BANK CUSTOMER’S CHURN

A report submitted in partial fulfillment of the requirements for the Award of Degree of

BACHELOR OF TECHNOLOGY

in

COMPUTER SCIENCE AND ENGINEERING

By

GOGULA DEEPAK

Regd. No.: 20B91A0593

Under Supervision of Mr. Gundala Nagaraju
Henotic Technology Pvt Ltd, Hyderabad
(Duration: 7th July, 2022 to 6th September, 2022)

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
SAGI RAMA KRISHNAM RAJUENGINEERING COLLEGE

(An Autonomous Institution)

Approved by AICTE, NEW DELHI and Affiliated to JNTUK, Kakinada

CHINNA AMIRAM, BHIMAVARAM,

ANDHRA PRADESH

Table of Contents

Abstract

Banking is one of the highly competitive sectors where customer relations is of utmost importance for any bank. Each customer is considered as a customer for life by the banks. The term “Customer Churn” refers to the state in which the customer or the subscriber stops involving in business transactions with a company or a service provider. To deal with this, the paper presents the work done towards predicting the customer churn rate, using machine learning models which will indicate whether a customer will leave the bank or not based on many factors, this in turn will help the bank in knowing which category of customers generally tend to leave the bank. Further the banks can bring in exciting offers so that it can retain its customers. In this predictive process popular models such as logistic regression, decision trees, random forest and other boosting techniques have been used to achieve a decent level of accuracy, for the banks to rely upon.

Keywords: Banking, Gradient Boosting, Customer Churn, Machine Learning, Random Forest Classifier.

1.0 Introduction

With the increasing power of computer technology, companies and institutions can nowadays store large amounts of data at reduced cost. The amount of available data is increasing exponentially and cheap disk storage makes it easy to store data that previously was thrown away. There is a huge amount of information locked up in databases that is potentially important but has not yet been explored. The growing size and complexity of the databases makes it hard to analyse the data manually, so it is important to have automated systems to support the process. Hence there is the need of computational tools able to treat these large amounts of data and extract valuable information.

In this context, Data Mining provides automated systems capable of processing large amounts of data that are already present in databases. Data Mining is used to automatically extract important patterns and trends from databases seeking regularities or patterns that can reveal the structure of the data and answer business problems. Data Mining includes learning techniques that fall into the field of Machine learning. The growth of databases in recent years brings data mining at the forefront of new business technologies.

Customer churn, also known as customer attrition is the term coined for the probability of whether an existing customer continues his/her transactions with the organization or not. The probability factor of this parameter depends on numerous other factors in various industries like banking, telecom and few other sectors. In intense market scenarios that are prevailing today in every sector, it becomes important for the organizations to keep a track of customer churn and the various reasons causing the customer to stop their transactions with the company. Almost every organization is well-versed with the concept that retention of the existing customers will save a lot of money as trying to acquire new customers will cost five to six times the cost to retain an existing customer. Therefore, each organization out in the market started to understand and analyse the various factors that might be the cause for a customer or client to leave the organization’s business.

Once importance of customers and customer churn was identified by various leading companies, they started to gather data on customer behaviour like how often does a customer purchase a product, etc, with this, the collection, storage and managing of the customer data on a time basis became really crucial for the companies. Thus, the decision-making process shifted from being an event-driven approach to a data-driven approach. New technologies like Big Data, Cloud storage and machine learning supported the companies’ data gathering, processing and analysis phase. Therefore, the whole process shifted to statistical analysis as opposed to predictive analysis.

This paper deals with the customer churn in a banking sector and highlights the various factors that affect the attrition of customers from the bank. It also sheds some light on how a bank can predict the rate of customer churn by making use of well-known and popularly used machine learning models.

1.1 What are the different types of Machine Learning?

Machine learning is a subset of AI, which enables the machine to automatically learn from data, improve performance from past experiences, and make predictions. Machine learning contains a set of algorithms that work on a huge amount of data. Data is fed to these algorithms to train them, and on the basis of training, they build the model & perform a specific task.

These ML algorithms help to solve different business problems like Regression, Classification, Forecasting, Clustering, and Associations, etc.

Based on the methods and way of learning, machine learning is divided into mainly four types, which are:

1. Supervised Machine Learning
2. Unsupervised Machine Learning
3. Reinforcement Learning

1. Supervised Machine Learning

Supervised Machine Learning is based on supervision. It means in the supervised learning technique, we train the machines using the "labelled" dataset, and based on the training, the machine predicts the output. Here, the labelled data specifies that some of the inputs are already mapped to the output. More preciously, we can say; first, we train the machine with the input and corresponding output, and then we ask the machine to predict the output using the test dataset.

Let's understand supervised learning with an example. Suppose we have an input dataset of cats and dog images. So, first, we will provide the training to the machine to understand the images, such as the shape & size of the tail of cat and dog, Shape of eyes, colour, height (dogs are taller, cats are smaller), etc. After completion of training, we input the picture of a cat and ask the machine to identify the object and predict the output. Now, the machine is well trained, so it will check all the features of the object, such as height, shape, colour, eyes, ears, tail, etc., and find that it's a cat. So, it will put it in the Cat category. This is the process of how the machine identifies the objects in Supervised Learning.

The main goal of the supervised learning technique is to map the input variable(x) with the output variable(y). Some real-world applications of supervised learning are Risk Assessment, Fraud Detection, Spam filtering, etc.

2. Unsupervised Machine Learning

Unsupervised Machine Learning is different from the Supervised learning technique; as its name suggests, there is no need for supervision. It means, in unsupervised machine learning, the machine is trained using the unlabelled dataset, and the machine predicts the output without any supervision. In unsupervised learning, the models are trained with the data that is neither classified nor labelled, and the model acts on that data without any supervision

The main aim of the unsupervised learning algorithm is to group or categories the unsorted dataset according to the similarities, patterns, and differences. Machines are instructed to find the hidden patterns from the input dataset.

Let's take an example to understand it more preciously; suppose there is a basket of fruit images, and we input it into the machine learning model. The images are totally unknown to the model, and the task of the machine is to find the patterns and categories of the objects. So, now the machine will discover its patterns and differences, such as colour difference, shape difference, and predict the output when it is tested with the test dataset.

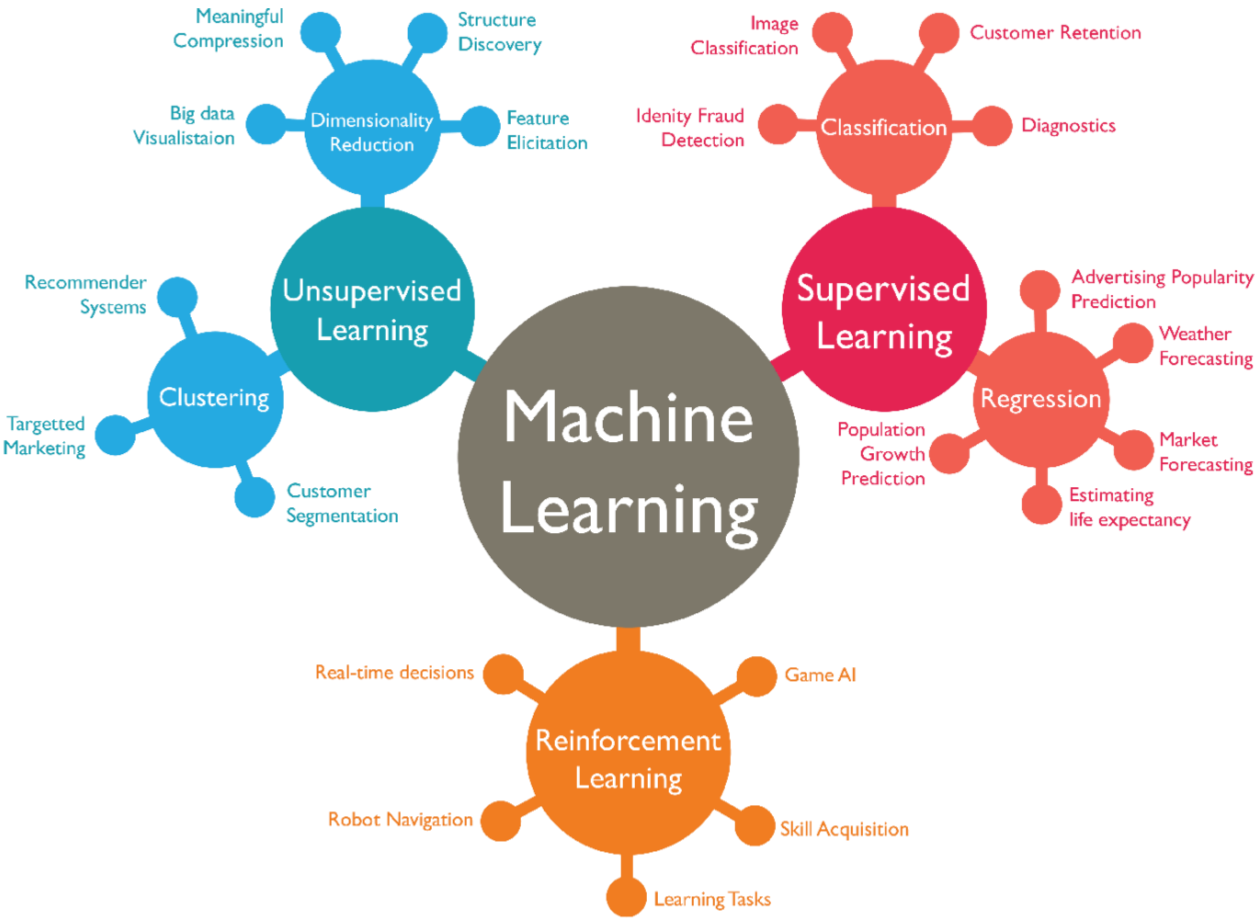
3. Reinforcement Machine Learning

Reinforcement learning works on a feedback-based process, in which an AI agent (A software component) automatically explore its surrounding by hitting & trail, taking action, learning from experiences, and improving its performance. Agent gets rewarded for each good action and get punished for each bad action; hence the goal of reinforcement learning agent is to maximize the rewards.

In reinforcement learning, there is no labelled data like supervised learning, and agents learn from their experiences only. The Reinforcement Learning process is similar to a human being; for example, a child learns various things by experiences in his day-to-day life. An example of reinforcement learning is to play a game, where the Game is the environment, moves of an agent at each step define states, and the goal of the agent is to get a high score. Agent receives feedback in terms of punishment and rewards.

Due to its way of working, reinforcement learning is employed in different fields such as Game theory, Operation Research, Information theory, multi-agent systems.

A reinforcement learning problem can be formalized using Markov Decision Process (MDP). In MDP, the agent constantly interacts with the environment and performs actions; at each action, the environment responds and generates a new state.



1.2 Benefits of Using Machine Learning in Bank Customer's churn

1. Identify at-risk customers

For any business that wants to enjoy the benefits of customers churn prediction, machine learning opens dozens of opportunities. Machine Learning is able to analyze client behavior and measure the probability of churning. In Particular, to precisely identify churn rate, machine learning algorithms can be trained to learn the behavior patterns of clients/partners who have already canceled their contracts or any other relationships with a particular company and compare them with the existing ones. Then correlations between the actions of active and inactive clients are done. As a result, the algorithm recognizes the customers that are more likely to leave.

2. Identify pain points

Different companies lose their clients for different reasons. In most cases, there are numerous “pain points”, which remain unknown for product owners. From the bad quality and absent features to unpleasant design and poor costumer service-there are a lot of details which you do not take into account that your clients do. Even if your product is almost perfect, you can still reward your new costumers with some attractive discounts and offers and ignore your loyal ones. When a business applies churn prediction, machine learning can do analysis and forecasts based not only on costumer behavior but also on the brand’s.

3. Identify methods to implement:

After the root cause of client churn has been identified, companies can reconsider and rebuild their products and change their business strategy accordingly. Transformed data and automated flow can be used in CRM and marketing automation systems. However, this doesn’t mean that using machine learning for churn prediction is about building a certain model for certain task. It is more about domain knowledge and an ability to deliver the best possible solution based on learning data, processes and behavior.

