CUSTOMER CHURN PREDICTION

-:: GROUP MEMBERS ::-

Subhasish Mukherjee, Techno Main Saltlake, (20309100120084)

Semanto Ghosh, Institute of Management Studies, (201941858110007)

Priyankar Biswas, Institute of Management Studies, (201941858110005)

Sankalpa Das, Institute of Management Studies, (201941858110010)

Sumaiya Shakil, Institute of Management Studies,
(201941858110011)

CONTENT::

1. ACKNOWLEDGEMENT
2. OBJECTIVE
3. DATA DESCRIPTION
    a. SOURCE
    b. DATA SET
    c. COLUMN DESCRIPTION
4. MODEL BUILDING
    a. MACHINE LEARNING (ML)
    b. ML APPLICATION IN REAL LIFE
    c. PROS AND CONS
    d. WHY IS MACHINE LEARNING IMPORTANT?
    e. TRADITIONAL PROGRAMMING VS MACHINE LEARNING
    f. ML PROCESS
    g. TYPES OF MACHINE LEARNING
    h. ML ALGORITHMS
    i. FEATURE ENGINEERING
    j. DATA PREPROCESSING
    k. IMPLEMENTING DIFFERENT MODELS TO SELECT THE BEST MODEL
5. FINAL MODEL
6. FUTURE SCOPE

ACKNOWLEDGEMENT

I take this opportunity to express my profound gratitude and deep regards to my faculty Prof. Mr. Titas Roy Chowdhury for his exemplary guidance, monitoring and constant encouragement throughout the course of this project. The blessing, help and guidance given by him/her time to time shall carry me a long way in the journey of life on which I am about to embark. I am obliged to my project team members for the valuable information provided by them in their respective fields. I am grateful for their cooperation during the period of my assignment.

OBJECTIVE

Customer churn is defined as when customers or subscribers discontinue doing business with a firm or service.

Customer Churn is one of the most important and challenging problems for businesses such as Credit Card companies, cable service providers, SASS and telecommunication companies worldwide. Even though it is not the most fun to look at, customer churn metrics can help businesses improve customer retention.

The telecom industries use advanced analytics to understand consumer behavior and

in-turn predict the association of the customers as whether or not they will leave the company(churn).

You can analyze all relevant customer data and develop focused customer retention programs.

It is stated that the cost of acquiring a new customer is far more than that for retaining the existing one.

The reasoning of customer churn can vary and would require domain knowledge in order to define properly, however some common ones are; lack of usage of the product, poor service and better price somewhere else. Regardless of the reasoning that can be specific for different industries, one thing applies for every domain is it costs more to acquire new customers than it

does to retain existing ones. This has a direct impact on operating costs and marketing budgets within the company.

If a corporation could forecast which customers are likely to leave ahead of time, it could focus customer retention efforts only on these "high risk" clients. The ultimate goal is to expand its coverage area and retrieve more customer’s loyalty.

Customer churn is a critical metric because it is much less expensive to retain existing customers than it is to acquire new

customers. To reduce customer churn, telecom companies need to predict which customers are at high risk of churn.

We need to reduce wrong predictions cases i.e.,

churned(1) --> not churned(0), which is False Negetive case

not churned(0) --> churned(1), which is False Positive case

The appropiate metric would be recall as we will consider False Negetive case more.

DATA DESCRIPTION

• Source-

Data is provided by Globsyn Finishing School.

The train data file is in csv format . There are 3333 rows and 21 columns.

• Data Set-

This data set consists of customers details in different columns like-

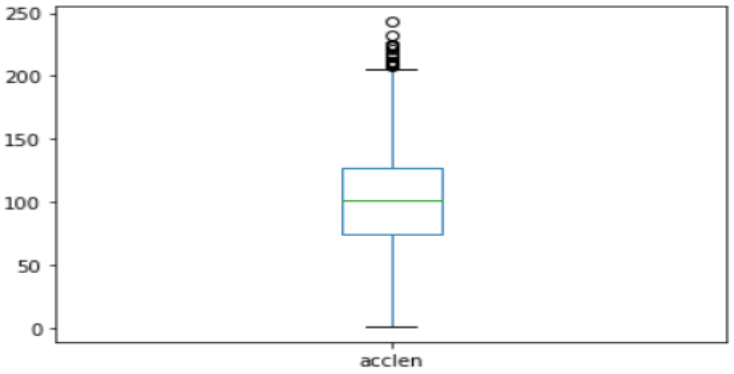
- st – state - 51 Unique States name
- acclen - account length- Length of The Account
- arcode - area code- Code Number of Area having some States
- phnum - phone number- Phone number of individual customers (unique value)
- intplan - internet plan (yes/no)- Yes Indicate Internet Plan is Present and No Indicates no subscription for Internet Plan
- voice – voice plan (yes/no)- Yes Indicate voice plan is Present and No Indicates no subscription for voice Plan
- nummailmes - no of email messages
- tdmin - total day minutes
- tdcal - total day time calls
- tdchar - total day time charges
- temin - total evening time minutes
- tecal - total evening time calls
- tecahr - total evening time charges
- tnmin - total night time minutes
- tncal - total night time calls
- tnchar - total night time charges
- timin - total international minutes
- tical - total international calls
- tichar - total international charges
- ncsc - no. of customer services calls made by customer
- label - Churned? (True/False)- True means customer churned, False means customer retained.

• Column Description-

1. Variable name- st (state)

Type- categorical
Data Type- object
Null values- 0 null values

2. Variable name- acclen (account length)
   Type- discrete
   Data Type- integer
   Null values-0 null values
   Outliers-



   Variable name- arcode
   Type- nominal (cannot be ordered in a meaningful way)
   Data Type- integer
   Null values- 0

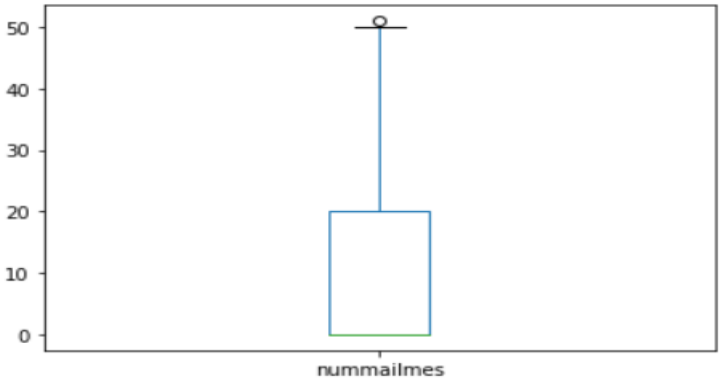
3. Variable name- phnum (phone number)
   Type-categorical
   Data Type- object
   Null values- 0

4. Variable name- intplan (internet plan)
   Type- categorical
   Data Type- object
   Null values- 0

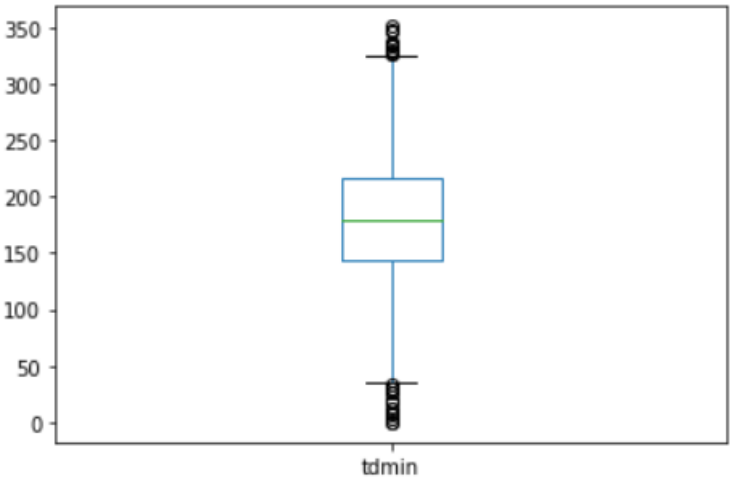
5. Variable name- voice
   Type- categorical

Data Type- object
Null values- 0

6. Variable name- nummailmes (no of emails)
   Type- discrete
   Data Type-integer
   Null values- 0
   Outliers-

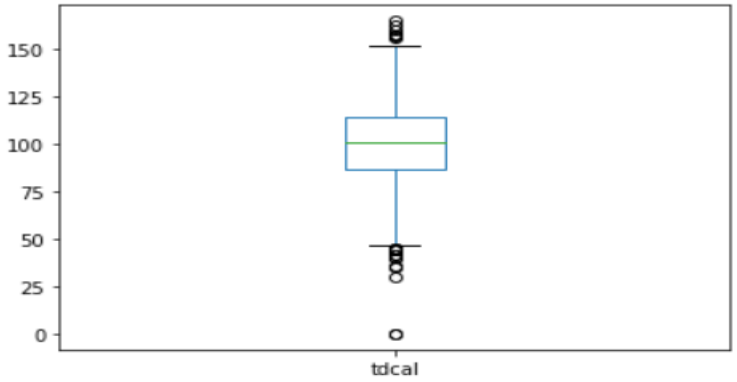


7. Variable name- tdmin(total day minutes)
   Type-continuous
   Data Type-float
   Null values- 0
   Outlier-

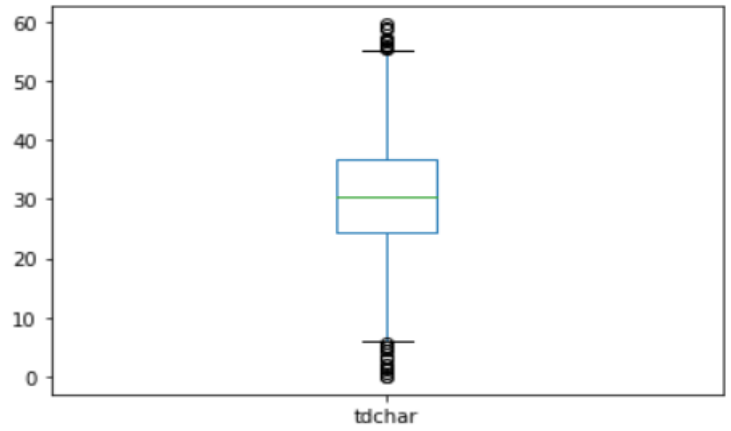


8. Variable name- tdcal(total day call)
   Type-discrete

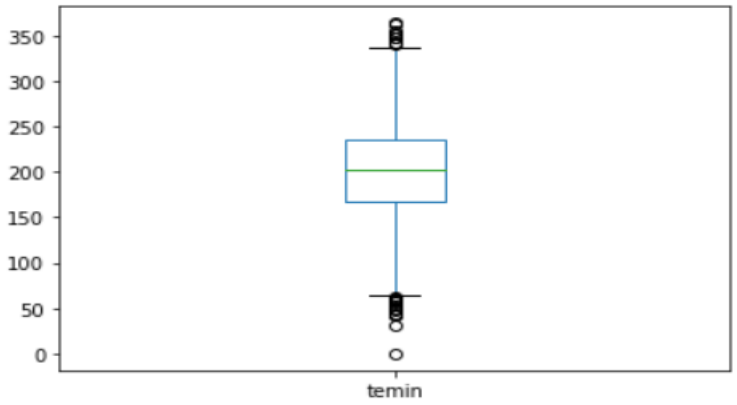
Data Type-integer
Null values- 0
Outlier-



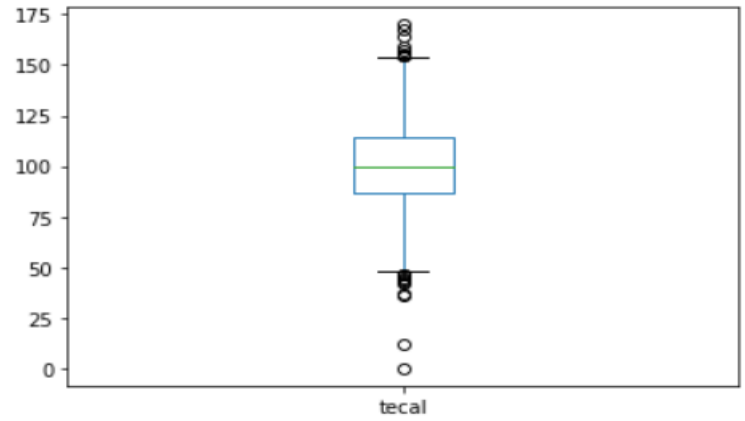
9. Variable name- tdchar
   Type-continuous
   Data Type- float
   Null values- 0
   Outlier-



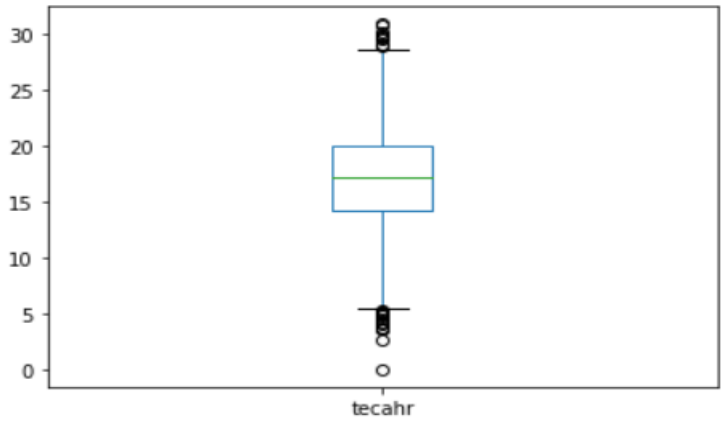
10. Variable name- temin(total evening minutes)
    Type- Continuous
    Data Type- float
    Null values-0
    Outlier-



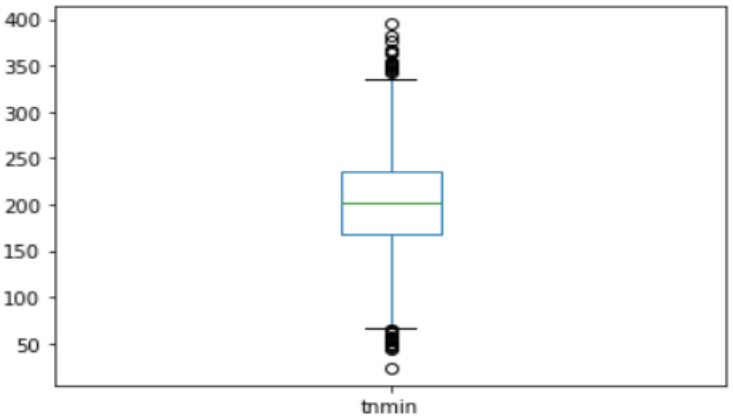
11. Variable name- tecal(total evening calls)
    Type- discrete
    Data Type- integer
    Null values- 0
    Outlier-



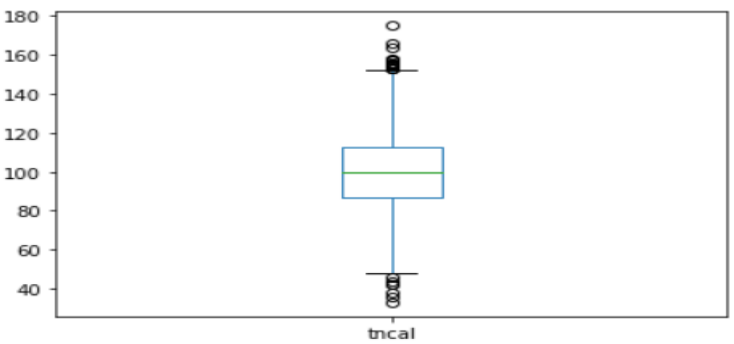
12. Variable name- tecahr(total evening charges)
    Type- continuous
    Data Type- float
    Null values-0
    Outlier-



