Capstone Project

Supervised ML Regression

Bike Sharing Demand Prediction

Team member

Yashwant Raul

Mayur Marathe

Sanket Gawali

## Project Overview

This project discusses the prediction model of Bank Marketing Effectiveness of a Portuguese Marketing institution. The marketing campaigns were based on phone calls. The classification goal is to predict if the client will subscribe to a term deposit.

First I explore the data, cleaned and preprocessed the data and then I performed the exploratory data analysis to extract information, in which we identified certain trends, relationships, correlation and found out the features that had some impact on our dependent variable and plotted the graph to visualize the impact on dependent variable. I also encoded the categorical variables.

I build the various machine learning algorithms on our split and standardized data. I tried different algorithms namely; Logistic Regression, Random Forest Classifier, Decision Tree Classifier, Gradient Boosting Classifier, K Neighbors Classifier, XG Boost and Naive Bayes Classifier. I did hyper parameter tuning and evaluated the performance of the model.

I analyze the data and build the model by considering the below

* **Problem Statement**

### The data is related to direct marketing campaigns (phone calls) of a Portuguese banking institution. The marketing campaigns were based on phone calls. Often, more than one contact to the same client was required, in order to assess if the product (bank term deposit) would be ('yes') or not ('no') subscribed. The classification goal is to predict if the client will subscribe to a term deposit (variable y).

* **Data Description**

## Data Description

## Input variables:

### Bank Client data:

* age (numeric)
* job : type of job (categorical: 'admin.','blue-collar','entrepreneur','housemaid','management','retired','self-employed','services','student','technician','unemployed','unknown')
* marital : marital status (categorical: 'divorced','married','single','unknown'; note: 'divorced' means divorced or widowed)
* education (categorical: 'basic.4y','basic.6y','basic.9y','high.school','illiterate','professional.course','university.degree','unknown')
* default: has credit in default? (categorical: 'no','yes','unknown')
* housing: has a housing loan? (categorical: 'no','yes','unknown')
* loan: has a personal loan? (categorical: 'no','yes','unknown')

### Related with the last contact of the current campaign:

* contact: contact communication type (categorical: 'cellular','telephone')
* month: last contact month of year (categorical: 'jan', 'feb', 'mar', ..., 'nov', 'dec')
* day\_of\_week: last contact day of the week (categorical: 'mon','tue','wed','thu','fri')
* duration: last contact duration, in seconds (numeric). Important note: this attribute highly affects the output target (e.g., if duration=0 then y='no'). Yet, the duration is not known before a call is performed. Also, after the end of the call y is obviously known. Thus, this input should only be included for benchmark purposes and should be discarded if the intention is to have a realistic predictive model.

### Other attributes:

* campaign: number of contacts performed during this campaign and for this client (numeric, includes last contact)
* pdays: number of days that passed by after the client was last contacted from a previous campaign (numeric; 999 means client was not previously contacted)
* previous: number of contacts performed before this campaign and for this client (numeric)
* poutcome: outcome of the previous marketing campaign (categorical: 'failure','nonexistent','success')

### Output variable (desired target):

* y - has the client subscribed a term deposit? (binary: 'yes','no')
* **Steps Involved**

**Data Wrangling**

After loading the dataset, I performed this method by cleaning, organizing, and transforming raw data into the desired format which makes us to understand the data clearly. This process helped us to tackle the unwanted data, to produce accurate results, to make better decision.

**Exploratory Data Analysis**

After Data wrangling, we performed EDA. Analysis to get hidden insights of data. This process helped us figuring out various aspects and relationships among the target and the independent variables. It gave us a better idea of which feature behaves in which manner compared to the target variable.

The goal of EDA is to leverage visualization tools, summary tables, and hypothesis testing to:

* Provide summary level insight into a dataset.
* Uncover underlying patterns and structures from data.
* Identify outliers, missing data, class balance, and other data-related issues.

**Continuous and categorical Features Analysis**

With the help of exploratory data analysis we analyzed the categorical as well as numerical features in the dataset.

**Analysis of Dependent Variable**

I analyze our dependent variable, A dependent variable is a variable whose value will change depending on the value of another variable.

**Feature Engineering**

In this dataset some categorical variables, we convert it into a numerical database.by label encoding and scaling the data. also solve the class imbalance before the training model.

**Correlation Analysis**

I plot the heat map to find the correlation between both dependent variables and independent variables.



* **Model Training**

Train test Split

Before fitting any model it is a rule of thumb to split the dataset into a training and test set. The proportions but mostly used is 80:20 for training and testing respectively.