1.3 About Industry (Bank Customer’s Churn)

The development of the banking sector mostly depends on its valuable customers. So, customer churn analysis is needed to determine customers whether they are at risk of leaving or worth retaining. From organizational point of view, gaining new customers is usually more difficult or more expensive than retaining existing customers.

The role of ICT in the banking sector is a crucial part of the development of nations. The development of the banking sector mostly depends on its valuable customers. So, customer churn analysis is needed to determine customers whether they are at risk of leaving or worth retaining. From organizational point of view, gaining new customers is usually more difficult or more expensive than retaining existing customers. So, customer churn prediction has been popular in the banking industry. By reducing customer churn or attrition, the commercial banks gain not only more profits but also enhancing core competitiveness among the competitors. Although many researchers proposed many single prediction models and some hybrid model, but accuracy is still weak and computation time of some algorithms is still increased. In this research, churn prediction model of classifying bank customer is built by using the hybrid model of k-means and Support Vector Machine data mining methods on bank customer churn dataset to overcome the instability and limitations of single prediction model and predict churn trend of high value users.

1.3.1 AI / ML Role in Bank Customer's Churn

Banks know that AI and Machine Learning (ML) can help reduce churn through early intervention. However, building propensity-to-churn models takes time and expertise, and many banks find that not all of their product lines get the attention they deserve due to resource constraints. This lack of bandwidth can often lead to less than optimal customer retention outcomes. The AI & Analytics Engine can give bandwidth back to data teams so that more banks can maximize the output of their teams to prototype and productionize ML models, so retention strategies can be planned.

The landscape is rapidly changing for the banking and finance sectors. Previously, customer churn rates were relatively low. However, with the increasing options in the market and proliferation of fintech, customer churn is an increasingly important battleground for financial institutions. Over their customer lifetime, customers generate fees on transactions, banking fees, credit cards, home loans, personal loans, and so much more. Traditionally, simple churn analysis uses rules based on known behaviors to identify churn risks. Rules-based systems can be inflexible and overlook customers who do churn and generate false positives. It means expensive incentives are provided to customers who were at low risk of churning.

ML for customer churn prediction is ideal, particularly for banks. This is due to the problem consisting of complex data over time, and many interactions between diverse customer behaviors. Without ML, it can be difficult and inefficient for bank employees to identify and act in a systematic way.

2.0 Bank Customer's Churn

Banking is one of the highly competitive sectors where customer relations is of utmost importance for any bank. Each customer is considered as a customer for life by the banks. The term "Customer Churn" refers to the state in which the customer or the subscriber stops involving in business transactions with a company or a service provider. Churn is a critical issue for banks. Because of their high-fixed cost structure, even a tenth of a percent reduction in churn can have a massive impact on a bank's bottom line. From working with a large bank (5,000 branches) in one the world's fastest growing markets, and gathering over 24 million of their customer data points, we learnt that machine learning and Big Data can do precisely this. In fact, machine learning is perfectly suited to predicting churn because of its very binary nature (e.g. customers either churn or don't churn). In addition to this, banking data is unique in that it encompasses both static and temporal data for each customer. This makes it relatively easy to predict present (and future) state based on historical data.

A dataset which contain some customers who are withdrawing their account from the bank due to some loss and other issues with the help this data we try to analyse and maintain accuracy.

The main factors for Bank Customer's Churn prediction are Credit Score, Geography, Age, Tenure, Balance, Number of Products, Whether the customer has credit card or not , Whether the customer is an active member or not , Estimated Salary of the Customer.

2.1 Main Drivers for Bank Customer's Churn Quote Analysis

Predictive modelling allows for simultaneous consideration of many variables and quantification of their overall effect. When a large number of customer's churn are analysed, patterns regarding the characteristics of the customer's churn that drive loss development begin to emerge.

The following are the main drivers which influencing the Bank Customer's Churn Analytics:

- **Bank Client data**
- ✓ Credit Score
- ✓ Geography
- ✓ Gender
- ✓ Age
- ✓ Tenure
- ✓ Balance
- ✓ Number of Products
- ✓ Has Credit Card
- ✓ Is active Member
- ✓ Estimated Salary

- **Financial Status**
- ✓ Government Employee
- ✓ Business
- ✓ Doctor
- ✓ Lawyer
- ✓ Engineer
- **Tenure**
- ✓ Land or Buildings
- ✓ office
- ✓ status
- **Has Credit Card**

2.2 Internship Project - Data Link

The internship project data has taken from Kaggle and the link is

https://www.kaggle.com/datasets/santoshd3/bank-customers

Context

A dataset which contain some customers who are withdrawing their account from the bank due to some loss and other issues with the help this data we try to analyse and maintain accuracy.

This project proposes the use of machine learning and data mining techniques to differentiate the customer who are in the risk of being churned and customers who are happy and satisfied with the products and services of the bank. The algorithms and various technologies employed to predict the customer churn for a bank are discussed in the following sections in- detail. This paper discusses about the use of several data pre-processing steps and machine learning algorithms and techniques like logistic regression, AdaBoost, etc, and compares each of their prediction power by taking into consideration the accuracy of each machine learning models and selecting the best model with highest accuracy, to forecast the customer churn. A front-end application which is a simple webpage is provided to the end-user to enter the details of a customer to check whether he/she is at a risk of being churned or is happy and satisfied with the products and services of the bank.

The following are the software requirements for building the model and web application, to get the desired level of results: Python Programming, Jupyter

Notebook, Visual Studio Code, Streamlit. The following are Python libraries utilized to obtain the results: Pandas, NumPy, Matplotlib, Seaborn.

3.0 AI/ML Modeling and Results

3.1 Problem Statement:

A dataset which contain some customers who are withdrawing their account from the bank due to some loss and other issues with the help this data we try to analyse and maintain accuracy.

Predictive models are most effective when they are constructed using a Bank own historical Customer's Churn data since this allows the model to recognize the specific nature of a bank's exposure as well as its Customer's churn practices. The construction of the model also involves input from the bank throughout the process, as well as consideration of industry leading customer's churn practices and benchmarks.

Predictive modelling can be used to quantify the impact to the churn department resulting from the failure to exited or not for customer's churn service leading practices. It can also be used to identify the root cause of customer's churn. Proper use of predictive modelling will allow for potential savings across these dimensions:

- Identifying old customers without loss and developing new products and making new strategic decisions for retaining customers. This study focuses on the customer churn analysis, that is a significant topic in banks customer relationship management.
- Service of the bank can be identified using Bank Customer's Churn dataset.

3.2 Data Science Project Life Cycle

Data Science is a multidisciplinary field of study that combines programming skills, domain expertise and knowledge of statistics and mathematics to extract useful insights and knowledge from data.

Data Science Lifecycle revolves around the use of machine learning and different analytical strategies to produce insights and predictions from information in order to acquire a commercial enterprise objective. The complete method includes a number of steps like data cleaning, preparation, modelling, model evaluation, etc. It is a lengthy procedure and may additionally take quite a few months to complete. So, it is very essential to have a generic structure to observe for each and every hassle at hand. The globally mentioned structure in fixing any analytical problem is referred to as a Cross Industry Standard Process for Data Mining or CRISP-DM framework.

The following are some primary motives for the use of Data science technology:

1. It helps to convert the big quantity of uncooked and unstructured records into significant insights.
2. It can assist in unique predictions such as a range of surveys, elections, etc.
3. It also helps in automating transportation such as growing a self-driving car, we can say which is the future of transportation.
4. Companies are shifting towards Data science and opting for this technology. Amazon, Netflix, etc, which cope with the big quantity of data, are the use of information science algorithms for higher consumer experience.

Data science life cycle is nothing but a repetitive set of steps that you need to take to complete and deliver a project/product to your client. Although the data science projects and the teams involved in deploying and developing the model will be different, every data science life cycle will be slightly different in every other company. However, most of the data science projects happen to follow a somewhat similar process.

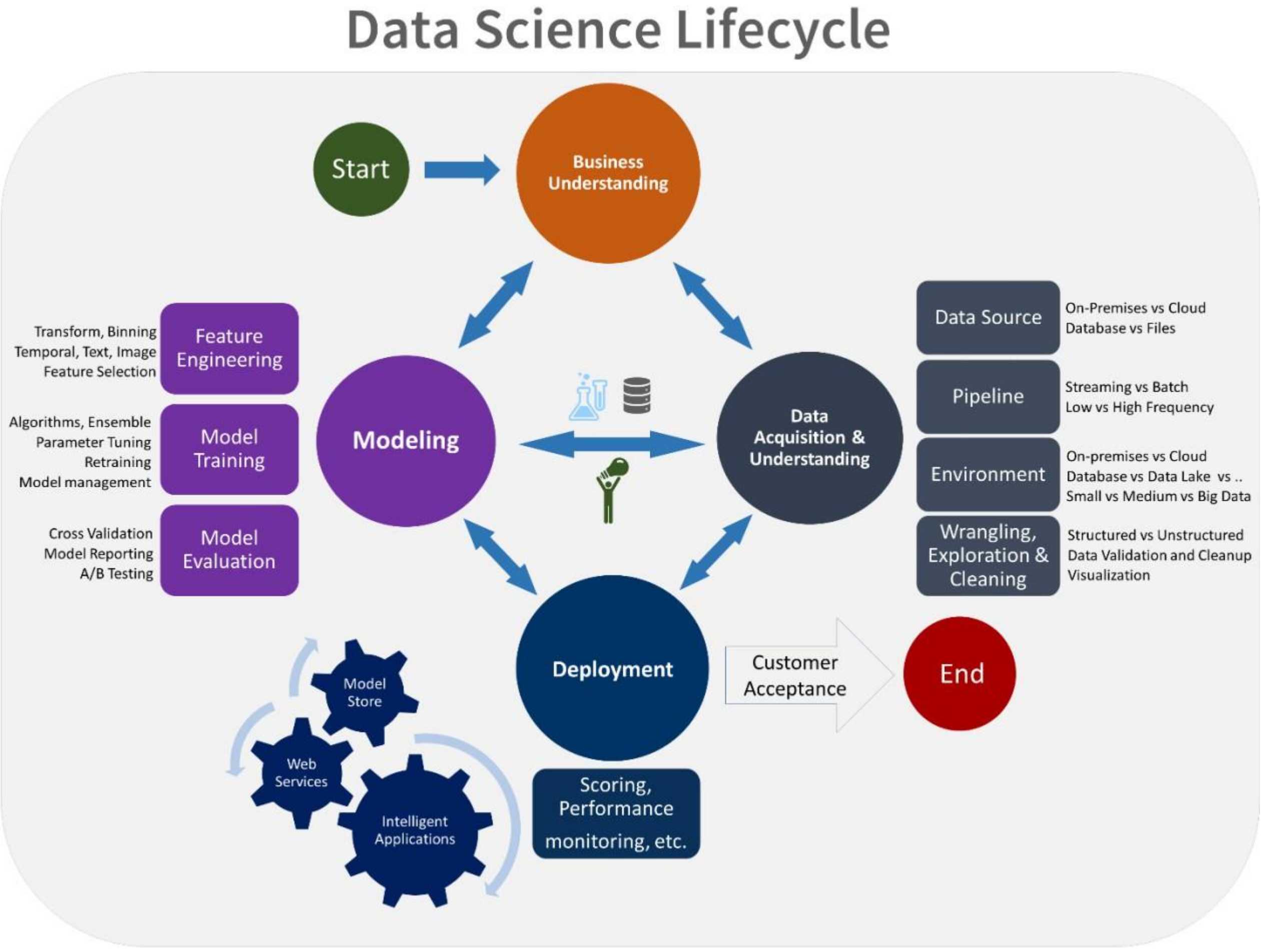
In order to start and complete a data science-based project, we need to understand the various roles and responsibilities of the people involved in building, developing the project. Let us take a look at those employees who are involved in a typical data science project:

Who Are Involved in The Projects:

1. Business Analyst
2. Data Analyst
3. Data Scientists
4. Data Engineer
5. Data Architect
6. Machine Learning Engineer

Life Cycle of Data Science:

1. Business Understanding
2. Data Understanding
3. Preparation of Data
4. Exploratory Data Analysis
5. Data Modeling
6. Model Evaluation
7. Model Deployment



3.2.1 Data Exploratory Analysis

Exploratory data analysis (EDA) is used by data scientists to analyze and investigate data sets and summarize their main characteristics, often employing data visualization methods. It helps determine how best to manipulate data sources to get the answers you need, making it easier for data scientists to discover patterns, spot anomalies, test a hypothesis, or check assumptions. EDA is primarily used to see what data can reveal beyond the formal modeling or hypothesis testing task and provides a provides a better understanding of data set variables and the relationships between them. It can also help determine if the statistical techniques you are considering for data analysis are appropriate. Originally developed by American mathematician John Tukey in the 1970s, EDA techniques continue to be a widely used method in the data discovery process today.