13. Variable name- tnmin
    Type-continuous
    Data Type-float
    Null values-0
    Outlier-

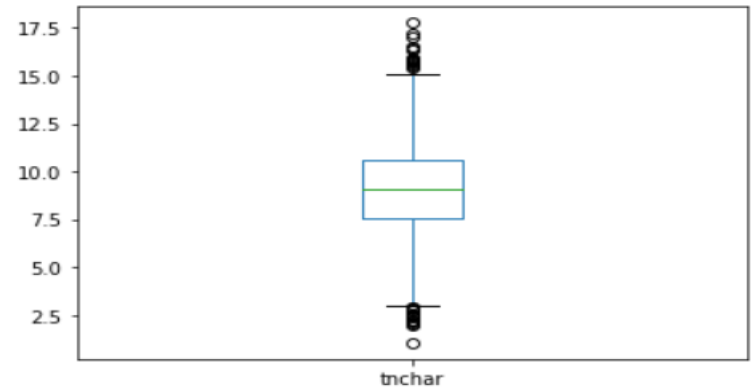


14. Variable name- tncal
    Type- discrete
    Data Type- integer
    Null values- 0
    Outlier-

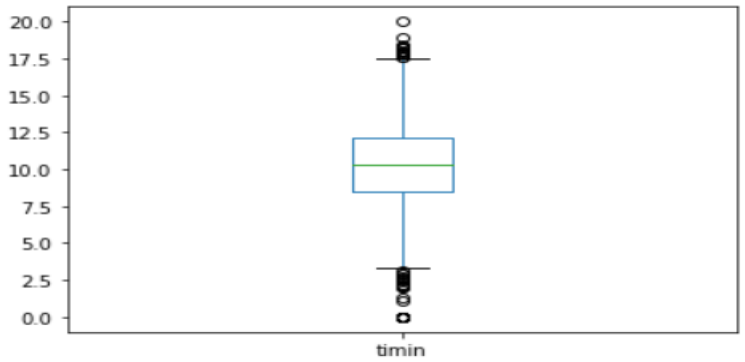


15. Variable name- tnchar

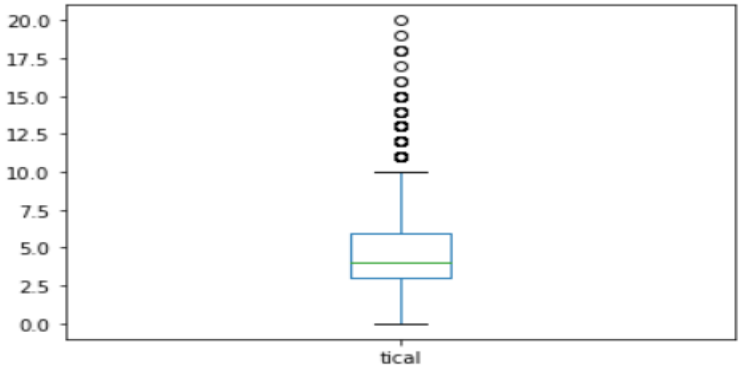
Type- continuous
Data Type- float
Null values- 0
Outlier-



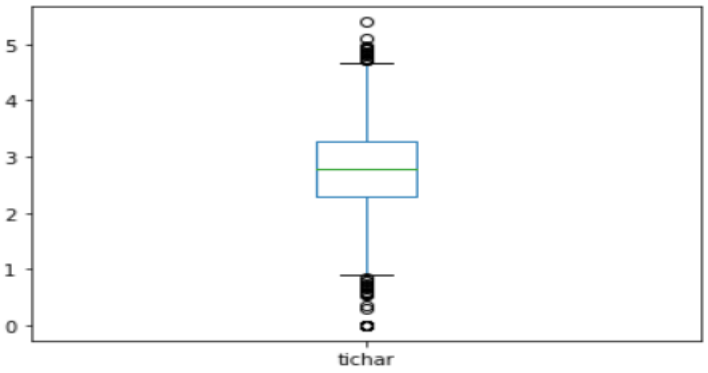
16. Variable name- timin
Type-- continuous
Data Type- float
Null values- 0
Outlier-



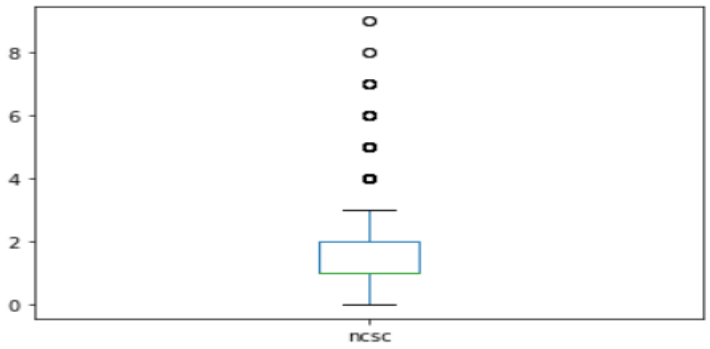
17. Variable name- tical
Type- discrete
Data Type- integer
Null values- 0
Outlier-



18. Variable name- tichar
    Type- continuous
    Data Type- float
    Null values- 0
    Outlier-



19. Variable name- ncsc
    Type- discrete
    Data Type- integer
    Null values- 0
    Outlier-



20. Variable name- label
    Type-categorical
    Data Type- object
    Null values- 0

GLIMPSE OF RAW DATA-

Code:

```
data=pd.read_csv('churn_train.csv')
data.head()
```

Output:

Out[3]:

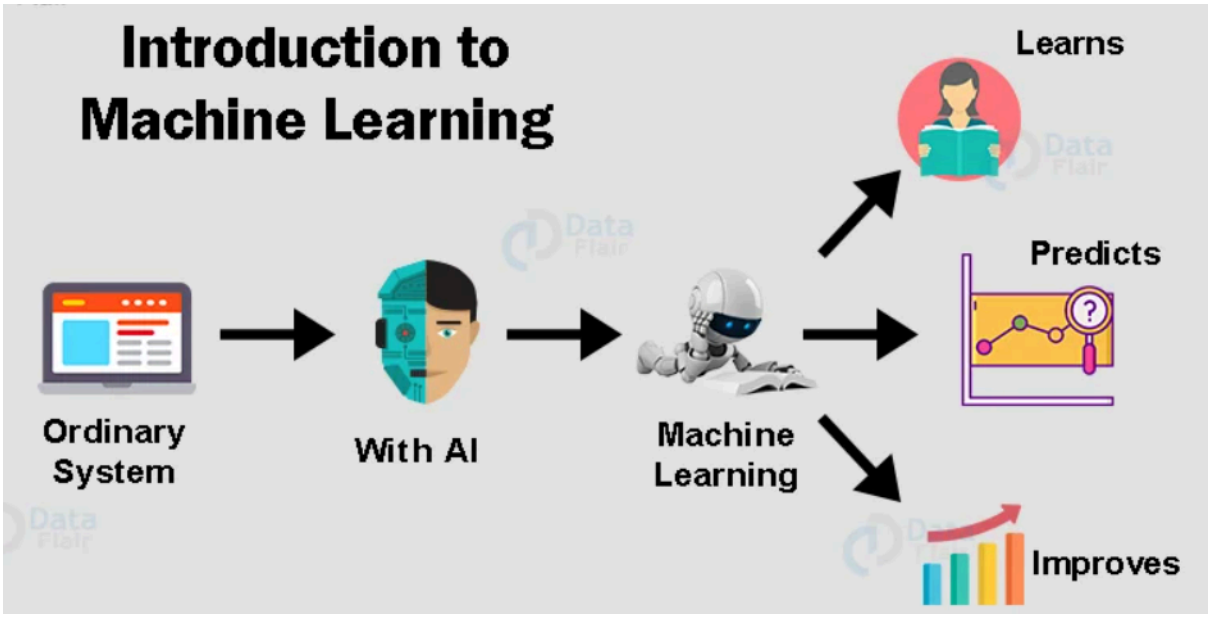
	st	acclen	arcode	phnum	intplan	voice	nummailmes	tdmin	tdcal	tdchar	temin	tecal	tecahr	tnmin	tncal	tnchar	timin	tical	tichar	ncsc	label
0	KS	128	415	382-4657	no	yes	25	265.1	110	45.07	197.4	99	16.78	244.7	91	11.01	10.0	3	2.70	1	False.
1	OH	107	415	371-7191	no	yes	26	161.6	123	27.47	195.5	103	16.62	254.4	103	11.45	13.7	3	3.70	1	False.
2	NJ	137	415	358-1921	no	no	0	243.4	114	41.38	121.2	110	10.30	162.6	104	7.32	12.2	5	3.29	0	False.
3	OH	84	408	375-9999	yes	no	0	299.4	71	50.90	61.9	88	5.26	196.9	89	8.86	6.6	7	1.78	2	False.
4	OK	75	415	330-6626	yes	no	0	166.7	113	28.34	148.3	122	12.61	186.9	121	8.41	10.1	3	2.73	3	False.

MODEL BUILDING

Machine learning-

Machine learning is an application of AI that enables systems to learn and improve from experience without being explicitly programmed. Machine learning focuses on developing computer programs that can access data and use it to learn for themself.

Machine learning is an important component of the growing field of data science. Through the use of statistical methods, algorithms are trained to make classifications or predictions, uncovering key insights within data mining projects. These insights subsequently drive decision-making within applications and businesses, ideally impacting key growth metrics.



With the constant evolution of the field, there has been a subsequent rise in the uses, demands, and importance of machine learning. Big data has become quite a buzzword in the last few years; that's in part due to increased sophistication of machine learning, which helps analyze those big chunks of big data. Machine learning has also changed the way data

extraction, and interpretation is done by involving automatic sets of generic methods that have replaced traditional statistical techniques.

Application of Machine Learning-

Below are some most trending real-world applications of Machine Learning:

- Image Recognition- Image recognition is one of the most common applications of machine learning. It is used to identify objects, persons, places, digital images, etc. The popular use case of image recognition and face detection is, Automatic friend tagging suggestion.
- Speech Recognition- While using Google, we get an option of "Search by voice," it comes under speech recognition, and it's a popular application of machine learning.
- Product Recommendations- Machine learning is widely used by various e-commerce and entertainment companies such as Amazon, Netflix, etc., for product recommendation to the user. Whenever we search for some product on Amazon, then we started getting an advertisement for the same product while internet surfing on the same browser and this is because of machine learning.
- Email Spam and Malware Filtering- Whenever we receive a new email, it is filtered automatically as important, normal, and spam. We always receive an important mail in our inbox with the important symbol and spam emails in our spam box, and the technology behind this is Machine learning
- Stock Market Trading- Machine learning is widely used in stock market trading. In the stock market, there is always

a risk of up and downs in shares, so for this machine learning's long short term memory neural network is used for the prediction of stock market trends

- Transportation and Commuting (Uber)- If you have used an app to book a cab, you are already using Page 29 Machine Learning to an extent. It provides a personalized application which is unique to you. Automatically detects your location and provides options to either go home or office or any other frequent place based on your History and Patterns.
- Traffic Prediction- If we want to visit a new place, we take help of Google Maps, which shows us the correct path with the shortest route and predicts the traffic conditions. It predicts the traffic conditions such as whether traffic is cleared, slow-moving, or heavily congested with the help of two ways:
- Real Time location of the vehicle form Google Map app and sensors
- Average time has taken on past days at the same time.

Pros and Cons of Machine Learning-

PROS::

- It is automatic: In machine learning, the whole process of data interpretation and analysis is done by computer. No men intervention is required for the prediction or interpretation of data. The whole process of machine learning is machine starts learning and predicting the algorithm or program to give the best result. One of the examples in the Google home that detect the voice and them accordingly finds out the result that the user wants, and antivirus software detects the virus of the computer and fixes it.
- It is used in various fields: Machine learning is used in various fields of life like education, medicine, engineering, etc. From a very small application to very big and complicated structured machines that help in the prediction and analysis of data. It not only becomes the healthcare provider but also provides more personal services to the potential customer.
- It can handle varieties of data: Even in an uncertain and dynamic environment, it can handle a variety of data. It is multidimensional as well as a multitasker.
- Scope of advancement: As humans after gaining experience improve themselves in the same way machine learning improve themselves and become more accurate and efficient in work. This led to better decisions.
- Can identify trends and patterns: A machine can learn more when it gets more data and since it gets more data it also learns the pattern and trend for example for a social networking site like Facebook people surf and browses several data and their interest is recorded and understand the pattern and shows the same or similar trend to them to keep their interest within the same app. In this way machine learning help in identifying trends and patterns.

- Considered best for Education: Machine learning is considered best for education as education is dynamic and nowadays smart classes, distance learning, and e-learning for students have increased a lot. Smart machine learning will act as a teacher and keep students updated with the current scenario of the world. The same thing happens in shopping or e-business people need to remain updated therefore they are shown the current trends of the world

CONS::

- Chance of error or fault is more: Although machine learning is considered to be more accurate it is highly vulnerable. For example, a set of programs provided to the machine may be biased or consist of errors. The same program is used to make another forecast or prediction then there will be a chain of errors.
- Data requirement is more: The more data a machine gets the more accurate and efficient it becomes thus more data is required to input to the machine for better forecasting or decision making. But it may sometimes not be possible. Also, the data must be unbiased and of good quality. Data requirements are problematic sometimes.
- Time-consuming and more resources required: There can be times when the learning process of the machine may take a lot of time because the effectiveness and efficiency can only come through experience which again requires time. Also, the resources required are more for example additional computers may be required.
- Inaccuracy of interpretation of data: As we have already seen that a little manipulation or biased data could lead to

a long drawn error chain and therefore there are chances of the inaccuracy of interpretation also.

- More space required: As more data is required for interpretation more space is required to store the data which is one of the shortcomings of machine learning. More data means more knowledge or material to learn from for the machine, this requires a lot of space to store or manage data for further decision making.

Why is Machine Learning important?

- Machine learning is important because it gives enterprises a view of trends in customer behavior and business operational patterns, as well as supports the development of new products. Many of today's leading companies, such as Facebook, Google and Uber, make machine learning a central part of their operations. Machine learning has become a significant competitive differentiator for many companies.

Traditional Programming vs Machine Learning

- Traditional programming is a manual process—meaning a person (programmer) creates the program. But without

anyone programming the logic, one has to manually formulate or code rules.In machine learning, on the other hand, the algorithm automatically formulates the rules from the data.

- Unlike traditional programming, machine learning is an automated process. It can increase the value of your embedded analytics in many areas, including data prep, natural language interfaces, automatic outlier detection, recommendations, and causality and significance detection. All of these features help speed user insights and reduce decision bias. For example, if you feed in customer demographics and transactions as input data and use historical customer churn rates as your output data, the algorithm will formulate a program that can predict if a customer will churn or not. That program is called a predictive model.
- Let's take an example to see how powerful machine learning is. Say you want to create a program that detects a person's activity (walking, running, jogging, or biking) from their speed. You'll have difficulties solving this problem with the traditional approach because people walk, run, and bike at different speeds depending on their age, health, environment, etc.
- However, suppose you chose machine learning to build the same problem. In that case, all you have to do is get tons of examples of people doing different activities along with their labels (i.e., the type of activity). The computer will then learn and create a model that can predict a person's actions based on their speed.

