# PREDICTING CREDIT CARD CUSTOMER ATTRITION

**BY**

**FELIX OPARAH**

# Contents

# 1. Introduction

## 1.1Background of Study

Current development in the financial services sector indicates that credit card attrition is one of the major concerns of financial institutions especially banks. This is basically due to its negative impact on profitability. Market dynamics suggests that the banking sector is highly competitive due to the presence of large number of banks, other financial services providers and financial technology companies (Fintechs) competing for the same customers. Hence, each bank tries to retain its customers and maintain their loyalty through the provision of essential products such as credit cards. Customer loyalty is directly related to profit growth; hence, banks should avoid losing customers (Jagadeesan and Indhuja, 2020)

Credit card customer attrition or churn is the loss of customers' usage of a bank's credit card product. Each year, many banks lose their credit card customers and this negatively affects their profit. Since banks seek to maximize profits, it is pertinent to build a system which would proactively predict customers who are likely to stop using this product. Thus appropriate measures could be put in place to forestall such customer churn, probably by offering them better services. Therefore, building a predictive machine learning model becomes pertinent (Parangi, 2022).

This study builds a credit card customer attrition prediction system using relevant historic data. Data on credit card attrition was obtained from the Kaggle website and predictive analytics was implemented on the data using Machine Learning algorithms. This formed a model of predictive analytical system which would predict credit card customer attrition with high accuracy levels.

The major analytical tool adopted for this study was the Statistical Analysis Software SAS programming. However, the Python programming tool was also applied to the same data in order to make comparison between the two programming languages.

### 1.2 Objectives of study

The general objective of this study is to build a system that would predict credit card customer attrition using the SAS programming language. The specific objectives are to achieve the following:

a. Analyse the chosen data to understand the characteristics of the data
b. Perform data visualization using tools such as histogram, charts, boxplots, heat maps, etc so as to discover trends and patterns.
c. Employ machine learning algorithms to build a predictive model that would predict credit card customer attrition.
d. Construct a decision making process based on the result.
e. Adopt the Python programming language to perform the same analysis so as to compare the two languages.

## 1.3Methodology

The methodology of this study is a combination of descriptive, analytical and quantitative techniques to create a thorough understanding of the subject, for the purpose of real life implementation. Data mining techniques and Machine Learning (ML) algorithms were employed to extract insights which would aid strategic business decisions. The SAS programming was adopted using the various advanced procedures (PROC) to clean data, visualize data and implement ML algorithms. Several data visualization types such as scatter plots, bar charts, histogram, boxplot, heat maps, etc were used in the study to provide insights from the data.

Three classifier ML algorithms were employed to build the model. These are:

i. Logistic regression
ii. Decision tree
iii. Random forest

Predicting credit card attrition is a binary decision making process, hence a Classification problem. Consequently, these algorithms were selected based on their reliable predictive ability and to provide robustness to the study.

## 1.4 Description of Dataset

Dataset for this study consists of 10,127 observations and 23 variables showing various customers' information, such as personal detail, demographic details and credit card status details. These include factors like age, gender, income groupings, credit card type, inactive period, utilization ratio, Attrition Flag, etc. The data also shows the spending behaviour of customers such as credit limit and revolving balance.

Exploration of this dataset will enable the understanding of customer- bank relationship dynamics and enable a reliable prediction of customer attrition. Given the set of large data points and multiple variables capture, it is believed customer attrition could be predicted with some high levels of certainty. This dataset was obtained from the Kaggle website however, the original source is Zhyli (2020).

# 2. Analytical Design

This study adopted the Unified Modeling Language (UML). It is a general purpose language that uses a graphical description which can create an abstract model for visualizing, specifying, constructing, and documenting the artifacts of a software-intensive system. It is also used for business modelling and other non-software systems. It offers a standard way to write a system's blueprints, including conceptual items such as business processes and system functions as well as concrete things such as programming language statements, database schemas, and reusable software components. (Waykar, 2013).

The Activity flow in this study is represented in figure 2 below. At the start of the process, data was obtained and cleaned. It was transformed by removing unwanted variables, visualization and changing the structure so as to prepare it for machine learning modeling. The modeling was done and results obtained. Based on the results, customer retention strategies were developed for implementation. Finally the report was submitted.

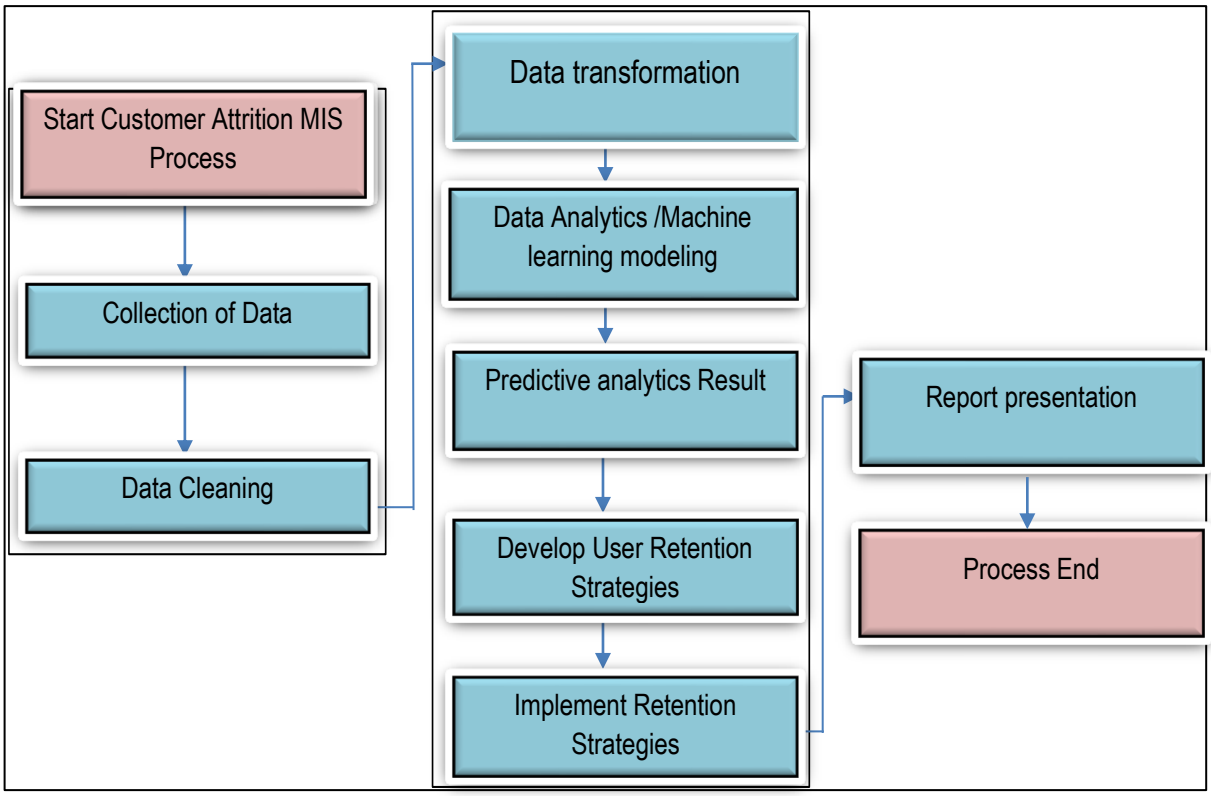



Fig 1: UML Activity design

## 3. Data Analytics

This section involves mining the data to bring out useful insights. Both SAS and Python were used for all the steps involved in this section. However, only the steps done with SAS are presented in the main body of this analysis, while the steps with Python are contained in the Appendix.

### 3.1 Data Importation

The dataset is a CSV file and was imported into the SAS Ondemand platform. The first 5 observations are viewed as shown below:

```
Libname cust_at '/home/u63398813/';
PROC IMPORT DATAFILE='/home/u63398813/BankChurners _1.csv'
    DBMS=CSV
    replace
    OUT=cust_at.Attrition;
    GETNAMES=YES;
RUN;
```

odamid-euw1.oda.sas.com/SASStudio/sasexec/submissions/0b49773f-0a73-4648-a00c-2259960a6521/results

| Obs | CLIENTNUM | Attrition_Flag | Customer_Age | Gender | Dependent_count | Education_Level | Marital_Status | Income_Category | Card_Category | Months_on_book | Total_Relationship_Count | Months_Inactive_12_mon | Contacts_Count_12_mon | Credit_Limit | Total_Revolving_Bal | Avg_ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 768805383 | Existing Customer | 45 | M | 3 | High School | Married | $60K - $80K | Blue | 39 | 5 | 1 | 3 | 12691 | 777 | |
| 2 | 818770008 | Existing Customer | 49 | F | 5 | Graduate | Single | Less than $40K | Blue | 44 | 6 | 1 | 2 | 8256 | 864 | |
| 3 | 713982108 | Existing Customer | 51 | M | 3 | Graduate | Married | $80K - $120K | Blue | 36 | 4 | 1 | 0 | 3418 | 0 | |
| 4 | 769911858 | Existing Customer | 40 | F | 4 | High School | Unknown | Less than $40K | Blue | 34 | 3 | 4 | 1 | 3313 | 2517 | |
| 5 | 709106358 | Existing Customer | 40 | M | 3 | Uneducated | Married | $60K - $80K | Blue | 21 | 5 | 1 | 0 | 4716 | 0 | |

Figure 2: Code and output showing first 5 observations in dataset

## 3.2 Data Cleaning

### 3.2.1 Variable type

The data was examined to determine the variable types and the number of variables using PROC Contents. The results confirm that data is made of up 22 variables and 10127 observations. The nature of the variables was also shown: numerical and character.

The CONTENTS Procedure

| | | | |
|---|---|---|---|
| Data Set Name | CUST_AT.ATTRITION | Observations | 10127 |
| Member Type | DATA | Variables | 22 |
| Engine | V9 | Indexes | 0 |
| Created | 05/12/2023 11:07:26 | Observation Length | 192 |
| Last Modified | 05/12/2023 11:07:26 | Deleted Observations | 0 |
| Protection | | Compressed | NO |
| Data Set Type | | Sorted | NO |
| Label | | | |
| Data Representation | SOLARIS_X86_64, LINUX_X86_64, ALPHA_TRU64, LINUX_IA64 | | |
| Encoding | utf-8 Unicode (UTF-8) | | |

Alphabetic List of Variables and Attributes

| # | Variable | Type | Len | Format | Informat |
|---|---|---|---|---|---|
| 1 | Attrition_Flag | Char | 17 | $17. | $17. |
| 15 | Avg_Open_To_Buy | Num | 8 | BEST12. | BEST32. |
| 20 | Avg_Utilization_Ratio | Num | 8 | BEST12. | BEST32. |
| 8 | Card_Category | Char | 6 | $6. | $6. |
| 12 | Contacts_Count_12_mon | Num | 8 | BEST12. | BEST32. |
| 13 | Credit_Limit | Num | 8 | BEST12. | BEST32. |
| 2 | Customer_Age | Num | 8 | BEST12. | BEST32. |
| 4 | Dependent_count | Num | 8 | BEST12. | BEST32. |
| 5 | Education_Level | Char | 13 | $13. | $13. |
| 3 | Gender | Char | 1 | $1. | $1. |
| 7 | Income_Category | Char | 14 | $14. | $14. |
| 6 | Marital_Status | Char | 7 | $7. | $7. |
| 11 | Months_Inactive_12_mon | Num | 8 | BEST12. | BEST32. |
| 9 | Months_on_book | Num | 8 | BEST12. | BEST32. |
| 21 | Naive_Bayes_Classifier_Attrition | Num | 8 | BEST12. | BEST32. |
| 16 | Total_Amt_Chng_Q4_Q1 | Num | 8 | BEST12. | BEST32. |
| 19 | Total_Ct_Chng_Q4_Q1 | Num | 8 | BEST12. | BEST32. |
| 10 | Total_Relationship_Count | Num | 8 | BEST12. | BEST32. |
| 14 | Total_Revolving_Bal | Num | 8 | BEST12. | BEST32. |
| 17 | Total_Trans_Amt | Num | 8 | BEST12. | BEST32. |
| 18 | Total_Trans_Ct | Num | 8 | BEST12. | BEST32. |
| 22 | VAR23 | Num | 8 | BEST12. | BEST32. |

Figure 3: Tables showing variable types in dataset

### 3.2.2 Missing values

Data was also checked for missing values (numeric and categorical). The result shows there are no missing values in the data:

of Contents

The MEANS Procedure

| Variable | N Miss |
|---|---|
| Customer_Age | 0 |
| Dependent_count | 0 |
| Months_on_book | 0 |
| Total_Relationship_Count | 0 |
| Months_Inactive_12_mon | 0 |
| Contacts_Count_12_mon | 0 |
| Credit_Limit | 0 |
| Total_Revolving_Bal | 0 |
| Avg_Open_To_Buy | 0 |
| Total_Amt_Chng_Q4_Q1 | 0 |
| Total_Trans_Amt | 0 |
| Total_Trans_Ct | 0 |
| Total_Ct_Chng_Q4_Q1 | 0 |
| Avg_Utilization_Ratio | 0 |
| Naive_Bayes_Classifier_Attrition | 0 |
| VAR23 | 0 |

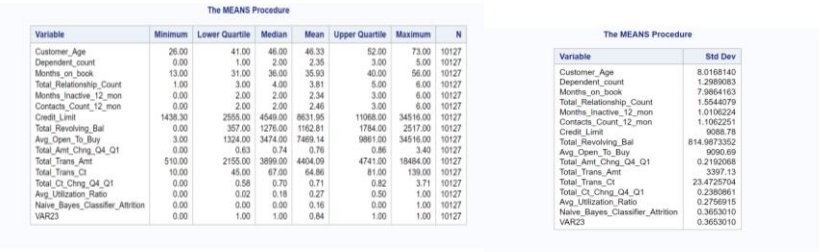
| Attrition_Flag_miss | Gender_miss | Educational_level_miss | Marital_status_miss | Income_Category_miss | Card_Category_miss |
|---|---|---|---|---|---|
| 0 | 0 | 0 | 0 | 0 | 0 |

Figure 4: Table showing zero missing values

### 3.2.3 Summary statistics

Summary statistics were obtained using PROC Means as shown in figure 5 below:

```
run;
/*summary of the numerical variables*/
proc means data = cust_at.attrition min q1 median mean q3 max n maxdec=2;
run;
title "Summary Statistics for numerical variables in Customer Churn Data";
run;
/*Calculate the standard deviation of the numerical variables*/
proc means data = cust_at.attrition Std;
run;
```



The MEANS Procedure

| Variable | Minimum | Lower Quartile | Median | Mean | Upper Quartile | Maximum | N |
|---|---|---|---|---|---|---|---|
| Customer_Age | 26.00 | 41.00 | 46.00 | 46.33 | 52.00 | 73.00 | 10127 |
| Dependent_count | 0.00 | 1.00 | 2.00 | 2.35 | 3.00 | 5.00 | 10127 |
| Months_on_book | 13.00 | 31.00 | 36.00 | 35.93 | 40.00 | 56.00 | 10127 |
| Total_Relationship_Count | 1.00 | 3.00 | 4.00 | 3.81 | 5.00 | 6.00 | 10127 |
| Months_Inactive_12_mon | 0.00 | 2.00 | 2.00 | 2.34 | 3.00 | 6.00 | 10127 |
| Contacts_Count_12_mon | 0.00 | 2.00 | 2.00 | 2.46 | 3.00 | 6.00 | 10127 |
| Credit_Limit | 1438.30 | 2555.00 | 4549.00 | 8631.95 | 11068.00 | 34516.00 | 10127 |
| Total_Revolving_Bal | 0.00 | 357.00 | 1276.00 | 1162.81 | 1784.00 | 2517.00 | 10127 |
| Avg_Open_To_Buy | 3.00 | 1324.00 | 3474.00 | 7469.14 | 9861.00 | 34516.00 | 10127 |
| Total_Amt_Chng_Q4_Q1 | 0.00 | 0.63 | 0.74 | 0.76 | 0.86 | 3.40 | 10127 |
| Total_Trans_Amt | 510.00 | 2155.00 | 3899.00 | 4404.09 | 4741.00 | 18484.00 | 10127 |
| Total_Trans_Ct | 10.00 | 45.00 | 67.00 | 64.86 | 81.00 | 139.00 | 10127 |
| Total_Ct_Chng_Q4_Q1 | 0.00 | 0.58 | 0.70 | 0.71 | 0.82 | 3.71 | 10127 |
| Avg_Utilization_Ratio | 0.00 | 0.02 | 0.18 | 0.27 | 0.50 | 1.00 | 10127 |
| Naive_Bayes_Classifier_Attrition | 0.00 | 0.00 | 0.00 | 0.16 | 0.00 | 1.00 | 10127 |
| VAR23 | 0.00 | 1.00 | 1.00 | 0.84 | 1.00 | 1.00 | 10127 |

The MEANS Procedure

| Variable | Std Dev |
|---|---|
| Customer_Age | 8.0168140 |
| Dependent_count | 1.2989083 |
| Months_on_book | 7.9864163 |
| Total_Relationship_Count | 1.5544079 |
| Months_Inactive_12_mon | 1.0106224 |
| Contacts_Count_12_mon | 1.1062251 |
| Credit_Limit | 9088.78 |
| Total_Revolving_Bal | 814.9873352 |
| Avg_Open_To_Buy | 9090.69 |
| Total_Amt_Chng_Q4_Q1 | 0.2192068 |
| Total_Trans_Amt | 3397.13 |
| Total_Trans_Ct | 23.4725704 |
| Total_Ct_Chng_Q4_Q1 | 0.2380861 |
| Avg_Utilization_Ratio | 0.2756915 |
| Naive_Bayes_Classifier_Attrition | 0.3653010 |
| VAR23 | 0.3653010 |

Figure 5: Summary statistics of numerical variables

The summary statistics show that the highest mean value occurred in Credit limit with a value of 11068. This is followed by Avg_Open_To_ Buy and Total_Trans_ Amt with 9861 and 4741 respectively. The Standard deviation which measures variability in the data shows that the highest variability occurred in Credit Limit with a value of 9088.78. This is followed by Total Revolving Bal with a value of 814.99. This high variability suggests presence of outliers. Most of the variables have zero as their minimum values.

### 3.2.4 Variable Dropping

The variables were examined for their suitability in the intended analysis. It was discovered that the variables `CLIENTNUM, Naive_Bayes_Classifier_Attrition and Var23` were not needed. Therefore they were dropped with the appropriate PROC as shown below:

```
run;
/*dropping variables not needed in the analysis*/
data cust_at.attrition;
  set cust_at.attrition (drop = CLIENTNUM   Naive_Bayes_Classifier_Attrition Var23);
run;
proc print data = cust_at.attrition(obs=5);
run;
ods graphics on;
/*Creating histogram with proc univariate*/
```

Figure 6: Code showing dropping of variables from dataset

## 3.3 Visualization

**Data was visualized to unravel patterns which would provide insight in the analysis.**

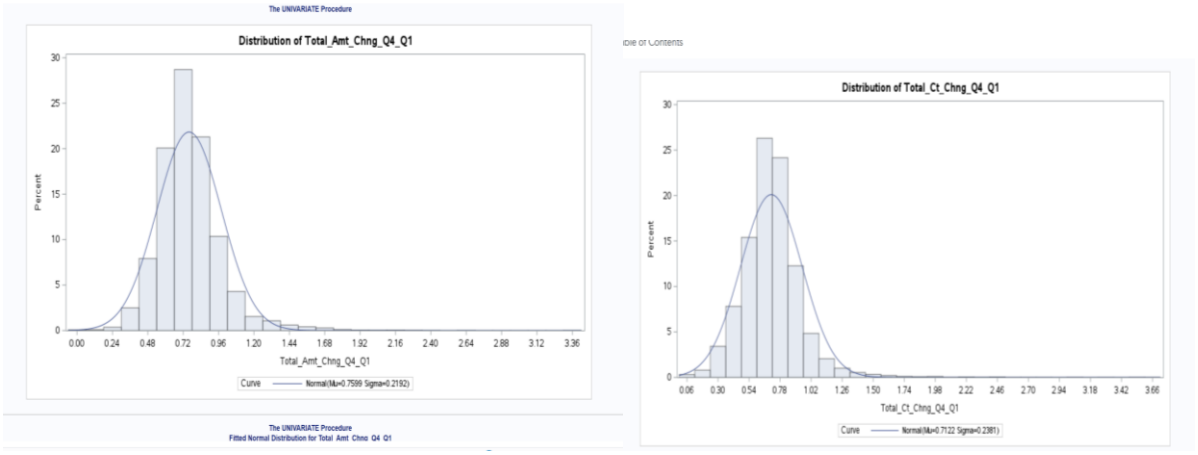



Figure 7: Histogram plots with PROC Univariate

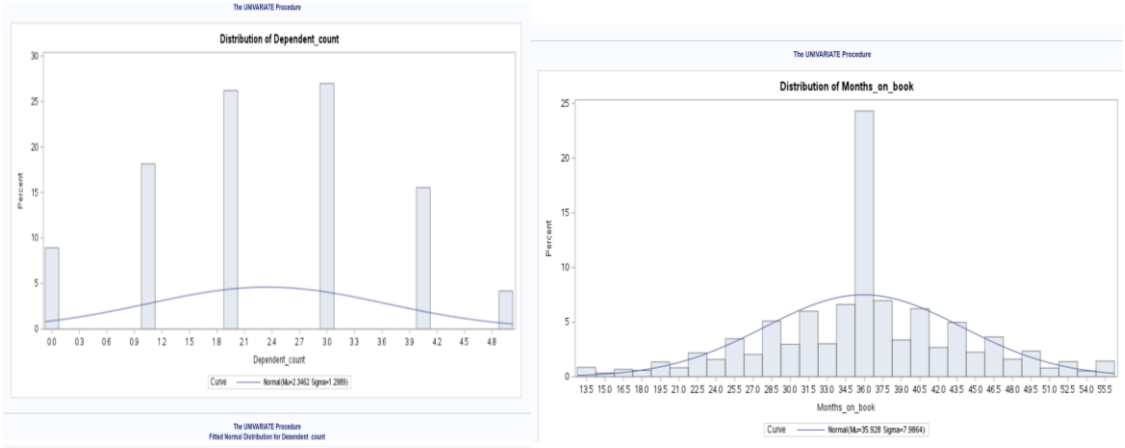



Figure 8: Histogram plots with PROC Univariate

The PROC Univariate was used to obtain histogram for the variables Total_Amt_Chng_Q4_Q1, Total_Ct_Chng_Q1_Q4, Dependent_Count and Months_on_book. The results show that Total_Amt_Chng_Q4_Q1 and Total_Ct_Chng_Q1_Q4 are normally distributed while Dependent_Count and Months_on_book are not normally distributed. This could have resulted from the heavy presence of outliers observed in Dependent_Count and Months_on_book.