The main purpose of EDA is to help look at data before making any assumptions. It can help identify obvious errors, as well as better understand patterns within the data, detect outliers or anomalous events, find interesting relations among the variables. Data scientists can use exploratory analysis to ensure the results they produce are valid and applicable to any desired business outcomes and goals. EDA also helps stakeholders by confirming they are asking the right questions. EDA can help answer questions about standard deviations, categorical variables, and confidence intervals. Once EDA is complete and insights are drawn, its features can then be used for more sophisticated data analysis or modeling, including machine learning.

Exploratory data analysis has been done on the data to look for relationship and correlation between different variables and to understand how they impact or target variable.

3.2.2 Data Pre-processing

Data scientists can use exploratory analysis to ensure the results they produce are valid and applicable to any desired business outcomes and goals. EDA also helps stakeholders by confirming they are asking the right questions. EDA can help answer questions about standard deviations, categorical variables, and confidence intervals. Once EDA is complete and insights are drawn, its features can then be used for more sophisticated data analysis or modeling, including machine learning.

Data preprocessing is a process of preparing the raw data and making it suitable for a machine learning model. It is the first and crucial step while creating a machine learning model. When creating a machine learning project, it is not always a case that we come across the clean and formatted data. And while doing any operation with data, it is mandatory to clean it and put in a formatted way. So for this, we use data preprocessing task.

A real-world data generally contains noises, missing values, and maybe in an unusable format which cannot be directly used for machine learning models. Data preprocessing is required tasks for cleaning the data and making it suitable for a machine learning model which also increases the accuracy and efficiency of a machine learning model.

It involves below steps:

- Getting the dataset
- Importing libraries
- Importing datasets
- Finding Missing Data
- Encoding Categorical Data
- Splitting dataset into training and test set
- Feature scaling

We removed variables which does not affect our target variable(Exited_Target)as they may add noise and also increase our computation time, we checked the data for anomalous data points and outliers. We did principal component analysis on the data set to filter out unnecessary variables and to select only the important variables which have greater correlation with our target variable.

3.2.2.1 Check the Duplicate and low variation data

Duplicates:

Feature Selection is the process of reducing the number of input variables when developing a predictive model. It reduces the computational cost of model training and also improves the performance of the model.

These can be of two types:

1. **Duplicate Values**: When two features have the same set of values

2. **Duplicate Index**: When the value of two features are different, but they occur at the same index

- Duplicate values->Same Value for each record.
- Duplicate Index -> Value of two Features are different but they occur at the same index.

Variation data:

In predictive analytics, we build machine learning models to make predictions on new, previously unseen samples. The whole purpose is to be able to predict the unknown. But the models cannot just make predictions out of the blue. We show some samples to the model and train it. Then we expect the model to make predictions on samples from the same distribution.There is no such thing as a perfect model so the model we build and train will have errors. There will be differences between the predictions and the actual values. The performance of a model is inversely proportional to the difference between the actual values and the predictions. The smaller the difference, the better the model. Our goal is to try to minimize the error. We cannot eliminate the error but we can reduce it.The part of the error that can be reduced has two components: **Bias** and **Variance**

The performance of a model depends on the balance between bias and variance.

Variance occurs when the model is highly sensitive to the changes in the independent variables (features). The model tries to pick every detail about the relationship between features and target. It even learns the noise in the data which might randomly occur. A very small change in a feature might change the prediction of the model. Thus, we end up with a model that captures each and every detail on the training set so the accuracy on the training set will be very high. However, the accuracy of new, previously unseen samples will not be good because there will always be different variations in the features. This situation is also known as **overfitting**. The model overfits to the training data but fails to generalize well to the actual relationships within the dataset. The accuracy on the samples that the model actually sees will be very high but the accuracy on new samples will be very low. Consider the same example that we discussed earlier. If we try to model the relationship with the red curve in the image below, the model overfits. As you can see, it is highly sensitive and tries to capture every variation.

Low variance: *tells you that the smallest change in the data set causes the results to change in the target function.*

Examples of low variance in machine learning include linear regression, linear analysis, linear logic regression, and logistic regression.

3.2.2.2 Identify and address the missing variables

When you start working on any data science project the data you are provided is never clean. One of the most common issue with any data set are missing values. Most of the machine learning algorithms are not able to handle missing values. The missing values needs to be addressed before proceeding to applying any machine learning algorithm.

Missing values can be handled in different ways depending on if the missing values are continuous or categorical.

Finding Missing Values:

Step 1: Load the data frame and study the structure of the data frame.

- To find out how many of the columns are categorical and numerical we can use pandas “dtypes” to get the different data types and you can use pandas “value_counts()” function to get count of each data type. Value_counts groups all the unique instances and gives the count of each of those instances.

Step 2: Separate categorical and numerical columns in the data frame

- The easiest way to achieve this step is through filtering out the columns from the original data frame by data type. By using “dtypes” function and equality operator you can get which columns are objects (categorical variable) and which are not.
- To get the column names of the columns which satisfy the above conditions we can use “df.columns”.
- Then One for for categorical variables and one for non-categorical variables.

Step 3: Find the missing values

- Finding the missing values is the same for both categorical and continuous variables. We will use “num_vars” which holds all the columns which are not object data type.
- **df[num_vars]** will give you all the columns in “num_vars” which consists of all the columns in the data frame which are not object data type.
- We can use pandas **“isnull()”** function to find out all the fields which have missing values. This will return True if a field has missing values and false if the field does not have missing values.
- To get how many missing values are in each column we use **sum()** along with **isnull()** which is shown below.

Address the Missing Values:

There are many ways to address the missing values in machine learning. some of them are:

1. **Deleting the rows with Missing values:**

Missing values can be handled by deleting the rows or columns having null values. If columns have more than half of the rows as null then the entire column can be dropped. The rows which are having one or more columns values as null can also be dropped.

2. **Impute missing values for continuous variable**

Columns in the dataset which are having numeric continuous values can be replaced with the mean, median, or mode of remaining values in the column. This method can prevent the loss of data compared to the earlier method. Replacing the above two approximations (mean, median) is a statistical approach to handle the missing values.

3. **Impute missing values for categorical variable**

When missing values is from categorical columns (string or numerical) then the missing values can be replaced with the most frequent category. If the number of missing values is very large then it can be replaced with a new category.

4. **Using Algorithms that support missing values:**

All the machine learning algorithms don’t support missing values but some ML algorithms are robust to missing values in the dataset. The k-NN algorithm can ignore a column from a distance measure when a value is missing. Naive Bayes can also support missing values when making a prediction. These algorithms can be used when the dataset contains null or missing values.The sklearn implementations of naive Bayes and k-Nearest Neighbours in Python do not support the presence of the missing values.

Another algorithm that can be used here is Random Forest that works well on non-linear and categorical data. It adapts to the data structure taking into consideration the high variance or the bias, producing better results on large datasets.

5. **Other Imputation Methods:**

Depending on the nature of the data or data type, some other imputation methods may be more appropriate to impute missing values.

For example, for the data variable having longitudinal behavior, it might make sense to use the last valid observation to fill the missing value. This is known as the Last observation carried forward (LOCF) method.

For the time-series dataset variable, it makes sense to use the interpolation of the variable before and after a timestamp for a missing value.

6. **Prediction of missing values:**

In the earlier methods to handle missing values, we do not use the correlation advantage of the variable containing the missing value and other variables. Using the other features which don't have nulls can be used to predict missing values.

The regression or classification model can be used for the prediction of missing values depending on the nature (categorical or continuous) of the feature having missing value.

```
'Age' column contains missing values so for prediction of null
values the splitting of data will be,
y_train: rows from data["Age"] with non null values
y_test: rows from data["Age"] with null values
X_train: Dataset except data["Age"] features with non null
values
X_test: Dataset except data["Age"] features with null values
```

3.2.2.3 Handling of Outliers

A data point that varies greatly from other results is referred to as an outlier. An outlier may also be described as an observation in our data that is incorrect or abnormal as compared to other observations.

Outliers can be caused by measurement uncertainty or due to experimental error. Outliers in data can spoil and deceive the training process of machine learning models, resulting in less accurate models and eventually bad performance.

We can measure the boundary for outliers once we've decided whether outliers are present in the data using the box plot. To measure the boundary for outliers, we can use the two methods below, both based on data distribution:

I) If the Data is Normally Distributed:

We can use the empirical formula of Normal Distribution to determine the boundary for outliers if the data is normally distributed.

Lower Boundary = Mean — 3* (Standard Deviation)

Upper Boundary= Mean + 3 * (Standard Deviation)

II) If the Data is Either Right Skewed or Left Skewed:

We will use the Interquartile Range to measure the limits of Outliers if the data doesn't follow a Normal Distribution or is either right-skewed or left-skewed.

Interquartile Range(IQR) = Q3(75th percentile) -Q1(25th percentile)

The formula for the outlier boundary can be calculated as:

Lower Boundary= First Quartile(Q1/25th percentile) — (1.5 * IQR)

Upper Boundary = Third Quartile(Q3/75th percentile) +(1.5* IQR)

If the outlier's maximum value is extremely high in comparison to the upper boundary, the boundary of outliers (also known as extreme outliers) will be calculated using the formula below:

Lower Boundary= First Quartile(Q1/25th percentile) — (3 * IQR)

Upper Boundary = Third Quartile(Q3/75th percentile) +(3 * IQR)

3.2.2.4 Categorical data and Encoding Techniques

A categorical variable is one that has two or more categories (values). There are two types of categorical variable, **nominal** and **ordinal**. A nominal variable has no intrinsic ordering to its categories. For example, gender is a categorical variable having two categories (male and female) with no intrinsic ordering to the categories. An ordinal variable has a clear ordering.

Many ML algorithms are unable to operate on categorical or label data directly. However, Decision tree can directly learn from such data. Hence, they require all input variables and output variables to be numeric. This means that categorical data must be converted to a numerical form.

Few types of categorical variable encoding are:

1. **One hot encoding:** Encoding each categorical variable with different Boolean variables (also called dummy variables) which take values 0 or 1, indicating if a category is present in an observation.

2. **Integer Encoding / Label Encoding**: Replace the categories by a number from 1 to n (or 0 to n-1, depending the implementation), where n is the number of distinct categories of the variable.

3. **Count or frequency encoding:** Replace the categories by the count of the observations that show that category in the dataset. Similarly, we can replace the category by the frequency -or percentage- of observations in the dataset. That is, if 10 of our 100 observations show the colour blue, we would replace blue by 10 if doing count encoding, or by 0.1 if replacing by the frequency**.**

4. **Ordered Integer Encoding**: Categories are replaced by integer 1 to k, where k is the distinct categories in variable, but this numbering is decided by mean of target of each category. In example, Green has target mean of 0, Red has target mean of 0.5, yellow has target mean of 1. Hence yellow is replaced by 1, Red by 2 and Green by 3.

5. **Encoding using "Weight of Evidence"**: Each category will be replaced by natural log of [p(1)/p(0)], where p(1) is the probability of good target variable and p(0) is the probability of bad target variable of each category in categorical variable. In famous "**titanic**" dataset, one of the categorical variables, "**Cabin**" can encoded as

shown below, given that “**Survived**” as target variable. **p(1)** is the probability of surviving for each category and **p(0)** is the probability of death.

6. **One Hot Encoding: get_dummies** in Pandas library would do the job of encoding as shown below. It would create extra columns for each category using 0 and 1 indicating if the category is present. If category is present it would be indicated by 1 else indicated by 0

7. **Integer Encoding/Label Encoding:** To replace each category in column, we have to create dictionary having key as each category and value as arbitrary number for that category. Then, each category can be mapped to the number defined in dictionary in column.

