BANK CUSTOMER CHURN

EHS Analytics Data Science Challenge

By Soane Mota

April 18, 2019

EHS ANALYTICS DATA SCIENCE CHALLENGE

Challenge Description

Develop a model to help identify which customers are at risk to discontinue their services with a bank.

Use a provided data set that contains details of a bank's customers and the target variable is a binary variable reflecting the fact whether or not the customer left the bank (closed his account).

In 60 minutes present on:

- Any findings or insights I discovered from analyzing the data.
- Factor analysis, e.g. factor importance.
- The methodology you used to develop and validate the model.

UNDERSTANDING CUSTOMER CHURN

Churner is generally defined as a customer who stops using a product or service for a given period of time.

FINANCIAL PERSPECTIVE

Acquiring new customers is way more expensive than retaining existing customers.

CUSTOMER PERSPECTIVE

Usually he/she has been unhappy with the product/service over a period of time before decided to churn.

Earning the loyalty of a customer by delivering remarkable products/services and building retention strategies

INTERNAL PROCESS PERSPECTIVE

The company needs to identify

- Unavoidable churn
- Avoidable churn

LEARNING/GROWTH PERSPECTIVE

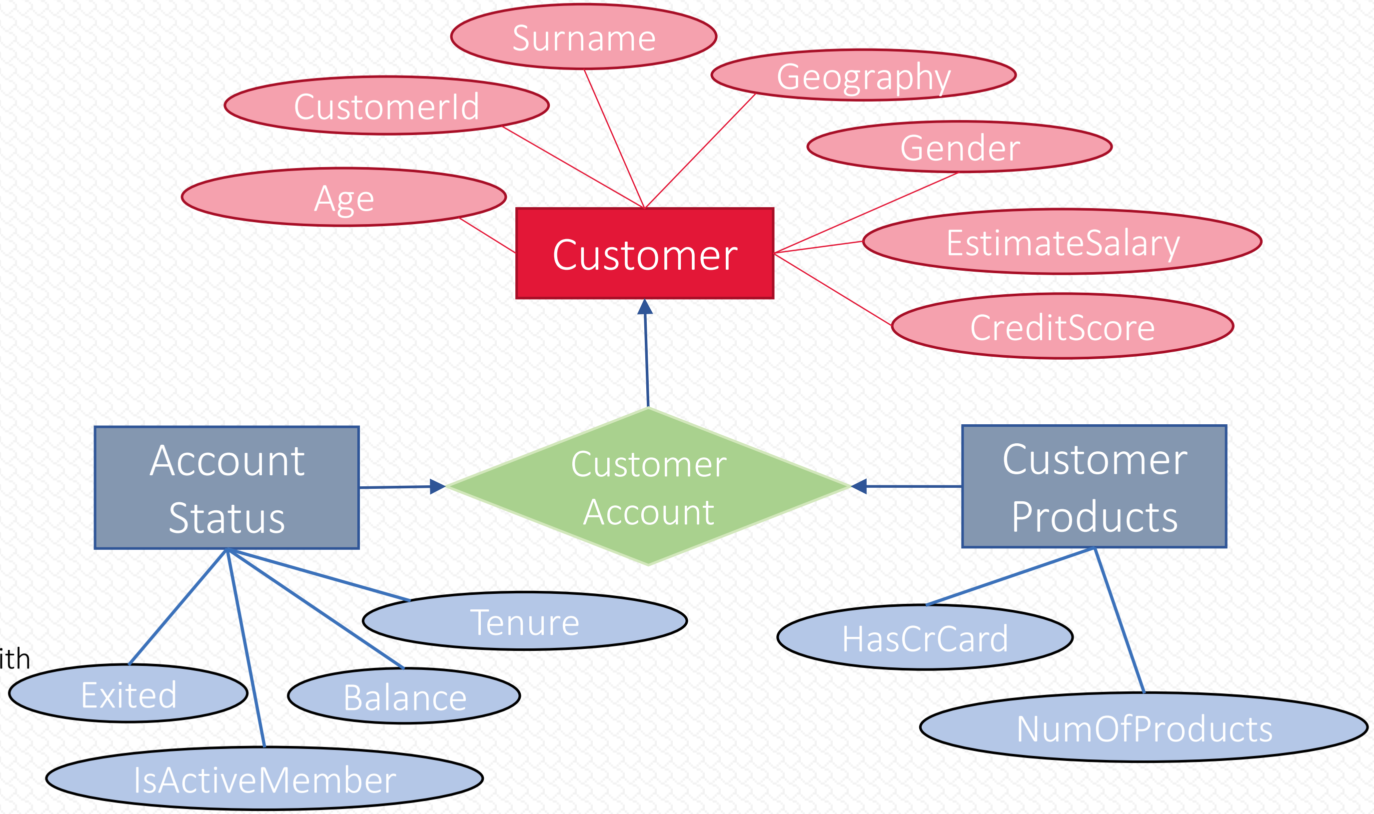
DATA DESCRIPTION

Information of **10,000 clients** of a bank with **no missing data**

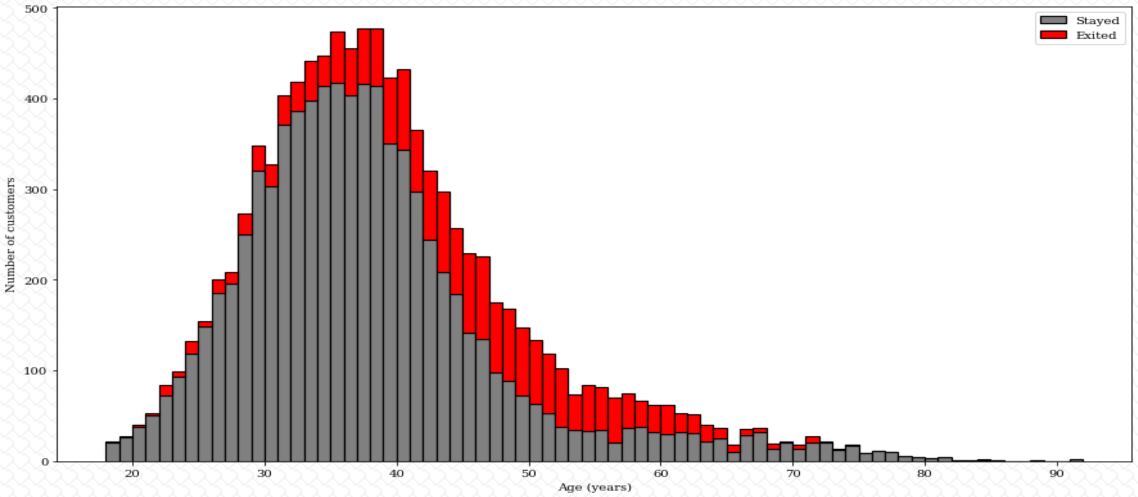
Columns:
● RowNumber: Row Numbers from 1 to 10000

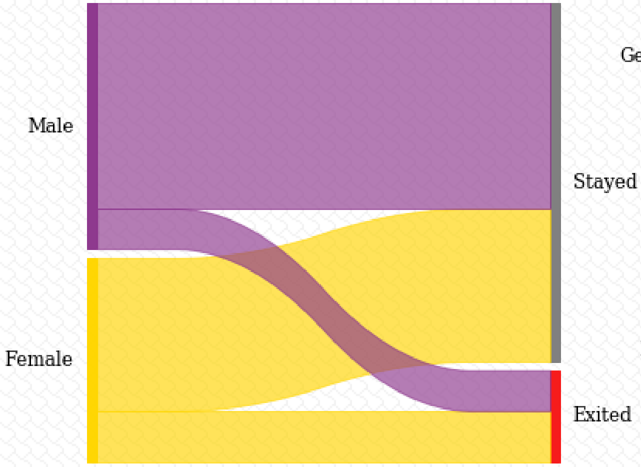
● CustomerId: Unique Ids for bank customer identification
● Surname: Customer's last name
● Geography: The country from which the customer belongs
● Gender: Male or Female
● Age: Age of the customer
● EstimatedSalary: Estimated salary of the customer in Dollars
● CreditScore: Credit score of the customer