Machine Learning Process-

There are five main steps in the machine learning process:

- **Step 1:** Data Acquisition: The first step in the machine learning process is to get the data. This will depend on the type of data you are gathering and the source of data. This can be either static data from an existing database or real-time data from an IoT system or data from other repositories.
- **Step 2:** Data Cleaning: All real-world data is often unorganized, redundant, or has missing elements. In order to feed data into the machine learning model, we need to first clean, prepare and manipulate the data. This is the most crucial step in the machine learning workflow and takes up the most time as well. Having clean data means that you can get a more accurate model down the road. Data can be in any format - CSV, XML, JSON, etc. After cleaning the data, you need to then convert these data into valid formats that can be fed onto the machine learning platform.
- **Step 3:** Model Training: The next step in the machine learning workflow is to train the model. A machine learning algorithm is used on the training dataset to train the model. This algorithm leverages mathematical modeling to learn and predict behaviors. These algorithms can fall into three broad categories - binary, classification, and regression.
- **Step 4:** Model Testing: After the model is trained, we need to test and validate it for further processing. By using the testing dataset obtained from Step 3, we can check the accuracy of the model. If the results are not satisfactory, the model should be further improved. The model is trained and improved over and over again until the results are satisfactory.

Here are some things you can do to refine and improve the model:

- Review the model with the business stakeholders and take in their inputs
- Reconsider the algorithm you have chosen to train the model
- Adjust the parameters of the algorithm you have chosen (even small adjustments can have significant impacts)

- **Step 5:** Deployment: Once the model is trained, deploy and pipeline it to production for application consumption.

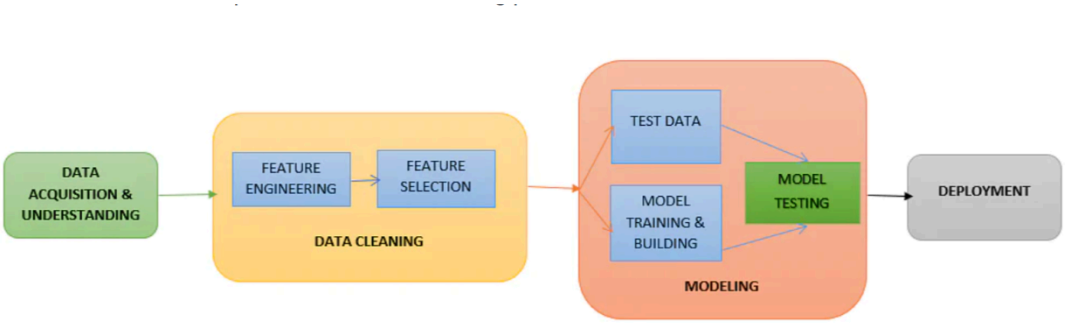


Fig: Machine learning process (source)

The important part is to keep iterating until you find a model that fits your project the most.

Types of Machine Learning-

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

1. SUPERVISED MACHINE LEARNING

As its name suggests, Supervised machine learning is based on supervision. It means in the supervised learning technique, we train the machines using the "labeled" dataset, and based on the training, the machine predicts the output. Here, the labeled 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.

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.

Supervised machine learning can be classified into two types of problems, which are given below:

Classification- Classification algorithms are used to solve the classification problems in which the output variable is categorical, such as ‘Yes’ or ‘No’, ‘Male’ or ‘Female’, ‘Red’ or ‘Blue’, etc. The classification algorithms predict the categories present in the dataset. Some real-world examples of classification algorithms are Spam Detection, Email filtering, etc.

Some popular classification algorithms are given below:

- *Random Forest Algorithm*
- *Decision Tree Algorithm*
- *Logistic Regression Algorithm*
- *Support Vector Machine Algorithm*

Regression- Regression algorithms are used to solve regression problems in which there is a linear relationship between input and output variables. These are used to predict continuous output variables, such as market trends, weather prediction, etc.

Some popular Regression algorithms are given below:

- *Simple Linear Regression Algorithm*
- *Multivariate Regression Algorithm*
- *Decision Tree Algorithm*
- *Lasso Regression*

2. UNSUPERVISED MACHINE LEARNING :

Unsupervised 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 unlabeled 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.

Example: 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.

Unsupervised Learning can be further classified into two types, which are given below:

Clustering- The clustering technique is used when we want to find the inherent groups from the data. It is a way to group the objects into a cluster such that the objects with the most similarities remain in one group and have fewer or no similarities with the objects of other groups. An example of the clustering algorithm is grouping the customers by their purchasing behavior.

Some of the popular clustering algorithms are given below:

- *K-Means Clustering algorithm*
- *Mean-shift algorithm*
- *DBSCAN Algorithm*
- *Principal Component Analysis*
- *Independent Component Analysis*

Association- Association rule learning is an unsupervised learning technique, which finds interesting relations among variables within a large dataset. The main aim of this learning algorithm is to find the dependency of one data item on another data item and map those variables accordingly so that it can generate maximum profit. This algorithm is mainly applied in Market Basket analysis, Web usage mining, continuous production, etc.

Some popular algorithms of Association rule learning are

- *Apriori Algorithm*
- *Eclat*
- *FP-growth algorithm*

3. SEMI-SUPERVISED LEARNING

Semi-Supervised learning is a type of Machine Learning algorithm that lies between Supervised and Unsupervised machine learning. It represents the intermediate ground between Supervised (With Labeled training data) and Unsupervised learning (with no labeled training data) algorithms and uses the combination of labeled and unlabeled datasets during the training period.

To overcome the drawbacks of supervised learning and unsupervised learning algorithms, the concept of Semi-supervised learning is introduced. The main aim of semi-supervised learning is to effectively use all the available data, rather than only labeled data like in supervised learning. Initially, similar data is clustered along with an unsupervised learning algorithm, and further, it helps to label the unlabeled data into labeled data. It is because labeled data is a comparatively more expensive acquisition than unlabeled data.

Example: Supervised learning is where a student is under the supervision of an instructor at home and college. Further, if that student is self-analysing the same concept without any help from the instructor, it comes under unsupervised learning. Under semi-supervised learning, the student has to revise himself after analyzing the same concept under the guidance of an instructor at college.

4. REINFORCEMENT 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 gets punished for each bad action; hence the goal of a reinforcement learning agent is to maximize the rewards.

In reinforcement learning, there is no labeled 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.

Reinforcement learning is categorized mainly into two types of methods/algorithms:

- Positive Reinforcement Learning: Positive reinforcement learning specifies increasing the tendency that the required behaviour would occur again by adding something. It enhances the strength of the behaviour of the agent and positively impacts it.
- Negative Reinforcement Learning: Negative reinforcement learning works exactly opposite to the positive RL. It increases the tendency that the specific behaviour would occur again by avoiding the negative condition.

Machine Learning Algorithms-

Linear Regression-

In this process, a relationship is established between independent and dependent variables by fitting them to a

line. This line is known as the regression line and represented by a linear equation

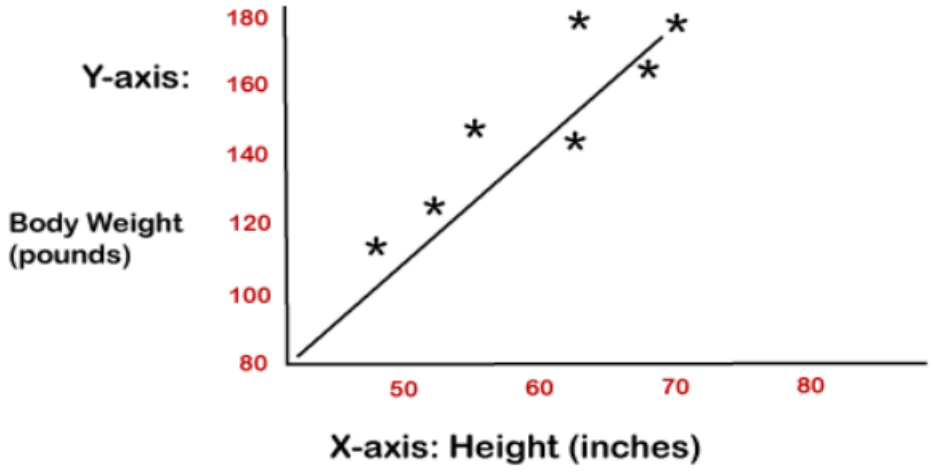
Y= a *X + b. In this equation:

- Y – Dependent Variable
- a – Slope
- X – Independent variable
- b – Intercept

The coefficients a & b are derived by minimizing the sum of the squared difference of distance between data points and the regression line.

Linear regression is further divided into two types:

- Simple Linear Regression: In simple linear regression, a single independent variable is used to predict the value of the dependent variable.
- Multiple Linear Regression: In multiple linear regression, more than one independent variables are used to predict the value of the dependent variable.
- The below diagram shows the linear regression for prediction of weight according to height:

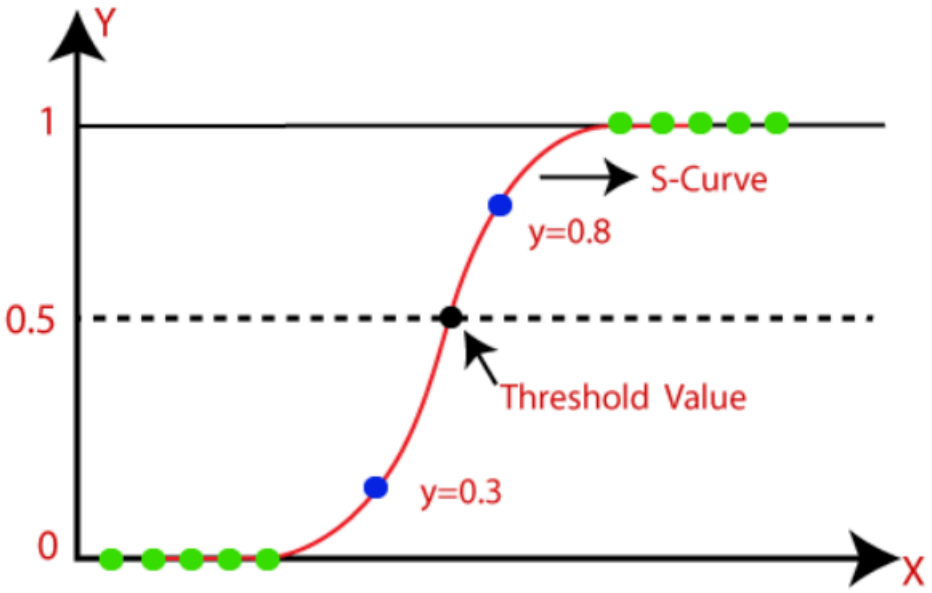


Logistic Regression-

Logistic regression is the supervised learning algorithm, which is used to predict the categorical variables or discrete values. It can be used for the classification problems in machine learning, and the output of the logistic regression algorithm can be either Yes or NO, 0 or 1, Red or Blue, etc.

Instead of fitting the best fit line, it forms an S-shaped curve that lies between 0 and 1. The S-shaped curve is also known as a logistic function that uses the concept of the threshold. Any value above the threshold will tend to 1, and below the threshold will tend to 0.

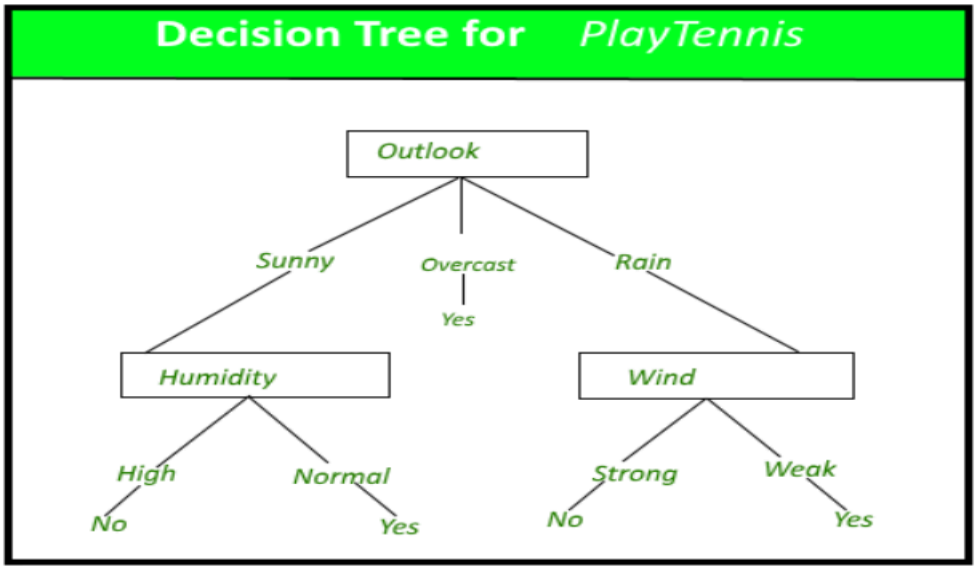
The below image is showing the logistic function:



Decision Tree-

Decision Tree algorithm in machine learning is one of the most popular algorithm in use today; this is a supervised learning algorithm that is used for classifying problems. It works well classifying for both categorical and continuous dependent variables. In this algorithm, we split the

population into two or more homogeneous sets based on the most significant attributes/ independent variables.

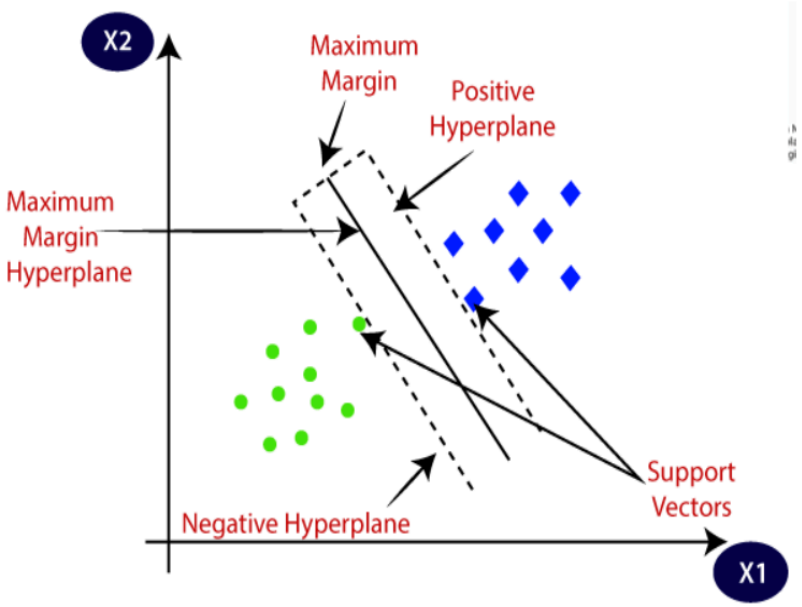


SVM (Support Vector Machine) Algorithm-

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. Consider the below diagram in which there are two different categories that are classified using a decision boundary or hyperplane.