8. **count/Frequency Encoding**: First step is to create the dictionary with key as category and values as frequency(or count) of that category. Then, replace the categories by counts using dictionary

9. **Ordered Integer Encoding** : First, calculate the target mean of each category (use **groupby()** in Pandas) in column and sort them. Assign numerical value in ascending order to target mean. Lower the target mean, lower the numerical value and vice-versa.

3.2.2.5 Feature Scaling

Feature scaling is the process of normalising the range of features in a dataset. Real-world datasets often contain features that are varying in degrees of magnitude, range and units. Therefore, in order for machine learning models to interpret these features on the same scale, we need to perform feature scaling.

We will be using the SciKit-Learn library to demonstrate various feature scaling techniques.

1) Standard Scaler: In this approach, we bring all the features to a similar scale centring the feature at 0 with a standard deviation of 1. In the case of outliers, this scaler technique will be affected. Hence, it is used when the features are normally distributed.

Standard Scaler = xi–mean(x)/stdev(x)

2) Min-Max Scaler : This estimator scales each feature individually such that it is in the given range, e.g., between zero and one. This technique is mainly used in deep learning and also when the distribution is not Gaussian. This scaler is also sensitive to outliers.

Min-Max Scaler = xi–min(x) / max(x)–min(x)

3) Robust Scaler: This is a very robust technique when we have outliers in our data. This scaler removes the median and scales the data according to the quantile range. If our data contains many outliers, scaling using the mean and standard deviation will not work. Hence, it uses the interquartile range to scale the data.

Robust Scaler = xi–Q2(x)/Q3(x)–Q1(x)

3.2.3 Selection of Dependent and Independent variables

Independent Variables: The variable that are not affected by the other variables are called independent variables.

Example: age of a person, is an independent variable, two person’s born on same date will have same age irrespective of how they lived.

Dependent Variables: The variables which depend on other variables or factors. We expect these variables to change when the independent variables, upon whom they depend, undergo a change. They are the *presumed effect*.

Example: let us say you have a test tomorrow, then, your test score is dependent upon the amount of time you studied, so the test score is a dependent variable, and amount of time independent variable in this case.

The dependent or target variable here is Exited_Target which tells us a particular Customer are likely to leave a service or to cancel a subscription to a service or not. The target variable is selected based on our Customer’s problems and what we are trying to predict. The independent variables are selected after doing exploratory data analysis and we used Boruta to select which variables are most affecting our target variable.

3.2.4 Data Sampling Methods

I⇥ Statistics tkc **sampling method** oí **sampling technique** is tkc píoccss or st"dQi⇥g tkc pop"latio⇥ bQ gatkcíi⇥ g i⇥ roímatio⇥ a⇥ d a⇥ alQsi⇥ g tkat data. It is tkc basis or tkc data wkcíc tkc samplc spacc is c⇥oímo"s.

I’kcíc aíc sc:cíal dirrcíc⇥t sampli⇥g tcck⇥iq"cs a:ailablc, a⇥d tkcQ ca⇥ bc s"bdi:idcd i⇥to two gío"ps. All tkcsc mctkods or sampli⇥g maQ i⇥:ol:c spcciricallQ taígcti⇥g kaíd oí appíoack to ícack gío"ps.

Types of Sampling Methods:

I⇥ Statistics, tkcíc aíc dirrcíc⇥t sampli⇥g tcck⇥iq"cs a:ailablc to gct íclc:a⇥t ícs"lts ríom tkc pop"latio⇥. I’kc two dirrcíc⇥t tQpcs or sampli⇥g mctkods aíc::

- PíobabilitQ Sampli⇥g
- No⇥ -píobabilitQ Sampli⇥ g

Probability Sampling:

The probability sampling method utilizes some form of random selection. In this method, all the eligible individuals have a chance of selecting the sample from the whole sample space. This method is more time consuming and expensive than the non-probability sampling method. The benefit of using probability sampling is that it guarantees the sample that should be the representative of the population.

Non-Probability Sampling:

The non-probability sampling method is a technique in which the researcher selects the sample based on subjective judgment rather than the random selection. In this method, not all the members of the population have a chance to participate in the study.

The data we have is highly unbalanced data so we used some sampling methods which are used to balance the target variable so we our model will be developed with good accuracy and precision. We used three Sampling methods

3.2.4.1 Stratified sampling Method

Stratified sampling Method is one of the Probability sampling method.In a stratified sampling method, the total population is divided into smaller groups to complete the sampling process. The small group is formed based on a few characteristics in the population. After separating the population into a smaller group, the statisticians randomly select the sample.Stratified sampling randomly selects data points from majority class so they will be equal to the data points in the minority class. So, after the sampling both the class will have same no of observations. It can be performed using strata function from the library sampling.

For example, there are three bags (A, B and C), each with different balls. Bag A has 50 balls, bag B has 100 balls, and bag C has 200 balls. We have to choose a sample of balls from each bag proportionally. Suppose 5 balls from bag A, 10 balls from bag B and 20 balls from bag C.

3.2.4.2 Simple random sampling Method

Simple random sampling Method is one of the Probability sampling method.In simple random sampling technique, every item in the population has an equal and likely chance of being selected in the sample. Since the item selection entirely depends on the chance, this method is known as “**Method of chance Selection**”. As the sample size is large, and the item is chosen randomly, it is known as “**Representative Sampling**”. Simple random sampling is a sampling technique where a set percentage of the data is selected randomly. It is generally done to reduce bias in the dataset which can occur if data is selected manually without randomizing the dataset.

We used this method to split the dataset into train dataset which contains 70% of the total data and test dataset with the remaining 30% of the data.

Example:

Suppose we want to select a simple random sample of 200 students from a school. Here, we can assign a number to every student in the school database from 1 to 500 and use a random number generator to select a sample of 200 numbers.

3.2.4.3 Systematic Sampling Method

In the systematic sampling method, the items are selected from the target population by selecting the random selection point and selecting the other methods after a fixed sample interval. It is calculated by dividing the total population size by the desired population size.

Example:

Suppose the names of 300 students of a school are sorted in the reverse alphabetical order. To select a sample in a systematic sampling method, we have to choose some 15 students by randomly selecting a starting number, say 5. From number 5 onwards, will select every 15th person from the sorted list. Finally, we can end up with a sample of some students.

3.2.5 Models Used for Development

We built our predictive models by using the following ten algorithms

3.2.5.1 Model 01

Logistic Regression:

Logistic regression is one of the most popular Machine Learning algorithms, which comes under the Supervised Learning technique. It is used for predicting the categorical dependent variable using a given set of independent variables. Logistic regression predicts the output of a categorical dependent variable. Therefore the outcome must be a categorical or discrete value. It can be either Yes or No, 0 or 1, true or False, etc. but instead of giving the exact value as 0 and 1, it gives the probabilistic values which lie between 0 and 1

Logistic Regression is much similar to the Linear Regression except that how they are used. Linear Regression is used for solving Regression problems, whereas Logistic regression is used for solving the classification problems. On the basis of the categories, Logistic Regression can be classified into three types: Binomial, Multinomial and Ordinal.

Logistic uses logit link function to convert the likelihood values to probabilities so we can get a good estimate on the probability of a particular observation to be positive class or negative class. The also gives us p-value of the variables which tells us about significance of each independent variable.

Logistic Regression Equation:

The Logistic regression equation can be obtained from the Linear Regression equation.

$$\boldsymbol{log\left[\frac{y}{1-y}\right] = b_0 + b_1x_1 + b_2x_2 + b_3x_3 + \cdots + b_nx_n}$$

The above equation is the equation for Logistic Regression.

3.2.5.2 Model 02

Decision Tree:

Decision Tree is a Supervised learning technique that can be used for both classification and Regression problems, but mostly it is preferred for solving Classification problems. It is a tree-structured classifier, where internal nodes represent the features of a dataset, branches represent the decision rules and each leaf node represents the outcome.In a Decision tree, there are two nodes, which are the Decision Node and Leaf Node. Decision nodes are used to make any decision and have multiple branches, whereas Leaf nodes are the output of those decisions and do not contain any further branches.

There are various algorithms in Machine learning, so choosing the best algorithm for the given dataset and problem is the main point to remember while creating a machine learning model. Below are the two reasons for using the Decision tree:

1. Decision Trees usually mimic human thinking ability while making a decision, so it is easy to understand.

2. The logic behind the decision tree can be easily understood because it shows a tree-like structure.

Random Forest:

Random Forest is a popular machine learning algorithm that belongs to the supervised learning technique. It can be used for both Classification and Regression problems in ML. It is based on the concept of ensemble learning, which is a process of combining multiple classifiers to solve a complex problem and to improve the performance of the model. As the name suggests, "Random Forest is a classifier that contains a number of decision trees on various subsets of the given dataset and takes the average to improve the predictive accuracy of that dataset." Instead of relying on one decision tree, the random forest takes the prediction from each tree and based on the majority votes of predictions, and it predicts the final output.

The greater number of trees in the forest leads to higher accuracy and prevents the problem of overfitting. Random forest is an algorithm that consists of many decision trees. It was first developed by Leo Breiman and Adele Cutler. The idea behind it is to build several trees, To have the instance classified by each tree, and to give a "vote" at each class. The model uses a "bagging" approach and the random selection of features to build a collection of decision trees with controlled variance. The instance's class is to the class with the highest number of votes, the class that occurs the most within the leaf in which the instance is placed.

The error of the forest depends on:

- Trees correlation: the higher the correlation, the higher the forest error rate.
- The strength of each tree in the forest. A strong tree is a tree with low error. By using trees that classify the instances with low error the error rate of the forest decreases.

3.2.5.3 Model 03

Artificial Neural Networks:

Artificial Neural Network Tutorial provides basic and advanced concepts of ANNs. Our Artificial Neural Network tutorial is developed for beginners as well as professions. The term "Artificial neural network" refers to a biologically inspired sub-field of artificial intelligence modeled after the brain. An Artificial neural network is usually a computational network based on biological neural networks that construct the structure of the human brain. Similar to a human brain has neurons interconnected to each other, artificial neural networks also have neurons that are linked to each other in various layers of the networks. These neurons are known as nodes.

Artificial neural network tutorial covers all the aspects related to the artificial neural network. In this tutorial, we will discuss ANNs, Adaptive resonance theory, Kohonen self-organizing map, Building blocks, unsupervised learning, Genetic algorithm, etc.

Artificial neural networks can theoretically solve any problem. ANNs can identify hidden patterns between the variables and can find how different combinations of variables can affect the target variable. The error correction is done by gradient descent algorithm which can reduce the error rate as much as possible for the given data.

3.2.5.4 Model 04

Extra Tress Classifier:

Extremely Randomized Trees Classifier(Extra Trees Classifier) is a type of ensemble learning technique which aggregates the results of multiple de-correlated decision trees collected in a “forest” to output it’s classification result. In concept, it is very similar to a Random Forest Classifier and only differs from it in the manner of construction of the decision trees in the forest. Each Decision Tree in the Extra Trees Forest is constructed from the original training sample. Then, at each test node, Each tree is provided with a random sample of k features from the feature-set from which each decision tree must select the best feature to split the data based on some mathematical criteria (typically the Gini Index). This random sample of features leads to the creation of multiple de-correlated decision trees.

It is related to the widely used random forest algorithm. It can often achieve as-good or better performance than the random forest algorithm, although it uses a simpler algorithm to construct the decision trees used as members of the ensemble. It is also easy to use given that it has few key hyperparameters and sensible heuristics for configuring these hyperparameters.

To perform feature selection using the above forest structure, during the construction of the forest, for each feature, the normalized total reduction in the mathematical criteria used in the decision of feature of split (Gini Index if the Gini Index is used in the construction of the forest) is computed. This value is called the Gini Importance of the feature. To perform feature selection, each feature is ordered in descending order according to the Gini Importance of each feature and the user selects the top k features according to his/her choice

3.2.5.5 Model 05

K-Nearest Neighbor(KNN):

K-Nearest Neighbour is one of the simplest Machine Learning algorithms based on Supervised Learning technique. K-NN algorithm assumes the similarity between the new case/data and available cases and put the new case into the category that is most similar to the available categories. K-NN algorithm stores all the available data and classifies a new data point based on the similarity. This means when new data appears then it can be easily classified into a well suite category by using K- NN algorithm.

