Improving Effectiveness of Direct Marketing campaigns through Predictive Modelling

Location: **Bangalore** *Group*: **5** *Batch:* **PGPDSE-FT Apr 21**

Submitted by:

Deep Ranjan Guha

Piyali Dey

Ronjini Konwar

Ankit Chatterjee

Harshal Chauhan

Sudipto Das

Under the Esteemed Guidance of

Anjana Agarwal

Table of Contents

- **Refitting the Tunned base models to compare the performances**
- **Checking the presence of Imbalance in the target feature(y)**
- **SMOTE ANALYSIS**
- **Refitting Logistic Regression, Decision tree, Random Forest algorithm on Resampled data.**
- **Feature Extraction**
- **Refitting the Logistic Regression model, Decision Tree model and Random Forest model on the new dataset containing features obtained after recursive elimination**
- **Boosting Algorithms on the selected resampled features**
- **Boosting Algorithms on full Resampled data**

- **Model Comparison for Final Model selection**
- **Final Predictive Model**
- **Conclusion**
- **Limitations and Future Prospective**
- **References**
- **Acknowledgement**

Industry Review

Direct marketing in the sector of banking, aims to establish a cost-effective way of marketing. To achieve efficient direct marketing, possessing information about the present and estimating future customer's preferences always provides an edge over the competitors. The availability of prospective customer data combined with improvements in data analysis has prompted firms to develop more customer-oriented strategies in recent decades to remain competitive.

Direct Marketing campaigns can have high failure rate if the target customers are not chosen properly. This project can help such Data Marketing campaigns to choose their target customer well so that time and resources can be saved for the banking institution. From the customer's point of view a smaller number of non-interested target audience will get disturbed by the Direct Marketing campaign.

Transactions in banks are recorded on day-to-day basis and customer-relationship is very important these days. Customers are made aware of new products or services offered by the bank.

Telemarketing, which is a very popular form of direct marketing, targets customers from a database of prospective customers. To ensure the success of techniques such as this, the company must focus on their potential customer database by predicting the customers who have higher chances of using the product or service.

Many banks have been using data mining techniques to predict the class of customers who are most likely to buy or use the products or services.

Nowadays, commercial banks implement telemarketing campaign to optimize the allocation of resources, satisfy the needs of customers, thus, enhancing the productivity of companies. Through a marketing campaign, contacting clients on telephone directly, the bank intends to select the best set of clients. It is beneficial for narrowing the range of potential customers, elevating the rate of success as well as reducing the cost of the marketing process efficiently.

Literature Review:

Direct Marketing:

Direct Marketing has started to show better results as it is cost effective as well as allows the organisation to interact with customers and they can have the idea of who their prospective

customers are apart from their current ones (Elsalamony, 2013). Telemarketing is one of the various types of Direct Marketing which can be used to create awareness to new and current customers about their products and services company (Bencin, 1992). The ability to identify prospective customers is one of the means of ensuring the success of telemarketing (McCausland,2000).

Portuguese Bank Marketing Dataset and reference work:

The dataset at that we are working on contains the information collected during direct marketing campaigns carried out by a Portuguese bank. Their goal was to get their customers to subscribe to a term deposit. Term deposit means a deposit held by a bank for fixed period of time.

The data was made available by Moro, Laureano and Cortez (2014) through the popular University of California at Irvine (UCI) machine learning repository. The dataset also contains record of the customer's background, socio-economic attributes and the campaign details (Moro, Laureano and Rita, 2014).

With respect to marketing campaign, to improve the success of telemarketing, data mining helps to construct various models to solve the problems about bank direct marketing. Moro et al. proposed a data mining approach to analyse the probability of success. For the sake of extracting the key information, feature selection was emphasized by employing NN (Moro, Laureano and Rita, 2014).

Imbalanced Data:

In real world tasks, the data is usually imbalanced. This happens because a class is always more predominant than the other. It cannot always be fully controlled as outliers, noise, etc. are present due to errors and deficiencies while generating the data (Sowah et al., 2016). The imbalance problem leads to the reduction of the generalization of the results generated from machine learning algorithms (Kim,2007).

Relevance of Data-Mining in this context:

There has been a high level of focus on customer retention and segmentation these days (Witten and Frank, 2005). Thus, in order to keep doing well in this highly competitive business world, the use of Data mining techniques to drive business growth is required. (Moro, Laureano and Cortez, 2011)

Objectives

- The ability to identify customers who has a better likelihood of subscribing to a term deposit is important because targeted marketing campaigns should be aimed at potential customers in order to help reduce time and resources spent on such campaigns.
- The purpose of this project is to predict the success of bank telemarketing to select the best consumer set.

To achieve efficient direct marketing strategy, possessing information about the present and estimating future customer preferences is a fundamental requirement. This data is commonly used to create, maintain relationships and direct connections with customers to target them individually in the sale of certain products or banking offerings. We aim to analyse this data and use Data Mining approaches to build a predictive model.

Dataset

Bank Marketing data is obtained from the UCI Machine Learning Repository. There are 41188 records and there are 20 features and 1 class (y) which contains the information of -if the client has subscribed to the term deposit or not.

The dataset contains:

bank client data:

1. age (numeric): age of the client
2. job: type of job (categorical: 'admin.','blue-collar','entrepreneur','housemaid','management','retired','self-employed','services','student','technician','unemployed','unknown')
3. marital : marital status (categorical: 'divorced', 'married', 'single', 'unknown'; note: 'divorced' means divorced or widowed)
4. education(categorical: 'basic.4y','basic.6y','basic.9y','high.school','illiterate','professional.course','university.degree','unknown')

5. default: has credit in default? (categorical: 'no', 'yes', 'unknown')
6. housing: has housing loan? (categorical: 'no', 'yes', 'unknown')
7. loan: has personal loan? (categorical: 'no',' yes',' unknown')

related with the last contact of the current campaign:

8. contact: contact communication type (categorical: 'cellular', 'telephone')
9. month: last contact month of year (categorical: 'jan' to ‘dec’)
10. day_of_week: last contact day of the week (categorical: 'mon','tue','wed','thu','fri')
11. duration: last contact duration, in seconds (numeric).

other attributes:

12. campaign: number of contacts performed during this campaign and for this client (numeric, includes last contact)
13. 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)
14. previous: number of contacts performed before this campaign and for this client (numeric)
15. poutcome: outcome of the previous marketing campaign (categorical: 'failure','nonexistent','success')

social and economic context attributes

16. emp.var.rate: employment variation rate : quarterly indicator (numeric)
17. cons.price.idx: consumer price index : monthly indicator (numeric)

18. cons.conf.idx: consumer confidence index : monthly indicator (numeric)
19. euribor3m: euribor 3 month rate : daily indicator (numeric)
20. nr.employed: number of employees : quarterly indicator (numeric)

Output variable (desired target):

21. y: has the client subscribed a term deposit? ‘yes’ or ‘no’

- The dataset is semi-colon separated (;). We converted it into comma separated values (csv).
- The output variable named "y" is string which has values either "yes" or "no". Thus we replaced it with binary values: 1 for ‘yes’ and 0 for ‘no’.

The number of **categorical columns** is 11, they are : 'job', 'marital', 'education', 'default', 'housing', 'loan', 'contact', 'month', 'day_of_week', 'poutcome', 'y'.

The number of **numerical columns** is 10, they are : 'age', 'duration', 'campaign', 'pdays', 'previous', 'emp.var.rate', 'cons.price.idx', 'cons.conf.idx', 'euribor3m', 'nr.employed'.

EXPLORATORY DATA ANALYSIS:

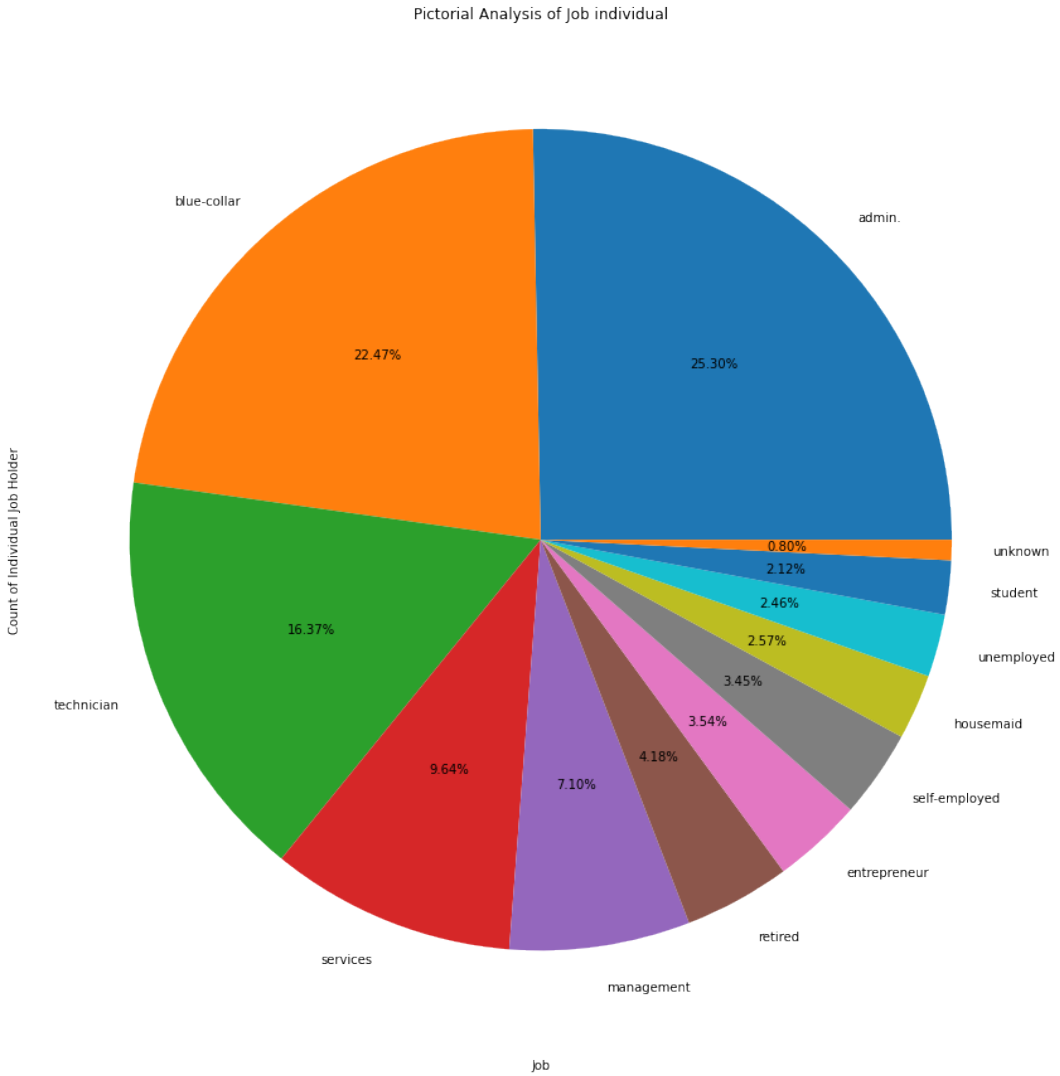
Univariate Analysis

Categorical

The percentage values in each categorical columns in the dataset:

Variable: Job

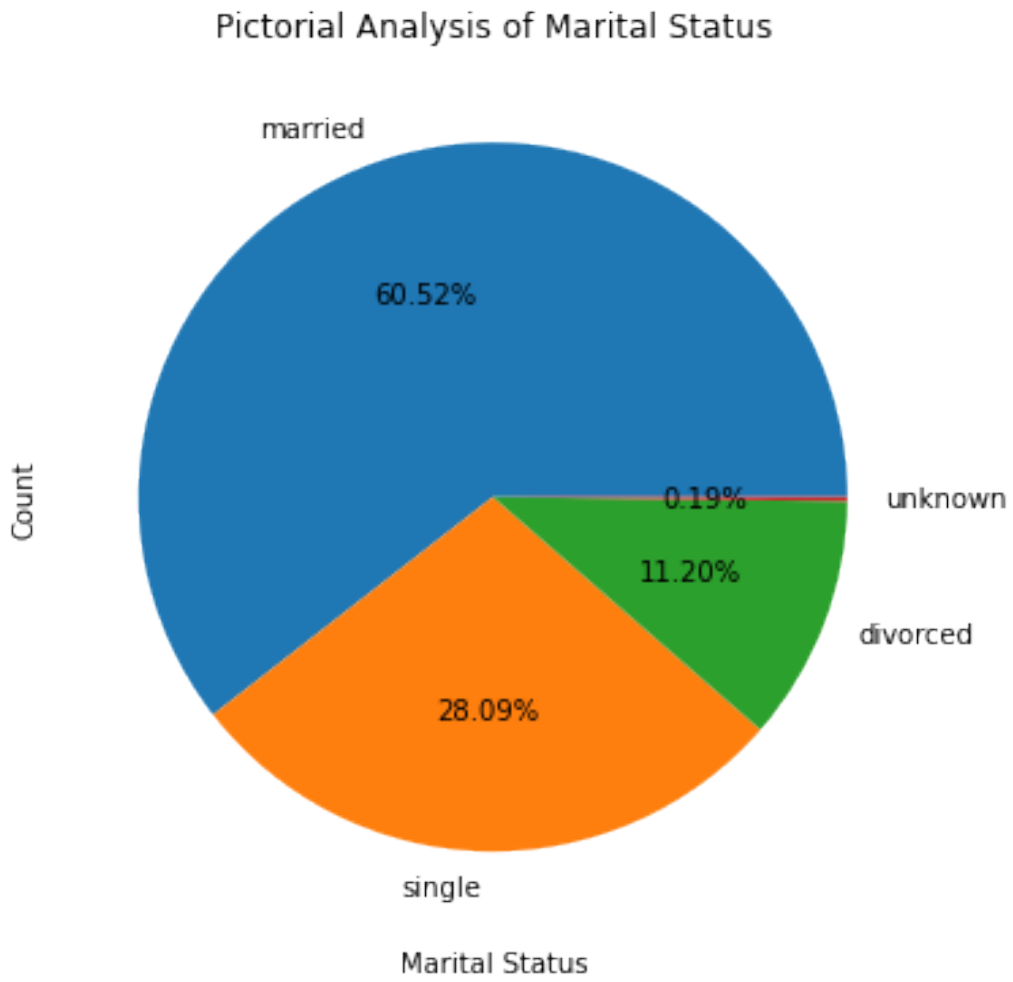
admin.	25.303486
blue-collar	22.467709
technician	16.371273
services	9.636302
management	7.099155
retired	4.175974
entrepreneur	3.535010
self-employed	3.450034
housemaid	2.573565
unemployed	2.461882
student	2.124405
unknown	0.801204



Most number of customers are in the admin job, followed by blue-collar and technician. Unknown segment is less client followed by students and uemployed.

Variable: Marital

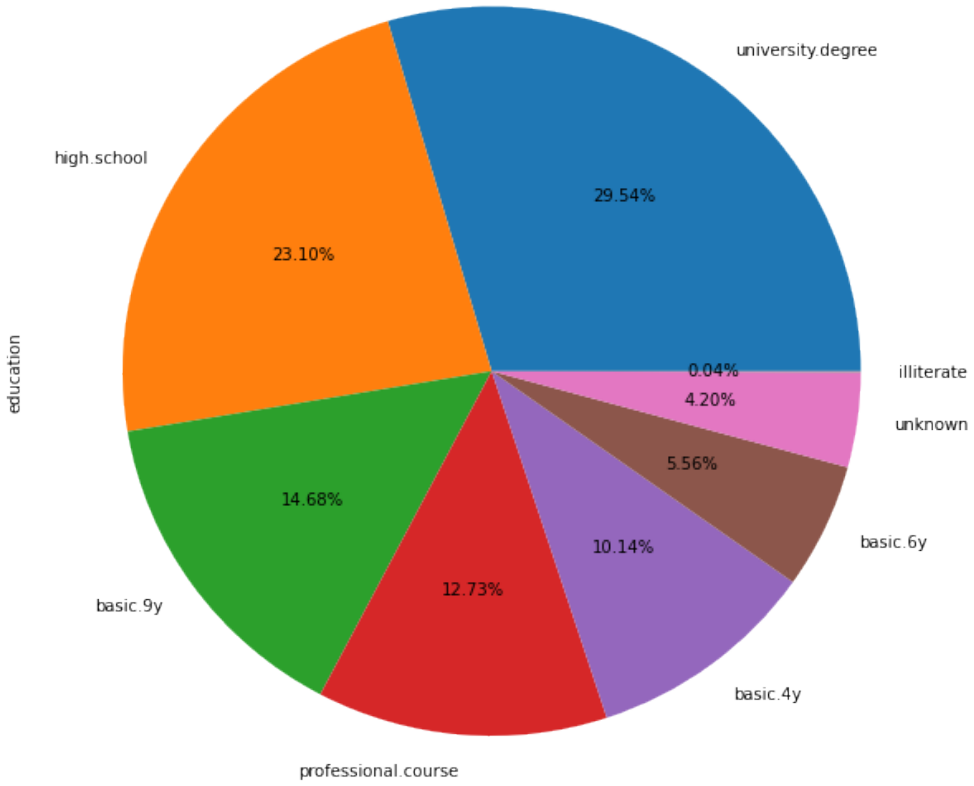
married	60.522482
single	28.085850
divorced	11.197436
unknown	0.194231



Most number of customers are Married. Followed by single and unknown is very less.

Variable: Education

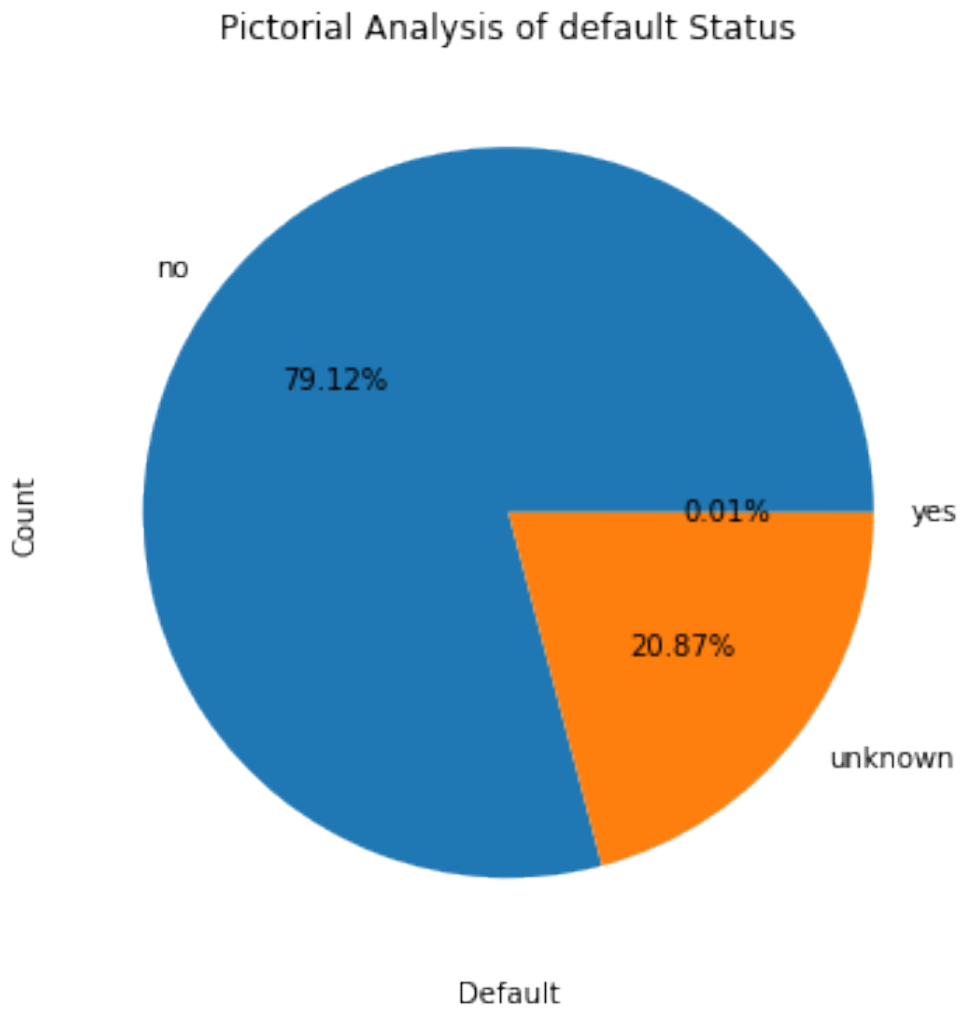
university.degree	29.542585
high.school	23.101389
basic.9y	14.676605
professional.course	12.729436
basic.4y	10.138875
basic.6y	5.564728
unknown	4.202680
illiterate	0.043702



Most number of customers have a university degree, followed by high school and basic 9y. Illiterate clients are very less followed by unknown

Variable: Default

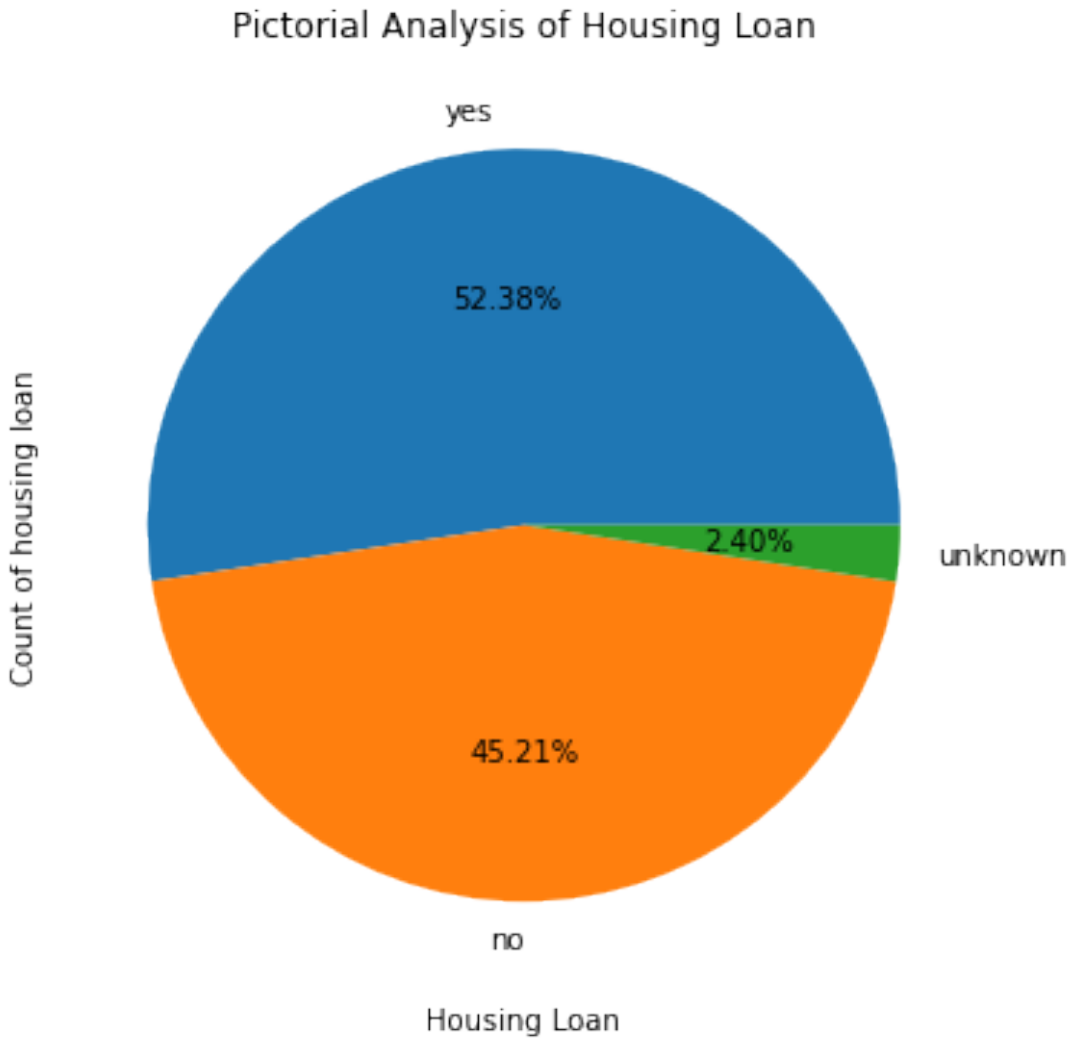
no	79.120132
unknown	20.872584
yes	0.007284



Most number of customers in this feature have not defaulted with 79%. Bank has client very le ss who are loan defaulter.

Variable: Housing

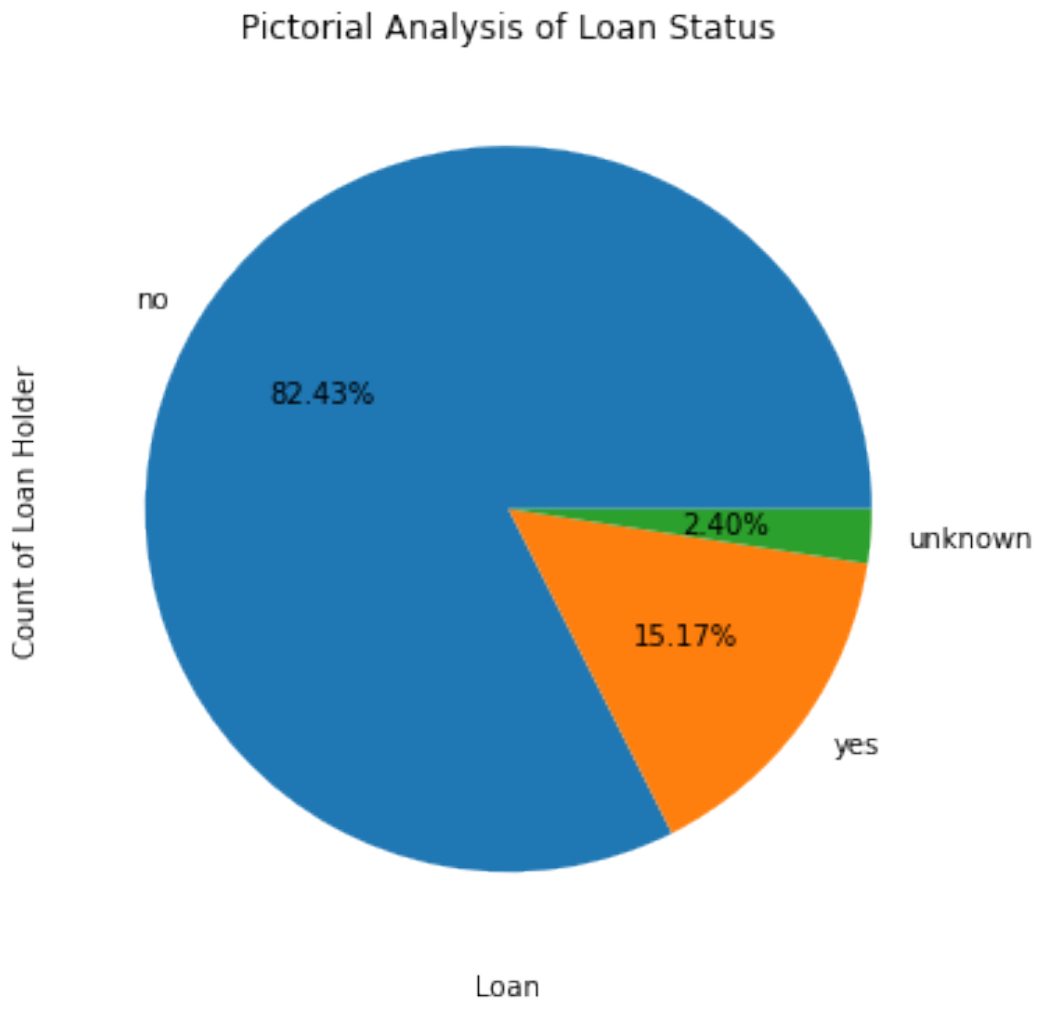
yes	52.384190
no	45.212198
unknown	2.403613



Most number of customers almost 52% have taken a housing loan and 45.21% have not taken any loan.

Variable: Loan

no	82.426920
yes	15.169467
unknown	2.403613

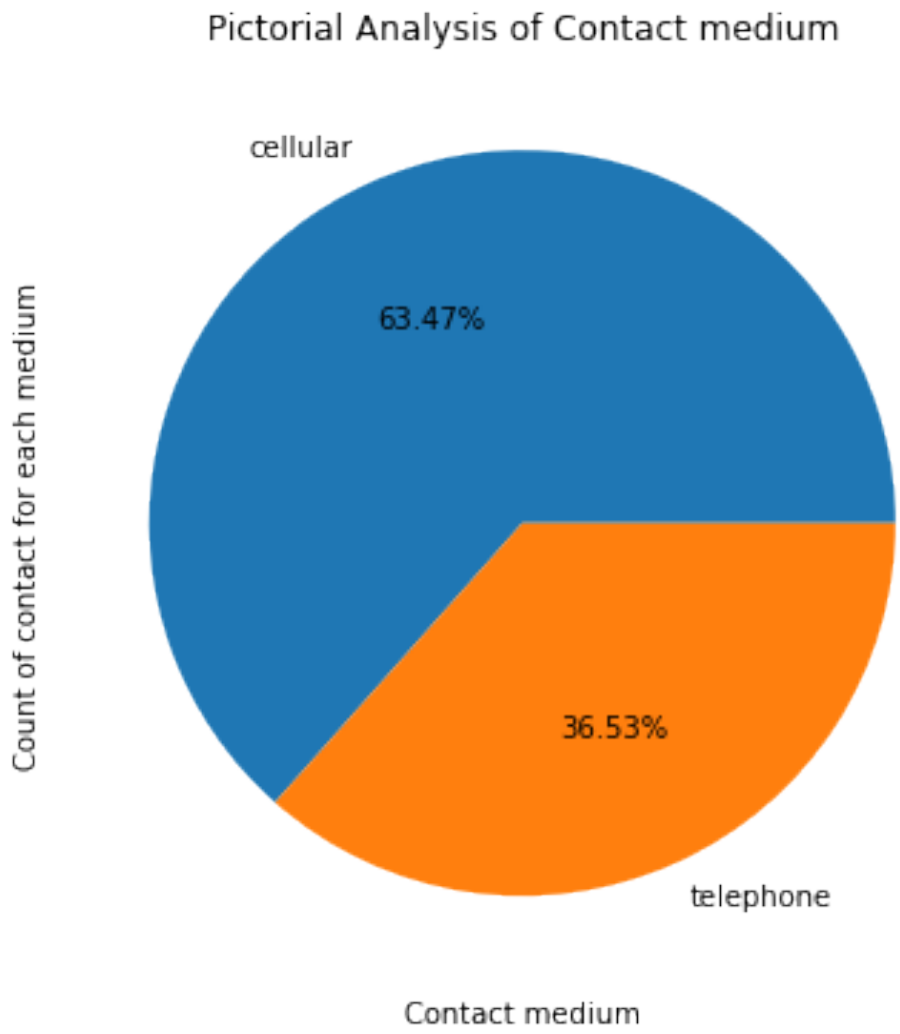


There are 82% customers who has no personal loan.

There are only a few customers almost 15.17% who have taken a personal loan. There are ve ry less almost 2% customers is unknown which we don't know has any loan or not.