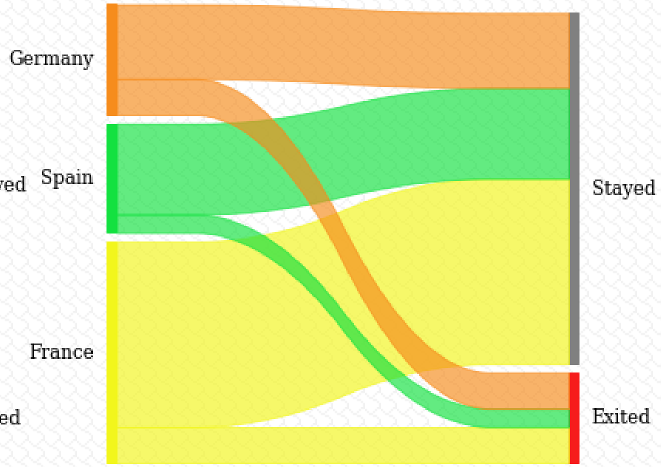
● Tenure: Number of years for which the customer has been with the bank
● Balance: Bank balance of the customer
● NumOfProducts: Number of bank products the customer is utilizing
● HasCrCard: Binary Flag for whether the customer holds a credit card with the bank or not
● IsActiveMember: Binary Flag for whether the customer is an active member with the bank or not
● Exited: Binary flag 1 if the customer closed account with bank and 0 if the customer is retained



DATA DESCRIPTION







79.6%

Stayed

20.4%

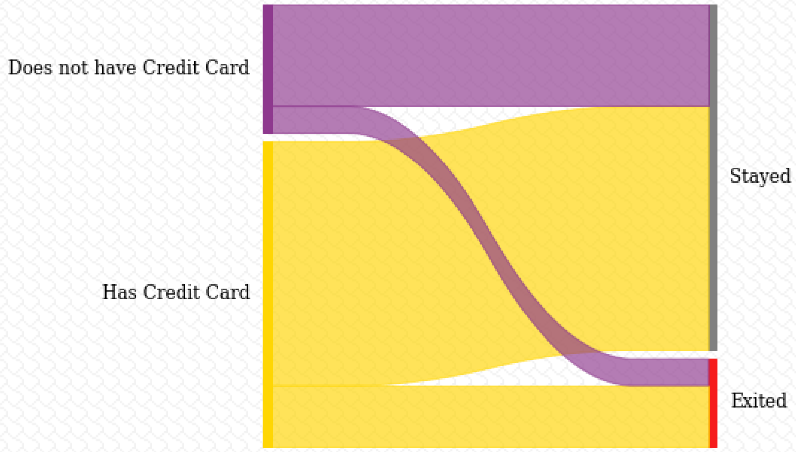
Exited

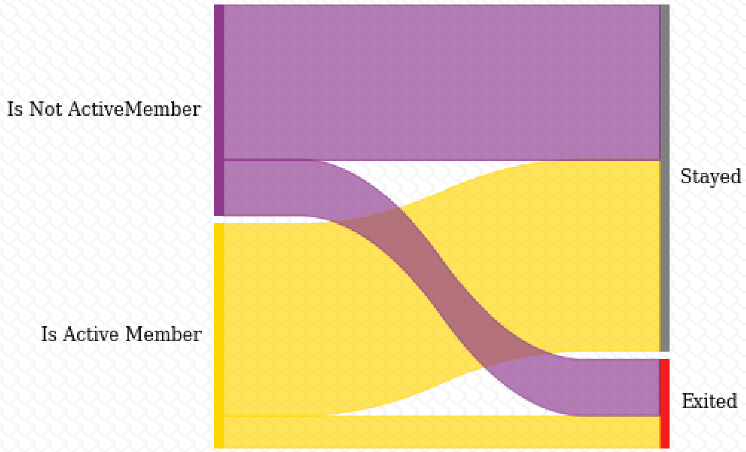
1. As for gender, women are lower in number than the men, but have a higher rate to close the account.

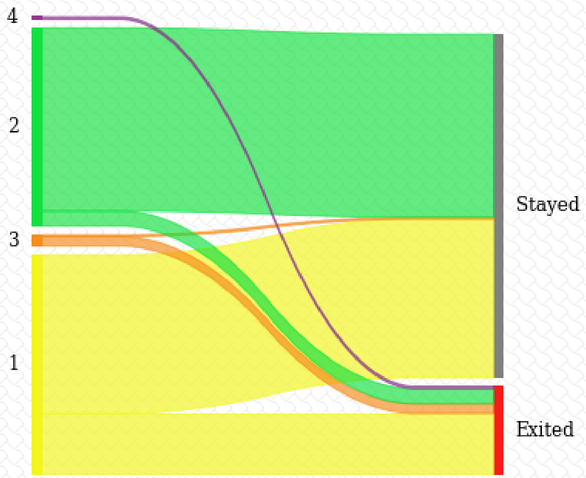
2. There is a higher rate of exited clients in Germany (32%, which is about 2x higher), and lower in Spain and France (around 16% each).