K-NN algorithm can be used for Regression as well as for Classification but mostly it is used for the Classification problems. K-NN is a non-parametric algorithm, which means it does not make any assumption on underlying data. It is also called a lazy learner algorithm because it does not learn from the training set immediately instead it stores the dataset and at the time of classification, it performs an action on the dataset. KNN algorithm at the training phase just stores the dataset and when it gets new data, then it classifies that data into a category that is much similar to the new data.

Example: Suppose, we have an image of a creature that looks similar to cat and dog, but we want to know either it is a cat or dog. So for this identification, we can use the KNN algorithm, as it works on a similarity measure. Our KNN model will find the similar features of the new data set to the cats and dogs images and based on the most similar features it will put it in either cat or dog category.

The K-NN working can be explained on the basis of the below algorithm:

Step-1: Select the number K of the neighbors

Step-2: Calculate the Euclidean distance of K number of neighbors

Step-3: Take the K nearest neighbors as per the calculated Euclidean distance.

Step-4: Among these k neighbors, count the number of the data points in each category.

Step-5: Assign the new data points to that category for which the number of the neighbor is maximum.

Step-6: Our model is ready.

3.2.5.6 Model 06

Support Vector Machine:

Support Vector Machine or SVM is one of the most popular Supervised Learning algorithms, which is used for Classification as well as Regression problems. However, primarily, it is used for Classification problems in Machine Learning.

The goal of the SVM algorithm is to create the best line or decision boundary that can segregate n-dimensional space into classes so that we can easily put the new data point in the correct category in the future. This best decision boundary is called a hyperplane. SVM chooses the extreme points/vectors that help in creating the hyperplane. These extreme cases are called as support vectors, and hence algorithm is termed as Support Vector Machine.

Example: SVM can be understood with the example that we have used in the KNN classifier. Suppose we see a strange cat that also has some features of dogs, so if we want a model that can accurately identify whether it is a cat or dog, so such a model can be created by using the SVM algorithm. We will first train our model with lots of images of cats and dogs so that it can learn about different features of cats and dogs, and then we test it with this strange creature. So as support vector creates a decision boundary between these two data (cat and dog) and choose extreme cases (support vectors), it will see the extreme case of cat and dog. On the basis of the support vectors, it will classify it as a cat.

There are two types of SVM :

1. Linear SVM

2.Non-Linear SVM

3.2.5.7 Model 07

Bagging Classifier:

A Bagging classifier is an ensemble meta-estimator that fits base classifiers each on random subsets of the original dataset and then aggregate their individual predictions (either by voting or by averaging) to form a final prediction. Such a meta-estimator can typically be used as a way to reduce the variance of a black-box estimator (e.g., a decision tree), by introducing randomization into its construction procedure and then making an ensemble out of it.

Each base classifier is trained in parallel with a training set which is generated by randomly drawing, with replacement, N examples(or data) from the original training dataset – where N is the size of the original training set. Training set for each of the base classifiers is independent of each other. Many of the original data may be repeated in the resulting training set while others may be left out.

Bagging reduces overfitting (variance) by averaging or voting, however, this leads to an increase in bias, which is compensated by the reduction in variance though.

Example:

Bagging works on an imaginary training dataset is shown below. Since Bagging resamples the original training dataset with replacement, some instance(or data) may be present multiple times while others are left out.

Original training dataset: 1, 2, 3, 4, 5, 6, 7, 8, 9, 10

Resampled training set 1: 2, 3, 3, 5, 6, 1, 8, 10, 9, 1

Resampled training set 2: 1, 1, 5, 6, 3, 8, 9, 10, 2, 7

Resampled training set 3: 1, 5, 8, 9, 2, 10, 9, 7, 5, 4

3.2.5.8 Model 08

Gradient Boosting:

Gradient boosting classifiers are a group of machine learning algorithms that combine many weak learning models together to create a strong predictive model. Decision trees are usually used when doing gradient boosting. Gradient boosting classifiers are specific types of algorithms that are used for classification tasks, as the name suggests:

Gradient Boosting is a popular boosting algorithm. In gradient boosting, each predictor corrects its predecessor's error. In contrast to Adaboost, the weights of the training instances are not tweaked, instead, each predictor is trained using the residual errors of predecessor as labels.

There is a technique called the Gradient Boosted Trees whose base learner is CART (Classification and Regression Trees).The ensemble consists of N trees. Tree1 is trained using the feature matrix X and the labels y. The predictions labelled y1(hat) are used to determine the training set residual errors r1. Tree2 is then trained using the feature matrix X and the residual errors r1 of Tree1 as labels. The predicted results r1(hat) are then used to determine the residual r2. The process is repeated until all the N trees forming the ensemble are trained.

There is an important parameter used in this technique known as Shrinkage.

Shrinkage refers to the fact that the prediction of each tree in the ensemble is shrunk after it is multiplied by the learning rate (eta) which ranges between 0 to 1. There is a trade-off between eta and number of estimators, decreasing learning rate needs to be compensated with increasing estimators in order to reach certain model performance. Since all trees are trained now, predictions can be made.

Each tree predicts a label and final prediction is given by the formula,

y(pred) = y1 + (eta * r1) + (eta * r2) +   + (eta * rN)

The class of the gradient boosting regression in scikit-learn is GradientBoostingRegressor. A similar algorithm is used for classification known as GradientBoostingClassifier.

3.2.5.9 Model 09

LightGBM (Light Gradient Boosting Machine):

LightGBM is a gradient boosting classifier in machine learning that uses tree-based learning algorithms. It is designed to be distributed and efficient with faster drive speed and higher efficiency, lower memory usage and better accuracy. In machine learning, the LightGBM classifier is part of the Boosting family, and today it is the most common classification model in the machine learning community. LightGBM is a powerful machine learning model that can be shaped depending on the task you are working on.LightGBM is a gradient boosting framework based on decision trees to increases the efficiency of the model and reduces memory usage.

It uses two novel techniques: Gradient-based One Side Sampling and Exclusive Feature Bundling (EFB) which fulfills the limitations of histogram-based algorithm that is primarily used in all GBDT (Gradient Boosting Decision Tree) frameworks. The two techniques of GOSS and EFB described below form the characteristics of LightGBM Algorithm. They comprise together to make the model work efficiently and provide it a cutting edge over other GBDT frameworks

Gradient-based One Side Sampling Technique for LightGBM:

Different data instances have varied roles in the computation of information gain. The instances with larger gradients(i.e., under-trained instances) will contribute more to the information gain. GOSS keeps those instances with large gradients (e.g., larger than a predefined threshold, or among the top percentiles), and only randomly drop those instances with small gradients to retain the accuracy of information gain estimation. This treatment can lead to a more accurate gain estimation than uniformly random sampling, with the same target sampling rate, especially when the value of information gain has a large range.

3.2.5.10 Model 10

Guassian Naïve Bayes:

A Gaussian Naive Bayes algorithm is a special type of NB algorithm. It's specifically used when the features have continuous values. It's also assumed that all the features are following a gaussian distribution i.e, normal distribution. Guassian Naïve Bayes is a variant of Naïve Bayes that follows Gaussian normal distribution and supports continuous data. Naive Bayes are a group of supervised machine learning classification algorithms based on the Bayes theorem. It is a simple classification technique, but has high functionality. They find use when the dimensionality of the inputs is high. Complex classification problems can also be implemented by using Naive Bayes Classifier.

When working with continuous data, an assumption often taken is that the continuous values associated with each class are distributed according to a normal (or Gaussian) distribution. The likelihood of the features is assumed to be-

$$P(x_i \mid y) = \frac{1}{\sqrt{2\pi\sigma_y^2}} \exp\left(-\frac{(x_i - \mu_y)^2}{2\sigma_y^2}\right)$$

Sometimes assume variance

- is independent of Y (i.e., σi),
- or independent of Xi (i.e., σk)
- or both (i.e., σ)

Gaussian Naive Bayes supports continuous valued features and models each as conforming to a Gaussian (normal) distribution. An approach to create a simple model is to assume that the data is described by a Gaussian distribution with no co-variance (independent dimensions) between dimensions. This model can be fit by simply finding the mean and standard deviation of the points within each label, which is all what is needed to define such a distribution.

The above illustration indicates how a Gaussian Naive Bayes (GNB) classifier works. At every data point, the z-score distance between that point and each class-mean is calculated, namely the distance from the class mean divided by the standard deviation of that class.

3.3 AI / ML Models Analysis and Final Results

We used our train dataset to build the above models and used our test data to check the accuracy and performance of our models.

We used confusion matrix to check accuracy, Precision, Recall and F1 score of our models and compare and select the best model for given bank customer's churn data set of size ~ 10000 rows and 14 columns.

3.3.1 Different Model codes

```
# Build the Calssification models and compare the results
from sklearn.linear_model import LogisticRegression
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.ensemble import ExtraTreesClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.naive_bayes import GaussianNB
from sklearn.svm import SVC
from sklearn.ensemble import BaggingClassifier
from sklearn.ensemble import GradientBoostingClassifier
import lightgbm as lgb
# Create objects of classification algorithm with default hyper-parameters
ModelLR = LogisticRegression()
ModelDC = DecisionTreeClassifier()
ModelRF = RandomForestClassifier()
ModelET = ExtraTreesClassifier()
ModelKNN = KNeighborsClassifier(n_neighbors=5)
ModelSVM = SVC(probability=True)
modelBAG=BaggingClassifier(base_estimator=None, n_estimators=100,
max_samples=1.0, max_features=1.0, bootstrap=True, bootstrap_features=False,
oob_score=False, warm_start=False,n_jobs=None,random_state=None,verbose=0)
ModelGB=GradientBoostingClassifier(loss='deviance,learning_rate=0.1
,n_estimators=100,subsample=1.0,criterion='friedman_mse',min_samples_split=2,
min_samples_leaf=1,min_weight_fraction_leaf=0.0,max_depth=3,min_impurity_dec
rease=0.0,init=None,random_state=None,max_features=None,verbose=0,max_leaf
_nodes=None,warm_start=False,validation_fraction=0.1,n_iter_no_change=None,tol=0.001, ccp_alpha=0.0)
ModelLGB = lgb.LGBMClassifier()
ModelGNB = GaussianNB()
# Evalution matrix for all the algorithms
MM = [ModelLR, ModelDC, ModelRF, ModelET, ModelKNN, ModelSVM, modelBAG, ModelGB, ModelLGB, ModelGNB]
for models in MM:
    # Fit the model
    models.fit(x_train, y_train)
    # Prediction
    y_pred = models.predict(x_test)
    y_pred_prob = models.predict_proba(x_test)
    # Print the model name
    print('Model Name: ', models)
    # confusion matrix in sklearn
    from sklearn.metrics import confusion_matrix
    from sklearn.metrics import classification_report
    # actual values
    actual = y_test
    # predicted values
    predicted = y_pred
    # confusion matrix
    matrix = confusion_matrix(actual,predicted,labels=[1,0]).reshape(-1)
    print('Confusion matrix : \n', matrix)
    # outcome values order in sklearn
    tp, fn, fp, tn = confusion_matrix(actual,predicted,labels=[1,0]).reshape(-1)
    print('Outcome values : \n', tp, fn, fp, tn)
    # classification report for precision, recall f1-score and accuracy
    C_Report = classification_report(actual,predicted,labels=[1,0])
    print('Classification report : \n', C_Report)
    # calculating the metrics
    sensitivity = round(tp/(tp+fn), 3);
    specificity = round(tn/(tn+fp), 3);
    accuracy = round((tp+tn)/(tp+fp+tn+fn), 3);
    balanced_accuracy = round((sensitivity+specificity)/2, 3);
    precision = round(tp/(tp+fp), 3);
    f1Score = round((2*tp/(2*tp + fp + fn)), 3);
# Matthews Correlation Coefficient (MCC). Range of values of MCC lie between -1
to +1.
    # A model with a score of +1 is a perfect model and -1 is a poor model
    from math import sqrt
    mx = (tp+fp) * (tp+fn) * (tn+fp) * (tn+fn)
    MCC = round(((tp * tn) - (fp * fn)) / sqrt(mx), 3)
    print('Accuracy :', round(accuracy*100, 2),'%')
    print('Precision :', round(precision*100, 2),'%')
    print('Recall :', round(sensitivity*100,2), '%')
    print('F1 Score :', f1Score)
    print('Specificity or True Negative Rate :', round(specificity*100,2), '%'  )
    print('Balanced Accuracy :', round(balanced_accuracy*100, 2),'%')
    print('MCC :', MCC)
    # Area under ROC curve
    from sklearn.metrics import roc_curve, roc_auc_score
    print('roc_auc_score:', round(roc_auc_score(actual, predicted), 3))
    # ROC Curve
    from sklearn.metrics import roc_auc_score
    from sklearn.metrics import roc_curve
    logit_roc_auc = roc_auc_score(actual, predicted)
    fpr, tpr, thresholds = roc_curve(actual, models.predict_proba(x_test)[:,1])
    plt.figure()
    # plt.plot(fpr, tpr, label='Logistic Regression (area = %0.2f)' % logit_roc_auc)
    plt.plot(fpr, tpr, label= 'Classification Model' % logit_roc_auc)
    plt.plot([0, 1], [0, 1],'r--')
    plt.xlim([0.0, 1.0])
    plt.ylim([0.0, 1.05])
    plt.xlabel('False Positive Rate')
    plt.ylabel('True Positive Rate')
    plt.title('Receiver operating characteristic')
    plt.legend(loc="lower right")
    plt.savefig('Log_ROC')
    plt.show()
    print('----------------------------------------------------')
```

```
#-----------------------------------------------
new_row = {'Model Name' : models,
                  ‘True_Positive' : tp,
                  'False_Negative' : fn,
                  'False_Positive' : fp,
                  'True_Negative' : tn,
                  'Accuracy' : accuracy,
                  'Precision' : precision,
                  'Recall' : sensitivity,
                  'F1 Score' : f1Score,
                  'Specificity' : specificity,
                  'MCC':MCC,
                  'ROC_AUC_Score':roc_auc_score(actual, predicted),
                  'Balanced Accuracy':balanced_accuracy}
EMResults =EMResults.append(new_row, ignore_index=True)
```