Variable: Contact

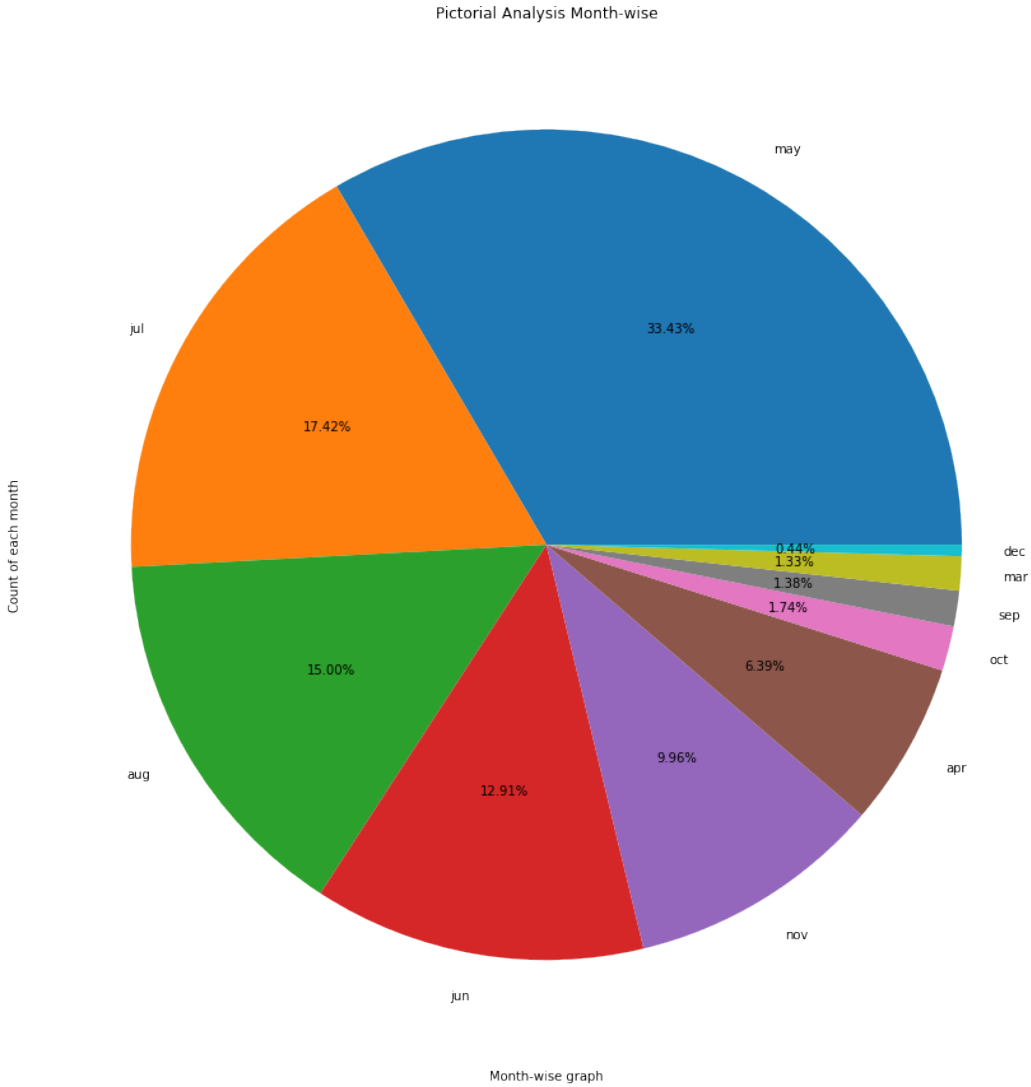
cellular	63.474798
telephone	36.525202



Most customers use a cellular medium of contact followed by telephone

Variable: Month

may	33.429640
jul	17.417694
aug	14.999514
jun	12.911528
nov	9.956784
apr	6.390211
oct	1.743226
sep	1.383898
mar	1.325629
dec	0.441876

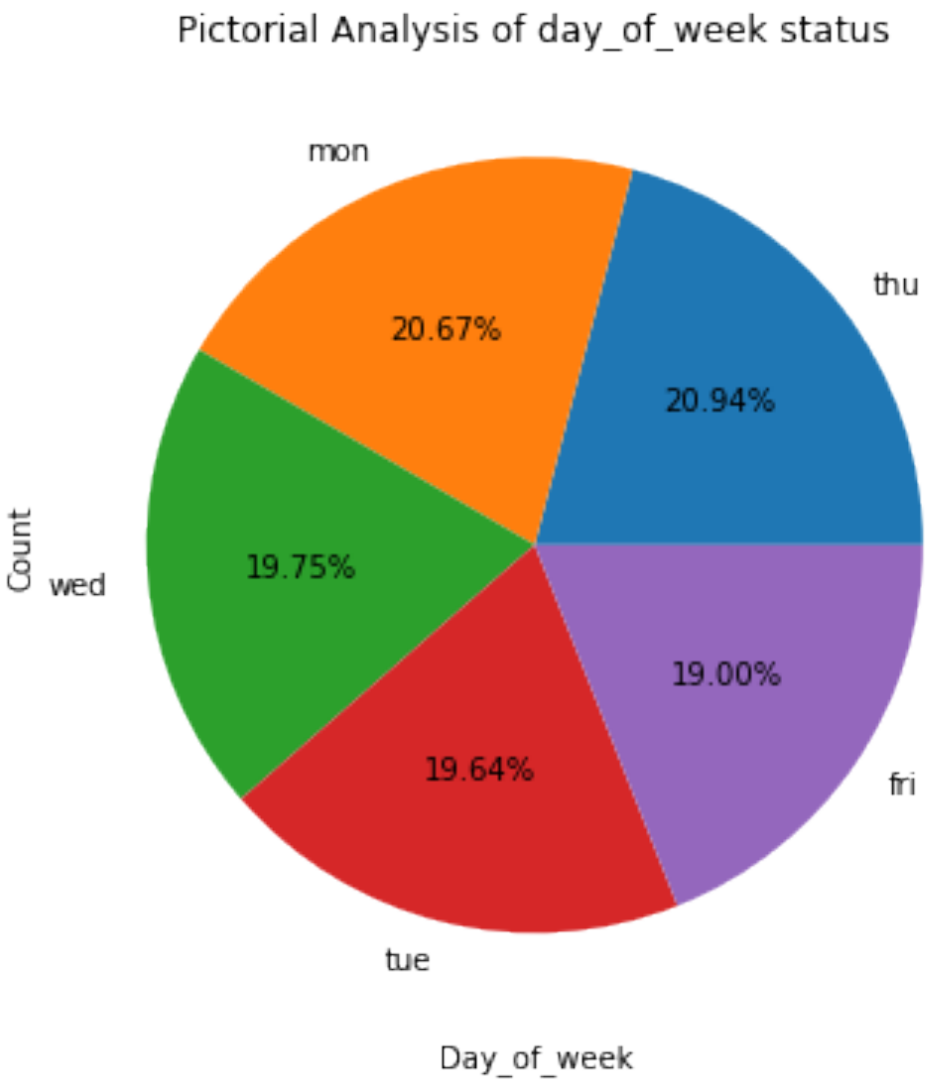


Most number of calls were made in May followed by July and August.

Very less call made in December followed by March and September.

Variable: Day of the week

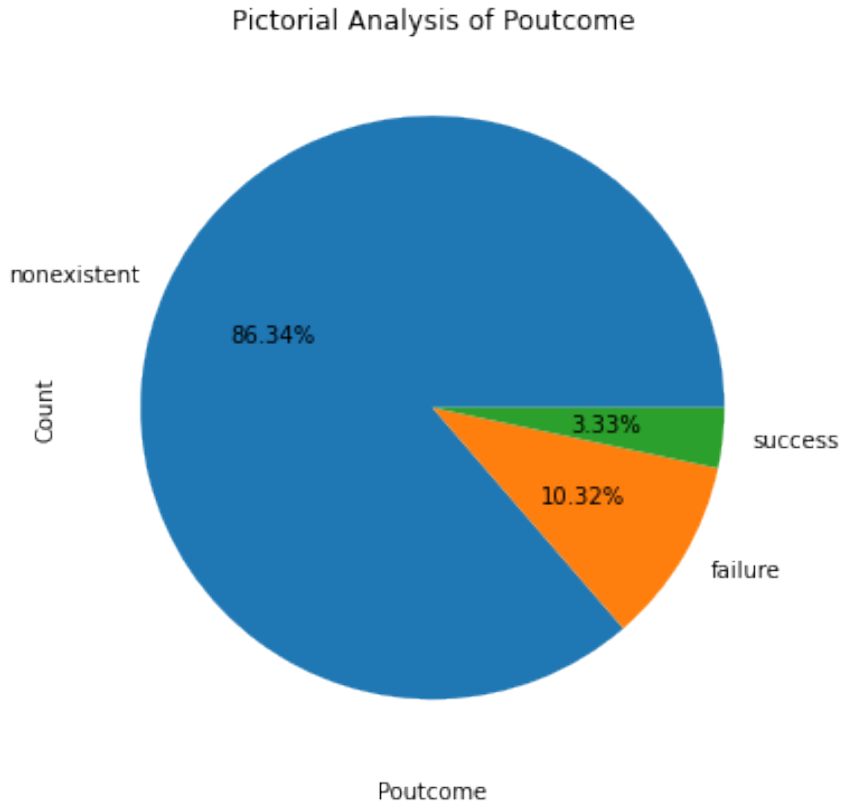
thu	20.935709
mon	20.671069
wed	19.748470
tue	19.641643
fri	19.003108



Similar number of calls are made each day throughout the week.

Variable: poutcome

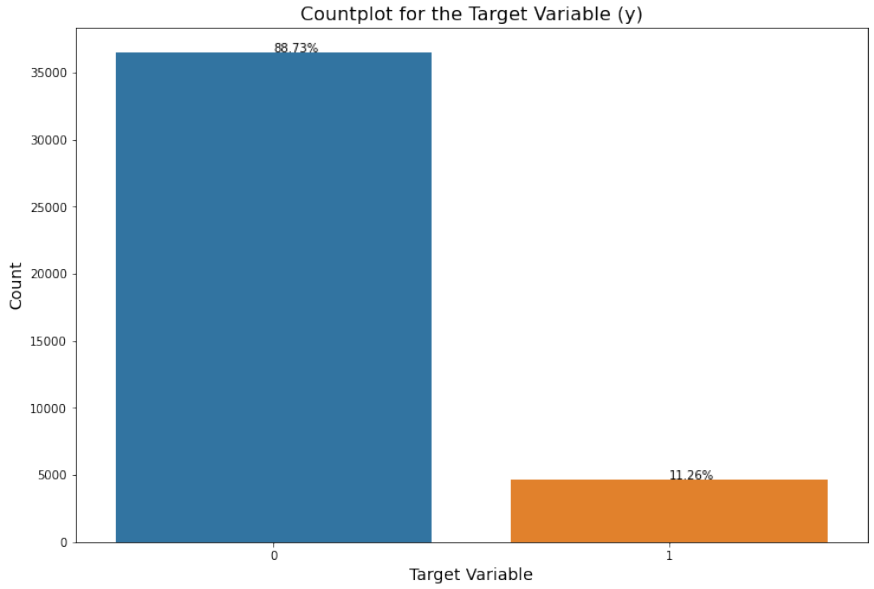
nonexistent	86.343110
failure	10.323395
success	3.333495



A huge numbers of customer falls under the 'non-existent' category followed by failure. There are very less customer under success category.

Variable: y and checking class imbalance

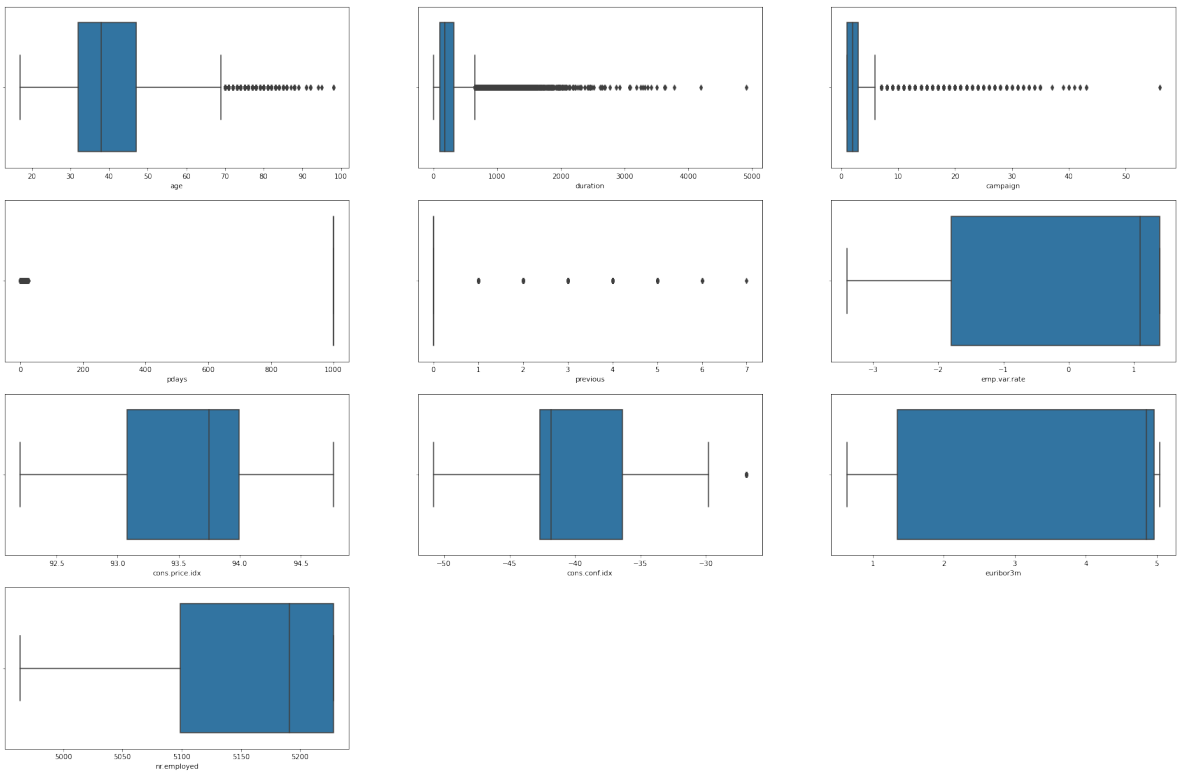
0	88.734583
1	11.265417



We do see an imbalance in our target variable as the 'No' variables are 88.73% of the data and the 'Yes' variables are only at 11.26%.

- **Univariate Analysis**

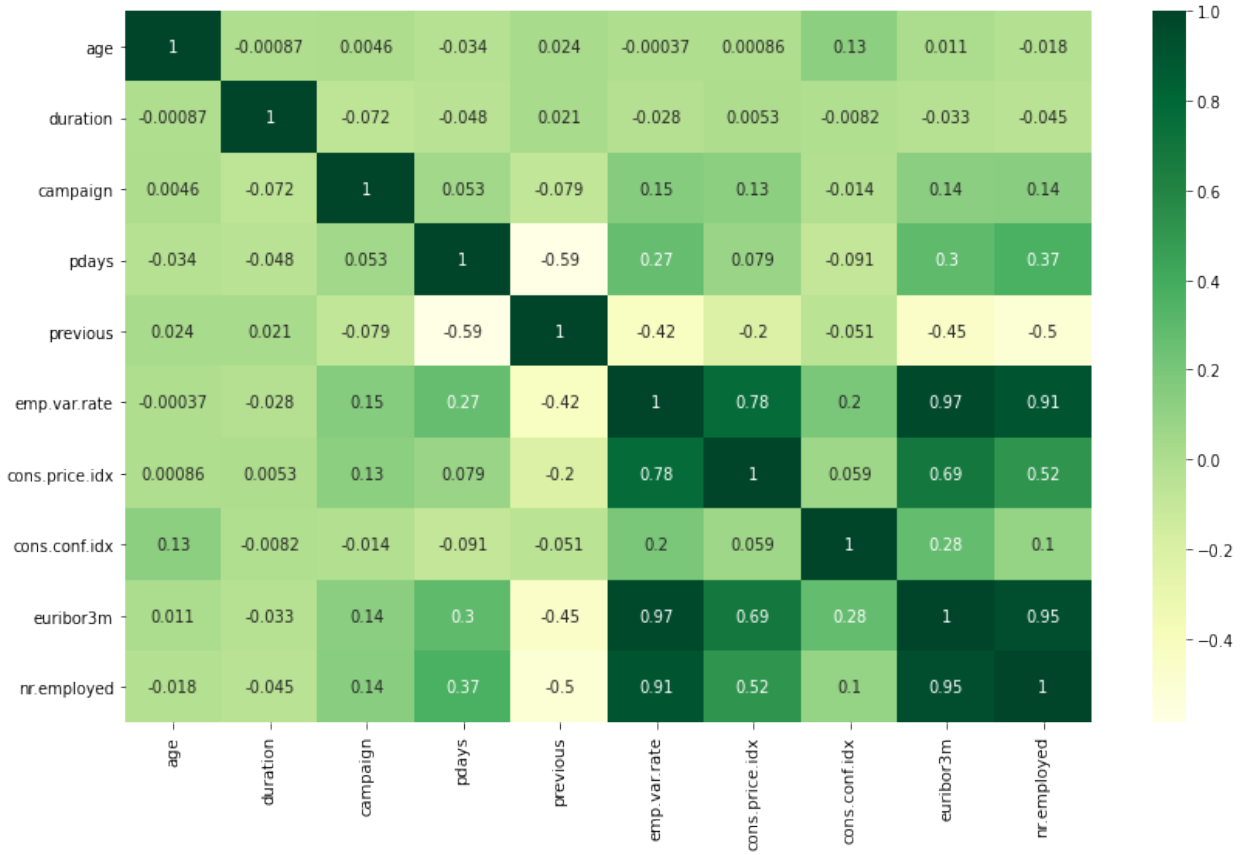
Numerical



- In the ‘age’ column we can clearly see that most of our clients age is between 30-50 years. And the average age is nearly 38 years.
- In the ‘duration’ column we can see that most of the call duration made by the bank is rarely 500 seconds.
- In the ‘campaign’ column nearly 1 to 4 times contacts are made for any clients for this campaign.
- In the ‘pdays’ column we can clearly see that the most of the clients are not contacted previously from the campaign.
- In the ‘Previous’ column , most of the client are not contacted before the campaign through the bank but we can see very few clients are contacted for at least once.
- We can see there is a high employee variation rate in the ‘emp.var.rate’ column rate from which we can say that they have made the campaign when there were high shifts in job due to the economic conditions.
- The ‘Consumer price index’ is good from which we can say the leads were having good price to pay for goods and services may be that could be the reason to stimulate these leads into making a deposit and plant the idea of savings.
- ‘Consumer confidence index’ is pretty low as we can say they have not much confidence about the economy.
- The ‘3-month Euribor interest rate’ is the interest rate from which we can say interest rate are high for lending a loan.
- The ‘number of employees’ has high peek we can say the income index are also high so that this campaign target the employed person for saying yes.

Checking for multicollinearity:

Heatmap



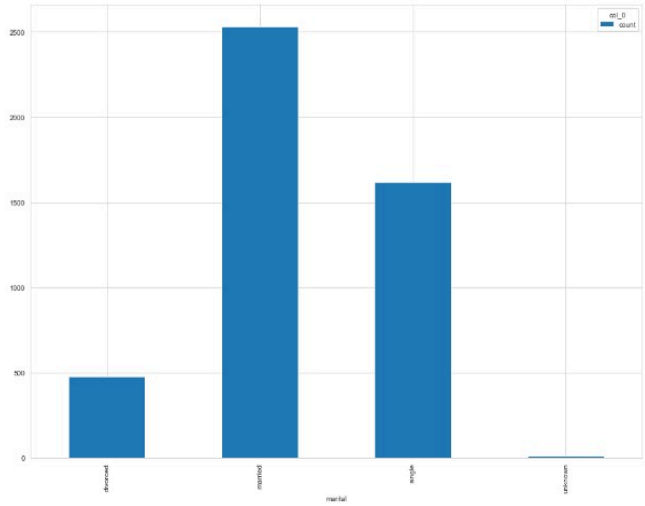
The indicators have correlation among themselves Number of employees rate is highly correlated with employee variation rate. Consumer price index is highly correlated with bank interest rate (higher the price index, higher the interest rate).Employee variation rate also correlates with the bank interest rates

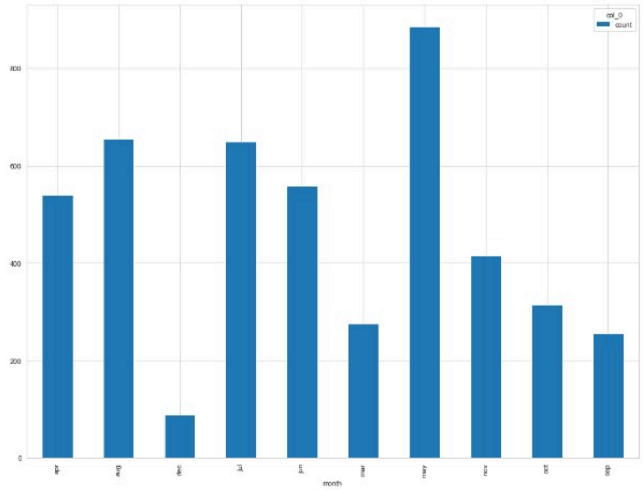
Pairplot:

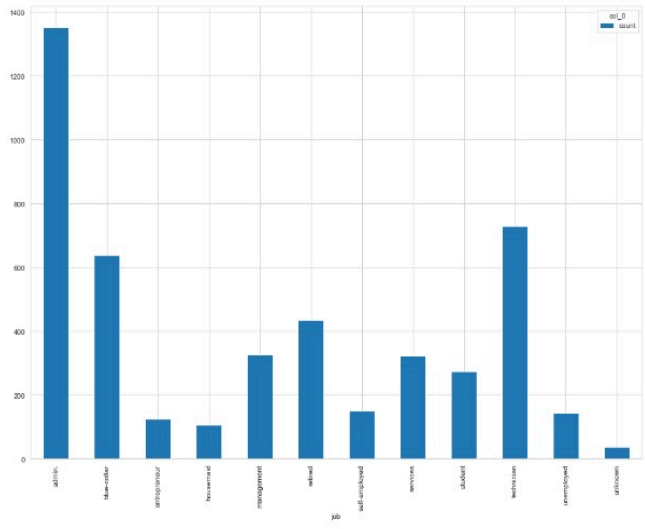
- In the pair plot, we can see that the variable emp.var.rate having high correlation with variables euribor3m and nr.employed, moderate correlation with cons.price.idx
- Cons.price.idx is moderately affecting euribor3m and nr.employed.

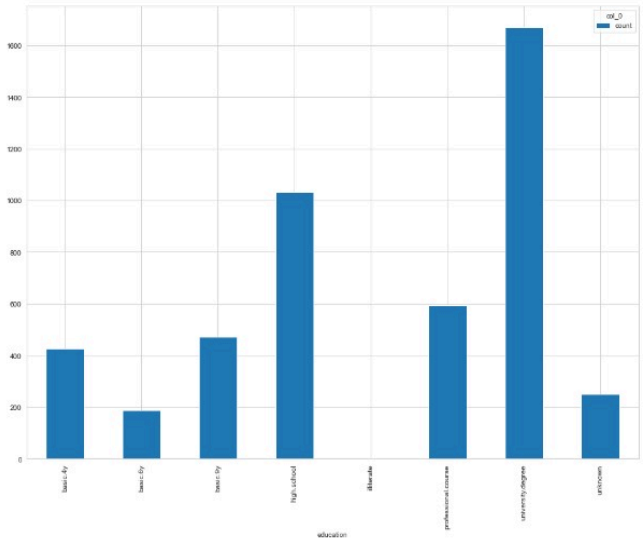
- euribor3m and nr.employed are highly affecting each other.

Positive deposit Analysis across all the attributes:





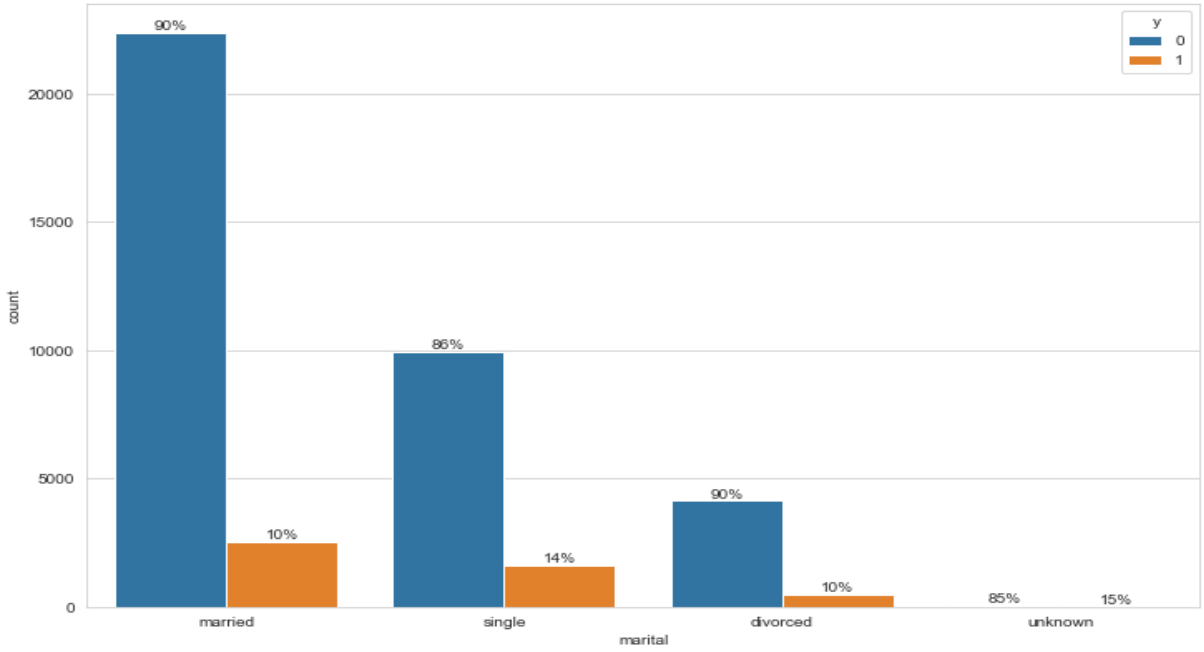




- Married leads have made high deposits followed by single
- There were much deposits made during may month as it is the start of bank period
- Leads who work in administrative position made deposits followed by technicians and blue collar employees
- Leads who had at least university degree had made the deposits followed by high school

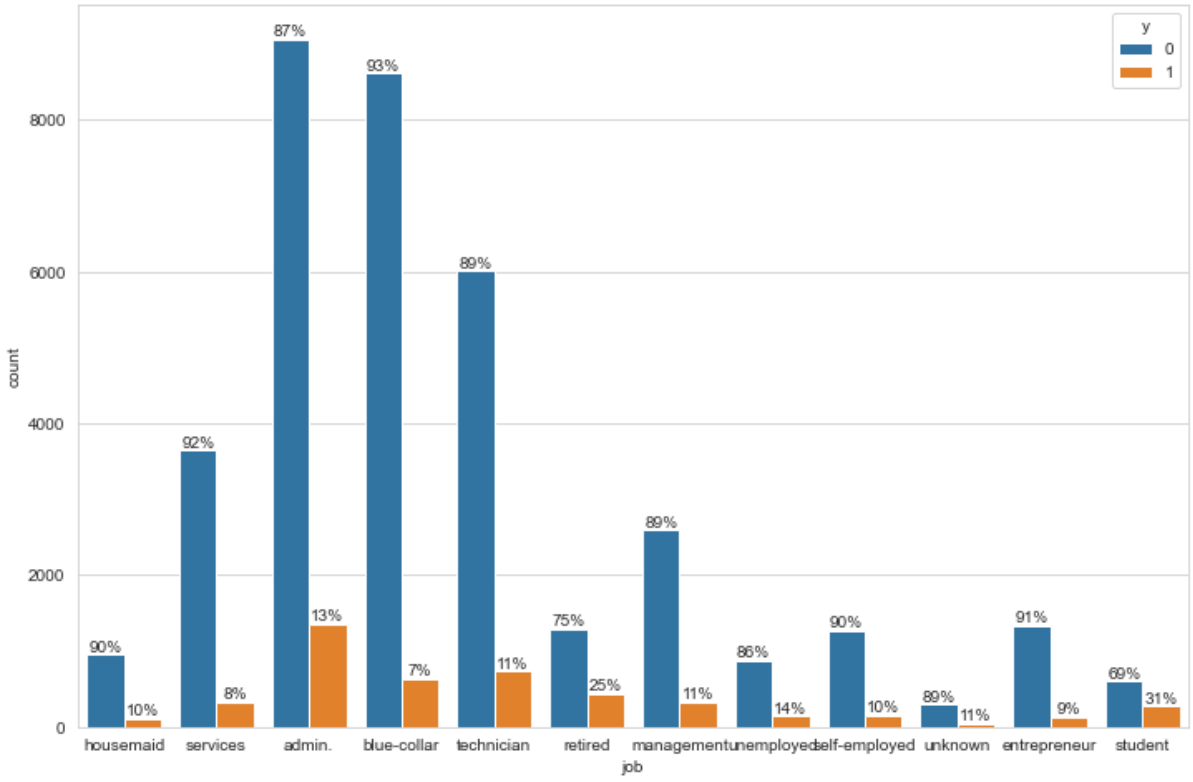
Positive Negative Target Analysis with Attributes:

Marital Status with the Target



According to all married person 10% has subscribed, among all the single customers 14% people subscribed and among all the divorced person 10% has subscription. There is an unknown sector which has very less clients among less clients 15% has subscription.

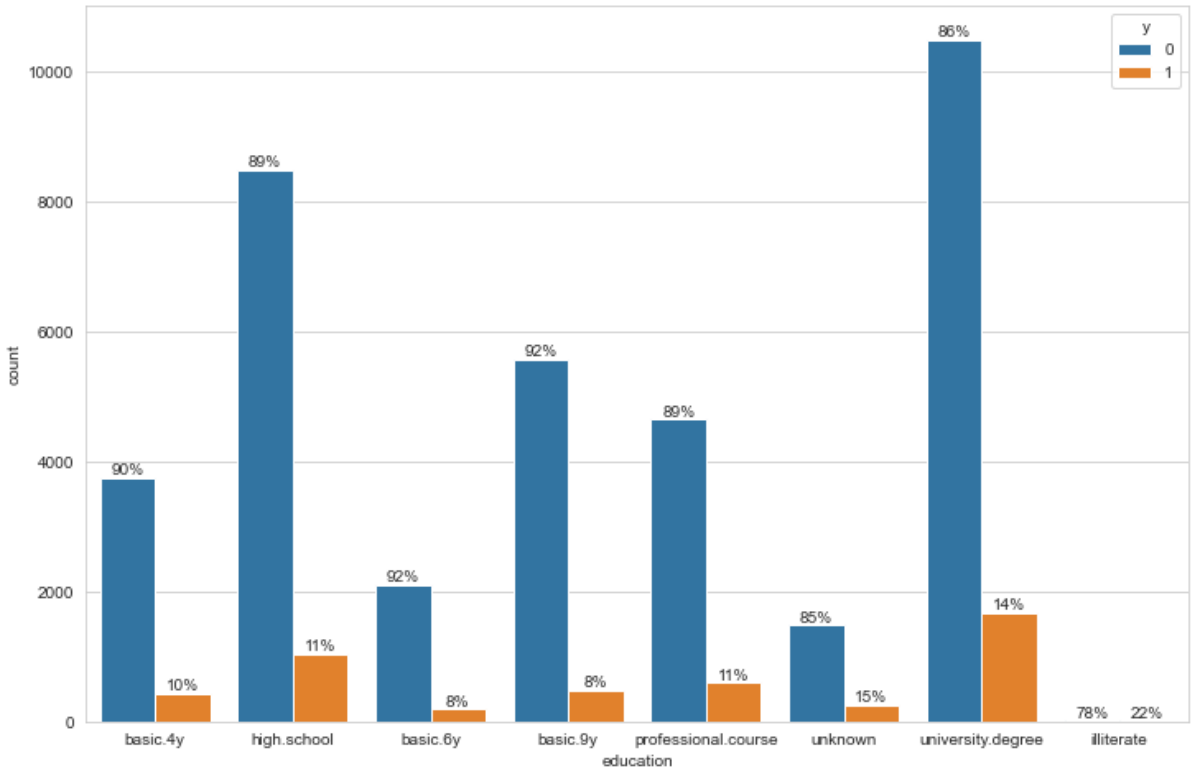
Job with the Target



Among all the contacted customer who are students 31% made the subscriptions, 25% of retired

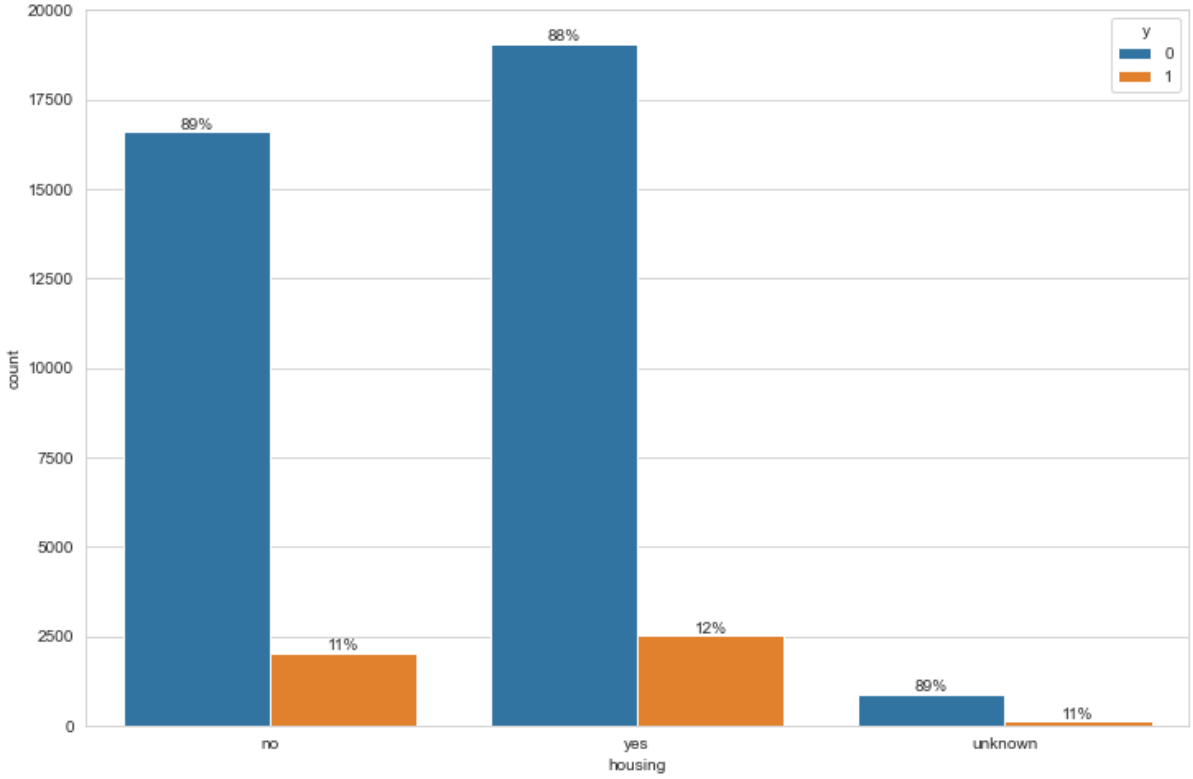
person has made subscription but most of the clients who are in the admin group but only 13% made the subscription.