3. On age, customer bellow 40 and above 65 years old have a tendency to keep their account.

DATA DESCRIPTION





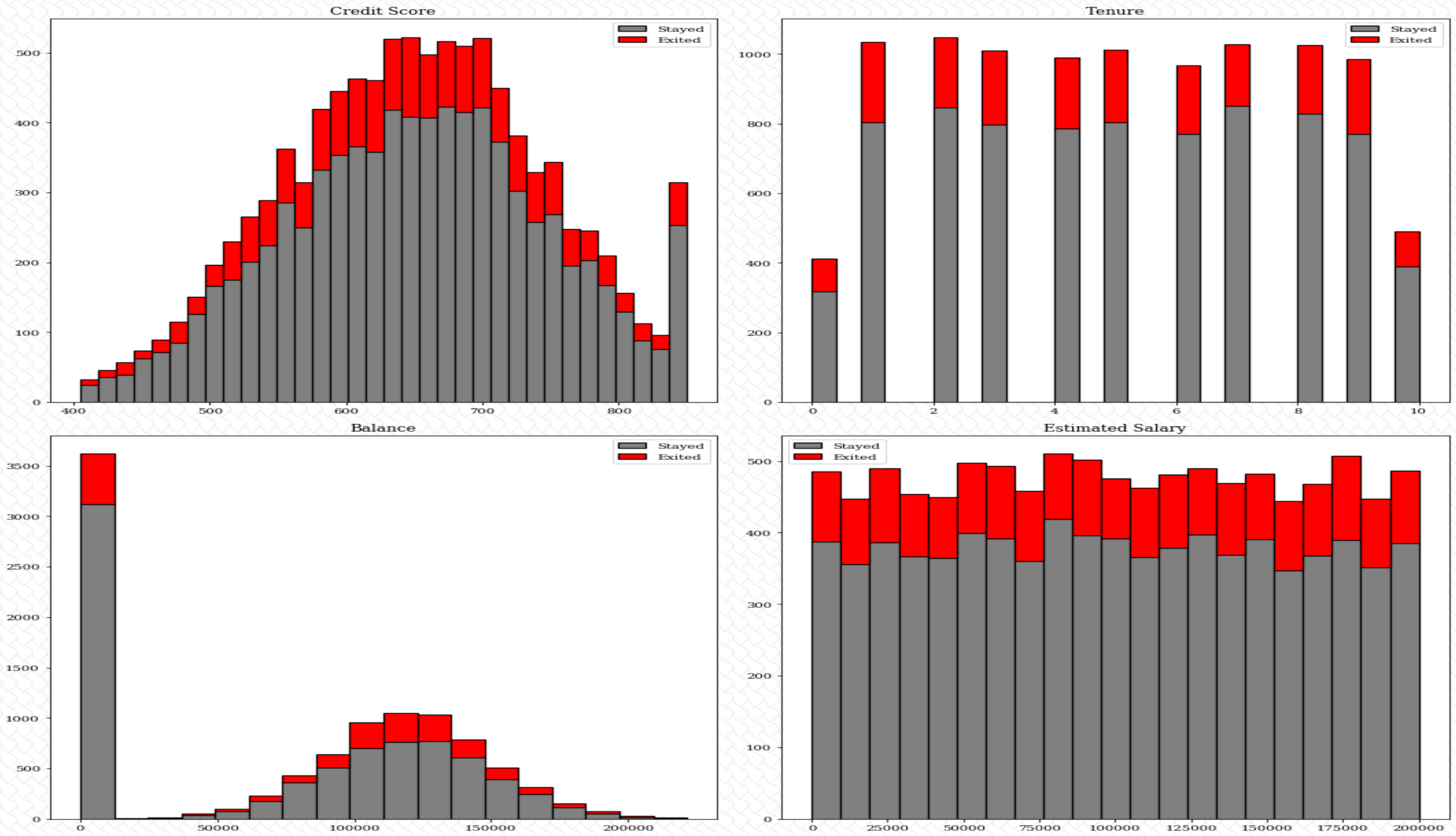


4. Has or not credit card does not impact on the decision to stay in the bank (both groups has 20% of exited customers)

5. Non active members tend to discontinue their services with a bank compared with the active clients (27% vs 14%).

6. The dataset has 96% of clients with 1 or 2 product, and customers with 1 product only have a higher rate to close the account than those with 2 products (around 3x higher).

