PREDICTING THE BANK MARKETING CAMPAIGN SUCCESS

Background and Motivation:

The primary business problem addressed in this project is to optimize the direct marketing campaigns conducted by a Portuguese banking institution. These campaigns were based on phone calls. Often, more than one contact with the same client was required, to assess if the product (bank term deposit) would be subscribed to or not.

This problem falls under the domain of supervised learning. The dataset provides labeled instances, indicating whether a client subscribed to the bank term deposit “Yes” or “No”. By utilizing this labeled data, supervised learning algorithms can be trained to predict future outcomes based on similar customer characteristics and interaction patterns. The goal is to develop a predictive model that can classify potential clients into subscription categories, enabling the bank to focus its resources on individuals more likely to subscribe to the term deposit.

Direct marketing campaigns are resource-intensive and require strategic planning to ensure a high return on investment. Understanding the factors influencing customers' subscription decisions is crucial for optimizing these campaigns. By leveraging machine learning algorithms, the bank can analyze historical data to identify patterns and trends that indicate potential subscribers. This predictive analysis can significantly enhance the efficiency of the marketing efforts.

Data Exploration and Pre-processing:

Data Source: https://www.kaggle.com/datasets/henriqueyamahata/bank-marketing/data

The dataset has 21 columns and 42K rows/observations. Following are some of the attributes from the data:

S. No.	Variables	Type	Description
1	age	int64	Age Of Individual
2	job	object	Type Of Job
3	marital	object	Marital Status
4	education	object	Education Level
5	default	object	Has Credit in Default?
6	housing	object	Has Housing Loan?
7	loan	object	Has Personal Loan?
8	contact	object	Contact Communication Type
9	month	object	Last Contact Month of Year
10	day_of_week	object	Last Contact Day of The Week
11	duration	int64	Last Contact Duration, In Seconds
12	campaign	int64	Number Of Contacts Performed During This Campaign for This Client
13	pdays	int64	Number Of Days That Passed by After the Client Was Last Contacted from A Previous Campaign
14	previous	int64	Number Of Contacts Performed Before This Campaign for This Client
15	poutcome	object	Outcome Of the Previous Marketing Campaign
16	emp.var.rate	float64	Employment Variation Rate - Quarterly Indicator
17	cons.price.idx	float64	Consumer Price Index - Monthly Indicator
18	cons.conf.idx	float64	Consumer Confidence Index - Monthly Indicator
19	euribor3m	float64	Euribor 3 Month Rate - Daily Indicator
20	nr.employed	float64	Number Of Employees - Quarterly Indicator
21	y	object	Has The Client Subscribed a Term Deposit?

Data Exploration and Pre-processing:

Cleaning:

- We checked for duplicate records in the data and found 12 duplicate records which we removed from the dataset.

```
In [6]: # Duplicates data check

        df.duplicated().sum()

Out[6]: 12

In [7]: # Removing duplicated records

        df_clean = df.drop_duplicates()

        df_clean
```

Out[7]:

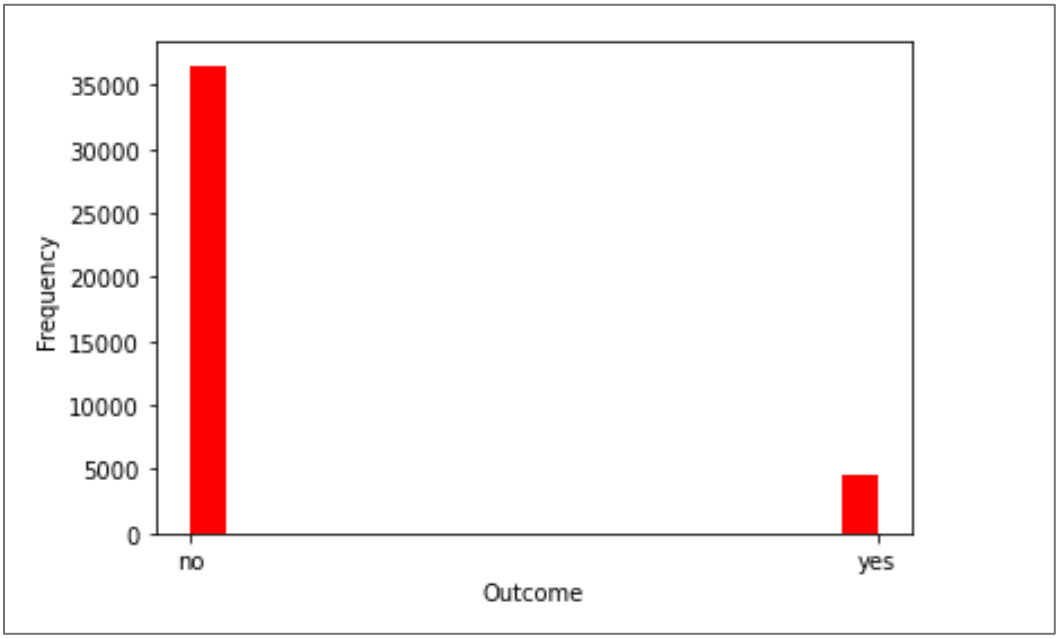
	age	job	marital	education	default	housing	loan	contact	month	day_of_week	...	campaign	pdays	previous	poutcome	emp.va
0	56	housemaid	married	basic.4y	no	no	no	telephone	may	mon	...	1	999	0	nonexistent	
1	57	services	married	high.school	unknown	no	no	telephone	may	mon	...	1	999	0	nonexistent	
2	37	services	married	high.school	no	yes	no	telephone	may	mon	...	1	999	0	nonexistent	
3	40	admin.	married	basic.6y	no	no	no	telephone	may	mon	...	1	999	0	nonexistent	
4	56	services	married	high.school	no	no	yes	telephone	may	mon	...	1	999	0	nonexistent	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
41183	73	retired	married	professional.course	no	yes	no	cellular	nov	fri	...	1	999	0	nonexistent	
41184	46	blue-collar	married	professional.course	no	no	no	cellular	nov	fri	...	1	999	0	nonexistent	
41185	56	retired	married	university.degree	no	yes	no	cellular	nov	fri	...	2	999	0	nonexistent	
41186	44	technician	married	professional.course	no	no	no	cellular	nov	fri	...	1	999	0	nonexistent	
41187	74	retired	married	professional.course	no	yes	no	cellular	nov	fri	...	3	999	1	failure	

41176 rows × 21 columns

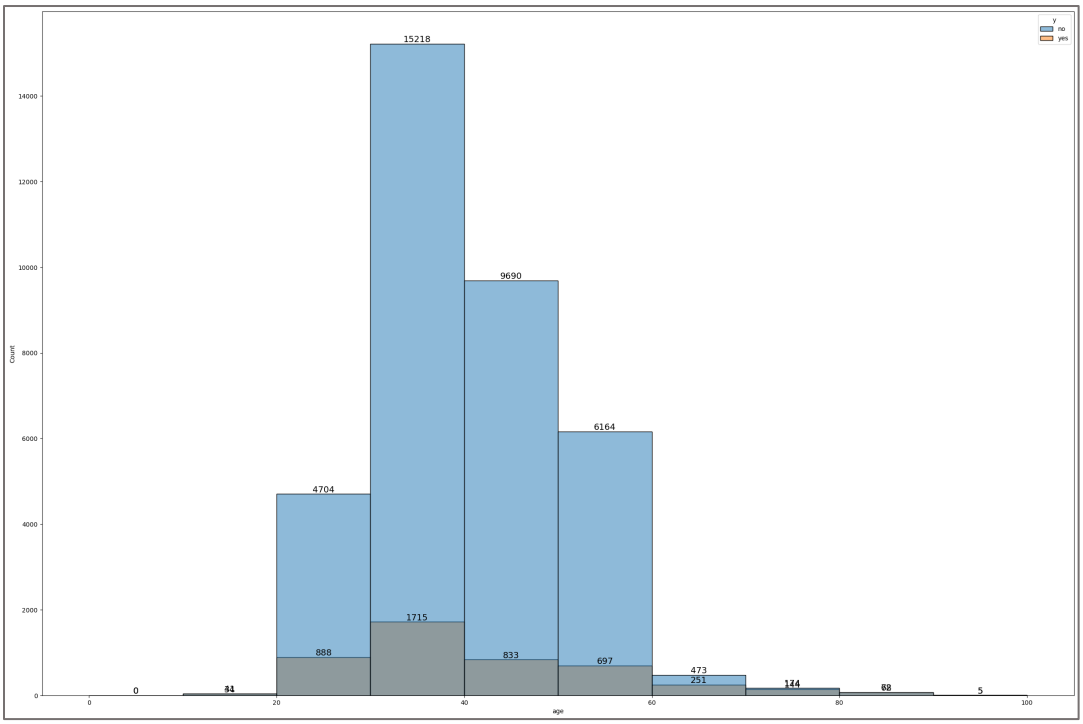
- We checked for missing entries but there were no missing entries in the dataset.