Education with the Target:



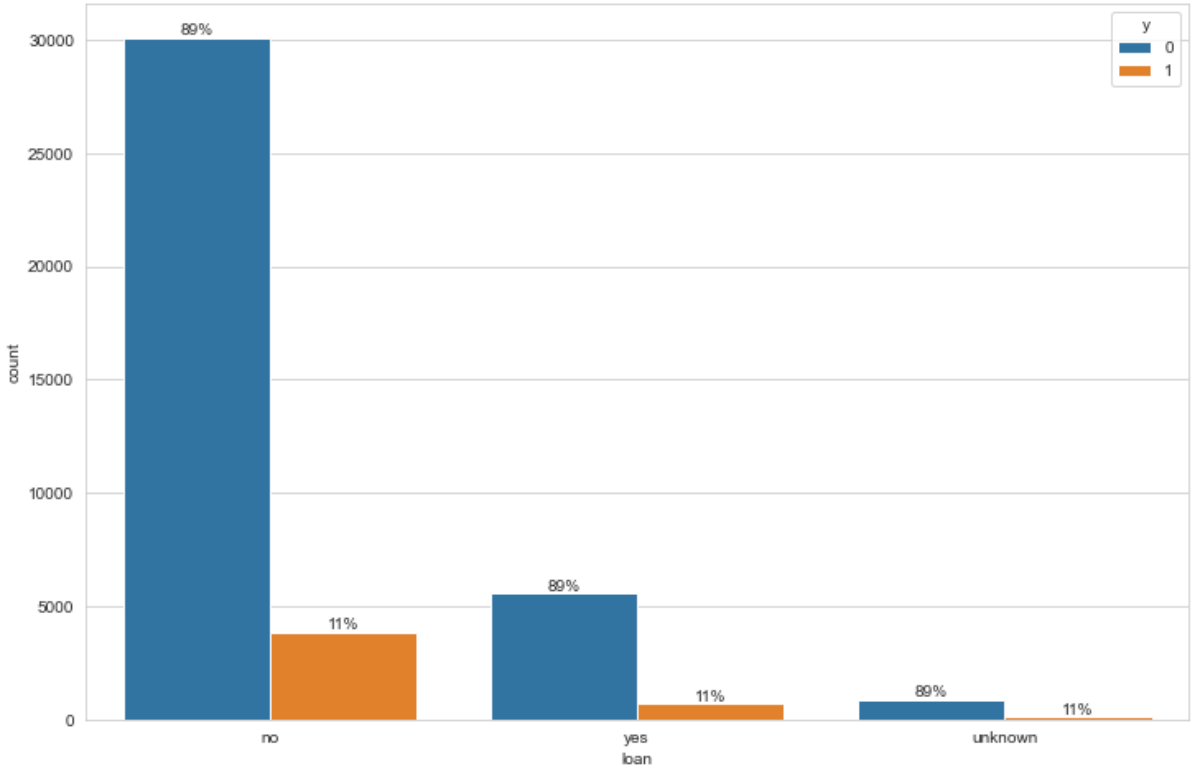
Among All the contacted illiterate person 22% has subscribed but university degree has the highest subscriber followed by High School.There are an unknown section whose education we don't know 15% did subscription.followed by university degree with 14%.

Housing with the Target



Bank got highest number of deposit who has housing loan previously 12% customer subscribe the deposit who has previous loan. But almost who has no housing loan has also subscribe the deposite.

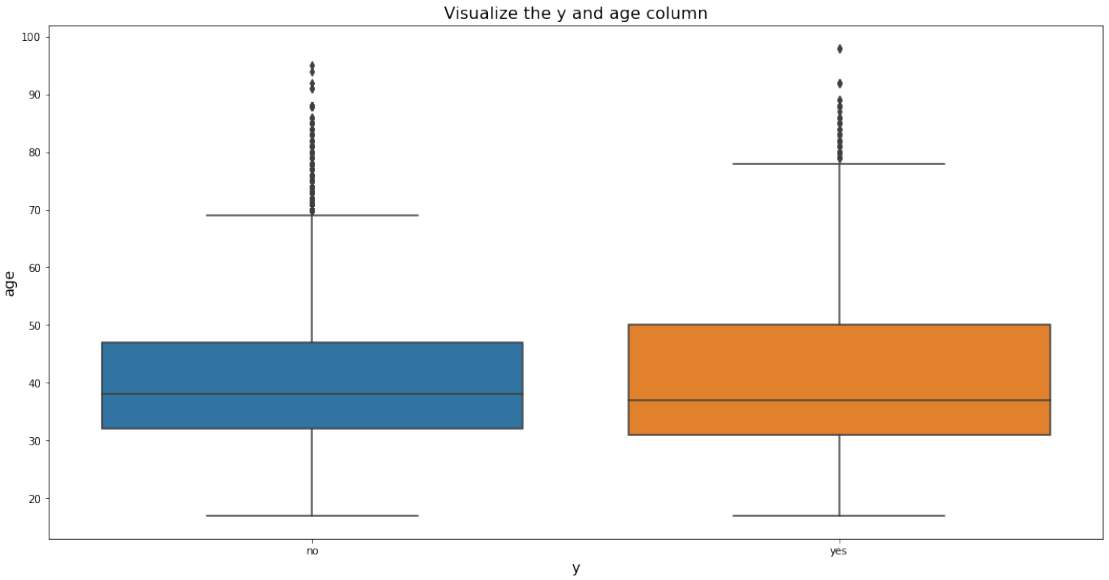
Loan with the Target:



Almost there are similar success ratio for every section who has previous personal loan and not any previous personal loan.

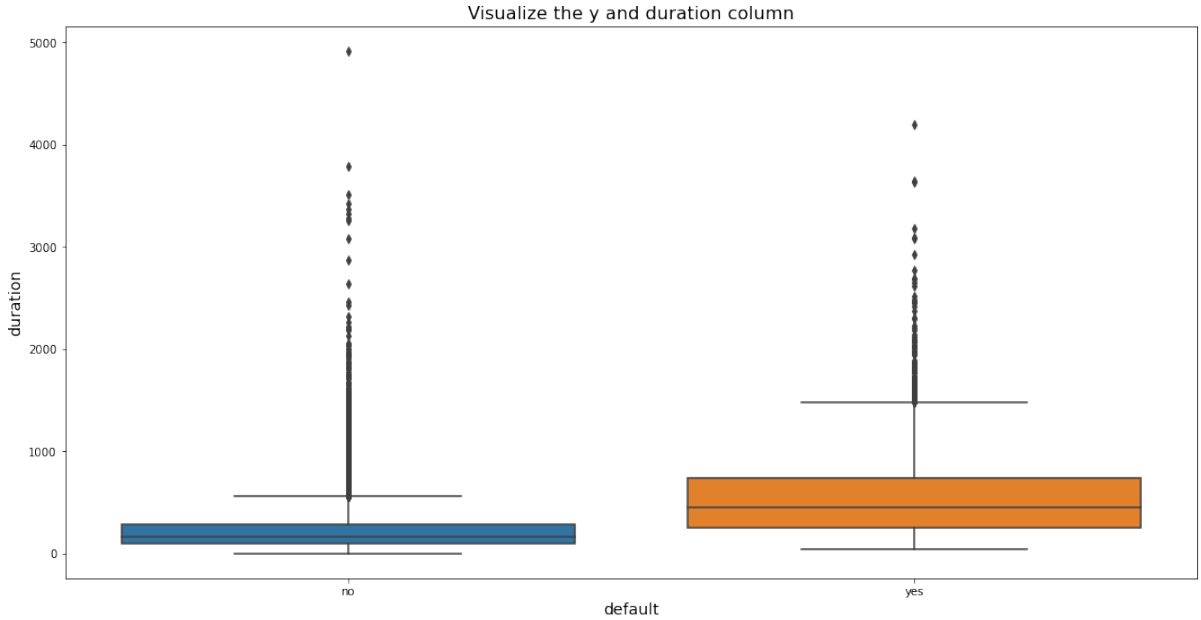
Analysis of the independent numerical variables with the target variable 'y':

Age and y:



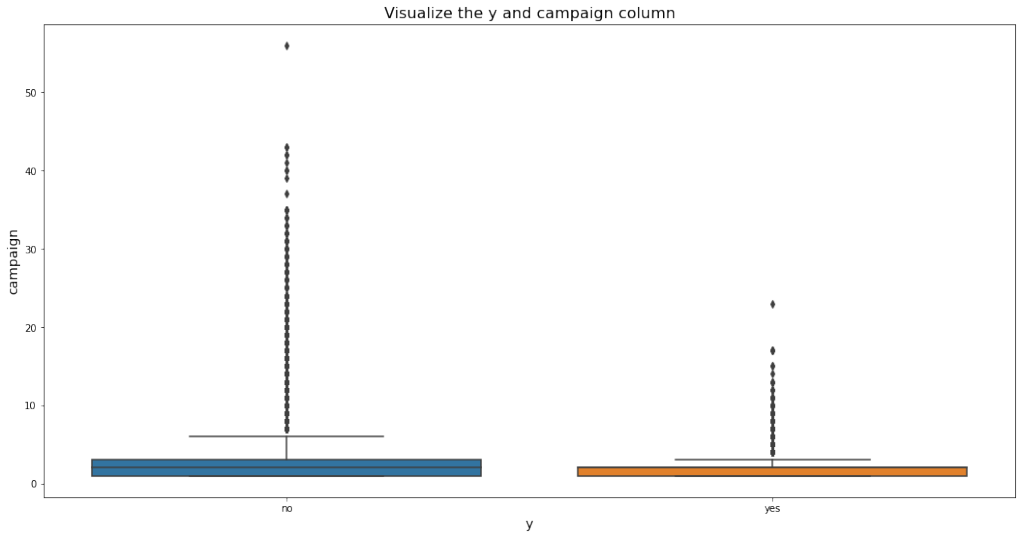
Subscribers are of a higher range of age than that of non-subscribers of term deposit.

Duration and y:



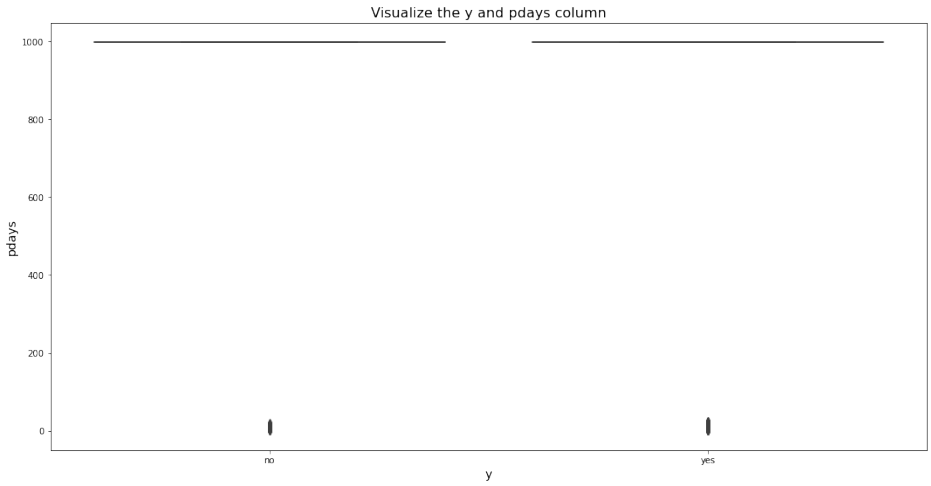
The customers who said yes had a longer duration of calls.

Campaign and y:



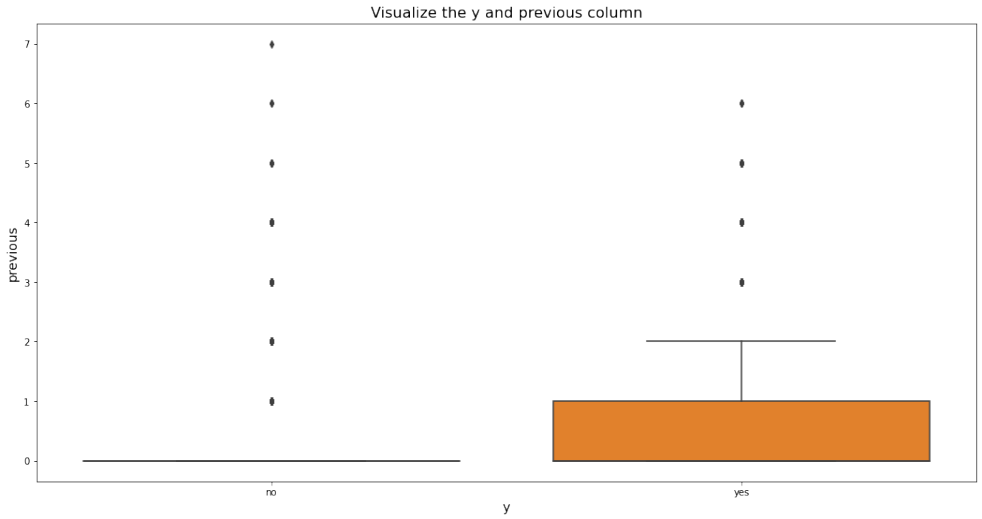
In this plot is observed that more the contact made to the client the outcome was non-subscription.

Pdays and y:



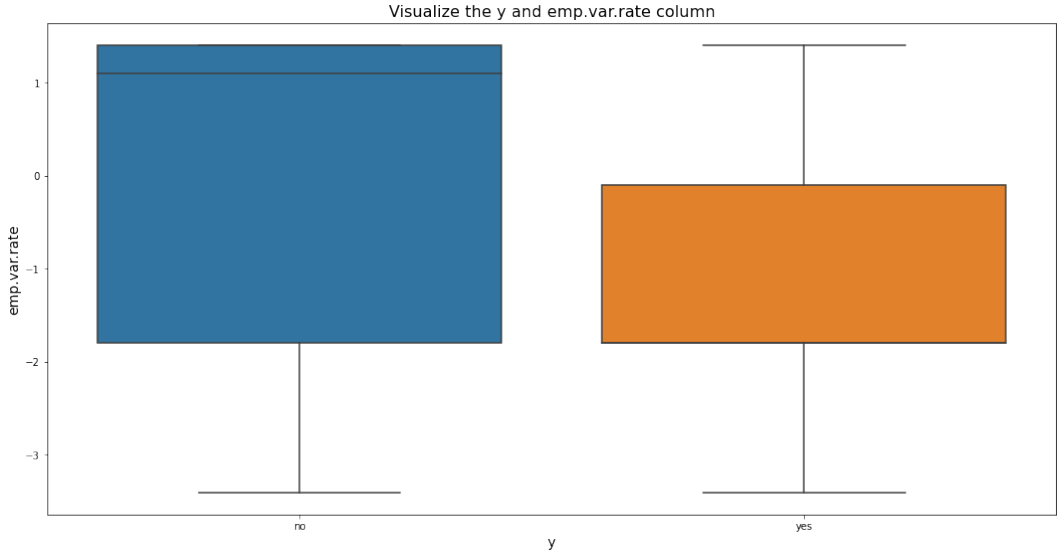
The number of days from last contacting the client does not seem to be affecting in any way.

Previous and y:



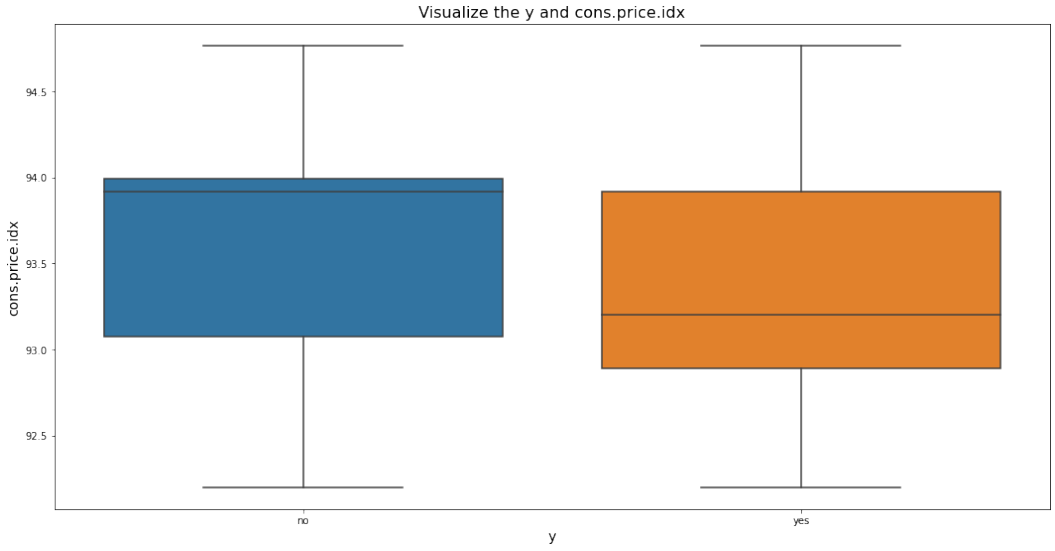
The number of contacts made before this particular campaign plays a role in the customer's decision of subscription.

Emp.var.rate and y:



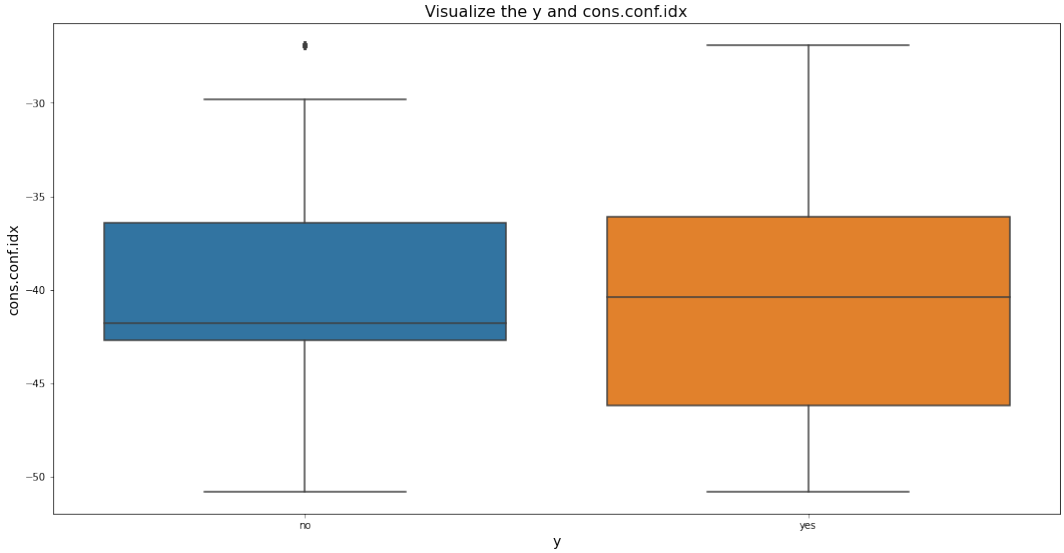
The higher range of employee variation rate results a no.

Cons.price.idx and y:



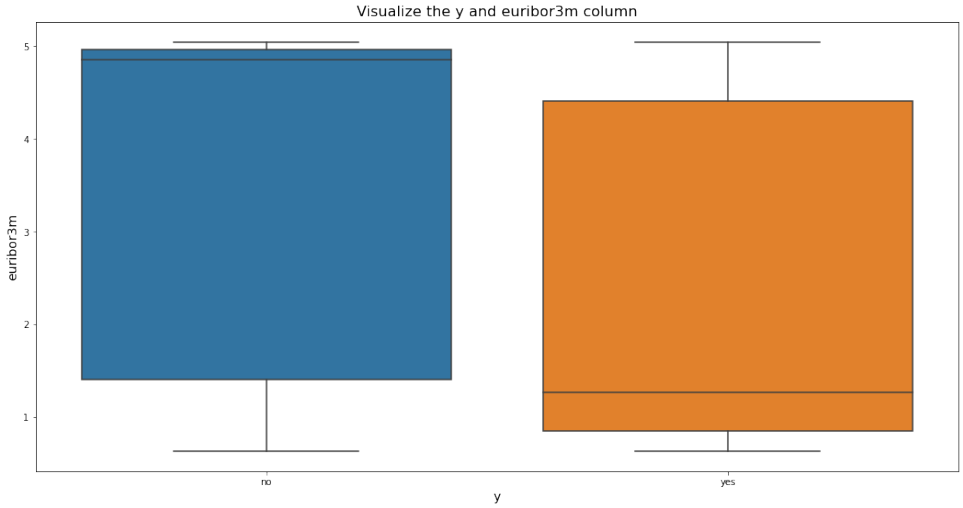
Higher range of cons.price.idx in the ones that have subscribed.

Cons.conf.idx and y:



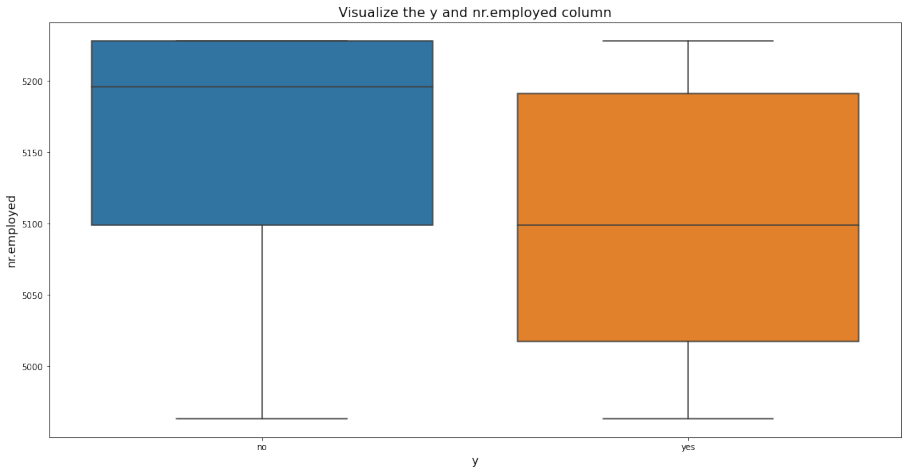
Higher range of cons.conf.idx is seen the subscribers than the non-subscribers.

Euribo3m and y:



Higher range of euribo3m in the subscribers in subscribers' box.

Nr.employed and y:

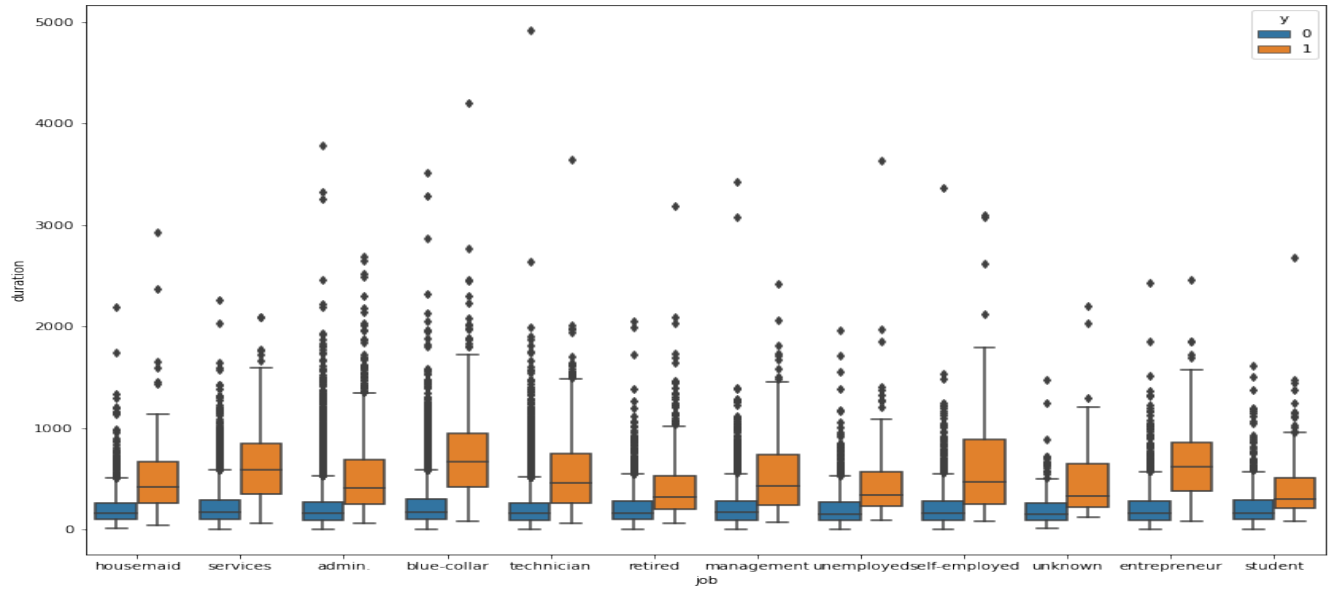


Higher range of number of employees in the subscribers' box.

From the above 4 plots, we observed a difference in the medians. It means that these variables will be useful for our model. But we can only know for sure when we move ahead with our feature engineering, selection and other treatments.

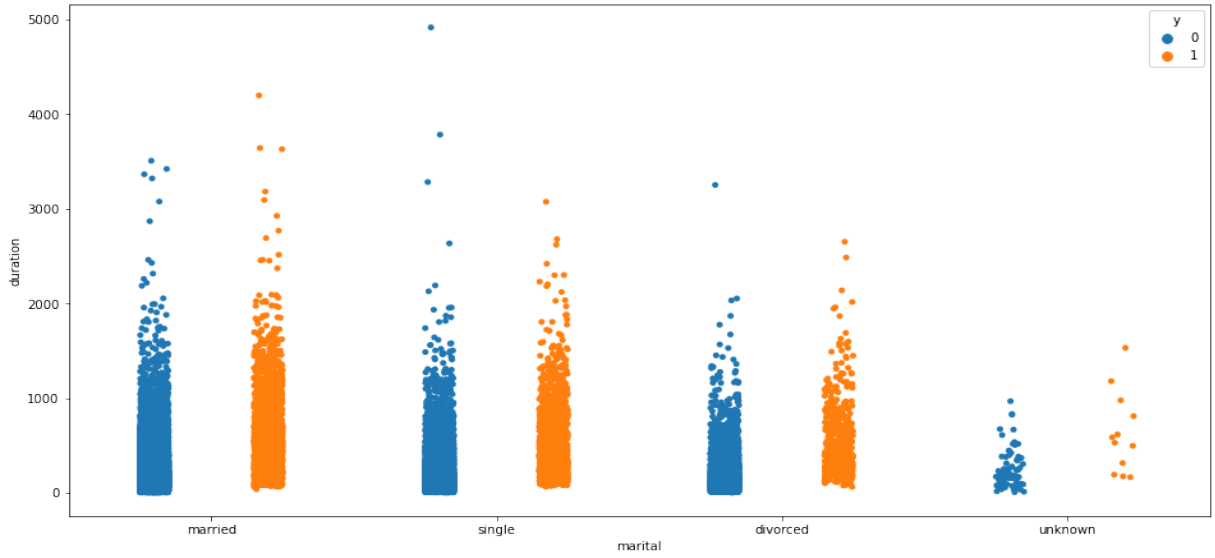
- **Multivariate Analysis:**

Variables Job,Duration, y



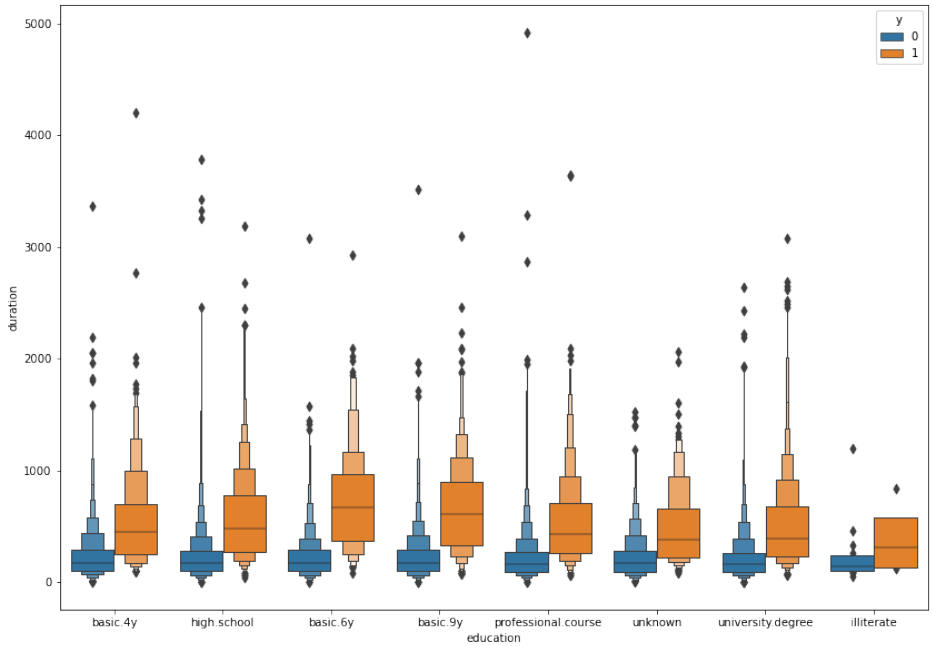
- The leads who have not made a deposit have lesser duration on calls
- Comparing the average, the blue collar, entrepreneur have high duration in calls and student, retired have less duration in average
- Large distribution of leads were from self-employed clients and management people.

Variables Marital,Duration, y



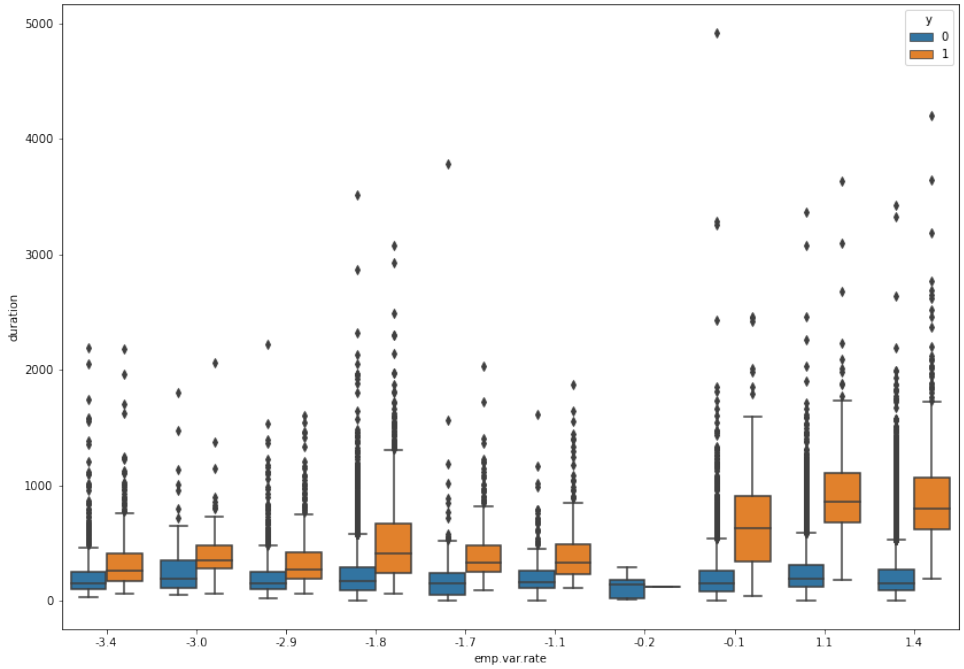
Married people who longer duration of calls comparatively have subscribed to the deposit.