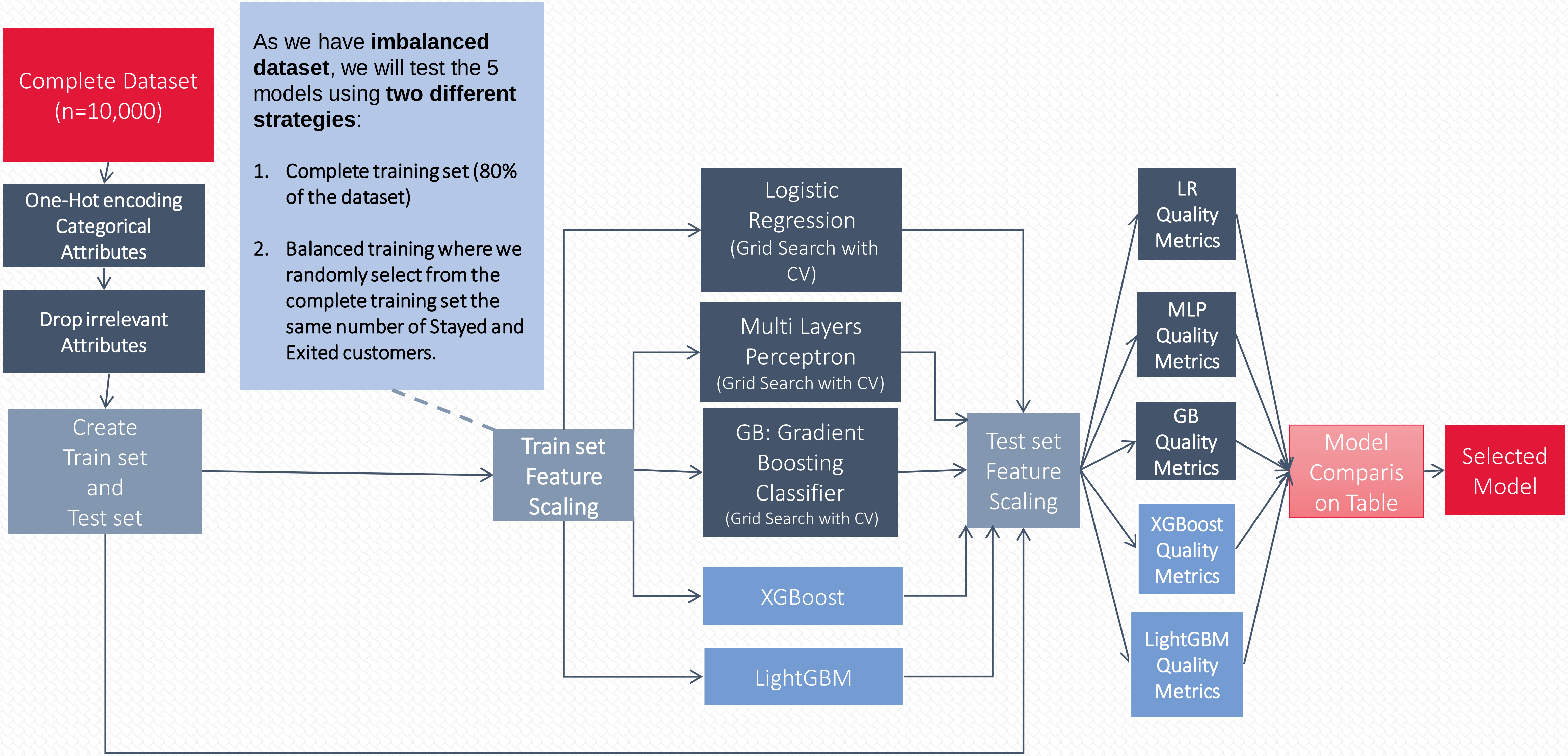
DATA DESCRIPTION



6. Estimated Salary does not seem to affect the churn rate

7. The other metrics is hard to say

Prediction Methodologies



sklearn: GridSearchCV

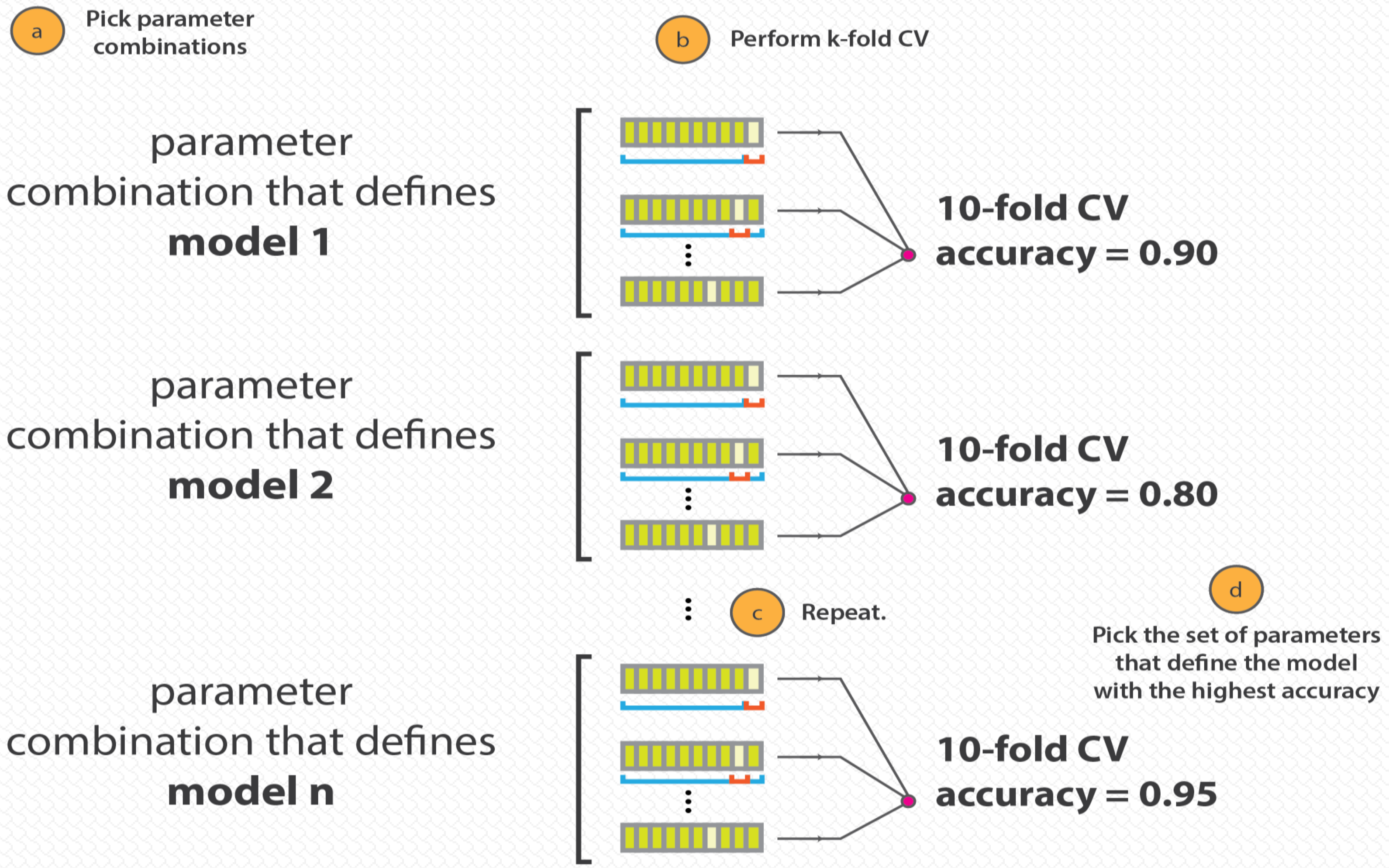


Image Source

sklearn: Logistic Regression

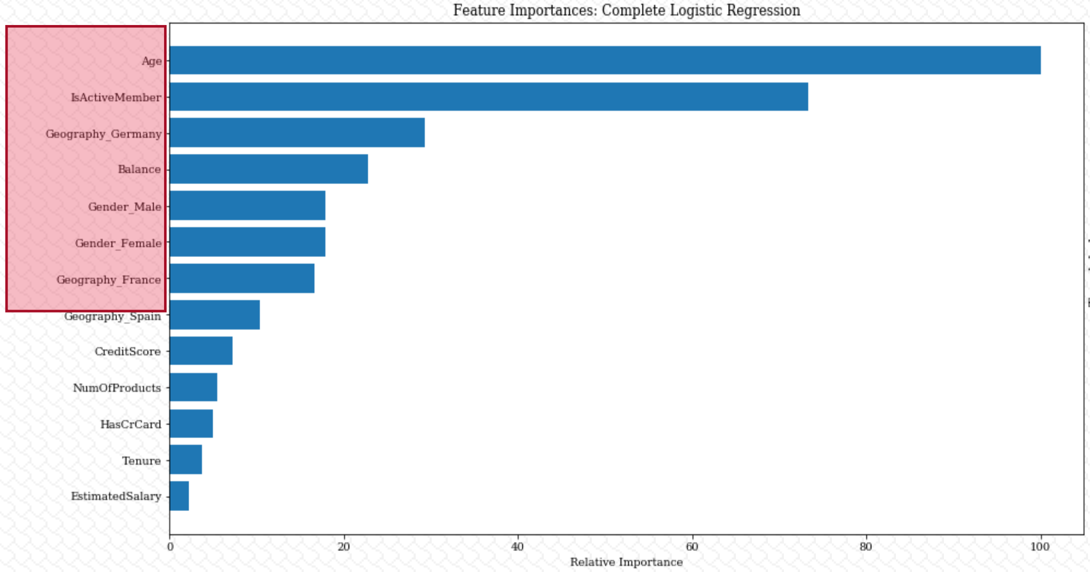
Pros

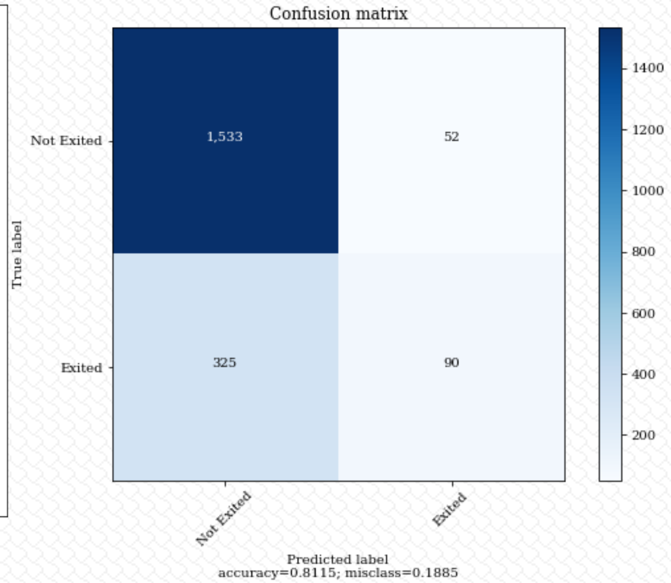
- Fast
- No tuning required
- Highly interpretable
- Well-understood

Cons

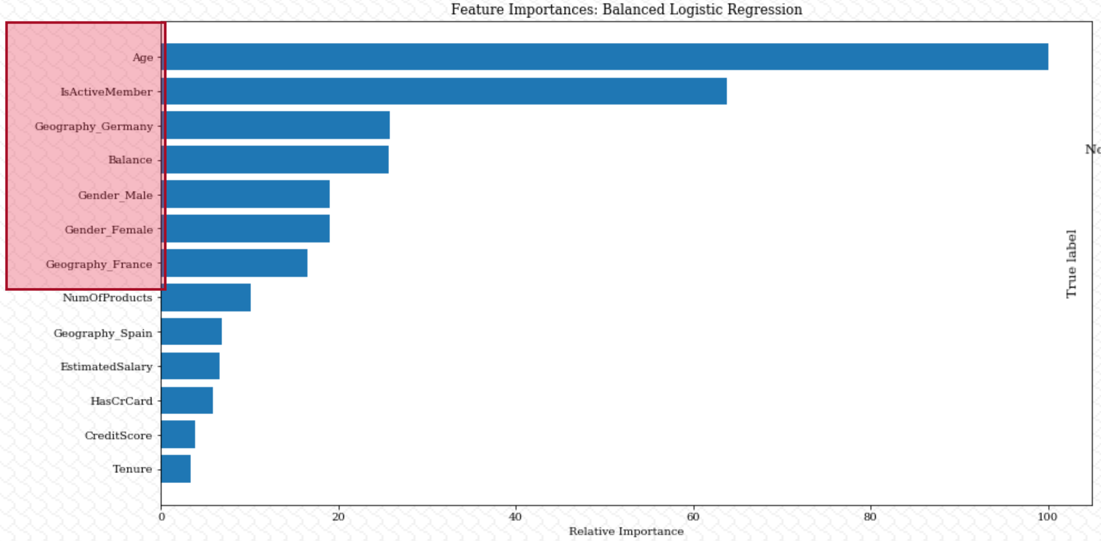
- Unlikely to produce the best predictive accuracy
- Presumes a linear relationship between the features and response
- If the relationship is highly non-linear as with many scenarios, linear relationship will not effectively model the relationship and its prediction would not be accurate