* **Model Building**

. Models

We uses 6 model to train the data and for predicting the accuracy

1. **Logistic Regression**
2. **Gradient Boosting Classifier**
3. **XG Boost**
4. **K Neighbors Classifier**
5. **Decision Tree Classifier**
6. **Random Forest Classifier**
7. **Naive Bayes Classifier.**

**1 Algorithms**

1. **Logistic regression**

Logistic regression is a supervised learning classification algorithm used to predict the probability of a target variable. The nature of target or dependent variable is dichotomous, which means there would be only two possible classes.

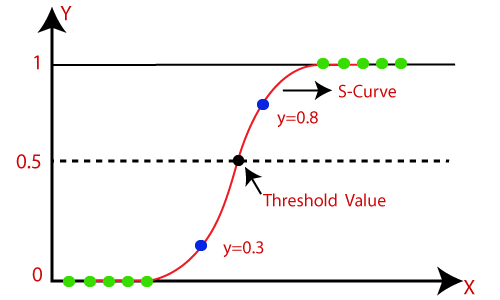
In simple words, the dependent variable is binary in nature having data coded as either 1 (stands for success/yes) or 0 (stands for failure/no).

Mathematically, a logistic regression model predicts P(Y=1) as a function of X. It is one of the simplest ML algorithms that can be used for various classification problems such as spam detection, Diabetes prediction, cancer detection etc.

## Logistic Regression Assumptions

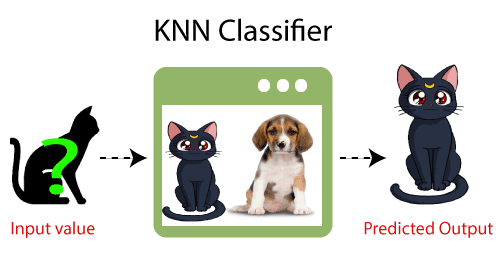
Before diving into the implementation of logistic regression, we must be aware of the following assumptions about the same −

* In case of binary logistic regression, the target variables must be binary always and the desired outcome is represented by the factor level 1.
* There should not be any multi-collinearity in the model, which means the independent variables must be independent of each other.
* We must include meaningful variables in our model.
* We should choose a large sample size for logistic regression.



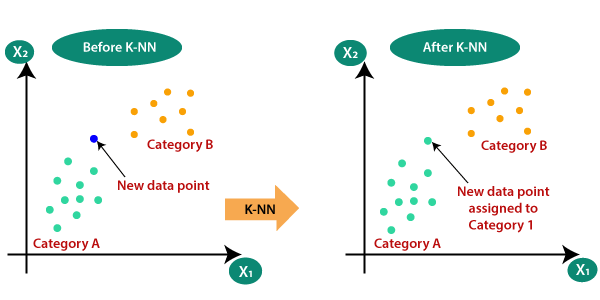
**K nearest neighbor classification**

* K-Nearest Neighbor is one of the simplest Machine Learning algorithms based on Supervised Learning technique.
* The K-NN algorithm assumes the similarity between the new case/data and available cases and puts 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 suited category by using K- NN algorithm.
* The 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.
* The 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 whether 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



## Why do we need a K-NN Algorithm?

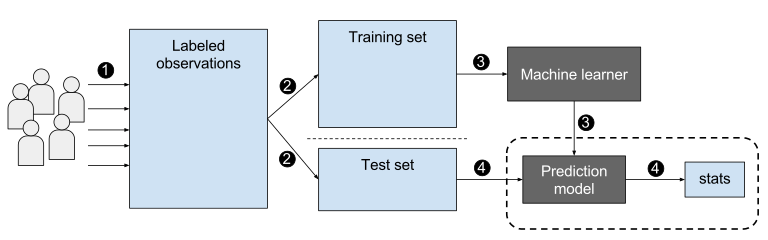
Suppose there are two categories, i.e., Category A and Category B, and we have a new data point x1, so this data point will lie in which of these categories. To solve this type of problem, we need a K-NN algorithm. With the help of K-NN, we can easily identify the category or class of a particular dataset. Consider the below diagram:



* **Gradient boosting classifier**

[Gradient boosting classifiers](https://en.wikipedia.org/wiki/Gradient_boosting) 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 models are becoming popular because of their effectiveness at classifying complex datasets,

### Steps to Gradient Boosting

In order to implement a gradient boosting classifier, we'll need to carry out a number of different steps. We'll need to:

* Fit the model
* Tune the model's parameters and Hyper parameters
* Make predictions
* Interpret the results

Fitting models with Scikit-Learn is fairly easy, as we typically just have to call the fit() command after setting up the model.