Variables Education,Duration, y



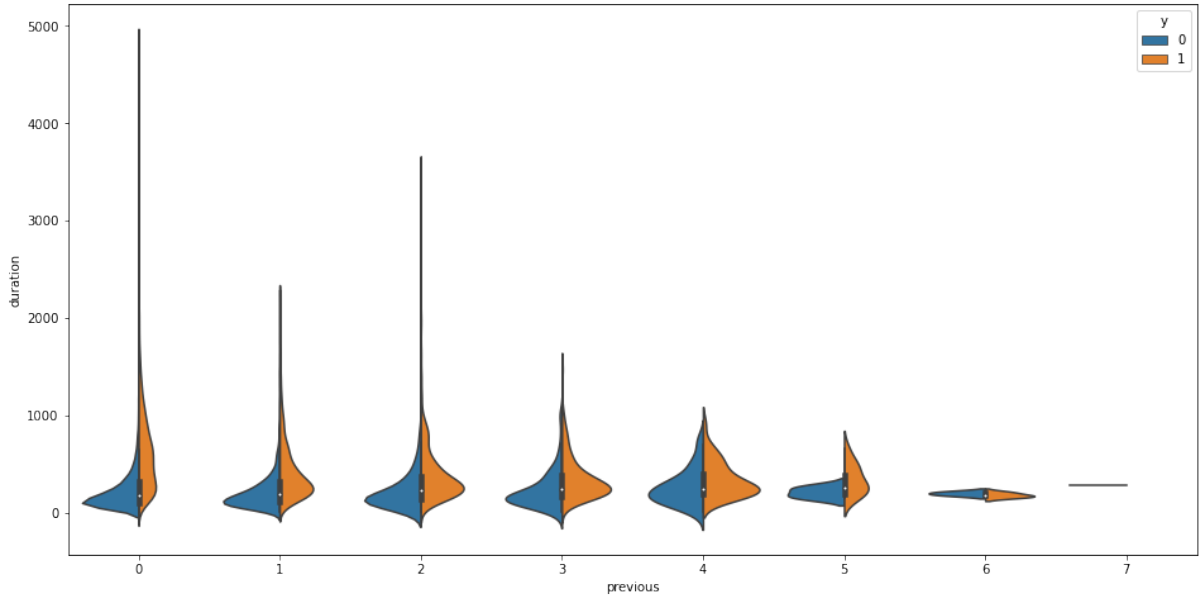
The ones who have a university degree and have subscribed had longer duration calls.

Variables emp.var.rate,Duration, y



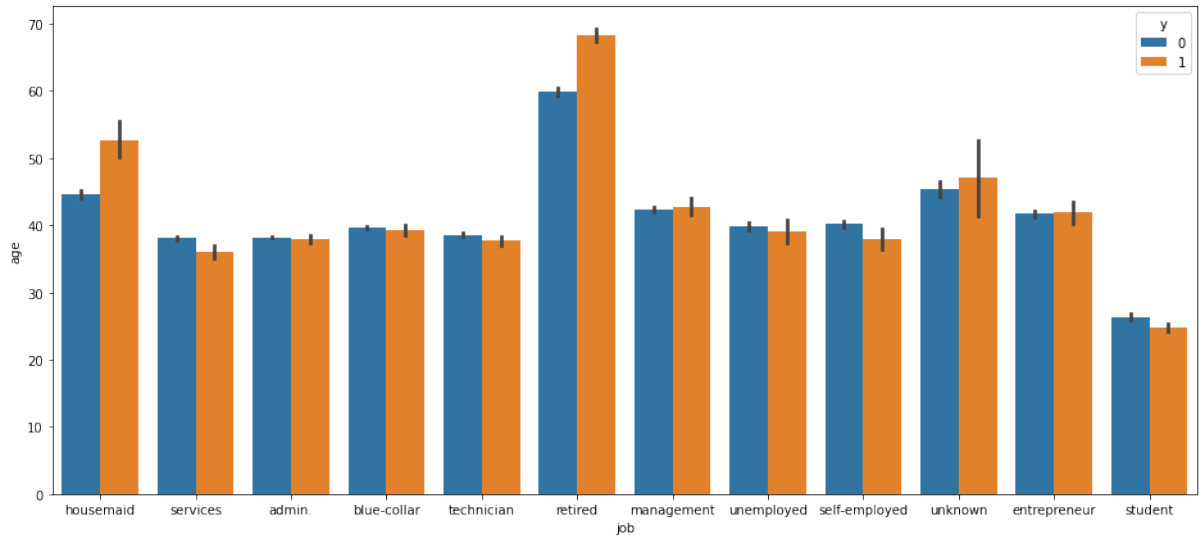
A emp.var.rate of -0.1 and have subscribed had longer duration calls.

Variables Previous,Duration, y



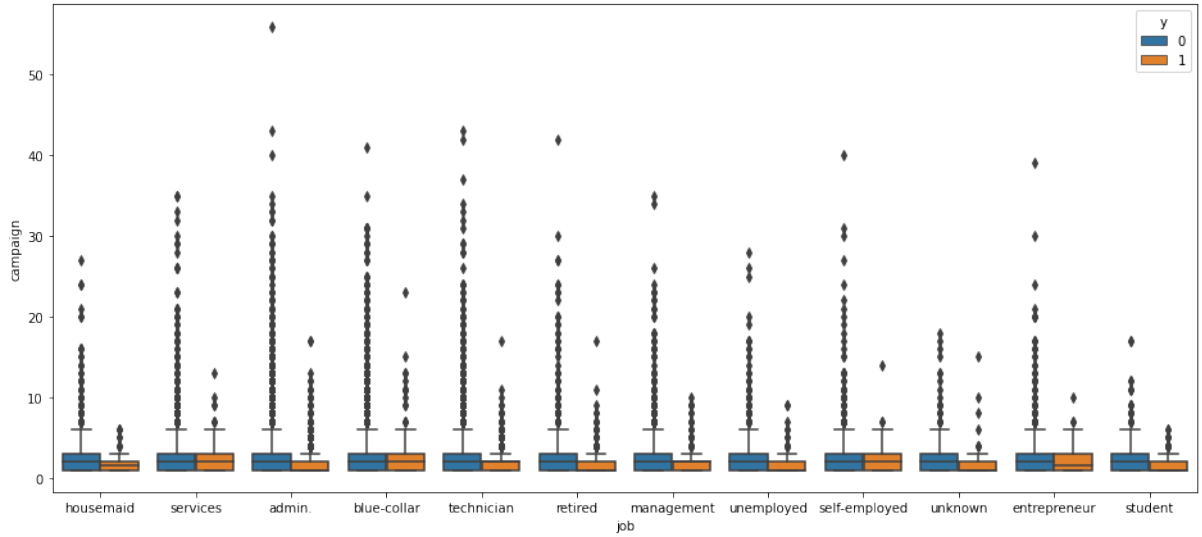
Ones having no previous contact have longer duration of calls.

Variables Job,Age, y



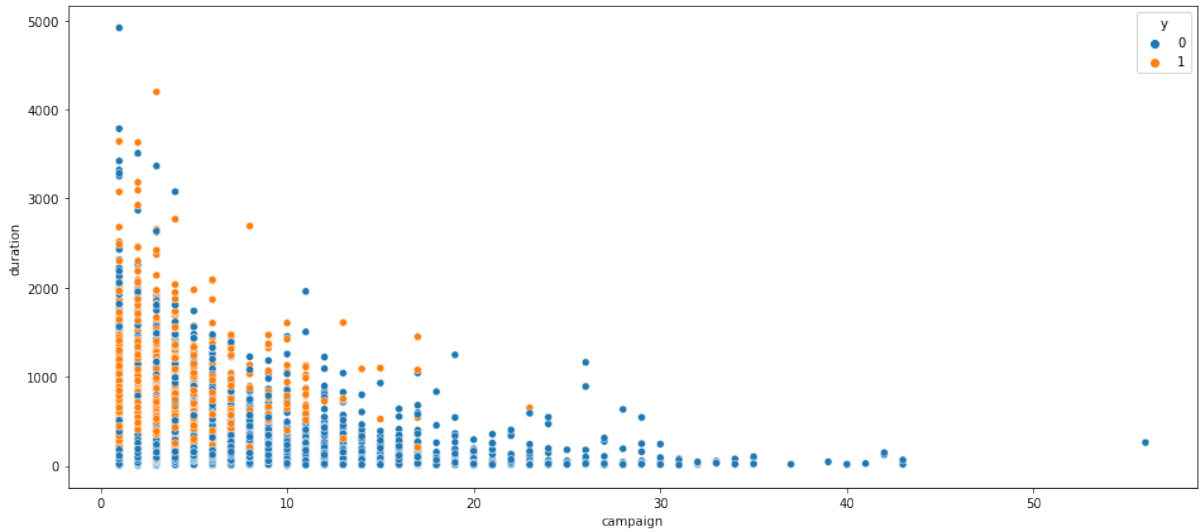
A lot of people who are retired have term deposits whose age is almost nearly 70.

Variables Job,Campaign, y



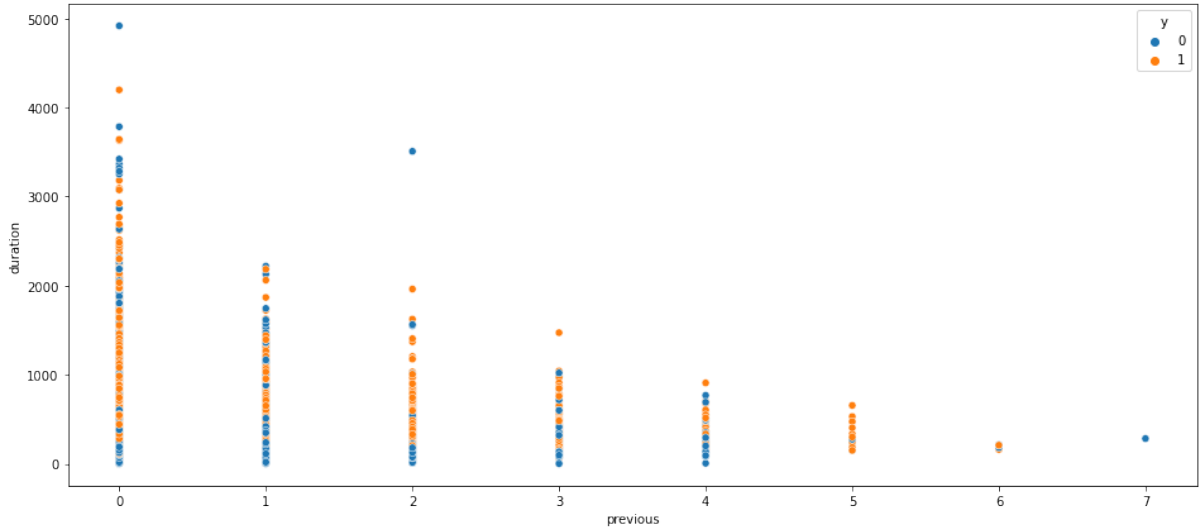
The number of campaigns done in the past with clients of specific job does not seem to be affecting in converting the decision to 'yes'.

Variables Campaign, Duration,y



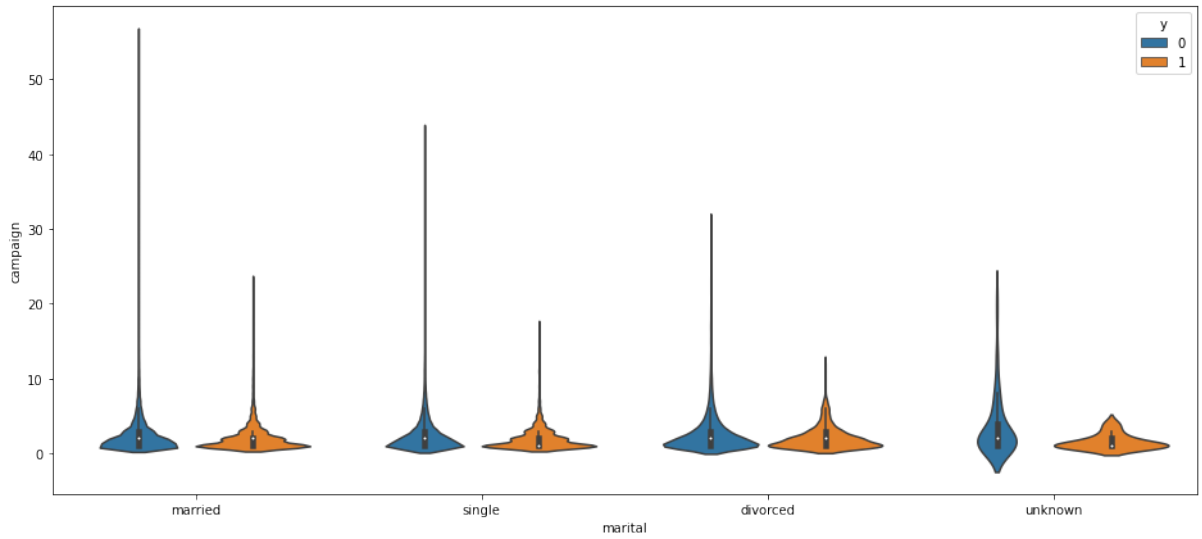
- The more the duration the calls were, they had higher probability in making a deposit
- Duration of calls faded as the time period of campaign extended further
- There were many positive leads in the initial days of campaign

Variables Previous, Duration,y



It looks like lesser previous calls made leads to longer durations of calls but we see successful subscriptions happening when the previous calls made are increasing although the number of subscriptions is getting less.

Variables Marital,Campaign,y



More previous campaigns conducted on were married people, but number of successful subscriptions is lesser.

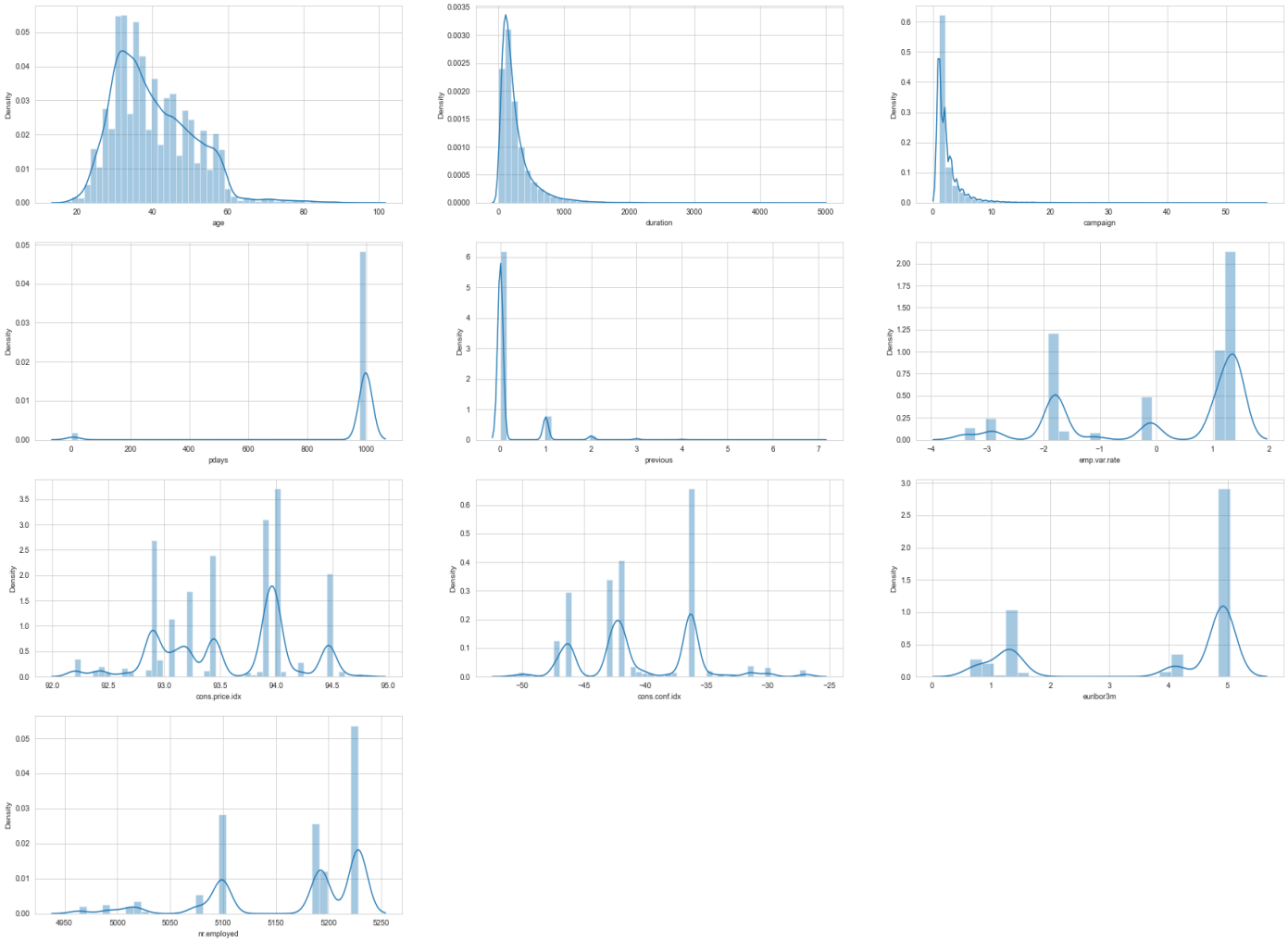
Outliers and Outlier Treatment:

We could see the presence of outliers in a lot of plots above.

Let us check the skewness of the numerical variables and treat them accordingly.

age	0.784697
duration	3.263141
campaign	4.762507
pdays	-4.922190
previous	3.832042
emp.var.rate	-0.724096
cons.price.idx	-0.230888
cons.conf.idx	0.303180
euribor3m	-0.709188
nr.employed	1.044262

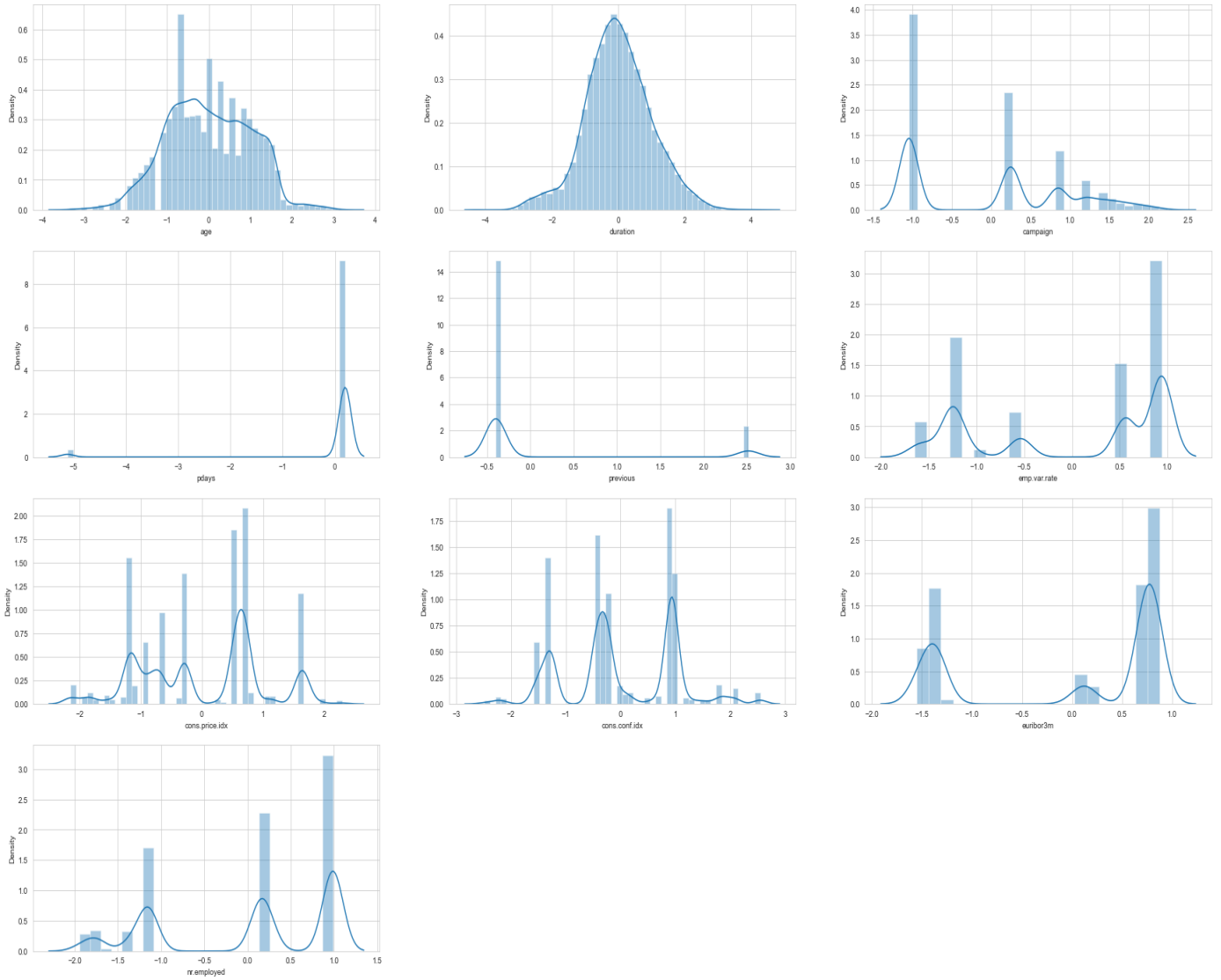
Visualising through distplots:



After doing PowerTransform:

age	0.007336
duration	0.016940
campaign	0.268430
pdays	-4.922068
previous	2.116794
emp.var.rate	-0.414214
cons.price.idx	-0.038410
cons.conf.idx	0.047975
euribor3m	-0.649814
nr.employed	-0.502649

Visualising through distplots: We can see the skewness reduced in the plots.



Statistical Analysis:

Chi-Square Test for Independence

1. Chi-Square Test for Independence for attributes job and target variable (y)

:>

The null and alternative hypothesis is:

H_0: The variables **job** and **target variable (y)** are **Independent.**
H_1: The variables **job** and **target variable (y)** are **Dependent.**

Chi-Square Test Output: -

```
Test statistic: 961.2424403289554
p-value: 4.189763287563861e-199
Degrees of freedom: 11
Expected values: [[9247.91822861 8211.49830048 1291.97552685  940.58657862 2594.59920365
  1526.23482568 1260.91842284 3521.87559483  776.42760027 5983.37292415
   899.76867049  292.82412353]
 [1174.08177139 1042.50169952  164.02447315  119.41342138  329.40079635
   193.76517432  160.08157716  447.12440517   98.57239973  759.62707585
   114.23132951   37.17587647]]
```

Decision rule: -
If the p-value of the chi-square test is greater than α=0.05 then we accept the null hypothesis.
Decision: -
As the p-value of the chi-square test is less than α=0.05.Hence, we reject null hypothesis.

Conclusion: -

The variables job and target variable(y) are Dependent.

2.Chi-Square Test for Independence for attributes `marital status` and `target variable (y)` :>

The null and alternative hypothesis is:

H_0: The variables **`marital status`** and **`target variable (y)`** are **`Independent`**.
H_1: The variables **`marital status`** and **`target variable (y)`** are **`Dependent`**.

Chi-Square Test Output: -

```
Test statistic: 122.65515182252989
p-value: 2.068014648442211e-26
Degrees of freedom: 3
Expected values: [[4.09243896e+03 2.21197568e+04 1.02648165e+04 7.09876663e+01]
 [5.19561037e+02 2.80824318e+03 1.30318345e+03 9.01233369e+00]]
```

Decision rule: -

If the p-value of the chi-square test is greater than α=0.05 then we accept the null hypothesis.

Decision: -
As the p-value of the chi-square test is less than α=0.05. Hence, we reject null hypothesis.

Conclusion: -

The variables marital status and target variable (y) are Dependent.

3.Chi-Square Test for Independence for attributes `education` and `target variable (y)` `:>`

The null and alternative hypothesis is:

H_0: The variables **`education`** and **`target variable (y)`** are **`Independent`**.
H_1: The variables **`education`** and **`target variable (y)`** are **`Dependent`**.

Chi-Square Test Output: -

```
Test statistic: 193.10590454149565
p-value: 3.3051890144025054e-38
Degrees of freedom: 7
Expected values: [[3.70555618e+03 2.03379664e+03 5.36400554e+03 8.44309556e+03
  1.59722249e+01 4.65235418e+03 1.07972240e+04 1.53599563e+03]
 [4.70443819e+02 2.58203360e+02 6.80994464e+02 1.07190444e+03
  2.02777508e+00 5.90645819e+02 1.37077595e+03 1.95004370e+02]]
```

Decision rule: -
If the p-value of the chi-square test is greater than α=0.05 then we accept the null hypothesis.

Decision: -
As the p-value of the chi-square test is less than α=0.05. Hence, we reject null hypothesis.

Conclusion: -

The variables `education` and `targetvariable(y)` are Dependent.

4.Chi-Square Test for Independence for attributes `default` and `target variable (y)`

:>

The null and alternative hypothesis is:

H_0: The variables **default** and **target variable (y)** are **Independent**.
H_1: The variables **default** and **target variable (y)** are **Dependent**.

Chi-Square Test Output: -

```
Test statistic: 406.5775146420093
p-value: 5.1619579513916376e-89
Degrees of freedom: 2
Expected values: [[2.89168259e+04 7.62851209e+03 2.66203749e+00]
 [3.67117413e+03 9.68487909e+02 3.37962513e-01]]
```

Decision rule: -
If the p-value of the chi-square test is greater than α=0.05 then we accept the null hypothesis.

Decision: -
As the p-value of the chi-square test is less than α=0.05. Hence, we reject null hypothesis.

Conclusion: -

The variables default and target variable (y) are Dependent.

5.Chi-Square Test for Independence for attributes housing and target variable (y)
:>

The null and alternative hypothesis is:

H_0: The variables **housing** and **target variable (y)** are **Independent**.
H_1: The variables **housing** and **target variable (y)** are **Dependent**.

Chi-Square Test Output: -

```
Test statistic: 5.684495858974168
p-value: 0.05829447669453452
Degrees of freedom: 2
Expected values: [[16524.15402544   878.47237059 19145.37360396]
 [ 2097.84597456   111.52762941  2430.62639604]]
```

Decision rule: -
If the p-value of the chi-square test is greater than α=0.05 then we accept the null hypothesis.

Decision: -
As the p-value of the chi-square test is greater than α=0.05. Hence, we fail to reject (i.e., accept) null hypothesis.

Conclusion: -

The variables housing and target variable(y) are Independent.

6.Chi-Square Test for Independence for attributes *loan* and *targetvariable (y)*

:>

The null and alternative hypothesis is:

H_0: The variables **loan** and **target variable (y)** are **Independent**.
H_1: The variables **loan** and **target variable (y)** are **Dependent**.

Chi-Square Test Output: -

```
Test statistic: 1.094027551150338
p-value: 0.5786752870441754
Degrees of freedom: 2
Expected values: [[30125.39089055   878.47237059  5544.13673886]
 [ 3824.60910945   111.52762941   703.86326114]]
```

Decision rule: -
If the p-value of the chi-square test is greater than α=0.05 then we accept the null hypothesis.

Decision: -
As the p-value of the chi-square test is greater than α=0.05. Hence, we fail to reject (i.e., accept) null hypothesis.

Conclusion: -

The variables *loan* and *target variable (y)* are Independent.

7. Chi-Square Test for Independence for attributes *contact* and *target variable(y)*

:>

The null and alternative hypothesis is:

H_0: The variables **contact** and **target variable (y)** are **Independent**.
H_1: The variables **contact** and **target variable (y)** are **Dependent**.

Chi-Square Test Output: -

```
Test statistic: 863.2690807479079
p-value: 9.481264285590743e-190
Degrees of freedom: 1
Expected values: [[23198.7693503 13349.2306497]
 [ 2945.2306497  1694.7693503]]
```

Decision rule: -
If the p-value of the chi-square test is greater than α=0.05 then we accept the null hypothesis.

Decision: -
As the p-value of the chi-square test is less than α=0.05. Hence, we reject null hypothesis.

Conclusion: -

The variables contact and target variable (y) are Dependent.

8.Chi-Square Test for Independence for attributes `month` and `target variable (y)`

:>

The null and alternative hypothesis is:

H_0: The variables **`month`** and **`target variable (y)`** are **`Independent`**.
H_1: The variables **`month`** and **`target variable (y)`** are **`Dependent`**.

Chi-Square Test Output: -

```
Test statistic: 3101.1493514116773
p-value: 0.0
Degrees of freedom: 9
Expected values: [[ 2335.49422162  5482.02253083   161.49694086  6365.8189764
   4718.905118     484.49082257 12217.86471788  3639.00524425
    637.11430514   505.78712246]
 [  296.50577838   695.97746917    20.50305914   808.1810236
    599.094882      61.50917743  1551.13528212   461.99475575
     80.88569486    64.21287754]]
```

Decision rule: -
If the p-value of the chi-square test is greater than α=0.05 then we accept the null hypothesis.

Decision: -
As the p-value of the chi-square test is less than α=0.05. Hence, we reject
null hypothesis.

Conclusion: -

The variables *`month`* and *`target variable (y)`* are Dependent.

9.Chi-Square Test for Independence for attributes `day_of_week` and `target variable (y)`

:>

The null and alternative hypothesis is:

H_0: The variables **day_of_week** and **target variable (y)** are **Independent**.
H_1: The variables **day_of_week** and **target variable (y)** are **Dependent**.

Chi-Square Test Output: -

```
Test statistic: 26.14493907587197
p-value: 2.9584820052785324e-05
Degrees of freedom: 4
Expected values: [[6945.25580266 7554.8623871  7651.58308245 7178.62775566 7217.67097213]
 [ 881.74419734  959.1376129   971.41691755  911.37224434  916.32902787]]
```

Decision rule: -
If the p-value of the chi-square test is greater than α=0.05 then we accept the null hypothesis.

Decision: -
As the p-value of the chi-square test is less than α=0.05. Hence, we reject
null hypothesis.

Conclusion: -
The variables day_of_week and target variable (y) are Dependent.

10. Chi-Square Test for Independence for attributes poutcome and target variable (y) :>

The null and alternative hypothesis is:

H_0: The variables **poutcome** and **target variable (y)** are **Independent**.
H_1: The variables **poutcome** and **target variable (y)** are **Dependent**.

Chi-Square Test Output: -

```
Test statistic: 4230.5237978319765
p-value: 0.0
Degrees of freedom: 2
Expected values: [[ 3772.99446441 31556.67971254  1218.32582306]
 [  479.00553559  4006.32028746   154.67417694]]
```

Decision rule: -
If the p-value of the chi-square test is greater than α=0.05 then we accept the null hypothesis.

Decision: -
As the p-value of the chi-square test is less than α=0.05. Hence, we reject
null hypothesis.

Conclusion: -

The variables poutcome and targetvariable (y) are Dependent.

Summary Data Frame for Chi-Square Test of Independence: -

	variable_a	variable_b	df	Critical Value	Test Statistic	P-Value	Conclusion
0	job	y	11	19.6751	961.242440	4.189763e-199	Variables are Dependent
1	marital	y	3	7.8147	122.655152	2.068015e-26	Variables are Dependent
2	education	y	7	14.0671	193.105905	3.305189e-38	Variables are Dependent
3	default	y	2	5.9915	406.577515	5.161958e-89	Variables are Dependent
4	housing	y	2	5.9915	5.684496	5.829448e-02	Variables are Independent
5	loan	y	2	5.9915	1.094028	5.786753e-01	Variables are Independent
6	contact	y	1	3.8415	863.269081	9.481264e-190	Variables are Dependent
7	month	y	9	16.9190	3101.149351	0.000000e+00	Variables are Dependent
8	day_of_week	y	4	9.4877	26.144939	2.958482e-05	Variables are Dependent
9	poutcome	y	2	5.9915	4230.523798	0.000000e+00	Variables are Dependent

Normality Test & Mann-Whitney U Test

1.a) Test of Normality for variable Age: -

The null and alternative hypothesis is:

H_0: Variable Age is normally Distributed.
H_1: Variable Age is not normally Distributed.