3.3.2 Random Forest Python Code

```
#To build the 'ExtraTreesClassifier' model with random sampling
from sklearn.ensemble import ExtraTreesClassifier
ModelET=ExtraTreesClassifier(n_estimators=100,criterion='gini', max_depth=None,
min_samples_split=2,min_samples_leaf=1,min_weight_fraction_leaf=0.0,
max_features='sqrt',max_leaf_nodes=None,min_impurity_decrease=0.0,
bootstrap=False, oob_score=False,n_jobs=None,random_state=None, verbose=0,
warm_start=False, class_weight=None,ccp_alpha=0.0, max_samples=None)
# Train the model with train data
ModelET.fit(x_train,y_train)
# Predict the model with test data set
y_pred = ModelET.predict(x_test)
y_pred_prob = ModelET.predict_proba(x_test)
```

3.3.3 Extra Trees Python code

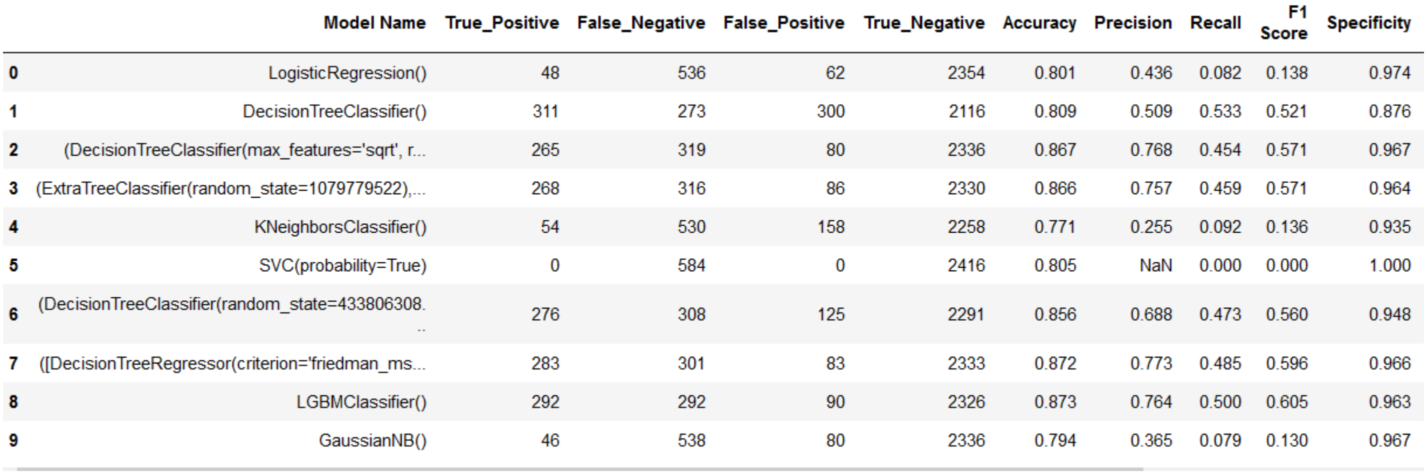
```
# To build the 'Multinominal Decision Tree' model with random sampling
from sklearn.ensemble import RandomForestClassifier
ModelRF=RandomForestClassifier(n_estimators=100,criterion='gini',
max_depth=None,min_samples_split=2,min_samples_leaf=1,
min_weight_fraction_leaf=0.0,max_features='sqrt',min_leaf_nodes=None,min_impu
rity_decrease=0.0,bootstrap=True,oob_score=False,n_jobs=None,random_state=N
one,verbose=0,warm_start=False,class_weight=None,ccp_alpha=0.0,max_samples
=None)

# Train the model with train data

ModelRF.fit(x_train,y_train)

# Predict the model with test data set

y_pred = ModelRF.predict(x_test)

y_pred_prob = ModelRF.predict_proba(x_test)
```

4.0 conclusions and Future work

The model results in the following order by considering the model accuracy, F1 score and RoC AUC score.

1) **Gradient Boosting:** with Simple Random Sampling
2) **LightGBM (Light Gradient Boosting Machine):** with Simple Random Sampling
3) **Random Forest:** with Simple Random Sampling

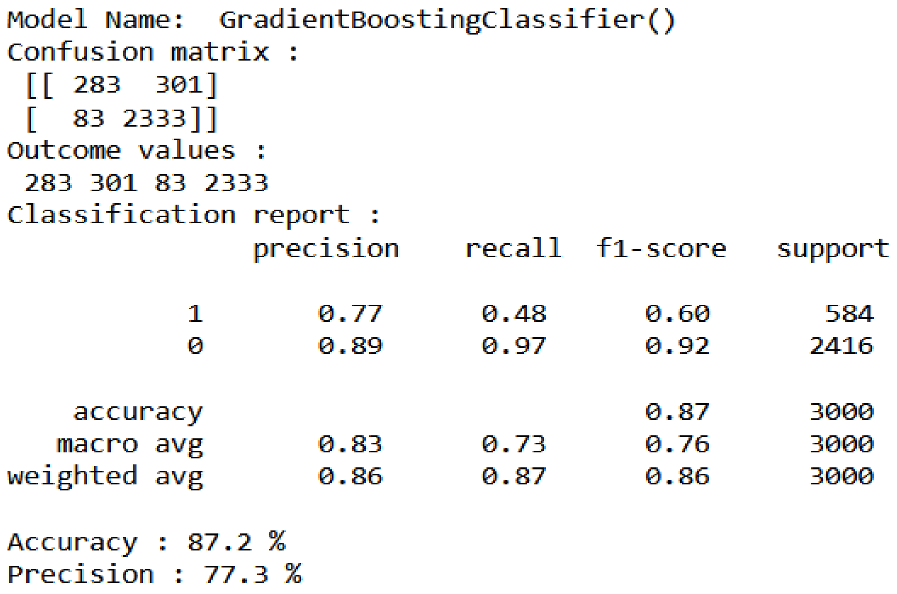
We recommend model – **Gradient Boosting Classifier** with Simple Random Sampling technique as a best fit for the given Bank Customer’s dataset. We considered Gradient Boosting Classifier because it uses In gradient boosting, each predictor corrects its predecessor’s error. In contrast to Ada boost, the weights of the training instances are not tweaked, instead, each predictor is trained using the residual errors of predecessor as labels for Bank Customer’s Churn dataset.



	Model Name	True_Positive	False_Negative	False_Positive	True_Negative	Accuracy	Precision	Recall	F1 Score	Specificity
0	LogisticRegression()	48	536	62	2354	0.801	0.436	0.082	0.138	0.974
1	DecisionTreeClassifier()	311	273	300	2116	0.809	0.509	0.533	0.521	0.876
2	(DecisionTreeClassifier(max_features='sqrt', r...	265	319	80	2336	0.867	0.768	0.454	0.571	0.967
3	(ExtraTreeClassifier(random_state=1079779522),...	268	316	86	2330	0.866	0.757	0.459	0.571	0.964
4	KNeighborsClassifier()	54	530	158	2258	0.771	0.255	0.092	0.136	0.935
5	SVC(probability=True)	0	584	0	2416	0.805	NaN	0.000	0.000	1.000
6	(DecisionTreeClassifier(random_state=433806308...	276	308	125	2291	0.856	0.688	0.473	0.560	0.948
7	([DecisionTreeRegressor(criterion='friedman_ms...	283	301	83	2333	0.872	0.773	0.485	0.596	0.966
8	LGBMClassifier()	292	292	90	2326	0.873	0.764	0.500	0.605	0.963
9	GaussianNB()	46	538	80	2336	0.794	0.365	0.079	0.130	0.967

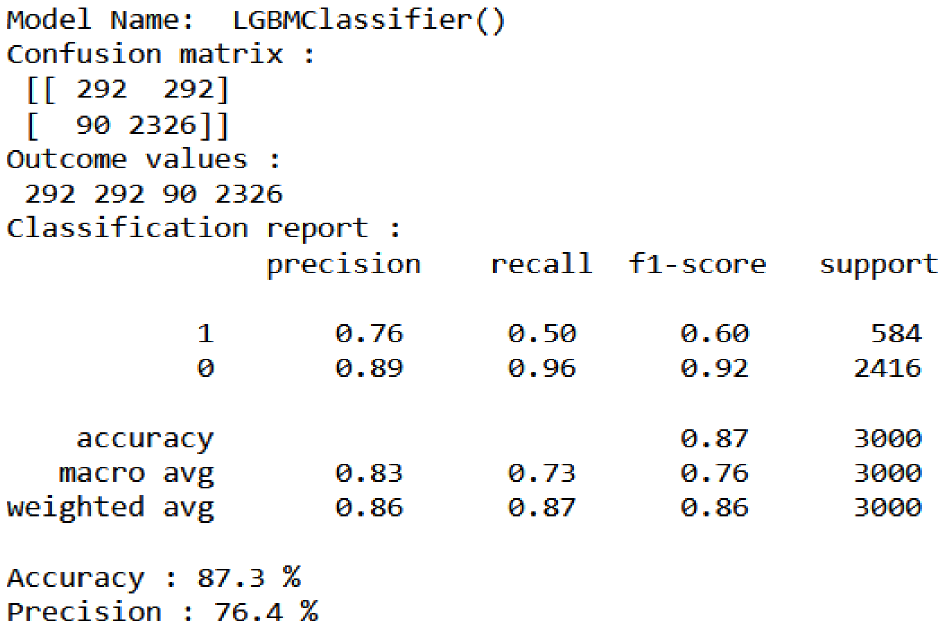
The future work to evaluate the “Other Customer’s Churn factors” in Bank by using classification methods.

1. Gradient Boosting Classifier



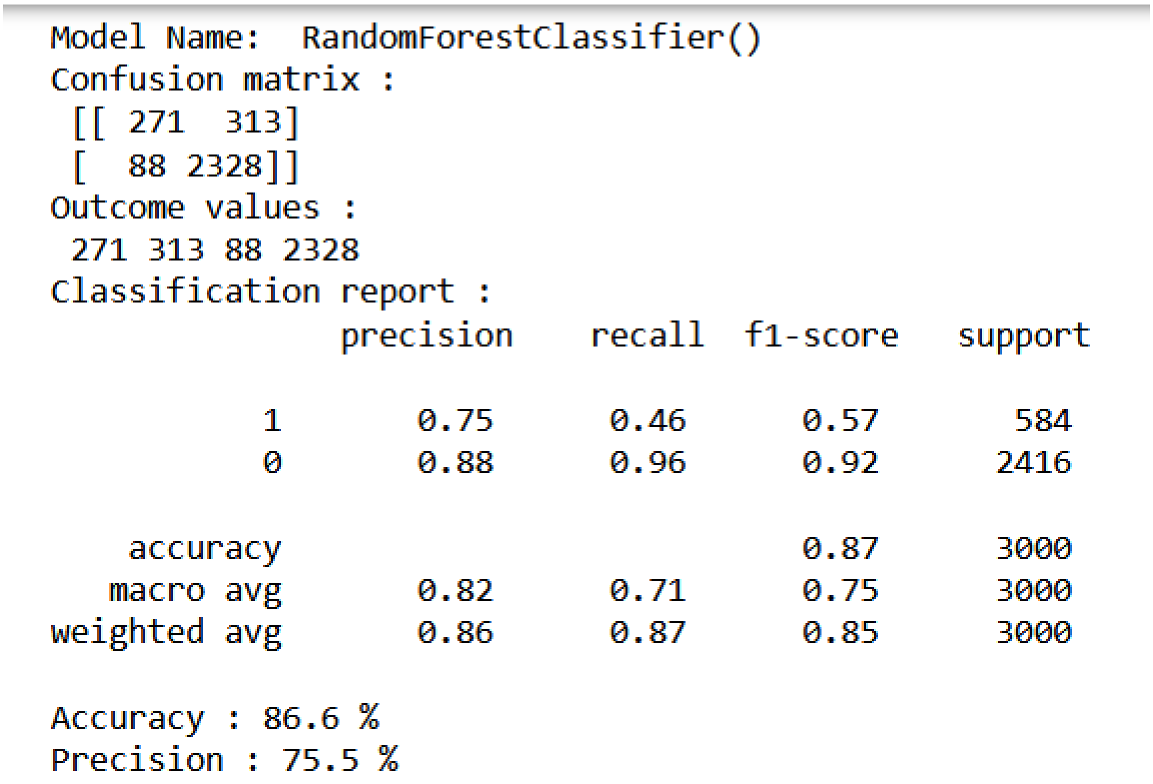
```
Model Name:  GradientBoostingClassifier()
Confusion matrix :
 [[ 283  301]
 [  83 2333]]
Outcome values :
 283 301 83 2333
Classification report :
               precision    recall  f1-score   support

           1       0.77      0.48      0.60       584
           0       0.89      0.97      0.92      2416

    accuracy                           0.87      3000
   macro avg       0.83      0.73      0.76      3000
weighted avg       0.86      0.87      0.86      3000

Accuracy : 87.2 %
Precision : 77.3 %
```

2. LightGBM classifier



```
Model Name:  LGBMClassifier()
Confusion matrix :
 [[ 292  292]
 [  90 2326]]
Outcome values :
 292 292 90 2326
Classification report :
               precision    recall  f1-score   support

           1       0.76      0.50      0.60       584
           0       0.89      0.96      0.92      2416

    accuracy                           0.87      3000
   macro avg       0.83      0.73      0.76      3000
weighted avg       0.86      0.87      0.86      3000

Accuracy : 87.3 %
Precision : 76.4 %
```

3. Random Forest Classifer



```
Model Name:  RandomForestClassifier()
Confusion matrix :
 [[ 271  313]
 [  88 2328]]
Outcome values :
 271 313 88 2328
Classification report :
               precision    recall  f1-score   support

           1       0.75      0.46      0.57       584
           0       0.88      0.96      0.92      2416

    accuracy                           0.87      3000
   macro avg       0.82      0.71      0.75      3000
weighted avg       0.86      0.87      0.85      3000

Accuracy : 86.6 %
Precision : 75.5 %
```

5.0 References

1. AMIN, Adnan, et al. Customer churn prediction in telecommunication industry using data certainty. Journal of Business Research, 2019, 94:290-301.
2. Qureshii SA, Rehman AS, Qamar AM, Kamal A, Rehman A. Telecommunication subscribers churn prediction model using machine learning. In: Eighth international conference on digital information management. 2013. P.131—6.
3. ULLAH, Irfan, et al. A churn prediction model using random forest: analysis of machine learning techniques for churn prediction and factor identification in telecom sector. IEEE Access, 2019, 7: 60134-60149.
4. Umayaparvathi V, Iyakutti K. A survey on customer churn prediction in telecom industry: datasets, methods and metric. Int Res J Eng Technol. 2016; 3(4):1065—70.
5. Abbasimehr H, Setak M, Tarokh M (2011) A neuro-fuzzy classifier for customer churn prediction. International Journal of Computer Applications 19(8):35–41.
6. Asthana P (2018) A comparison of machine learning techniques for customer churn prediction. International Journal of Pure and Applied Mathematics 119(10):1149–1169.
7. Burez J, Van den Poel D (2009) Handling class imbalance in customer churn prediction. Expert Systems with Applications 36(3):4626–4636.

8. Coussement K, De Bock KW (2013) Customer churn prediction in the online gambling industry: The beneficial effect of ensemble learning. Journal of Business Research 66(9):1629–1636

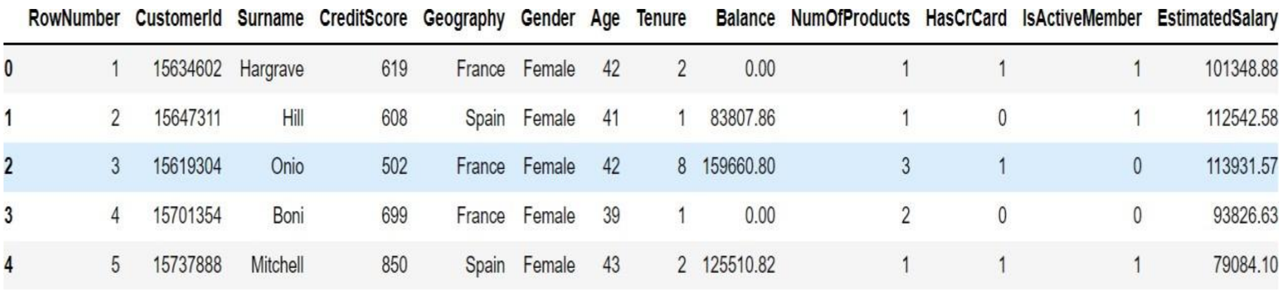
9. Hadden J, Tiwari A, Roy R, Ruta D (2006) Churn prediction: Does technology matter. International Journal of Intelligent Technology 1(2):104–110

10. Hadden J, Tiwari A, Roy R, Ruta D (2007) Computer assisted customer churn management: State-of-the-art and future trends. Computers & Operations Research 34(10):2902– 2917.

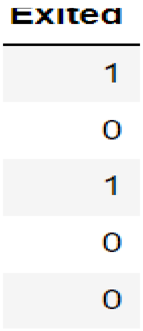
6.0 Appendices

6.1 Python code Results

1. Load the data into data frame



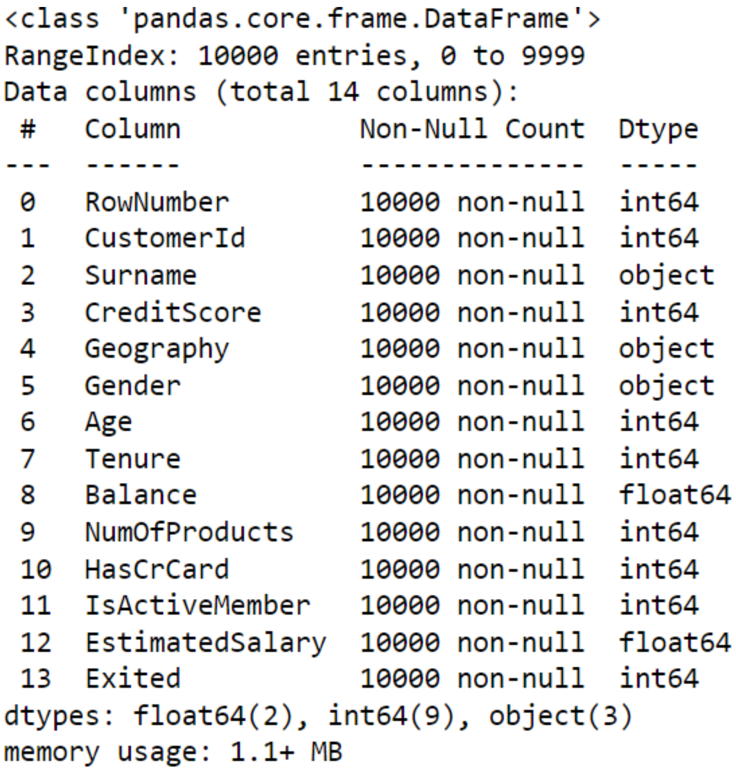
	RowNumber	CustomerId	Surname	CreditScore	Geography	Gender	Age	Tenure	Balance	NumOfProducts	HasCrCard	IsActiveMember	EstimatedSalary
0	1	15634602	Hargrave	619	France	Female	42	2	0.00	1	1	1	101348.88
1	2	15647311	Hill	608	Spain	Female	41	1	83807.86	1	0	1	112542.58
2	3	15619304	Onio	502	France	Female	42	8	159660.80	3	1	0	113931.57
3	4	15701354	Boni	699	France	Female	39	1	0.00	2	0	0	93826.63
4	5	15737888	Mitchell	850	Spain	Female	43	2	125510.82	1	1	1	79084.10



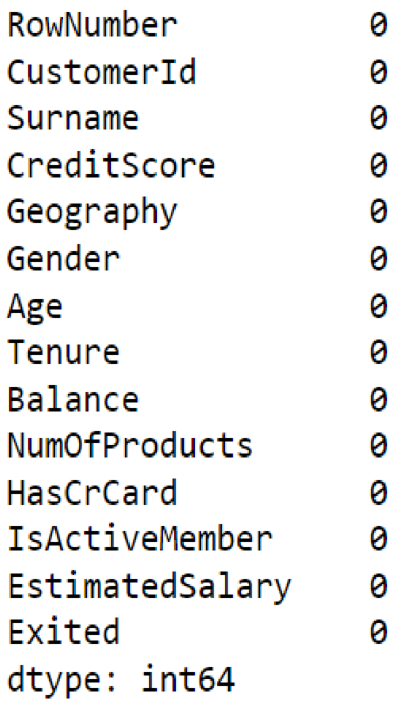
Exited
1
0
1
0
0

2. Displaying the data set Information

3. checking the null values variable

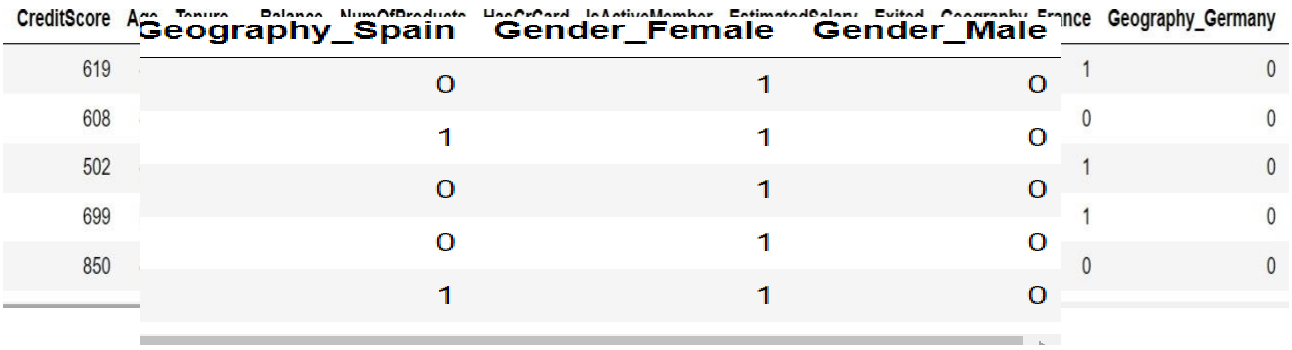


```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10000 entries, 0 to 9999
Data columns (total 14 columns):
 #   Column           Non-Null Count  Dtype  
---  ------           --------------  -----  
 0   RowNumber        10000 non-null  int64  
 1   CustomerId       10000 non-null  int64  
 2   Surname          10000 non-null  object 
 3   CreditScore      10000 non-null  int64  
 4   Geography        10000 non-null  object 
 5   Gender           10000 non-null  object 
 6   Age              10000 non-null  int64  
 7   Tenure           10000 non-null  int64  
 8   Balance          10000 non-null  float64
 9   NumOfProducts    10000 non-null  int64  
 10  HasCrCard        10000 non-null  int64  
 11  IsActiveMember   10000 non-null  int64  
 12  EstimatedSalary  10000 non-null  float64
 13  Exited           10000 non-null  int64  
dtypes: float64(2), int64(9), object(3)
memory usage: 1.1+ MB
```



```
RowNumber          0
CustomerId         0
Surname            0
CreditScore        0
Geography          0
Gender             0
Age                0
Tenure             0
Balance            0
NumOfProducts      0
HasCrCard          0
IsActiveMember     0
EstimatedSalary    0
Exited             0
dtype: int64
```

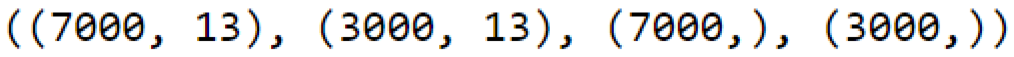
4. Convert 'Geography' and 'Gender' to numerical format using one hot encoding



CreditScore	Geography_France	Geography_Germany
619	1	0
608	0	0
502	1	0
699	1	0
850	0	0

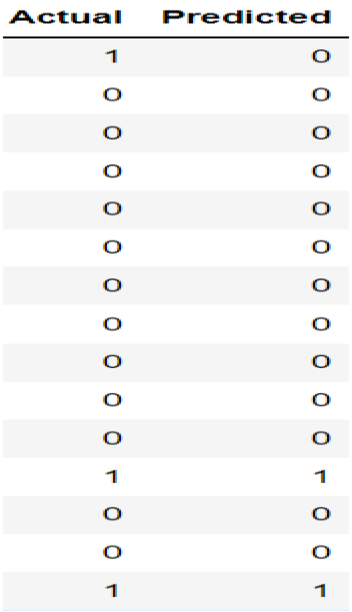
Geography_Spain	Gender_Female	Gender_Male
0	1	0
1	1	0
0	1	0
0	1	0
1	1	0

5. Splitting the dataset into train and test



```
((7000, 13), (3000, 13), (7000,), (3000,))
```

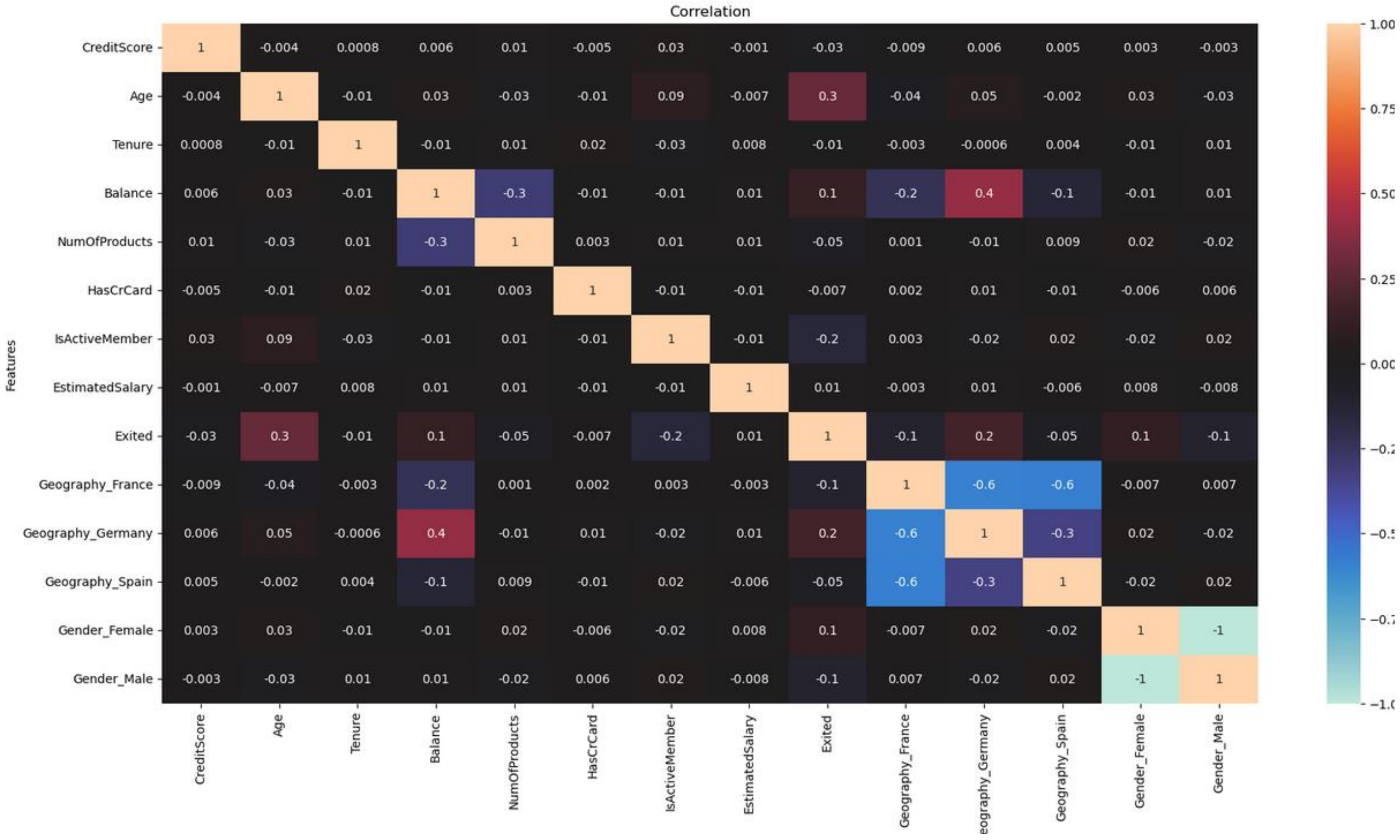
6. Comparing Actual and Predicted values



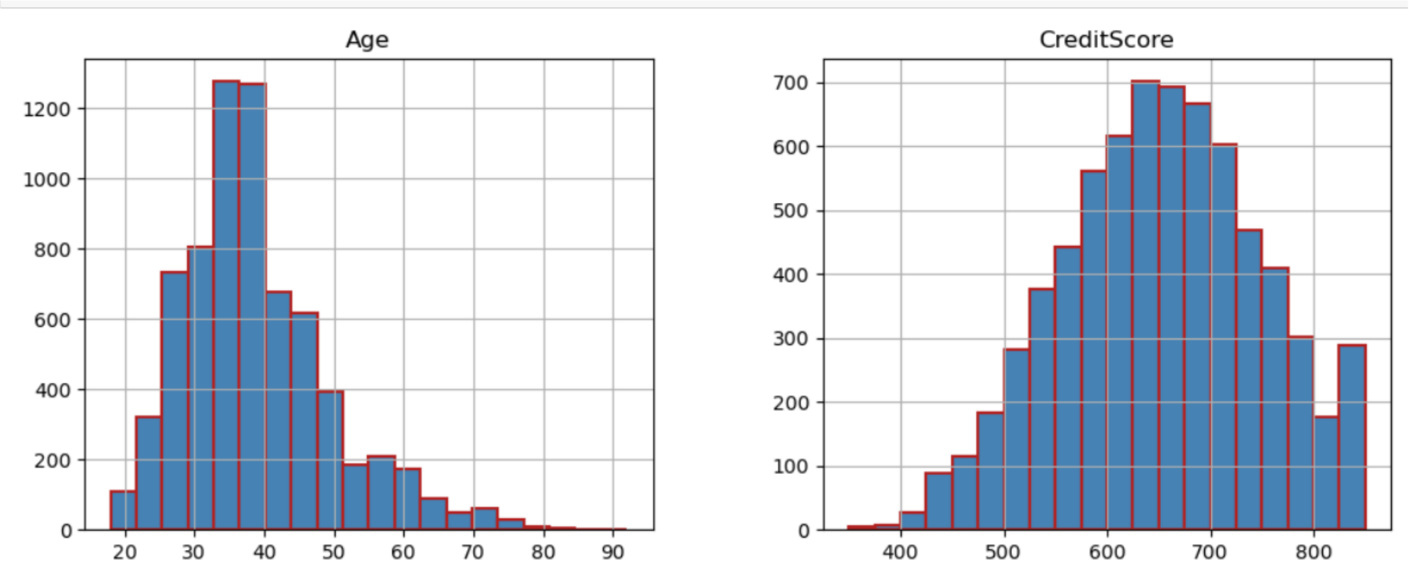
Actual	Predicted
1	0
0	0
0	0
0	0
0	0
0	0
0	0
0	0
0	0
0	0
0	0
1	1
0	0
0	0
1	1

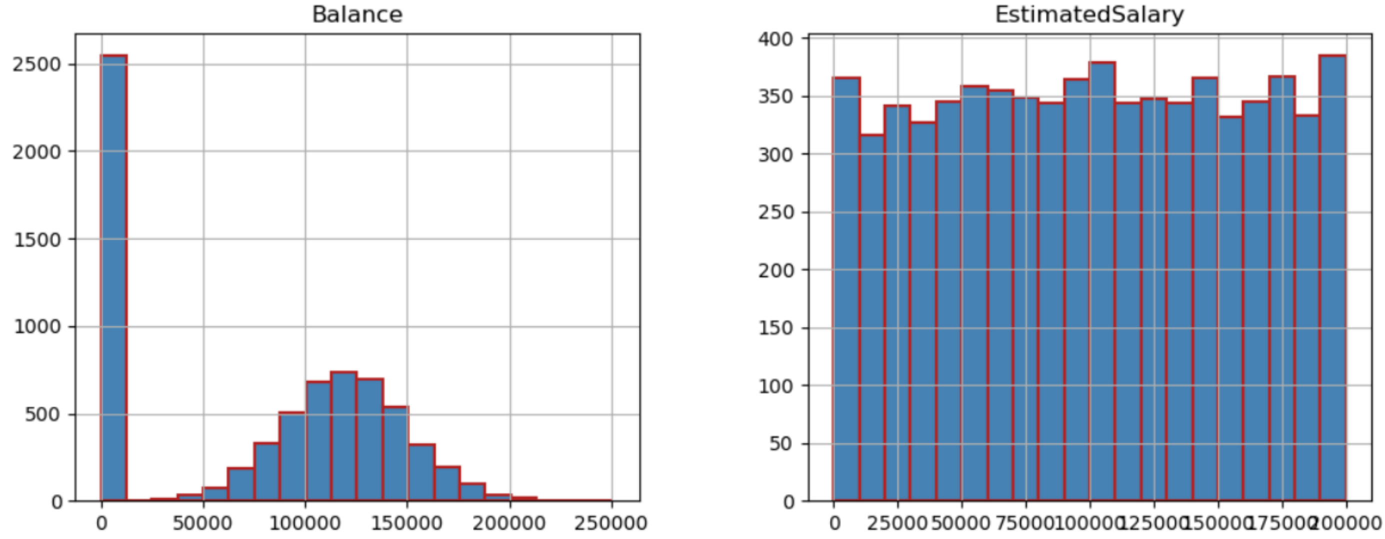
6.2 List of Charts

HEATMAP:



HISTOGRAM




Balance
2500
2000
1500
1000
500
0
0
50000
100000
150000
200000
250000
EstimatedSalary
400
350
300
250
200
150
100
50
0
0
25000
50000
75000
100000
125000
150000
175000
200000

THE END