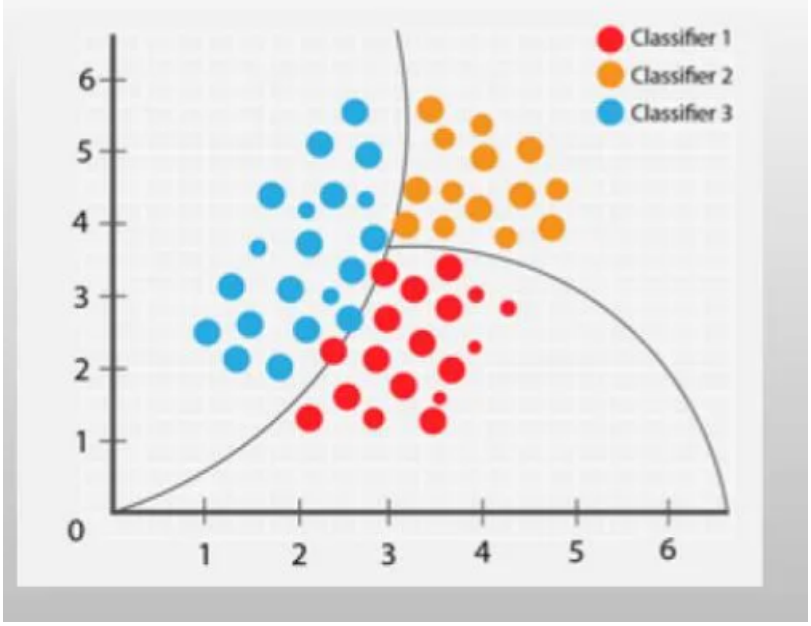


Naive Bayes Algorithm-

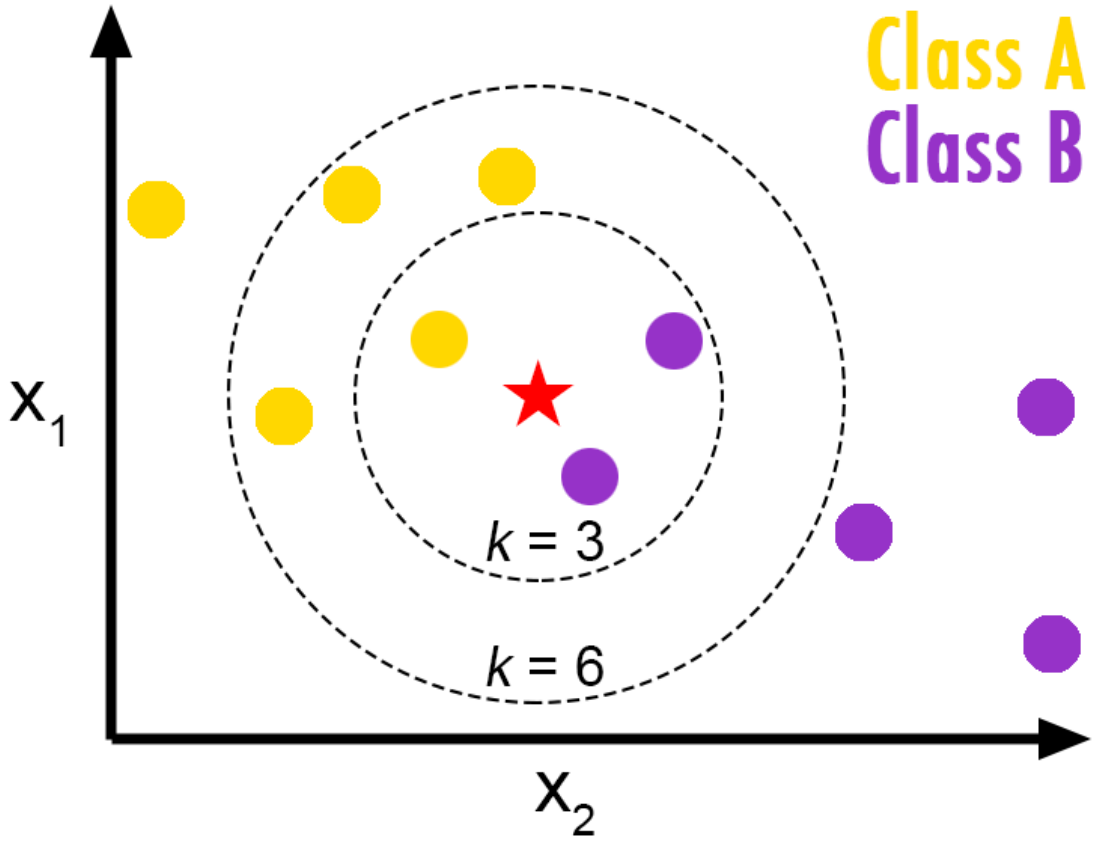
A Naive Bayes classifier assumes that the presence of a particular feature in a class is unrelated to the presence of any other feature.

Even if these features are related to each other, a Naive Bayes classifier would consider all of these properties independently when calculating the probability of a particular outcome.

A Naive Bayesian model is easy to build and useful for massive datasets. It's simple and is known to outperform even highly sophisticated classification methods.



- ***KNN (K- Nearest Neighbors) Algorithm-***
  This algorithm can be applied to both classification and regression problems. Apparently, within the Data Science industry, it's more widely used to solve classification problems. It’s a simple algorithm that stores all available cases and classifies any new cases by taking a majority vote of its k neighbors. The case is then assigned to the class with which it has the most in common. A distance function performs this measurement.

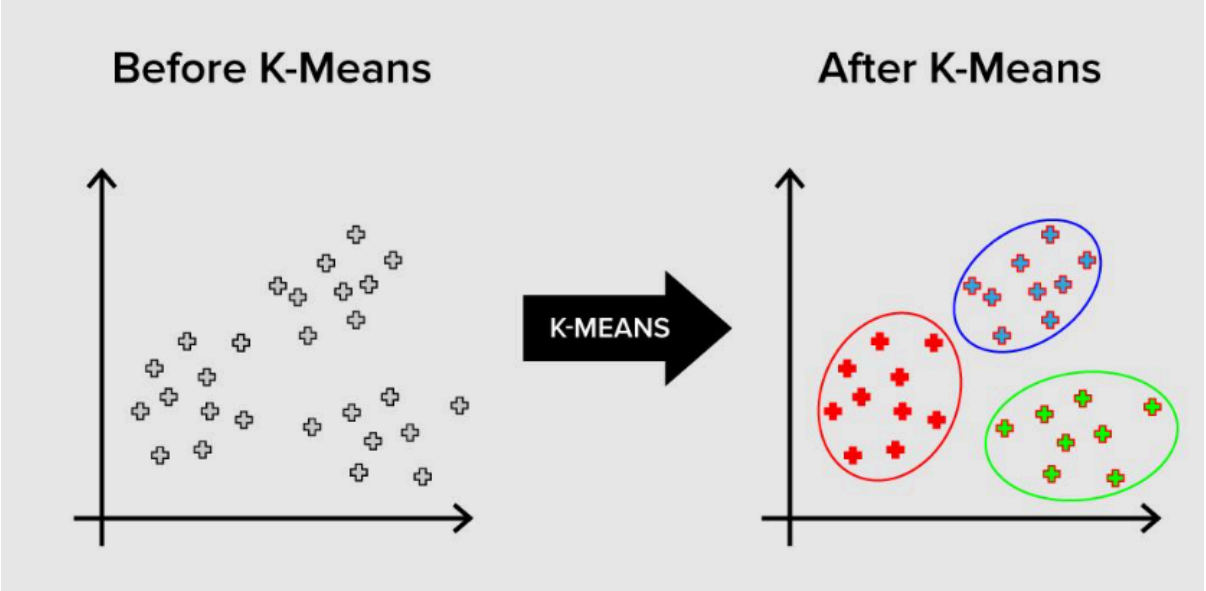


K-Means-

It is an unsupervised learning algorithm that solves clustering problems. Data sets are classified into a particular number of clusters (let's call that number K) in such a way that all the data points within a cluster are homogenous and heterogeneous from the data in other clusters.

- The K-means algorithm picks k number of points, called centroids, for each cluster.
- Each data point forms a cluster with the closest centroids, i.e., K clusters.
- It now creates new centroids based on the existing cluster members.

- With these new centroids, the closest distance for each data point is determined. This process is repeated until the centroids do not change

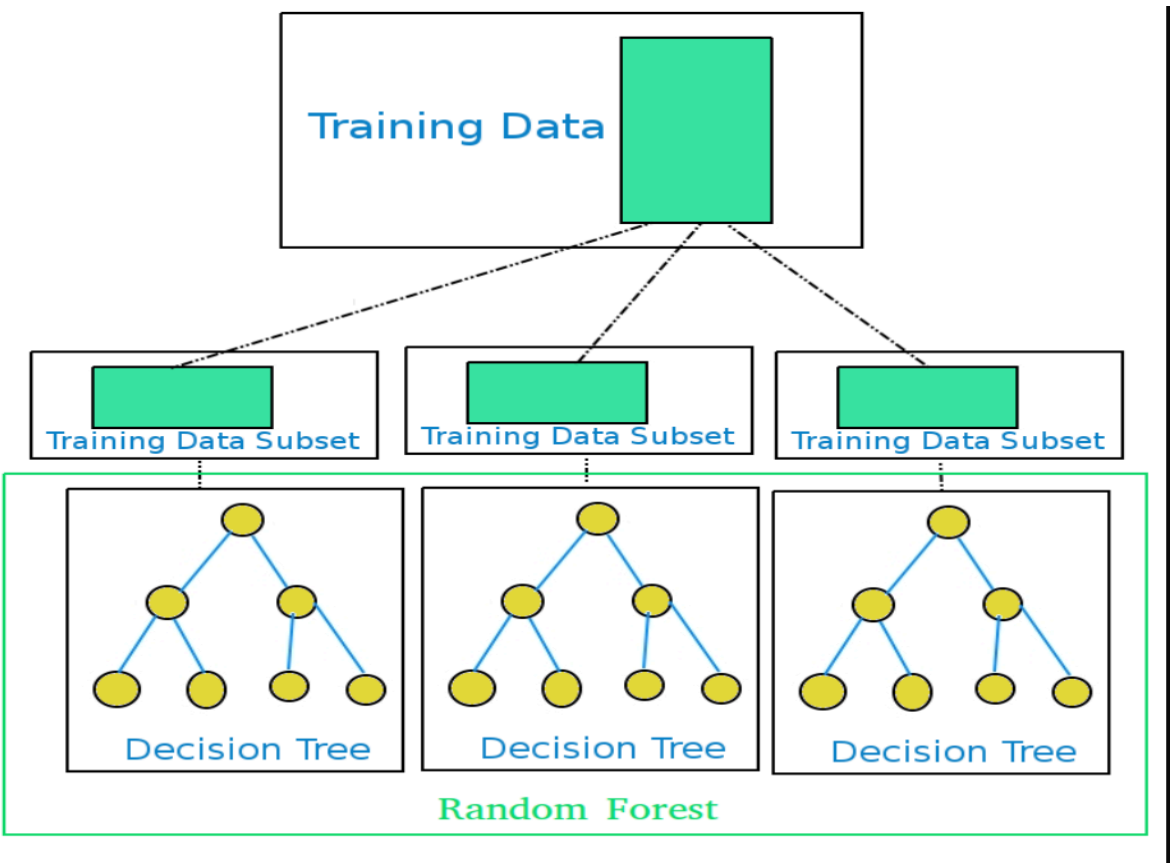


Random Forest Algorithm-

A collective of decision trees is called a Random Forest. To classify a new object based on its attributes, each tree is classified, and the tree “votes” for that class. The forest chooses the classification having the most votes (over all the trees in the forest).

Each tree is planted & grown as follows:

- If the number of cases in the training set is N, then a sample of N cases is taken at random. This sample will be the training set for growing the tree.
- If there are M input variables, a number m<<M is specified such that at each node, m variables are selected at random out of the M, and the best split on this m is used to split the node. The value of m is held constant during this process.
- Each tree is grown to the most substantial extent possible. There is no pruning.



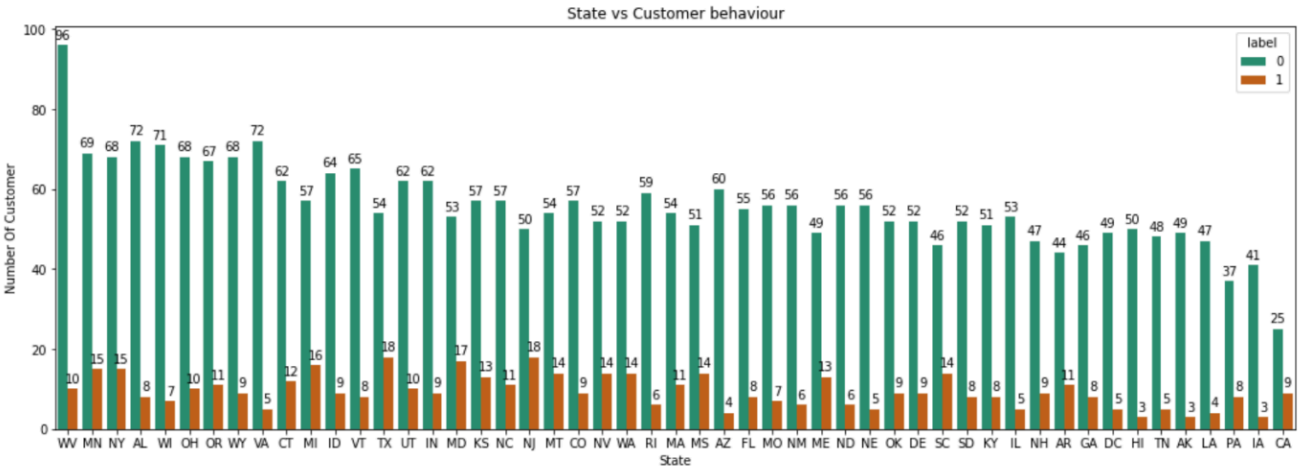
- ***Dimensionality Reduction Algorithms-***

  In today's world, vast amounts of data are being stored and analyzed by corporates, government agencies, and research organizations. As a data scientist, you know that this raw data contains a lot of information - the challenge is in identifying significant patterns and variables. Dimensionality reduction algorithms like Decision Tree, Factor Analysis, Missing Value Ratio, and Random Forest can help you find relevant details.

- ***Gradient Boosting Algorithm and AdaBoosting Algorithm-***
  These are boosting algorithms used when massive loads of data have to be handled to make predictions with high accuracy. Boosting is an ensemble learning algorithm that combines the predictive power of several base estimators to improve robustness.

Feature Engineering

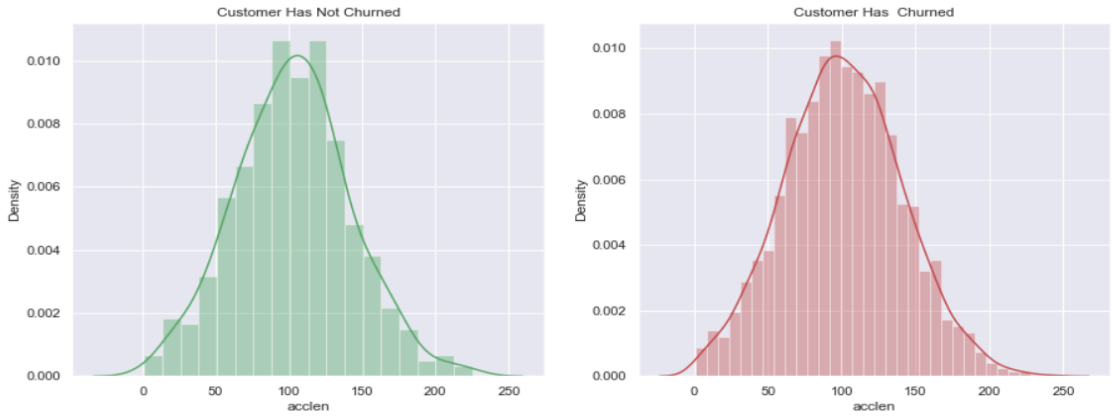
- ***State(st)***- This column tells us about the different states from where we have got most for the customers. In our dataset, we have got 51 different unique values in state column. We can use this data to make a conclusion like, the customers of which state has a high percentage of churn rate or we can say the telecom service needs to be improved there.