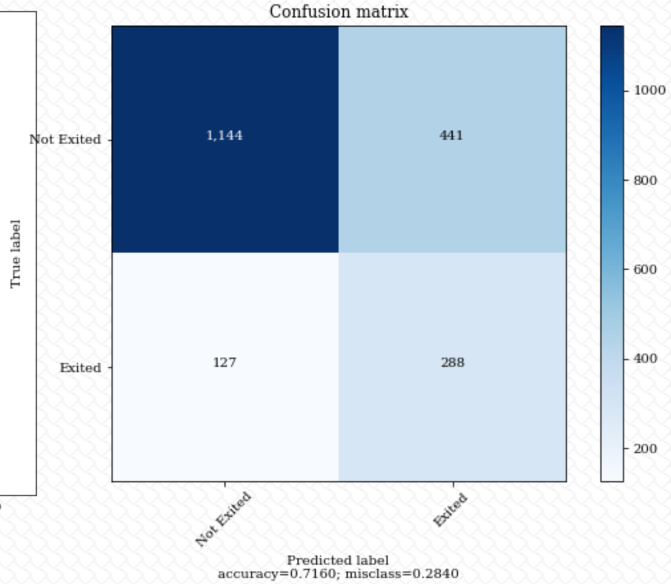
sklearn: Logistic Regression





Accuracy:0.8115
Balanced Accuracy: 0.592
Recall 1: 0.217





Accuracy: 0.716
Balanced Accuracy: 0.708
Recall 1: 0.694

The top Feature importance is more related to demographic information

sklearn: MLP – Multi-Layers Perceptron

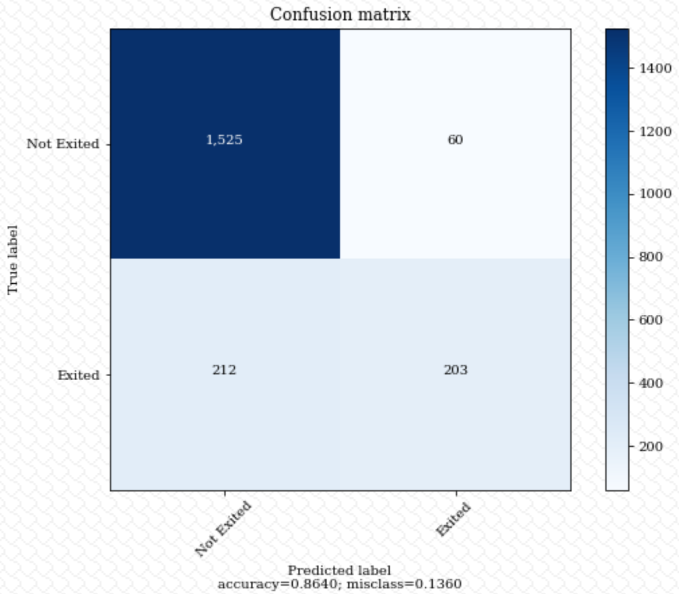
Pros

- Can learn and model complex and non-linear relationships.
- Can generalize well on unseen data. They can work with incomplete data.
- MLP are fault tolerant. An error in one node cannot affect the entire network.

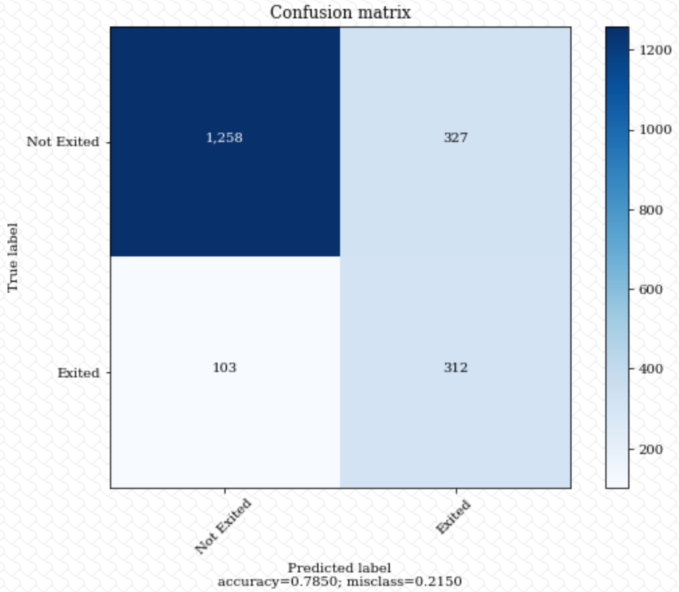
Cons

- They require powerful hardware specifications.
- Difficult to understand how the prediction was arrived at.
- There is no defined guidelines in determining the suitable network structure to use and it all depends on the experience and trial and error.

sklearn: MLP – Multi-Layers Perceptron



Accuracy: 0.864
Balanced Accuracy: 0.726
Recall 1: 0.489



Accuracy: 0.785
Balanced Accuracy: 0.773
Recall 1: 0.752

sklearn: GB - Gradient Boosting Classifier

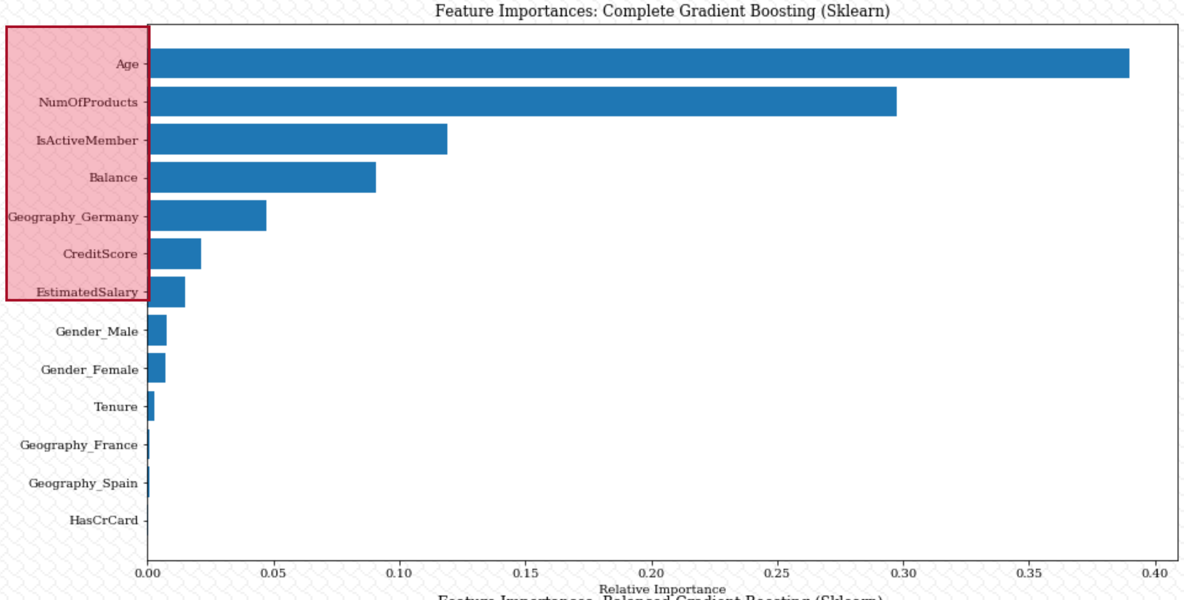
Pros

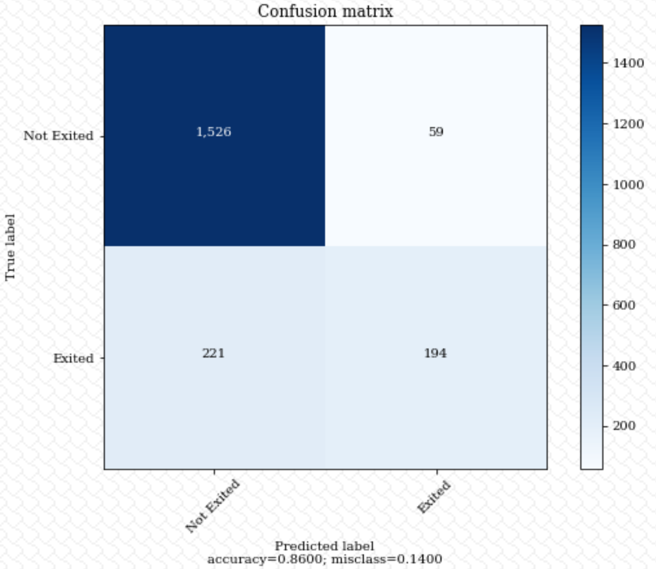
- Often provides predictive accuracy that cannot be beat.
- Lots of flexibility - can optimize on different loss functions and provides several hyperparameter tuning options that make the function fit very flexible.
- No data pre-processing required - often works great with categorical and numerical values as is.
- Handles missing data - imputation not required.