Exploration:

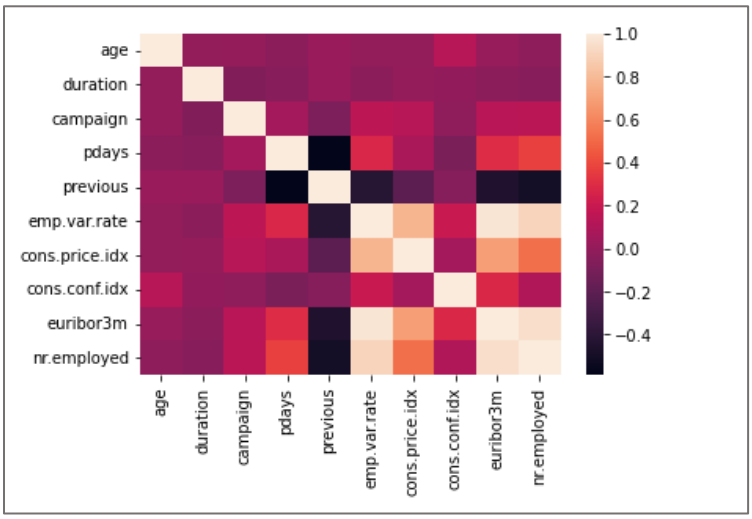
- While checking if the data is balanced or not, we found that our data is highly unbalanced. The target variable “y” has 89% of no and 11% of yes as the response.



- Age groups from 20-60 were the most contacted and the highest positive count was from 30-40. However, the positive response was barely 20%. Teenagers or senior citizens are the customers who gave the most positive outcomes (more than 50%).