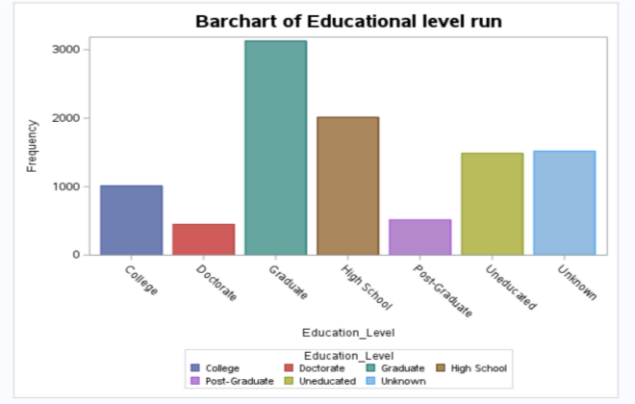



Figure 9: Barchart of Educational Level

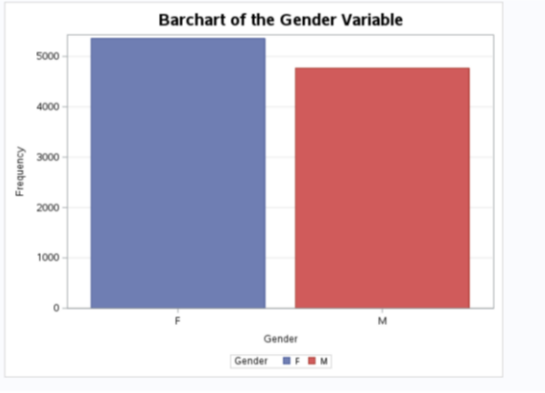



Figure 10: Barchart of Gender

The bar chart in figure 8 shows that graduate education has the highest frequency of 3000 while doctorate degree has the lowest frequency of 500. Similarly, the Gender barchart shows that female customers are more than male customers.

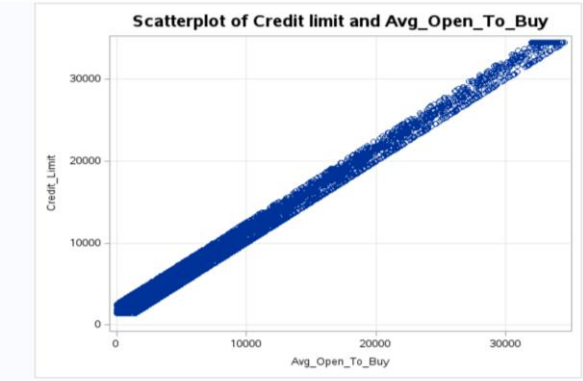



Figure 11: Scatterplot of Credit limit and Avg open to buy

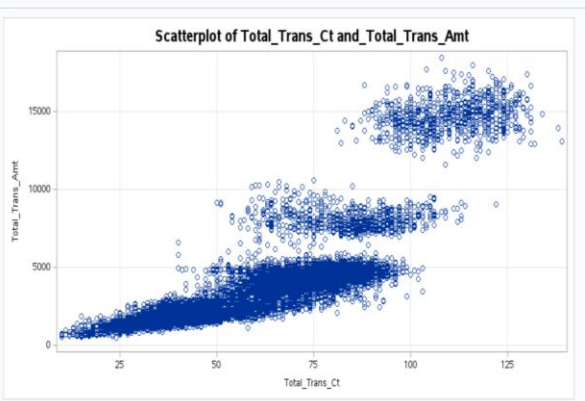



Figure 12: Scatter plot of Total Trans Ct & Total Trans Amt

The Scatterplot in figure 10 shows the points are closely clustered on a straight line and sloped upwards from the origin (zero) to the right hand side of the diagram. This implies a very high positive correlation between Credit limit and Avg open to buy. In other words, the two variables move in the same direction. Similarly, figure 11 shows a positive relationship between Total Trans Ct and Total Trans Amt.

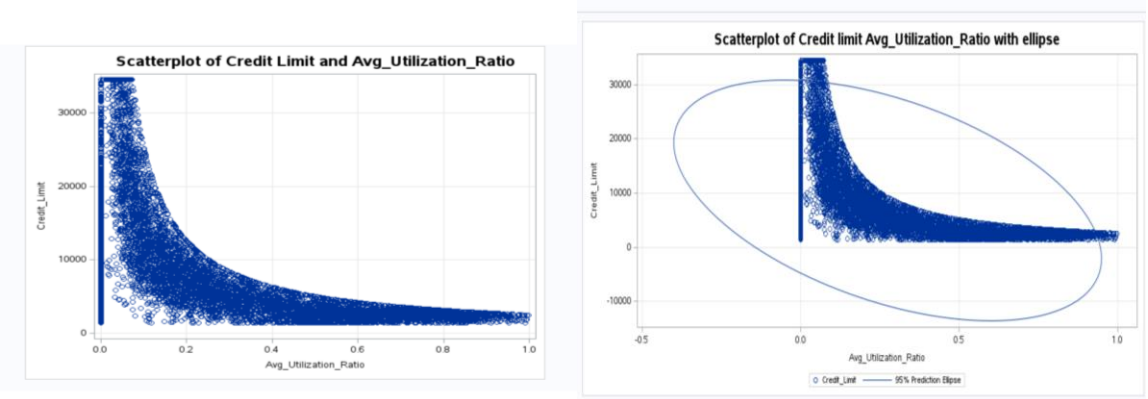



Fig 13 Scatterplots of Credit limit and Ave Utilization Ratio

Scatterplots in figure 11 show an inverse relationship between Credit Limit and Average Utilization Ratio. As Credit limit increases, Average utilization falls, indicating a negative correlation.

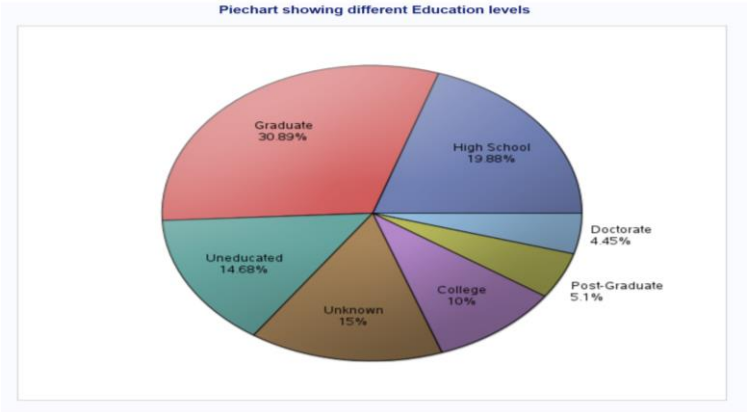



**Figure 14: Pie chart of Education levels**

The pie chart in figure 14 shows the percentage of the various educational levels. Graduate education with 30.89% is the highest followed by High school with 19.88%. Doctorate education is the lowest with 4.45% while all the other types of education ranges from 5.1% to 15%.

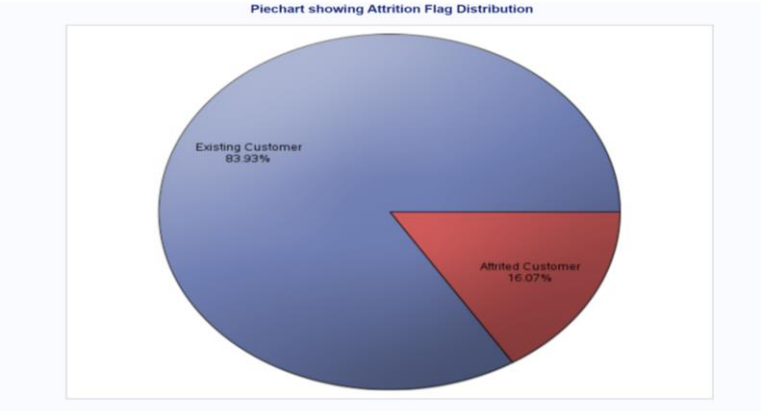



**Figure 15: Pie chart of Attrition flag**

Figure 15 shows the two types of Attrition flag and their percentage. Existing customers are 83.93% of the total dataset while Attrited Customers are 16.07% or 0.1607 of the entire data base.

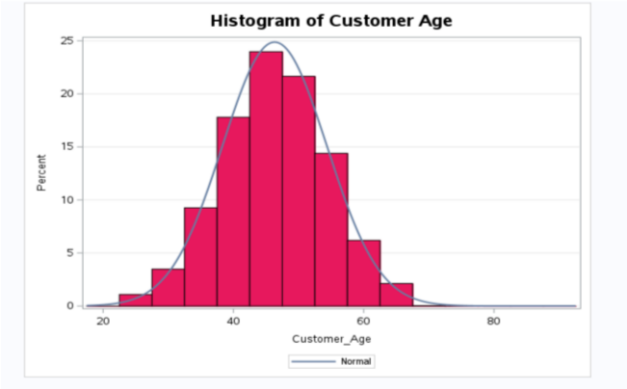



**Figure 16: Histogram of Customer Age**

The histogram in figure 16 shows the distribution of Customer Age. The bell shaped histogram in this figure suggests normality of the underlying data. It also suggest no outliers.

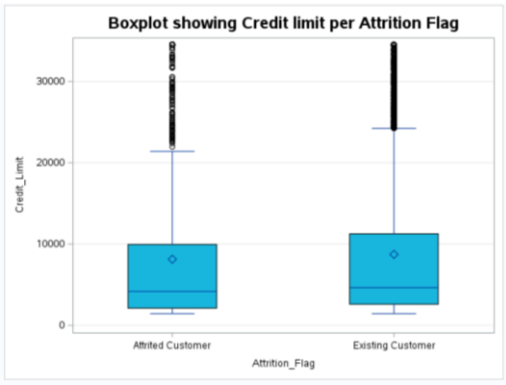



**Fig 17: Boxplot of Credit limit & Attrition flag**

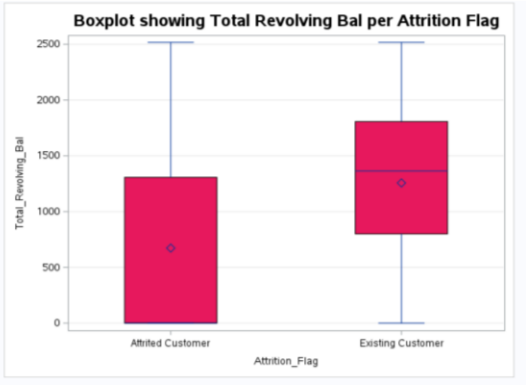



**Fig 18: Boxplot of Total Rev bal & Attrition flag**

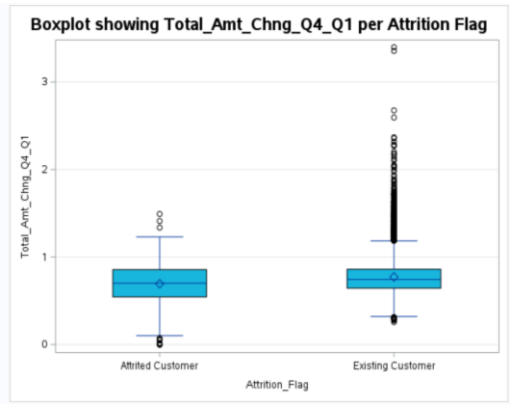



**Fig 19: Boxplot of Total Amt Chng & Attrition flag**

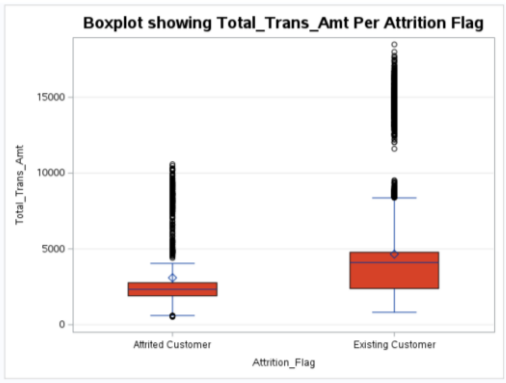



**Fig 20: Boxplot of Total Trans Amt & Attrition flag**

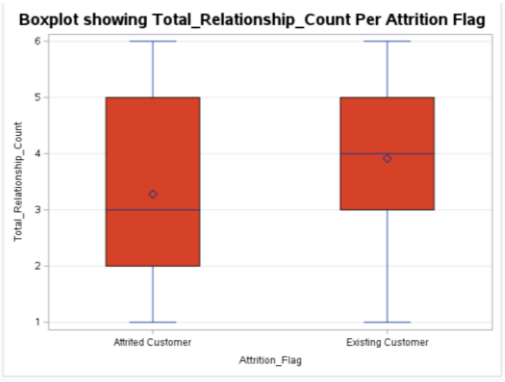



**Fig 21: Boxplot Total r/ship count & Attrition flag**

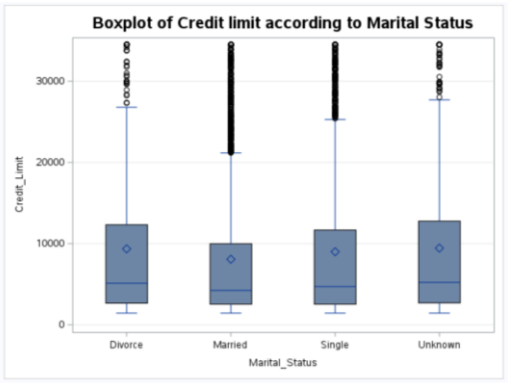
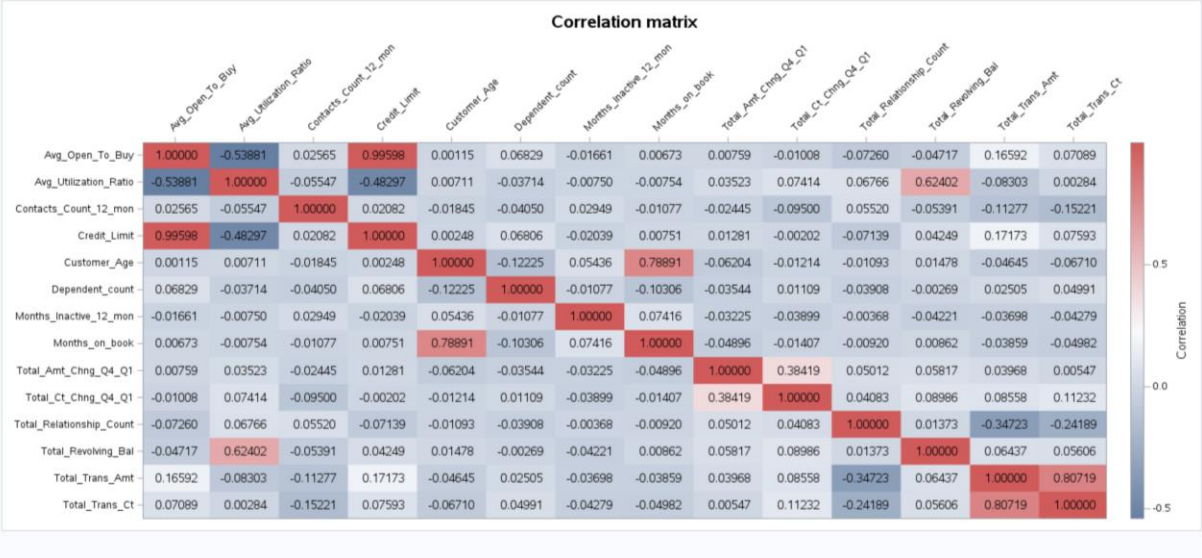



**Fig 22: Boxplot of Credit limit & marital status**

Figures 17, 18, 19, 20 and 21 are boxplots showing the distribution of Credit limit, Total Revolving bal, Total_Amt_Chng_Q4_Q1, Total Trans Amt and Total_Relationship_Count per Attrition Flag. Similarly, Figure 22 shows boxplot of Credit Limit in relation to marital status. Heavy presence of outliers where

noticed in figures 17, 19, 20 and 22. These outliers impinged on the convergence to normality of these variables as shown in the diagrams.




| | Avg_Open_To_Buy | Avg_Utilization_Ratio | Contacts_Count_12_mon | Credit_Limit | Customer_Age | Dependent_count | Months_Inactive_12_mon | Months_on_book | Total_Amt_Chng_Q4_Q1 | Total_Ct_Chng_Q4_Q1 | Total_Relationship_Count | Total_Revolving_Bal | Total_Trans_Amt | Total_Trans_Ct |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Avg_Open_To_Buy | 1.00000 | -0.53881 | 0.02565 | 0.99598 | 0.00115 | 0.06829 | -0.01661 | 0.00673 | 0.00759 | -0.01008 | -0.07260 | -0.04717 | 0.16592 | 0.07089 |
| Avg_Utilization_Ratio | -0.53881 | 1.00000 | -0.05547 | -0.48297 | 0.00711 | -0.03714 | -0.00750 | -0.00754 | 0.03523 | 0.07414 | 0.06766 | 0.62402 | -0.08303 | 0.00284 |
| Contacts_Count_12_mon | 0.02565 | -0.05547 | 1.00000 | 0.02082 | -0.01845 | -0.04050 | 0.02949 | -0.01077 | -0.02445 | -0.09500 | 0.05520 | -0.05391 | -0.11277 | -0.15221 |
| Credit_Limit | 0.99598 | -0.48297 | 0.02082 | 1.00000 | 0.00248 | 0.06806 | -0.02039 | 0.00751 | 0.01281 | -0.00202 | -0.07139 | 0.04249 | 0.17173 | 0.07593 |
| Customer_Age | 0.00115 | 0.00711 | -0.01845 | 0.00248 | 1.00000 | -0.12225 | 0.05436 | 0.78891 | -0.06204 | -0.01214 | -0.01093 | 0.01478 | -0.04645 | -0.06710 |
| Dependent_count | 0.06829 | -0.03714 | -0.04050 | 0.06806 | -0.12225 | 1.00000 | -0.01077 | -0.10306 | -0.03544 | 0.01109 | -0.03908 | -0.00269 | 0.02505 | 0.04991 |
| Months_Inactive_12_mon | -0.01661 | -0.00750 | 0.02949 | -0.02039 | 0.05436 | -0.01077 | 1.00000 | 0.07416 | -0.03225 | -0.03899 | -0.00368 | -0.04221 | -0.03698 | -0.04279 |
| Months_on_book | 0.00673 | -0.00754 | -0.01077 | 0.00751 | 0.78891 | -0.10306 | 0.07416 | 1.00000 | -0.04896 | -0.01407 | -0.00920 | 0.00862 | -0.03859 | -0.04982 |
| Total_Amt_Chng_Q4_Q1 | 0.00759 | 0.03523 | -0.02445 | 0.01281 | -0.06204 | -0.03544 | -0.03225 | -0.04896 | 1.00000 | 0.38419 | 0.05012 | 0.05817 | 0.03968 | 0.00547 |
| Total_Ct_Chng_Q4_Q1 | -0.01008 | 0.07414 | -0.09500 | -0.00202 | -0.01214 | 0.01109 | -0.03899 | -0.01407 | 0.38419 | 1.00000 | 0.04083 | 0.08986 | 0.08558 | 0.11232 |
| Total_Relationship_Count | -0.07260 | 0.06766 | 0.05520 | -0.07139 | -0.01093 | -0.03908 | -0.00368 | -0.00920 | 0.05012 | 0.04083 | 1.00000 | 0.01373 | -0.34723 | -0.24189 |
| Total_Revolving_Bal | -0.04717 | 0.62402 | -0.05391 | 0.04249 | 0.01478 | -0.00269 | -0.04221 | 0.00862 | 0.05817 | 0.08986 | 0.01373 | 1.00000 | 0.06437 | 0.05606 |
| Total_Trans_Amt | 0.16592 | -0.08303 | -0.11277 | 0.17173 | -0.04645 | 0.02505 | -0.03698 | -0.03859 | 0.03968 | 0.08558 | -0.34723 | 0.06437 | 1.00000 | 0.80719 |
| Total_Trans_Ct | 0.07089 | 0.00284 | -0.15221 | 0.07593 | -0.06710 | 0.04991 | -0.04279 | -0.04982 | 0.00547 | 0.11232 | -0.24189 | 0.05606 | 0.80719 | 1.00000 |

Fig 23: Heatmap of numerical variables

The correlation matrix showing correlation among all the numerical variables was presented in a heatmap as shown in Figure 23. The highest positive correlation is between Credit limit and Avg_Open_To_Buy with a value of 0.996. This is followed by correlation between Total_Trans_Ct and Total_Trans_Amt with a value of 0.807. Most of the variables have very low negative correlation.