However, tuning the model's hyper parameters requires some active decision making on our part. There are various arguments hyper parameters we can tune to try and get the best accuracy for the model. One of the ways we can do this is by altering the learning rate of the model. We'll want to check the performance of the model on the training set at different learning rates, and then use the best learning rate to make predictions.

Predictions can be made in Scikit-Learn very simply by using the predict () function after fitting the classifier. You'll want to predict on the features of the testing dataset, and then compare the predictions to the actual labels. The process of evaluating a classifier typically involves checking the accuracy of the classifier and then tweaking the parameters hyper parameters of the model until the classifier has an accuracy that the user is satisfied with.

* **Decision tree classifier**

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.

The decisions or the test are performed on the basis of features of the given dataset.

It is a graphical representation for getting all the possible solutions to a problem/decision based on given conditions.

It is called a decision tree because, similar to a tree, it starts with the root node, which expands on further branches and constructs a tree-like structure.

In order to build a tree, we use the CART algorithm, which stands for Classification and Regression Tree algorithm.

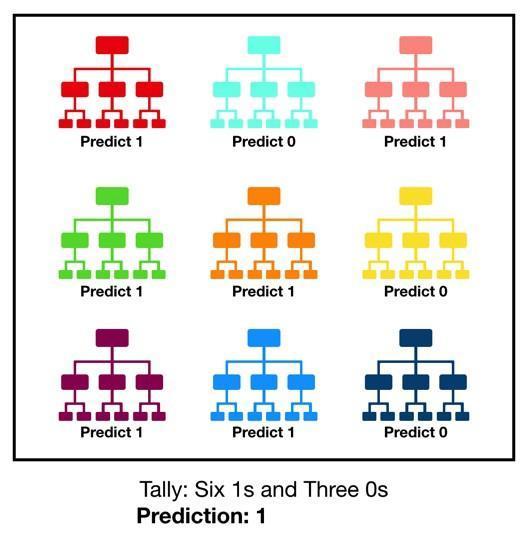
A decision tree simply asks a question, and based on the answer (Yes/No), it further splits the tree into sub trees.

Below diagram explains the general structure of a decision tree:



* **Random forest classifier**

Random forest, like its name implies, consists of a large number of individual decision trees that operate as an [ensemble](https://en.wikipedia.org/wiki/Ensemble_learning). Each individual tree in the random forest spits out a class prediction and the class with the most votes becomes our model’s prediction (see figure below).



* **Naive Bayes classifier**

The Naïve Bayes algorithm is a supervised learning algorithm, which is based on Bayes theorem and used for solving classification problems.

The Naïve Bayes Classifier is one of the simplest and most effective Classification algorithms which helps in building the fast machine learning models that can make quick predictions.

It is a probabilistic classifier, which means it predicts on the basis of the probability of an object.

Bayes' Theorem:

Bayes' theorem is also known as Bayes' Rule or Bayes' law, which is used to determine the probability of a hypothesis with prior knowledge. It depends on the conditional probability.

The formula for Bayes' theorem is given as:

Naïve Bayes Classifier Algorithm

Where,

P(A|B) is Posterior probability: Probability of hypothesis A on the observed event B.

P(B|A) is Likelihood probability: Probability of the evidence given that the probability of a hypothesis is true.

P(A) is Prior Probability: Probability of hypothesis before observing the evidence.

P(B) is Marginal Probability: Probability of Evidence.

Stratified Sampling:

Stratified random sampling is a type of probability sampling using which a [research](https://www.questionpro.com/market-research.html) organization can branch off the entire [population](https://www.questionpro.com/blog/population-data/) into multiple non-overlapping, homogeneous groups (strata) and randomly choose final members from the various strata for research which reduces cost and improves efficiency.

Stratified Sampling: Is an important concept that is often missed when developing a model either for regression or classification. Remember, that in order to avoid over fitting of our data we must implement a cross validation; however, we must make sure that at least the features that have the greatest influence on our label (whether a potential client will open a term deposit or not) is equally distributed. What do I mean by this?

Personal Loans:

For instance, having a personal loan is an important feature that determines whether a potential client will open a term deposit or not. To confirm it has a heavy weight on the final output you can check the correlation matrix above and you can see it has a -11% correlation with opening a deposit. What steps should we take before implementing stratified sampling in our train and test data?

1) We need to see how our data is distributed.

2) After noticing that the column of loan contains 87% of "no" (Does not have personal loans) and 13% of "yes" (Have personal loans.)

3) We want to make sure that our training and test set contains the same ratio of 87% "no" and 13% "yes"." Stratified Sampling: Is an important concept that is often missed when developing a model either for regression or classification. Remember, that in order to avoid overfitting of our data we must implement a cross validation; however, we must make sure that at least the features that have the greatest influence on our label (whether a potential client will open a term deposit or not) is equally distributed.