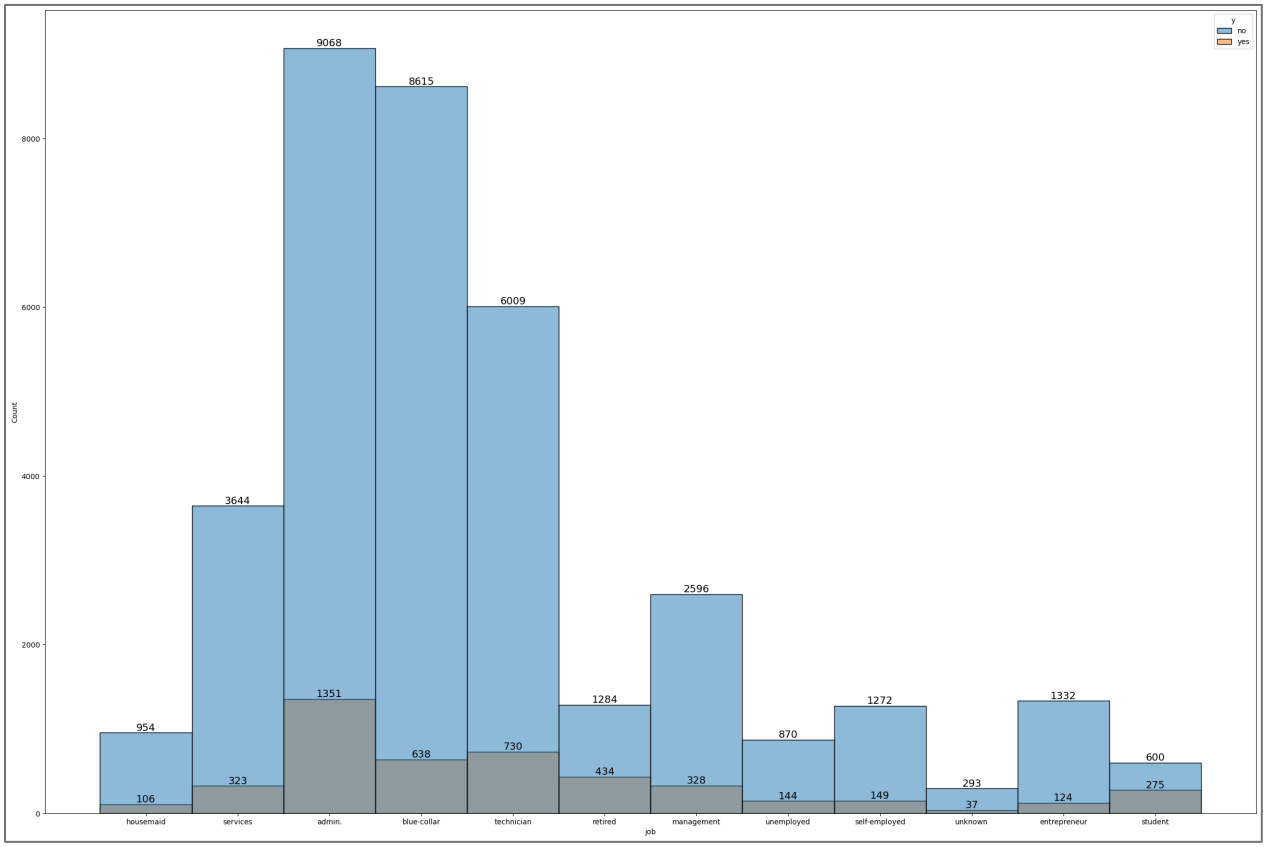


- While checking for correlation we found that the number of contacts made before this campaign and number of days past by after contacting this client before this campaign has a negative correlation.
- Socio economic variables such as employment variation rate and Euribor has a strong positive correlation between them.



	age	duration	campaign	pdays	previous	emp.var.rate	cons.price.idx	cons.conf.idx	euribor3m	nr.employed
age	1.000000	-0.000808	0.004622	-0.034381	0.024379	-0.000242	0.001009	0.129075	0.010852	-0.017607
duration	-0.000808	1.000000	-0.071765	-0.047556	0.020600	-0.027941	0.005303	-0.008126	-0.032861	-0.044672
campaign	0.004622	-0.071765	1.000000	0.052606	-0.079182	0.150786	0.127826	-0.013657	0.135169	0.144129
pdays	-0.034381	-0.047556	0.052606	1.000000	-0.587508	0.271063	0.078920	-0.091374	0.296946	0.372659
previous	0.024379	0.020600	-0.079182	-0.587508	1.000000	-0.420587	-0.203197	-0.050929	-0.454571	-0.501411
emp.var.rate	-0.000242	-0.027941	0.150786	0.271063	-0.420587	1.000000	0.775293	0.196257	0.972244	0.906949
cons.price.idx	0.001009	0.005303	0.127826	0.078920	-0.203197	0.775293	1.000000	0.059170	0.688180	0.521945
cons.conf.idx	0.129075	-0.008126	-0.013657	-0.091374	-0.050929	0.196257	0.059170	1.000000	0.277864	0.100679
euribor3m	0.010852	-0.032861	0.135169	0.296946	-0.454571	0.972244	0.688180	0.277864	1.000000	0.945146
nr.employed	-0.017607	-0.044672	0.144129	0.372659	-0.501411	0.906949	0.521945	0.100679	0.945146	1.000000

- People with Admin, Blue collar and technician jobs were highly contacted, and the highest positive response was from Admin and technician. However, Students were least contacted though they had maximum positive outcome of 50%. Percentagewise, students and retired people had the highest positive outcome. The above statement also supports the age graph.