## 4. Predictive Machine Learning modeling

The dataset was evaluated and it was observed the business problem here is a Classification problem. The central problem is to determine how to predict credit card customers that would churn, thereby exposing the business to reduced profitability. Consequently, three machine learning (ML) algorithms were selected and used to model the data accordingly. These ML algorithms are:

i. Logistic Regression
ii. Decision Tree
iii. Random Forest.

Classification algorithm reduces the Attrition_Flag variable to a binary variable of 0 and 1. The data was divided randomly into training and validation (test) set using PROC Surveyselect with a sample rate of 0.7. Hence the training data is 70% of the data while validation data is 30%. The PROC Surveyselect statement is shown below:

```
/* Split data into two datasets : 70%- training 30%- validation*/
Proc Surveyselect data=CUST_ATT.ATTRITION out=split seed=2345 samprate=.7 outall;
Run;
Data training validation;
Set split;
if selected = 1 then output training;
else output validation;
Run;
```

## 4.1 Logistic regression

Logistic regression is also called a logit model, is employed to investigate dichotomous outcome variables. In the present case, a binary logistics regression is employed. In the model the log odds of the outcome is demonstrated as a linear combination of the predictor variables.

PROC Logistic was used to implement this as follows:

```
/* Logistic regression Model*/
ods graphics on;
Proc Logistic Data = training descending;
class Attrition_Flag Gender Education_Level Marital_Status Income_Category Card_Category/ param = ref;
Model Attrition_Flag = Gender Education_Level
        Marital_Status Income_Category Card_Category Customer_Age Dependent_count
        Months_on_book Total_Relationship_Count Months_Inactive_12_mon
        Contacts_Count_12_mon Credit_Limit Total_Revolving_Bal Avg_Open_To_Buy
        Total_Amt_Chng_Q4_Q1 Total_Trans_Amt Total_Trans_Ct Total_Ct_Chng_Q4_Q1
        Avg_Utilization_Ratio / selection = stepwise slstay=0.15 slentry=0.15 stb;
score data=training out = Logit_Training fitstat outroc=troc;
score data=validation out = Logit_Validation fitstat outroc=vroc;
Run;
```

**The result of the logistic regression is presented below**

The LOGISTIC Procedure

| Model Information | |
|---|---|
| Data Set | WORK.TRAINING |
| Response Variable | Attrition_Flag |
| Number of Response Levels | 2 |
| Model | binary logit |
| Optimization Technique | Fisher's scoring |

| | |
|---|---|
| Number of Observations Read | 7089 |
| Number of Observations Used | 7089 |

| Response Profile | | |
|---|---|---|
| Ordered Value | Attrition_Flag | Total Frequency |
| 1 | Attrited Customer | 1146 |
| 2 | Existing Customer | 5943 |

Probability modeled is Attrition_Flag='Attrited Customer'.

Stepwise Selection Procedure

| Class Level Information | | | | | | | |
|---|---|---|---|---|---|---|---|
| Class | Value | Design Variables | | | | | |
| Gender | F | 1 | | | | | |
| | M | 0 | | | | | |
| Education_Level | College | 1 | 0 | 0 | 0 | 0 | 0 |
| | Doctorate | 0 | 1 | 0 | 0 | 0 | 0 |
| | Graduate | 0 | 0 | 1 | 0 | 0 | 0 |
| | High School | 0 | 0 | 0 | 1 | 0 | 0 |
| | Post-Graduate | 0 | 0 | 0 | 0 | 1 | 0 |
| | Unknown | 0 | 0 | 0 | 0 | 0 | 0 |
| Marital_Status | Divorce | 1 | 0 | 0 | | | |
| | Married | 0 | 1 | 0 | | | |
| | Single | 0 | 0 | 1 | | | |
| | Unknown | 0 | 0 | 0 | | | |
| Income_Category | $120K + | 1 | 0 | 0 | 0 | 0 | |
| | $40K - $60K | 0 | 1 | 0 | 0 | 0 | |
| | $60K - $80K | 0 | 0 | 1 | 0 | 0 | |
| | $80K - $120K | 0 | 0 | 0 | 1 | 0 | |
| | Less than $40K | 0 | 0 | 0 | 0 | 1 | |
| | Unknown | 0 | 0 | 0 | 0 | 0 | |
| Card_Category | Blue | 1 | 0 | 0 | | | |
| | Gold | 0 | 1 | 0 | | | |
| | Platin | 0 | 0 | 1 | | | |
| | Silver | 0 | 0 | 0 | | | |

Step 0. Intercept entered:

| Model Convergence Status |
|---|
| Convergence criterion (GCONV=1E-8) satisfied. |

| -2 Log L | = | 6272.496 |
|---|---|---|

| Residual Chi-Square Test | | |
|---|---|---|
| Chi-Square | DF | Pr > ChiSq |
| 2695.9991 | 31 | <.0001 |

Step 1. Effect Total_Trans_Ct entered:

Step 2. Effect Total_Revolving_Bal entered:

| Model Convergence Status |
|---|
| Convergence criterion (GCONV=1E-8) satisfied. |

| Model Fit Statistics | | |
|---|---|---|
| Criterion | Intercept Only | Intercept and Covariates |
| AIC | 6274.496 | 4634.705 |
| SC | 6281.363 | 4655.304 |
| -2 Log L | 6272.496 | 4628.705 |

| Testing Global Null Hypothesis: BETA=0 | | | |
|---|---|---|---|
| Test | Chi-Square | DF | Pr > ChiSq |
| Likelihood Ratio | 1643.7909 | 2 | <.0001 |
| Score | 1406.3520 | 2 | <.0001 |
| Wald | 1057.6380 | 2 | <.0001 |

| Residual Chi-Square Test | | |
|---|---|---|
| Chi-Square | DF | Pr > ChiSq |
| 1324.4823 | 29 | <.0001 |

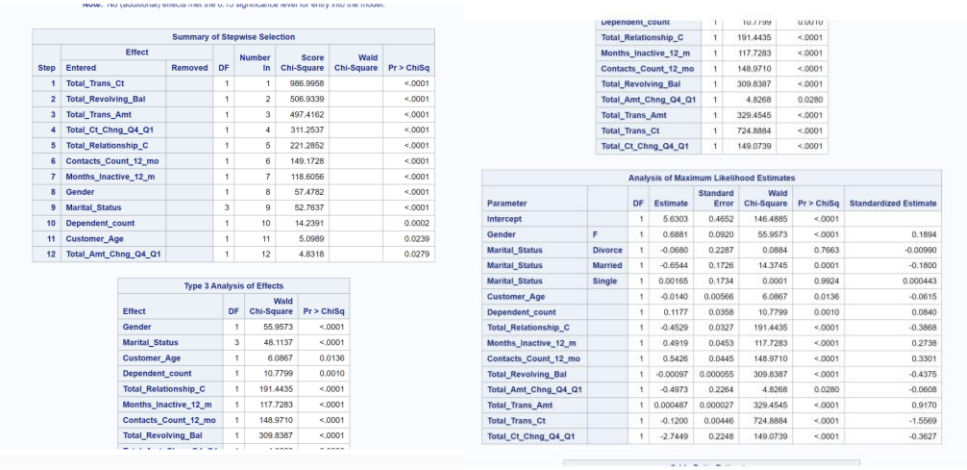
Note: No effects for the model in Step 2 are removed.

Step 3. Effect Total_Trans_Amt entered:

| Model Convergence Status |
|---|
| Convergence criterion (GCONV=1E-8) satisfied. |

Model Fit Statistics

Figure 24: LR model information Figure 25: Model convergence report (selected variable)



| Summary of Stepwise Selection | | | | | | | |
|---|---|---|---|---|---|---|---|
| Step | Effect Entered | Removed | DF | Number In | Score Chi-Square | Wald Chi-Square | Pr > ChiSq |
| 1 | Total_Trans_Ct | | 1 | 1 | 986.9958 | | <.0001 |
| 2 | Total_Revolving_Bal | | 1 | 2 | 506.9339 | | <.0001 |
| 3 | Total_Trans_Amt | | 1 | 3 | 497.4162 | | <.0001 |
| 4 | Total_Ct_Chng_Q4_Q1 | | 1 | 4 | 311.2537 | | <.0001 |
| 5 | Total_Relationship_C | | 1 | 5 | 221.2852 | | <.0001 |
| 6 | Contacts_Count_12_mo | | 1 | 6 | 149.1728 | | <.0001 |
| 7 | Months_Inactive_12_m | | 1 | 7 | 118.6056 | | <.0001 |
| 8 | Gender | | 1 | 8 | 57.4782 | | <.0001 |
| 9 | Marital_Status | | 3 | 9 | 52.7637 | | <.0001 |
| 10 | Dependent_count | | 1 | 10 | 14.2391 | | 0.0002 |
| 11 | Customer_Age | | 1 | 11 | 5.0989 | | 0.0239 |
| 12 | Total_Amt_Chng_Q4_Q1 | | 1 | 12 | 4.8318 | | 0.0279 |

| Type 3 Analysis of Effects | | | |
|---|---|---|---|
| Effect | DF | Wald Chi-Square | Pr > ChiSq |
| Gender | 1 | 55.9573 | <.0001 |
| Marital_Status | 3 | 48.1137 | <.0001 |
| Customer_Age | 1 | 6.0867 | 0.0136 |
| Dependent_count | 1 | 10.7799 | 0.0010 |
| Total_Relationship_C | 1 | 191.4435 | <.0001 |
| Months_Inactive_12_m | 1 | 117.7283 | <.0001 |
| Contacts_Count_12_mo | 1 | 148.9710 | <.0001 |
| Total_Revolving_Bal | 1 | 309.8387 | <.0001 |
| Total_Amt_Chng_Q4_Q1 | 1 | 4.8268 | 0.0280 |
| Total_Trans_Amt | 1 | 329.4545 | <.0001 |
| Total_Trans_Ct | 1 | 724.8884 | <.0001 |
| Total_Ct_Chng_Q4_Q1 | 1 | 149.0739 | <.0001 |

| Analysis of Maximum Likelihood Estimates | | | | | | | |
|---|---|---|---|---|---|---|---|
| Parameter | | DF | Estimate | Standard Error | Wald Chi-Square | Pr > ChiSq | Standardized Estimate |
| Intercept | | 1 | 5.6303 | 0.4652 | 146.4885 | <.0001 | |
| Gender | F | 1 | 0.6881 | 0.0920 | 55.9573 | <.0001 | 0.1894 |
| Marital_Status | Divorce | 1 | -0.0680 | 0.2287 | 0.0884 | 0.7663 | -0.00990 |
| Marital_Status | Married | 1 | -0.6544 | 0.1726 | 14.3745 | 0.0001 | -0.1800 |
| Marital_Status | Single | 1 | 0.00165 | 0.1734 | 0.0001 | 0.9924 | 0.000443 |
| Customer_Age | | 1 | -0.0140 | 0.00566 | 6.0867 | 0.0136 | -0.0615 |
| Dependent_count | | 1 | 0.1177 | 0.0358 | 10.7799 | 0.0010 | 0.0840 |
| Total_Relationship_C | | 1 | -0.4529 | 0.0327 | 191.4435 | <.0001 | -0.3868 |
| Months_Inactive_12_m | | 1 | 0.4919 | 0.0453 | 117.7283 | <.0001 | 0.2738 |
| Contacts_Count_12_mo | | 1 | 0.5426 | 0.0445 | 148.9710 | <.0001 | 0.3301 |
| Total_Revolving_Bal | | 1 | -0.00097 | 0.000055 | 309.8387 | <.0001 | -0.4375 |
| Total_Amt_Chng_Q4_Q1 | | 1 | -0.4973 | 0.2264 | 4.8268 | 0.0280 | -0.0608 |
| Total_Trans_Amt | | 1 | 0.000487 | 0.000027 | 329.4545 | <.0001 | 0.9170 |
| Total_Trans_Ct | | 1 | -0.1200 | 0.00446 | 724.8884 | <.0001 | -1.5569 |
| Total_Ct_Chng_Q4_Q1 | | 1 | -2.7449 | 0.2248 | 149.0739 | <.0001 | -0.3627 |

Fig 26: Stepwise selection, Analysis of Effects and Maximum Likelihood estimates

| | | | | | | |
|---|---|---|---|---|---|---|
| Total_Trans_Amt | | 1 | 0.000487 | 0.000027 | 329.4545 | <.0001 | 0.9170 |
| Total_Trans_Ct | | 1 | -0.1200 | 0.00446 | 724.8884 | <.0001 | -1.5569 |
| Total_Ct_Chng_Q4_Q1 | | 1 | -2.7449 | 0.2248 | 149.0739 | <.0001 | -0.3627 |

**Odds Ratio Estimates**

| Effect | Point Estimate | 95% Wald Confidence Limits | |
|---|---|---|---|
| Gender F vs M | 1.990 | 1.662 | 2.383 |
| Marital_Status Divorce vs Unknown | 0.934 | 0.597 | 1.463 |
| Marital_Status Married vs Unknown | 0.520 | 0.371 | 0.729 |
| Marital_Status Single vs Unknown | 1.002 | 0.713 | 1.407 |
| Customer_Age | 0.986 | 0.975 | 0.997 |
| Dependent_count | 1.125 | 1.049 | 1.207 |
| Total_Relationship_C | 0.636 | 0.596 | 0.678 |
| Months_Inactive_12_m | 1.635 | 1.496 | 1.787 |
| Contacts_Count_12_mo | 1.720 | 1.577 | 1.877 |
| Total_Revolving_Bal | 0.999 | 0.999 | 0.999 |
| Total_Amt_Chng_Q4_Q1 | 0.608 | 0.390 | 0.948 |
| Total_Trans_Amt | 1.000 | 1.000 | 1.001 |
| Total_Trans_Ct | 0.887 | 0.879 | 0.895 |
| Total_Ct_Chng_Q4_Q1 | 0.064 | 0.041 | 0.100 |

**Association of Predicted Probabilities and Observed Responses**

| | | | |
|---|---|---|---|
| Percent Concordant | 92.6 | Somers' D | 0.853 |
| Percent Discordant | 7.4 | Gamma | 0.853 |
| Percent Tied | 0.0 | Tau-a | 0.231 |

Fig 27: Odds Ratio estimates

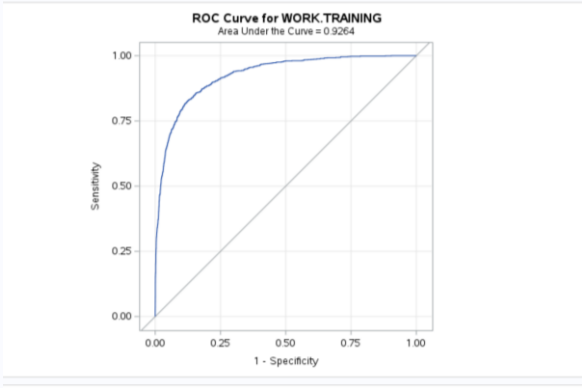
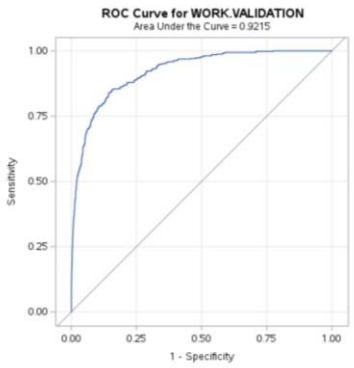



Fig 28: ROC curve for model validation




**Fit Statistics for SCORE Data**

| Data Set | Total Frequency | Log Likelihood | Error Rate | AIC | AICC | BIC | SC | R-Square | Max-Rescaled R-Square | AUC | Brier Score |
|---|---|---|---|---|---|---|---|---|---|---|---|
| WORK.TRAINING | 7089 | -1652.8 | 0.0945 | 3335.617 | 3335.685 | 3438.612 | 3438.612 | 0.341979 | 0.582378 | 0.926361 | 0.069503 |
| WORK.VALIDATION | 3038 | -724.8 | 0.1007 | 1479.681 | 1479.839 | 1569.965 | 1569.965 | 0.327391 | 0.561922 | 0.921536 | 0.071323 |

Fig 29: ROC and Model Fit Statistics

### 4.1.1 Interpretation of results

**i.** The model information (fig 24) shows that the response variable is Attrition flag with 2 level of response in a binary Logit model. The observation used for the training data was 7089 made up of 1146 for Attrited customers and 5943 for Existing Customer. The probability modeled is Attrited Customer. This implies the model is built to predict the probability of Customers leaving the credit card scheme.

**ii.** Model conversion criteria were satisfied for all the explanatory variables as the model iteratively refines the estimates to arrive at the optimal solution. This is shown in figure 25.

**iii.** The "Odds Ratio Estimates" table (figure 27) gives a summary of the significant explanatory variables and shows their associated odds ratios with 95% confidence limits. The coefficients of the model are exponentiated to derive the odds ratio. It could be interpreted as the multiplicative change in the odds for a one unit change in the predictor variable. The odds for Customer Age is 0.986. This means that for 1 year increase in Customer age, the odds of customer attrition will increase by 0.986. Similarly, the odds for Dependent_count is 1.125. This means that for one unit increase in Dependent_count, the odds of Customer Attrition will increase by a factor 1.125. The same explanation hold for other variables.

**iv.** The ROC curve (receiver operating characteristic curve) in figure 29 shows the area under the curve for training and validation models are 0.9264 and 0.9215. This implies 92.64% and 92.15% predictive abilities for the training and validation data respectively. This is a very high predictive ability.

## 4.2 Decision Tree

Decision tree is one of the supervised machine learning algorithms utilized in classification modeling. It employs a hierarchical, tree structure containing root nodes, branches, internal nodes and leaf nodes. The node types perform evaluations to form homogenous subsets represented by leaf nodes, or terminal nodes. The leaf nodes symbolize all the possible outcomes within the dataset. Decision tree also uses a divide and conquer strategy by conducting search to identify the optimal split points within a tree. This process of splitting is then repeated recursively in a top-down fashion until all, or the bulk of records have been classified (Miao and Wang, 2022).