Cons

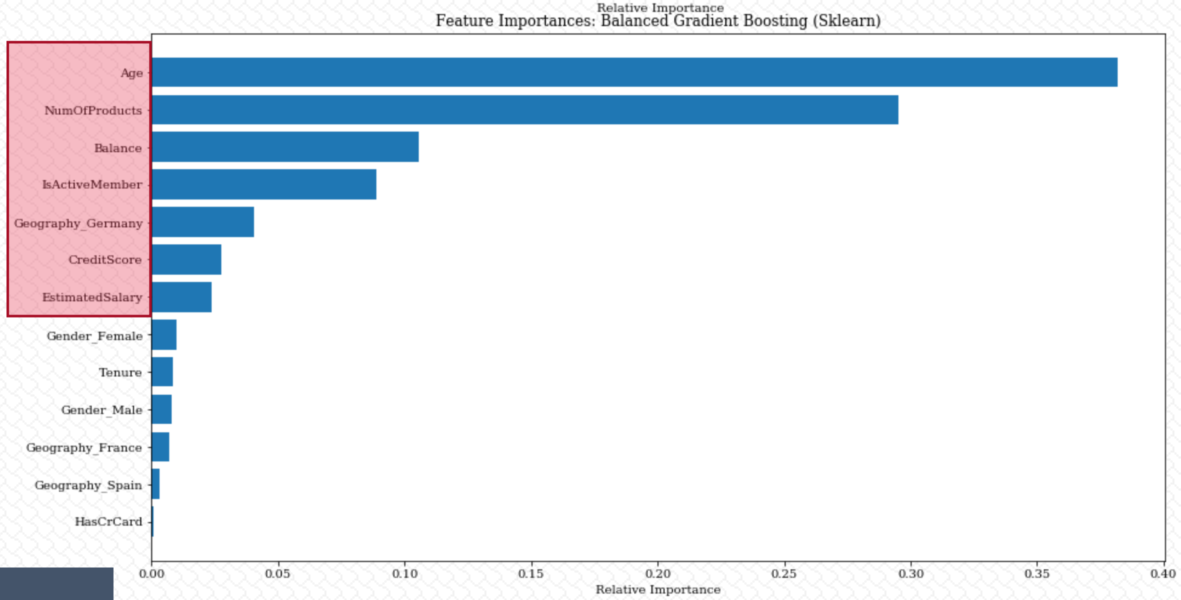
- GBMs will continue improving to minimize all errors. This can overemphasize outliers and cause overfitting. Must use cross-validation to neutralize.
- Computationally expensive - GBMs often require many trees (>1000) which can be time and memory exhaustive.
- The high flexibility results in many parameters that interact and influence heavily the behavior of the approach (number of iterations, tree depth, regularization parameters, etc.). This requires a large grid search during tuning.
- Less interpretable although this is easily addressed with various tools (variable importance, partial dependence plots, etc.).

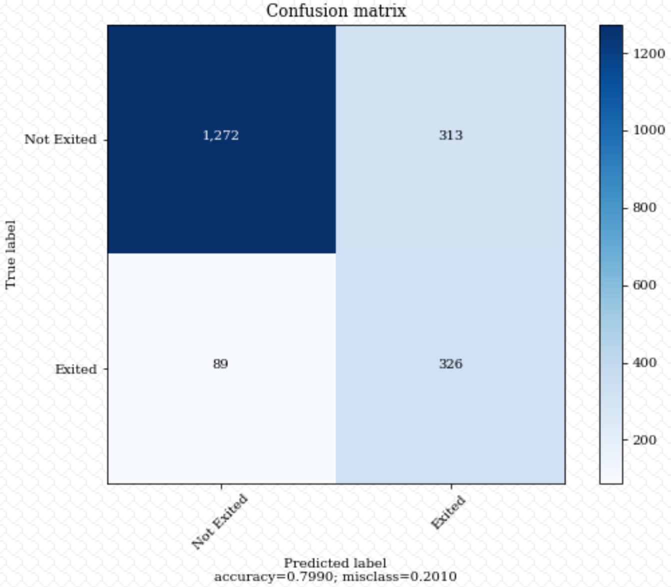
sklearn: GB - Gradient Boosting Classifier





Accuracy: 0.860
Balanced Accuracy: 0.715
Recall 1: 0.467





Accuracy: 0.799
Balanced Accuracy: 0.794
Recall 1: 0.786

The top Feature importance is more related to the customer account updates

XGBoost: XGB – Extreme Gradient Boosting

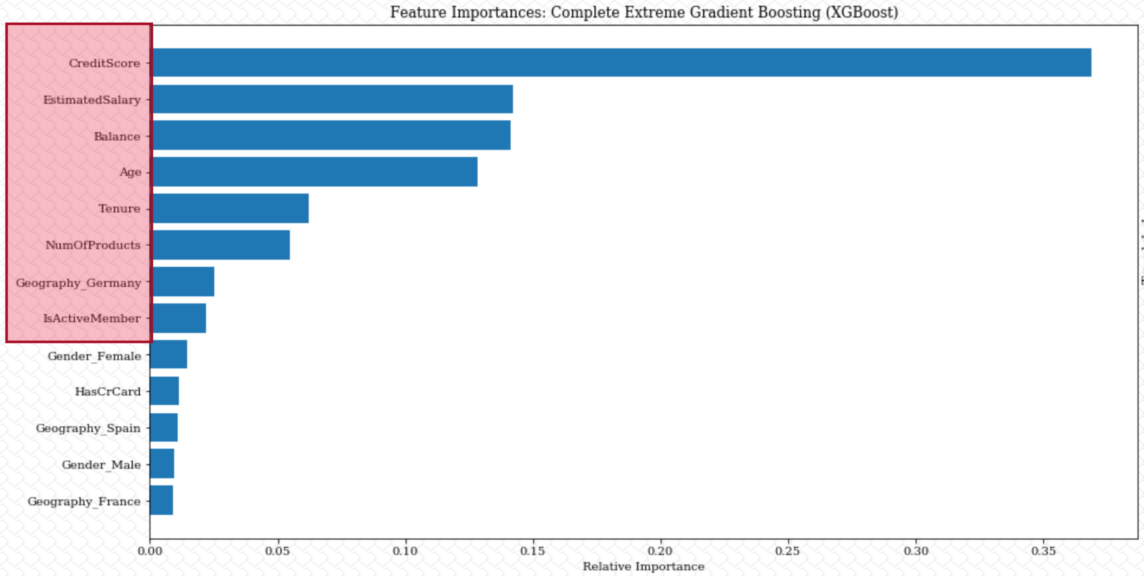
Pros

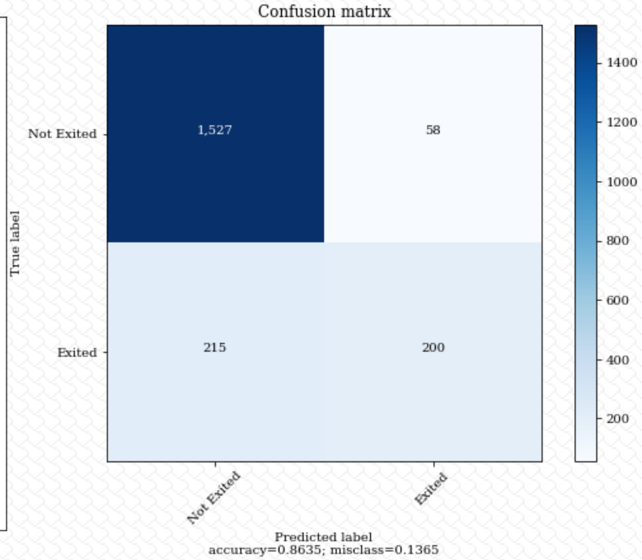
- Fast
- Produces highly accurate models as a result of multiple decision trees & regularization
- Very good model training performance
- Users can specify custom optimization objectives and evaluation criteria
- Can run on the GPU