Preprocessing Steps:

- We have started with finding all the unique values in all character variables which are shown in the below table.

	job	marital	education	default	housing	loan	contact	month	day_of_week	poutcome	y
0	housemaid	married	basic.4y	no	no	no	telephone	may	mon	nonexistent	no
1	services	single	high.school	unknown	yes	yes	cellular	jun	tue	failure	yes
2	admin.	divorced	basic.6y	yes	unknown	unknown	None	jul	wed	success	None
3	blue-collar	unknown	basic.9y	None	None	None	None	aug	thu	None	None
4	technician	None	professional.course	None	None	None	None	oct	fri	None	None
5	retired	None	unknown	None	None	None	None	nov	None	None	None
6	management	None	university.degree	None	None	None	None	dec	None	None	None
7	unemployed	None	illiterate	None	None	None	None	mar	None	None	None
8	self-employed	None	None	None	None	None	None	apr	None	None	None
9	unknown	None	None	None	None	None	None	sep	None	None	None
10	entrepreneur	None	None	None	None	None	None	None	None	None	None
11	student	None	None	None	None	None	None	None	None	None	None

- Later we used the one-hot encoding for the categorical variables and used standard scaling for all the numerical variables.

	age	duration	campaign	pdays	previous	emp.var.rate	cons.price.idx	cons.conf.idx	euribor3m	nr.employed	...	month_oct	month_sep	day
0	1.533143	0.010352	-0.565963	0.195443	-0.349551	0.648101	0.722628	0.886568	0.712463	0.331695	...	0	0	
1	1.629107	-0.421577	-0.565963	0.195443	-0.349551	0.648101	0.722628	0.886568	0.712463	0.331695	...	0	0	
2	-0.290177	-0.124626	-0.565963	0.195443	-0.349551	0.648101	0.722628	0.886568	0.712463	0.331695	...	0	0	
3	-0.002284	-0.413864	-0.565963	0.195443	-0.349551	0.648101	0.722628	0.886568	0.712463	0.331695	...	0	0	
4	1.533143	0.187751	-0.565963	0.195443	-0.349551	0.648101	0.722628	0.886568	0.712463	0.331695	...	0	0	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
41183	3.164534	0.291876	-0.565963	0.195443	-0.349551	-0.752402	2.058076	-2.225059	-1.495197	-2.815689	...	0	0	
41184	0.573501	0.480845	-0.565963	0.195443	-0.349551	-0.752402	2.058076	-2.225059	-1.495197	-2.815689	...	0	0	
41185	1.533143	-0.267317	-0.204990	0.195443	-0.349551	-0.752402	2.058076	-2.225059	-1.495197	-2.815689	...	0	0	
41186	0.381573	0.708379	-0.565963	0.195443	-0.349551	-0.752402	2.058076	-2.225059	-1.495197	-2.815689	...	0	0	
41187	3.260499	-0.074492	0.155984	0.195443	1.670821	-0.752402	2.058076	-2.225059	-1.495197	-2.815689	...	0	0	

41176 rows × 64 columns

- We split our data set into 30% test and 70% train with random state as 22, using the train_test_split package from sklearn library.

- Since our data is unbalanced to prepare it for our classification models, we use SMOTE technique to balance this data set using the oversampling technique.

Models and Performance Evaluation:

Since our data belongs to the banking marketing campaign, we will be focusing on correct classification of the target variables. Hence the performance metric that we are looking at will be the accuracy score.

1. **Support Vector Machine:**

   We performed a linear SVM model where we did the hyperparameter tuning using the values for C as "[0.001, 0.01, 0.1, 1, 10, 100, 100000]".

   Our goal for incorporating these C values was that smaller C values will have soft margins and the higher C values will have hard margins which will penalize the misclassification.

   After mentioning the values for C we used GridsearchCV with 5 fold cross validation and found the best parameter as C = 0.001.

   Using this best parameter value we performed a linear SVM model and found these results:

```
SVM Model Train Accuracy is: 0.9207669478288035
SVM Model Test Accuracy is: 0.8739577430583664
[[9612 1317]
 [ 240 1184]]
TP is: 1184
TN is: 9612
FP is: 1317
FN is: 240
```

```
Precision score: 0.47341063574570175
Recall score: 0.8314606741573034
Accuracy score: 0.8739577430583664
F1 score: 0.6033121019108281
```

   Hence the accuracy for this linear SVM model with best parameter is 87.40%.

2. **Naïve Bayes:**

   Here we are using Gaussian Naïve Bayes model for the classification and found the following results.

```
Naive Bayes Model Train Accuracy is: 0.7719657919400188
Naive Bayes Model Test Accuracy is: 0.659839715048976
[[7096 3833]
 [ 369 1055]]
TP is: 1055
TN is: 7096
FP is: 3833
FN is: 369
Precision score: 0.21583469721767595
Recall score: 0.7408707865168539
Accuracy score: 0.659839715048976
F1 score: 0.3342839036755386
```

The Accuracy of the Naïve Bayes model comes out to be 65.98%.

3. **KNN Model:**

We performed KNN classifier using the Euclidean distance measure and tuned the model with number of neighbors from 1 to 25 using 5-fold cross validation. We found the 2 as the best parameter and then incorporated that in our model.

```
KNN Model Train Accuracy is: 0.993439550140581
KNN Model Test Accuracy is: 0.878491054804501
[[10085   844]
 [  657   767]]
TP is: 767
TN is: 10085
FP is: 844
FN is: 657
Precision score: 0.4761018001241465
Recall score: 0.538623595505618
Accuracy score: 0.878491054804501
F1 score: 0.5054365733113675
```

The accuracy from this model comes out to be 87.85%.

4. **Logistic Regression:**

We performed a basic logistic regression model on our target variable and found the accuracy score as 90.98%, which in fact is the best among all the models that we tested.

```
Log reg Train Accuracy is: 0.9432599187753827
Log reg Test Accuracy is: 0.9098194770501092
```

```
[[10555   374]
 [  740   684]]
TP is: 684
TN is: 10555
FP is: 374
FN is: 740
```

5. **Decision Tree:**

We performed a decision tree classifier with default parameters and criterion as entropy and max depth as 4. The accuracy comes out be 80.51%.
Later we did the hyperparameter tuning for this model taking max depth from 1 to 10, min_sample_split from 2 to 10 and max_leaf_node from 2 to 10.
Although we understand that these choices of hyperparameters are not accurate for our data set but because of the lack of technical resources, we are using these. If given favorable conditions, we could have used max depth from 1 to 50, min_sample_split from 2 to 500 and max_leaf_node from 10 to 1000.
Using default cross validation, we found the best parameters as the following:

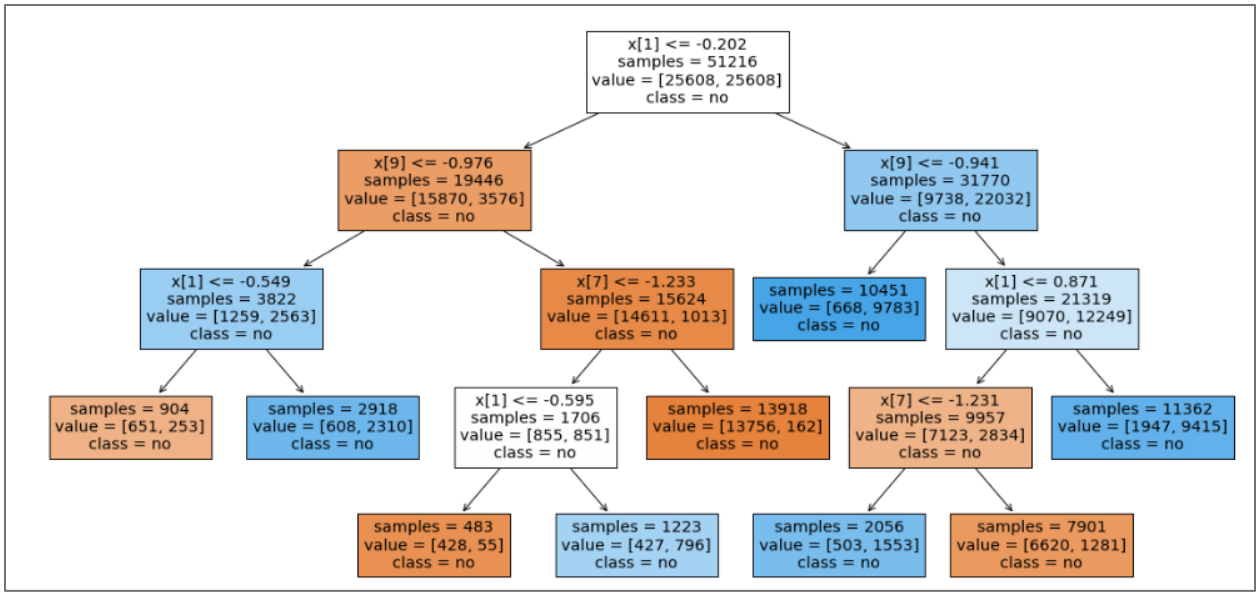
```
grid_tree.best_params_
```

```
{'max_depth': 4, 'max_leaf_nodes': 9, 'min_samples_split': 2}
```

Using these parameters, we found the accuracy of the model as 80.50%

```
DT Train accuracy score:  0.8767963136519837
DT Test accuracy score:  0.8047437869343479
[[8637 2292]
 [ 120 1304]]
TP is: 1304
TN is: 8637
FP is: 2292
FN is: 120
Precision score: 0.36262513904338156
Recall score: 0.9157303370786517
Accuracy score: 0.8047437869343479
F1 score: 0.5195219123505976
```

The tree from these best parameters model will be visualized like this:



6. Ensemble Methods:

Using the ensemble methods we have used decision tree and SVM model as our base model and applied bagging and adaptive boost on them.

We use decision tree if there is high variance in the data and SVM because they focus on the points that are hardest to classify (support vectors), which can be a desirable property in an ensemble method, as these are the instances where gaining accuracy is most beneficial.

a. Decision tree Bagging

Using bagging technique on decision tree classifier with n_estimators = 200 and max_samples = 100, we secured a test accuracy of 84.42%.

```
DT with Bagging Train Accuracy:  0.892260231177757
DT with Bagging Test Accuracy:  0.8441674087266251
[[9155 1774]
 [ 151 1273]]
TP is: 1273
TN is: 9155
FP is: 1774
FN is: 151
Precision score: 0.4177879881851001
Recall score: 0.8939606741573034
Accuracy score: 0.8441674087266251
F1 score: 0.5694475508834713
```

b. SVM Bagging:

Using the bagging technique on our best SVM model which we achieved using hyperparameter tuning earlier, we secured an accuracy score of 89.85%.

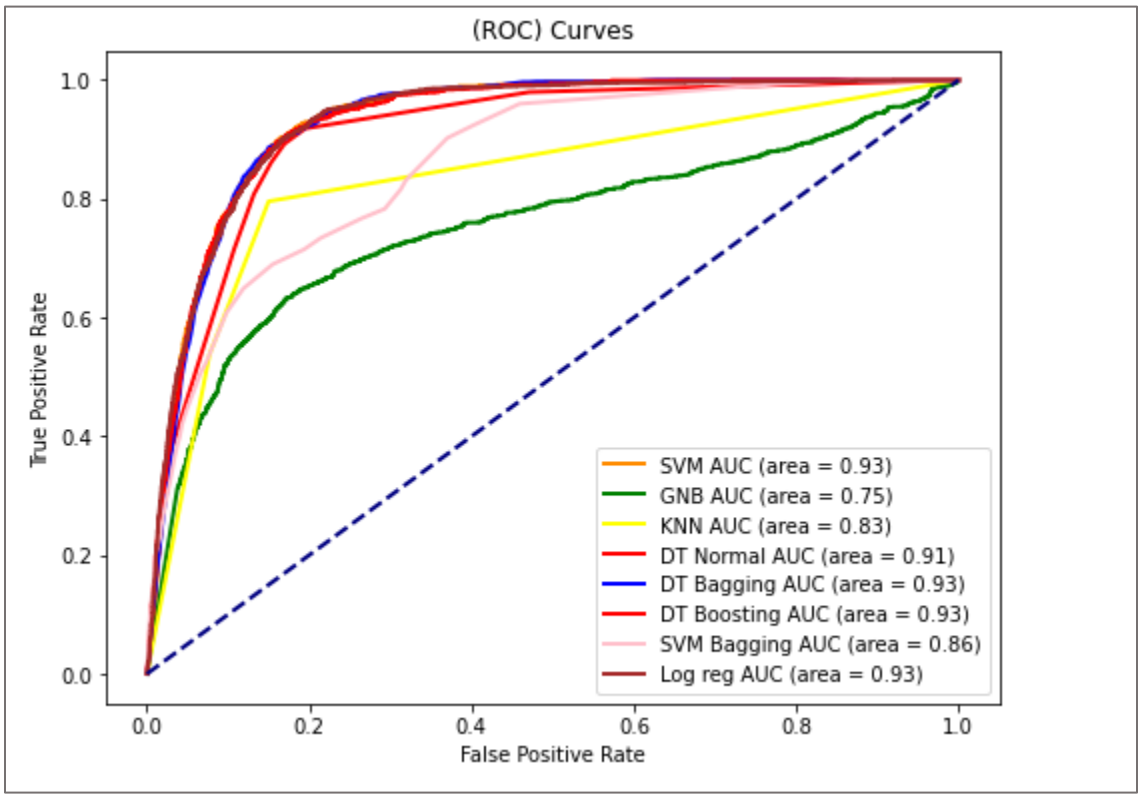
```
SVM with Bagging Train Accuracy:  0.5933497344579819
SVM with Bagging Test Accuracy:  0.8984861976847729
[[10819   110]
 [ 1144   280]]
TP is: 280
TN is: 10819
FP is: 110
FN is: 1144
Precision score: 0.717948717948718
Recall score: 0.19662921348314608
Accuracy score: 0.8984861976847729
F1 score: 0.308710033076075
```

c. Decision Tree Adaboost:

Using the adaptive boost technique on our decision tree classifier model we secured an accuracy score of 90.20%.

```
DT Adaptive Boost Train Accuracy:  0.9366994689159638
DT Adaptive Boost Test Accuracy:  0.9020480854853072
[[10187   742]
 [  468   956]]
TP is: 956
TN is: 10187
FP is: 742
FN is: 468
Precision score: 0.5630153121319199
Recall score: 0.6713483146067416
Accuracy score: 0.9020480854853072
F1 score: 0.612427930813581
```

Conclusion:



S. No.	Models	AUC %	Train Accuracy %	Test Accuracy %
1	Support Vector Machine	93	92.08	87.4
2	Naive Bayes	75	77.2	65.98
3	KNN Model	83	99.34	87.85
4	**Logistic Regression**	**93**	**94.32**	**90.98**
5	Decision Tree	91	87.68	80.5
6	DT Bagging	93	89.23	84.42
8	SVM Bagging	86	59.33	89.85
9	DT Adaboost	93	93.67	90.2

The above table describes that the ***Logistic regression model*** has the best accuracy of 91% and best area under the curve of 93%. Hence, using both performance metrics we can say that logistic regression model is the best model for our classification problem.

*****End of Document*****