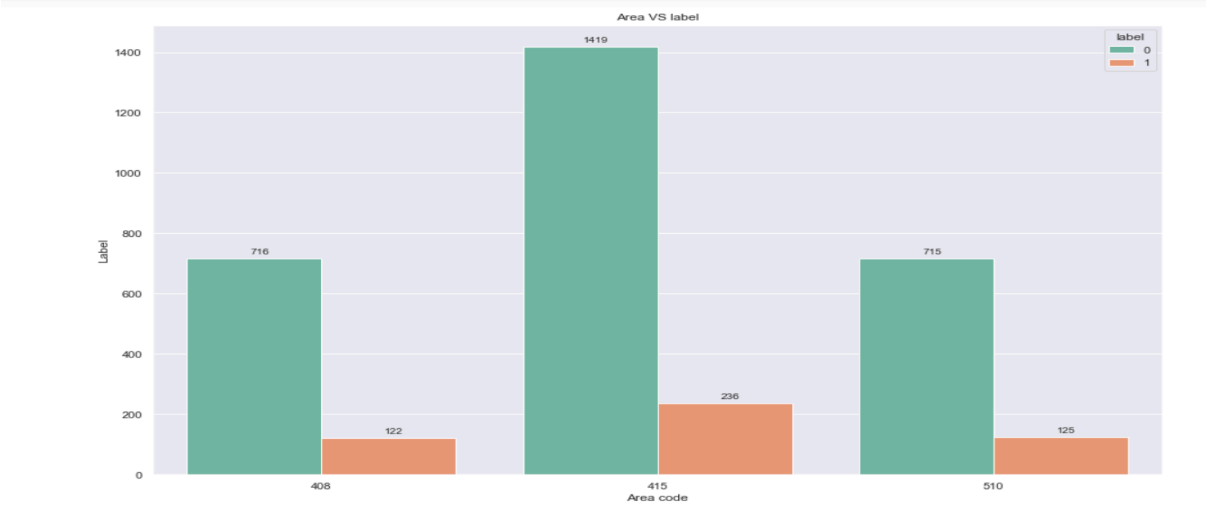
Further, we have implemented ‘ONE HOT ENCODING’ to the state column before creating the model.

- ***Account length(acclen)-*** This column describes the duration of service taken by each customer. We have used

this column to check the density of account length of the two groups i.e., churned customers and non churned customers.



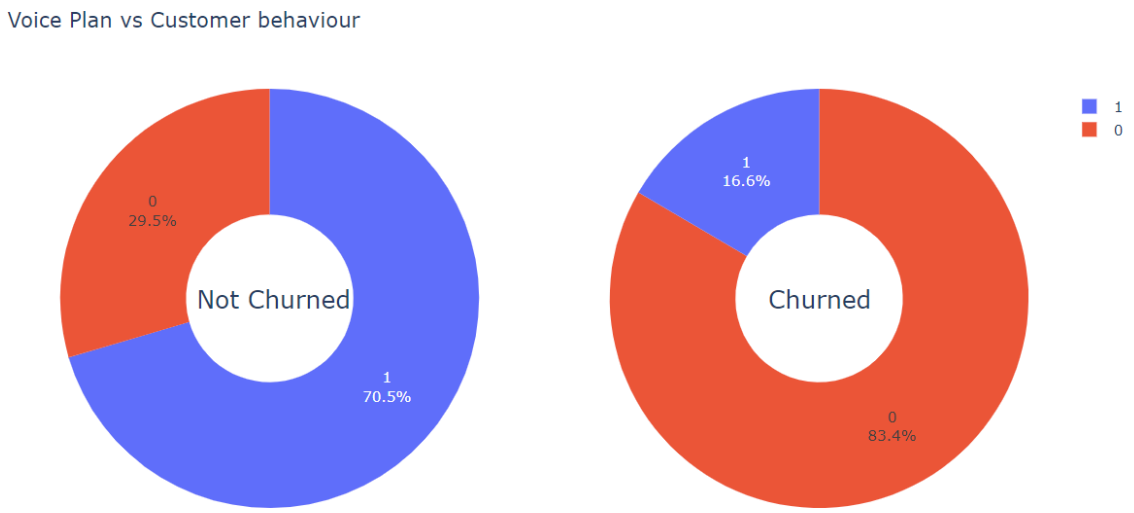
- ***Area code(arcode)-*** Area code tells us the area to which each customers belongs to. In our data set, there are 3 unique values in area code, i.e., 415, 408, 510. We have used this column to check which area has a high churning rate.
  Further, we have implemented ‘ONE HOT ENCODING’ to the arcode column before creating the model.



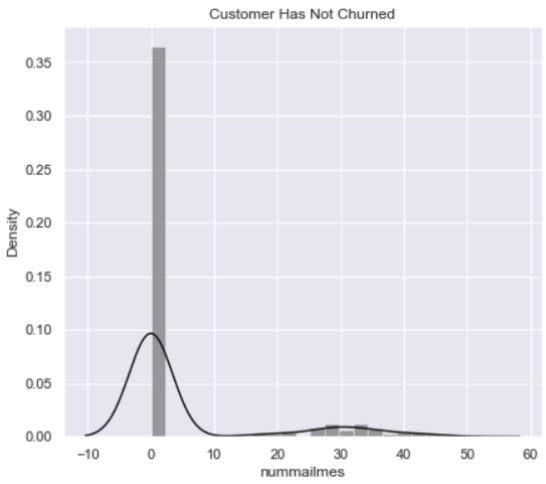
- ***Phone number (phnum)-***This is a unique column which has phone numbers of each customers. This column is used to identify every customer uniquely, it doesn't have any impact on the churn rate. Since we don't need to work on it, we will drop this column.

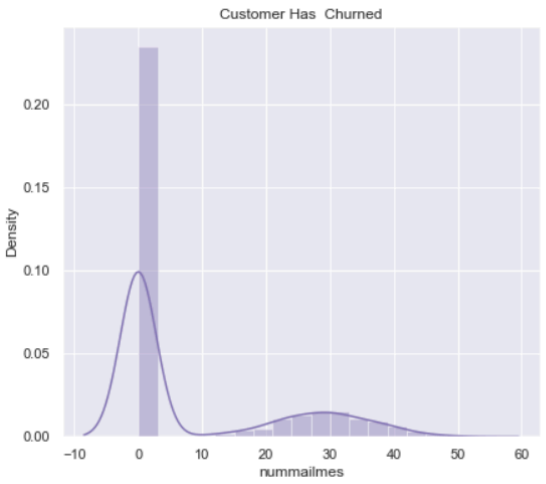
- ***Internet plan (intplan)-*** This is a column which gives us an idea that whether a customer has got internet plan or not ,so the column has binary value as 'yes' or 'no'. We will use this column to see whether the internet plan has an impact on the churn rate or not.
  We have also used label encoding in this column and replaced 'yes' into '1' and 'no' into '0'.

- ***Voice –*** This column tells us whether or not the customer has taken voice plan service and how is the voice plan throwing an impact on churn rate. The column has binary values 'yes' and 'no', further we have implemented label encoding to the column where 'yes' means '1' and 'no' means '0'.

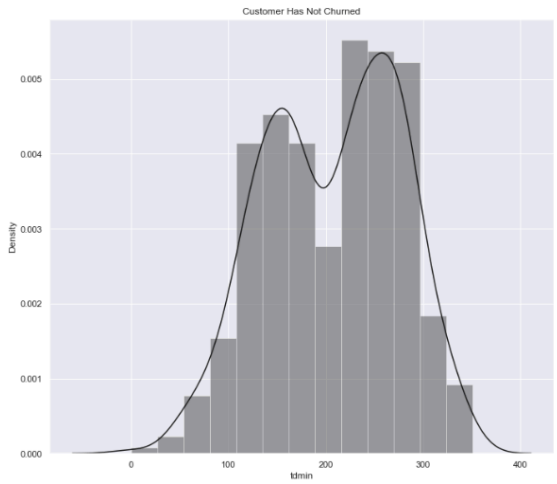


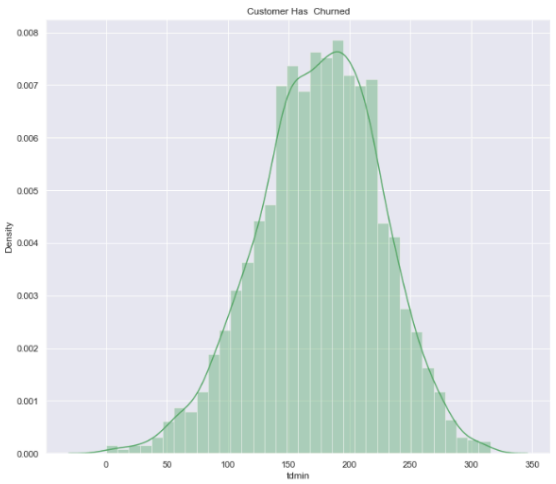
- ***Number of email messages (nummailmes)-*** This column tells us about the number of emails each customer has done. We have checked the density of email messages on both categories of customers to see whether number of email is anyhow effecting the rate of churn.



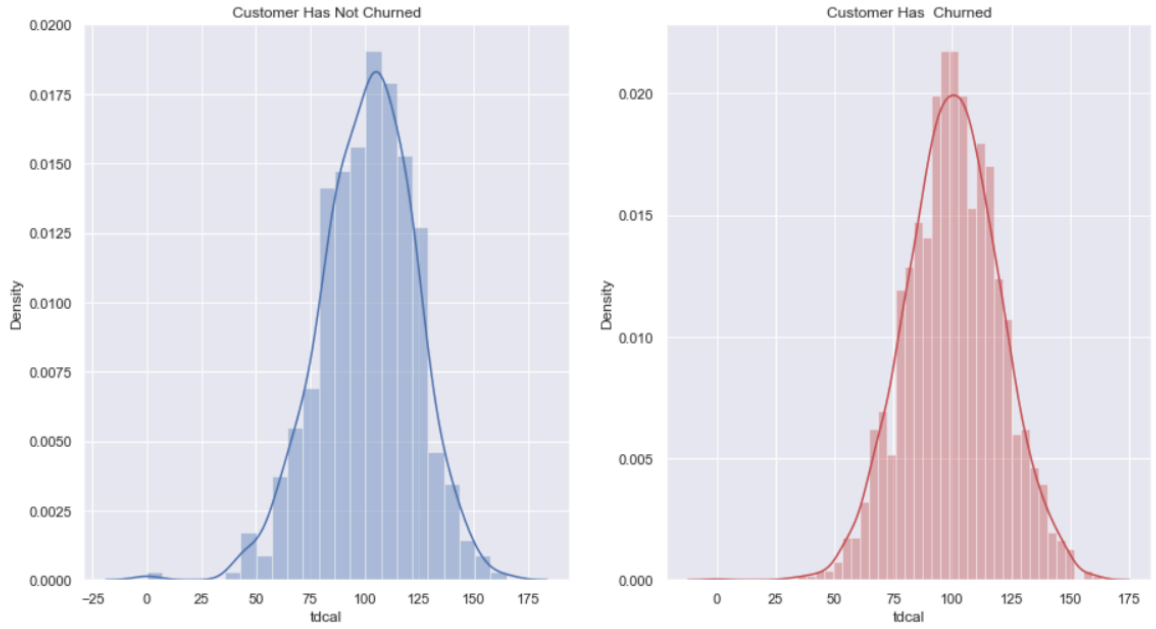


- ***Total day minutes (tdmin) -*** This column gives us an idea about the duration of calls (in minutes) of every customers in day time. This column can be used to know the service of the company at particularly at day.
  We are checking day call duration of different categories of customers separately.

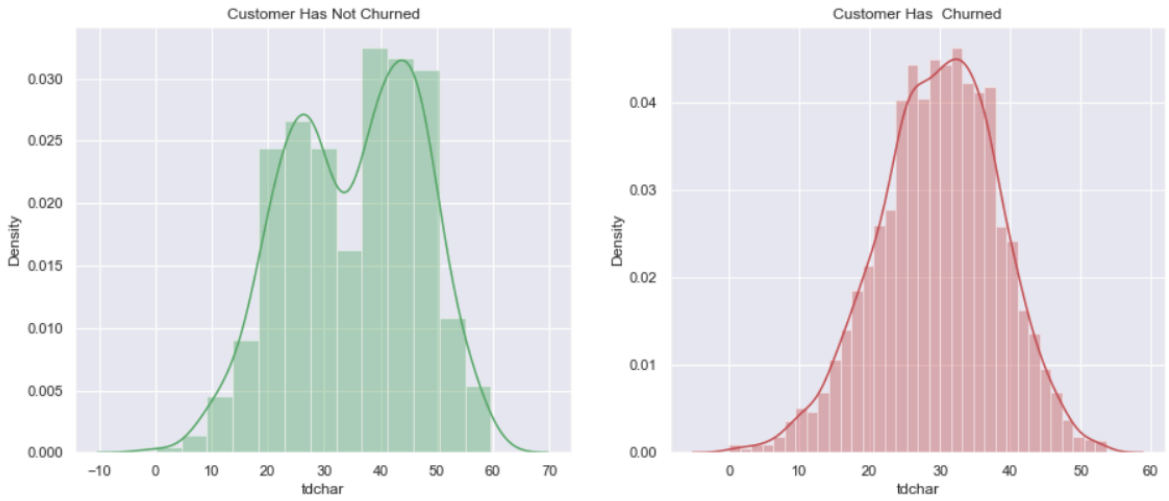




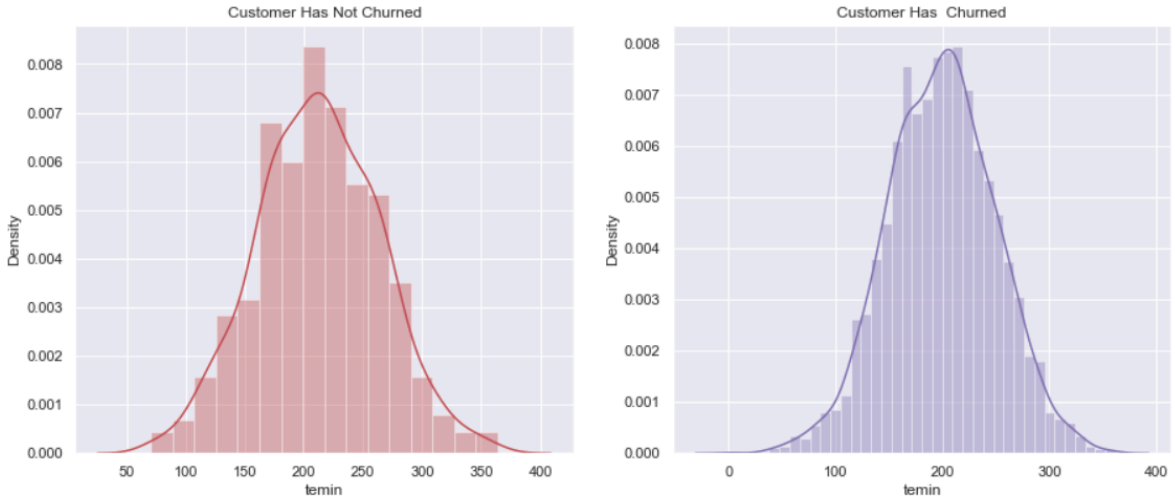
- ***Total day call (tdcal)-*** This column tells us about the number of calls made by every customers during day time. It help us to understand that how frequently a customer needs to call. We are checking no. of day call of different categories of customers separately.



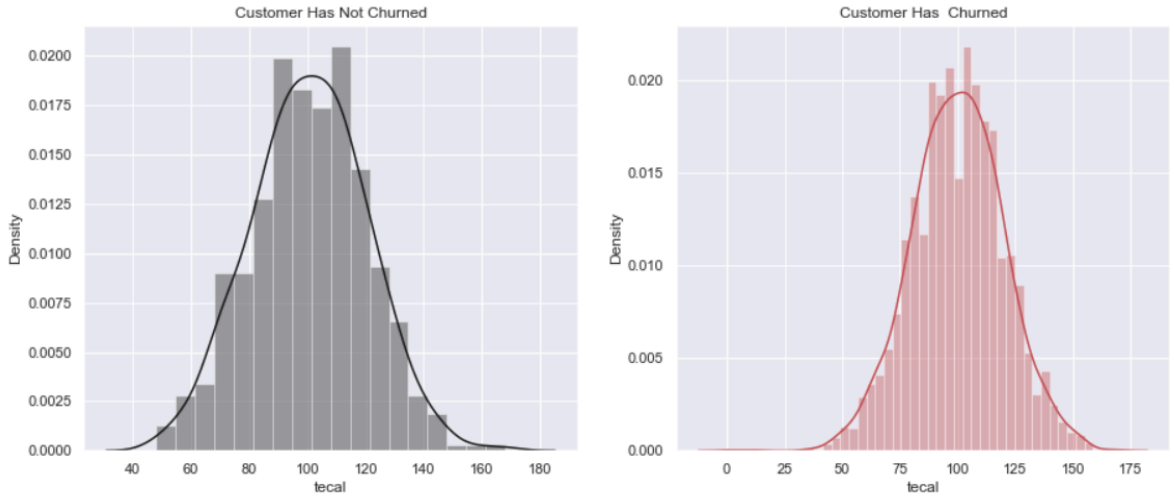
- ***Total day charges (tdchar)-*** This columns tells us about a customer's day time charges. The charge for each customer is proportional or positively related to total day calls and total day minutes.



