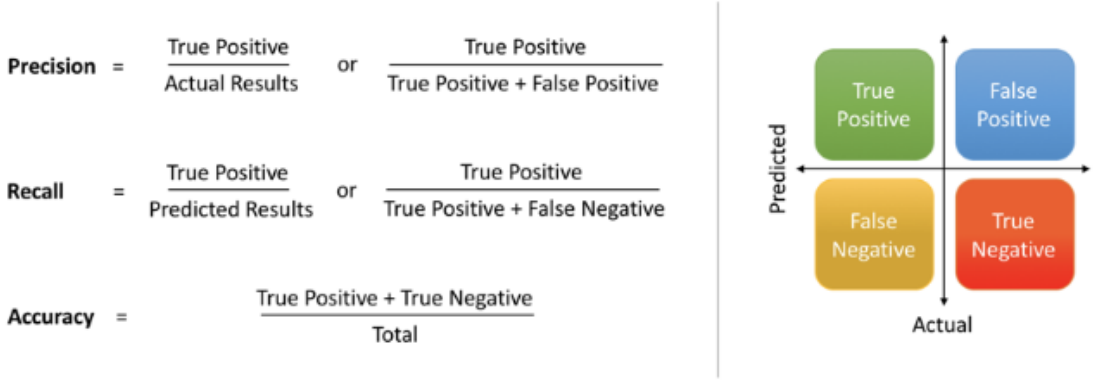

Figure 13 Model Performance metrics. Saxena 2018

Each experient includes commentary on interpretability, complexity and uses model run-time[4] as a proxy to "How long does it take to make predictions using the model?". In the conclusions, the report addresses the question of meeting the business goal and discusses scalability. The second experiment replicates the first experiment but on a randomly selected

[4] Run time is calculated within the Python 3 code running in a local Jupyter notebook v 3.0.14. The local hardware is an AMD Ryzen 9 5900HS with Radeon Graphics at 3.30 GHz, 16.0 GB of RAM and an NVIDA GeForce RTX 3060 GPU GDDR6 at 6GB. A different hardware configuration or running in cloud services such as AWS' ML or Google Cloud's AI and ML services would show different results.

subset of 1,000 observations of the original dataset. Finally, the researcher compares the results across both experiments on different population sizes, and the findings are reported in the Conclusions section.

4.1 Data preparation:

Based on the insights obtained from EDA, we can conclude that *EstimatedSalary*, *Tenure*, and *HsCreditCrd* variables are unlikely to provide incremental predictive power, given how similar the distributions are between the Remain and Churned segments.

Because many algorithms require their features to be numerical, the researcher converted *Gender* and *Geography* variables into integer values: Male = 1, Female = 0, and Germany =1, Spain and France = 0. Similarly, the researcher used feature scaling for *CreditScore*, *Age*, and *Balance* variables, given how some algorithms are sensitive to different scales. Lastly, the class sizes differ dramatically because the target variable (Churn) is significantly less prevalent than Remain. Class imbalance is known to impact model performance (Japkowicz 2002) , and SMOTE has proven to be an effective technic to solve this problem. (Chawla 2002). Python has a simple routine for SMOTE, so the researcher applied the function to the training set. For model fitting and specifically on parameters, the researcher used Scikit-Learn's function GridSearchCV, which performs an exhaustive search over specified parameter values for an optimal estimator. The parameters of the estimator used to apply these methods are optimized by cross-validated grid-search over a parameter grid (Pedregosa 2011).

4.2 Modeling methods results

In this section, the researcher introduces each model and, as mentioned in the Introduction section, will review and discuss the models' accuracy (measured with recall),

interpretability, and complexity. The researcher also uses a timestamp in each model fitting to measure run-time as a proxy for how long it takes to make predictions.

4.2.1 Logistic Regression results

The first model fitted is logistic regression, using the Scikit-Learn.LogisticRegression classifier. For our 10,000 record dataset, the results are the following:

Accuracy: (Recall): 0.73

Complexity: the model has low complexity, given that it only uses seven features with its correspondent coefficients. Compared with Machine Learning algorithms, Logistic Regression doesn't have various activation functions or hyperparameters to be optimized.

Interpretability: using the lr_model.coef_[0] function, we can observe that *IsActiveMember* is the most important variable, with *Age* a distant second.

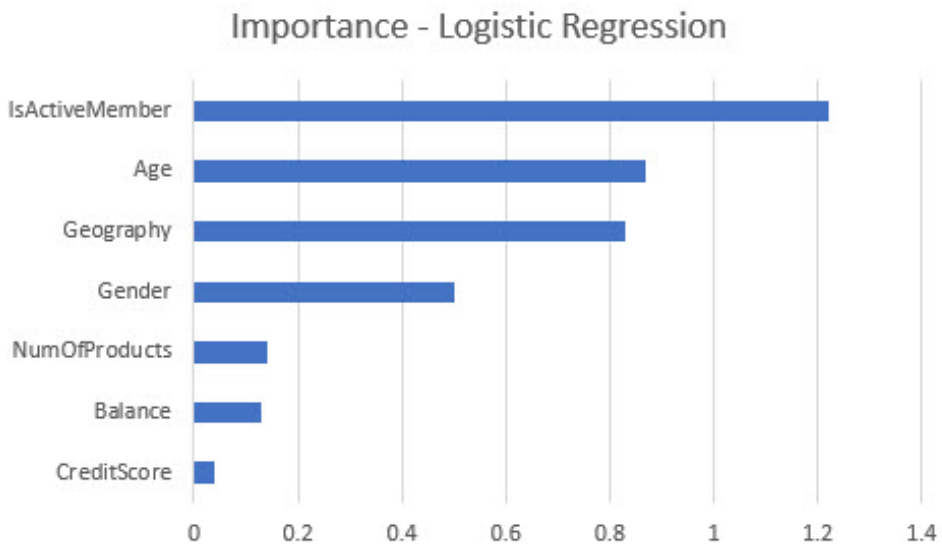

Figure 14 Feature importance - Logistic Regression

Time to fit model in train set: 0:00:00.68 seconds.