# Handling imbalance target variable using SMOTE

#### SMOTE: Synthetic Minority Oversampling Technique

SMOTE is an oversampling technique where the synthetic samples are generated for the minority class. This algorithm helps to overcome the over fitting problem posed by random oversampling. It focuses on the feature space to generate new instances with the help of interpolation between the positive instances that lie together.

SMOTE first selects a minority class instance at random and finds its k nearest minority class neighbors. The synthetic instance is then created by choosing one of the k nearest neighbors b at random and connecting a and b to form a line segment in the feature space. The synthetic instances are generated as a convex combination of the two chosen instances a and b.

Python code for SMOTE Algorithm :-

print("Before Oversampling, counts of label '1': {}".format(sum(y\_train == 1)))

print("Before Oversampling, counts of label '0': {} \n".format(sum(y\_train == 0)))

# import SMOTE module from imblearn library

# pip install imblearn (if you don't have imblearn in your system)

from imblearn.over\_sampling import SMOTE

sm = SMOTE(random\_state = 2)

x\_train\_smote, y\_train\_smote = sm.fit\_resample(x\_train, y\_train.ravel())

print('After OverSampling, the shape of train\_X: {}'.format(x\_train.shape))

print('After OverSampling, the shape of train\_y: {} \n'.format(y\_train.shape))

print("After OverSampling, counts of label '1': {}".format(sum(y\_train == 1)))

print("After OverSampling, counts of label '0': {}".format(sum(y\_train == 0)))

Though this algorithm is quite useful, it has few drawbacks associated with it.

i) The synthetic instances generated are in the same direction i.e. connected by an artificial line to its diagonal instances. This in turn complicates the decision surface generated by few classifier algorithms.

ii) SMOTE tends to create a large no. of noisy data points in feature space.

Among those is chosen. When max\_features is set 1, this amounts to building a totally random decision tree

### Avoiding Over fitting:

**Brief Description of Over fitting?**

This is an error in the modeling algorithm that takes into consideration random noise in the fitting process rather than the pattern itself. You can see that this occurs when the model gets an awesome score in the training set but when we use the test set (Unknown data for the model) we get an awful score. This is likely to happen because of overfitting of the data (taking into consideration random noise in our pattern). What we want our model to do is to take the overall pattern of the data in order to correctly classify whether a potential client will subscribe to a term deposit or not. In the examples above, it is most likely that the Decision Tree Classifier and Random Forest classifiers are overfitting since they both give us nearly perfect scores (100% and 99%) accuracy scores.

**How can we avoid Overfitting?**

The best alternative to avoid overfitting is to use cross validation. Taking the training test and splitting it. For instance, if we split it by 3, 2/3 of the data or 66% will be used for training and 1/3 33% will be used for testing and we will do the testing process three times. This algorithm will iterate through all the training and test sets and the main purpose of this is to grab the overall pattern of the data.

**Hyperparameter Tuning:**

We are using Gradient Boosting Classifiers as our final model.

The overall parameters of the Gradient Boosting Model can be divided into three categories:

1. *Tree-Specific Parameters*: These affect each individual tree in the model.
2. *Boosting Parameters*: These affect the boosting operation in the model.
3. *Miscellaneous Parameters*: Other parameters for overall functioning.

**Model Evaluation matrix:-**

It is necessary to obtain the accuracy on training data, but it is also important to get a genuine and approximate result on unseen data, otherwise the Model is of no use.

So to build and deploy a generalized model we require to evaluate the model on different metrics which helps us to better optimize the performance, fine-tune it, and obtain a better result.

If one metric is perfect, there is no need for multiple metrics. To understand the benefits and disadvantages of Evaluation metrics because different evaluation metrics fit on a different set of a dataset.

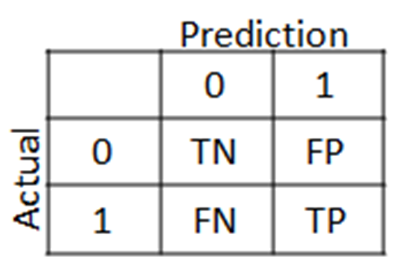
Let's start exploring various Evaluation metrics.

## Confusion Matrix:-

## Confusion Matrix is a performance measurement for the machine learning classification problems where the output can be two or more classes. It is a table with combinations of predicted and actual values.

A confusion matrix is defined as the table that is often used to describe the performance of a classification model on a set of the test data for which the true values are known.

The main purpose of a confusion matrix is to see how our model is performing when it comes to classifying potential clients that are likely to subscribe to a term deposit. We will see in the confusion matrix four terms the True Positives, False Positives, True Negatives and False Negatives.



## 

## Positive/Negative: Type of Class (label) ["No", "Yes"] True/False: Correctly or incorrectly classified by the model.

## True Negatives (Top-Left Square): This is the number of correct classifications of the "No" class or potential clients that are not willing to subscribe to a term deposit.

## False Negatives (Top-Right Square): This is the number of incorrect classifications of the "No" class or potential clients that are not willing to subscribe to a term deposit.

## False Positives (Bottom-Left Square): This is the number of incorrect classifications of the "Yes" class or potential clients that are willing to subscribe to a term deposit.

## True Positives (Bottom-Right Square): This is the number of correct classifications of the "Yes" class or potential clients that are willing to subscribe to a term deposit.

## 

Fig:- Confusion matrix before smote on training dataset

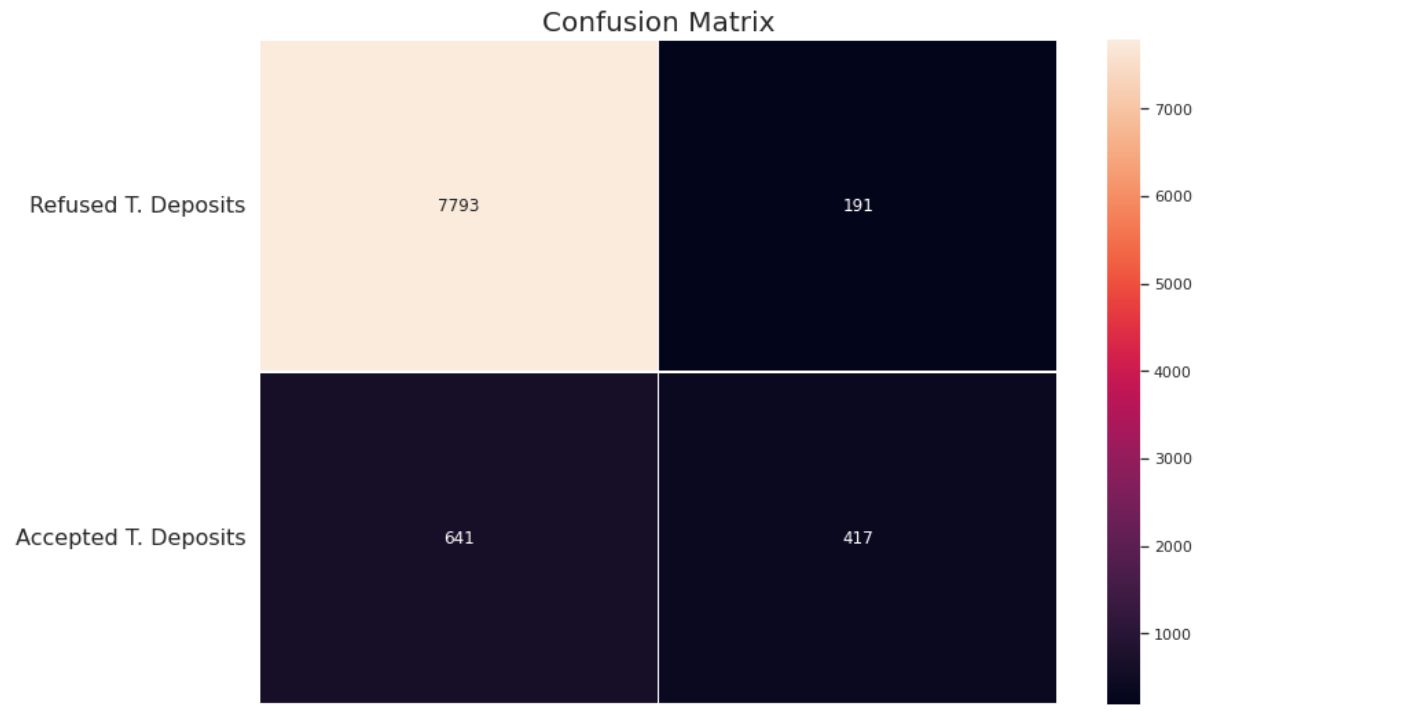


Fig:-Confusion matrix before smote on testing dataset

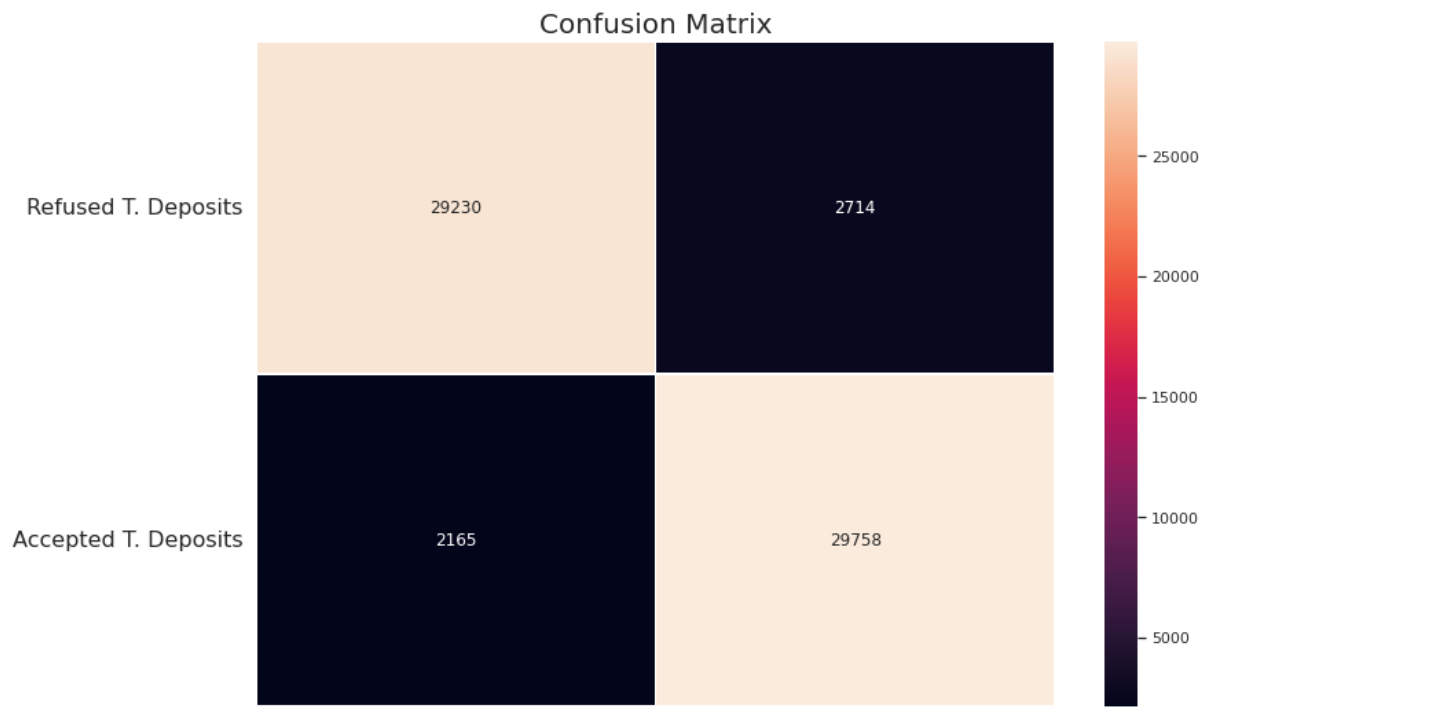


Fig:- Confusion matrix after SMOTE on training dataset

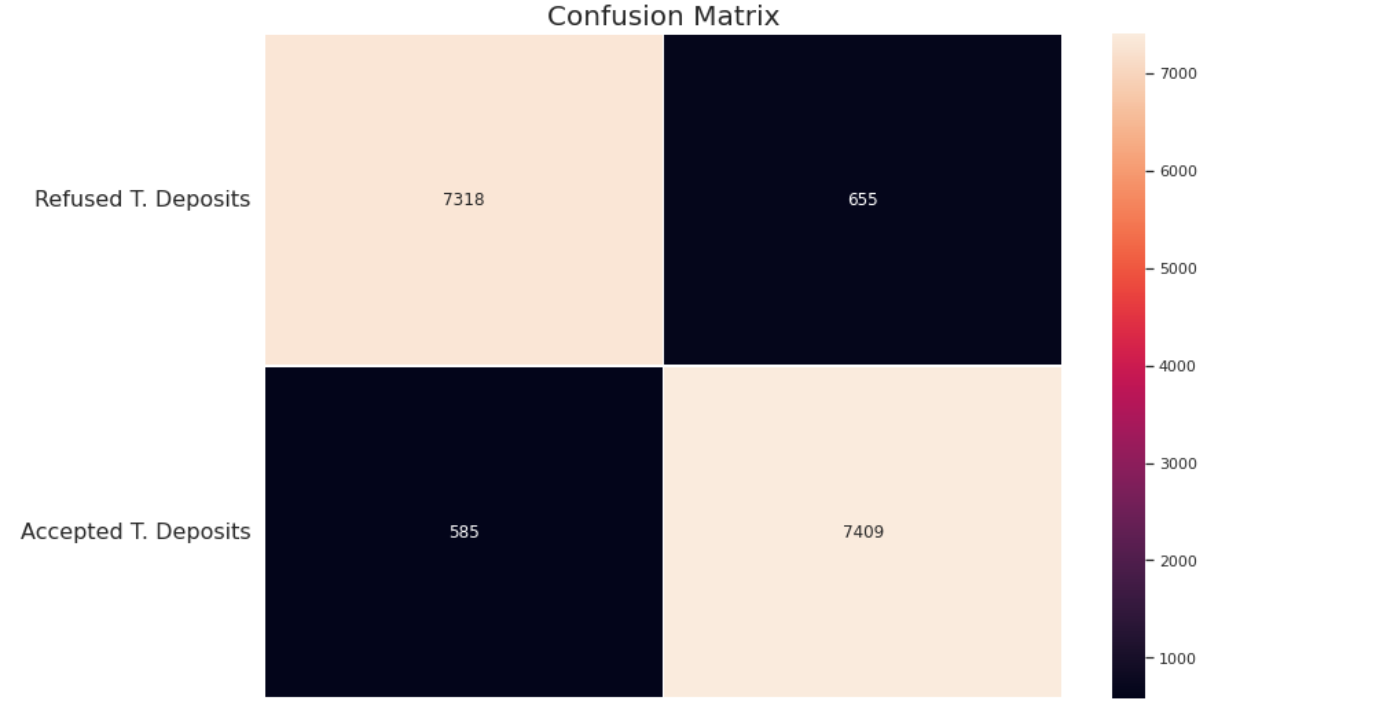
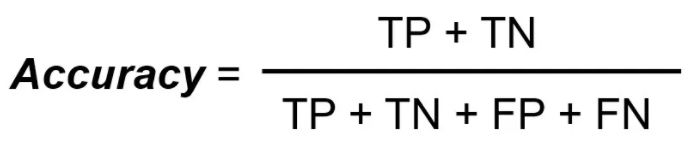


Fig:- Confusion matrix after SMOTE on testing dataset

## Accuracy:-

Accuracy simply measures how often the classifier correctly predicts. We can define accuracy as the ratio of the number of correct predictions and the total number of predictions.

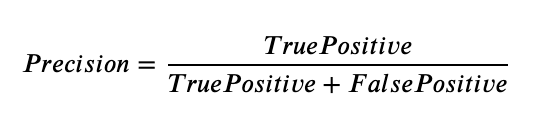


Accuracy is useful when the target class is well balanced but is not a good choice for the unbalanced classes.

Precision:-

Precision explains how many of the correctly predicted cases actually turned out to be positive. Precision is useful in the cases where False Positive is a higher concern than False Negatives. The importance of *Precision is in music or video recommendation systems, e-commerce websites, etc. where wrong results could lead to customer churn and this could be harmful to the business.*

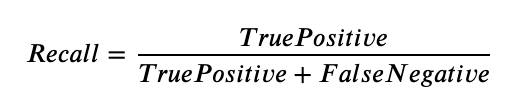
Precision for a label is defined as the number of true positives divided by the number of predicted positives.



Recall (Sensitivity) :-

Recall explains how many of the actual positive cases we were able to predict correctly with our model. It is a useful metric in cases where False Negative is of higher concern than False Positive. It is important in medical cases where it doesn’t matter whether we raise a false alarm but the actual positive cases should not go undetected!

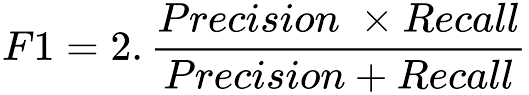
Recall for a label is defined as the number of true positives divided by the total number of actual positives.

****

**F1 Score:-**

 It gives a combined idea about Precision and Recall metrics. It is maximum when Precision is equal to Recall.

F1 Score is the harmonic mean of precision and recall.



The F1 score punishes extreme values more. F1 Score could be an effective evaluation metric in the following cases:

* When FP and FN are equally costly.
* Adding more data doesn’t effectively change the outcome
* True Negative is high

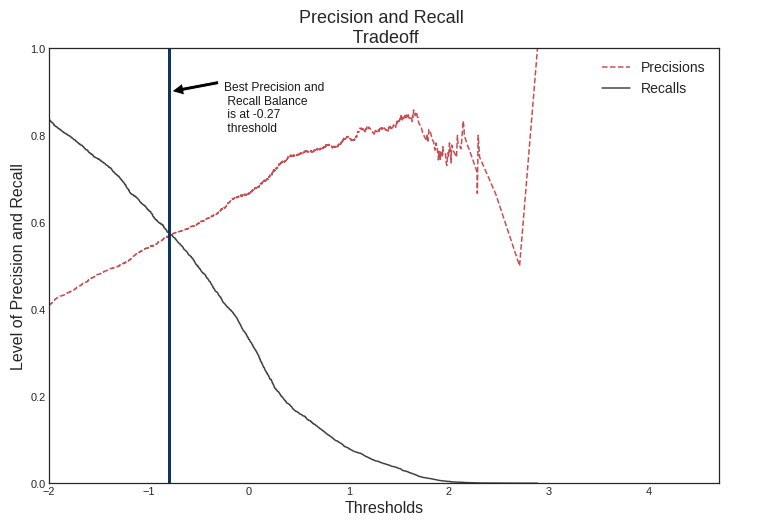


Fig:- Precision And Recall Tradeoff

**AUC-ROC: -** The Receiver Operator Characteristic (ROC) is a probability curve that plots the TPR(True Positive Rate) against the FPR(False Positive Rate) at various threshold values and separates the ‘signal’ from the ‘noise’.

The Area Under the Curve (AUC) is the measure of the ability of a classifier to distinguish between classes.

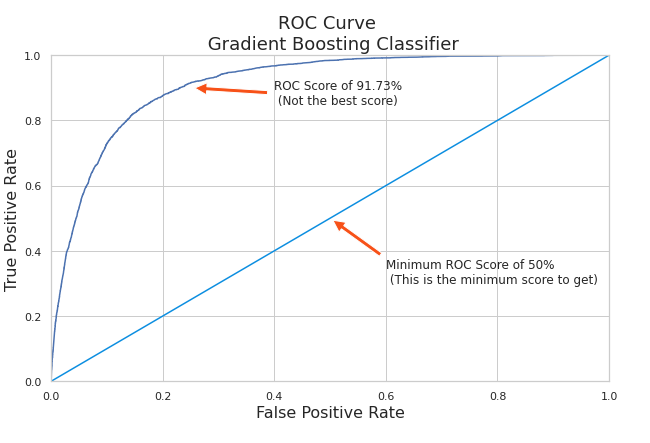


Fig: - ROC Curve (Gradient Boosting Classifier)

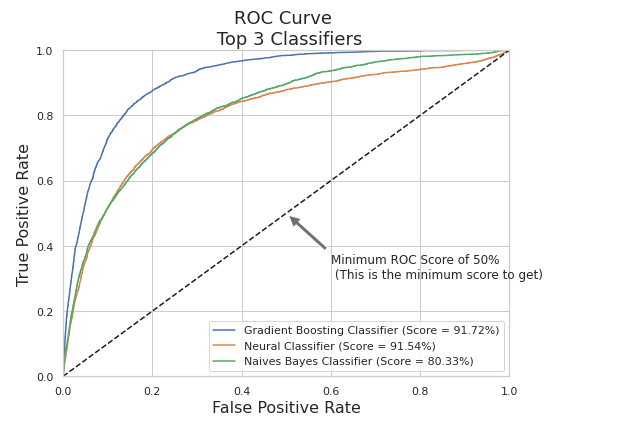


Fig: - ROC Curve (Top 3 Classifiers)

**Conclusion**

* Blue-collar, management and technician showed maximum interest in subscription.
* Divorce people have no interest in term deposits.
* People with secondary and tertiary education were more driven towards paying term deposits in banks.
* Generally people who don't have credit in default are interested in a deposit. Majority of the people have a home loan but only a few of them opted for a term deposit.
* Cellular communication is more effective in comparison to other communication types.
* There were maximum subscriptions in the summer season.
* The calls with large duration have more tendency for conversion. People were mostly contacted only once.
* Majority of people were not contacted previously before this campaign and there are no significant contacts after 11 times already done.
* Success rate is high for unknown poutcome.
* We can choose our model either **Gradient Boosting Classifier, Random Forest Classifier and XG boost** to predict Effectiveness as they both are showing maximum accuracy

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Model** | **Logistic Regression** | **Random Forest Classifier** | **Decision Tree Classifier** | **Gradient Boosting Classifier** | **K Neighbor**  **Classifier** | **XGBoost** | **Naïve Bayes Classifier** |
| **Accuracy** | **0.87** | **0.89** | **0.85** | **0.88** | **0.87** | **0.88** | **0.80** |