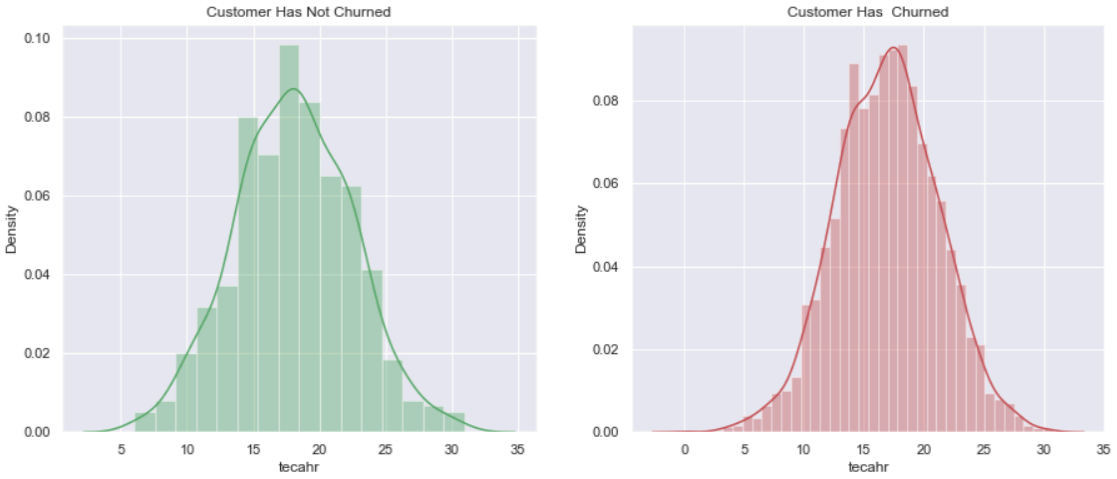
- ***Total evening minutes (temin) -*** This column gives us an idea about the duration of calls (in minutes) of every customer in evening time. This column can be used to know the service of the company at particularly at day. We are checking day call duration of different categories of customers separately.



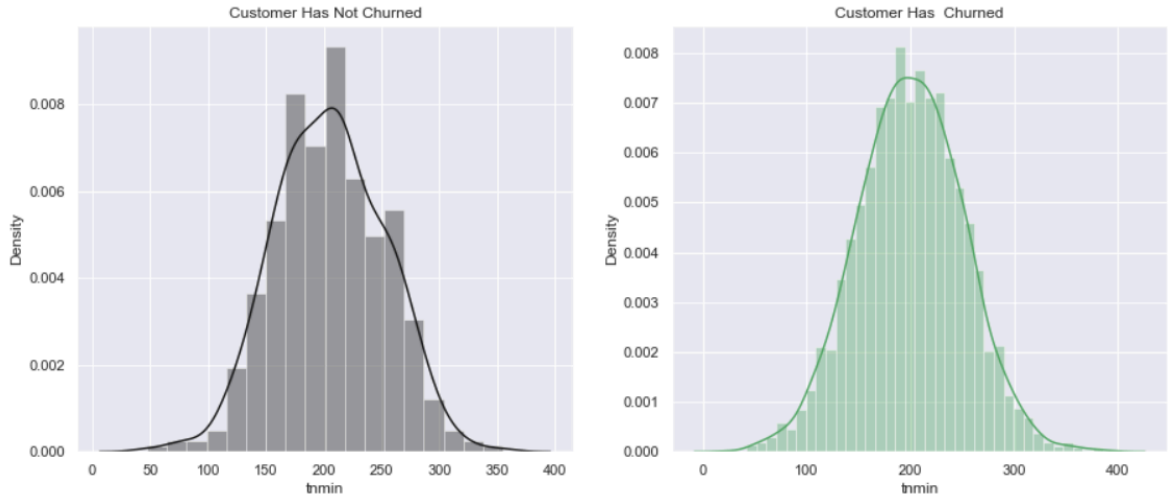
- ***Total evening call (tecal)-*** This column tells us about the number of calls made by every customers during evening time. IIt help us to understand that how frequently a customer needs to call.



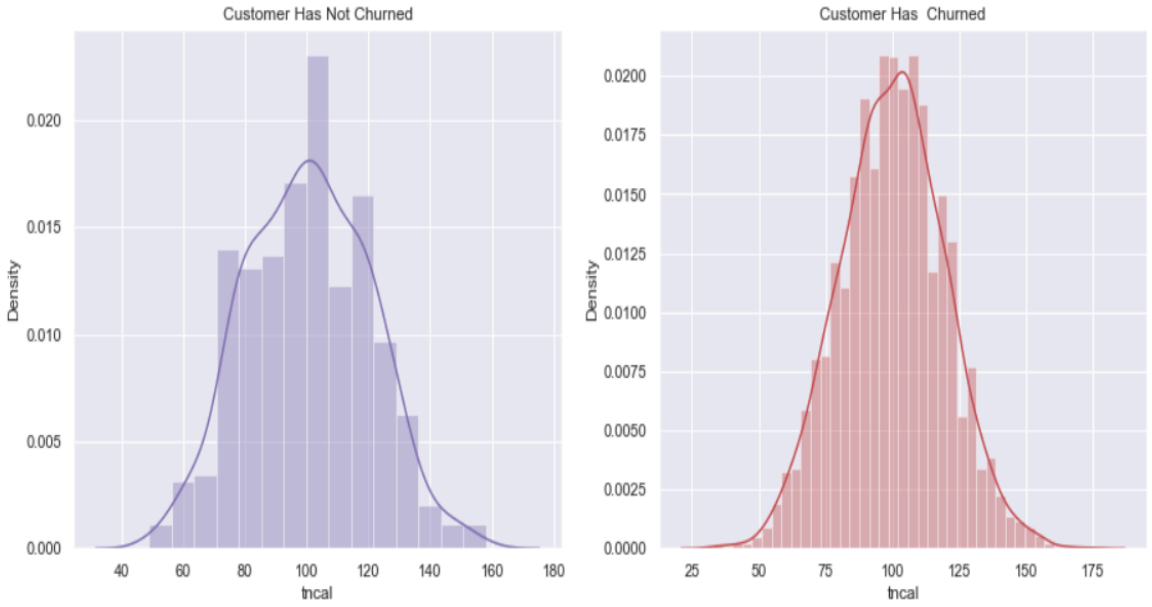
- ***Total evening charges (techar)***- This columns tells us about a customer's evening time charges. The charge for each customer is proportional or positively related to total evening calls and total evening minutes.



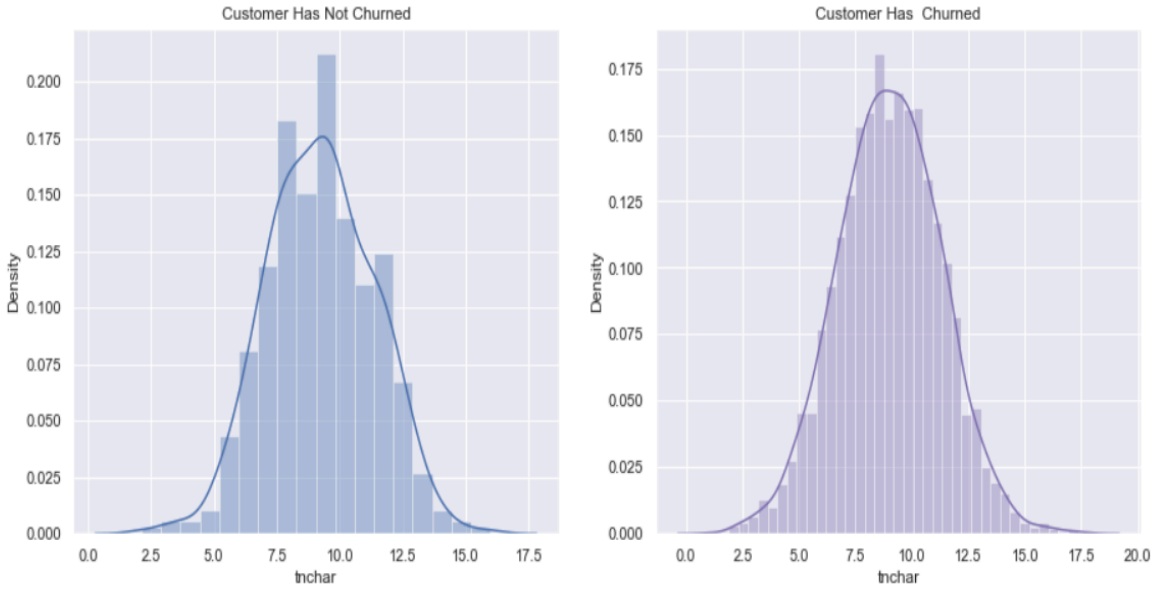
- ***Total night minutes (tnmin)*** - This column gives us an idea about the duration of calls (in minutes) of every customer in night time. This column can be used to know the service of the company at particularly at night. We are checking day call duration of different categories of customers separately.



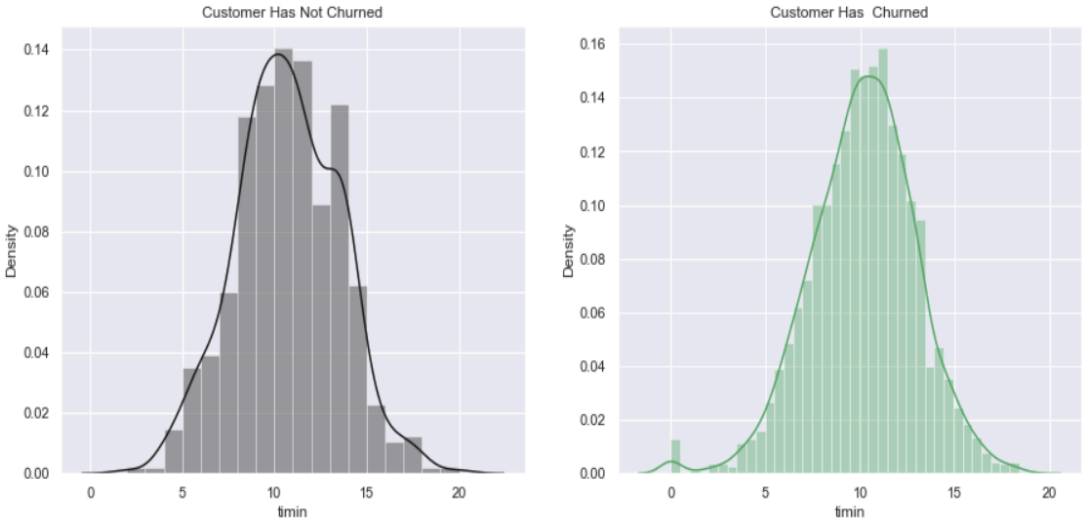
- ***Total night call (tncal)-*** This column tells us about the number of calls made by every customers during night time. ## It help us to understand that how frequently a customer needs to call.



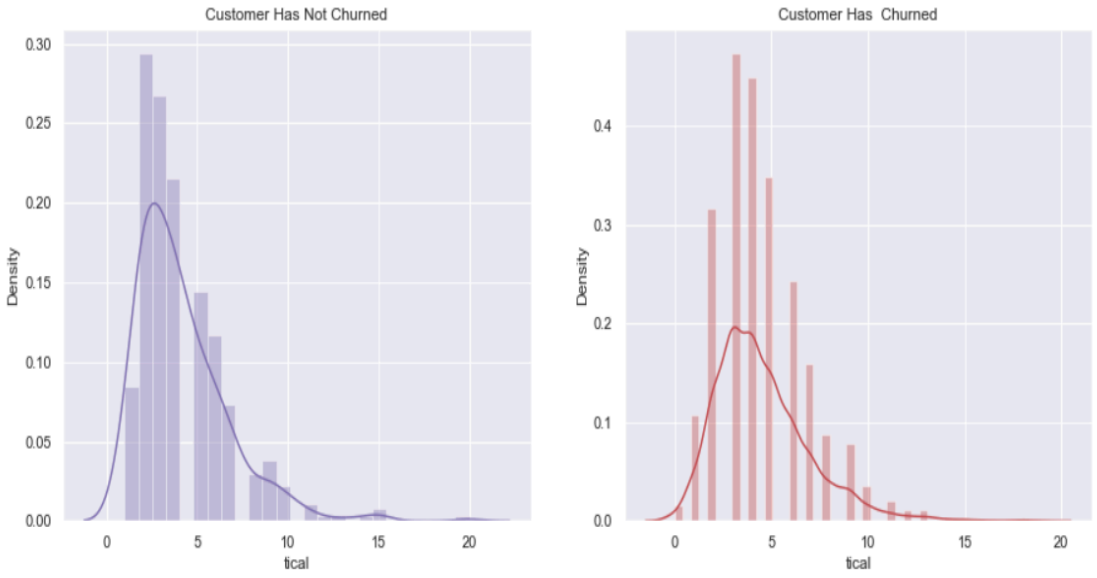
- ***Total night charges (techar)-*** This columns tells us about a customer's night time charges. The charge for each customer is proportional or positively related to total night calls and total night minutes.



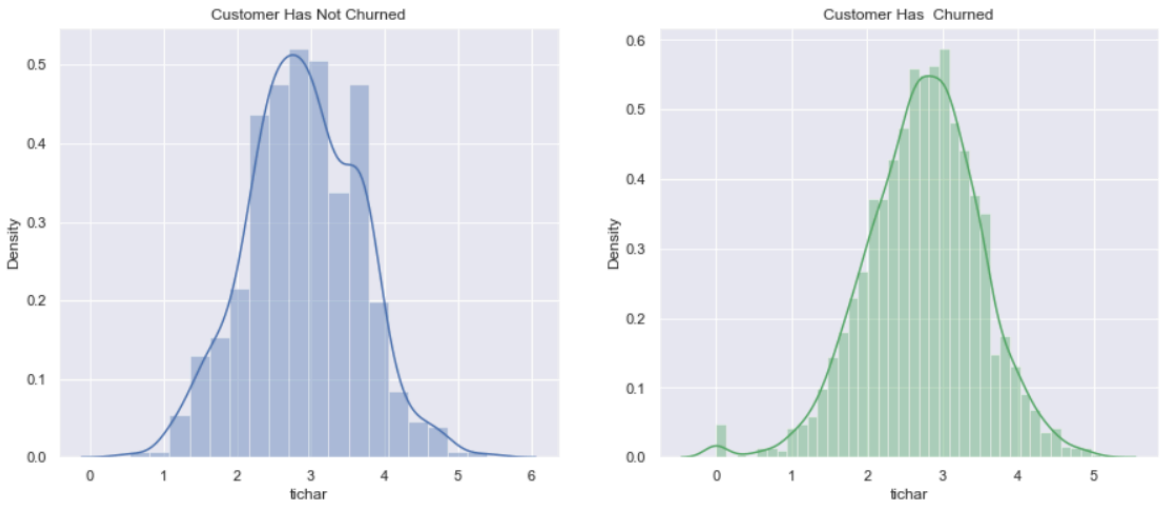
- ***Total international minutes (timin) -*** This column gives us an idea about the international call duration (in minutes) of every customer. This column can be used to know the international service of the company. We are checking day call duration of different categories of customers separately.



