Bank Unity

# Predicting Churn Credit Card Customers

**(Report)**

Nouf Alotaibi

Email: noufmitla@gmail.com

## Abstract

This is the third T5 Data Science Bootcamp project, which is about building classification models that address a useful prediction and/or interpretation problem using Python with Sklearn. In this project, I build classification models to predict who the customers will get churn in Bank Unity using a Credit Card Customers dataset from Kaggle.

## 1. Design

A manager at *Bank Unity* is disturbed by more and more customers leaving their credit card services. They would appreciate it if one could predict who is going to get churn, so they can proactively go to the customer to provide them better services and turn customers' decisions in the opposite direction [1]. Moreover, in this project, I got familiar with Machine Learning modeling methods such as Logistic Regression, Decision Trees, Random Forest, and more.

## 2. Dataset

For this project, I gathered the data from Kaggle:

- **Credit Card Customers Dataset:** it was uploaded to Kaggle.com, see [1]. The dataset contains 10,127 customers records and 21 features about the customers such as their age, salary, marital status, credit card limit, credit card category, etc. The table below illustrates the dataset's features and their types.

| Column Name | Column type |
|---|---|
| CLIENTNUM | int64 |

| | |
|---|---|
| Attrition_Flag | Object |
| Customer_Age | int64 |
| Gender | Object |
| Dependent_count | int64 |
| Education_Level | Object |
| Marital_Status | Object |
| Income_Category | Object |
| Card_Category | Object |
| Months_on_book | int64 |
| Total_Relationship_Count | int64 |
| Months_Inactive_12_mon | int64 |
| Contacts_Count_12_mon | int64 |
| Credit_Limit | float64 |
| Total_Revolving_Bal | int64 |
| Avg_Open_To_Buy | float64 |
| Total_Amt_Chng_Q4_Q1 | float64 |
| Total_Trans_Amt | int64 |
| Total_Trans_Ct | int64 |
| Total_Ct_Chng_Q4_Q1 | float64 |
| Avg_Utilization_Ratio | float64 |

## 3. Feature Engineering

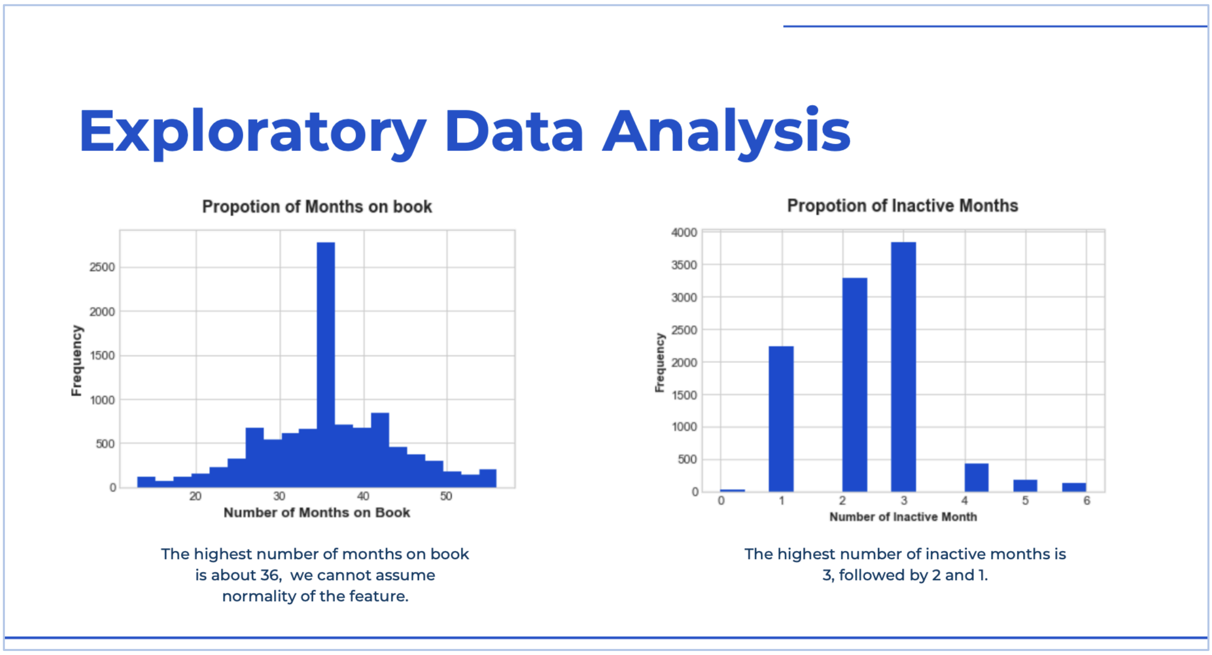
- Converting the target column *Attrition_Flag* to 0's and 1's.
- Creating dummy variables for the categorical columns.
- Factorizing the categorical columns.
- Scaling the numeric columns using StandardScaler().

## 4. Tools

These are the technologies and libraries that I used for this project:

- **Technologies:** Python, Jupyter Notebook.
- **Libraries:** NumPy, Pandas, Matplotlib, Seaborn, Statsmodels, Scikit-learn, imblearn.

## 5. Communications



# Baseline Model

| | | Predicted Values: Existing | Predicted Values: Attrited |
|---|---|---|---|
| **Actual Values** | Existing | 1222 | 45 |
| | Attrited | 135 | 116 |

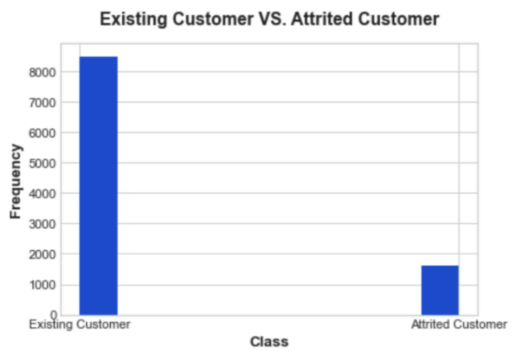
### Training Scores

```
Accuracy = 0.889
Precision = 0.728
Recall = 0.494
F-1 = 0.588
F-beta of 2 = 0.528
```

### Validation Scores

```
Accuracy = 0.881
Precision = 0.72
Recall = 0.462
F-1 = 0.563
F-beta of 2 = 0.498
```

# Data Imbalance




**Number of Observations**

**10127**

**Number of Attrited Customer**

**1627**

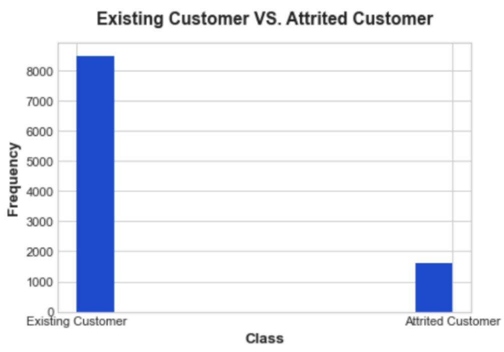
**Number of Existing Customer**

**8500**

**Event Rate**

16.1%

# Data Imbalance




**Approaches for Imbalanced Data**

**Resampling Techniques**

- Random Under-Sampling
- Random Over-Sampling
- ✓ SMOTE

**Bagging Based techniques**

- Random Forest

**Boosting-Based techniques**

- Ada Boost

# Experiments

| Classifier | | Recall | Precision | F-1 | F-2 |
|---|---|---|---|---|---|
| Logistic Regression | train | 0.813 | 0.82 | 0.817 | 0.815 |
| | val | 0.776 | 0.450 | 0.57 | 0.678 |
| Decision Tree | train | 1.0 | 1.0 | 1.0 | 1.0 |
| | val | 0.84 | 0.70 | 0.768 | 0.810 |
| Random Forest | train | 1.0 | 1.0 | 1.0 | 1.0 |
| | val | 0.90 | 0.83 | 0.865 | 0.886 |
| Ada | train | 0.967 | 0.959 | 0.963 | 0.965 |
| | val | 0.912 | 0.760 | 0.829 | 0.877 |

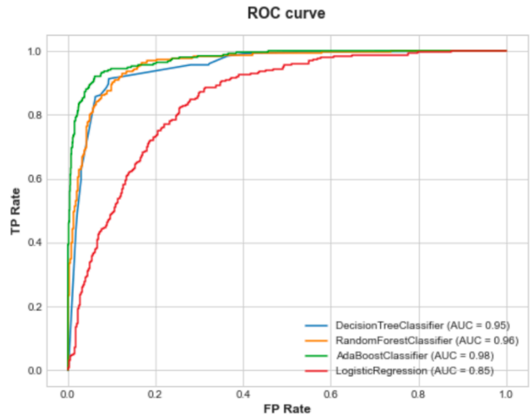
**Split Data**

%75 train,
%15 validation
%10 test

**Resampling Technique**

SMOTE

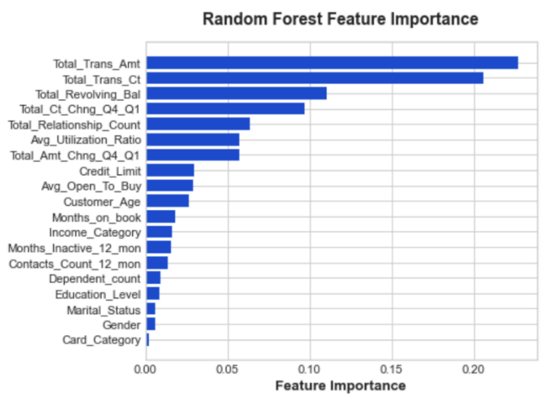
# Experiments




Ada Boost Classifier has best results with **98%** AUC, followed by Random Forest and Decision Tree Classifiers with **96%, 95%** AUC respectively.

# Feature Importance

## For Random Forest




**Low Feature Importance**

Gender
Card_Category
Marital_Status

Were removed.

**Validation Scores**

```
Accuracy = 0.956
Precision =  0.841
Recall = 0.904
F-1 = 0.871
F-beta of 2 = 0.891
```

## 6. Resources

*[1] Credit Card customers. (n.d.). Kaggle: Your Machine Learning and Data Science Community. https://www.kaggle.com/sakshigoyal7/credit-card-customers*