**PROC hpsplit** was utilized to perform decision tree modeling in this study. This is shown below:

```
/* Decision Tree Model*/
proc hpsplit data= cust_at.Attrition;
class Attrition_Flag Gender Education_Level Marital_Status Income_Category Card_Category _CHARACTER_;
Model Attrition_Flag = Gender Education_Level
        Marital_Status Income_Category Card_Category Customer_Age Dependent_count
        Months_on_book Total_Relationship_Count Months_Inactive_12_mon
        Contacts_Count_12_mon Credit_Limit Total_Revolving_Bal Avg_Open_To_Buy
        Total_Amt_Chng_Q4_Q1 Total_Trans_Amt Total_Trans_Ct Total_Ct_Chng_Q4_Q1
        Avg_Utilization_Ratio ;
PARTITION FRACTION(VALIDATE=0.3 SEED=12345);
grow entropy;
prune costcomplexity;
run;
```

The results are presented in the diagrams below:

The HPSPLIT Procedure

| Performance Information | |
|---|---|
| Execution Mode | Single-Machine |
| Number of Threads | 2 |

| Data Access Information | | | |
|---|---|---|---|
| Data | Engine | Role | Path |
| CUST_AT.ATTRITION | V9 | Input | On Client |

| Model Information | |
|---|---|
| Split Criterion Used | Entropy |
| Pruning Method | Cost-Complexity |
| Subtree Evaluation Criterion | Cost-Complexity |
| Number of Branches | 2 |
| Maximum Tree Depth Requested | 10 |
| Maximum Tree Depth Achieved | 10 |
| Tree Depth | 10 |
| Number of Leaves Before Pruning | 185 |
| Number of Leaves After Pruning | 80 |
| Model Event Level | Attrited Customer |

| | |
|---|---|
| Number of Observations Read | 10127 |
| Number of Observations Used | 10127 |
| Number of Training Observations Used | 7114 |
| Number of Validation Observations Used | 3013 |

Fig. 30: Model information

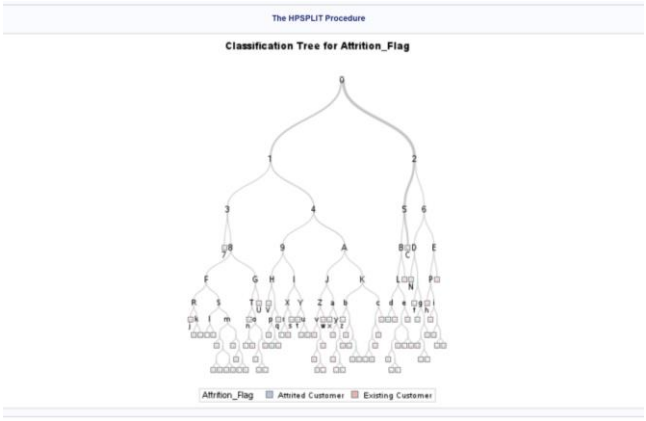



Fig. 31: Classification tree

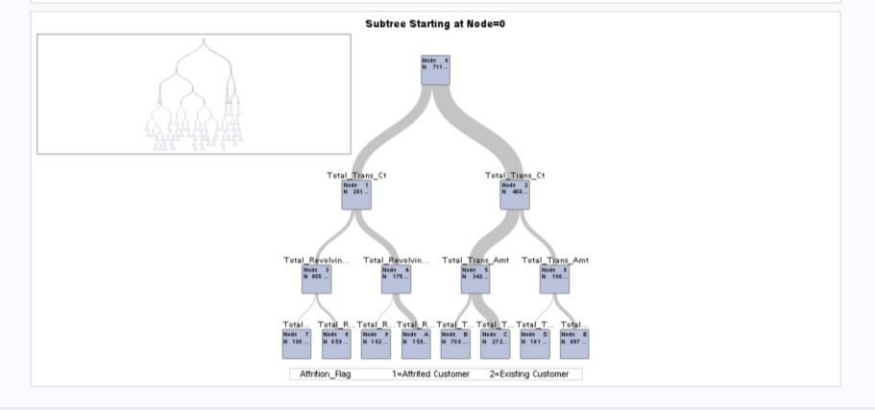



Fig. 32: Sub tree

The HPSPLIT Procedure

Confusion Matrices

| | Actual | Predicted: Attrited Customer | Predicted: Existing Customer | Error Rate |
|---|---|---|---|---|
| Training | Attrited Customer | 992 | 144 | 0.1268 |
| | Existing Customer | 73 | 5905 | 0.0122 |
| Validation | Attrited Customer | 382 | 109 | 0.2220 |
| | Existing Customer | 52 | 2470 | 0.0206 |

Fit Statistics for Selected Tree

| | N Leaves | ASE | Mis-class | Sensitivity | Specificity | Entropy | Gini | RSS | AUC |
|---|---|---|---|---|---|---|---|---|---|
| Training | 80 | 0.0252 | 0.0305 | 0.8732 | 0.9878 | 0.1381 | 0.0503 | 358.0 | 0.9850 |
| Validation | 80 | 0.0449 | 0.0534 | 0.7780 | 0.9794 | 0.1791 | 0.0718 | 270.8 | 0.9432 |

Fig. 33: Confusion Matrix & Fit Stats

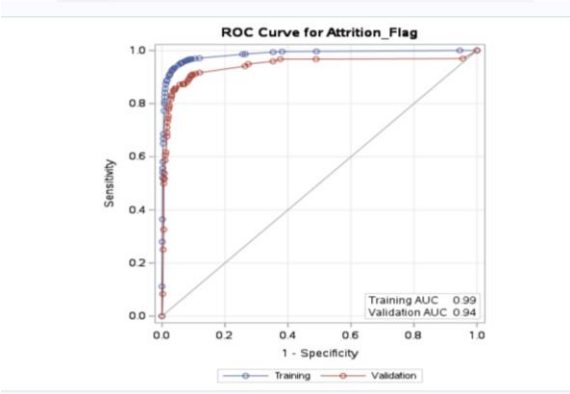



Fig 34: ROC Curve

Variable Importance

| Variable | Training: Relative | Training: Importance | Validation: Relative | Validation: Importance | Relative Ratio | Count |
|---|---|---|---|---|---|---|
| Total_Trans_Ct | 1.0000 | 23.1944 | 1.0000 | 14.9171 | 1.0000 | 10 |
| Total_Revolving_Bal | 0.7883 | 18.2840 | 0.7949 | 11.8583 | 1.0084 | 5 |
| Total_Trans_Amt | 0.5611 | 13.0143 | 0.5948 | 8.8727 | 1.0601 | 19 |
| Total_Relationship_Count | 0.5525 | 12.8148 | 0.5407 | 8.0655 | 0.9786 | 4 |
| Total_Ct_Chng_Q4_Q1 | 0.4901 | 11.3682 | 0.4869 | 7.2632 | 0.9934 | 8 |
| Total_Amt_Chng_Q4_Q1 | 0.2832 | 6.5687 | 0.2015 | 3.0064 | 0.7117 | 7 |
| Months_on_book | 0.2172 | 5.0389 | 0.1832 | 2.7328 | 0.8433 | 3 |
| Months_Inactive_12_mon | 0.3119 | 7.2349 | 0.1814 | 2.7059 | 0.5815 | 4 |
| Contacts_Count_12_mon | 0.1468 | 3.4042 | 0.1296 | 1.9332 | 0.8830 | 3 |
| Customer_Age | 0.1921 | 4.4557 | 0.1120 | 1.6711 | 0.5831 | 3 |
| Income_Category | 0.1892 | 4.3881 | 0.0942 | 1.4055 | 0.4980 | 4 |
| Avg_Open_To_Buy | 0.1726 | 4.0031 | 0.0867 | 1.2934 | 0.5024 | 2 |
| Credit_Limit | 0.1722 | 3.9942 | 0.0000 | 0 | 0.0000 | 3 |
| Dependent_count | 0.1105 | 2.5640 | 0.0000 | 0 | 0.0000 | 2 |
| Education_Level | 0.1048 | 2.4300 | 0.0000 | 0 | 0.0000 | 2 |

Fig 35: Variable importance

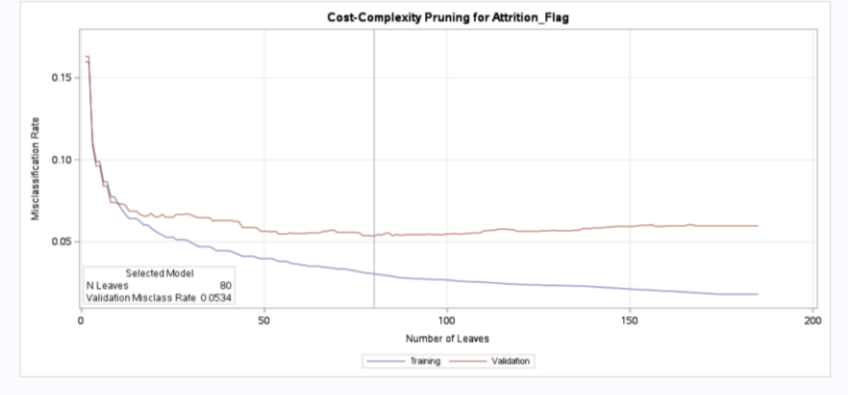



Fig 36: Cost complexity Pruning

### 4.2.1 Interpretation of result

i. The data was randomly split into two (training and validation data) at the ratio of 70:30. The The decision tree was developed on the bank_train data set.
ii. The training data consist of 7114 observations while validation data consist of 3013 observations.
iii. The tree has two branches with initial leaves of 185. However, this was pruned down to 80 leaves as shown by 30.
iv. Figure 33 shows the confusion matrix and fit statistics. The final tree consists of 80 leaves. The training Entropy is 0.1381 while the validation Entropy is 0.1791. The model misclassification rate 0.0305 and 0.0534 for training and validation models respectively.
v. Figure 35 shows the variable importance arranged in descending order of magnitude. The 5 most important variables are Total_trans_ct, Total_Revolving_bal, Total_trans_amt, Total_Relationship_count and Total_ct_chng_Q4_Q1.
vi. The ROC curve show 99% area under the curve for training model and 94% for validation model. This implies a very high predictive ability of the model.

## 4.3 Random Forest

Random forest is a Supervised ML algorithm used extensively in Classification. It pools the results of multiple decision trees to arrive at a single result. Random forest is an extension of the bagging method as it employs both bagging and feature randomness to generate an uncorrelated forest of decision trees. It has 3 key hyperparameters, which are usually set up before training. The hyperparameters comprise the number of trees, node size, and the number of sample features.

**PROC hpforest** was used to implement this algorithm as follows:

```
/* Random Forest Model*/
proc hpforest data = cust_at.Attrition maxtrees = 100 seed = 12345 trainfraction=0.7 leafsize=5
    vars_to_try=4 alpha= 0.1 maxdepth=50;
 target Attrition_Flag / level = Nominal;
  input Gender Education_Level
        Marital_Status Income_Category Card_Category /level=nominal;
    input Customer_Age Dependent_count Months_on_book Total_Relationship_Count Months_Inactive_12_mon
        Contacts_Count_12_mon Credit_Limit Total_Revolving_Bal Avg_Open_To_Buy
        Total_Amt_Chng_Q4_Q1 Total_Trans_Amt Total_Trans_Ct Total_Ct_Chng_Q4_Q1
        Avg_Utilization_Ratio /level=interval;
 ods output FitStatistics = fit_at_runtime;
 ods output VariableImportance = Variable_Importance;
 ods output Baseline = Baseline;
 save file = "/home/u63398813/model_fit.bin";
 run;
title "The Average Square Error";
run;
 /* visualizing the error/accuracy of the Model*/
 proc sgplot data = fit_at_runtime;
  series x=NTrees y=PredAll/legendlabel='Train Error';
  series x=NTrees y=PredOOB/legendlabel='OOB Error';
  xaxis values=(0 to 50 by 1);
  yaxis values=(0 to 0.3 by 0.05) label='Average Square Error';
 run;
  /* visualizing the importance level of each variable included in the Model*/

 proc sgplot data = Variable_Importance;
 vbar Variable /response=Gini  groupdisplay = cluster categoryorder=respdesc;
 title "Variable Importance (Gini)";
```

The results are presented in the diagrams below:

The HPFOREST Procedure

| Performance Information | |
|---|---|
| Execution Mode | Single-Machine |
| Number of Threads | 2 |

| Data Access Information | | | |
|---|---|---|---|
| Data | Engine | Role | Path |
| CUST_AT.ATTRITION | V9 | Input | On Client |

| Model Information | | |
|---|---|---|
| Parameter | Value | |
| Variables to Try | 4 | |
| Maximum Trees | 100 | |
| Actual Trees | 100 | |
| Inbag Fraction | 0.7 | |
| Prune Fraction | 0 | (Default) |
| Prune Threshold | 0.1 | (Default) |
| Leaf Fraction | 0.00001 | (Default) |
| Leaf Size Setting | 5 | |
| Leaf Size Used | 5 | |
| Category Bins | 30 | (Default) |
| Interval Bins | 100 | |
| Minimum Category Size | 5 | (Default) |
| Node Size | 100000 | (Default) |
| Maximum Depth | 50 | |
| Alpha | 0.1 | |
| Exhaustive | 5000 | (Default) |
| Rows of Sequence to Skip | 5 | (Default) |

Fig 37: Model Information

Baseline Fit Statistics

| Statistic | Value |
|---|---|
| Average Square Error | 0.135 |
| Misclassification Rate | 0.161 |
| Log Loss | 0.441 |

Fit Statistics

| Number of Trees | Number of Leaves | Average Square Error (Train) | Average Square Error (OOB) | Misclassification Rate (Train) | Misclassification Rate (OOB) | Log Loss (Train) | Log Loss (OOB) |
|---|---|---|---|---|---|---|---|
| 1 | 102 | 0.0398 | 0.0501 | 0.0522 | 0.0652 | 0.182 | 0.313 |
| 2 | 202 | 0.0383 | 0.0602 | 0.0489 | 0.0774 | 0.139 | 0.342 |
| 3 | 286 | 0.0342 | 0.0556 | 0.0437 | 0.0693 | 0.127 | 0.294 |
| 4 | 382 | 0.0348 | 0.0567 | 0.0408 | 0.0719 | 0.130 | 0.272 |
| 5 | 443 | 0.0365 | 0.0578 | 0.0424 | 0.0751 | 0.140 | 0.244 |
| 6 | 557 | 0.0368 | 0.0574 | 0.0399 | 0.0758 | 0.141 | 0.247 |
| 7 | 645 | 0.0365 | 0.0556 | 0.0411 | 0.0728 | 0.140 | 0.225 |
| 8 | 745 | 0.0358 | 0.0544 | 0.0406 | 0.0704 | 0.139 | 0.213 |
| 9 | 854 | 0.0352 | 0.0525 | 0.0406 | 0.0674 | 0.137 | 0.196 |
| 10 | 957 | 0.0357 | 0.0530 | 0.0419 | 0.0682 | 0.138 | 0.200 |
| 11 | 1026 | 0.0351 | 0.0511 | 0.0411 | 0.0652 | 0.136 | 0.192 |
| 12 | 1151 | 0.0351 | 0.0506 | 0.0415 | 0.0646 | 0.136 | 0.191 |
| 13 | 1243 | 0.0349 | 0.0506 | 0.0407 | 0.0642 | 0.136 | 0.189 |
| 14 | 1345 | 0.0345 | 0.0492 | 0.0406 | 0.0620 | 0.135 | 0.180 |
| 15 | 1433 | 0.0346 | 0.0486 | 0.0403 | 0.0617 | 0.136 | 0.175 |
| 16 | 1542 | 0.0351 | 0.0490 | 0.0407 | 0.0634 | 0.138 | 0.177 |
| 17 | 1642 | 0.0349 | 0.0485 | 0.0405 | 0.0621 | 0.137 | 0.175 |

Fig 38: Fit Statistics

Loss Reduction Variable Importance

| Variable | Number of Rules | Gini | OOB Gini | Margin | OOB Margin |
|---|---|---|---|---|---|
| Total_Trans_Ct | 1099 | 0.040640 | 0.03638 | 0.081279 | 0.076630 |
| Total_Trans_Amt | 1576 | 0.032739 | 0.02737 | 0.065478 | 0.059833 |
| Total_Revolving_Bal | 798 | 0.022798 | 0.02024 | 0.045596 | 0.042690 |
| Total_Ct_Chng_Q4_Q1 | 798 | 0.022014 | 0.01883 | 0.044028 | 0.040590 |
| Avg_Utilization_Ratio | 421 | 0.012608 | 0.01112 | 0.025217 | 0.023593 |
| Total_Relationship_Count | 669 | 0.012390 | 0.01018 | 0.024780 | 0.022554 |
| Total_Amt_Chng_Q4_Q1 | 745 | 0.008414 | 0.00445 | 0.016828 | 0.012822 |
| Contacts_Count_12_mon | 440 | 0.004592 | 0.00324 | 0.009184 | 0.007824 |
| Months_Inactive_12_mon | 531 | 0.004266 | 0.00287 | 0.008532 | 0.007105 |
| Gender | 389 | 0.002500 | 0.00171 | 0.005000 | 0.004239 |
| Customer_Age | 411 | 0.002376 | 0.00087 | 0.004752 | 0.003197 |
| Months_on_book | 312 | 0.001652 | 0.00036 | 0.003304 | 0.001967 |
| Marital_Status | 211 | 0.000789 | 0.00015 | 0.001578 | 0.001172 |
| Credit_Limit | 284 | 0.001174 | 0.00009 | 0.002348 | 0.001238 |
| Avg_Open_To_Buy | 306 | 0.001233 | 0.00001 | 0.002466 | 0.001249 |
| Card_Category | 111 | 0.000139 | -0.00002 | 0.000277 | 0.000250 |
| Dependent_count | 191 | 0.000632 | -0.00021 | 0.001265 | 0.000437 |
| Education_Level | 129 | 0.000253 | -0.00035 | 0.000505 | 0.000065 |
| Income_Category | 158 | 0.000500 | -0.00039 | 0.001000 | 0.000219 |

Fig 39: Loss reduction variable importance

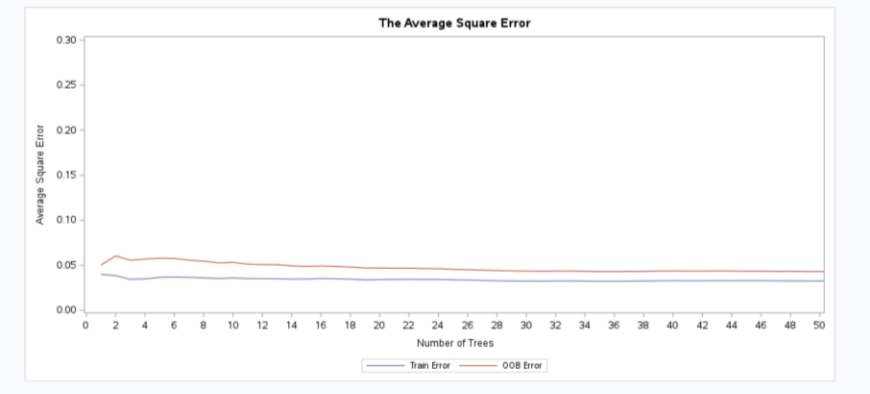


Fig 40: The Average Square Error (ASE)

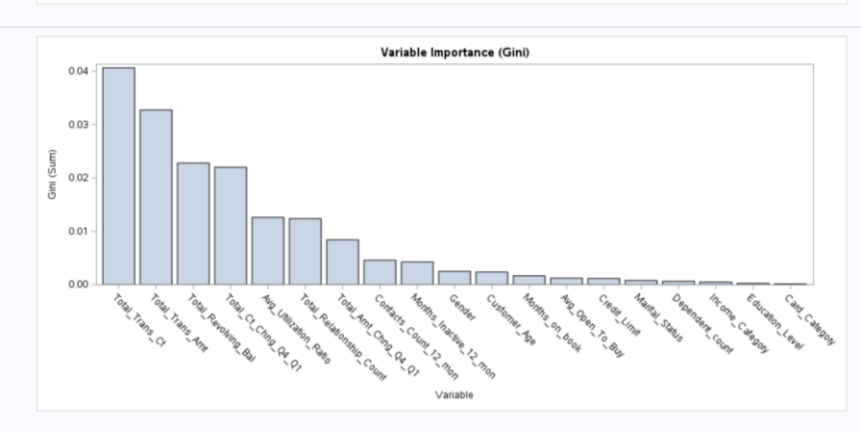


Fig 41: Variable importance

### 4.3.1 Interpretation of result

i. From Fig 37, there are 4 variables selected for the possible split of tree or to test each node and the maximum number of tree was 100.

ii. From the Baseline fit statistics (fig 38), misclassification rate is 0.161 or 16.1%. This means that 83.9% was correctly classified.

iii. The model also shows the Average Square error for the train and OOB (out-of-bag-evaluation) data (fig 40). The results show that model has misclassification reached its lowest point when the tree number is 100.

iv. From the Variable importance diagram (fig 41), the contribution of each variable to the classification of Attrition flag has been shown in order of importance using a bar chart. The chart shows that Total_trans_ct is of highest importance. This is followed by Total_trans_Amt while Card category is of least importance.

# 5. Comparison of Predictive Accuracy powers of the models.

The Accuracy levels of all the ML algorithms are evaluated in table 1 below:

| Predictive Accuracy powers of the models | | |
|---|---|---|
| S/N | Model | Accuracy level |
| 1 | Logistic regression | 92.15% |
| 2 | Decision tree | 94% |
| 3 | Random forest | 83.90% |

Table 1: Model Prediction power

The predictive powers of the models in table 1 above show that Decision tree with 94% has the highest predictive power. It is therefore recommended that Decision tree model should be given preference over the other models in the prediction of credit card customer churn.

# 6. Critical comparison of SAS and Python programing.

## 6.1 Introduction

In this section of the study, the SAS (including SAS Ondemand for Academics) and Python programing tools were critically investigated. Both were used to analyze the dataset hence a comparison of their efficiency levels is made based on defined criteria.