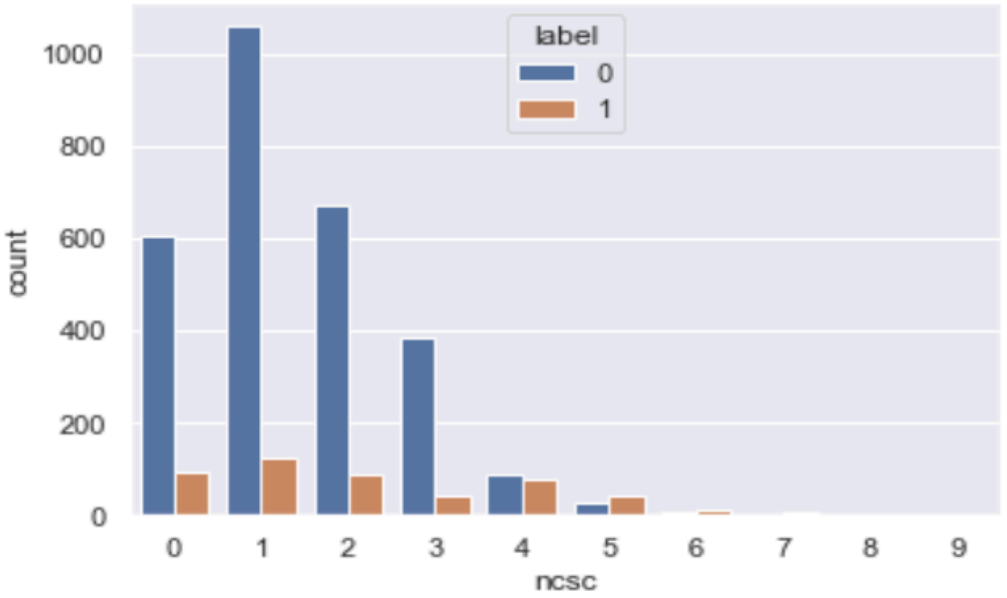
- ***Total international call (tical)-*** This column tells us about the number of international calls made by every customers . It helps us to understand that how often a customer needs to make international calls.



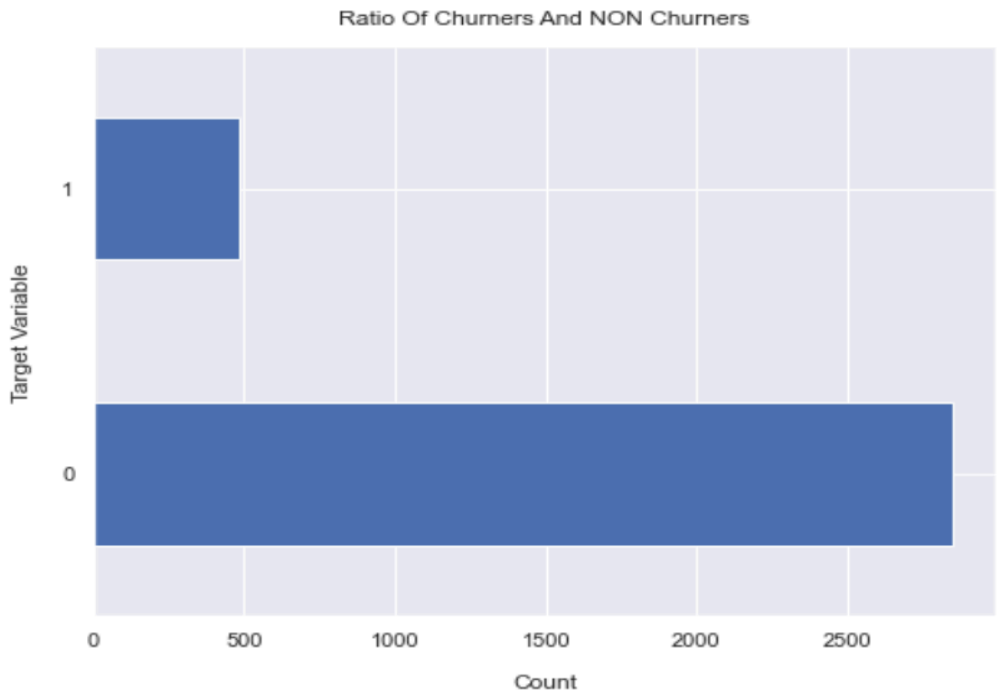
- ***Total international charges (tichar)-*** This columns tells us about a customer's international call charges. The charge for each customer is proportional or positively related to total international calls and total international minutes.



- ***Number of customer services calls made by customer (ncsc)-*** This column tells us how often the customer needs to make customer service calls. It is a known fact that customers makes customer service calls to the company when they need to complain about an issue in the service. So this column may have a positive relation with the churn rate.



- ***Labels-*** This is the key column which says that which particular customers has churned and which customers have retained(not churned).This column consists of binary values like ‘True’ and ‘False’. True means customer has churned and False means customer has not churned. This column also helps us to check if our dataset is balanced or not. We have implemented label encoding to this column and changed ‘True’ as ‘1’ and ‘False’ as ‘0’.



Data Pre-processing

- ***One Hot Encoding***-It is an encoding technique, for categorical variables where no such ordinal relationship exists and the integer encoding is not

enough. Here, the integer encoded variable is removed and a new binary variable is added for each unique integer value.

We have applied One Hot Encoding on ‘state’ and ‘area code’ columns

```
In [50]: # Applying 'ONE HOT ENCODING' to state and areacode

         df=pd.get_dummies(df,columns=["st","arcode"],drop_first=True)

In [51]: df.head()
```

Out[51]:

st_NV	st_NY	st_OH	st_OK	st_OR	st_PA	st_RI	st_SC	st_SD	st_TN	st_TX	st_UT	st_VA	st_VT	st_WA	st_WI	st_WV	st_WY	arcode_415	arcode_510
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0
0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	1	0

- ***Label Encoding-*** Label Encoding refers to converting the labels into a numeric form so as to convert them into the machine-readable form. Machine learning algorithms can then decide in a better way how those labels must be operated. It is an important pre-processing step for the structured dataset in supervised learning.
  We have applied ‘Label Encoding ‘, to intplan, label and voice columns.

```
In [11]: from sklearn import preprocessing
         label_encoder = preprocessing.LabelEncoder()
         df['intplan'] = label_encoder.fit_transform(df['intplan'])
         df['label'] = label_encoder.fit_transform(df['label'])
         df['voice'] = label_encoder.fit_transform(df['voice'])
         print(df.dtypes)
```

```
In [12]: df.sample(5)
```

Out[12]:

	st	acclen	arcode	phnum	intplan	voice	nummailmes	tdmin	tdcal	tdchar	temin	tecal	tecahr	tnmin	tncal	tnchar	timin	tical	tichar	ncsc	label
1333	NV	7	408	355-8299	0	1	30	221.4	114	37.64	165.8	116	14.09	247.0	105	11.12	10.8	12	2.92	1	0
2037	NE	86	408	399-6852	0	0	0	83.8	121	14.25	240.2	96	20.42	158.6	108	7.14	6.7	8	1.81	1	0
2093	WA	106	408	416-4464	0	0	0	193.6	66	32.91	238.2	82	20.25	176.4	107	7.94	12.9	3	3.48	0	0
2144	VA	164	415	375-1746	0	0	0	123.3	78	20.96	170.0	85	14.45	165.9	78	7.47	12.7	2	3.43	1	0
2075	ID	73	510	394-4512	0	1	28	198.2	107	33.69	139.1	123	11.82	199.1	139	8.96	8.8	1	2.38	2	0

- ***Dropping Phone number column-*** Phone number is used to identify every customer uniquely, it doesn't have any impact on the churn rate. Since we don't need to work on it, we will drop this column.

```
In [125]: df.drop(columns=["phnum"],axis=1,inplace=True)
          df.sample(5)
```

Out[125]:

	st	acclen	arcode	intplan	voice	nummailmes	tdmin	tdcal	tdchar	temin	tecal	tecahr	tnmin	tncal	tnchar	timin	tical	tichar	ncsc	label
1472	MD	76	415	1	0	0	273.3	66	46.46	263.6	121	22.41	165.2	84	7.43	12.0	7	3.24	1	1
2459	HI	105	415	0	0	0	211.1	99	35.89	176.7	66	15.02	221.5	96	9.97	14.7	7	3.97	4	0
549	OK	121	408	0	1	31	237.1	63	40.31	205.6	117	17.48	196.7	85	8.85	10.1	5	2.73	4	0
2478	TN	123	415	0	1	34	305.2	80	51.88	156.5	109	13.30	280.0	81	12.60	13.2	7	3.56	1	0
2773	NJ	134	510	0	1	34	247.2	105	42.02	225.5	133	19.17	186.3	76	8.38	6.1	5	1.65	2	1

- ***Creating 3 new columns*** to check if the total calls of a day ,total charges of a day and total minutes of a day is affecting the churn rate or not.

```
In [22]: #Creating 3 new columns for total charges,total calls and total minutes
         df["t_charges"]=df[["tdchar","tecahr","tnchar"]].sum(axis=1)
         df["t_call"]=df[["tdcal","tecal","tncal"]].sum(axis=1)
         df["t_min"]=df[["tdmin","temin","tnmin"]].sum(axis=1)
```

```
In [23]: df.sample(5)
```

Out[23]:

n	arcode	intplan	voice	nummailmes	tdmin	tdcal	tdchar	temin	tecal	tecahr	tnmin	tncal	tnchar	timin	tical	tichar	ncsc	label	t_charges	t_call	t_min
7	510	1	0	0	234.1	91	39.80	163.1	105	13.86	282.5	100	12.71	10.0	3	2.70	1	0	66.37	296	679.7
8	408	0	0	0	102.1	75	17.36	219.5	97	18.66	73.7	92	3.32	9.8	5	2.65	0	0	39.34	264	395.3
2	415	0	0	0	279.1	124	47.45	180.5	108	15.34	217.5	104	9.79	9.5	11	2.57	2	1	72.58	336	677.1
8	408	0	0	0	108.6	108	18.46	209.9	126	17.84	222.6	117	10.02	7.9	5	2.13	1	1	46.32	351	541.1
2	415	0	1	19	146.5	73	24.91	246.4	65	20.94	199.0	114	8.96	4.1	4	1.11	1	0	54.81	252	591.9

- ***Standardizing the data using StandardScaler() –*** Since,all continuous columns does not have same scale, we need to standardize them.

```
In [55]: scaler = preprocessing.StandardScaler()
         X = scaler.fit_transform(X)
         print(X)

[[ 0.67648946 -0.32758048  1.6170861  ... -0.15378117  1.00692466
  -0.5804683 ]
 [ 0.14906505 -0.32758048  1.6170861  ... -0.15378117  1.00692466
  -0.5804683 ]
 [ 0.9025285  -0.32758048 -0.61839626 ... -0.15378117  1.00692466
  -0.5804683 ]
 ...
 [-1.83505538 -0.32758048 -0.61839626 ... -0.15378117 -0.99312296
   1.72274698]
 [ 2.08295458  3.05268496 -0.61839626 ... -0.15378117 -0.99312296
   1.72274698]
 [-0.67974475 -0.32758048  1.6170861  ... -0.15378117  1.00692466
  -0.5804683 ]]
```

- ***Balancing the data using SMOTE-*** SMOTE (synthetic minority oversampling technique) is one of the most commonly used oversampling methods to solve the imbalance problem. It aims to balance class distribution by randomly increasing minority class examples by replicating them. SMOTE synthesizes new minority instances between existing minority instances.

Using the label column we have seen that our data is not balanced. So to balance the data we are using an over sampling technique.

```
In [48]: # CHECKING FOR CLASS IMBALANCE
         df['label'].value_counts()/df.shape[0]*100

Out[48]: 0    85.508551
         1    14.491449
         Name: label, dtype: float64
```

BALANCING THE DATASET USING SMOTE

```
In [55]: print(f'Original dataset shape : {Counter(y)}')

         Original dataset shape : Counter({0: 2850, 1: 483})

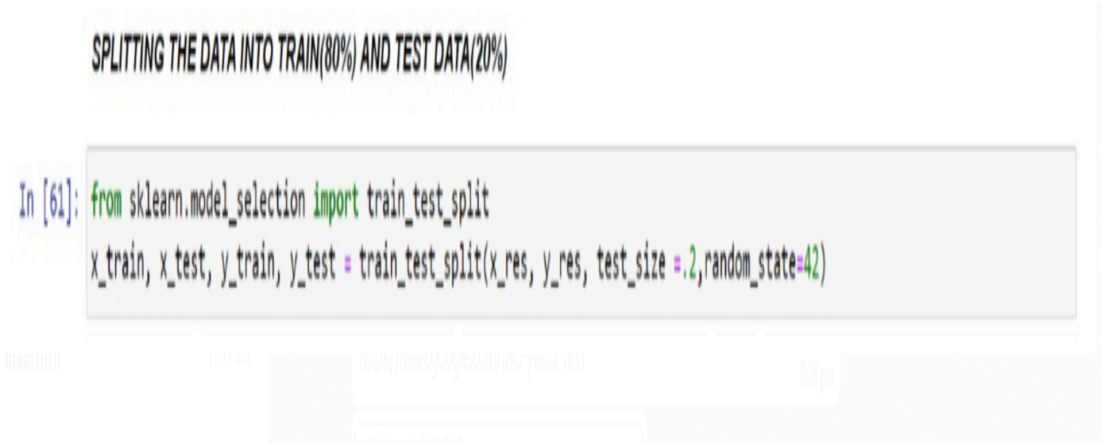
In [56]: smote = SMOTE(random_state=42)
         x_res, y_res = smote.fit_resample(X, y)

In [57]: print(f'Resampled dataset shape {Counter(y_res)}')

         Resampled dataset shape Counter({0: 2850, 1: 2850})
```

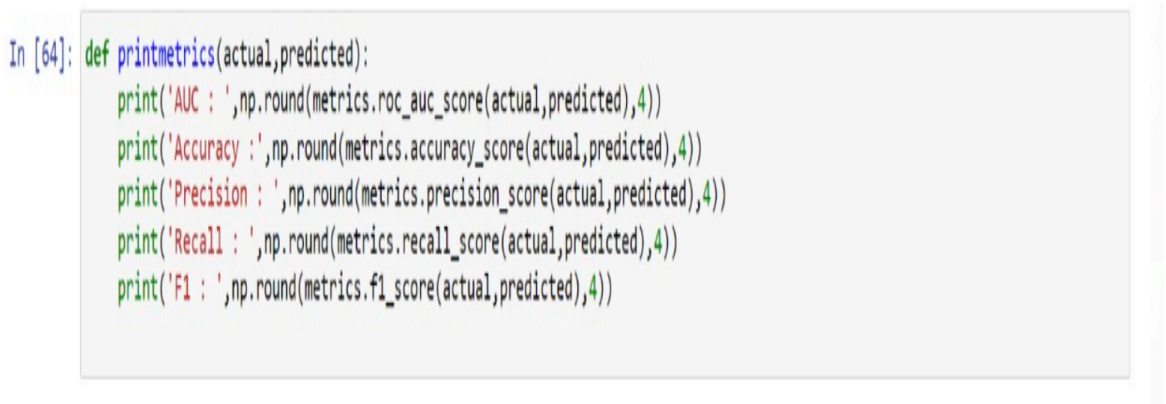
- ***Splitting the data into Train and Test-***

  We are dividing the data into train and test data. Train data is used by the model to learning and test data is used to validate the model.