Cons

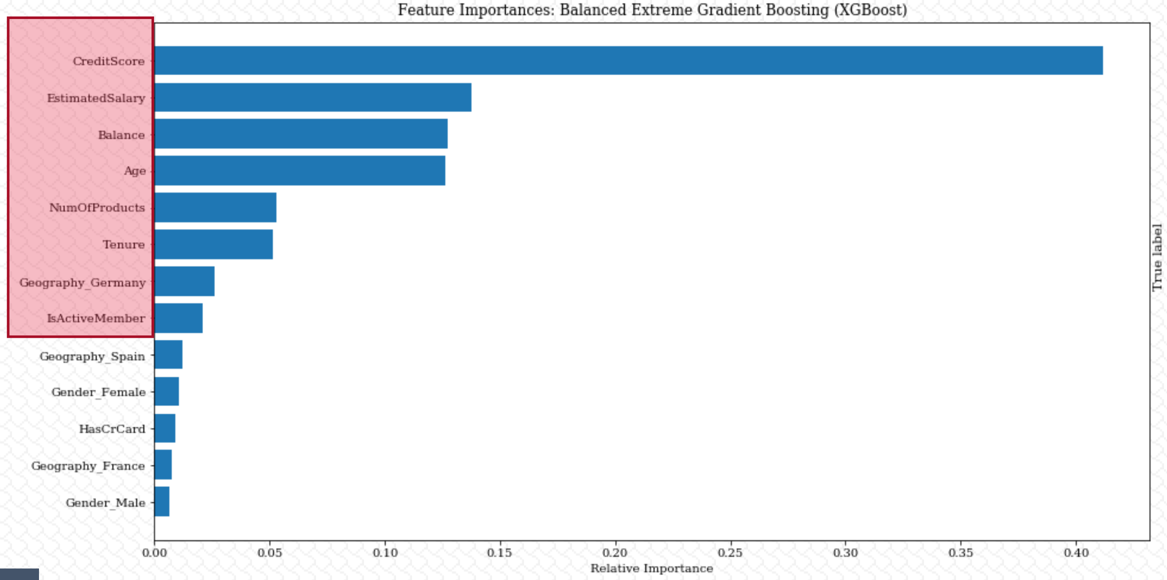
- XGBoost GPU implementation does not scale well to large datasets and ran out of memory often
- Usually slower than the LightGBM

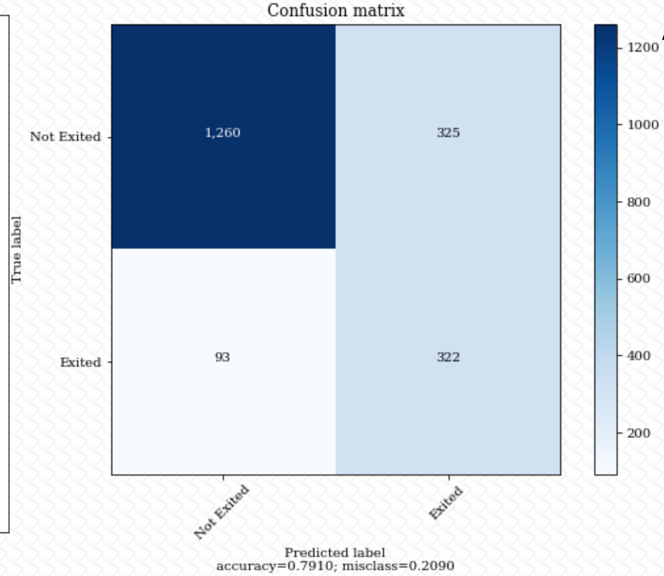
XGBoost: XGB – Extreme Gradient Boosting





Accuracy: 0.864
Balanced Accuracy: 0.723
Recall 1: 0.482





Accuracy: 0.791
Balanced Accuracy: 0.785
Recall 1: 0.776

The top Feature importance is more related to the customer account updates

LightGBM: Light Gradient Boosting Machine

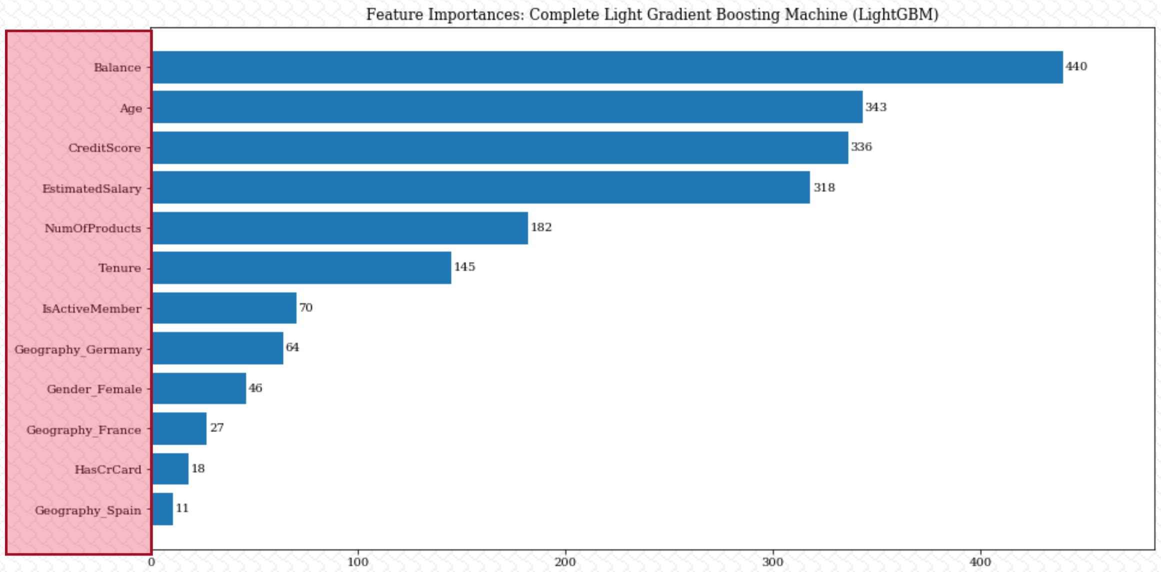
Pros

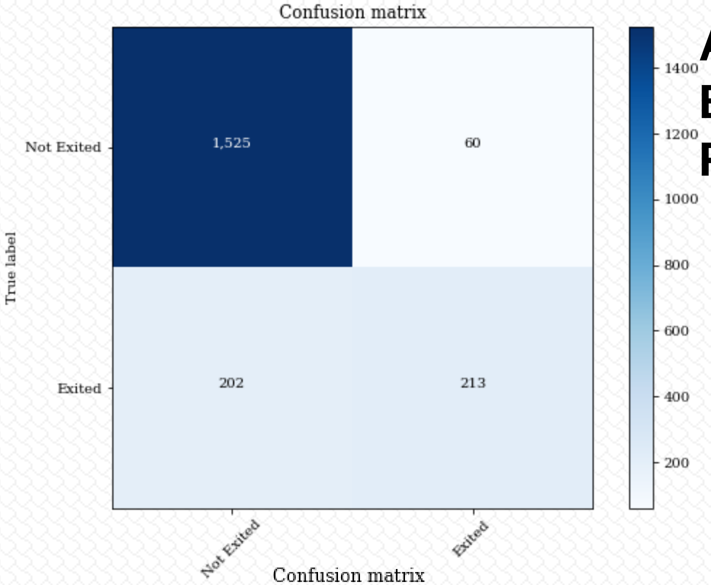
- Fast and accurate
- Can handle categorical features by taking the input of feature names
- XGBoost and LightGBM achieve similar accuracy metrics
- Also runs on GPU
- LightGBM has lower training time than XGBoost

Cons

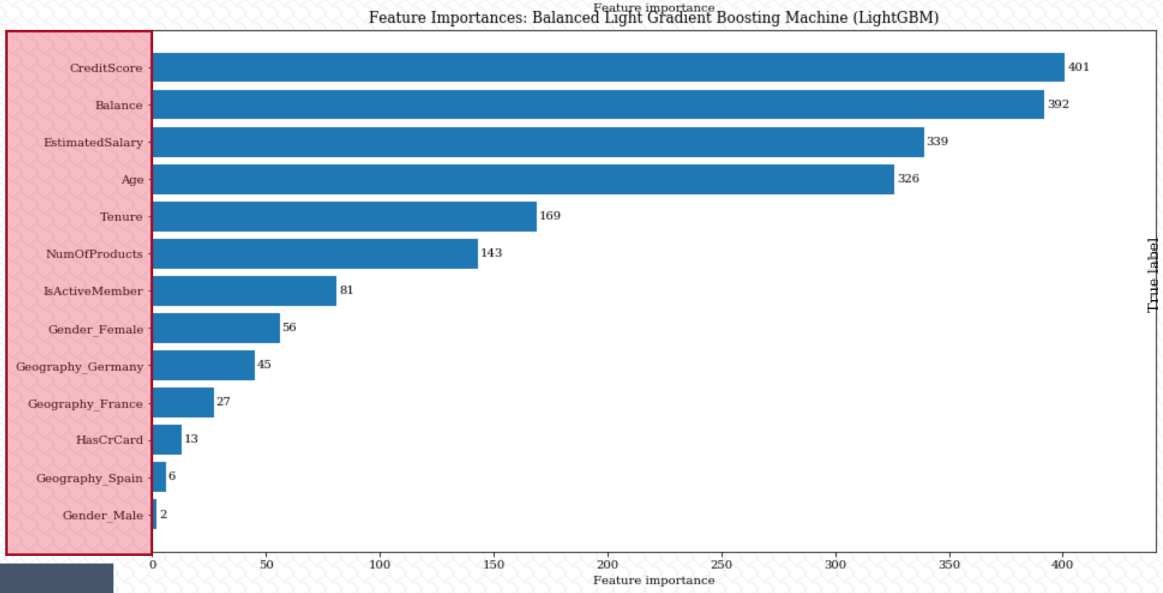
- They require powerful hardware specifications.
- Difficult to understand how the prediction was arrived at.
- There is no defined guidelines in determining the suitable network structure to use and it all depends on the experience and trial and error.

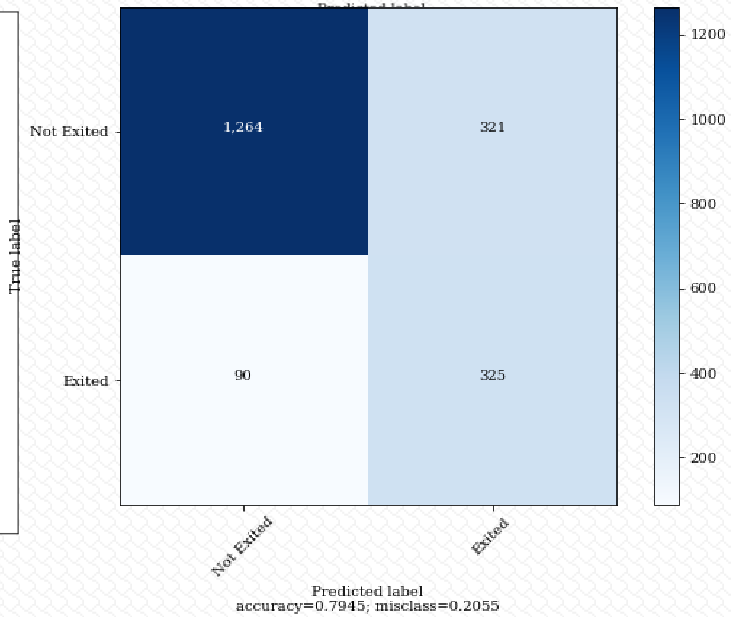
LightGBM: Light Gradient Boosting Machine





Accuracy: 0.869
Balanced Accuracy: 0.738
Recall 1: 0.513





Accuracy: 0.795
Balanced Accuracy: 0.790
Recall 1: 0.783

The top Feature importance is more related to the customer account updates

MODELS COMPARISONS

Model	Balanced	Accuracy	Balanced_Accuracy	AUC	precision_0	recall_0	f1-score_0	precision_1	recall_1	f1-score_1
Logistic Regression	no	0.812	0.592	0.778	0.825	0.967	0.891	0.634	0.217	0.323
Multi-Layer Perceptron (MLP)	no	0.864	0.726	0.871	0.878	0.962	0.918	0.772	0.489	0.599
Gradient Boosting (Sklearn)	no	0.860	0.715	0.878	0.873	0.963	0.916	0.767	0.467	0.581
Gradient Boosting (XGBoost)	no	0.864	0.723	0.875	0.877	0.963	0.918	0.775	0.482	0.594
Gradient Boosting (LightGBM)	no	0.869	0.738	0.881	0.883	0.962	0.921	0.780	0.513	0.619
Logistic Regression	yes	0.716	0.708	0.780	0.900	0.722	0.801	0.395	0.694	0.503
Multi-Layer Perceptron (MLP)	yes	0.785	0.773	0.870	0.924	0.794	0.854	0.488	0.752	0.592
Gradient Boosting (Sklearn)	yes	0.799	0.794	0.878	0.935	0.803	0.864	0.510	0.786	0.619
Gradient Boosting (XGBoost)	yes	0.791	0.785	0.869	0.931	0.795	0.858	0.498	0.776	0.606
Gradient Boosting (LightGBM)	yes	0.795	0.790	0.872	0.934	0.797	0.860	0.503	0.783	0.613

Unbalanced Training set

- All models perform well when looking the pure accuracy score.
- For the balanced accuracy, LR does not perform well, and the other models had the score from 72% to 74%
- The LightGBM performs slightly better also when we looking at the recall of the exited clients

Balanced Training set

- For all the models, the pure accuracy decreased, compared to the unbalanced training set
- The balanced accuracy increased to up to 79% (LightGBM) and LR was the model that improved more.
- The tree based classifier models worked better in all the evaluated metrics.

By looking at the **score metrics and speed performance**, the model I would chose is the **Gradient Boosting Classifier** from the **LightGBM package**. But the **XGBoost** is close behind.

FUTURE WORK

- Do some feature transformations (feature engineering)
- Tuning better the models, most specifically the MLP classifier
- Try other models as SVM –Support Vector Machine
- Try other packages as Keras and Pytorch to implement MLP classifier


THANK YOU