## 6.2 Overview of SAS.

SAS stands for Statistical Analysis Software. SAS stands for Statistical Analysis Software. It was created in the year 1960 by the SAS Institute. From 1st January 1960, SAS was used for data management, business intelligence, Predictive Analysis, Descriptive and Prescriptive Analysis etc. It is employed to explore, visualize and analyze data at high levels of data granularity, so as to extract insight for informed decision making. Unlike open source data analytics tools such as R, Python, Apache Spark, etc, SAS is a proprietary or commercial software. However, it offers some open source platforms such as the SAS OnDemand for Academics which comes at no cost.

## 6.3 Overview of Python programing language

Python is a powerful open source and easy to learn programming language. Python is a general-purpose language, meaning it can be used to create a variety of different programs and everyday tasks. It could be used for developing websites and software, task automation, data analytics, and data visualization. It has a well-designed syntax and together with its interpreted nature (Jupiter Notebook), makes it a superlative language for scripting and application development in many areas. Python can be used as a scripting language or can be compiled to byte-code for building large applications. It can also be easily integrated with C, C++, COM, ActiveX, CORBA, and Java.

## 6.4 Review of SAS and Python programing

i. **Installation and setup:** For SAS the right to set up would be bought from the company however, the SAS Ondemand version requires only creating an online account. Python requires downloading the Jupiter Notebook program from https://www.python.org/.

ii. **Navigation and User experience.** Both involve understanding the syntax or a set of rules associated with each. The SAS uses PROC statements while Python uses physical lines as well as logical lines or statements. Both require some form of learning and mastering. However, the SAS has some form of point and click features which make it a bit easier for those who find the codes more difficult to learn.

iii. **Data mining:** Python provides a robust platform for data mining. Data importation, cleaning, manipulation and visualization could be handled with simple codes. It also produces powerful visuals that make for enriching and insightful reports. On the other hand, the process of data importation in SAS is a bit more complex. File conversion and library creation make it somewhat cumbersome. Besides, certain PROC cannot run on the SAS Ondemand for Academics .

iv. **Scalability:** Python provides a better structure and support for large programs than shell scripting. SAS Ondemand for Academics has a smaller processing capacity than Python. For SAS Ondemand, any result greater than 3MB will result in some form of malfunction while Python can produce results far more than 3MB without any issues. The main SAS can run higher capacity processing.

v. **Statistical Analysis and Machine learning.** Both can handle statistical analysis however, Python have an edge over SAS as it could handle more complex statistical models and machine learning modeling with ease

vi. **Database management:** Python supports a wide range of other third party platforms and databases. It is easily integrated and can run on a wide variety of hardware platforms maintaining the same same interface on all platforms. SAS is more restrictive due to proprietary rights.

vii. **Output.**

The output quality of SAS and Python are compared in the table below. The highest score is 5 while the lowest is 1.

| Output scoring | | |
|---|---|---|
| **Criteria** | **SAS Ondemand for Academics** | **Python** |
| **Installation and setup** | | |
| Ease of installation | 3 | 4 |
| Platform Dependence | 4 | 5 |
| Navigation and user experience | | |
| User Friendliness | 4 | 4 |
| Easy of navigation | 5 | 4 |
| Efficiency | 4 | 5 |
| **Data mining** | | |
| Data importation | 3 | 5 |
| Data cleaning | 4 | 5 |
| Data combination | 4 | 4 |
| Data sub-setting | 4 | 4 |
| Data transformation | 4 | 4 |
| | | |
| **Other criteria** | | |
| Scalability | 3 | 5 |
| Speed | 3 | 5 |
| Statistical analysis and machine learning modeling | 4 | 5 |
| Output | 4 | 5 |
| Data visualization | 4 | 5 |
| Reporting | 5 | 4 |
| Integration with other platforms | 4 | 5 |
| Storage, retrieval and sharing | 4 | 4 |
| Data base management system adaptability | 4 | 5 |
| **Total** | **72** | **82** |

Table 2. SAS Ondemand for Academics and Python scoring

Table 2 indicates that overall Python scored more that SAS Ondemand for Academics. However, a good knowledge of the two is very relevant so that one could serve as a backup. For businesses, such decision may depend on the strategic goals of the company.

# 7. Conclusion

This study investigated customer attrition in a credit card scheme with a view to providing a predictive system that would correctly predict customer attrition. Such prediction will provide financial managers pragmatic decision making basis in eliminating or reducing customer attrition.

Three Machine Learning models Logistic regression, Decision tree and Random forest were deployed to investigate the phenomenon. The result shows that Decision tree performed better than the other models

in making the required prediction. Decision tree provided 94% predictive accuracy hence should be the preferred model. Logistic regression and Random forest provided 92.15% and 83.9 predictive accuracy respectively. It is believed that the adoption of this result will offer the bank (financial institution) opportunity to put measure in place that will improve their services so as to effectively retain their customers. This will ultimately improve the profitability and market share of the bank.

# 8.References

Jagadeesan A.P and Indhuja. R, 2020. Bank Customer Retention Prediction And Customer Ranking Based On Deep Neural Networks. *International Journal of Scientific Development and Research (IJSDR) Volume 5 Issue 9*

Kumar, D.A.; Ravi, V.(2008) Predicting credit card customer churn in banks using data mining. *Int. J. Data Anal. Tech. Strateg.* **2008**, *1*, 4–28

Miao, X.; Wang, H.(2022) Customer churn prediction on credit card services using random forest method. In Proceedings of the 2022 7th International Conference on Financial Innovation and Economic Development (ICFIED 2022), Online, 14–16 January 2022; Atlantis Press: Paris, France; pp. 649–656.

Parangi, A. (2022). Predicting Credit Card Customer Churn. Accessed 11/05/2023. https://www.akkio.com/post/credit-card-churn-prediction

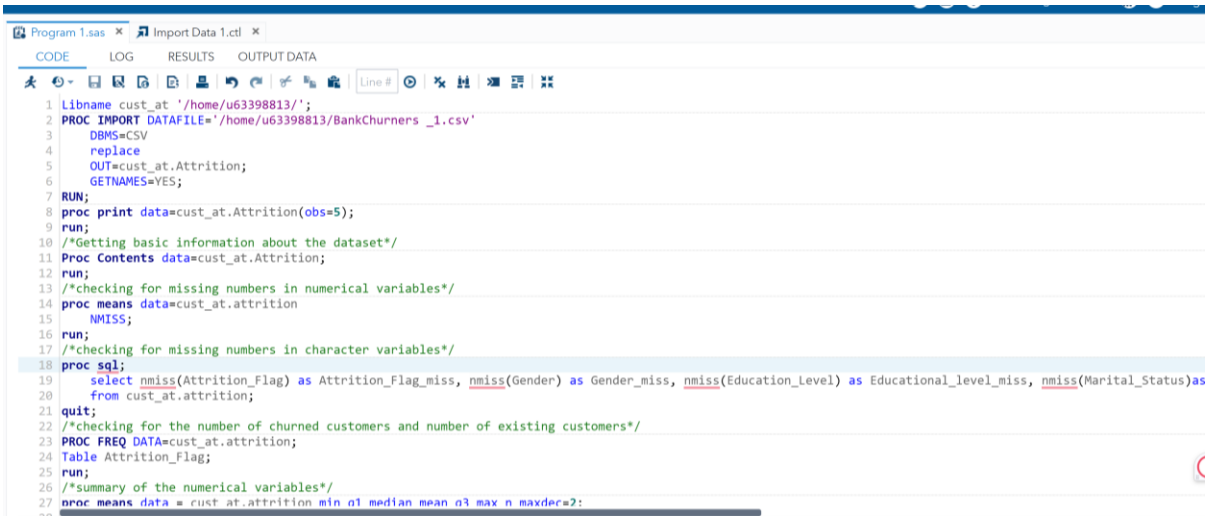
Waykar, Y. (2013) A Study of Importance of UML diagrams: With Special Reference to Very Large-sized Projects. Conference paper at International Conference on Reinventing Thinking beyond boundaries to Excel At: FARIDABAD, INDIA

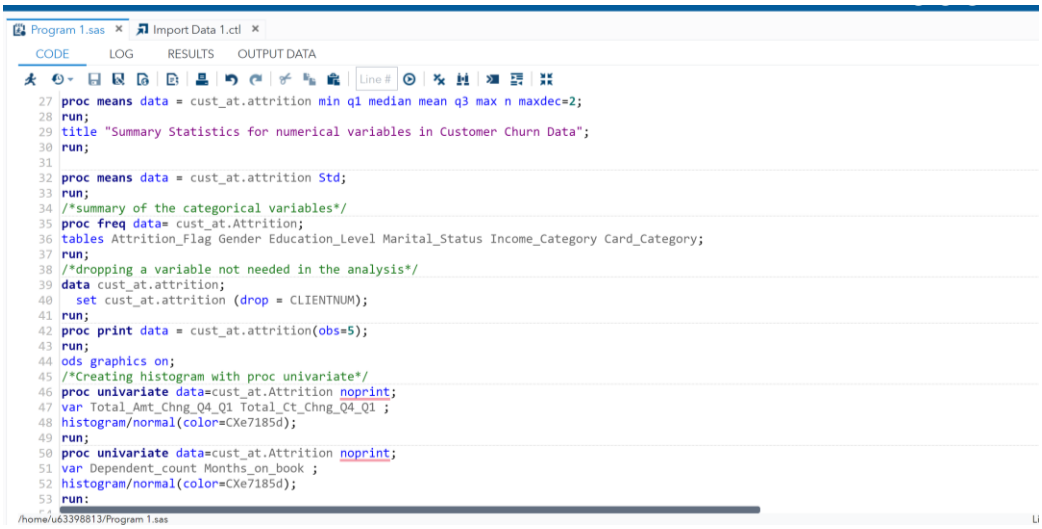
Zhyli. (2020). Prediction of Churning Credit Card Customers [Data set]. Zenodo. https://doi.org/10.5281/zenodo.4322342