3.5 Support Vector Machine Classifier results

The second model fitted is a Support Vector Machine Classifier, using the Scikit-Learn. sklearn.svm.SVC classifier. For our 10,000 record dataset, the results are the following:

Accuracy: (Recall): 0.69

Complexity: SVMCs have a higher complexity than Logistic Regression models, given that it has several parameters to consider, such as the kernel type, Kernel coefficient for 'rbf,' 'poly' and 'sigmoid' (Gamma), the Regularization parameter (C), well as other less-used parameters.

Interpretability: using the SVC_clf2.coef_[0] function, we can observe that, inversely than in Logistic Regression, *Age* is the most important variable, with *IsActiveMember* a distant second.

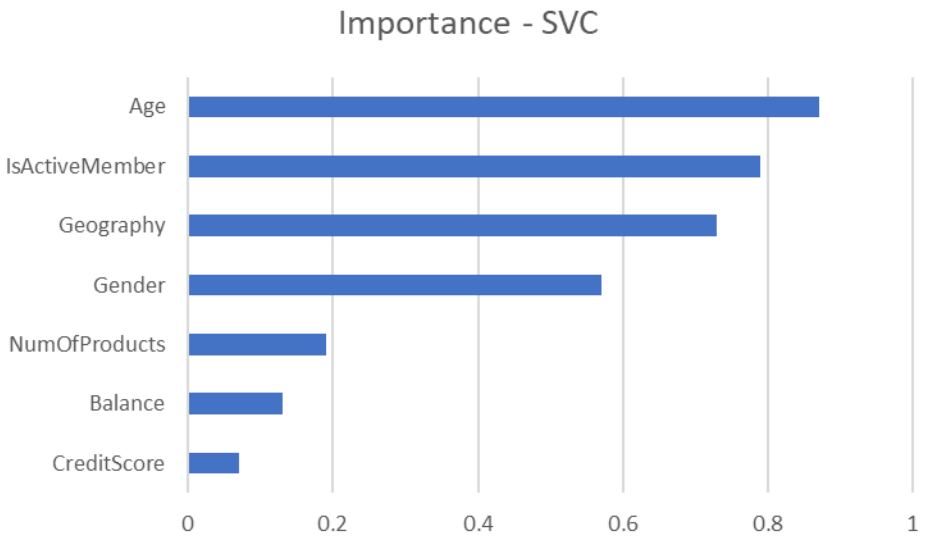

Figure 15 Feature Importance - SVC

Time to fit model in train set: 0:03:57.35, significantly longer than Logistic Regression.

3.6 Random Forest Classifier results

The next model fitted is a Random Forest Classifier, using the Scikit-Learn. sklearn. RandomForestClassifier and GridSearchCV to optimize the parameters. For our 10,000 record dataset, the results are the following:

Accuracy: (Recall): 0.77

Complexity: Random Forest algorithms have a significantly higher complexity than Logistic Regression models, given that it has several parameters to consider. The key parameters are the number of trees in the forest (*n_estimators*), the function to measure the quality of a split (*criterion*), and the maximum depth of the tree in terms of nodes (*max_depth*), among others. These were the parameters for the model fitted:

bootstrap: True
criterion: gini
max_depth: 6
max_features: auto
min_samples_leaf: 5
min_samples_split: 2
n_estimators: 100

Interpretability: Using the best_rf_clf.best_estimator_ function included in the GridSearchCV, we can observe that, as in the SVC, *Age* is the most important variable. As Figure 16 shows, with *NumOfProducts* is the second most valuable variable, different than the first two models.

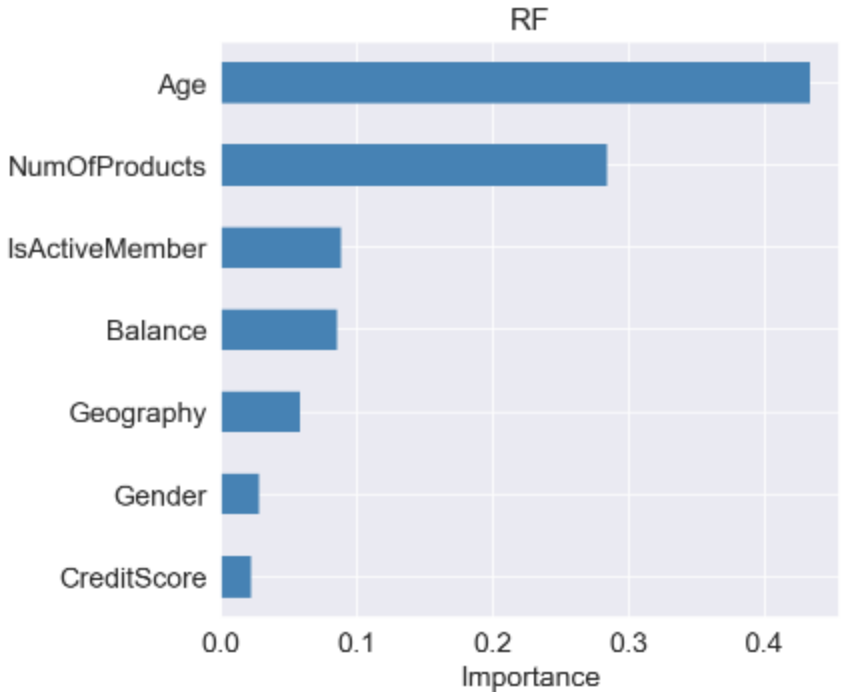

Figure 16 Feature Importance - Random Forest

Time to fit model in train set: 0:00:27.67. It is relevant to highlight how quicker it is to t rain a Random Forest Classifier than an SVMC.