Shapiro Test Output: -

```
Test statistic: 0.9926980137825012
P-Value: 8.828488610908499e-40
```

Decision rule: -
If the p-value of the Shapiro test is greater than α=0.05 then we accept the null hypothesis.

Decision: -
As the p-value of the Shapiro test is less than α=0.05. Hence, we reject null hypothesis.

Conclusion of Normality: -

Variable Age is not normally Distributed

As the variable is not normally distributed we have to use Mann-Whitney Test.

1.b) Mann Whitney U Test for variable age and Target variable (y).

The null and alternative hypothesis is:

H_0: The two populations are equal for variable age.

H_1: The two populations are not equal for variable age.

Mann-Whitney Test Output: -

```
MannwhitneyuResult(statistic=768810324.0, pvalue=4.177118120619642e-131)
```

Decision rule: -
If the p-value of the Mann-Whitney test is greater than α=0.05 then we accept the null hypothesis.

Decision: -
As the p-value of the Mann-Whitney test is less than α=0.05. Hence, we reject null hypothesis.

Conclusion: -

Two population are different for variable age.

2.a) Test of Normality for variable Duration: -

The null and alternative hypothesis is:

H_0: Variable Duration is normally Distributed.
H_1: Variable Duration is not normally Distributed.

Shapiro Test Output: -

```
Test statistic: 0.9958550930023193
P-Value: 2.5406262681864 61e-31
```

Decision rule: -
If the p-value of the Shapiro test is greater than α=0.05 then we accept the null hypothesis.

Decision: -
As the p-value of the Shapiro test is less than α=0.05. Hence, we reject
null hypothesis.

Conclusion of Normality: -

Variable Duration is not normally Distributed

As the variable is not normally distributed we have to use Mann-Whitney Test.

2.b) Mann Whitney U Test for variable Duration and Target variable (y).

The null and alternative hypothesis is:

H_0: The two populations are equal for variable duration.

H_1: The two populations are not equal for variable duration.

Mann-Whitney Test Output: -

```
MannwhitneyuResult(statistic=762372168.0, pvalue=7.282266821171232e-153)
```

Decision rule: -

If the p-value of the Mann-Whitney test is greater than α=0.05 then we accept the null hypothesis.

Decision: -
As the p-value of the Mann-Whitney test is less than α=0.05. Hence, we reject null hypothesis.

Conclusion: -

Two population are different for variable Duration.

3.a) Test of Normality for variable Campaign: -

The null and alternative hypothesis is:

H_0: Variable Campaign is normally Distributed.
H_1: Variable Campaign is not normally Distributed.

Shapiro Test Output: -

```
Test statistic: 0.8355314135551453
P-Value: 0.0
```

Decision rule: -
If the p-value of the Shapiro test is greater than α=0.05 then we accept the null hypothesis.

Decision: -
As the p-value of the Shapiro test is less than α=0.05. Hence, we reject null hypothesis.

Conclusion of Normality: -

Variable Campaign is not normally Distributed

As the variable is not normally distributed we have to use Mann-Whitney Test.

3.b) Mann Whitney U Test for variable campaign and Target variable (y).

The null and alternative hypothesis is:

H_0: The two populations are equal for variable campaign.

H_1: The two populations are not equal for variable campaign.

Mann-Whitney Test Output: -

```
MannwhitneyuResult(statistic=895985608.0, pvalue=3.0601724250488146e-49)
```

Decision rule: -
If the p-value of the Mann-Whitney test is greater than α=0.05 then we accept the null hypothesis.

Decision: -
As the p-value of the Mann-Whitney test is less than α=0.05. Hence, we reject null hypothesis.

Conclusion: -

Two population are different for variable Campaign.

4.a) Test of Normality for variable pdays: -

The null and alternative hypothesis is:

H_0: Variable Pdays is normally Distributed.

H_1: Variable Pdays is not normally Distributed.

Shapiro Test Output: -

```
Test statistic: 0.18285870552062988
P-Value: 0.0
```

Decision rule: -
If the p-value of the Shapiro test is greater than α=0.05 then we accept the null hypothesis.

Decision: -
As the p-value of the Shapiro test is less than α=0.05. Hence, we reject null hypothesis.

Conclusion of Normality: -

Variable Pdays is not normally Distributed

As the variable is not normally distributed we have to use Mann-Whitney Test.

4.b) Mann Whitney U Test for variable Pdays and Target variable (y).

The null and alternative hypothesis is:

H_0: The two populations are equal for variable pdays.

H_1: The two populations are not equal for variable pdays.

Mann-Whitney Test Output: -

```
MannwhitneyuResult(statistic=1449968804.0, pvalue=0.0)
```

Decision rule: -
If the p-value of the Mann-Whitney test is greater than α=0.05 then we accept the null hypothesis.

Decision: -
As the p-value of the Mann-Whitney test is less than α=0.05. Hence, we reject null hypothesis.

Conclusion: -

Two population are different for variable Pdays.

5.a) Test of Normality for variable Previous: -

The null and alternative hypothesis is:

H_0: Variable Previous is normally Distributed.
H_1: Variable Previous is not normally Distributed.

Shapiro Test Output: -

```
Test statistic: 0.40625739097595215
P-Value: 0.0
```

Decision rule: -
If the p-value of the Shapiro test is greater than α=0.05 then we accept the null hypothesis.

Decision: -
As the p-value of the Shapiro test is less than α=0.05. Hence, we reject null hypothesis.

Conclusion of Normality: -

Variable Previous is not normally Distributed

As the variable is not normally distributed we have to use Mann-Whitney Test.

5.b) Mann Whitney U Test for variable Previous and Target variable (y).

The null and alternative hypothesis is:

H_0: The two populations are equal for variable previous.

H_1: The two populations are not equal for variable previous.

Mann-Whitney Test Output: -

```
MannwhitneyuResult(statistic=231682500.0, pvalue=0.0)
```

Decision rule: -
If the p-value of the Mann-Whitney test is greater than α=0.05 then we accept the null hypothesis.

Decision: -
As the p-value of the Mann-Whitney test is less than α=0.05. Hence, we reject null hypothesis.

Conclusion: -

Two population are different for variable Previous.

6.a) Test of Normality for variable emp.var.rate: -

The null and alternative hypothesis is:

H_0: Variable emp.var.rate is normally Distributed.
H_1: Variable emp.var.rate is not normally Distributed.

Shapiro Test Output: -

```
Test statistic: 0.7650172710418701
P-Value: 0.0
```

Decision rule: -
If the p-value of the Shapiro test is greater than α=0.05 then we accept the null hypothesis.

Decision: -
As the p-value of the Shapiro test is less than α=0.05. Hence, we reject
null hypothesis.

Conclusion of Normality: -

Variable emp.var.rate is not normally Distributed

As the variable is not normally distributed we have to use Mann-Whitney Test.

6.b) Mann Whitney U Test for variable emp.var.rate and Target variable (y).

The null and alternative hypothesis is:

H_0: The two populations are equal for variable emp.var.rate.

H_1: The two populations are not equal for variable emp.var.rate.

Mann-Whitney Test Output: -

```
MannwhitneyuResult(statistic=877042356.0, pvalue=6.1862262834707515e-19)
```

Decision rule: -
If the p-value of the Mann-Whitney test is greater than α=0.05 then we accept the null hypothesis.

Decision: -
As the p-value of the Mann-Whitney test is less than α=0.05. Hence, we reject
null hypothesis.

Conclusion: -

Two population are different for variable

emp.var.rate.

7.a) Test of Normality for variable cons.price.idx: -

The null and alternative hypothesis is:

H_0: Variable cons.price.idx is normally Distributed.
H_1: Variable cons.price.idx is not normally Distributed.

Shapiro Test Output: -

```
Test statistic: 0.935579240322113
P-Value: 0.0
```

Decision rule: -
If the p-value of the Shapiro test is greater than α=0.05 then we accept the null hypothesis.

Decision: -
As the p-value of the Shapiro test is less than α=0.05. Hence, we reject
null hypothesis.

Conclusion of Normality: -

Variable cons.price.idx is not normally Distributed

As the variable is not normally distributed we have to use Mann-Whitney Test.

7.b) Mann Whitney U Test for variable cons.price.idx and Target variable (y).

The null and alternative hypothesis is:

H_0: The two populations are equal for variable cons.price.idx.

H_1: The two populations are not equal for variable cons.price.idx.

Mann-Whitney Test Output: -

```
MannwhitneyuResult(statistic=780607084.0, pvalue=8.374679993757288e-96)
```

Decision rule: -
If the p-value of the Mann-Whitney test is greater than α=0.05 then we accept the null hypothesis.
Decision: -
As the p-value of the Mann-Whitney test is less than α=0.05. Hence, we reject

null hypothesis.

Conclusion: -

Two population are different for variable

Cons.price.idx.

8.a) Test of Normality for variable cons.conf.idx: -

The null and alternative hypothesis is:

H_0: Variable cons.conf.idx is normally Distributed.
H_1: Variable cons.conf.idx is not normally Distributed.

Shapiro Test Output: -

```
Test statistic: 0.9292590618133545
P-Value: 0.0
```

Decision rule: -
If the p-value of the Shapiro test is greater than α=0.05 then we accept the null hypothesis.

Decision: -
As the p-value of the Shapiro test is less than α=0.05. Hence, we reject
null hypothesis.

Conclusion of Normality: -

Variable cons.conf.idx is not normally Distributed

As the variable is not normally distributed we have to use Mann-Whitney Test.

8.b) Mann Whitney U Test for variable cons.conf.idx and Target variable (y).

The null and alternative hypothesis is:

H_0: The two populations are equal for variable cons.conf.idx .

H_1: The two populations are not equal for variable cons.conf.idx .

Mann-Whitney Test Output: -

```
MannwhitneyuResult(statistic=630663816.0, pvalue=0.0)
```

Decision rule: -
If the p-value of the Mann-Whitney test is greater than α=0.05 then we accept the null hypothesis.

Decision: -
As the p-value of the Mann-Whitney test is less than α=0.05. Hence, we reject null hypothesis.

Conclusion: -

Two population are different for variable

Cons.conf.idx.

9.a) Test of Normality for variable euribor3m: -

The null and alternative hypothesis is:

H_0: Variable euribor3m is normally Distributed.
H_1: Variable euribor3m is not normally Distributed.

Shapiro Test Output: -

```
Test statistic: 0.6889737844467163
P-Value: 0.0
```

Decision rule: -
If the p-value of the Shapiro test is greater than α=0.05 then we accept the null hypothesis.

Decision: -
As the p-value of the Shapiro test is less than α=0.05. Hence, we reject null hypothesis.

Conclusion of Normality: -

Variable euribor3m is not normally Distributed

As the variable is not normally distributed we have to use Mann-Whitney Test.

9.b) Mann Whitney U Test for variable euribor3m and Target variable (y).

The null and alternative hypothesis is:

H_0: The two populations are equal for variable euribor3m .

H_1: The two populations are not equal for variable euribor3m .

Mann-Whitney Test Output: -

```
MannwhitneyuResult(statistic=1011502448.0, pvalue=0.0)
```

Decision rule: -
If the p-value of the Mann-Whitney test is greater than α=0.05 then we accept the null hypothesis.
Decision: -
As the p-value of the Mann-Whitney test is less than α=0.05. Hence, we reject null hypothesis.

Conclusion: -

Two population are different for variable

Euribor3m.

10.a) Test of Normality for variable nr.employed: -

The null and alternative hypothesis is:

H_0: Variable nr.employed is normally Distributed.
H_1: Variable nr.employed is not normally Distributed.

Shapiro Test Output: -

```
Test statistic: 0.8143253922462463
P-Value: 0.0
```

Decision rule: -
If the p-value of the Shapiro test is greater than α=0.05 then we accept the null hypothesis.

Decision: -
As the p-value of the Shapiro test is less than α=0.05. Hence, we reject null hypothesis.

Conclusion of Normality: -

Variable nr.employed is not normally Distributed

As the variable is not normally distributed we have to use Mann-Whitney Test.

10.b) Mann Whitney U Test for variable nr.employed and Target variable (y).

The null and alternative hypothesis is:

H_0: The two populations are equal for variable nr.employed .

H_1: The two populations are not equal for variable nr.employed .

Mann-Whitney Test Output: -

```
MannwhitneyuResult(statistic=1011648640.0, pvalue=0.0)
```

Decision rule: -
If the p-value of the Mann-Whitney test is greater than α=0.05 then we accept the null hypothesis.
Decision: -
As the p-value of the Mann-Whitney test is less than α=0.05. Hence, we reject null hypothesis.

Conclusion: -

Two population are different for variable nr.employed.

Predictive Modelling:

Logit Model Summary:

Logit Regression Results

Dep. Variable:	y	**No. Observations:**	32950
Model:	Logit	**Df Residuals:**	32901
Method:	MLE	**Df Model:**	48
Date:	Thu, 09 Sep 2021	**Pseudo R-squ.:**	0.4473
Time:	22:53:09	**Log-Likelihood:**	-6410.9
converged:	False	**LL-Null:**	-11599.
Covariance Type:	nonrobust	**LLR p-value:**	0.000

	coef	std err	z	P>|z|	[0.025	0.975]
const	-3.7327	0.166	-22.421	0.000	-4.059	-3.406
age	-0.0541	0.029	-1.871	0.061	-0.111	0.003
duration	1.8577	0.032	58.185	0.000	1.795	1.920
campaign	-0.0921	0.025	-3.640	0.000	-0.142	-0.042
previous	-0.1355	0.025	-5.475	0.000	-0.184	-0.087
cons.price.idx	0.0663	0.030	2.247	0.025	0.008	0.124
cons.conf.idx	0.1258	0.028	4.512	0.000	0.071	0.180
nr.employed	-1.1888	0.035	-33.650	0.000	-1.258	-1.120
job_blue-collar	-0.2335	0.090	-2.588	0.010	-0.410	-0.057
job_entrepreneur	-0.0702	0.138	-0.508	0.611	-0.341	0.201
job_housemaid	0.0851	0.165	0.515	0.607	-0.239	0.409
job_management	-0.0994	0.097	-1.020	0.308	-0.290	0.092
job_retired	0.3881	0.117	3.313	0.001	0.159	0.618
job_self-employed	-0.1164	0.134	-0.867	0.386	-0.379	0.147
job_services	-0.2204	0.098	-2.250	0.024	-0.412	-0.028
job_student	0.1592	0.136	1.173	0.241	-0.107	0.425
job_technician	0.0143	0.081	0.176	0.860	-0.145	0.173
job_unemployed	0.0403	0.150	0.268	0.789	-0.255	0.335

job_unknown	0.1548	0.273	0.566	0.571	-0.381	0.691
marital_married	0.0214	0.078	0.274	0.784	-0.132	0.175
marital_single	0.0392	0.090	0.437	0.662	-0.137	0.215
marital_unknown	0.1354	0.440	0.307	0.759	-0.728	0.998
education_basic.6y	0.1335	0.135	0.987	0.324	-0.132	0.399
education_basic.9y	-0.0196	0.106	-0.184	0.854	-0.228	0.189
education_high.school	-0.0064	0.104	-0.062	0.951	-0.210	0.197
education_illiterate	1.5637	0.858	1.822	0.068	-0.118	3.246
education_professional.course	0.0316	0.115	0.274	0.784	-0.194	0.258
education_university.degree	0.1664	0.104	1.594	0.111	-0.038	0.371
education_unknown	0.1294	0.136	0.948	0.343	-0.138	0.397
default_unknown	-0.2864	0.074	-3.884	0.000	-0.431	-0.142
default_yes	-20.6579	2.22e+05	-9.3e-05	1.000	-4.35e+05	4.35e+05
housing_yes	-0.0243	0.047	-0.515	0.607	-0.117	0.068
loan_unknown	-0.0324	0.159	-0.204	0.838	-0.343	0.278
loan_yes	-0.0158	0.065	-0.244	0.807	-0.143	0.111
contact_telephone	-0.2610	0.081	-3.216	0.001	-0.420	-0.102
month_aug	0.6405	0.122	5.241	0.000	0.401	0.880
month_dec	0.5057	0.231	2.191	0.028	0.053	0.958
month_jul	0.7441	0.113	6.584	0.000	0.523	0.966
month_jun	0.7519	0.108	6.951	0.000	0.540	0.964
month_mar	1.8916	0.143	13.184	0.000	1.610	2.173
month_may	-0.7693	0.087	-8.801	0.000	-0.941	-0.598
month_nov	0.1385	0.111	1.248	0.212	-0.079	0.356
month_oct	0.5776	0.145	3.988	0.000	0.294	0.861
month_sep	0.2092	0.159	1.312	0.189	-0.103	0.522
day_of_week_mon	-0.0527	0.075	-0.701	0.483	-0.200	0.095
day_of_week_thu	0.0612	0.073	0.842	0.400	-0.081	0.204
day_of_week_tue	0.0970	0.075	1.290	0.197	-0.050	0.244
day_of_week_wed	0.1115	0.075	1.492	0.136	-0.035	0.258
poutcome_success	1.9316	0.101	19.032	0.000	1.733	2.130

Statistical Analysis of the features:

Significant features affecting the model:

The following variables have a p-value of Wald test statistic less than 0.05.

- duration
- campaign
- previous
- cons.price.idx
- cons.conf.idx
- nr.employed
- job_blue-collar
- job_retired
- job_services
- default_unknown
- contact_telephone
- month_aug
- month_dec
- month_jul
- month_jun
- month_mar
- month_may
- month_oct
- poutcome_success

Odds Value for each feature sorted in descending order:

	Odds value
poutcome_success	6.90
month_mar	6.63
duration	6.41
education_illiterate	4.78
month_jun	2.12
month_jul	2.10
month_aug	1.90

	Odds value
month_oct	1.78
month_dec	1.66
job_retired	1.47
month_sep	1.23
education_university.degree	1.18
job_unknown	1.17
job_student	1.17
month_nov	1.15
education_unknown	1.14
marital_unknown	1.14
education_basic.6y	1.14
cons.conf.idx	1.13
day_of_week_wed	1.12
day_of_week_tue	1.10
job_housemaid	1.09
cons.price.idx	1.07
day_of_week_thu	1.06
marital_single	1.04
job_unemployed	1.04
education_professional.course	1.03
marital_married	1.02
job_technician	1.01
education_high.school	0.99
loan_yes	0.98
education_basic.9y	0.98
housing_yes	0.98

	Odds value
loan_unknown	0.97
day_of_week_mon	0.95
age	0.95
job_entrepreneur	0.93
job_management	0.91
campaign	0.91
job_self-employed	0.89
previous	0.87
job_services	0.80
job_blue-collar	0.79
contact_telephone	0.77
default_unknown	0.75
month_may	0.46
nr.employed	0.30
default_yes	0.00

Thus, from the above observations, we can understand which feature is highly related with the target variable y. Greater the odds value, higher is the relation.

Model Evaluation : McFadden's R^2 value(Pseudo R^2):

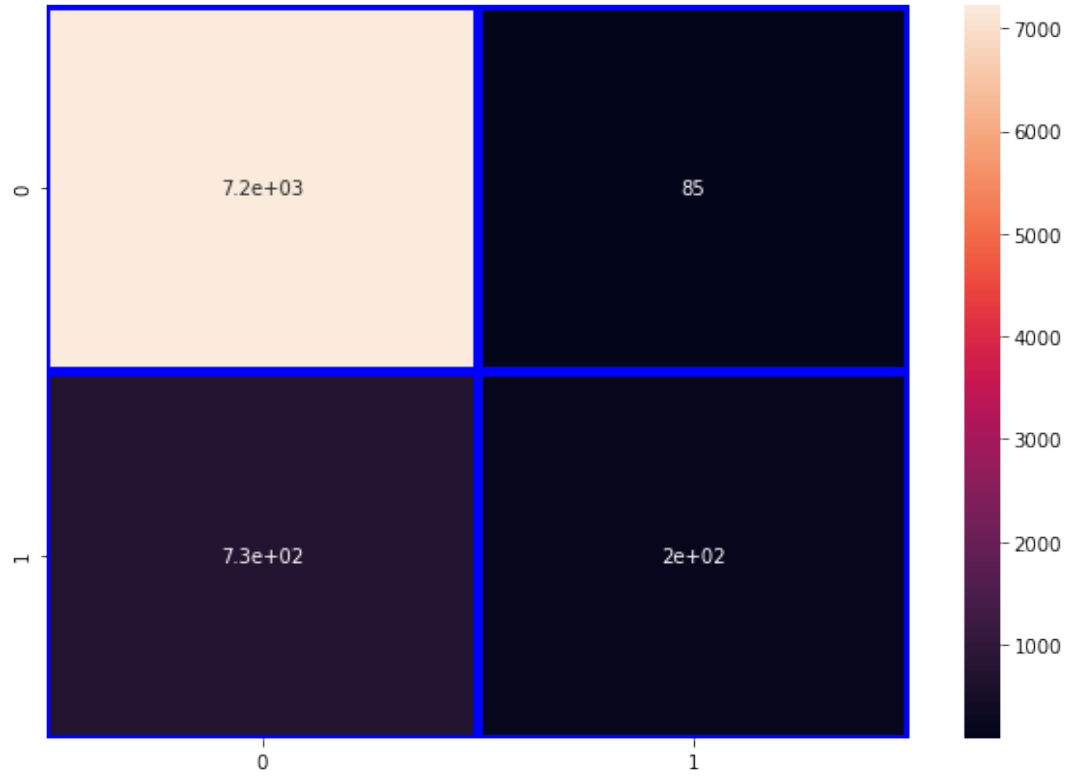
Logit model accuracy : 0.44731193954881476

Optimal value threshold for full model using the Youden's index:

```
      TPR       FPR     thres       YI
0  0.898707  0.151710  0.110695  0.746997
1  0.899784  0.152804  0.109174  0.746980
2  0.900862  0.153899  0.108453  0.746963
3  0.897629  0.151300  0.111410  0.746330
4  0.896552  0.150616  0.111762  0.745936
----------------------------------------
```

Optimal threshold for probability: 0.75

Graphical Representation of Confusion Matrix using Youden's Index:



Confusion Matrix for Logit Model using Youden's Index :

[[7225 85]
 [732 196]]

Logistic Regression Using Scikit-Learn:

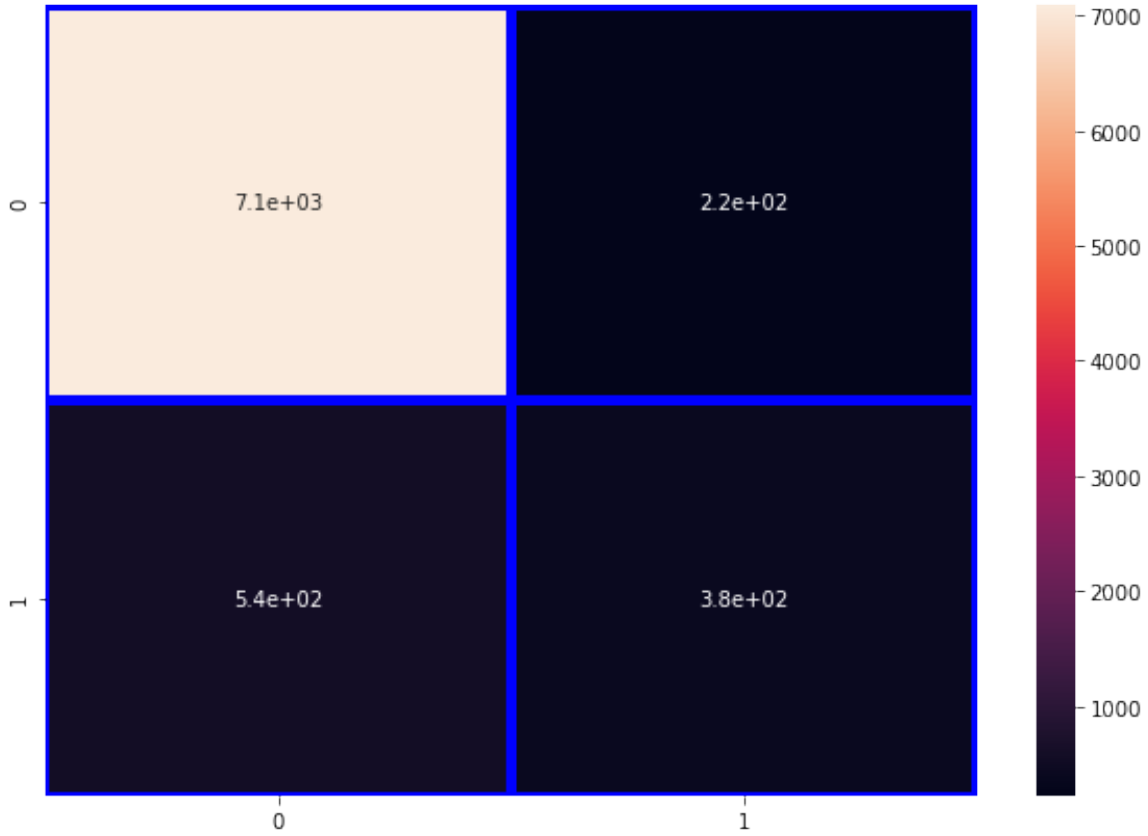
Model Performance Evaluation:

1. Confusion Matrix for Logistic Regression Model:

Confusion Matrix form Logistic Regression Model :

[[7089 221]
 [544 384]]

2. Graphical Representation of Confusion Matrix:



3. Sensitivity and Specificity:

Sensitivity of the Logistic Regression Model:

0.41379310344827586

Specificity of the Logistic Regression Model:

0.9697674418604652

While building a model on Logistic Regression using sklearn we have come across the above confusion matrix which is telling us about the sensitivity and the specificity of the Regression Model. The sensitivity is coming around to be 41 percent and the specificity is around 97 percent.

***4. Accuracy Analysis**:*

The Accuracy score of the test data for Logistic Regression model :
0.9071376547705754

The Accuracy score of the train data for Logistic Regression model :
0.9129286798179059

By doing the accuracy analysis of the Logistic Regression model, the model accuracy or the test accuracy is coming around 90.71 percent. The accuracy score for the train data is coming around 91.29 percent. So it states that it is a generalized model with no over fitting.

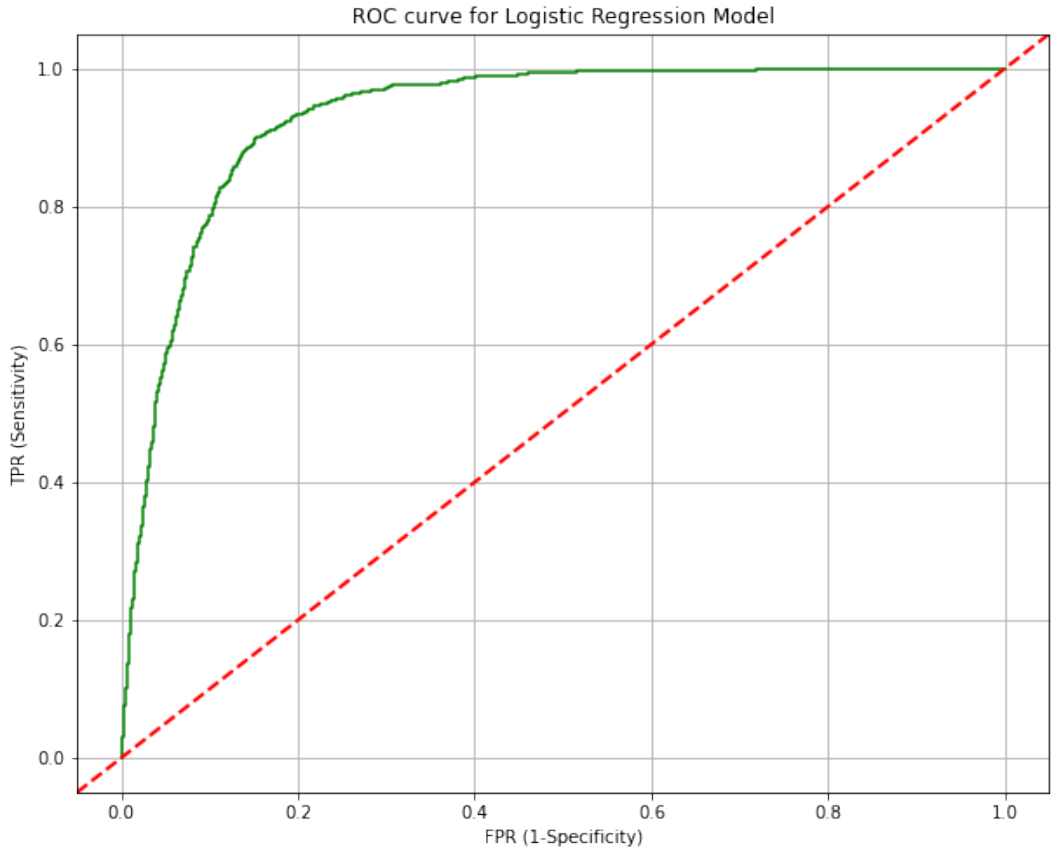
***5. Logistic Regression model classification report**:*

Logit model classification report:
---

	precision	recall	f1-score	support
0	0.93	0.97	0.95	7310
1	0.63	0.41	0.50	928
accuracy			0.91	8238
macro avg	0.78	0.69	0.72	8238
weighted avg	0.90	0.91	0.90	8238

The F1_score for weighted average is coming around 90 percent, which states that our model is 90 percent accurate.

6. ROC Curve for Logistic Regression Model:



Here in the Roc Curve we are comparing the True Positive rate and the False Positive rate. The Greater the area under the curve, the better the model.

7. ROC AUC Score:

The ROC AUC score of the Logistic Regression model:

ROC AUC : 0.9315524906835226

Here we have calculated the ROC AUC score. The score is coming around 93 percent which states that the 93 percent of the area is covered under the curve by the logistic Regression model.

***8. Cross Entropy**:*

The Cross Entropy score of the Logistic Regression model

Cross Entropy : 3.2073732281061846

The cross entropy for the Logistic Regression model is 3.20

Inferences for Logistic Regression Model:

- Cross Entropy for Logistic Regression Model is 3.20
- ROC AUC Score for the Logistic Regression Model is 93.15
- The Model Accuracy for the Logistic Regression Model is coming out to be around 91%.
- f1 weighted avg for Logistic Regression Model is around 90%.
- Specificity : 97%
- Sensitivity : 41%

K-Nearest Neighbors Classification Model using Scikit-learn:

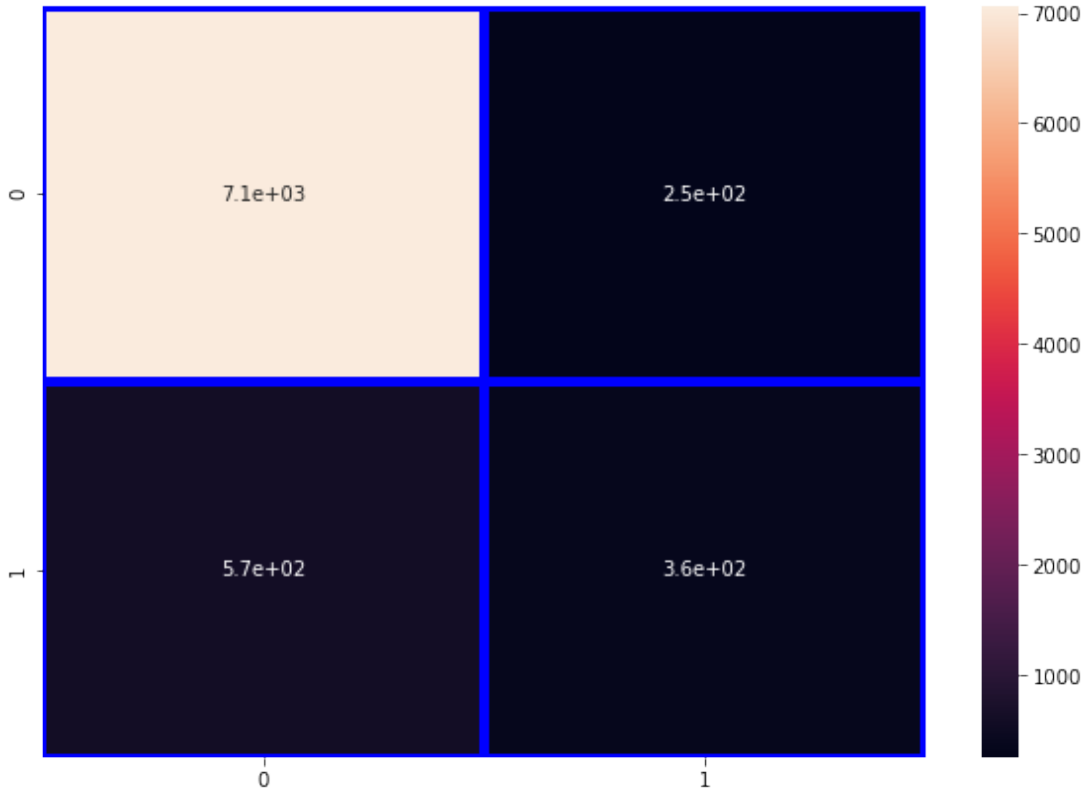
Model Performance Evaluation:

1. Confusion Matrix for K-Nearest Neighbors Model:

Confusion Matrix for K-Nearest Neighbors Model :

[[7059 251]
 [572 356]]

2. Graphical Representation of Confusion Matrix:



The above plot is the confusion matrix for the K Nearest Neighors model.