**https://olvy.co>blog>customer.churn.analysis**

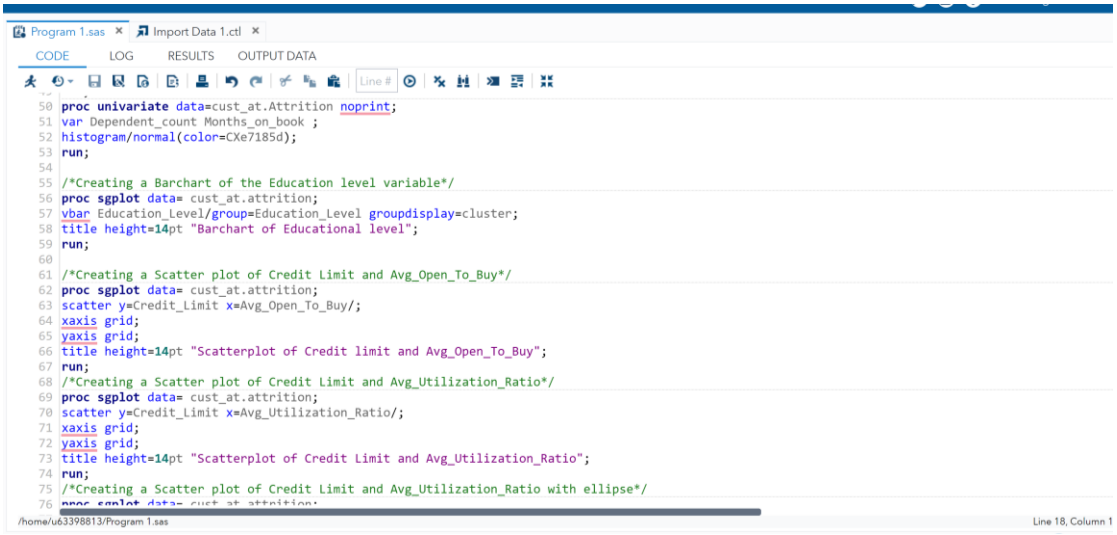
# 9. Appendix 1 : SAS codes

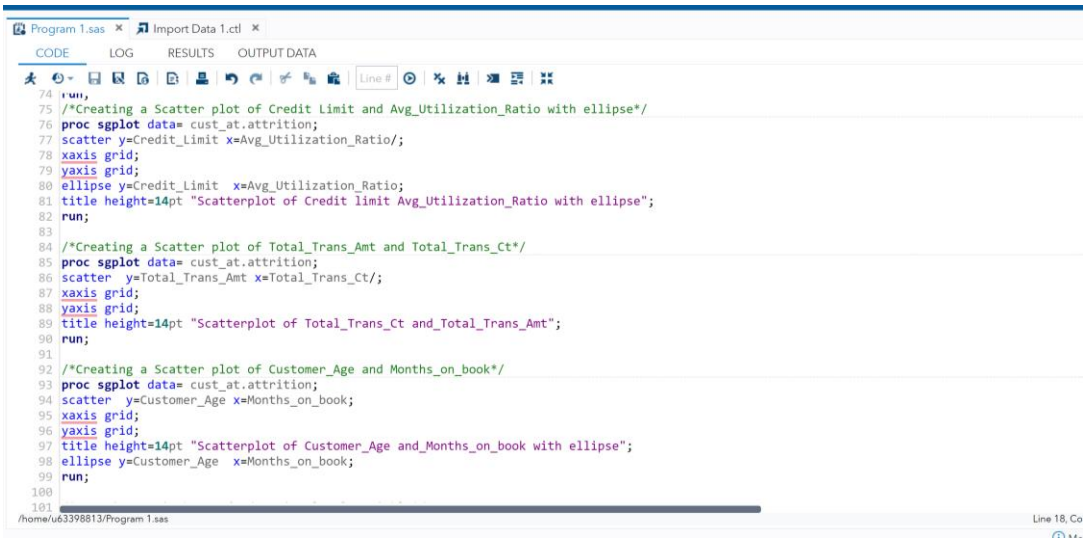
## 9.1 Data pre-processing and mining

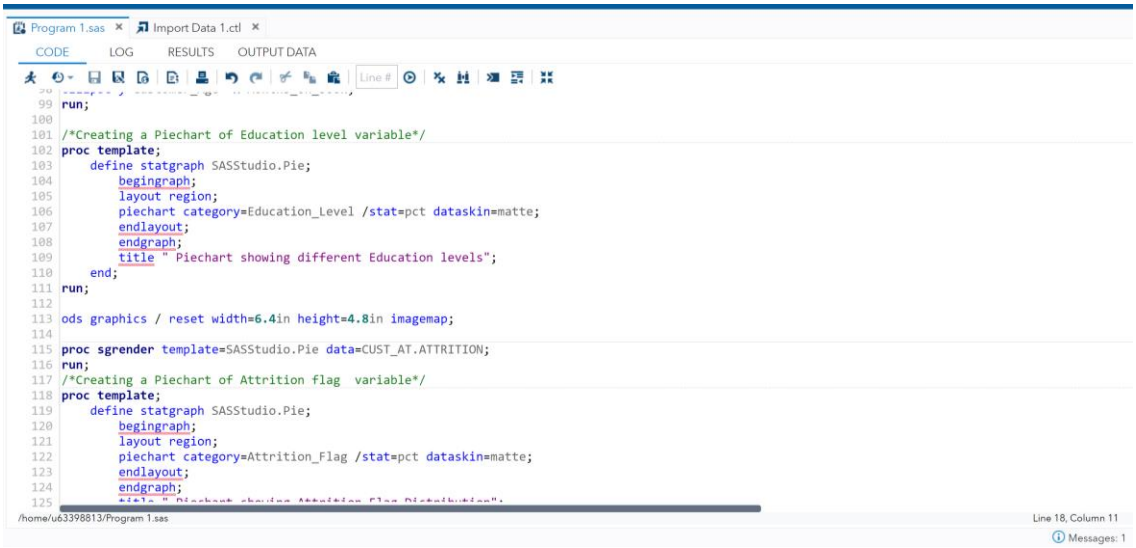


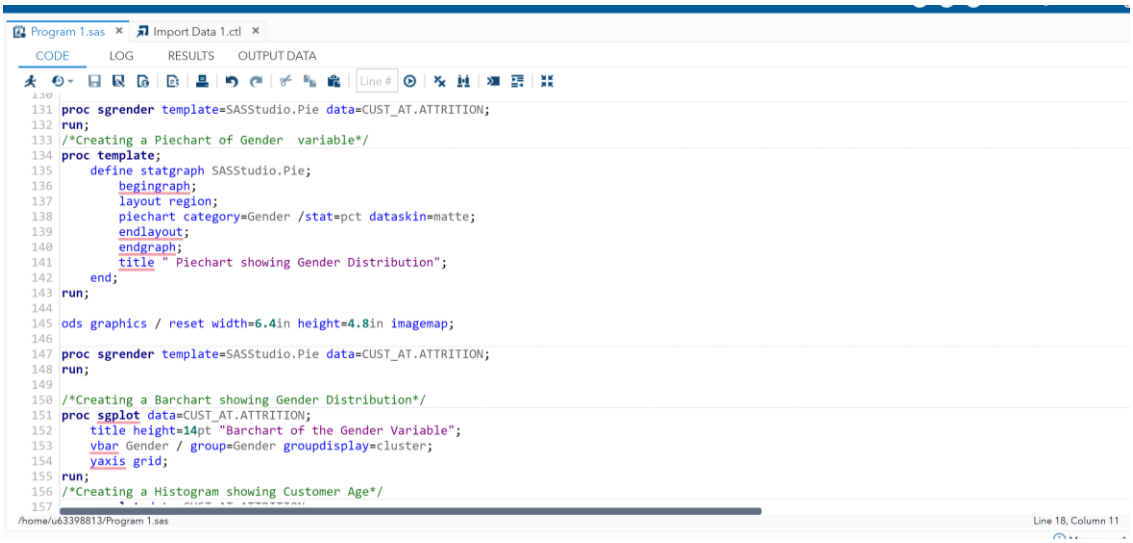



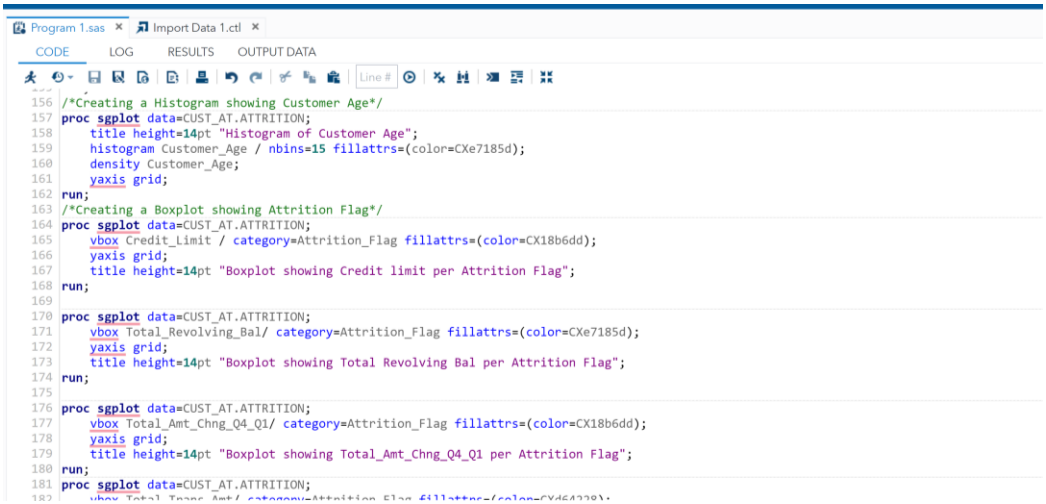
## 9.2: Data Visualization

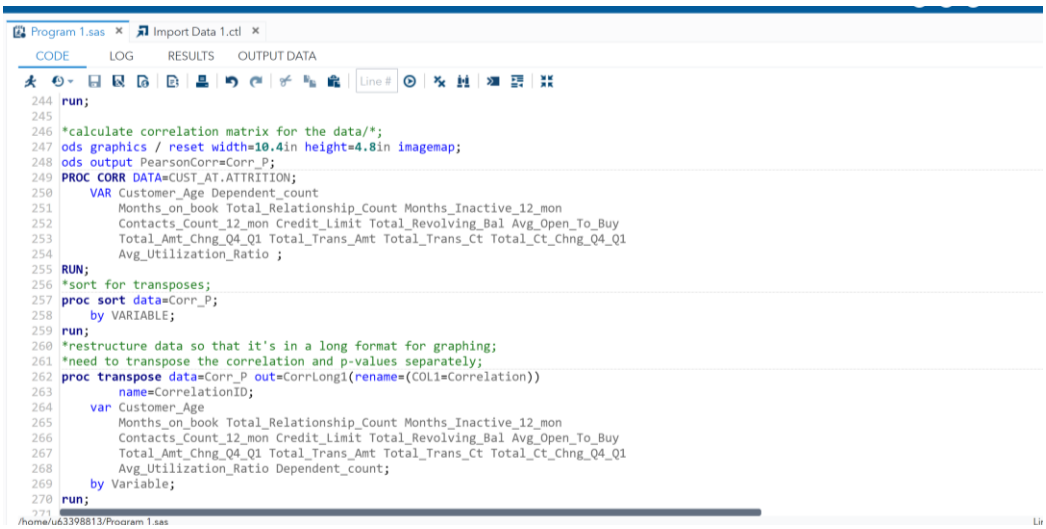


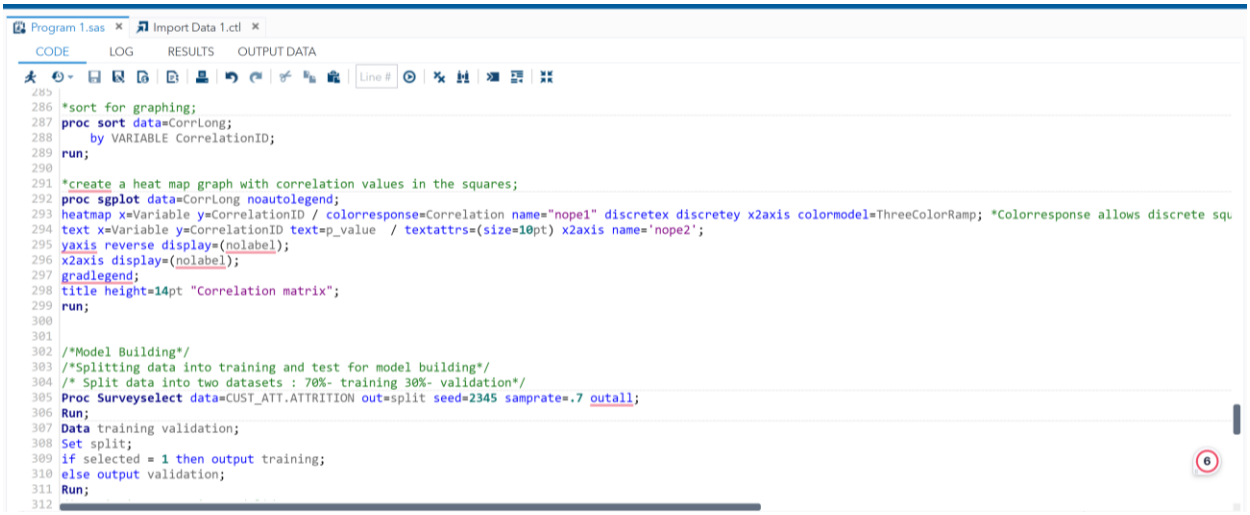

Program 1.sas
Import Data 1.ctl
CODE
LOG
RESULTS
OUTPUT DATA
Line #
run;
/*Creating a Piechart of Education level variable*/
proc template;
define statgraph SASStudio.Pie;
begingraph;
layout region;
piechart category=Education_Level /stat=pct dataskin=matte;
endlayout;
endgraph;
title " Piechart showing different Education levels";
end;
run;
ods graphics / reset width=6.4in height=4.8in imagemap;
proc sgrender template=SASStudio.Pie data=CUST_AT.ATTRITION;
run;
/*Creating a Piechart of Attrition flag variable*/
proc template;
define statgraph SASStudio.Pie;
begingraph;
layout region;
piechart category=Attrition_Flag /stat=pct dataskin=matte;
endlayout;
endgraph;
/home/u63398813/Program 1.sas
Line 18, Column 11
Messages: 1

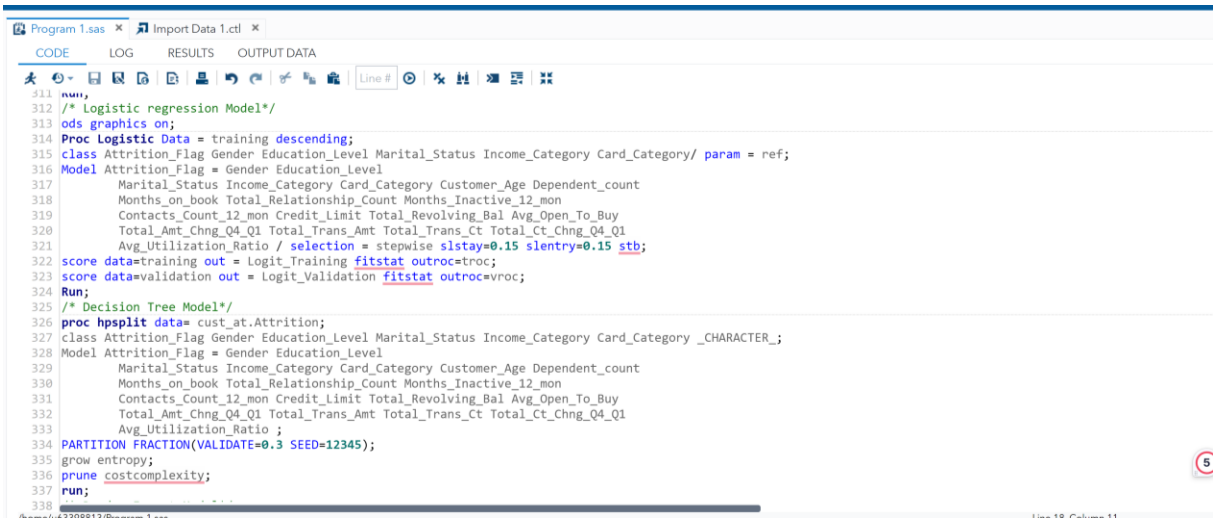

Program 1.sas
Import Data 1.ctl
CODE
LOG
RESULTS
OUTPUT DATA
Line #
proc sgrender template=SASStudio.Pie data=CUST_AT.ATTRITION;
run;
/*Creating a Piechart of Gender variable*/
proc template;
define statgraph SASStudio.Pie;
begingraph;
layout region;
piechart category=Gender /stat=pct dataskin=matte;
endlayout;
endgraph;
title " Piechart showing Gender Distribution";
end;
run;
ods graphics / reset width=6.4in height=4.8in imagemap;
proc sgrender template=SASStudio.Pie data=CUST_AT.ATTRITION;
run;
/*Creating a Barchart showing Gender Distribution*/
proc sgplot data=CUST_AT.ATTRITION;
title height=14pt "Barchart of the Gender Variable";
vbar Gender / group=Gender groupdisplay=cluster;
yaxis grid;
run;
/*Creating a Histogram showing Customer Age*/
/home/u63398813/Program 1.sas
Line 18, Column 11

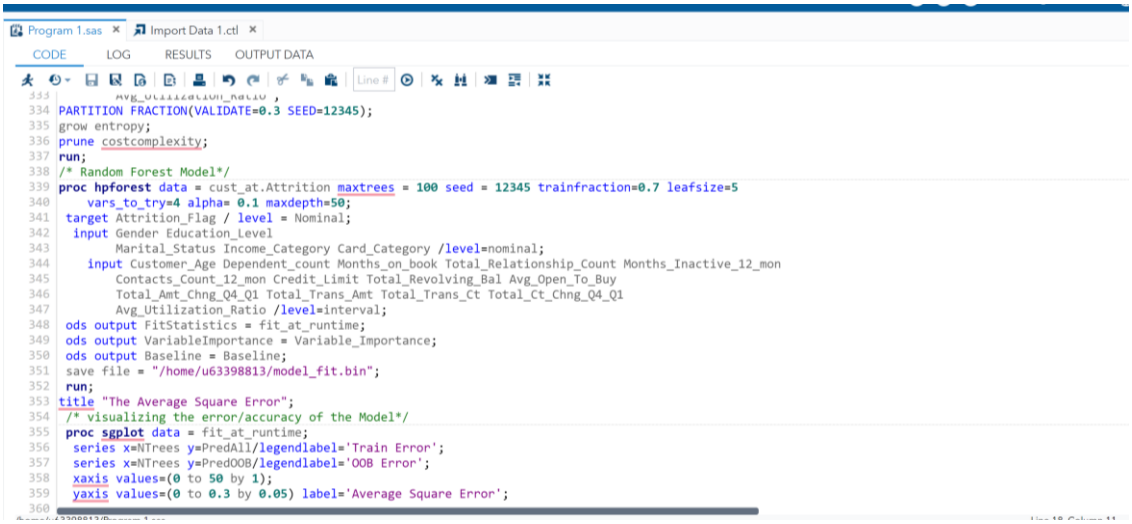

Program 1.sas
Import Data 1.ctl
CODE
LOG
RESULTS
OUTPUT DATA
Line #
/*Creating a Histogram showing Customer Age*/
proc sgplot data=CUST_AT.ATTRITION;
title height=14pt "Histogram of Customer Age";
histogram Customer_Age / nbins=15 fillattrs=(color=CXe7185d);
density Customer_Age;
yaxis grid;
run;
/*Creating a Boxplot showing Attrition Flag*/
proc sgplot data=CUST_AT.ATTRITION;
vbox Credit_Limit / category=Attrition_Flag fillattrs=(color=CX18b6dd);
yaxis grid;
title height=14pt "Boxplot showing Credit limit per Attrition Flag";
run;
proc sgplot data=CUST_AT.ATTRITION;
vbox Total_Revolving_Bal/ category=Attrition_Flag fillattrs=(color=CXe7185d);
yaxis grid;
title height=14pt "Boxplot showing Total Revolving Bal per Attrition Flag";
run;
proc sgplot data=CUST_AT.ATTRITION;
vbox Total_Amt_Chng_Q4_Q1/ category=Attrition_Flag fillattrs=(color=CX18b6dd);
yaxis grid;
title height=14pt "Boxplot showing Total_Amt_Chng_Q4_Q1 per Attrition Flag";
run;
proc sgplot data=CUST_AT.ATTRITION;

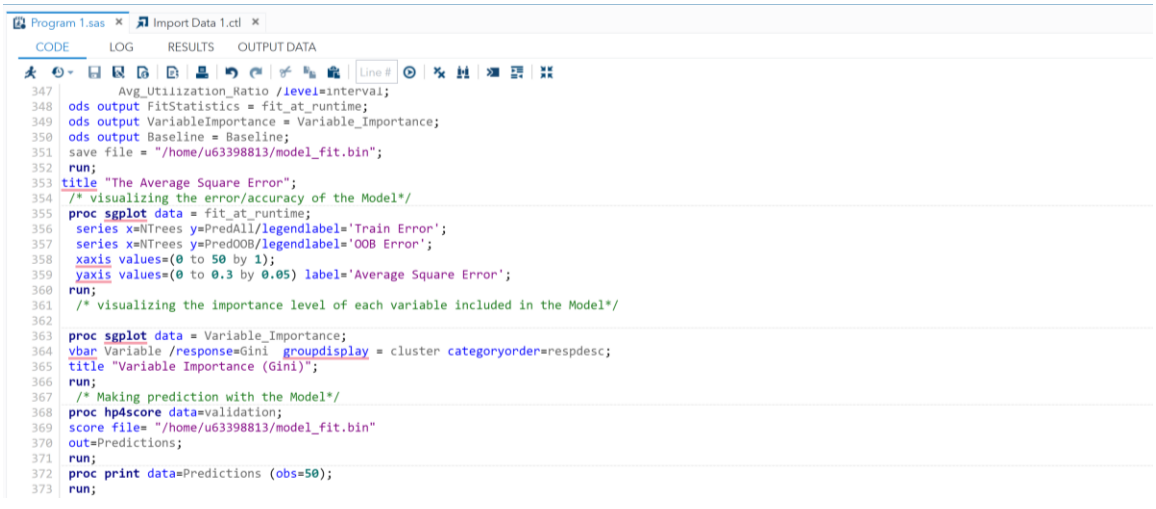



## 9.3: Machine learning Modeling




Program 1.sas
Import Data 1.ctl
CODE
LOG
RESULTS
OUTPUT DATA
/* Logistic regression Model*/
ods graphics on;
Proc Logistic Data = training descending;
class Attrition_Flag Gender Education_Level Marital_Status Income_Category Card_Category/ param = ref;
Model Attrition_Flag = Gender Education_Level
Marital_Status Income_Category Card_Category Customer_Age Dependent_count
Months_on_book Total_Relationship_Count Months_Inactive_12_mon
Contacts_Count_12_mon Credit_Limit Total_Revolving_Bal Avg_Open_To_Buy
Total_Amt_Chng_Q4_Q1 Total_Trans_Amt Total_Trans_Ct Total_Ct_Chng_Q4_Q1
Avg_Utilization_Ratio / selection = stepwise slstay=0.15 slentry=0.15 stb;
score data=training out = Logit_Training fitstat outroc=troc;
score data=validation out = Logit_Validation fitstat outroc=vroc;
Run;
/* Decision Tree Model*/
proc hpsplit data= cust_at.Attrition;
class Attrition_Flag Gender Education_Level Marital_Status Income_Category Card_Category _CHARACTER_;
Model Attrition_Flag = Gender Education_Level
Marital_Status Income_Category Card_Category Customer_Age Dependent_count
Months_on_book Total_Relationship_Count Months_Inactive_12_mon
Contacts_Count_12_mon Credit_Limit Total_Revolving_Bal Avg_Open_To_Buy
Total_Amt_Chng_Q4_Q1 Total_Trans_Amt Total_Trans_Ct Total_Ct_Chng_Q4_Q1
Avg_Utilization_Ratio ;
PARTITION FRACTION(VALIDATE=0.3 SEED=12345);
grow entropy;
prune costcomplexity;
run;


Program 1.sas
Import Data 1.ctl
CODE
LOG
RESULTS
OUTPUT DATA
PARTITION FRACTION(VALIDATE=0.3 SEED=12345);
grow entropy;
prune costcomplexity;
run;
/* Random Forest Model*/
proc hpforest data = cust_at.Attrition maxtrees = 100 seed = 12345 trainfraction=0.7 leafsize=5
vars_to_try=4 alpha= 0.1 maxdepth=50;
target Attrition_Flag / level = Nominal;
input Gender Education_Level
Marital_Status Income_Category Card_Category /level=nominal;
input Customer_Age Dependent_count Months_on_book Total_Relationship_Count Months_Inactive_12_mon
Contacts_Count_12_mon Credit_Limit Total_Revolving_Bal Avg_Open_To_Buy
Total_Amt_Chng_Q4_Q1 Total_Trans_Amt Total_Trans_Ct Total_Ct_Chng_Q4_Q1
Avg_Utilization_Ratio /level=interval;
ods output FitStatistics = fit_at_runtime;
ods output VariableImportance = Variable_Importance;
ods output Baseline = Baseline;
save file = "/home/u63398813/model_fit.bin";
run;
title "The Average Square Error";
/* visualizing the error/accuracy of the Model*/
proc sgplot data = fit_at_runtime;
series x=NTrees y=PredAll/legendlabel='Train Error';
series x=NTrees y=PredOOB/legendlabel='OOB Error';
xaxis values=(0 to 50 by 1);
yaxis values=(0 to 0.3 by 0.05) label='Average Square Error';


Program 1.sas
Import Data 1.ctl
CODE
LOG
RESULTS
OUTPUT DATA
Avg_Utilization_Ratio /level=interval;
ods output FitStatistics = fit_at_runtime;
ods output VariableImportance = Variable_Importance;
ods output Baseline = Baseline;
save file = "/home/u63398813/model_fit.bin";
run;
title "The Average Square Error";
/* visualizing the error/accuracy of the Model*/
proc sgplot data = fit_at_runtime;
series x=NTrees y=PredAll/legendlabel='Train Error';
series x=NTrees y=PredOOB/legendlabel='OOB Error';
xaxis values=(0 to 50 by 1);
yaxis values=(0 to 0.3 by 0.05) label='Average Square Error';
run;
/* visualizing the importance level of each variable included in the Model*/
proc sgplot data = Variable_Importance;
vbar Variable /response=Gini groupdisplay = cluster categoryorder=respdesc;
title "Variable Importance (Gini)";
run;
/* Making prediction with the Model*/
proc hp4score data=validation;
score file= "/home/u63398813/model_fit.bin"
out=Predictions;
run;
proc print data=Predictions (obs=50);
run;

# 10. Appendix 2: Python codes and results

## 10.1 Data pre-processing and mining

```
In [2]: import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns

import warnings
warnings.simplefilter("ignore")
```

```
In [6]:
import warnings
warnings.filterwarnings("ignore")
from datetime import datetime, timedelta
```

```
In [15]: df=pd.read_csv("BankChurners _1.csv")
```

```
In [16]: df.head()
```

Out[16]:

| | CLIENTNUM | Attrition_Flag | Customer_Age | Gender | Dependent_count | Education_Level | Marital_Status | Income_Category | Card_Category | Months_on_book | . |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 768805383 | Existing Customer | 45 | M | 3 | High School | Married | $60K - 80K | Blue | 39 | . |
| 1 | 818770008 | Existing Customer | 49 | F | 5 | Graduate | Single | Less than $40K | Blue | 44 | . |
| 2 | 713982108 | Existing Customer | 51 | M | 3 | Graduate | Married | $80K - 120K | Blue | 36 | . |
| 3 | 769911858 | Existing Customer | 40 | F | 4 | High School | Unknown | Less than $40K | Blue | 34 | . |
| 4 | 709106358 | Existing | 40 | M | 3 | Uneducated | Married | $60K - 80K | Blue | 21 | |

jupyter Untitled8-Copy1 Last Checkpoint: 15 minutes ago (autosaved)

Logout

File Edit View Insert Cell Kernel Widgets Help

Not Trusted | Python 3 (ipykernel)

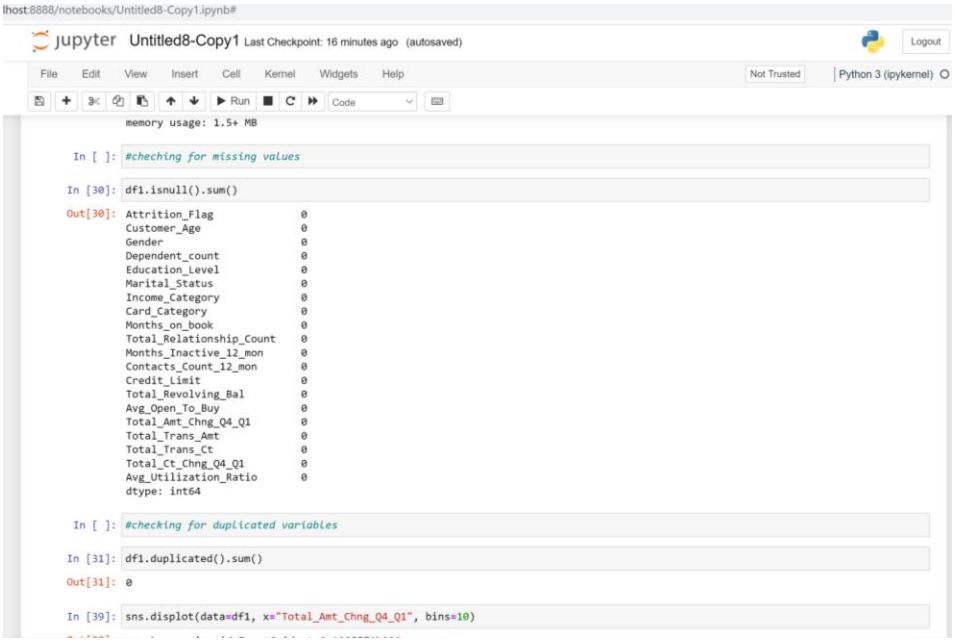
Run Code

```
In [29]: df1.info()
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10127 entries, 0 to 10126
Data columns (total 20 columns):
 #   Column                    Non-Null Count  Dtype
---  ------                    --------------  -----
 0   Attrition_Flag            10127 non-null  object
 1   Customer_Age              10127 non-null  int64
 2   Gender                    10127 non-null  object
 3   Dependent_count           10127 non-null  int64
 4   Education_Level           10127 non-null  object
 5   Marital_Status            10127 non-null  object
 6   Income_Category           10127 non-null  object
 7   Card_Category             10127 non-null  object
 8   Months_on_book            10127 non-null  int64
 9   Total_Relationship_Count  10127 non-null  int64
 10  Months_Inactive_12_mon    10127 non-null  int64
 11  Contacts_Count_12_mon     10127 non-null  int64
 12  Credit_Limit              10127 non-null  float64
 13  Total_Revolving_Bal       10127 non-null  int64
 14  Avg_Open_To_Buy           10127 non-null  float64
 15  Total_Amt_Chng_Q4_Q1      10127 non-null  float64
 16  Total_Trans_Amt           10127 non-null  int64
 17  Total_Trans_Ct            10127 non-null  int64
 18  Total_Ct_Chng_Q4_Q1       10127 non-null  float64
 19  Avg_Utilization_Ratio     10127 non-null  float64
dtypes: float64(5), int64(9), object(6)
memory usage: 1.5+ MB
```