3.7 Gradient Boosting Machine Classifier results

The next model fitted is a Gradient Booster Machine Classifier, using the Scikit-Learn. sklearn. GradientBoostingClassifier. According to Scikit-Learn's guide, GBMC builds an additive model in a forward stage-wise fashion; it allows for the optimization of arbitrary differentiable loss functions. In each stage, n_classes_ regression trees are fit on the negative gradient of the binomial or multinomial deviance loss function. (Pedregosa 2011)

For our 10,000 record dataset, the results are the following:

Accuracy: (Recall): 0.76

Complexity: Gradient Boosting algorithms are a step up in sophistication beyond Random Forests, given that it combines several models and thus have an increased level of complexity. They have a long list of parameters, such as the loss function to be optimized (*loss*), *learning_rate* that shrinks the contribution of each tree and *n_estimators,* that is the number of boosting stages to perform. For our model, these are the key parameters:

learning_rate: 0.01
max_depth: 3
max_features: auto
min_samples_leaf: 3
min_samples_split: 5
n_estimators: 600 – optimal: 284
n_iter_no_change: 20
subsample: 0.66
tol: 0.01
validation_fraction: 0.2

Interpretability: Using the .best_estimator_ function included in the GridSearchCV, we can observe that the relative weight importance is very similar to the Random Forest model.

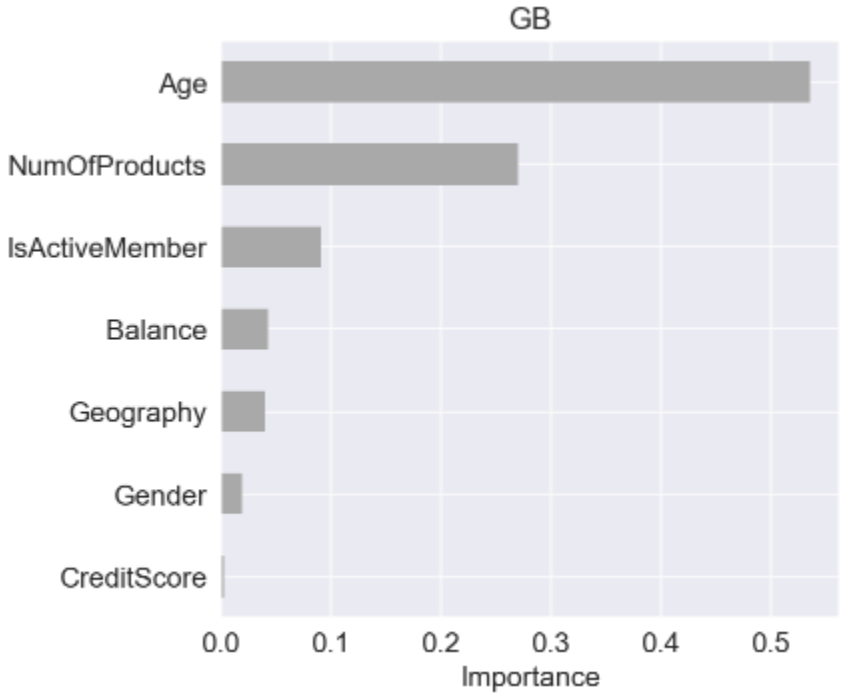

Figure 17 Feature Importance - Gradient Booster

Time to fit model in train set: 0:00:36.65, similar to the time it took to fit the Random Forest algorithm.

3.8 Xtreme Gradient Boosting Machine Classifier results

The next model fitted is an Xtreme Gradient Booster Machine Classifier, using the xgboost XGBCClassifier. XGBoost is an optimized distributed gradient boosting library designed to be highly efficient, flexible and portable. It implements machine learning algorithms under the Gradient Boosting framework. XGBoost provides a parallel tree boosting (also known as GBDT, GBM) that solve many data science problems in a fast and accurate way. (Tianqi Chen 2021). Contrary to regular Gradient Boosting, XGBoost uses its own method of building trees where the Similarity Score and Gain determine the best node splits (Dobilas 2021).

For our 10,000 record dataset, the results are the following:

Accuracy: (Recall): 0.79

Complexity: Extreme Gradient Boosting algorithms are similar to Gradient Boosting algorithms but are more complex given the increased amount of parameters to optimize. The best parameters in our fit are:

colsample_bytree: 0.5
gamma: 1
learning_rate: 0.001
max_depth: 4
min_child_weight: 1
n_estimators: 50
reg_alpha: 1
reg_lambda: 1
subsample: 0.75

Interpretability: Using the .best_estimator_ function included in the GridSearchCV, we can observe that the relative weight importance is quite different from the previous two models, with *NumOfProducts* as the top variable, followed by *IsActiveMember* and *Age*.

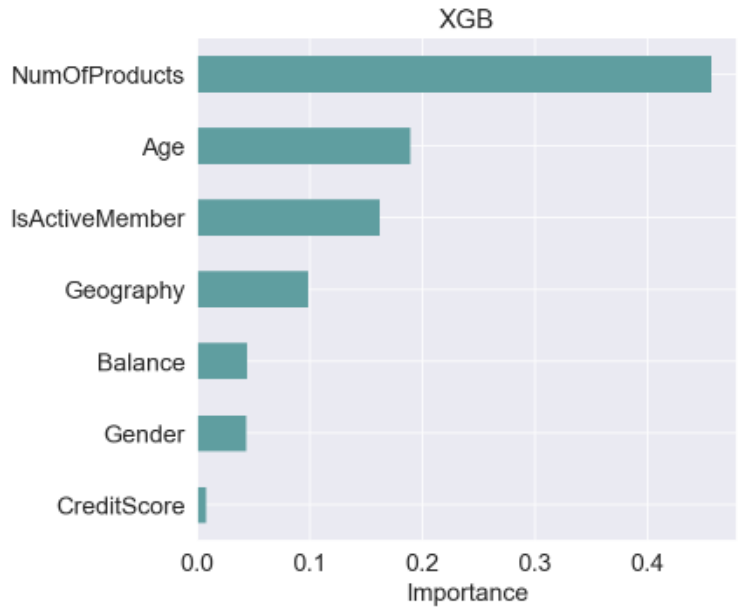

Figure 18 Feature Importance - Extreme Gradient Booster

Time to fit model in train set: 0:00:47.25. The time is very similar to Gradient Booster.

3.9 Light Gradient Boosting Machine Classifier results

The next model fitted is a Light Gradient Boosting Machine Classifier, using the lightGBM's LGBMClassifier. LightGBM is a gradient boosting framework that uses tree-based learning algorithms. It is designed to be distributed and efficient with the following advantages:

- Faster training speed and higher efficiency.
- Lower memory usage.
- Better accuracy.
- Support of parallel, distributed, and GPU learning.

- Capable of handling large-scale data. (Ke 2022)

For our 10,000 record dataset, the results are the following:

Accuracy: (Recall): 0.78

Complexity: Light Gradient Boosting algorithms are very similar to Extreme Gradient Boosting algorithms in terms of complexity. The best parameters in our fit are:

feature_fraction: 0.5
learning_rate: 0.01
max_depth: 5
min_child_samples: 10
n_estimators: 200
num_leaves: 10
reg_alpha: 0.1
reg_lambda: 0.5

Interpretability: Using the .best_estimator_ function included in the GridSearchCV, we can observe in Figure 19 that the relative weight importance is very similar to in Random Forest in the top three variables (Age, NumOfProducts and Balance), inverting the order of *CreditScore* and *IsActive Member*. It is interesting how this change improved predictive performance marginally.

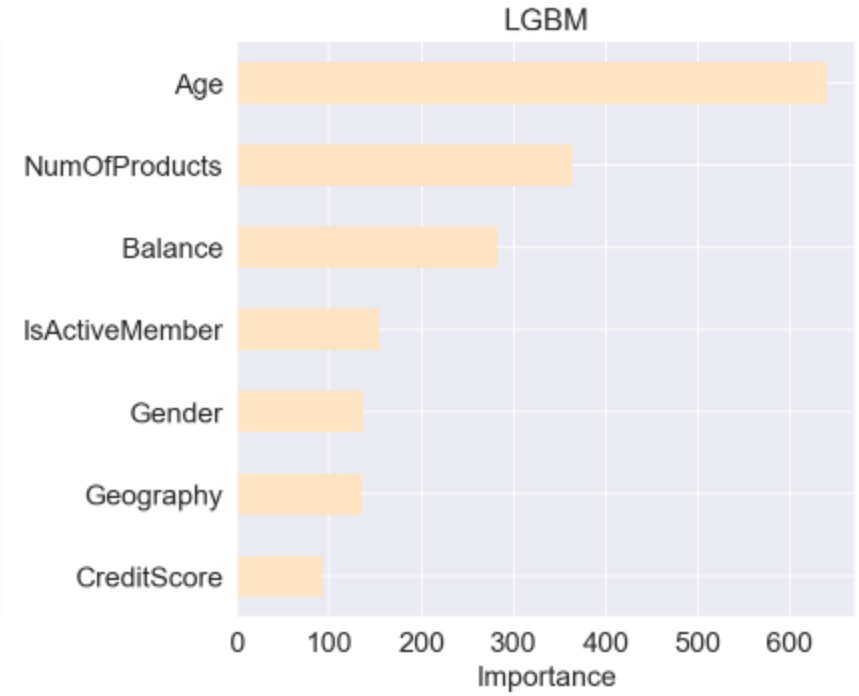

Figure 19Feature Importance - Light Gradient Booster

Time to fit model in train set: 0:00:07.26, which is a fraction of what it took to fit the X GBM. Thus we can confirm that LGBM is far more efficient than the other Gradient Boost algori thms.

3.10 Results Summary

Results for the whole 10,000 observations population.

Predictive Performance

For simplicity purposes, the main performance metric used throughout the report has been Recall, but a broader view of other metrics allows us to identify some remarkable insights. On Recall alone, the best performing model is XGBMC, but looking at other metrics (particularly AUC), the best performing model is LGBMC (Table 2).

Model	Accuracy	Precision	Recall	AUC
Logistic Regression	0.68	0.67	0.73	0.73
Support Vector Machine Classifier	0.71	0.72	0.69	0.77
Random Forest Classifier	0.8	0.81	0.77	0.88
Gradient Booster Machine Classifier	0.79	0.81	0.75	0.88
Xtreme Gradient Boosting Machine Classifier	0.79	0.8	0.78	0.87
Light Gradient Boosting Machine Classifier	0.81	0.82	0.78	0.89

Table 2 Predictive Performance across models

Looking at the ROC curves in Fig. 20, we can see how the Tree-based models perform similarly, while the Logistic Regression and Support Vector do a poorer job.

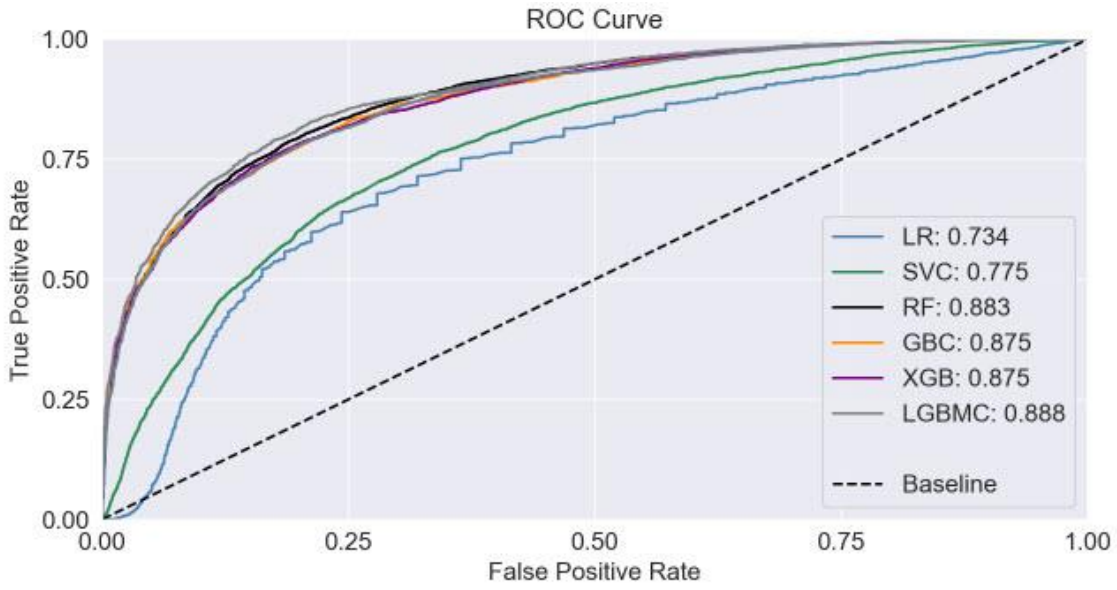

Figure 20 - ROC Curves

Finally, when fitting the models into the Test population, we can observe in Table 3 that the best performing model is XGBMC. Still, the difference is minimal compared to the other algorithms. Worth noting is that the performance gap between Logitistical Regression and the tree-based models is now marginal, whereas it was pretty significant in the Train set. These results suggest overfitting from the tree-based models.

	Accuracy	Precision	Recall
LR	0.65	0.33	0.77
SVC	0.73	0.40	0.73
RF	0.81	0.51	0.77
GB	0.81	0.52	0.76
XGB	0.80	0.50	0.78
LGBM	0.82	0.53	0.77

Table 3 - Prediction metrics for Test set

How long does it take to make predictions

Using the model fitting time (run-time) in Python, we can see four classes of performance: Logistic Regression takes the shortest time at 0.68 seconds, followed by LGBMC with 7.26 seconds. RFC, GBMC and XGBMC have similar running times, with SVC taking the longest at 3'57.35", which is 300% greater than the average running time.

Model	Time	Recall
Logistic Regression	00:00.68	0.73
Support Vector Machine Classifier	03:57.35	0.69
Random Forest Classifier	00:27.67	0.77
Gradient Booster Machine Classifier	00:36.65	0.75
Xtreme Gradient Boosting Machine Classifier	00:47.25	0.78
Light Gradient Boosting Machine Classifier	00:07.26	0.78

Table 4 Model running time and Recall

Interpretability

There are noteworthy findings when comparing the variables' importance versus recall: *CreditScore* isn't, on its own, a good predictor of attrition. Conversely, the best-performing models have *Age* and *NumOfProducts* as the most significant variables, as can be observed in Table 5. These insights are very relevant, as they may suggest areas to build retention strategies around.

Variable	LR	SVCM	RFC	GBMC	XGBMC	LGBMC
CreditScore	0.040	0.070	0.020	0.002	0.009	92
Gender	0.500	0.570	0.030	0.019	0.043	136
Geography	0.830	0.730	0.060	0.041	0.098	134
Balance	0.130	0.130	0.090	0.042	0.044	282
IsActiveMember	1.220	0.870	0.090	0.091	0.162	154
NumOfProducts	0.140	0.190	0.280	0.270	0.456	363
Age	0.870	0.790	0.430	0.535	0.189	639
Recall	0.73	0.69	0.77	0.75	0.78	0.78

Table 5 Feature importance by Model + Recall

Results for the 1,000 observations segment.

For practical purposes, the researcher only includes the summary tables and commentary on comparing the results between both segments. The details are included in the Appendix, in the "EDA for 1k population" section.

Predictive Performance

Comparing prediction performance metrics Recall and AUC between both Train set segments, we can see that the tree-based models performed better in the smaller sample than in the larger population (Table 6). Worth calling out is how Logistic Regression's recall was reasonably consistent between both segments.

Train Set	Recall		AUC	
Model	10,000	1,000	10,000	1,000
Logistic Regression	0.73	0.73	0.73	0.78
Support Vector Machine Classifier	0.69	0.7	0.77	0.78
Random Forest Classifier	0.77	0.83	0.88	0.91
Gradient Booster Machine Classifier	0.75	0.83	0.88	0.9
Xtreme Gradient Boosting Machine Classifier	0.78	0.83	0.87	0.9
Light Gradient Boosting Machine Classifier	0.78	0.82	0.89	0.9

Table 6 Prediction performance comparison

The performance in the Test set shows some fascinating findings (Table 7). The recall dropped significantly, particularly in the tree-based models. This performance decline can be attributed to overfitting, which became more extreme with a smaller population. The biggest

surprise was the improved performance of the SVCM, which has been a consistent underperformer in the Train set.

Test set / Recall	10,000	1,000
Logistic Regression	0.77	0.69
Support Vector Machine Classifier	0.73	0.75
Random Forest Classifier	0.77	0.67
Gradient Booster Machine Classifier	0.76	0.72
Xtreme Gradient Boosting Machine Classifier	0.78	0.67
Light Gradient Boosting Machine Classifier	0.77	0.64

Table 7 Recall performance in the Test set between 10k and 1k segments

How long does it take to make predictions

There were notable results when comparing run-time performance across both segments (Table 8): the average run time went from 59.48" to 7.55", an 87% reduction. It is close to the observed decrease in the number of records, but not quite. The most impressive reduction was for the SVMC, which went down from almost 4 minutes to just 1.5 seconds. An unexpected finding was that the LGBMC went up by an astonishing 72%.

Model / Time	10,000		1,000	
	Time	Recall	Time	Recall
Logistic Regression	00:00.68	0.73	00:00.007	0.73
Support Vector Machine Classifier	03:57.35	0.69	00:01.5	0.7
Random Forest Classifier	00:27.67	0.77	00:07.2	0.83
Gradient Booster Machine Classifier	00:36.65	0.75	00:06.2	0.83
Xtreme Gradient Boosting Machine Classifier	00:47.25	0.78	00:18.0	0.83
Light Gradient Boosting Machine Classifier	00:07.26	0.78	00:12.4	0.82

Table 8 Run time and Recall for 10k and 1k segments

Interpretability

When comparing Table 5 above with Table 9 below, the findings are consistent regarding the most important variables: *Age* and *NumOfProducts* variables are still the most relevant features in the best performing models. Conversely, *CreditScore* and *Gender* are the least valuable variables, consistently across the board.

Variable	LR	SVMC	RFC	GBMC	XGBMC	LGBMC
CreditScore	0.04	0.07	0.09	0.04	0.03	274
Gender	0.50	0.52	0.03	0.03	0.05	58
Geography	0.83	0.95	0.06	0.06	0.15	160
Balance	0.13	0.03	0.13	0.11	0.06	315
IsActiveMember	1.22	0.94	0.09	0.12	0.23	176
NumOfProducts	0.14	0.14	0.24	0.25	0.30	363
Age	0.87	0.87	0.36	0.39	0.19	454
Recall	0.73	0.7	0.83	0.83	0.83	0.82

Table 9 Variable importance by model - 1k segment

4 Conclusions

In the Introduction, the stated objective of this research was to evaluate several classifiers in two datasets. The evaluation was done on accuracy, interpretability, complexity, scalability, and run-time. And fundamentally, how well would the approach meet the business goal of predicting customer attrition to build retention strategies.

On accuracy, the Tree-based algorithms (Random Forest Classifier, Gradient Booster Machine Classifier, Xtreme Gradient Booster Machine Classifier, and Light Gradient Boosting Machine Classifier) had better prediction performance in the Train set versus Logistic Regression and Support Vector Machine Classifier models. This finding was consistent across the 10,000 record population and the 1,000 record segment. However, when fitting those models into the Test test, the performance wasn't consistent between the 10k population and the 1k segment. In the 10k population, Logistic Regression performed on par with the Tree-based models with a recall score of 0.77 (the average recall score of the Tree-based models is 0.77). The SVCM had a recall score of 0.73. In the 1k subset, the Logistic Regression model performed comparably to the Tree-based models (0.69 Recall vs. an average of 0.675, respectively). Surprisingly, the best performing model was the Support Vector Machine Classifier, with a recall

score of 0.75. This phenomenon could be due to a statistical anomaly given the small size of the Test set for the 1k subset, just 200 observations.

On interpretability, the findings in terms of the most valuable features (*Age* and *NumOfProducts*) were consistent across both datasets and highlighted the main differences between the Tree-based models and the rest.

On run-time, the number of observations had a direct impact on performance: the average running time between the 10k population and the 1k sample had an 87% reduction. Looking at the results per model, Logistic Regression was the fastest and had the second highest accuracy performance in the Test set and thus is a top contender. Running time may be a challenge for scalability, as RFC, GBMC, and XGBMC took an average of 37 sec to run on the 10k sample (10.4 sec in the 1k sample). To ensure the solution provides the expected outcome, project owners would need to closely evaluate the operational infrastructure and model choice for datasets with millions of records and real-time applications.

Based on the findings, it is worth exploring not only more sophisticated Tree-based models or SVMCs but even the more basic ones like Logistic Regression. This insight holds when considering the model running time, as Logistic Regression ran significantly quicker than the other models. Logistic Regression is the best solution for real-time applications such as online offers, given the high recall and shortest running times. For batch-processing campaigns (like outbound mail or email offers), processing time is not as relevant, and thus accuracy would remain the main focus. For this use case, the best performing model in the Test set was XGBMC, followed closely by LGBMC. Given the 85% shorter running time and its high performance, LGBMC might be the best suitable model, given its efficiency.

In summary, using any of these models would help predict potential attrition and build retention strategies based on the model's predictive power and the insights derived from the interpretabilty study. Depending on the dataset and business constraints, some will work better than others, so we advocate including Tree-based and non-Tree-based models as part of the modeling process.

5 Future directions

Given the findings in this research, the researcher suggests more analysis in several areas. First, it is critical to understand better the relationship in running time between the 10,000 observation population and the 1,000 sub-set: in this experiment, the average running time increased 87% while the number of observations increased 900%. If this ratio were to hold, it would have severe implications for scalability. Therefore, the researcher calls for more ad-hoc experiments with larger datasets. These datasets should include not only more trials, but also more features, to understand the marginal impact of either. It will also be worth exploring different hardware configurations, both on-premise processing and cloud services. The scalability of using time-intensive models for this application will depend on finding the answers to the questions around run-time drivers. Then there is the matter of improving the predictive power of the models. Acknowledging that this is a limited dataset, it would be a worthwhile goal to identify a richer dataset and perform the experiment, as an improved recall would have a significant impact on the economics of any retention strategy. And finally, the researcher recommends designing scientifically-sound experiments contrasting the results of different retention strategies to identify the most effective and optimize the return of the investment dollars.

6 Code referenced in this document

The code for this research can be found at https://github.com/MrLWhite01

7 Acknowledgments

Special thanks to James Thomas and Sarthak Dighe for their insightful feedback. This report is more robust and clearer thanks to their observations, their time was invaluable.

8 Appendix

EDA for 1k population

The objective of this EDA is to compare the 1k segment to the 10k segment to validate it was a random selection and that the underlying assumptions in the experiment still hold. While the full EDA is included in the code, for brevity purposes the researcher only focuses on the most relevant variables. Table 5 shows the same descriptive statistics shown in Table 1 and Table 2 compares, in percentage terms, the values between the two. The differences are marginal and thus we can conclude that the selection was random and the 1k segment is a subset of the 10K population.

	CreditScore	Age	Tenure	Balance	NumOfProducts	HasCrCard	IsActiveMember	EstimatedSalary	Exited
count	1000.00	1000.00	1000.00	1000.00	1000.00	1000.00	1000.00	1000.00	1000.00
mean	651.27	38.98	5.03	75338.82	1.54	0.67	0.51	101444.09	0.21
std	97.71	10.42	2.83	62928.36	0.60	0.47	0.50	56911.71	0.41
min	359.00	18.00	0.00	0.00	1.00	0.00	0.00	96.27	0.00
25%	583.00	32.00	3.00	0.00	1.00	0.00	0.00	54658.25	0.00
50%	653.50	37.50	5.00	95340.12	1.00	1.00	1.00	101353.49	0.00
75%	717.00	44.00	7.00	128717.40	2.00	1.00	1.00	150251.06	0.00
max	850.00	88.00	10.00	250898.09	4.00	1.00	1.00	199693.84	1.00

Table 10 Descriptive statistics 1k segment

	CreditScore	Age	Tenure	Balance	NumOfProducts	HasCrCard	IsActiveMember	EstimatedSalary	Exited
count	-90%	-90%	-90%	-90%	-90%	-90%	-90%	-90%	-90%
mean	0%	0%	0%	-1%	1%	-6%	-2%	1%	5%
std	1%	-1%	-2%	1%	3%	2%	0%	-1%	2%
min	3%	0%	0%	0%	0%	0%	0%	731%	0%
25%	0%	0%	0%	0%	0%	0%	0%	7%	0%
50%	0%	1%	0%	-2%	0%	0%	0%	1%	0%
75%	0%	0%	0%	1%	0%	0%	0%	1%	0%
max	0%	-4%	0%	0%	0%	0%	0%	0%	0%

Table 11 Descriptive statistics % change of 1k segment versus 10k population

The most relevant categorical variables (*Geography* and *Gender*) show similarities between both populations. From *Geography*, comparing Figure 23 and Figure 24, we can observe a slightly higher attrition rate for *Spain* and *Germany* in the 1k segment than in the 10k population. In *Gender*, *Male* shows a higher churn rate in the 1k segment than in the 10k population.

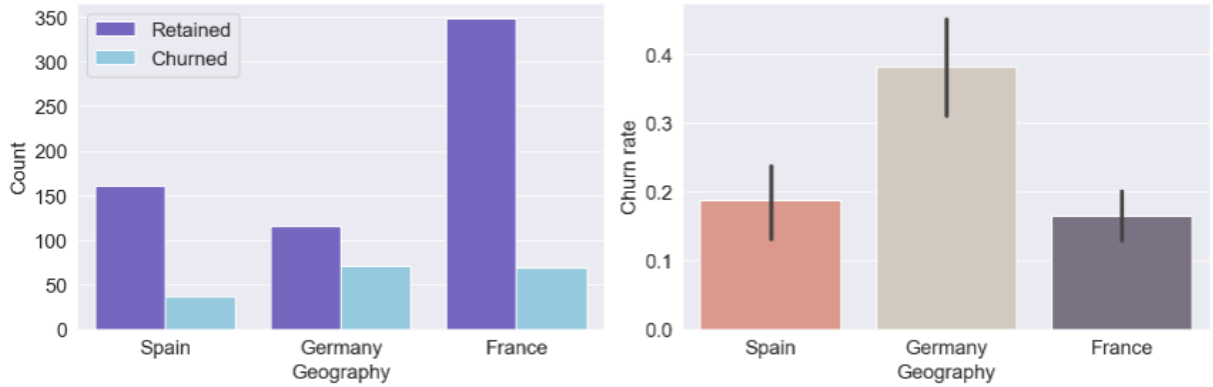

Figure 21 - Geo distribution for 1k segment

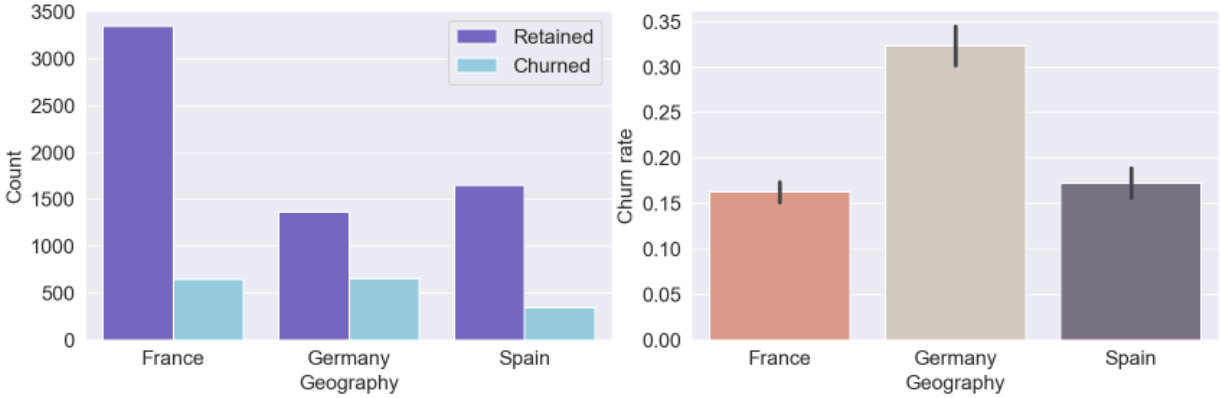

Figure 22 Geo distribution for 10k population

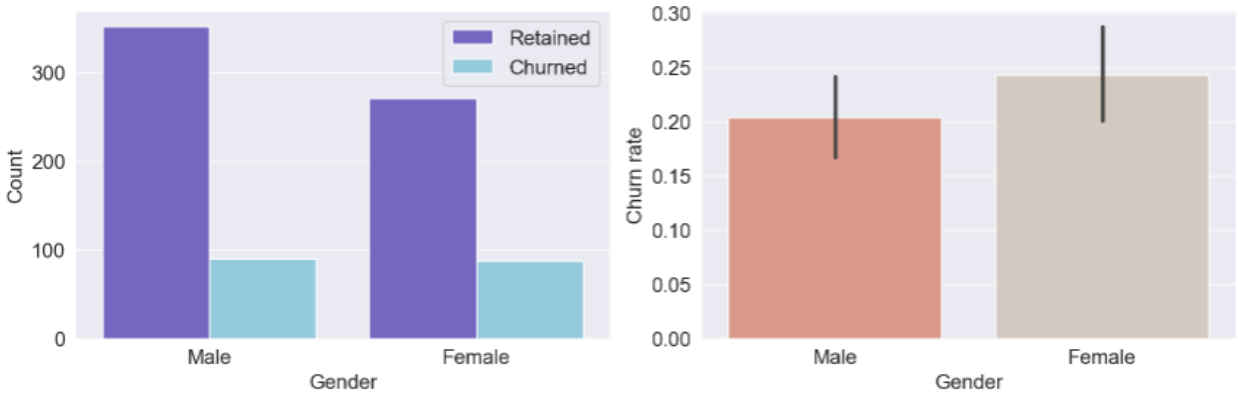

Figure 23 Gender distribution for 1k segment

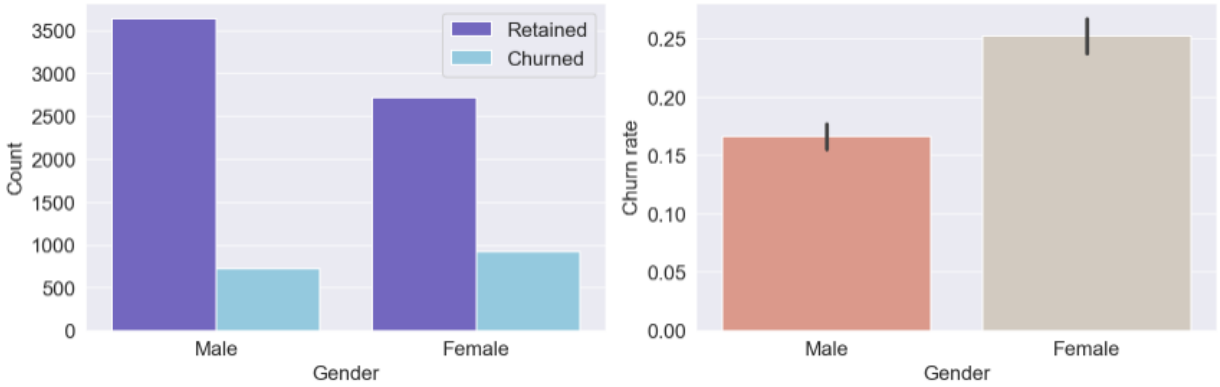

Figure 24 Gender distribution 10k population

9 References

Almaliki, Zaid Alissa. 2019. *Towards Data Science.* Edited by Medium. March 19. Accessed 05 01, 2022. https://towardsdatascience.com/do-you-know-how-to-choose-the-right-machine-learning-algorithm-among-7-different-types-295d0b0c7f60.

Chawla, N. V., K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer. 2002. "SMOTE: Synthetic Minority Over-sampling Technique." *Journal of Artificial Intelligence Research* 16: p. 321-357. doi:https://doi.org/10.1613/jair.953.

Dobilas, Saul. 2021. *Towards Data Science.* Edited by Medium. 03 21. Accessed 05 2022. https://towardsdatascience.com/xgboost-extreme-gradient-boosting-how-to-improve-on-regular-gradient-boosting-5c6acf66c70a.

Eremenko, Kirill and de Ponteves, Hadelin. 2018. *SuperDataScience.* December 03. Accessed 04 2022. https://www.superdatascience.com/pages/deep-learning.

Géron, Aurélien. 2017. *Hands-on machine learning with scikit-learn & tensorflow Concepts, Tools, and Techniques to Build Intelligent Systems.* 1st . p. 7-14 Sebastopol, CA: O'Reilly.

Google Developers. 2020. *Machine Learning Crash Course: Classification: Precision and Recall.* 02 10. Accessed 04 2022. https://developers.google.com/machine-learning/crash-course/classification/precision-and-recall.

Japkowicz, Nathalie. 2002. "The class imbalance problem: A systematic study." *Intelligent Data Analysis* p. 429-449.

Karp, Andrew H. 1998. "Using logistic regression to predict customer." *The Northeast SAS User Group (NESUG)* p. 1-5.

Ke, Guolin, et. al. 2022. *LightGBM's documentation.* Accessed 05 2022. https://lightgbm.readthedocs.io/en/latest/index.html.

Pedregosa, et.al. 2011. "Scikit-learn: Machine Learning in Python." *JMLR 12* p. 2825-2830.

Sabbeh, Sahar F. 2018. "Machine-Learning Techniques for Customer Retention: A Comparative Study." *(IJACSA) International Journal of Advanced Computer Science and Applications* 9 (2) doi: 10.14569/ijacsa.2018.090238.

Saxena, Shruti. 2018. *Medium.com / Precision vs Recall.* 05 11. Accessed 04 2022. https://medium.com/@shrutisaxena0617/precision-vs-recall-386cf9f89488.

Sperandei, Sandro. 2014. "Understanding logistic regression analysis." *Biochem Med* (Croatian Society for Medical Biochemistry and Laboratory Medicine) 12-18. doi:10.11613/bm.2014.003.

Statista Research Department. 2021. *Statista.* 03. Accessed 05 2022. https://www.statista.com/statistics/497485/financial-services-ad-spend-usa/.

Theofilatos A, Chen C, Antoniou C. 2019. "Comparing Machine Learning and Deep Learning Methods for Real-Time Crash Prediction." *Transportation Research Record* (SAGE Publications) 8: 169-178. doi:10.1177/0361198119841571.

Tianqi Chen, et. al. 2021. *dmlc XGBoost.* Accessed 2022. https://xgboost.readthedocs.io/en/stable/index.html#.

Wei Chen, Xiaoshen Xie, et. al. 2017. "A comparative study of logistic model tree, random forest, and classification and regression tree models for spatial prediction of landslide susceptibility." *CATENA* (Elsevier BV) 147-160. doi:10.1016/j.catena.2016.11.032.