3.Sensitivity and Specificity:

Sensitivity of the KNN Model:

0.38362068965517243

Specificity of the KNN Model:

0.9656634746922025

While building a model K-Nearest Neighbor using sklearn we have come across the above confusion matrix which is telling us about the sensitivity and the specificity of the KNN Model. The sensitivity is coming around to be 38.36 percent and the specificity is around 96.56 percent.

4. Accuracy Analysis:

The Accuracy score of the test data for K-Nearest Neighbors Model :
0.9000971109492595

The Accuracy score of the train data for K-Nearest Neighbors Model :
0.9298634294385433

By doing the accuracy analysis of the KNN model, the model accuracy or the test accuracy is coming around 90 percent. The accuracy score for the train data is coming around 92.98 percent. So it states that it is a generalized model with no over fitting.

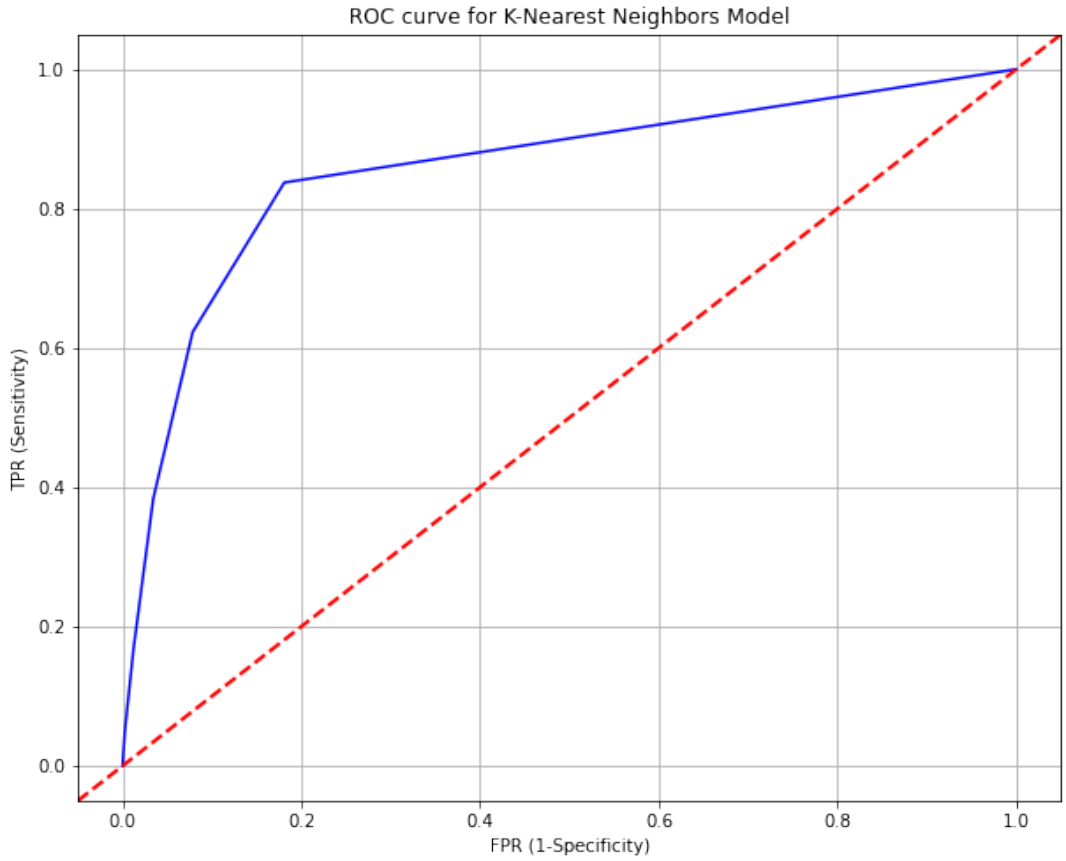
5. K-Nearest Neighbors Model classification report:

K-Nearest Neighbors Model classification report:
---

	precision	recall	f1-score	support
0	0.93	0.97	0.94	7310
1	0.59	0.38	0.46	928
accuracy			0.90	8238
macro avg	0.76	0.67	0.70	8238
weighted avg	0.89	0.90	0.89	8238

The F1_score for weighted average is coming around 89 percent, which states that our model is 89 percent accurate.

6. ROC Curve for K-Nearest Neighbors Model:



Here in the Roc Curve we are comparing the True Positive rate and the False Positive rate. The Greater the area under the curve, the better the model.

7. ROC AUC Score:

The ROC AUC score of the K-Nearest Neighbors Model

ROC AUC : 0.8566297348931553

The ROC AUC score for the model is coming around to be 85.66 percent. This states th at the model is able to cover 85 percent of area under the Curve.
8. Cross Entropy:

The Cross Entropy score of K-Nearest Neighbors Model

Cross Entropy : 3.450547908711638

Inferences for K-Nearest Neighbors Classification Model:

- Cross Entropy for K-Nearest Neighbors Model is 3.45

- ROC AUC Score for the K-Nearest Neighbors Model is 85.66
- The Model Accuracy for the K-Nearest Neighbors Model is coming out to be around 90%.
- f1 weighted avg for the K-Nearest Neighbors Model is around 89%.
- Specificity : 97%
- Sensitivity : 38%

In the above algorithm we have created a model using default parameters. The test accuracy can be improved by doing Hyper parameter tuning with a given set of hyper parameters. Using Grid Search CV we can get the best parameters for our model that will improve the accuracy if necessary.

Decision Tree Classification Model using Scikit-learn:

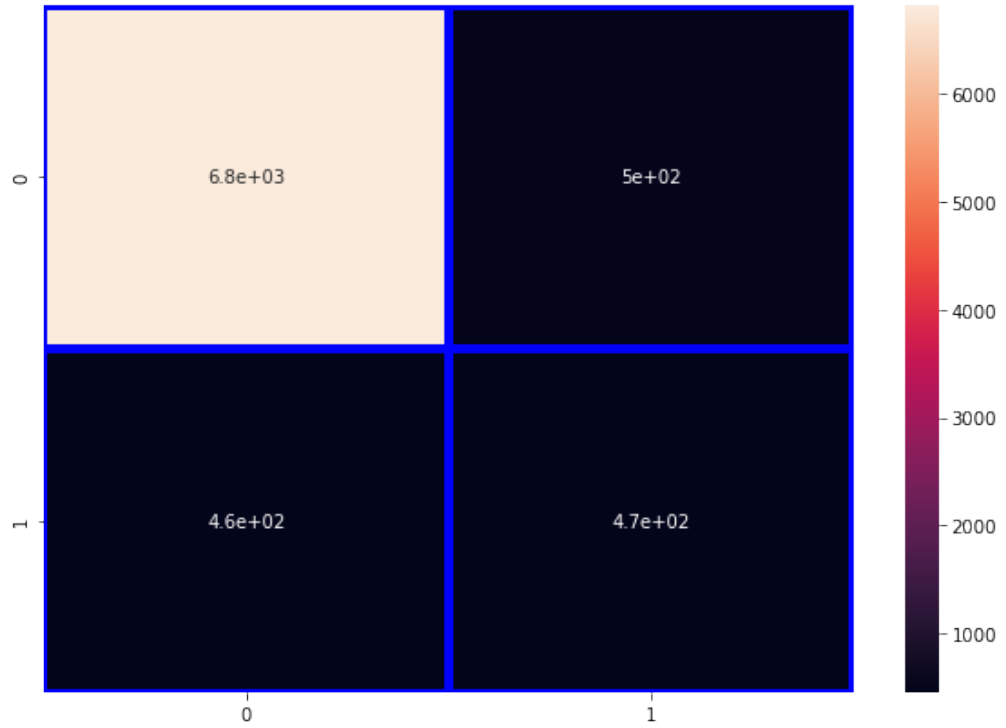
Model Performance Evaluation:

1.Confusion Matrix for Decision Tree classification:

Confusion Matrix for Decision Tree Model :

[[6812 498]
[458 470]]

2.Graphical Representation of Confusion Matrix:



The above plot is the confusion matrix for the Decision Tree model.

3.Sensitivity and Specificity:

Sensitivity of the Decision Tree Model:

0.5064655172413793

Specificity of the Decision Tree Model:

0.93187414500684

While building a Decision Tree Model using sklearn we have come across the above confusion matrix which is telling us about the sensitivity and the specificity of the Decision Tree Model. The sensitivity is coming around to be 50.64 percent and the specificity is around 93.18 percent.

4. Accuracy Analysis:

The Accuracy score of test data for the Decision Tree Model:

0.8839524156348628

The Accuracy score of the train data for the Decision Tree Model:
1.0

By doing the accuracy analysis of the decision tree model, the model accuracy or the test accuracy is coming around 88.39 percent. The accuracy score for the train data is coming around 100 percent. So it states that the model is over fitting. We can overcome the problem of over fitting by doing decision tree pruning or hyper parameter tuning with a given set of parameters using Grid Search CV that will help us getting the best parameters for our model building.

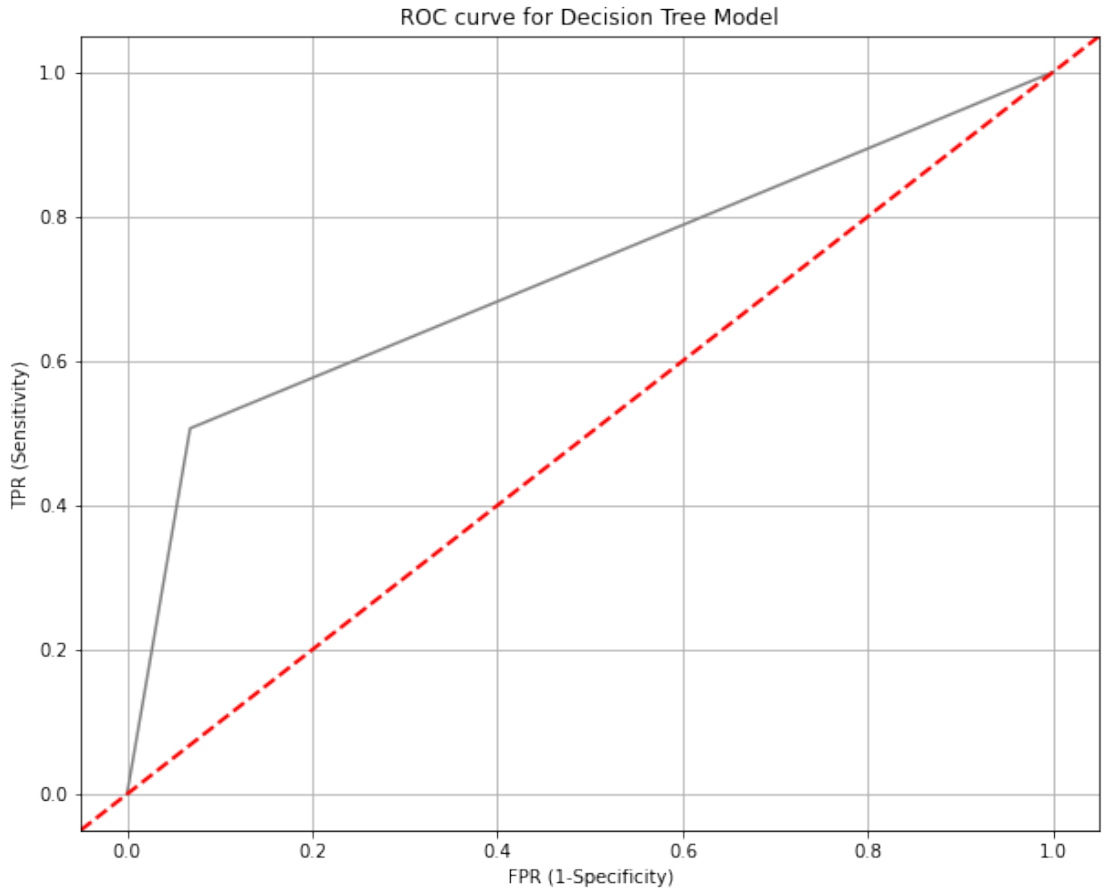
5. Decision Tree Model classification report:

Decision Tree Model classification report:
---

	precision	recall	f1-score	support
0	0.94	0.93	0.93	7310
1	0.49	0.51	0.50	928
accuracy			0.88	8238
macro avg	0.71	0.72	0.72	8238
weighted avg	0.89	0.88	0.89	8238

The F1_score for weighted average is coming around 89 percent, which states that our model is 89 percent accurate.

6.ROC Curve for Decision Tree Model:



Here in the Roc Curve we are comparing the True Positive rate and the False Positive rate.
The Greater the area under the curve, the better the model.

7.ROC AUC Score:

The ROC AUC score of the Decision Tree Model

ROC AUC : 0.7191698311241097

The ROC AUC score for the model is coming around to be 71.91 percent. This states that the model is able to cover 72 percent of area under the
Curve. So there is definitely a scope of improvement.

***8.Cross Entropy**:*

The Cross Entropy score of Decision Tree Model

Cross Entropy : 4.008189904473762

Inferences for Decision Tree Classification Model:

- The Cross entropy for the Decision Tree model is 4.01
- ROC AUC Score for the Decision Tree Model is 71.91
- The Model Accuracy for the Decision Tree Model is coming out to be around 89%.
- f1 weighted avg for the Decision Tree Model is around 89%.
- Specificity : 93%
- Sensitivity : 51%

Naïve Bayes Classification Model using Scikit-learn:

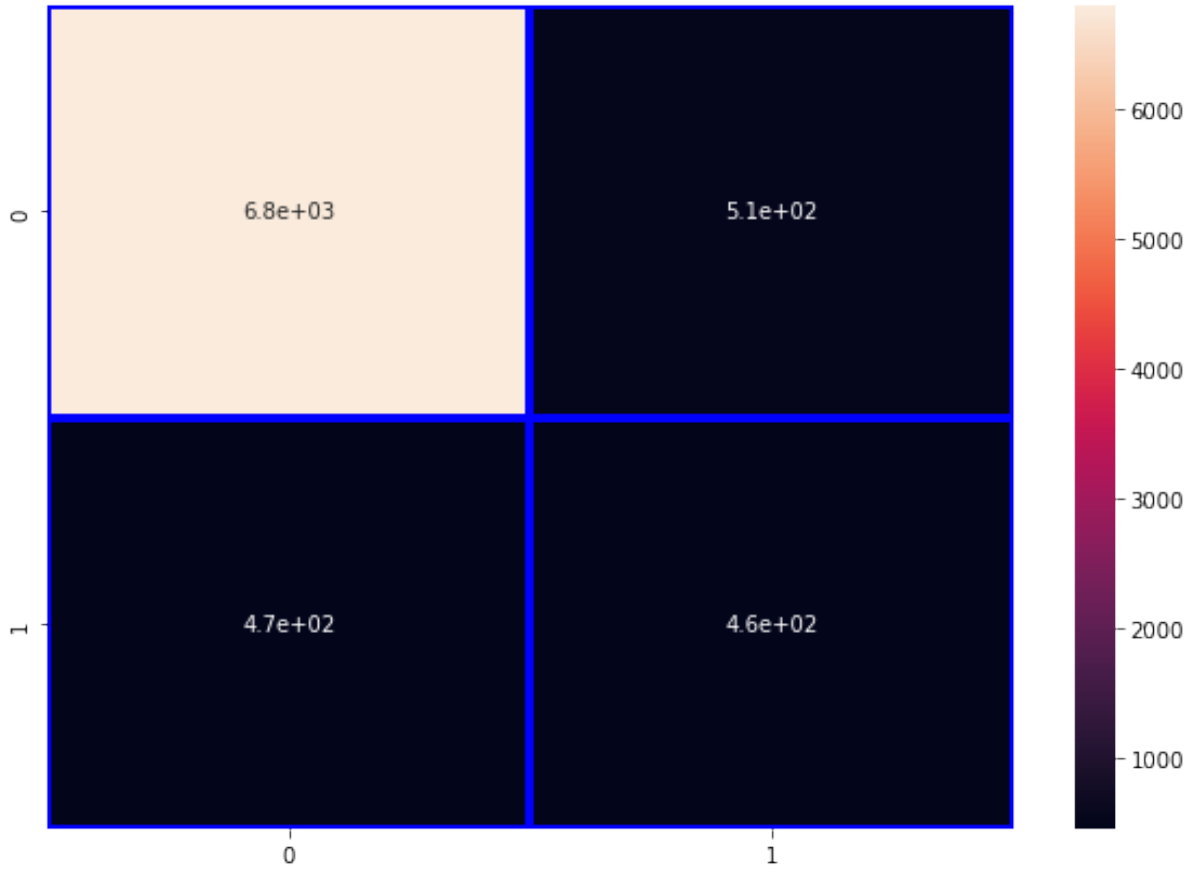
Model Performance Evaluation:

1.Confusion Matrix for Naïve Bayes classification:

Confusion Matrix for Naïve Bayes Model :

[[6801 509]
 [472 456]]

2.Graphical Representation of Confusion Matrix:



The above plot is the confusion matrix for the Naïve Bayes model.

3.Sensitivity and Specificity:

Sensitivity of the Naive Bayes Model:

0.49137931034482757

Specificity of the naive Bayes Model:

0.9303693570451437

While building a Naïve Bayes model we have come across the above confusion matrix which is telling us about the sensitivity and the specificity of the Naïve Bayes Model. The sensitivity is coming around to be 49.13% percent and the specificity is around 93.03% percent.

4. Accuracy Analysis:

The Accuracy score of test data for the Naive Bayes Model:
0.836489439184268

The Accuracy score of the train data for the Naive Bayes Model:
0.8401820940819423

By doing the accuracy analysis of the Naïve Bayes model, the model accuracy or the test accuracy is coming around 83.64 percent. The accuracy score for the train data is coming around 84.01 percent. So it states that the model is over fitting. We can overcome the problem of over fitting by hyper parameter tuning with a given set of parameters using Grid Search CV that will help us getting the best parameters for our model building.

5. Naïve Bayes Model classification report:

Naive Bayes Model classification report for train data:
---

	precision	recall	f1-score	support
0	0.94	0.87	0.91	29238
1	0.37	0.57	0.45	3712
accuracy			0.84	32950
macro avg	0.65	0.72	0.68	32950
weighted avg	0.88	0.84	0.85	32950

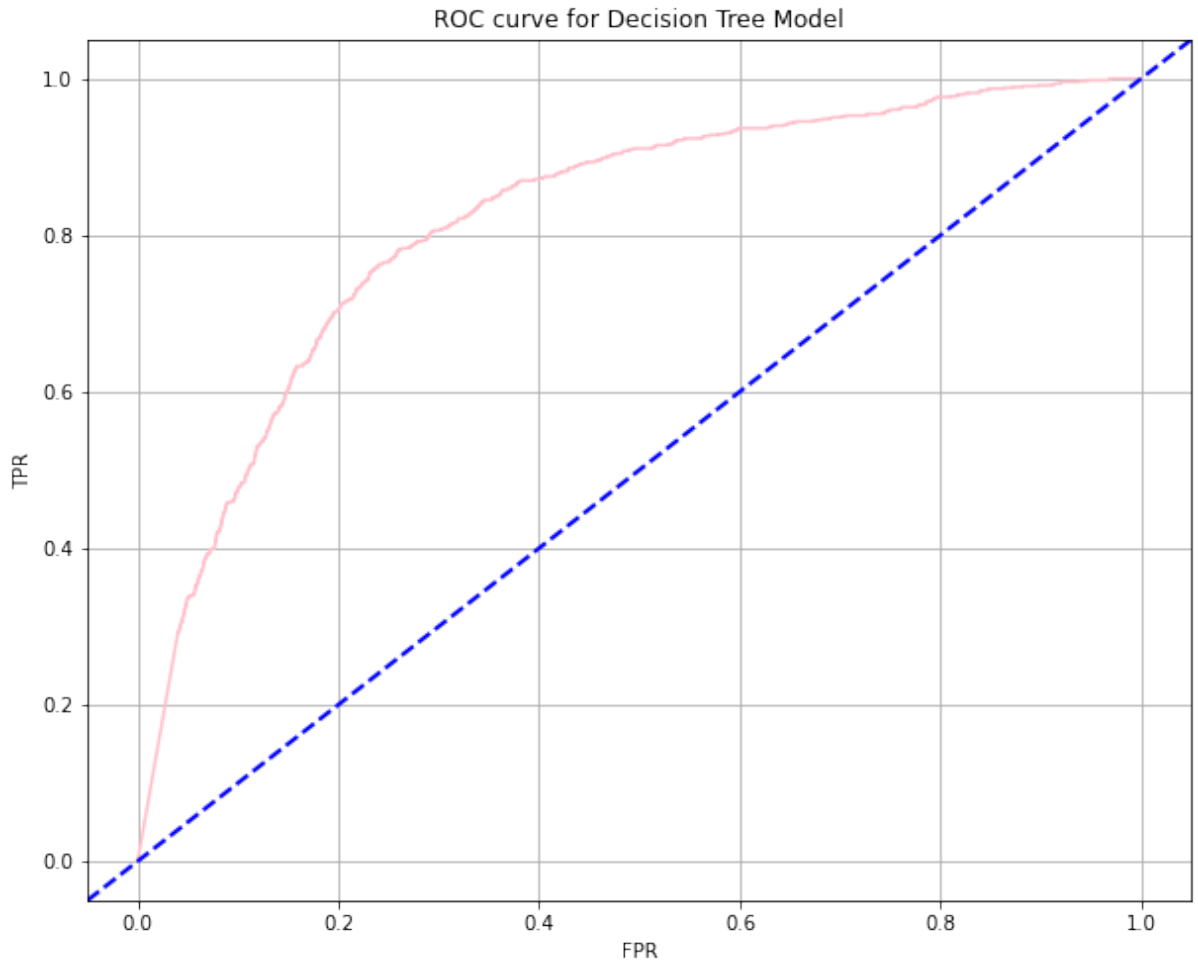
Naive Bayes Model classification report for test data:
---

	precision	recall	f1-score	support
0	0.94	0.87	0.90	7310
1	0.35	0.54	0.43	928
accuracy			0.84	8238
macro avg	0.65	0.71	0.67	8238

weighted avg 0.87 0.84 0.85 8238

The F1_score for weighted average is coming around 85 percent, which states that our model is 85 percent accurate.

6.ROC Curve for Decision Tree Model:



Here in the Roc Curve we are comparing the True Positive rate and the False Positive rate.
The Greater the area under the curve, the better the model.

7.ROC AUC Score:

The ROC AUC score of the Naïve Bayes Model

ROC AUC : 0.8193490111561865

The ROC AUC score for the model is coming around to be 81.93 percent. This states that the model is able to cover 81.93 percent of area under the
Curve. So there is definitely a scope of improvement.

***8.Cross Entropy**:*

The Cross Entropy score of Naive Bayes Model

Cross Entropy : 5.647544092398385

9. Bias Error and Variance Error:

Mean Score : 0.8394233687405158
Bias error : 16.05766312594842
Variance error : 0.7219069070443396

Inferences for Naïve Bayes Model:

- The Cross entropy for the Naïve Bayes model is 5.64
- ROC AUC Score for the Naïve Bayes Model is 81.93.
- The Model Accuracy for the Naïve Bayes Model is coming out to be around 84%.
- f1 weighted avg for the Decision Tree Model is around 85%.
- Specificity : 49.1
- Sensitivity :93.03
- The mean score is 0.83, Bias Error is 16.05 and variance error is 0.72.

Inferences based on model comparison:

1. Overall Accuracy Score:

1. Logistic Regression Model has the highest overall accuracy of about 91%.

2. Naïve Bayes Model yeilds the lowest overall model accuracy of about 84%.

2. Overfitting/Underfitting:

1. All the classification models exhibit overfitting of the trained data with respect to the test data.

2. The model accuracy for train data and test data for both Logistic Regression Mod el and K-Nearest Neighbors Model has very less overfitting.

3. As observed,the model accuracy for train data and test data for the Decision Tree Model has a considerably high difference in accuracies which can be considered a h igh overfitting condition in comparison to other models.

3. f1 score weighted avg:

1. The Logistic Regression Model has the highest weighted harmonic mean betwee n precison and recall of about 90% based on the classification report.

2. Both K-Nearest Neighbors Model and Descision Tree Model have nearly similar weighted harmonic mean between precison and recall of about 89%.

3. We can further look at the recall values of the positive class and negative class to get more insights on the specificity and sensitivity.

4. Sensitivity and Specificity

1.Logistic Regression Model:

- Sensitivity – 41.37%
- Specificity – 96.97%

2. K-Nearest Neighbors Model:

- Sensitivity – 38%
- Specificity – 96.56%

3.Decision Tree Model :

- Sensitivity – 50.64%
- Specificity – 93.18%

4.Naive Bayes Model:

- Sensitivity –49%
- Specificity – 93%

5. ROC AUC Score:

1. The Logistic Regression Model has the maximum area under the ROC curve wit h a ROC AUC Score of 93.15.

2. The Decision Tree Model has the minimum area under the ROC curve with a RO C AUC Score of 71.91.

6. Cross Entropy:

1. Minimum Cross Entropy score for Logistic Regression model:

H_logistic(y) = 3.20

2. Maximum Cross Entropy score for Naive Bayes Model:

H_NaiveBayes(y) = 5.65

Tabular representation of derived Inferences:

Model Comparison:

	Overall Accuracy Score	Accuracy for train data	Accuracy for test data	Specificity	Sensitivity	f1 score weighted avg	ROC AUC Score	Cross Entropy
Logistic Regresion Model	91%	91.24%	90.82%	97%	42%	90%	93.28	3.17
K-Nearest Neighbors Model	90%	93.01%	90.28%	97%	39%	89%	85.86	3.35
Descision Tree Model	89%	100%	88.94%	94%	51%	89%	72.51	3.82
Naive Bayes	84%	84%	84%	93%	49%	85%	81.93	5.64

Model Optimization:

1.Hyperparameter Tunning Base models to find out best parameters for further optimization:

Each base model has been chosen and a set of parameters has been considered for fine tuning. Using these set of parameters we have used Grid Search Cross Validation technique for Hyperparameter tunning the models where the Cross Validation method considered is 10-Fold Cross Validation.

The Logistic Regression model, KNN Model and Decision Tree model will be considere d for tuning based on the above analysis.

- Logistic Regression model : {'max_iter': 110, 'penalty': 'l2'}
- KNN Model : {'algorithm': 'auto', 'leaf_size': 30, 'n_neighbors': 8, 'weights': 'distance'}
- Decision Tree Model : {'criterion': 'entropy',
  'max_depth': 10,
  'min_samples_split': 20,
  'splitter': 'best'}

2.Refitting the Tunned Based models to compare the performances:

Overall accuracy of LogisticRegression(max_iter=110) for resampled train data : 91.0
Overall accuracy of LogisticRegression(max_iter=110) for resampled test data : 91.0

Confusion Matrix for LogisticRegression(max_iter=110) Model :
[[7089 221]
[545 383]]

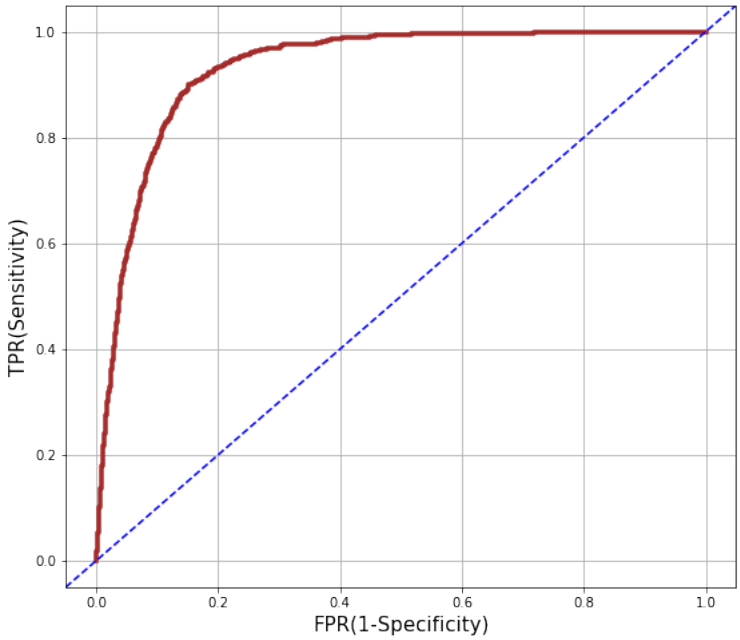
Sensitivity : 41.27
Specificity : 96.98

Classification report of LogisticRegression(max_iter=110) for resampled train data :

	precision	recall	f1-score	support
0	0.93	0.97	0.95	29238
1	0.67	0.45	0.54	3712
accuracy			0.91	32950
macro avg	0.80	0.71	0.75	32950
weighted avg	0.90	0.91	0.91	32950

Classification report of LogisticRegression(max_iter=110) for resampled test data :

	precision	recall	f1-score	support
0	0.93	0.97	0.95	7310
1	0.63	0.41	0.50	928
accuracy			0.91	8238
macro avg	0.78	0.69	0.72	8238
weighted avg	0.90	0.91	0.90	8238



AUC Score of LogisticRegression(max_iter=110) for resampled test data : 0.931534506344639

The Cross Entropy score of LogisticRegression(max_iter=110) Model : 3.21156584480865

Mean Score : 0.9121092564491654
Bias error : 8.789074355083459
Variance error : 0.359353164304254

Overall accuracy of DecisionTreeClassifier(criterion='entropy', max_depth=10, min_samples_split=20) for resampled train data : 93.0
Overall accuracy of DecisionTreeClassifier(criterion='entropy', max_depth=10, min_samples_split=20) for resampled test data : 90.0

Confusion Matrix for DecisionTreeClassifier(criterion='entropy', max_depth=10, min_samples_split=20) Model :
[[7014 296]
[498 430]]

Sensitivity : 46.34
Specificity : 95.95

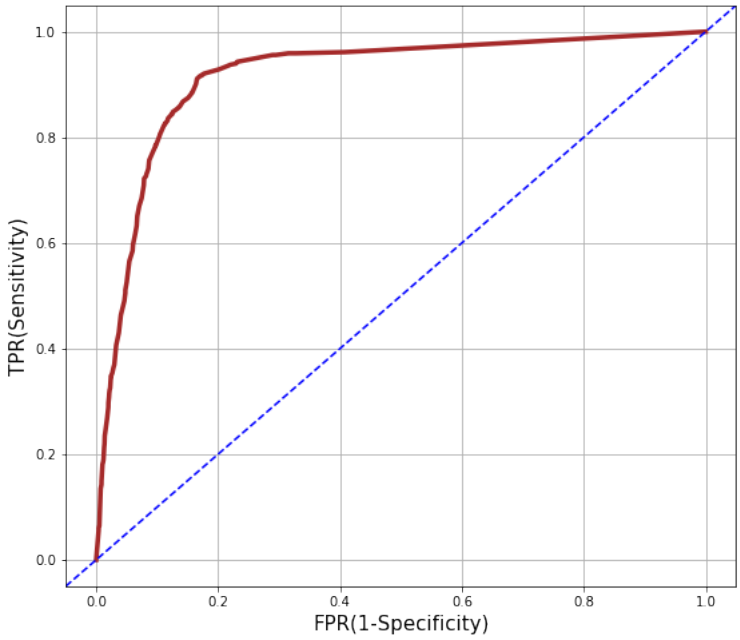
Classification report of DecisionTreeClassifier(criterion='entropy', max_depth=10, min_samples_split=20) for resampled train data :
---

	precision	recall	f1-score	support
0	0.95	0.97	0.96	29238
1	0.73	0.60	0.66	3712
accuracy			0.93	32950
macro avg	0.84	0.79	0.81	32950
weighted avg	0.93	0.93	0.93	32950

Classification report of DecisionTreeClassifier(criterion='entropy', max_depth=10, min_samples_split=20) for resampled test data :

---

	precision	recall	f1-score	support
0	0.93	0.96	0.95	7310
1	0.59	0.46	0.52	928
accuracy			0.90	8238
macro avg	0.76	0.71	0.73	8238
weighted avg	0.90	0.90	0.90	8238



AUC Score of DecisionTreeClassifier(criterion='entropy', max_depth=10, min_samples_split=20) for resampled test data :
0.9159395048115477

The Cross Entropy score of DecisionTreeClassifier(criterion='entropy', max_depth=10, min_samples_split=20) Model : 3.3289663921338217

Mean Score : 0.911380880121396
Bias error : 8.861911987860404
Variance error : 0.2634704021958375

Overall accuracy of KNeighborsClassifier(n_neighbors=8, weights='distance') for resampled train data : 100.0
Overall accuracy of KNeighborsClassifier(n_neighbors=8, weights='distance') for resampled test data : 90.0

Confusion Matrix for KNeighborsClassifier(n_neighbors=8, weights='distance') Model :
[[7074 236]
[581 347]]

Sensitivity : 37.39
Specificity : 96.77

Classification report of KNeighborsClassifier(n_neighbors=8, weights='distance') for resampled train data :

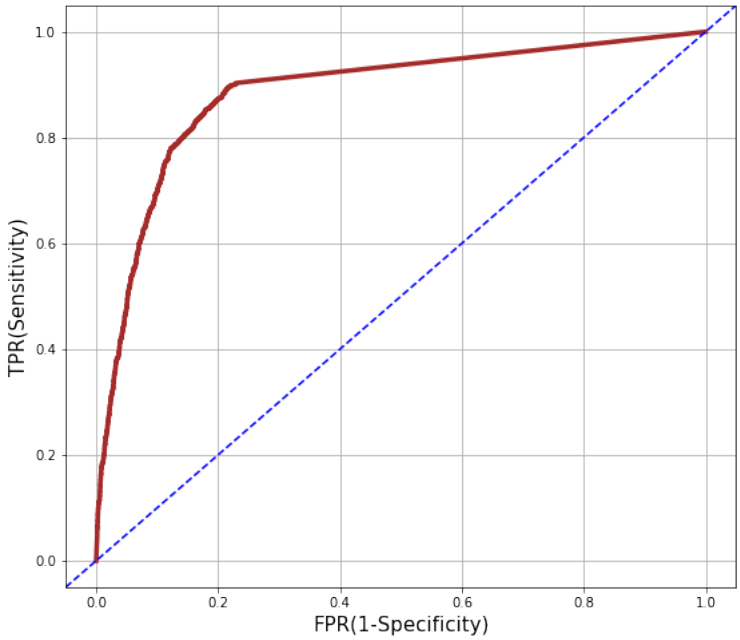
---

	precision	recall	f1-score	support
0	1.00	1.00	1.00	29238
1	1.00	1.00	1.00	3712
accuracy			1.00	32950
macro avg	1.00	1.00	1.00	32950
weighted avg	1.00	1.00	1.00	32950

Classification report of KNeighborsClassifier(n_neighbors=8, weights='distance') for resampled test data :

---

	precision	recall	f1-score	support
0	0.92	0.97	0.95	7310
1	0.60	0.37	0.46	928
accuracy			0.90	8238
macro avg	0.76	0.67	0.70	8238
weighted avg	0.89	0.90	0.89	8238



AUC Score of KNeighborsClassifier(n_neighbors=8, weights='distance') for resampled test d
ata :
0.8849600806641825

The Cross Entropy score of KNeighborsClassifier(n_neighbors=8, weights='distance') Model
: 3.425390752565618

Mean Score : 0.9035508345978756
Bias error : 9.644916540212446
Variance error : 0.5015398751828585

Overall accuracy of GaussianNB() for resampled train data : 84.0
Overall accuracy of GaussianNB() for resampled test data : 84.0

Confusion Matrix for GaussianNB() Model :
[[6389 921]
[426 502]]

Sensitivity : 54.09
Specificity : 87.4

Classification report of GaussianNB() for resampled train data :
---

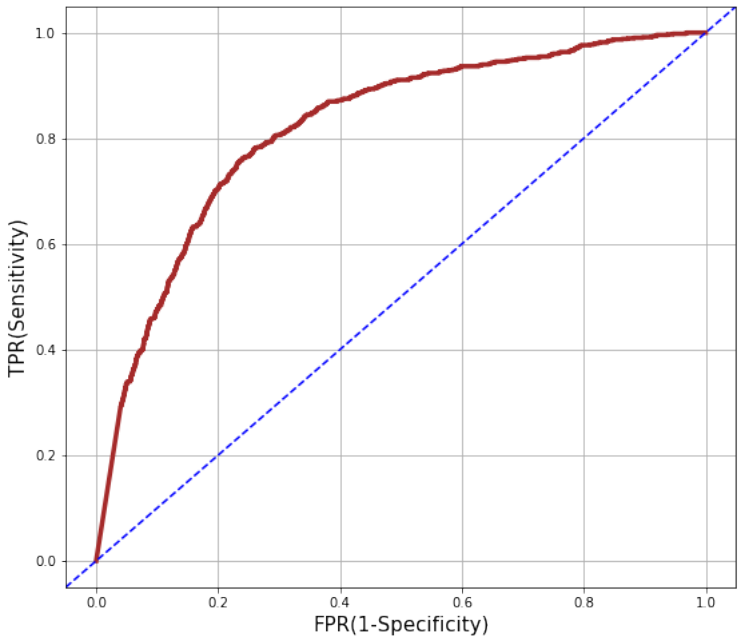
	precision	recall	f1-score	support
0	0.94	0.87	0.91	29238
1	0.37	0.57	0.45	3712

accuracy			0.84	32950
macro avg	0.65	0.72	0.68	32950
weighted avg	0.88	0.84	0.85	32950

Classification report of GaussianNB() for resampled test data :

---

	precision	recall	f1-score	support
0	0.94	0.87	0.90	7310
1	0.35	0.54	0.43	928
accuracy			0.84	8238
macro avg	0.65	0.71	0.67	8238
weighted avg	0.87	0.84	0.85	8238



AUC Score of GaussianNB() for resampled test data :
0.8193490111561865

The Cross Entropy score of GaussianNB() Model : 5.647544092398385

Mean Score : 0.8394233687405158
Bias error : 16.05766312594842
Variance error : 0.7219069070443396

Overall accuracy of RandomForestClassifier() for resampled train data : 100.0

Overall accuracy of RandomForestClassifier() for resampled test data : 91.0

Confusion Matrix for RandomForestClassifier() Model :
[[7066 244]
 [521 407]]

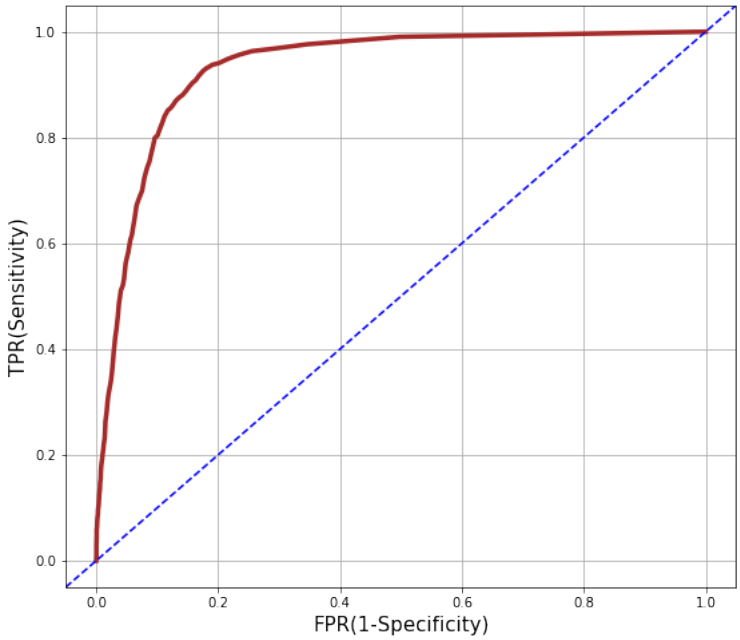
Sensitivity : 43.86
Specificity : 96.66

Classification report of RandomForestClassifier() for resampled train data :
---

	precision	recall	f1-score	support
0	1.00	1.00	1.00	29238
1	1.00	1.00	1.00	3712
accuracy			1.00	32950
macro avg	1.00	1.00	1.00	32950
weighted avg	1.00	1.00	1.00	32950

Classification report of RandomForestClassifier() for resampled test data :
---

	precision	recall	f1-score	support
0	0.93	0.97	0.95	7310
1	0.63	0.44	0.52	928
accuracy			0.91	8238
macro avg	0.78	0.70	0.73	8238
weighted avg	0.90	0.91	0.90	8238



AUC Score of RandomForestClassifier() for resampled test data :
0.9306856308080571

The Cross Entropy score of RandomForestClassifier() Model : 3.2073754605340667

Mean Score : 0.9129893778452199
Bias error : 8.701062215478006
Variance error : 0.23601369544260956

Conclusion:

From the refitted base models with tunned parameters and Random Forest Model on ori ginal data it is observed that the Random Forest Model has better overall performance in terms of ROC AUC Score, Bias Error and variance error and Mean accuracy Score obtai ned through 10-Fold Cross Validation as compared to the other models. The Logistic Re gression and Decision Tree Model can be considered further for tunning or re-modelling depending on the further analysis.

3.Checking the presence of Imbalance in the target feature(y):

% of Positive class(1) : 11.27 %

% of Negative class(0) : 88.73 %

Thus we can see that there is imbalance present in the target variable (y).

- 88.73% of 0 class which is the majority class.
- 11.26% of 1 class which is the minority class.

Presence of class imbalance in the target feature can be a defect which might result in faulty predictions by the classification algorithm with a tendency to classify the testing data more towards the majority class.

Thus, in order to remove Imbalance we will use **SMOTE** analysis or the **Synthetic Minority Over Sampling Technique** by which new examples can be synthesized from the existing examples. This type of data augmentation for the minority class can reduce the huge difference between the % distribution of each unique sub-category in the target variable.

4.SMOTE ANALYSIS:

***CODE*:**

```
from imblearn.over_sampling import SMOTE

smote=SMOTE(sampling_strategy=0.95,random_state=10)

X_res, y_res = smote.fit_resample(X,y)
```

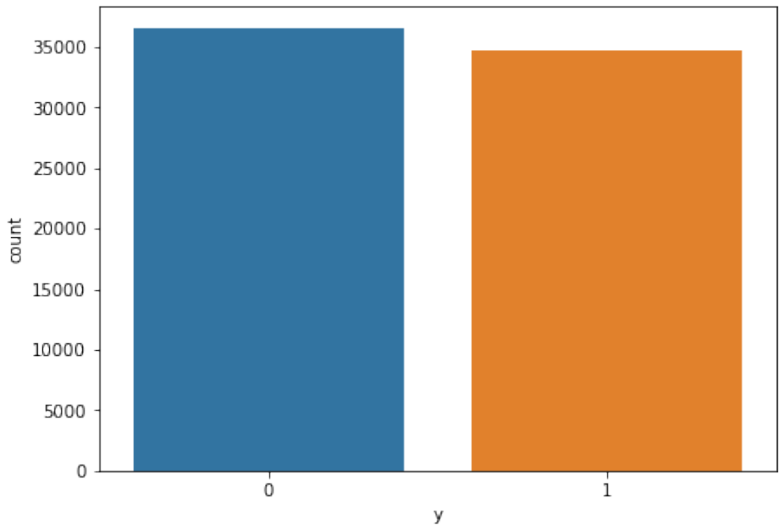
***OUTPUT*:**

Resampled X : (71268, 48)
Resampled y : (71268, 1)

Checking % of sub-categories in the Target variable (y) after SMOTE analysis:

% of Positive class(1) : 48.72 %
% of Negative class(0) : 51.28 %

Hence it is observed that the original dataset which had been split into the independent variab les (X) and target feature (y) has been resampled with 95% sampling strategy for better ML p redictions.



Therefore the imbalance in the target feature (y) has been reduced by 95% sampling rate of Up sampling of Minority class(1) and Down sampling of Majority class(0).

5. Refitting Logistic Regression, Decision tree, Random Forest algorithm on Resampled data:

The Accuracy score of resampled test data for the LogisticRegression(max_iter=110) Model:
0.9051494317384594
The Accuracy score of resampled train data for the LogisticRegression(max_iter=110) Model :
0.9049531693969902

Confusion Matrix of LogisticRegression(max_iter=110) Model for resampled test data :

[[6566 745]
[607 6336]]

LogisticRegression(max_iter=110) Model classification report after resampling for test data:
---

	precision	recall	f1-score	support
0	0.92	0.89	0.91	29237
1	0.89	0.92	0.90	27777
accuracy			0.90	57014
macro avg	0.90	0.91	0.90	57014
weighted avg	0.91	0.90	0.90	57014

LogisticRegression(max_iter=110) Model classification report after resampling for test data:
---

	precision	recall	f1-score	support
0	0.92	0.90	0.91	7311
1	0.89	0.91	0.90	6943
accuracy			0.91	14254
macro avg	0.91	0.91	0.91	14254
weighted avg	0.91	0.91	0.91	14254

Sensitivity : 0.9125738153535935
Specificity : 0.8980987553002325

AUC score of the LogisticRegression(max_iter=110) Model for resampled test data : 0.9637036231069915

The Cross Entropy score of LogisticRegression(max_iter=110) Model : 3.276064359899309

Mean Score : 0.9432595590957324
Bias error : 5.674044090426755
Variance error : 0.12939135794455692

The Accuracy score of resampled test data for the DecisionTreeClassifier(criterion='entropy', max_depth=10, min_samples_split=20) Model:
0.9019222674337029
The Accuracy score of resampled train data for the DecisionTreeClassifier(criterion='entropy', max_depth=10, min_samples_split=20) Model:
0.9130213631739573

Confusion Matrix of DecisionTreeClassifier(criterion='entropy', max_depth=10, min_samples_split=20) Model for resampled test data :

[[6355 956]
[442 6501]]

DecisionTreeClassifier(criterion='entropy', max_depth=10, min_samples_split=20) Model classification report after resampling for test data:
---

	precision	recall	f1-score	support
0	0.95	0.88	0.91	29237
1	0.88	0.95	0.91	27777

accuracy			0.91	57014
macro avg	0.91	0.91	0.91	57014
weighted avg	0.92	0.91	0.91	57014

DecisionTreeClassifier(criterion='entropy', max_depth=10, min_samples_split=20) Model cla ssification report after resampling for test data:

---

	precision	recall	f1-score	support
0	0.93	0.87	0.90	7311
1	0.87	0.94	0.90	6943
accuracy			0.90	14254
macro avg	0.90	0.90	0.90	14254
weighted avg	0.90	0.90	0.90	14254

Sensitivity : 0.9363387584617601
Specificity : 0.8692381343181508

AUC score of the DecisionTreeClassifier(criterion='entropy', max_depth=10, min_samples_s plit=20) Model for resampled test data : 0.954019425782048

The Cross Entropy score of DecisionTreeClassifier(criterion='entropy', max_depth=10, min_ samples_split=20) Model : 3.387538502541631

Mean Score : 0.9433471891773065
Bias error : 5.665281082269347
Variance error : 0.16018118838211745

The Accuracy score of resampled test data for the RandomForestClassifier() Model:
0.9473130349375614
The Accuracy score of resampled train data for the RandomForestClassifier() Model:
1.0

Confusion Matrix of RandomForestClassifier() Model for resampled test data :

[[6739 572]
[179 6764]]

RandomForestClassifier() Model classification report after resampling for test data:

---

	precision	recall	f1-score	support
0	1.00	1.00	1.00	29237
1	1.00	1.00	1.00	27777
accuracy			1.00	57014
macro avg	1.00	1.00	1.00	57014
weighted avg	1.00	1.00	1.00	57014

RandomForestClassifier() Model classification report after resampling for test data:
---

	precision	recall	f1-score	support
0	0.97	0.92	0.95	7311
1	0.92	0.97	0.95	6943
accuracy			0.95	14254
macro avg	0.95	0.95	0.95	14254
weighted avg	0.95	0.95	0.95	14254

Sensitivity : 0.9742186374765951
Specificity : 0.9217617289016551

AUC score of the RandomForestClassifier() Model for resampled test data : 0.9905469972551172

The Cross Entropy score of RandomForestClassifier() Model : 1.8197753923325388

Mean Score : 0.9431367520857442
Bias error : 5.686324791425578
Variance error : 0.1346738054128647

Observations:

- Mean Score : 94.73 %
- Overall accuracy of RandomForestClassifier() for resampled train data : 100%
- Overall accuracy of RandomForestClassifier() for resampled test data : 94.6%
- Sensitivity : 97 %
- Specificity : 92 %
- f1-score weighted average : 95%
- AUC score of the RandomForestClassifier() Model for resampled test data : 0.99
- Cross Entropy score of Random Forest Model : 1.8197753923325388
- Bias Error : 5.686324791425578
- Variance error : 0.1346738054128647

Conclusion:

The Random Forest Model computed on resampled data has higer Precision and Recall % as compared to the previous model built with the actual training data which can be observed from the classification report on the train and test data. Hence we can consider this model as a very efficient model from the overall analysis till further Optimizations are implemented.

6.Feature Extraction:

Feature extraction is a process of dimensionality reduction by which an initial set of raw data is reduced to more manageable groups for processing. A characteristic of these large data sets is a large number of variables that require a lot of computing resources to process. Feature extraction is the name for methods that select and /or combine variables into features, effectively reducing the amount of data that must be processed, while still accurately and completely describing the original data set. The process of feature extraction is useful when you need to reduce the number of resources needed for processing without losing important or relevant information. Feature extraction can also reduce the amount of redundant data for a given analysis. Also, the reduction of the data and the machine's efforts in building variable combinations (features) facilitate the speed of learning and generalization steps in the machine learning process.

Using Recursive Feature Elimination:

Since the analysis predicts that the Random Forest Model has the best overall performance from all the previously deployed base and tunned models, it will be considered for deriving the final set of features affecting the ML algorithm.

	Features	Rank
24	education_high.school	1
46	day_of_week_tue	1
45	day_of_week_thu	1
44	day_of_week_mon	1
40	month_may	1
34	contact_telephone	1
33	loan_yes	1
31	housing_yes	1
29	default_unknown	1
27	education_university.degree	1
47	day_of_week_wed	1
23	education_basic.9y	1
20	marital_single	1
19	marital_married	1
16	job_technician	1
48	poutcome_success	1
4	previous	1

5	cons.price.idx	1
1	age	1
6	cons.conf.idx	1
7	nr.employed	1
2	duration	1
8	job_blue-collar	1
3	campaign	1

Thus we have obtained the Features which contribute significantly for the model.

['education_high.school', 'day_of_week_tue', 'day_of_week_thu', 'day_of_week_mon', 'month_may', 'contact_telephone', 'loan_yes', 'housing_yes', 'default_unknown', 'education_university.degree', 'day_of_week_wed', 'education_basic.9y', 'marital_single', 'marital_married', 'job_technician', 'poutcome_success', 'previous', 'cons.price.idx', 'age', 'cons.conf.idx', 'nr.employed', 'duration', 'job_blue-collar', 'campaign']

7.Refitting the Logistic Regression model, Decision Tree model and Random Forest model on the new dataset containing features obtained after recursive elimination:

Overall accuracy of LogisticRegression() for resampled train data : 89.0
Overall accuracy of LogisticRegression() for resampled test data : 89.0

Confusion Matrix for LogisticRegression() Model :
[[6420 891]
[648 6295]]

Sensitivity : 0.9066685870661098
Specificity : 0.8781288469429627

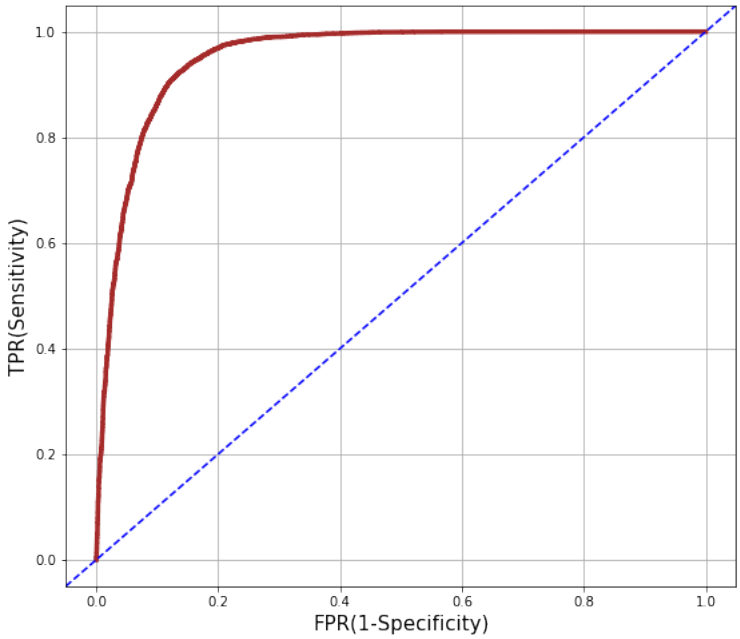
Classification report of LogisticRegression() for resampled train data :
---

	precision	recall	f1-score	support
0	0.91	0.88	0.89	29237
1	0.88	0.91	0.89	27777
accuracy			0.89	57014
macro avg	0.89	0.89	0.89	57014
weighted avg	0.89	0.89	0.89	57014

Classification report of LogisticRegression() for resampled test data :

---

	precision	recall	f1-score	support
0	0.91	0.88	0.89	7311
1	0.88	0.91	0.89	6943
accuracy			0.89	14254
macro avg	0.89	0.89	0.89	14254
weighted avg	0.89	0.89	0.89	14254



AUC Score of LogisticRegression() for resampled test data :
0.9516637942431871

The Cross Entropy score of LogisticRegression() Model : 3.7291910560599026

Mean Score : 0.8924123749942398
Bias error : 10.758762500576024
Variance error : 0.21879522316532932

Overall accuracy of DecisionTreeClassifier() for resampled train data : 100.0
Overall accuracy of DecisionTreeClassifier() for resampled test data : 91.0

Confusion Matrix for DecisionTreeClassifier() Model :
[[6596 715]
[603 6340]]

Sensitivity : 0.9131499351865188
Specificity : 0.9022021611270687

Classification report of DecisionTreeClassifier() for resampled train data :

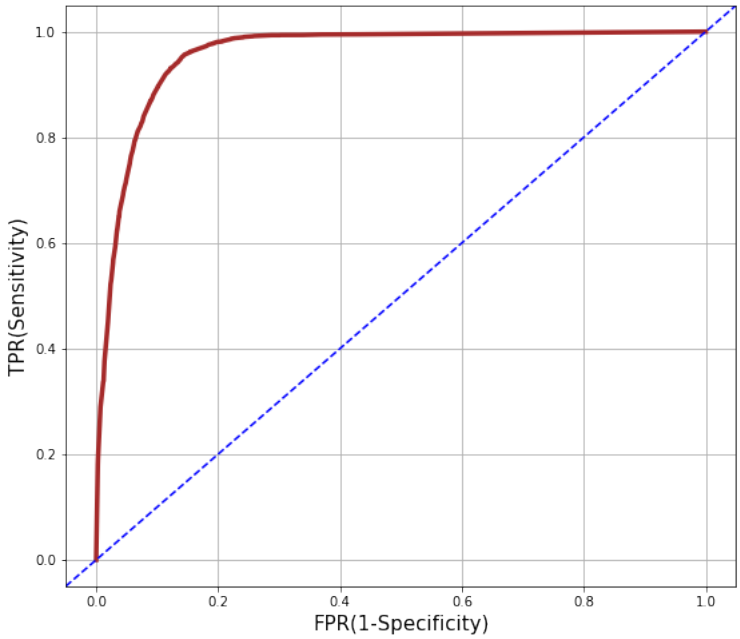
---

	precision	recall	f1-score	support
0	1.00	1.00	1.00	29237
1	1.00	1.00	1.00	27777
accuracy			1.00	57014
macro avg	1.00	1.00	1.00	57014
weighted avg	1.00	1.00	1.00	57014

Classification report of DecisionTreeClassifier() for resampled test data :

---

	precision	recall	f1-score	support
0	0.92	0.90	0.91	7311
1	0.90	0.91	0.91	6943
accuracy			0.91	14254
macro avg	0.91	0.91	0.91	14254
weighted avg	0.91	0.91	0.91	14254



AUC Score of DecisionTreeClassifier() for resampled test data :
0.9076760481567937

The Cross Entropy score of DecisionTreeClassifier() Model : 3.193677494082705

Mean Score : 0.9051810862746239
Bias error : 9.48189137253761
Variance error : 0.2299501130526659

Overall accuracy of RandomForestClassifier() for resampled train data : 100.0
Overall accuracy of RandomForestClassifier() for resampled test data : 95.0

Confusion Matrix for RandomForestClassifier() Model :
[[6710 601]
[179 6764]]

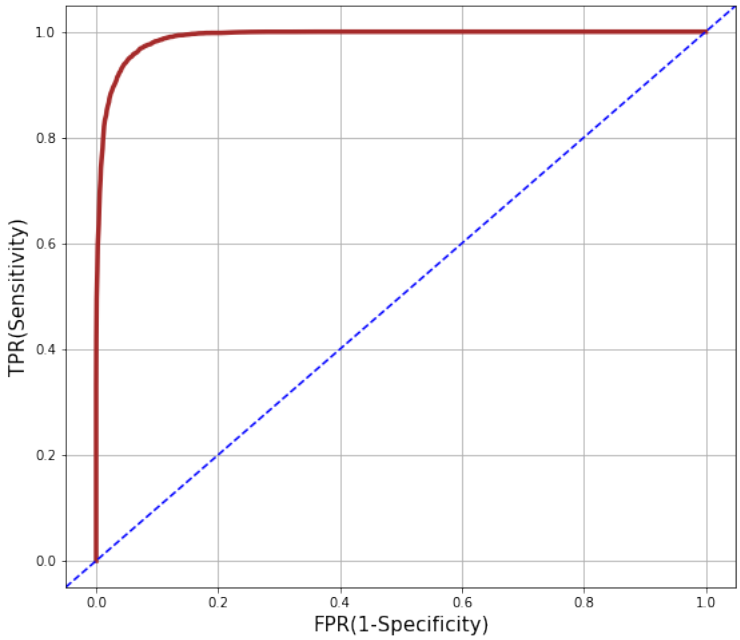
Sensitivity : 0.9742186374765951
Specificity : 0.9177951032690467

Classification report of RandomForestClassifier() for resampled train data :
---

	precision	recall	f1-score	support
0	1.00	1.00	1.00	29237
1	1.00	1.00	1.00	27777
accuracy			1.00	57014
macro avg	1.00	1.00	1.00	57014
weighted avg	1.00	1.00	1.00	57014

Classification report of RandomForestClassifier() for resampled test data :
---

	precision	recall	f1-score	support
0	0.97	0.92	0.95	7311
1	0.92	0.97	0.95	6943
accuracy			0.95	14254
macro avg	0.95	0.95	0.95	14254
weighted avg	0.95	0.95	0.95	14254



AUC Score of RandomForestClassifier() for resampled test data :
0.989728118680528

The Cross Entropy score of RandomForestClassifier() Model : 1.890046733975438

Mean Score : 0.942382476912276
Bias error : 5.761752308772405
Variance error : 0.3371929297281598

From the above analysis it can be inferred that :

* The Sensitivity and Specificity of both Logistic Regression model and Decision Tree model has increased significantly from the base tunned models resulting in an improved weighted-average of Precision and Recall rates.

* The Decision Tree Model has a better overall performance than the Logistic Model for the new dataset obtained after recursive feature elimination and resampling the data.

* The Random Forest Model has not shown any major improvement from the previo us model but has the best overall performance as compared to the Logistic Regression a nd Decision Tree model. Thus it will be an efficient model for the problem statement w ith and mean accuracy score of 95 % on test data through 10-Fold Cross Validation.

Therefore we will be considering the Random Forest Classifier model.

Random Forest Model :

* Overall accuracy of RandomForestClassifier() for resampled train data : 100%
* Overall accuracy of RandomForestClassifier() for resampled test data : 95%
* Sensitivity : 97 %
* Specificity : 92 %
* f1-score weighted average : 95%
* AUC Score of RandomForestClassifier() for resampled test data : 0.99
* Bias Error : 5.7612318385071755
* variance error : 0.3371929297281598

8.Boosting Algorithms on the selected resampled features:

Overall accuracy of AdaBoostClassifier() for resampled train data : 90.0
Overall accuracy of AdaBoostClassifier() for resampled test data : 90.0

Confusion Matrix for AdaBoostClassifier() Model :
[[6540 771]
[690 6253]]

Sensitivity : 0.9006193288203946
Specificity : 0.8945424702503078

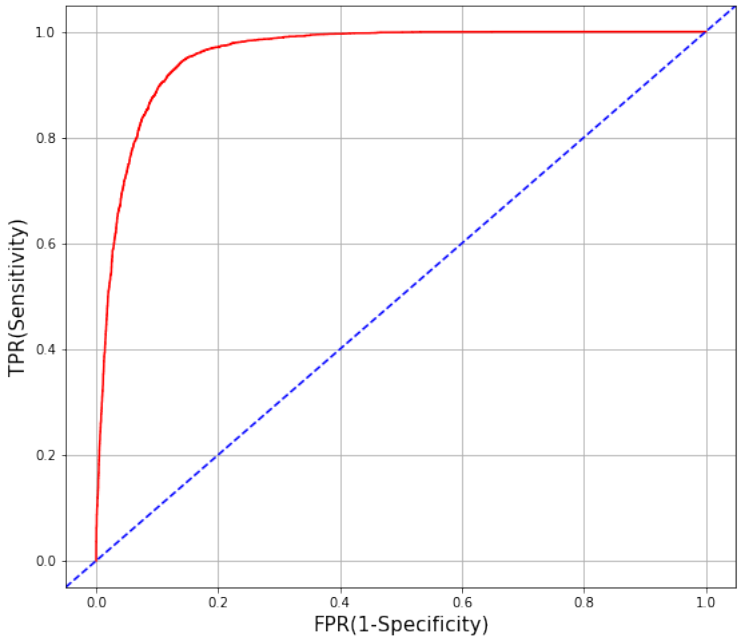
Classification report of AdaBoostClassifier() for resampled train data :

	precision	recall	f1-score	support
0	0.90	0.89	0.90	29237
1	0.89	0.90	0.89	27777
accuracy			0.90	57014
macro avg	0.90	0.90	0.90	57014

weighted avg 0.90 0.90 0.90 57014

Classification report of AdaBoostClassifier() for resampled test data :
---

	precision	recall	f1-score	support
0	0.90	0.89	0.90	7311
1	0.89	0.90	0.90	6943
accuracy			0.90	14254
macro avg	0.90	0.90	0.90	14254
weighted avg	0.90	0.90	0.90	14254



AUC Score for AdaBoostClassifier() for resampled test data :
0.9578587668352375

The Cross Entropy score of AdaBoostClassifier() Model : 3.5401830224907536

Mean Score : 0.8952187525052218
Bias error : 10.478124749477825
Variance error : 0.3794162941775748

Overall accuracy of GradientBoostingClassifier() for resampled train data : 91.0
Overall accuracy of GradientBoostingClassifier() for resampled test data : 91.0

Confusion Matrix for GradientBoostingClassifier() Model :

[[6417 894]
 [411 6532]]

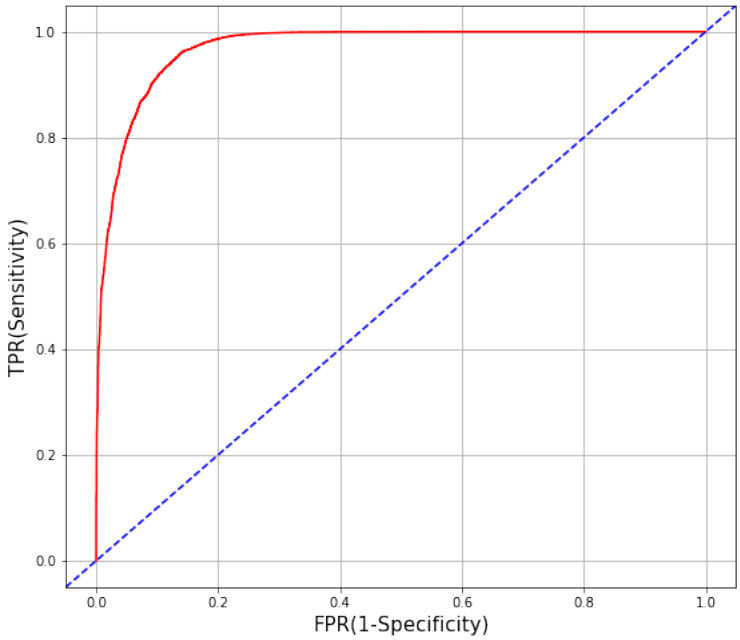
Sensitivity : 0.9408036871669307
Specificity : 0.877718506360279

Classification report of GradientBoostingClassifier() for resampled train data :
---

	precision	recall	f1-score	support
0	0.94	0.88	0.91	29237
1	0.88	0.94	0.91	27777
accuracy			0.91	57014
macro avg	0.91	0.91	0.91	57014
weighted avg	0.91	0.91	0.91	57014

Classification report of GradientBoostingClassifier() for resampled test data :
---

	precision	recall	f1-score	support
0	0.94	0.88	0.91	7311
1	0.88	0.94	0.91	6943
accuracy			0.91	14254
macro avg	0.91	0.91	0.91	14254
weighted avg	0.91	0.91	0.91	14254



AUC Score for GradientBoostingClassifier() for resampled test data :
0.9698130169630885

The Cross Entropy score of GradientBoostingClassifier() Model : 3.1621873183289635

Mean Score : 0.9058477313665179
Bias error : 9.415226863348213
Variance error : 0.3081681195676774

Overall accuracy of XGBClassifier(base_score=0.5, booster='gbtree', colsample_bylevel=1,
colsample_bynode=1, colsample_bytree=1, eval_metric='logloss',
gamma=0, gpu_id=-1, importance_type='gain',
interaction_constraints='', learning_rate=0.300000012,
max_delta_step=0, max_depth=6, min_child_weight=1, missing=nan,
monotone_constraints='()', n_estimators=100, n_jobs=4,
num_parallel_tree=1, random_state=0, reg_alpha=0, reg_lambda=1,
scale_pos_weight=1, subsample=1, tree_method='exact',
validate_parameters=1, verbosity=None) for resampled train data : 97.0
Overall accuracy of XGBClassifier(base_score=0.5, booster='gbtree', colsample_bylevel=1,
colsample_bynode=1, colsample_bytree=1, eval_metric='logloss',
gamma=0, gpu_id=-1, importance_type='gain',
interaction_constraints='', learning_rate=0.300000012,
max_delta_step=0, max_depth=6, min_child_weight=1, missing=nan,
monotone_constraints='()', n_estimators=100, n_jobs=4,
num_parallel_tree=1, random_state=0, reg_alpha=0, reg_lambda=1,
scale_pos_weight=1, subsample=1, tree_method='exact',
validate_parameters=1, verbosity=None) for resampled test data : 95.0
Confusion Matrix for XGBClassifier(base_score=0.5, booster='gbtree', colsample_bylevel=1,
colsample_bynode=1, colsample_bytree=1, eval_metric='logloss',
gamma=0, gpu_id=-1, importance_type='gain',
interaction_constraints='', learning_rate=0.300000012,
max_delta_step=0, max_depth=6, min_child_weight=1, missing=nan,
monotone_constraints='()', n_estimators=100, n_jobs=4,
num_parallel_tree=1, random_state=0, reg_alpha=0, reg_lambda=1,
scale_pos_weight=1, subsample=1, tree_method='exact',
validate_parameters=1, verbosity=None) Model :
[[6862 449]
[325 6618]]

Sensitivity : 0.9531902635748236
Specificity : 0.9385856927916838

Classification report of XGBClassifier(base_score=0.5, booster='gbtree', colsample_bylevel=
1,
colsample_bynode=1, colsample_bytree=1, eval_metric='logloss',

gamma=0, gpu_id=-1, importance_type='gain',
interaction_constraints='', learning_rate=0.300000012,
max_delta_step=0, max_depth=6, min_child_weight=1, missing=nan,
monotone_constraints='()', n_estimators=100, n_jobs=4,
num_parallel_tree=1, random_state=0, reg_alpha=0, reg_lambda=1,
scale_pos_weight=1, subsample=1, tree_method='exact',
validate_parameters=1, verbosity=None) for resampled train data :

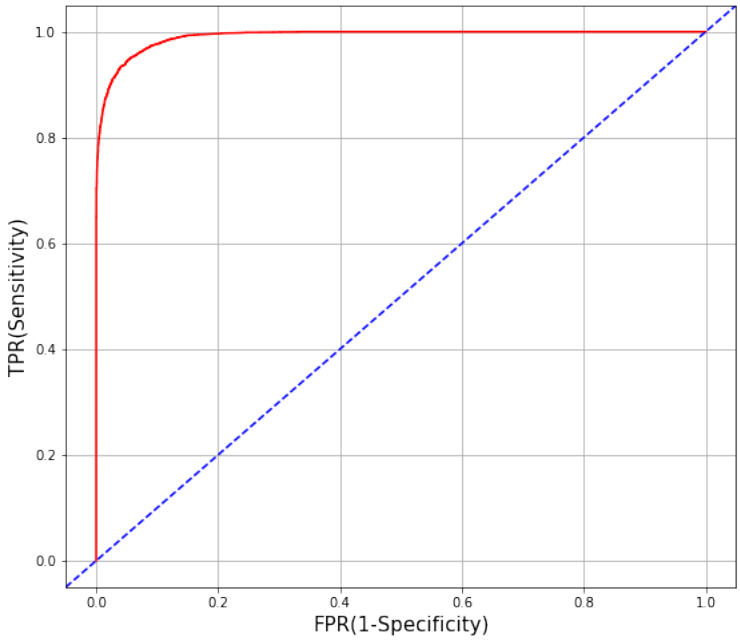
---

	precision	recall	f1-score	support
0	0.97	0.96	0.97	29237
1	0.96	0.97	0.96	27777
accuracy			0.97	57014
macro avg	0.97	0.97	0.97	57014
weighted avg	0.97	0.97	0.97	57014

Classification report of XGBClassifier(base_score=0.5, booster='gbtree', colsample_bylevel=1,
colsample_bynode=1, colsample_bytree=1, eval_metric='logloss',
gamma=0, gpu_id=-1, importance_type='gain',
interaction_constraints='', learning_rate=0.300000012,
max_delta_step=0, max_depth=6, min_child_weight=1, missing=nan,
monotone_constraints='()', n_estimators=100, n_jobs=4,
num_parallel_tree=1, random_state=0, reg_alpha=0, reg_lambda=1,
scale_pos_weight=1, subsample=1, tree_method='exact',
validate_parameters=1, verbosity=None) for resampled test data :

---

	precision	recall	f1-score	support
0	0.95	0.94	0.95	7311
1	0.94	0.95	0.94	6943
accuracy			0.95	14254
macro avg	0.95	0.95	0.95	14254
weighted avg	0.95	0.95	0.95	14254



AUC Score for XGBClassifier(base_score=0.5, booster='gbtree', colsample_bylevel=1,
colsample_bynode=1, colsample_bytree=1, eval_metric='logloss',
gamma=0, gpu_id=-1, importance_type='gain',
interaction_constraints='', learning_rate=0.300000012,
max_delta_step=0, max_depth=6, min_child_weight=1, missing=nan,
monotone_constraints='()', n_estimators=100, n_jobs=4,
num_parallel_tree=1, random_state=0, reg_alpha=0, reg_lambda=1,
scale_pos_weight=1, subsample=1, tree_method='exact',
validate_parameters=1, verbosity=None) for resampled test data :
0.9911896848939327

The Cross Entropy score of XGBClassifier(base_score=0.5, booster='gbtree', colsample_byle
vel=1,
colsample_bynode=1, colsample_bytree=1, eval_metric='logloss',
gamma=0, gpu_id=-1, importance_type='gain',
interaction_constraints='', learning_rate=0.300000012,
max_delta_step=0, max_depth=6, min_child_weight=1, missing=nan,
monotone_constraints='()', n_estimators=100, n_jobs=4,
num_parallel_tree=1, random_state=0, reg_alpha=0, reg_lambda=1,
scale_pos_weight=1, subsample=1, tree_method='exact',
validate_parameters=1, verbosity=None) Model : 1.8754996456368063

Mean Score : 0.9453642714751995
Bias error : 5.4635728524800475
Variance error : 0.26113486446281947

Conclusion: XGBoost Model shows the best overall performance as compared to ot
her models on the selected features obtained by recursive feature elimination.

9.Boosting Algorithms on full Resampled data :

Overall accuracy of AdaBoostClassifier() for resampled train data : 90.0
Overall accuracy of AdaBoostClassifier() for resampled test data : 90.0

Confusion Matrix for AdaBoostClassifier() Model :
[[6577 734]
[653 6290]]

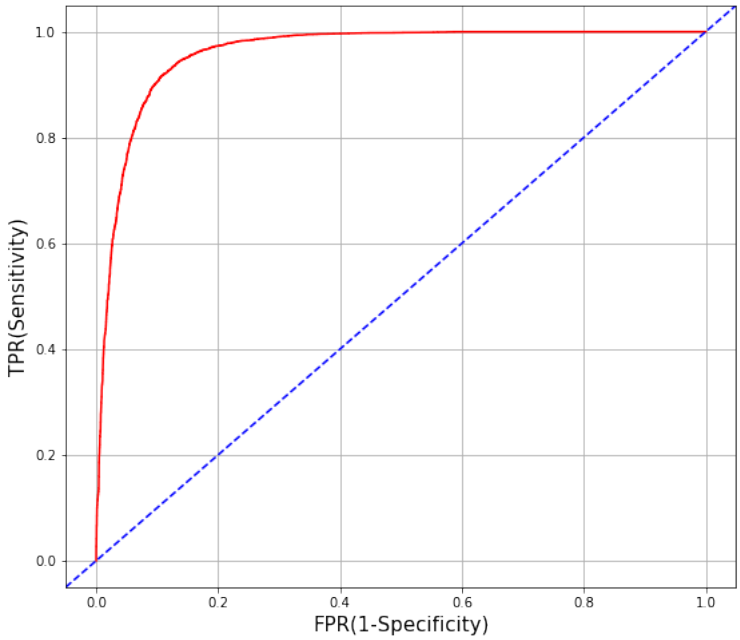
Sensitivity : 0.9059484372749532
Specificity : 0.8996033374367391

Classification report of AdaBoostClassifier() for resampled train data :
---

	precision	recall	f1-score	support
0	0.91	0.90	0.90	29237
1	0.89	0.90	0.90	27777
accuracy			0.90	57014
macro avg	0.90	0.90	0.90	57014
weighted avg	0.90	0.90	0.90	57014

Classification report of AdaBoostClassifier() for resampled test data :
---

	precision	recall	f1-score	support
0	0.91	0.90	0.90	7311
1	0.90	0.91	0.90	6943
accuracy			0.90	14254
macro avg	0.90	0.90	0.90	14254
weighted avg	0.90	0.90	0.90	14254



AUC Score for AdaBoostClassifier() for resampled test data :
0.9605577968424244

The Cross Entropy score of AdaBoostClassifier() Model : 3.360872019380868

Mean Score : 0.8991474786032911
Bias error : 10.085252139670885
Variance error : 0.36424727176963523

Overall accuracy of GradientBoostingClassifier() for resampled train data : 91.0
Overall accuracy of GradientBoostingClassifier() for resampled test data : 91.0

Confusion Matrix for GradientBoostingClassifier() Model :
[[6418 893]
[359 6584]]

Sensitivity : 0.9482932449949589
Specificity : 0.877855286554507

Classification report of GradientBoostingClassifier() for resampled train data :
---

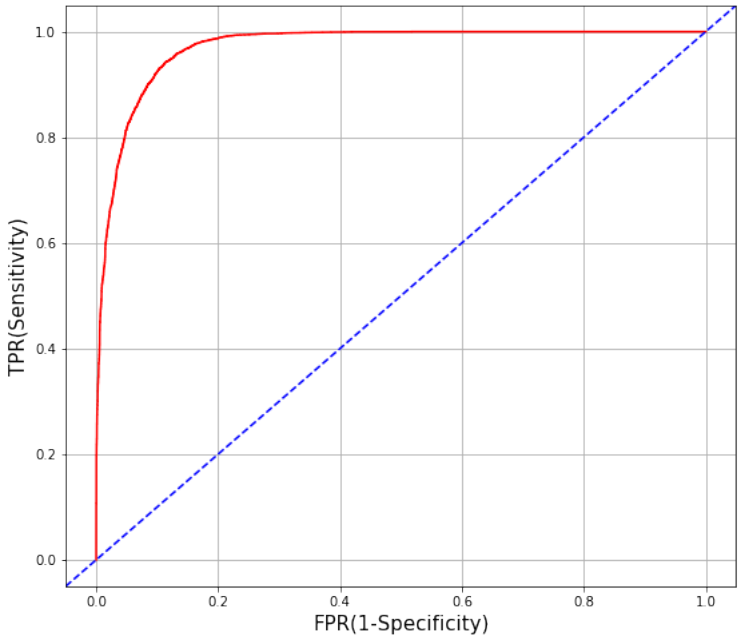
	precision	recall	f1-score	support
0	0.95	0.88	0.91	29237
1	0.88	0.95	0.91	27777

accuracy			0.91	57014
macro avg	0.91	0.91	0.91	57014
weighted avg	0.92	0.91	0.91	57014

Classification report of GradientBoostingClassifier() for resampled test data :

---

	precision	recall	f1-score	support
0	0.95	0.88	0.91	7311
1	0.88	0.95	0.91	6943
accuracy			0.91	14254
macro avg	0.91	0.91	0.91	14254
weighted avg	0.91	0.91	0.91	14254



AUC Score for GradientBoostingClassifier() for resampled test data :
0.9711705746736232

The Cross Entropy score of GradientBoostingClassifier() Model : 3.0337633006126947

Mean Score : 0.9101273838559955
Bias error : 8.987261614400454
Variance error : 0.2082533457686795

Overall accuracy of XGBClassifier(base_score=0.5, booster='gbtree', colsample_bylevel=1,
colsample_bynode=1, colsample_bytree=1, eval_metric='logloss',

gamma=0, gpu_id=-1, importance_type='gain',
interaction_constraints='', learning_rate=0.300000012,
max_delta_step=0, max_depth=6, min_child_weight=1, missing=nan,
monotone_constraints='()', n_estimators=100, n_jobs=4,
num_parallel_tree=1, random_state=0, reg_alpha=0, reg_lambda=1,
scale_pos_weight=1, subsample=1, tree_method='exact',
validate_parameters=1, verbosity=None) for resampled train data : 97.0
Overall accuracy of XGBClassifier(base_score=0.5, booster='gbtree', colsample_bylevel=1,
colsample_bynode=1, colsample_bytree=1, eval_metric='logloss',
gamma=0, gpu_id=-1, importance_type='gain',
interaction_constraints='', learning_rate=0.300000012,
max_delta_step=0, max_depth=6, min_child_weight=1, missing=nan,
monotone_constraints='()', n_estimators=100, n_jobs=4,
num_parallel_tree=1, random_state=0, reg_alpha=0, reg_lambda=1,
scale_pos_weight=1, subsample=1, tree_method='exact',
validate_parameters=1, verbosity=None) for resampled test data : 95.0

Confusion Matrix for XGBClassifier(base_score=0.5, booster='gbtree', colsample_bylevel=1,
colsample_bynode=1, colsample_bytree=1, eval_metric='logloss',
gamma=0, gpu_id=-1, importance_type='gain',
interaction_constraints='', learning_rate=0.300000012,
max_delta_step=0, max_depth=6, min_child_weight=1, missing=nan,
monotone_constraints='()', n_estimators=100, n_jobs=4,
num_parallel_tree=1, random_state=0, reg_alpha=0, reg_lambda=1,
scale_pos_weight=1, subsample=1, tree_method='exact',
validate_parameters=1, verbosity=None) Model :
[[6877 434]
[324 6619]]

Sensitivity : 0.9533342935330549
Specificity : 0.9406373957051019

Classification report of XGBClassifier(base_score=0.5, booster='gbtree', colsample_bylevel=
1,
colsample_bynode=1, colsample_bytree=1, eval_metric='logloss',
gamma=0, gpu_id=-1, importance_type='gain',
interaction_constraints='', learning_rate=0.300000012,
max_delta_step=0, max_depth=6, min_child_weight=1, missing=nan,
monotone_constraints='()', n_estimators=100, n_jobs=4,
num_parallel_tree=1, random_state=0, reg_alpha=0, reg_lambda=1,
scale_pos_weight=1, subsample=1, tree_method='exact',
validate_parameters=1, verbosity=None) for resampled train data :
---

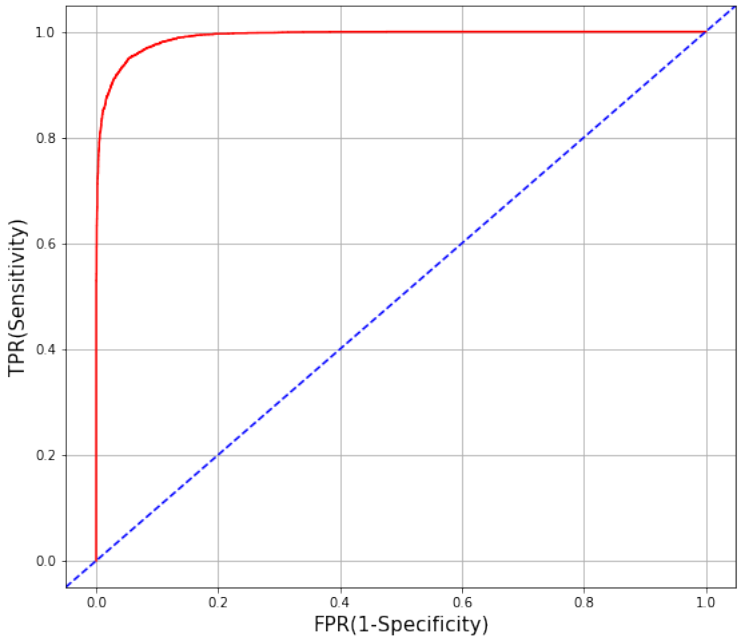
	precision	recall	f1-score	support
0	0.97	0.96	0.97	29237
1	0.96	0.97	0.97	27777

accuracy			0.97	57014
macro avg	0.97	0.97	0.97	57014
weighted avg	0.97	0.97	0.97	57014

Classification report of XGBClassifier(base_score=0.5, booster='gbtree', colsample_bylevel=1,
colsample_bynode=1, colsample_bytree=1, eval_metric='logloss',
gamma=0, gpu_id=-1, importance_type='gain',
interaction_constraints='', learning_rate=0.300000012,
max_delta_step=0, max_depth=6, min_child_weight=1, missing=nan,
monotone_constraints='()', n_estimators=100, n_jobs=4,
num_parallel_tree=1, random_state=0, reg_alpha=0, reg_lambda=1,
scale_pos_weight=1, subsample=1, tree_method='exact',
validate_parameters=1, verbosity=None) for resampled test data :

---

	precision	recall	f1-score	support
0	0.96	0.94	0.95	7311
1	0.94	0.95	0.95	6943
accuracy			0.95	14254
macro avg	0.95	0.95	0.95	14254
weighted avg	0.95	0.95	0.95	14254



AUC Score for XGBClassifier(base_score=0.5, booster='gbtree', colsample_bylevel=1,
colsample_bynode=1, colsample_bytree=1, eval_metric='logloss',
gamma=0, gpu_id=-1, importance_type='gain',
interaction_constraints='', learning_rate=0.300000012,

max_delta_step=0, max_depth=6, min_child_weight=1, missing=nan,
monotone_constraints='()', n_estimators=100, n_jobs=4,
num_parallel_tree=1, random_state=0, reg_alpha=0, reg_lambda=1,
scale_pos_weight=1, subsample=1, tree_method='exact',
validate_parameters=1, verbosity=None) for resampled test data :
0.9904988690663661

The Cross Entropy score of XGBClassifier(base_score=0.5, booster='gbtree', colsample_byle
vel=1,
colsample_bynode=1, colsample_bytree=1, eval_metric='logloss',
gamma=0, gpu_id=-1, importance_type='gain',
interaction_constraints='', learning_rate=0.300000012,
max_delta_step=0, max_depth=6, min_child_weight=1, missing=nan,
monotone_constraints='()', n_estimators=100, n_jobs=4,
num_parallel_tree=1, random_state=0, reg_alpha=0, reg_lambda=1,
scale_pos_weight=1, subsample=1, tree_method='exact',
validate_parameters=1, verbosity=None) Model : 1.8367293063439745

Mean Score : 0.9430840866712756
Bias error : 5.691591332872436
Variance error : 0.16665850131776008

Conclusion: Taking into consideration both the cases, i.e. performance of Boosting algorithms on complete resampled data and performance on selected significant features, no major change in the performance has been observed. Hence we will consider the results on the selected significant features. It can be observed that the XGBoost model has shown a much better performance in terms of Precision and Recall rates in the train and test dataset thus having a more accurate model Sensitivity and Specificity. The ROC AUC score is highest for the XGBoost model as compared to the AdaBoost and Gradient Boost Model. Therefore, we will be considering the XGBoost model from the above analysis.

The Model optimization has been successfully carried out and from the above conclusion it can be inferred that both the Random Forest Model on selected features and the XGBoost model on selected features are very efficient models for classifying the customers of the Bank as to whether he/she is a subscriber of term deposit or not.
But we will further make detailed and critical analysis from the above considered models based on balance between model sensitivity and specificity on the test data and will consider the final model based on minimum overfitting condition on train data.

Model Comparison for Final Model selection :

Models	Mean Overall Score	Accuracy on Train Data	Accuracy on Test Data	Sensitivity	Specificity	F1-weighted average	AUC score	Bias Error	Variance Error
XGBoost Model	95%	97%	95%	95%	94%	95%	0.99	5.46	0.26
Random Forest Model	95%	100%	95%	97%	92%	95%	0.99	5.76	0.33
Gradient Boost Model	91%	91%	91%	94%	88%	91%	0.97	9.42	0.31
AdaBoost Model	90%	90%	90%	90%	89%	90%	0.96	10.47	0.38

Inferences:

- XGBoost Model and Random Forest Model have the same overall mean accur acy of 95% on test data after application of 10 Fold Cross Validation method.
- XGBoost has less overfitting of train data on the test data as compared to the R andom Forest Model.
- XGBoost Model has a better balance between model sensitivity and specificity as compared to Random Forest Model.
- XGBoost has a slightly lesser Bias error and Variance error than the Random F orest Model.
- f1-score weighted for XGBoost Model is 95%
- ROC AUC score for XGBoost Model is 0.99.

Final Prediction Model (BankPredicti on_XGBoostModel):

Source Code:

```
def BankPrediction_XGBoostModel(fin_model):
    X_new=X[['education_high.school', 'day_of_week_tue', 'day_of_week_thu',
             'day_of_week_mon', 'month_may', 'contact_telephone', 'loan_yes',
             'housing_yes', 'default_unknown', 'education_university.degree',
             'day_of_week_wed', 'education_basic.9y', 'marital_single',
             'marital_married', 'job_technician', 'poutcome_success', 'previous',
             'cons.price.idx', 'age', 'cons.conf.idx', 'nr.employed',
             'duration', 'job_blue-collar', 'campaign']]
    Y=bank_df['y']
    smt=SMOTE(sampling_strategy=0.95,random_state=10)
    Xnew_res, Y_res = smt.fit_resample(X_new,Y)
    XRes_train,XRes_test,YRes_train,YRes_test=train_test_split(Xnew_res,Y_res,test_s
ize=0.2,random_state=10)
    plt.figure(figsize=(10,9))
    Res_model=fin_model.fit(XRes_train,YRes_train)
    YRespred_train=Res_model.predict(XRes_train)
    YRespred_test=Res_model.predict(XRes_test)
    YRespred_test_prob=Res_model.predict_proba(XRes_test)
    print('Overall accuracy of XGBoost Model for resampled train data : ',
    np.round(accuracy_score(YRes_train,YRespred_train),2)*100)
    print('Overall accuracy of XGBoost Model for resampled test data : ',
    np.round(accuracy_score(YRes_test,YRespred_test),2)*100,'\n')
    confusion_mat=confusion_matrix(YRes_test,YRespred_test)
```

```
tn = confusion_mat[0,0]
tp = confusion_mat[1,1]
fp = confusion_mat[0,1]
fn = confusion_mat[1,0]
print('Confusion Matrix for XGBoost Model Model : ')
print(confusion_mat,'\n')
Sensitivity=((tp/(tp+fn))*100)
print('Sensitivity : ',np.round(Sensitivity,2))
Specificity=((tn/(tn+fp))*100)
print('Specificity : ',np.round(Specificity,2),'\n')
print('Classification report of XGBoost Model for resampled train data : ')
print('-------------------------------------------------------------------------------','\n')
print(classification_report(YRes_train,YRespred_train),'\n')
print('Classification report of XGBoost Model for resampled test data : ')
print('-------------------------------------------------------------------','\n')
print(classification_report(YRes_test,YRespred_test),'\n')
fpr,trp,th=roc_curve(YRes_test,YRespred_test_prob[:,1])
plt.plot(fpr,trp,color='orange',linewidth=6.5)
plt.xlim([-0.05,1.05])
plt.ylim([-0.05,1.05])
plt.grid()
plt.title('ROC Curve for XGBoost Model',fontsize=15)
plt.xlabel('FPR(1-Specificity)',fontsize=15)
plt.ylabel('TPR(Sensitivity)',fontsize=15)
plt.plot([-0.05,1.05],[-0.05,1.05],'--r',linewidth=4.5)
plt.show()
```

```
    print('AUC Score for XGBoost Model for resampled test data : ')

    print(roc_auc_score(YRes_test,YRespred_test_prob[:,1]),'\n')

    cross_entropy=log_loss(YRes_test,YRespred_test)

    print('The Cross Entropy score of XGBoost Model : ',cross_entropy,'\n')

    k=KFold(n_splits=10,shuffle=True, random_state=10)

    scores=cross_val_score(estimator=fin_model,X=XRes_train,y=YRes_train,cv=k,scoring='accuracy')

    print("Mean Score : ",np.mean(scores))

    print("Bias error : ",(1-np.mean(scores))*100)

    print("Variance error : ",(np.std(scores)/np.mean(scores))*100,'\n')

    print('-------------------------------------------------------------------------------------------------------------','\n')

fin_model=XGBClassifier(eval_metric='logloss')

BankPrediction_XGBoostModel(fin_model)
```

Output:

Overall accuracy of XGBoost Model for resampled train data : 97.0
Overall accuracy of XGBoost Model for resampled test data : 95.0

Confusion Matrix for XGBoost Model Model :
[[6862 449]
[325 6618]]

Sensitivity : 95.32
Specificity : 93.86

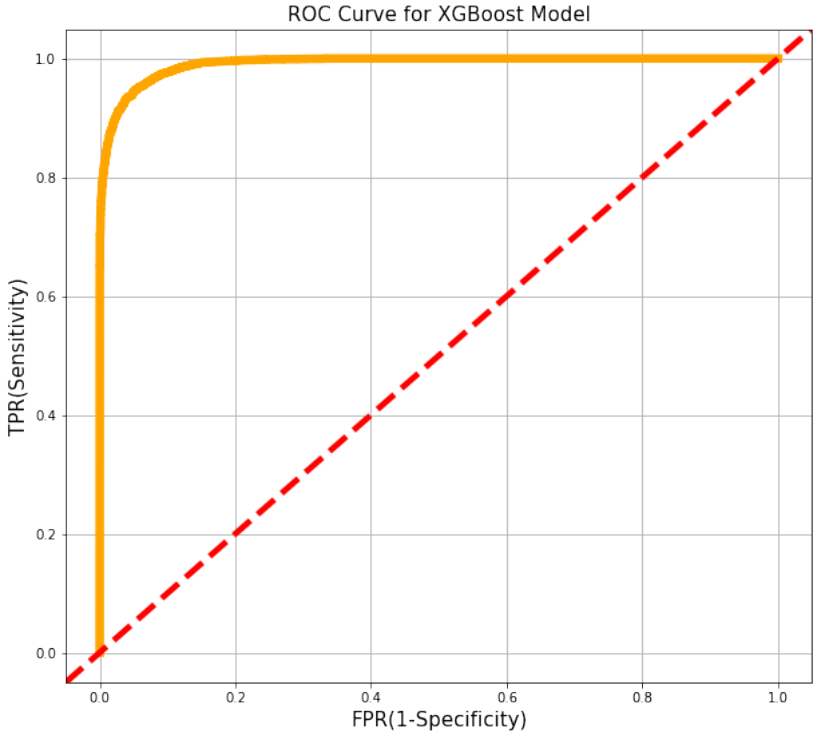
Classification report of XGBoost Model for resampled train data :
---

	precision	recall	f1-score	support
0	0.97	0.96	0.97	29237
1	0.96	0.97	0.96	27777
accuracy			0.97	57014
macro avg	0.97	0.97	0.97	57014
weighted avg	0.97	0.97	0.97	57014

Classification report of XGBoost Model for resampled test data :

---

	precision	recall	f1-score	support
0	0.95	0.94	0.95	7311
1	0.94	0.95	0.94	6943
accuracy			0.95	14254
macro avg	0.95	0.95	0.95	14254
weighted avg	0.95	0.95	0.95	14254



AUC Score for XGBoost Model for resampled test data :
0.9911896848939327

The Cross Entropy score of XGBoost Model : 1.8754996456368063

Mean Score : 0.9453642714751995
Bias error : 5.4635728524800475
Variance error : 0.26113486446281947

Conclusion

In the banking sector, a huge amount of data is generated continuously and this data can be used to extract meaningful information. The main objective of our project was to work towards building a suitable model to predict whether a customer will subscribe to a term deposit. The dataset that we used faced the problem of imbalance. We tried to solve this by using SMOTE analysis.

In this project, eight algorithms in total were deployed using the data as well as the resampled data and from the overall analysis we have shortlisted the best model in terms of performance. Therefore the XGBoost model given by the user defined function

def BankPrediction_XGBoostModel(fin_model):

has highest overall mean accuracy score of 94.5 which is rounded to 95%, ROC AUC score of 0.99, Sensitivity of 95.32% and Specificity is 93.86%. It can be observed that the XGBoost Model and Random Forest Model have the same overall accuracy of 95% on test data and can be an equally important model for consideration but we have considered the XGBoost as it exhibits less over fitting of train data on the testing set as compared to Random Forest Model. XGBoost has a slightly lesser Bias error and variance error than Random Forest Model. The f1-score weighted for XGBoost Model is 95%.

Through this project, we are trying to contribute to the acceleration of telemarketing campaigns and help to sustain a competitive advantage over the existing and prospective service providers in the banking industry.

These results may help service providers to frame strategies on telemarketing campaigns that will help increase a bank's revenue.

After using eight algorithms, we can say that the XGBoost method is showing the best results among them. And this method can be further used to classify potential customers' responses by predicting whether the client will subscribe to a term deposit or not.

The project also provides information on the process of obtaining the best results when we apply various features selection techniques and balancing techniques like Oversampling and Under sampling, which enables the development of the machine learning models and offers a substantial contribution to the existing literature.

Limitations and Future Prospect

- The base models could have been better tunned considering the computational capacity of the systems on which the models were deployed.
- SMOTE analysis is not very practical for high dimensional data and is not a convenient way to treat imbalance which can be better resolved with more data from the domain.

- The Random Forest algorithm requires very high computational resources for hyper parameter tuning and the model considered has not been fine tunned due to lack of time constraint.
- Complex algorithms like neural networks can be deployed.
- The XGBoost Model can be further fine tunned to enhance the performance.
- The model has very high potential to predict accurate results with an error rate of 5 wrong predictions(false negatives and false positives) out of 100 samples taken for consideration.
- The model can be further improved with more detailed analysis and deployed on a cloud based environment like AWS, Microsoft Azure, Heroku etc.

References

- https://paginas.fe.up.pt/~ec/files_0405/slides/02%20CRISP.pdf
- https://en.wikipedia.org/wiki/Cross-industry_standard_process_for_data_mining
- https://archive.ics.uci.edu/ml/datasets/bank+marketing
- https://towardsdatascience.com/developing-a-multi-layer-perceptron-for-a-bank-marketing-dataset-f754deee49fb
- https://stackoverflow.com/
- https://www.geeksforgeeks.org/python-programming-language/

Acknowledgement

Firstly, I would like to express my sincere gratitude to my mentor Mrs Anjana Agarwal for helping and guiding me throughout the entire duration of this project without which it would have been impossible to deliver such accurate results within the stipulated time period.

Moreover, I would like to express my special thanks to Great Learning who gave me this golden opportunity to learn so many interesting concepts and apply them on such an important topic as a capstone project. This journey has made me more inquisitive about the never ending possibilities with Machine learning and Data science in the near future and has helped me to successfully identify the path which I would like to pursue as my profession.

Last but not the least, I would like to extend my warm wishes and thanks to all my fellow team mates for their commitments, encouragements and continued support during the course of this project.