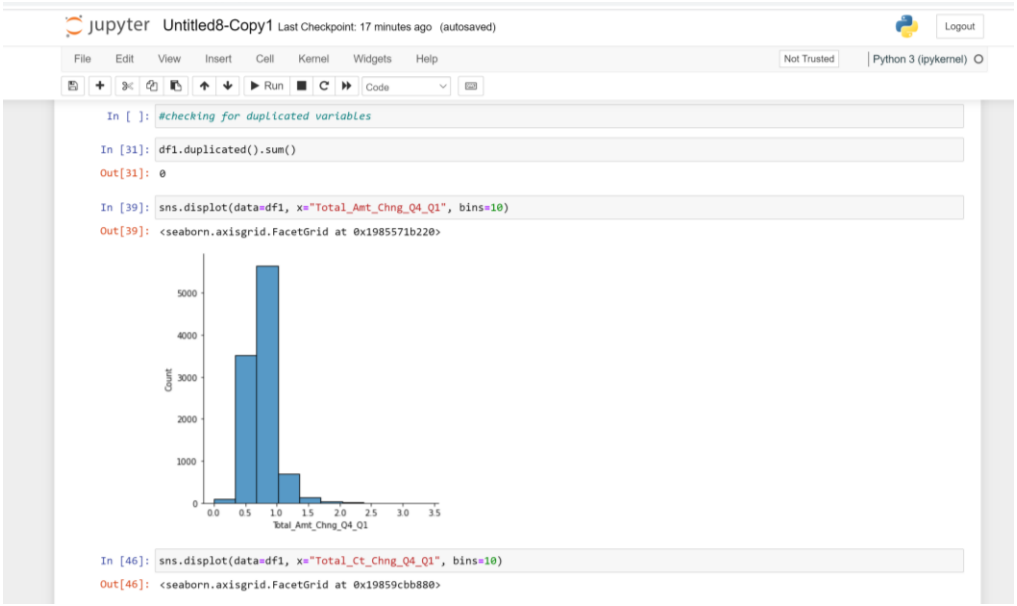
```
In [ ]: #cheching for missing values
```

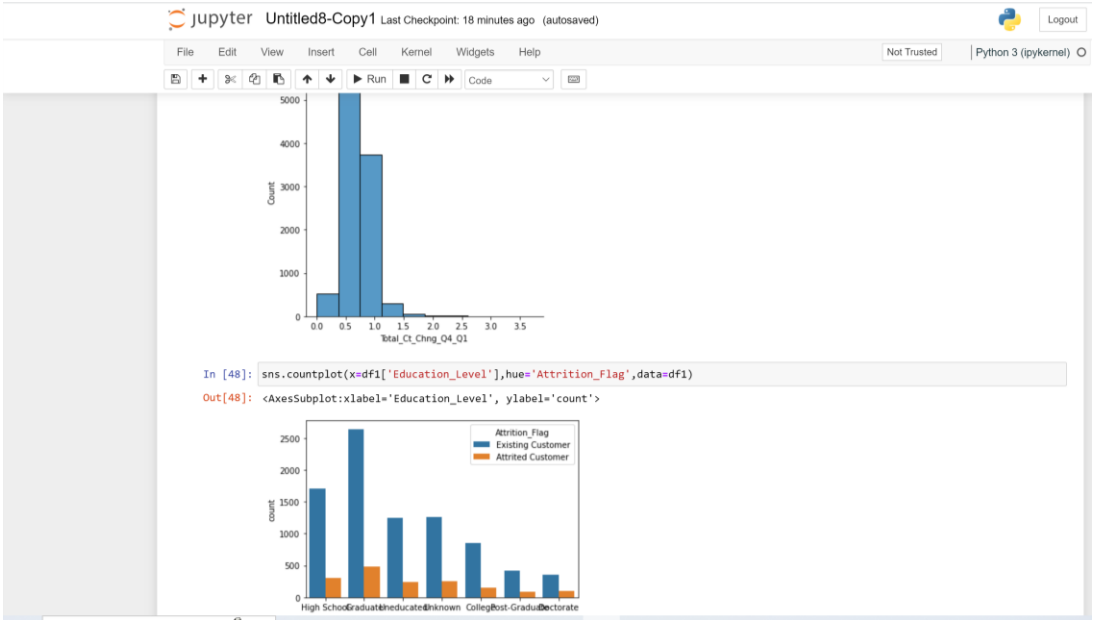
```
In [30]: df1.isnull().sum()
```

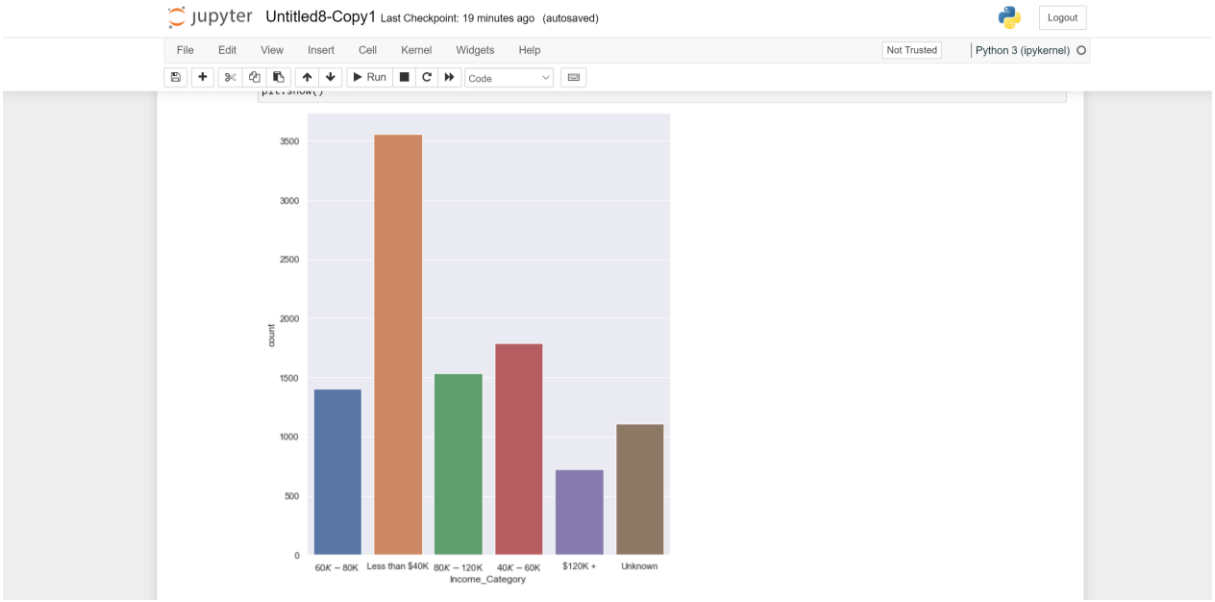
```
Out[30]: Attrition_Flag            0
Customer_Age              0
```

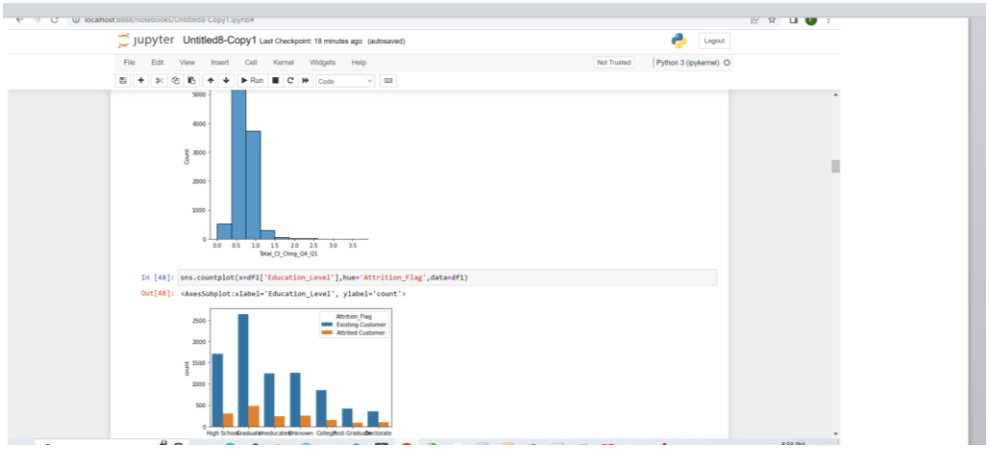


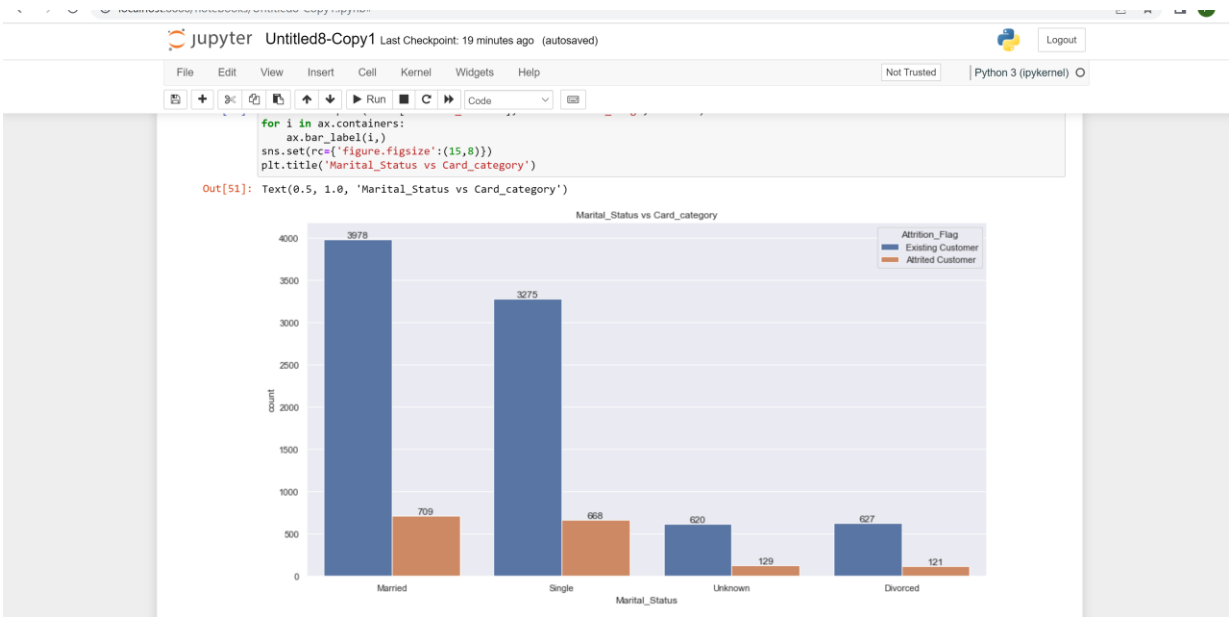

## 10.2: Data Visualization

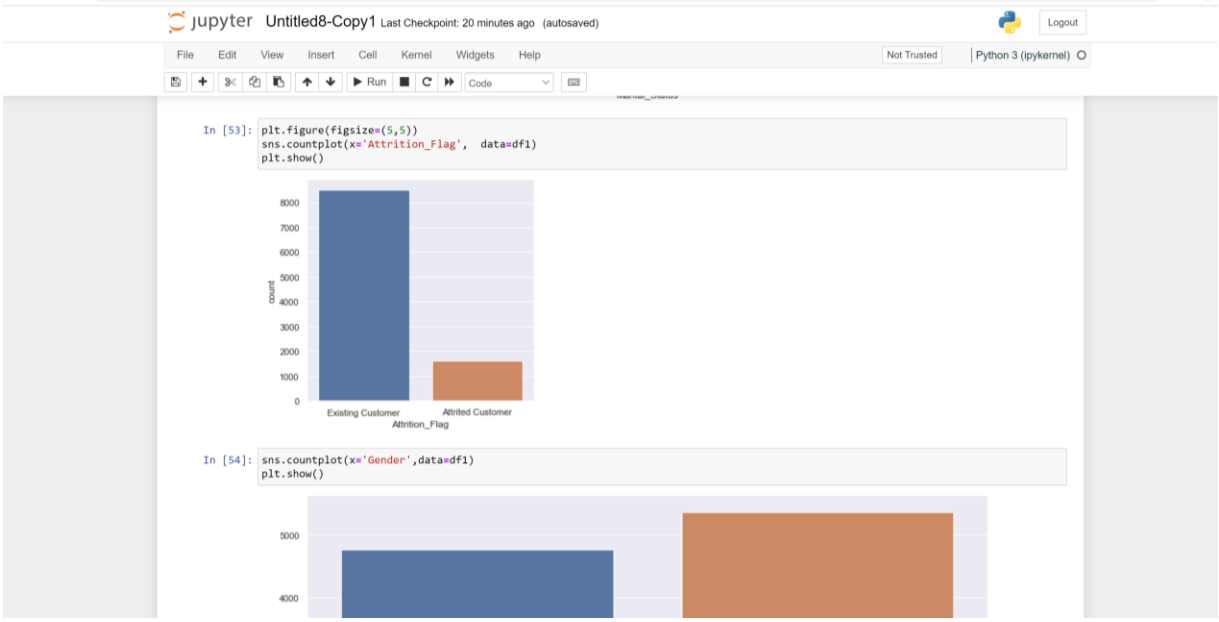


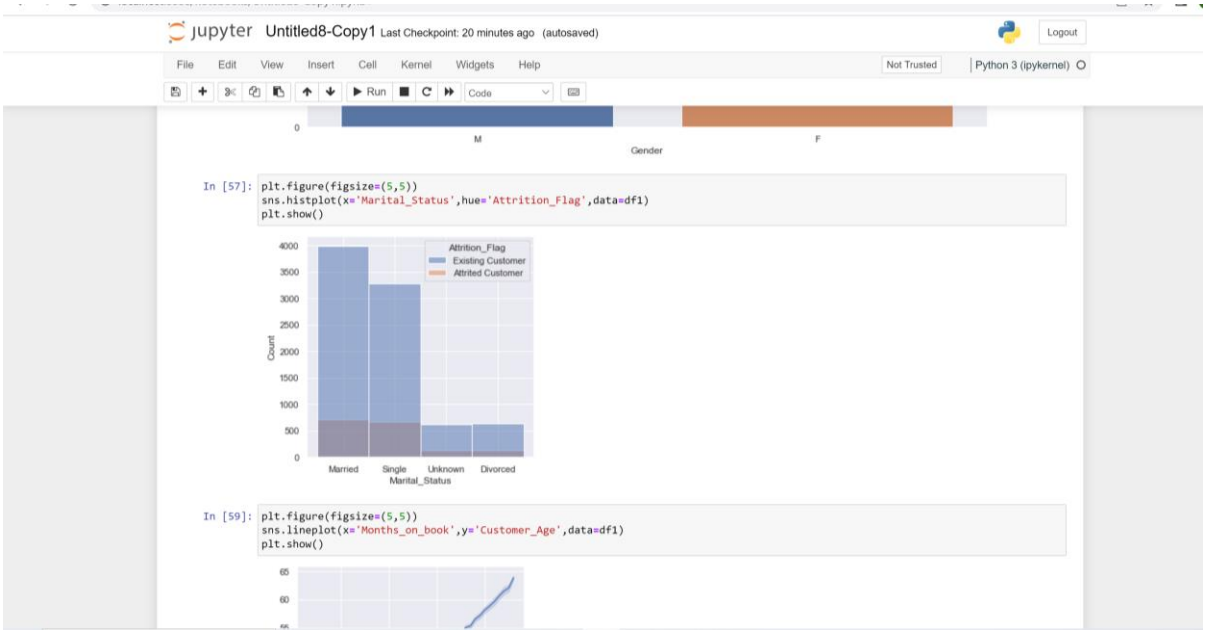

jupyter Untitled8-Copy1 Last Checkpoint: 18 minutes ago (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3 (ipykernel)
Run
Code
Count
Total_Ct_Chng_Q4_Q1
In [48]: sns.countplot(x=df1['Education_Level'],hue='Attrition_Flag',data=df1)
Out[48]: <AxesSubplot:xlabel='Education_Level', ylabel='count'>
Attrition_Flag
Existing Customer
Attrited Customer
count
High School
Graduate
Uneducated
Unknown
College
Post-Graduate
Doctorate


jupyter Untitled8-Copy1 Last Checkpoint: 19 minutes ago (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3 (ipykernel)
Run
Code
count
60K – 80K
Less than $40K
80K – 120K
40K – 60K
$120K +
Unknown
Income_Category




jupyter Untitled8-Copy1 Last Checkpoint: 19 minutes ago (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3 (ipykernel)
Run
Code
for i in ax.containers:
ax.bar_label(i,)
sns.set(rc={'figure.figsize':(15,8)})
plt.title('Marital_Status vs Card_category')
Out[51]: Text(0.5, 1.0, 'Marital_Status vs Card_category')
Marital_Status vs Card_category
Attrition_Flag
Existing Customer
Attrited Customer
3978
709
3275
668
620
129
627
121
count
Married
Single
Unknown
Divorced
Marital_Status


jupyter Untitled8-Copy1 Last Checkpoint: 20 minutes ago (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3 (ipykernel)
Run
Code
In [53]:
plt.figure(figsize=(5,5))
sns.countplot(x='Attrition_Flag', data=df1)
plt.show()
count
Existing Customer
Attrited Customer
Attrition_Flag
In [54]:
sns.countplot(x='Gender',data=df1)
plt.show()


jupyter Untitled8-Copy1 Last Checkpoint: 20 minutes ago (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3 (ipykernel)
Run
Code
0
M
F
Gender
In [57]:
plt.figure(figsize=(5,5))
sns.histplot(x='Marital_Status',hue='Attrition_Flag',data=df1)
plt.show()
Attrition_Flag
Existing Customer
Attrited Customer
4000
3500
3000
2500
2000
1500
1000
500
0
Count
Married
Single
Unknown
Divorced
Marital_Status
In [59]:
plt.figure(figsize=(5,5))
sns.lineplot(x='Months_on_book',y='Customer_Age',data=df1)
plt.show()
65
60


localhost:8888/notebooks/Untitled8-Copy1.ipynb#
jupyter Untitled8-Copy1 Last Checkpoint: 21 minutes ago (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3 (ipykernel)
Run
Code
Custom
45
40
35
30
20
30
40
50
Months_on_book
In [60]:
plt.figure(figsize=(5,5))
sns.lineplot(x='Total_Trans_Ct',y='Total_Trans_Amt',data=df1)
plt.show()
16000
14000
12000
10000
8000
6000
4000
2000
0
Total_Trans_Amt
20
40
60
80
100
120
140
Total_Trans_Ct
In [61]:
plt.figure(figsize=(5,5))
sns.scatterplot(x='Avg_Open_To_Buy',y='Credit_Limit',data=df1)

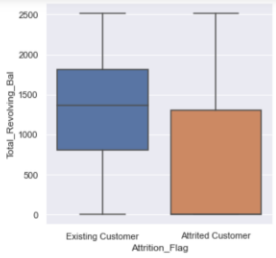
jupyter Untitled8-Copy1 Last Checkpoint: 22 minutes ago (autosaved)

Logout

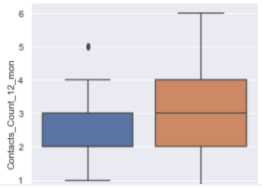
File Edit View Insert Cell Kernel Widgets Help

Not Trusted | Python 3 (ipykernel)

Run Code




In [64]:
```
plt.figure(figsize=(5,5))
sns.boxplot(x='Attrition_Flag', y='Contacts_Count_12_mon',data=df1)
plt.show()
```



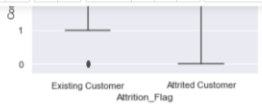

jupyter Untitled8-Copy1 Last Checkpoint: 22 minutes ago (autosaved)

Logout

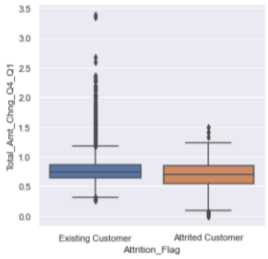
File Edit View Insert Cell Kernel Widgets Help

Not Trusted | Python 3 (ipykernel)

Run Code




In [65]:
```
plt.figure(figsize=(5,5))
sns.boxplot(x='Attrition_Flag', y='Total_Amt_Chng_Q4_Q1',data=df1)
plt.show()
```




In [87]:
```
sns.scatterplot(x='Total_Trans_Amt', y='Total_Trans_Ct', data=df1)
```

Out[87]: <AxesSubplot:xlabel='Total_Trans_Amt', ylabel='Total_Trans_Ct'>

jupyter Untitled8-Copy1 Last Checkpoint: 22 minutes ago (autosaved)

Logout

File Edit View Insert Cell Kernel Widgets Help

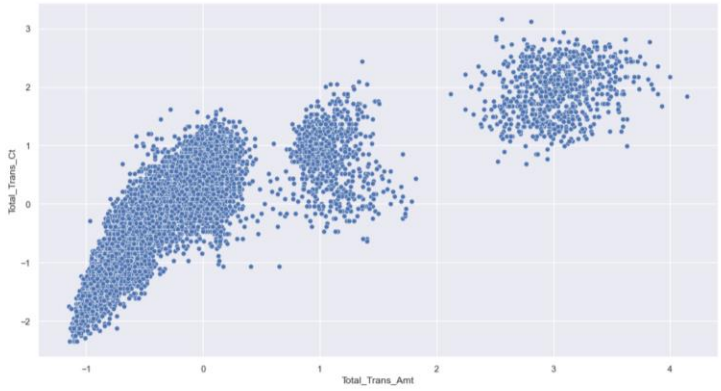
Not Trusted | Python 3 (ipykernel)