IMPLEMENTING DIFFERENT MODELS TO SELECT THE BEST MODEL

Creating a function to check the different metrics of all models-



❖ *Logistic Regression-*

Before hyperparameter tuning

- **Code:**

```
model_lr=linear_model.LogisticRegression()
model.fit(x_train,y_train)
predtrain=model.predict(x_train)
predtest=model.predict(x_test)
```

- **Result:**

```
TRAINING METRICS
----------------
AUC :  1.0
Accuracy : 1.0
Precision :  1.0
Recall :  1.0
F1 :  1.0

TEST METRICS
----------------
AUC :  0.9356
Accuracy : 0.9351
Precision :  0.9168
Recall :  0.9532
F1 :  0.9346
```

The model is overfit, so we have to do hyperparameter tuning .

For hyperparameter tuning we are using GridSearchCV and for feature selection we are using RFE.

After hyperparameter tuning and feature selection

- **Code:**

```
# Grid Search

param_grid = {'C': np.logspace(-4, 4, 50), 'penalty':['l1', 'l2']}

clf = GridSearchCV(LogisticRegression(random_state=1),
param_grid, cv=5, verbose=0, n_jobs=-1)

best_model = clf.fit(x_train,y_train)

print(best_model.best_estimator_)

print("The mean accuracy of the model
is:",best_model.score(x_train,y_train))

#Creating the model using GridSearchCV result

Lr=linear_model.LogisticRegression( C=4714.8663634573895,
random_state= 42 )

#feature selection

select = RFE(Lr, n_features_to_select=7, step=1)

select.fit(x_train,y_train)

print("X_train.shape: {}".format(x_train.shape))

select.fit(x_train,y_train)

x_train_selected = select.transform(x_train)

x_test_selected=  select.transform(x_test)

clf = model_lr.fit(x_train_selected,y_train)

predtrain=clf.predict(x_train_selected)

predtest=clf.predict(x_test_selected)
```

- **Result:**

```
TRAINING METRICS
----------------
AUC :  0.7845
Accuracy : 0.7846
Precision :  0.7771
Recall :  0.8022
F1 :  0.7895
TEST METRICS
----------------
AUC :  0.7937
Accuracy : 0.793
Precision :  0.7699
Recall :  0.8198
F1 :  0.7941
```

❖ *Decision Tree-*

Before hyperparameter tuning

- **Code:**

```
model=tree.DecisionTreeClassifier(random_state=42
)
model=model.fit(x_train,y_train)
predtrain=model.predict(x_train)
predtest=model.predict(x_test)
```

- **Result:**

```
TRAINING METRICS
----------------
AUC :  1.0
Accuracy : 1.0
Precision :  1.0
Recall :  1.0
F1 :  1.0


TEST METRICS
----------------
AUC :  0.9356
Accuracy : 0.9351
Precision :  0.9168
Recall :  0.9532
F1 :  0.9346
```

The model is overfit, so we have to do hyperparameter tuning .

For hyperparameter tuning we are using GridSearchCV and for feature selection we are using RFE.

After hyperparameter tuning and feature selection

- **Code:**

Hyper parameter tuning using GridSearchCV

```
hyper_dist = {
        'max_depth':[5,6,7,8,9,10],
        'min_samples_split':[75,80,90,100],
        'min_samples_leaf':[35,40,45,50]
}

from sklearn.model_selection import GridSearchCV
```

```
grid = GridSearchCV(model,param_grid = hyper_dist, cv = 5, n_jobs=-1, scoring='recall')

#Creating the model using GridSearchCV result

model_DT=DecisionTreeClassifier( criterion='gini', random_state = 42,max_depth=10, min_samples_leaf=50, max_leaf_nodes= None)

#feature selection

from sklearn.feature_selection import RFE

select = RFE(estimator=model_DT, step=1, n_features_to_select = 7)

select.fit(x_train,y_train)

print("X_train.shape: {}".format(x_train.shape))

x_train_selected = select.transform(x_train)

x_test_selected=  select.transform(x_test)

clf = model_DT.fit(x_train_selected,y_train)

predtrain_Dec=clf.predict(x_train_selected)

predtest_Dec=clf.predict(x_test_selected)
```

```
TRAINING METRICS
----------------
AUC :   0.9169
Accuracy : 0.9164
Precision :   0.9785
Recall :   0.8527
F1 :   0.9113

TEST METRICS
----------------
AUC :   0.9305
Accuracy : 0.9316
Precision :   0.9686
Recall :   0.8883
F1 :   0.9267
```

❖ *Random Forest-*

Before hyperparameter tuning

- **Code:**

```
model=ensemble.RandomForestClassifier(random_st
ate= 42)
model.fit(x_train,y_train)
predtrain=model.predict(x_train)
predtest=model.predict(x_test)
```

- **Result:**

```
TRAINING METRICS
----------------
AUC :  1.0
Accuracy : 1.0
Precision :  1.0
Recall :  1.0
F1 :  1.0


TEST METRICS
----------------
AUC :  0.9728
Accuracy : 0.9728
Precision :  0.9712
Recall :  0.973
F1 :  0.9721
```

The model is overfit, so we have to do hyperparameter tuning .

For hyperparameter tuning we are using GridSearchCV and for feature selection we are using RFE.

After hyperparameter and feature selection tuning

- **Code:**

```
## Hyper parameter Tuning using GridSearchCV

hyper_dist = {
              'max_depth':[5,6,7,8,9,10],
               'min_samples_split':[75,80,90,100],
               'min_samples_leaf':[35,40,45,50],
               'n_estimators': [10,20, 40, 60, 70, 80, 90, 100]
}

from sklearn.model_selection import GridSearchCV
```

```
grid = GridSearchCV(model,param_grid = hyper_dist,
cv=5,scoring='recall',n_jobs=-1)

#Creating the model using GridSearchCV result

model_rf=ensemble.RandomForestClassifier(n_estimators=200
, criterion='gini', random_state = 42,max_depth=10,
min_samples_leaf=35)

#Feature selection

from sklearn.feature_selection import RFE

select = RFE(estimator=model_rf, step=1, n_features_to_select
= 7)

select.fit(x_train,y_train)

print("X_train.shape: {}".format(x_train.shape))

x_train_selected = select.transform(x_train)

x_test_selected=  select.transform(x_test)

clf = model_rf.fit(x_train_selected,y_train)

predtrain_RF=clf.predict(x_train_selected)

predtest_RF=clf.predict(x_test_selected)
```

- **Result:**

```
TRAINING METRICS
----------------
AUC :   0.9021
Accuracy : 0.902
Precision :   0.9129
Recall :   0.8902
F1 :   0.9014
```

```
TEST METRICS
----------------
AUC :   0.9143
Accuracy : 0.914
Precision :   0.903
Recall :   0.9225
F1 :   0.9127
```

❖ *K Nearest Neighbors-*

Before hyperparameter tuning

- **Code:**

```
from sklearn import neighbors
knn=KNeighborsClassifier()
model=knn.fit(x_train, y_train)
predtrain=model.predict(x_train)
predtest=model.predict(x_test)
```

- **Result:**

```
TRAINING METRICS
----------------
AUC :   0.8976
Accuracy : 0.8982
Precision :   0.8347
Recall :   0.9948
F1 :   0.9078

TEST METRICS
----------------
AUC :   0.8623
Accuracy : 0.8588
Precision :   0.7767
Recall :   0.9964
F1 :   0.8729
```

After hyperparameter tuning and implementing selected feature

- **Code:**

```
k_range = list(range(1, 31))

param_grid = dict(n_neighbors=k_range)

grid = GridSearchCV(knn, param_grid, cv=10,
return_train_score=False,verbose=1, n_jobs =-1)

grid=grid.fit(x_train, y_train)

model_knn=neighbors.KNeighborsClassifier(n_neighbor
s=2)

 # applying GridSEarchCV's value

clf=model_knn.fit(df_selected_fea_train, y_train)

predtrain=clf.predict(df_selected_fea_train)
```

predtest=clf.predict(df_selected_fea_test)

- **Result:**

```
TRAINING METRICS
----------------
AUC :  0.951
Accuracy : 0.9507
Precision :  1.0
Recall :  0.902
F1 :  0.9485

TEST METRICS
----------------
AUC :  0.8921
Accuracy : 0.8939
Precision :  0.9502
Recall :  0.8252
F1 :  0.8833
```

❖ *Naïve Bayes-*

Before implementing selected feature

- **Code:**

```
gnb = GaussianNB() gnb.fit(x_train,y_train)
predtrain = gnb.predict(x_train)
predtest = gnb.predict(x_test)
```

- **Result**:

```
TRAINING METRICS
----------------
AUC :  0.6539
Accuracy : 0.6546
Precision :  0.6306
Recall :  0.7573
F1 :  0.6882
TEST METRICS
----------------
AUC :  0.6207
Accuracy : 0.6175
Precision :  0.5849
Recall :  0.7387
F1 :  0.6529
```

After implementing selected feature

- **Code**:

```
model=gnb.fit(df_selected_fea_train, y_train)

predtrain=model.predict(df_selected_fea_train)
predtest=model.predict(df_selected_fea_test)
```

- **Result:**

```
TRAINING METRICS
----------------
AUC :  0.7722
Accuracy : 0.7721
Precision :  0.7821
Recall :  0.7586
F1 :  0.7702

TEST METRICS
----------------
AUC :  0.7958
Accuracy : 0.7956
Precision :  0.7825
Recall :  0.8036
F1 :  0.7929
```

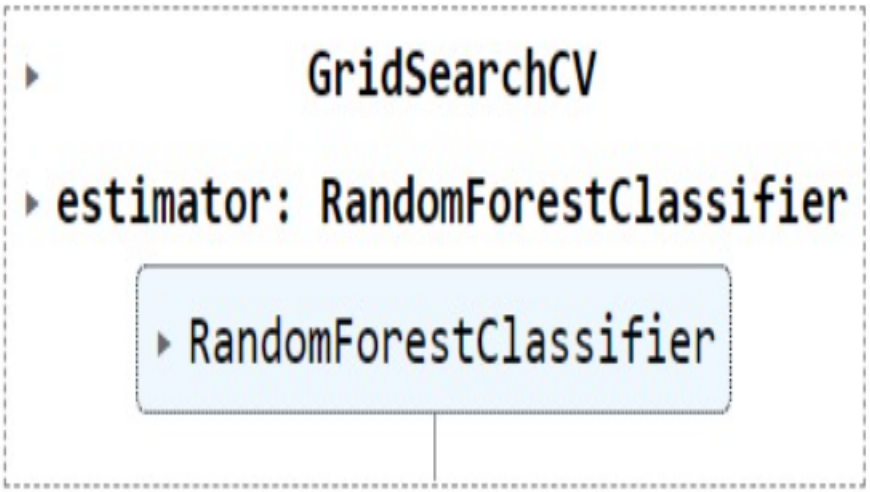
FINAL MODEL-

Random Forest is the finally selected model as it gives best recall and AUC metrics in both train and test data.

- **Code:**

```
model=ensemble.RandomForestClassifier(random_state= 42)


## Hyper parameter Tuning using GridSearchCV
hyper_dist = {
   'max_depth':[5,6,7,8,9,10],
    'min_samples_split':[75,80,90,100],
    'min_samples_leaf':[35,40,45,50],
    'n_estimators': [10,20, 40, 60, 70, 80, 90, 100]
}
grid = GridSearchCV(model,param_grid = hyper_dist, cv=5,scoring='recall',n_jobs=-1)
grid.fit(x_train,y_train)
```



grid.best_params_

```
Out[71]: {'max_depth': 10,
          'min_samples_leaf': 35,
          'min_samples_split': 100,
          'n_estimators': 100}
```

model_rf=ensemble.RandomForestClassifier(n_estimators=200, criterion='gini', random_state = 42,max_depth=10, min_samples_leaf=35)

from sklearn.feature_selection import RFE

select = RFE(estimator=model_rf, step=1, n_features_to_select = 7)

select.fit(x_train,y_train)

print("X_train.shape: {}".format(x_train.shape))

print(select)

```
RFE(estimator=RandomForestClassifier(max_depth=10, min_samples_leaf=35,
                                     n_estimators=200, random_state=42),
    n_features_to_select=7)
```

```
x_train_selected = select.transform(x_train)
x_test_selected=  select.transform(x_test)
print(np.where(select.support_ == True)[0])
```

```
[ 1  2  4  6 16 17 70]
```

```
df_selected_fea_train = x_train.iloc[:, [1,  2,  4,  6, 16, 17, 70]]
df_selected_fea_test = x_test.iloc[:, [1,  2,  4,  6, 16, 17, 70]]
clf = model_rf.fit(x_train_selected,y_train)

predtrain_RF=clf.predict(x_train_selected)
predtest_RF=clf.predict(x_test_selected)

#By selected features
print('TRAINING METRICS')
print('---------------')
printmetrics(y_train,predtrain_RF)
print("\n")
print('TEST METRICS')
print('---------------')
printmetrics(y_test,predtest_RF)
```

- **Result:**

```
TRAINING METRICS
----------------
AUC :   0.9021
Accuracy : 0.902
Precision :   0.9129
Recall :   0.8902
F1 :   0.9014

TEST METRICS
----------------
AUC :   0.9143
Accuracy : 0.914
Precision :   0.903
Recall :   0.9225
F1 :   0.9127
```

print(classification_report(y_test,predtest_RF, labels=[0,1]))

	precision	recall	f1-score	support
0	0.92	0.91	0.92	585
1	0.90	0.92	0.91	555
accuracy			0.91	1140
macro avg	0.91	0.91	0.91	1140
weighted avg	0.91	0.91	0.91	1140

cm = confusion_matrix(y_test, predtest_RF)

print('Confusion matrix\n\n', cm)

```
Confusion matrix

 [[530  55]
 [ 43 512]]
```

FUTURE SCOPE-

For more improvement we will add some more features to it so that the recall of the model will increase.

If we get more data we can run this on cloud platform for better result.

If this model is done using neural network or reinforcement learning it will work more efficiently.

We can develop a software or an app which will be useful for company members to detect the churning customers.

CERTIFICATE

This is to certify that Mr. Subhasish Mukherjee of Techno Main Saltlake,

Registration number: 20309100120084 has successfully completed a project on Customer Churn Prediction using **Machine Learning with Python** under the guidance of **Mr. Titas Roy Chowdhury**.

Mr. Titas Roy Chowdhury

Globsyn Finishing School

CERTIFICATE

This is to certify that Mr. Semanto Ghosh of Institute of Management Studies, Registration number: 201941858110007 has successfully completed a project on Customer Churn Prediction using **Machine Learning with Python** under the guidance of **Mr. Titas Roy Chowdhury**.

Mr. Titas Roy Chowdhury

Globsyn Finishing School

CERTIFICATE

This is to certify that Mr. Priyankar Basu of Institute of Management Studies,

Registration number: 201941858110005 has successfully completed a project on Customer Churn Prediction using **Machine Learning with Python** under the guidance of **Mr. Titas Roy Chowdhury**.

Mr. Titas Roy Chowdhury

Globsyn Finishing School

CERTIFICATE

This is to certify that Mr. Sankalpa Das of Institute of Management Studies,

Registration number: 201941858110010 has successfully completed a project on Customer Churn Prediction using **Machine Learning with Python** under the guidance of **Mr. Titas Roy Chowdhury**.

Mr. Titas Roy Chowdhury

Globsyn Finishing School

CERTIFICATE

This is to certify that Ms. Sumaiya Shakil of Institute of Management Studies, Registration number: 201941858110011 has successfully completed a project on Customer Churn Prediction using **Machine Learning with Python** under the guidance of **Mr. Titas Roy Chowdhury**.

Mr. Titas Roy Chowdhury

Globsyn Finishing School