Run Code

Existing Customer Attrited Customer
Attrition_Flag

In [87]:
```
sns.scatterplot(x='Total_Trans_Amt', y='Total_Trans_Ct', data=df1)
```

Out[87]: <AxesSubplot:xlabel='Total_Trans_Amt', ylabel='Total_Trans_Ct'>




In [88]:
```
plt.hist(x='Customer_Age', data=df1, bins=10)
```

jupyter Untitled8-Copy1 Last Checkpoint: 23 minutes ago (autosaved)

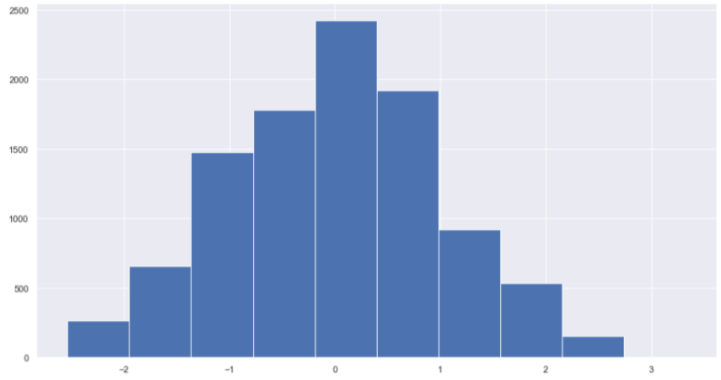
Logout

File Edit View Insert Cell Kernel Widgets Help

Not Trusted | Python 3 (ipykernel)

Run Code

In [88]:
```
plt.hist(x='Customer_Age', data=df1, bins=10)
plt.show()
```



In [89]:
```
# Plot count of attrition flag by gender
plt.figure(figsize=(10,6))
sns.countplot(data=df1, x='Attrition_Flag', hue='Gender')
plt.title('Count of Attrition Flag by Gender')
plt.xlabel('Attrition Flag')
```

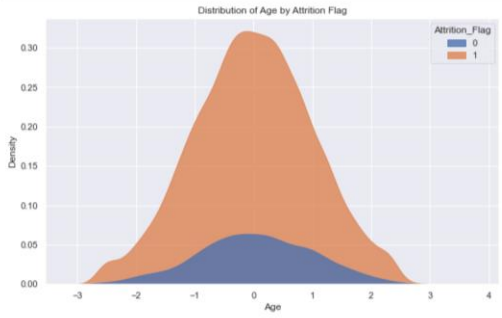
jupyter Untitled8-Copy1 Last Checkpoint: 23 minutes ago (autosaved)

Logout

File Edit View Insert Cell Kernel Widgets Help | Not Trusted | Python 3 (ipykernel)

Run Code

In [91]:
```
# Plot distribution of age by attrition flag
plt.figure(figsize=(10,6))
sns.kdeplot(data=df, x='Customer_Age', hue='Attrition_Flag', shade=True, alpha=0.8)
plt.title('Distribution of Age by Attrition Flag')
plt.xlabel('Age')
plt.ylabel('Density')
plt.show()
```



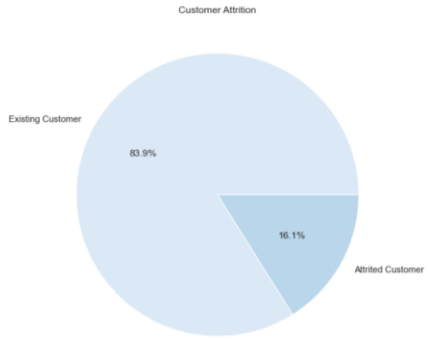

In [93]:
```
plt.pie(df.Attrition_Flag.value_counts(), labels = ["Existing Customer", "Attrited Customer"], colors = sns.color_palette("Blues'
        autopct = '%1.1f%%')
plt.title("Customer Attrition")
pass
```

jupyter Untitled8-Copy1 Last Checkpoint: 23 minutes ago (autosaved)

Logout

File Edit View Insert Cell Kernel Widgets Help | Not Trusted | Python 3 (ipykernel)

Run Code




In [95]:
```
plt.pie(df.Gender.value_counts(), labels = ["Male", "Female"], colors = sns.color_palette("Blues"),
        autopct = '%1.1f%%')
plt.title("Gender Distribution")
pass
```

Gender Distribution

jupyter Untitled8-Copy1 Last Checkpoint: 24 minutes ago (autosaved)

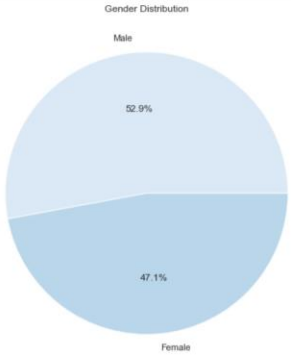
Logout

File Edit View Insert Cell Kernel Widgets Help

Not Trusted | Python 3 (ipykernel)

Run Code

In [95]:
```
plt.pie(df.Gender.value_counts(), labels = ["Male", "Female"], colors = sns.color_palette("Blues"),
        autopct = '%1.1f%%')
plt.title("Gender Distribution")
pass
```




In [ ]:
```
#headmap
```

jupyter Untitled8-Copy1 Last Checkpoint: 24 minutes ago (autosaved)

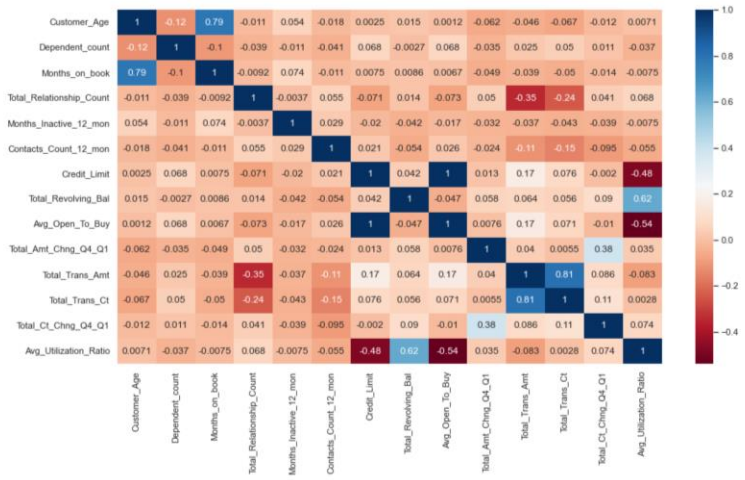
Logout

File Edit View Insert Cell Kernel Widgets Help

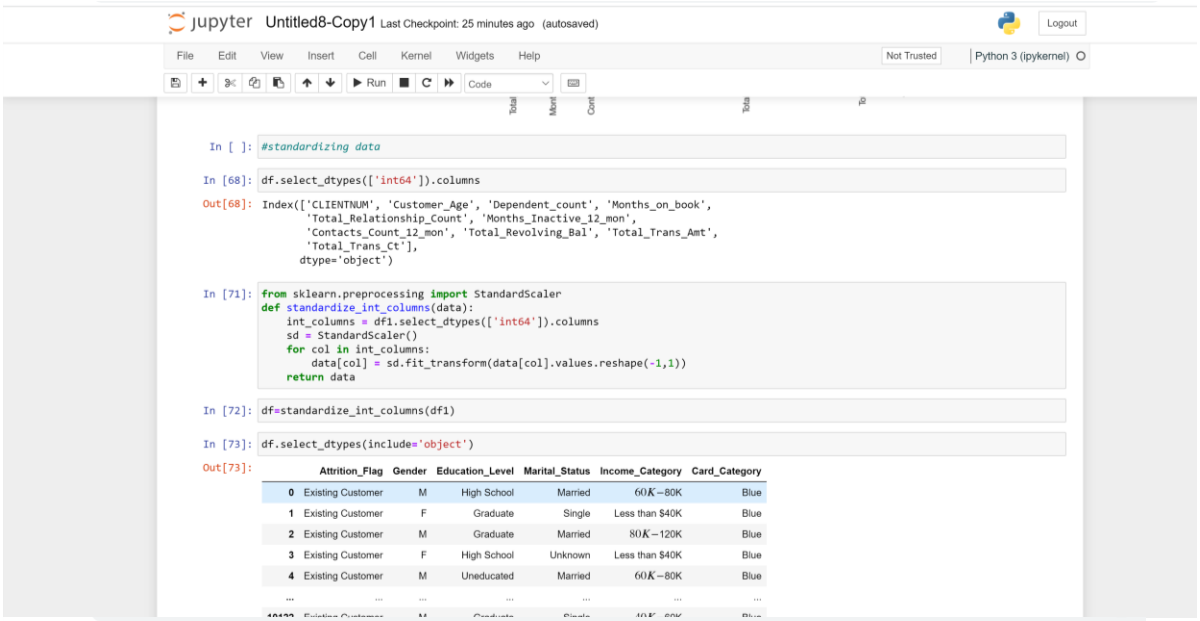
Not Trusted | Python 3 (ipykernel)

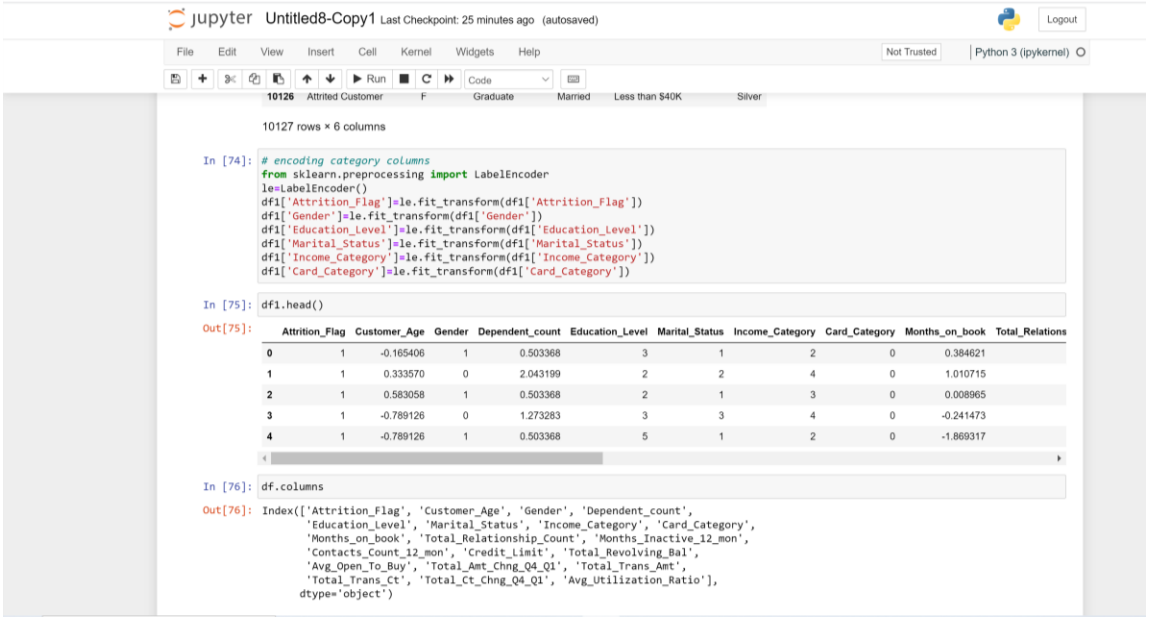
Run Code

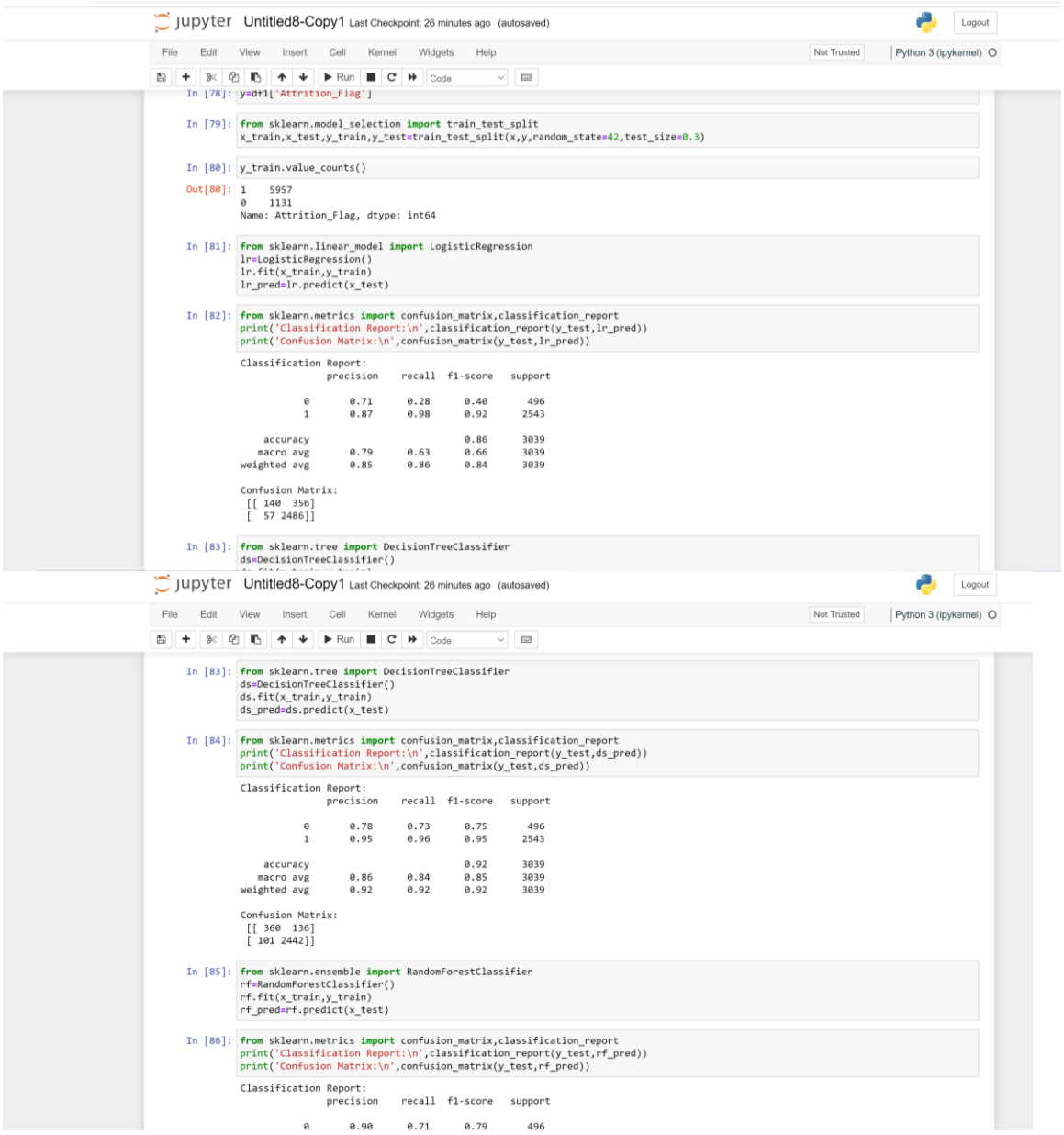
Out[52]: <AxesSubplot:>



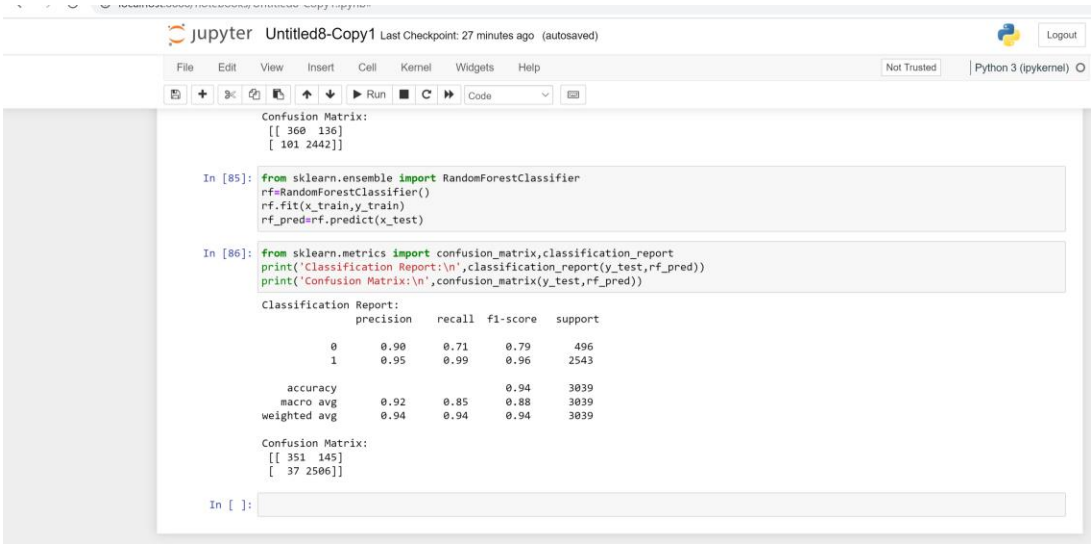
## 10.3: Machine learning Modeling




jupyter Untitled8-Copy1 Last Checkpoint: 26 minutes ago (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3 (ipykernel)
Run
Code
In [78]: y=df1['Attrition_Flag']
In [79]: from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x,y,random_state=42,test_size=0.3)
In [80]: y_train.value_counts()
Out[80]: 1 5957
0 1131
Name: Attrition_Flag, dtype: int64
In [81]: from sklearn.linear_model import LogisticRegression
lr=LogisticRegression()
lr.fit(x_train,y_train)
lr_pred=lr.predict(x_test)
In [82]: from sklearn.metrics import confusion_matrix,classification_report
print('Classification Report:\n',classification_report(y_test,lr_pred))
print('Confusion Matrix:\n',confusion_matrix(y_test,lr_pred))
Classification Report:
precision recall f1-score support
0 0.71 0.28 0.40 496
1 0.87 0.98 0.92 2543
accuracy 0.86 3039
macro avg 0.79 0.63 0.66 3039
weighted avg 0.85 0.86 0.84 3039
Confusion Matrix:
[[ 140 356]
[ 57 2486]]
In [83]: from sklearn.tree import DecisionTreeClassifier
ds=DecisionTreeClassifier()


jupyter Untitled8-Copy1 Last Checkpoint: 26 minutes ago (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3 (ipykernel)
Run
Code
In [83]: from sklearn.tree import DecisionTreeClassifier
ds=DecisionTreeClassifier()
ds.fit(x_train,y_train)
ds_pred=ds.predict(x_test)
In [84]: from sklearn.metrics import confusion_matrix,classification_report
print('Classification Report:\n',classification_report(y_test,ds_pred))
print('Confusion Matrix:\n',confusion_matrix(y_test,ds_pred))
Classification Report:
precision recall f1-score support
0 0.78 0.73 0.75 496
1 0.95 0.96 0.95 2543
accuracy 0.92 3039
macro avg 0.86 0.84 0.85 3039
weighted avg 0.92 0.92 0.92 3039
Confusion Matrix:
[[ 360 136]
[ 101 2442]]
In [85]: from sklearn.ensemble import RandomForestClassifier
rf=RandomForestClassifier()
rf.fit(x_train,y_train)
rf_pred=rf.predict(x_test)
In [86]: from sklearn.metrics import confusion_matrix,classification_report
print('Classification Report:\n',classification_report(y_test,rf_pred))
print('Confusion Matrix:\n',confusion_matrix(y_test,rf_pred))
Classification Report:
precision recall f1-score support
0 0.90 0.71 0.79 496


jupyter Untitled8-Copy1 Last Checkpoint: 27 minutes ago (autosaved)
Logout
File Edit View Insert Cell Kernel Widgets Help
Not Trusted
Python 3 (ipykernel)
Run
Code
Confusion Matrix:
[[ 360 136]
[ 101 2442]]
In [85]:
from sklearn.ensemble import RandomForestClassifier
rf=RandomForestClassifier()
rf.fit(x_train,y_train)
rf_pred=rf.predict(x_test)
In [86]:
from sklearn.metrics import confusion_matrix,classification_report
print('Classification Report:\n',classification_report(y_test,rf_pred))
print('Confusion Matrix:\n',confusion_matrix(y_test,rf_pred))
Classification Report:
precision recall f1-score support
0 0.90 0.71 0.79 496
1 0.95 0.99 0.96 2543
accuracy 0.94 3039
macro avg 0.92 0.85 0.88 3039
weighted avg 0.94 0.94 0.94 3039
Confusion Matrix:
[[ 351 145]
[ 37 2506]]
In [ ]: