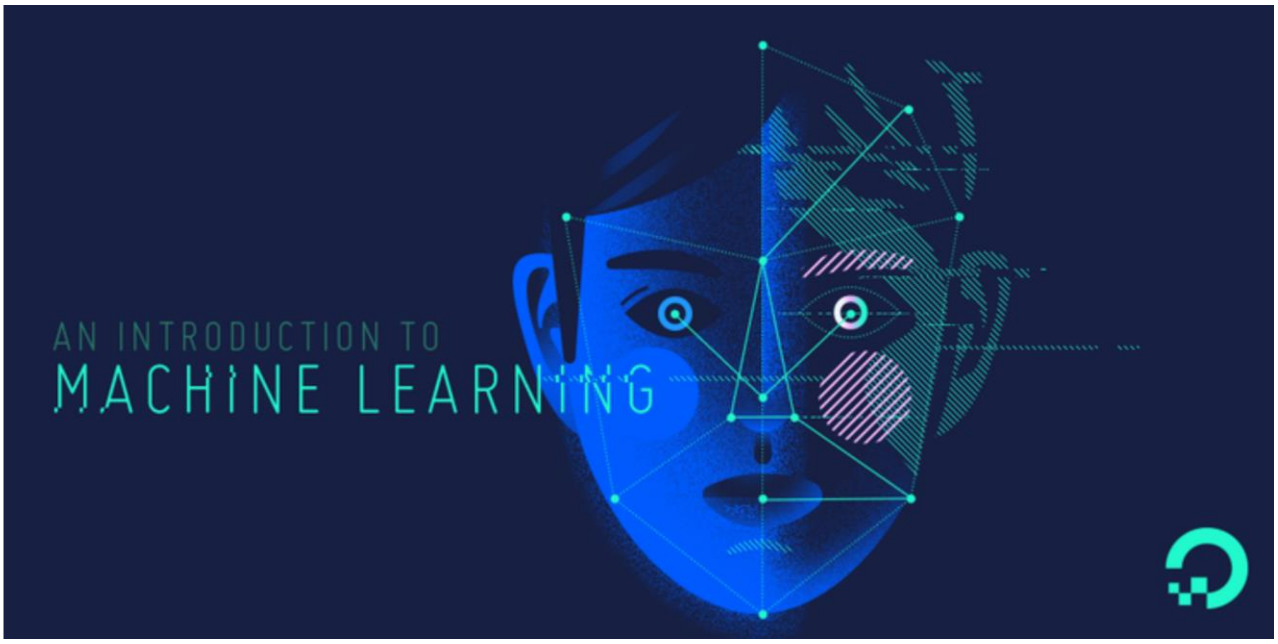
EE559: Project Report

ISHAN MOHANTY
USC ID: 4461-3447-18
Email: imohanty@usc.edu



I. Abstract and Motivation

For this project, we are utilizing the bank-additional.csv data which describes the marketing campaigns undertaken by a certain Portuguese banking firm. Our aim here would be to analyse the dataset, select appropriate features that would aid in proper classification of training data and finally evaluate our model that we employed for different classifiers using F1 score and ROC-AUC score and plot the curves. Essentially, we are getting an experience into the Machine learning world by building an end-to-end project. Hence, we put in all our knowledge to build a model that would predict whether a customer will take a term deposit in the Portuguese bank or not. Our model if well done could be very useful to the financial industry which heavily rely on machine learning models to gather useful information in order to advertise and target customers and clients.

II. Approach and Procedures

We use the following approach to build a machine learning model on our bank-additional dataset:

1. Pre-processing the dataset
2. Feature Engineering
3. Cross validation or Data Modelling
4. Training and Classification
5. Model Evaluation Metrics

1.Pre-processing

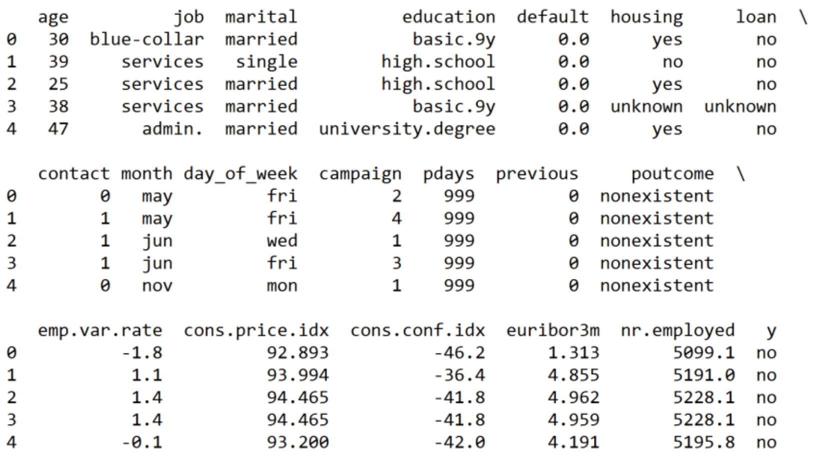
1. Acquire data

We import the bank-additional data from the pandas-library using python. The data in the csv contains the 19 essential attributes along with the output label of whether the customer has made a term deposit or not. The label is binary in nature.

We can get a preview of the dataset by executing the command:

```
display(df.head())
```

We can see this data in the next page.



```
   age          job  marital          education  default  housing     loan  \
0   30  blue-collar  married           basic.9y      0.0      yes       no
1   39     services   single        high.school      0.0       no       no
2   25     services  married        high.school      0.0      yes       no
3   38     services  married           basic.9y      0.0  unknown  unknown
4   47       admin.  married  university.degree      0.0      yes       no

   contact month day_of_week  campaign  pdays  previous     poutcome  \
0        0   may         fri         2    999         0  nonexistent
1        1   may         fri         4    999         0  nonexistent
2        1   jun         wed         1    999         0  nonexistent
3        1   jun         fri         3    999         0  nonexistent
4        0   nov         mon         1    999         0  nonexistent

   emp.var.rate  cons.price.idx  cons.conf.idx  euribor3m  nr.employed   y
0          -1.8          92.893          -46.2      1.313       5099.1  no
1           1.1          93.994          -36.4      4.855       5191.0  no
2           1.4          94.465          -41.8      4.962       5228.1  no
3           1.4          94.465          -41.8      4.959       5228.1  no
4          -0.1          93.200          -42.0      4.191       5195.8  no
```

Figure 1: Dataset Info for Visualization

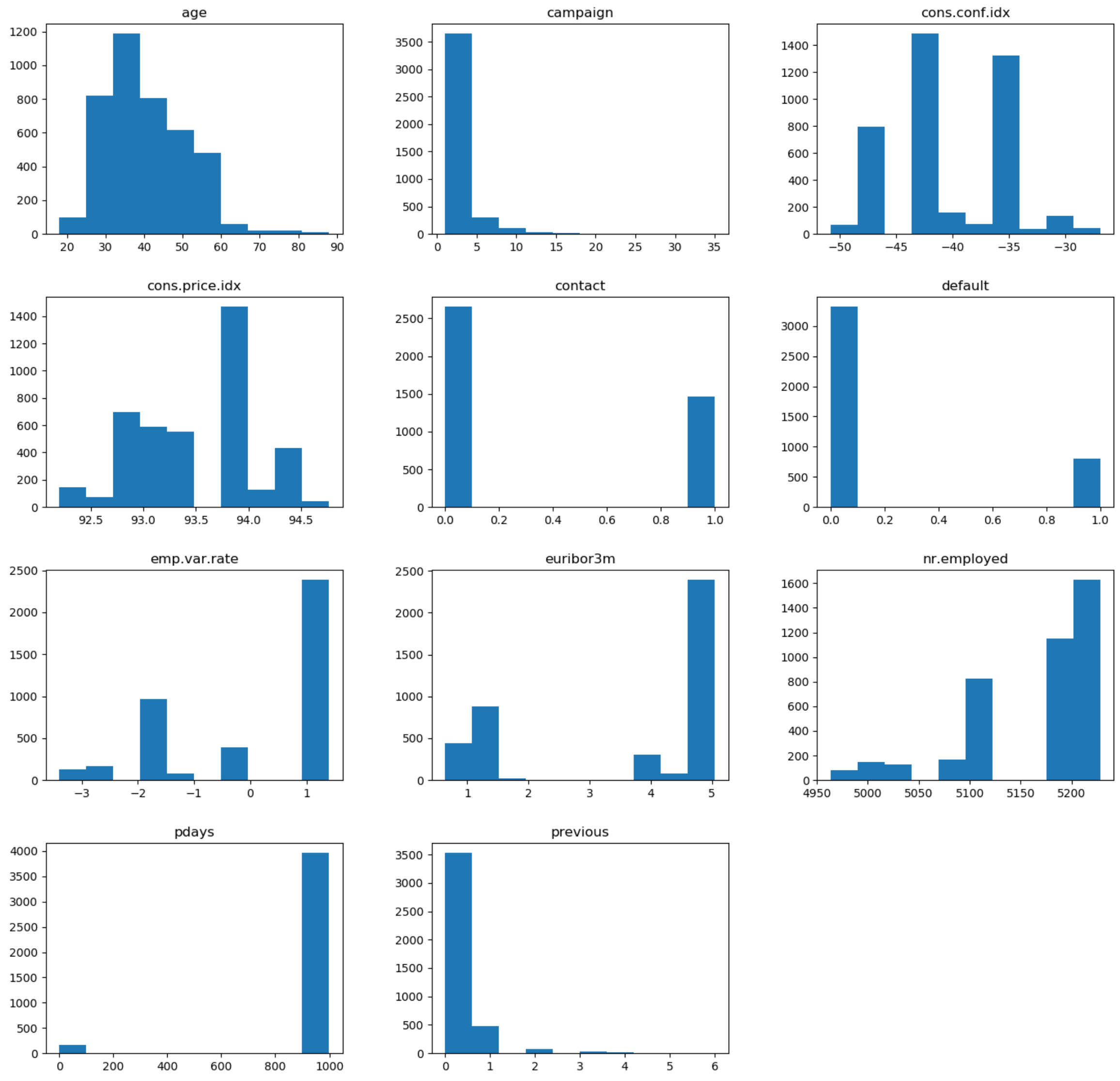


Figure 2: Histogram for the different continuous or numerical features

We can visualize the numerical or continuous data of the above features from the above histogram.

2. Handling Missing values

In our bank-data set we have no missing values but we do have "**unknown**" values in the following attribute or feature columns. The following features have unknown values in them. So, we use two strategies to prepare the data set:

- we prepare a data-set treating all unknowns as another category of data. This is our first dataset to consider for further pre-processing. Let us name it as "**Dataset with unknowns**".

- we prepare another dataset removing all the rows containing "**unknown**" values present for the specific features. We also remove the "**default**" column or feature because all the values in the default column has "**no**" as values, hence it will not affect the classification model. We also remove the row containing "**illiterate**" as there is only one of this value in the education feature column. Let us call this "**Dataset without unknowns**"

3. Dealing with Categorical Data:

- Label encoding: for the "dataset without unknowns" let us assign labels 0 or 1 to yes or no for the following columns: loan, housing and cellular contact as they only have 2 categories for the "data set without unknowns". This is strictly done for all features that have only two categories. There is only 1 yes present in the default column hence we treat it as the unknown and make it a two category feature and perform label encoding, this done for the dataset with unknowns.

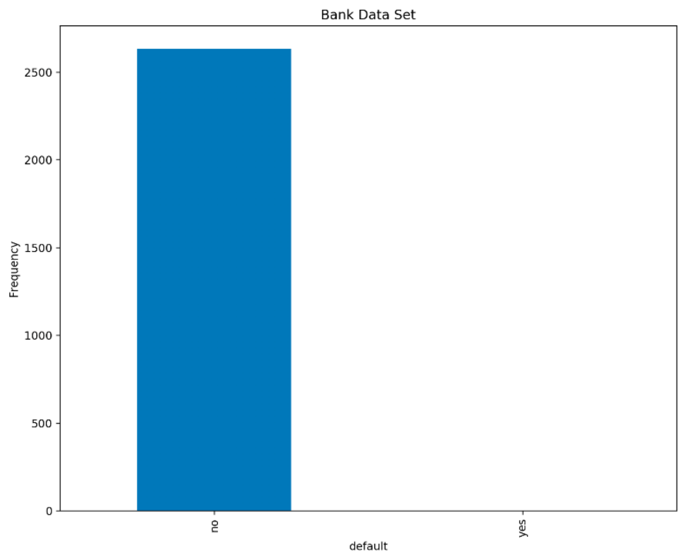


Figure 3: Default feature Visualization

- One Hot encoding: The one hot encoding is done to help Machine learning algorithms process the data without giving the categorical data higher or lower importance mathematically. For example, if we have three countries like france, germany and spain. If we label encode them as 1,2 and 3 respectively, the ML algorithm might give more weight to Spain which has value 3 during internal calculations. Hence, we binarize them and include three columns with '1' if the particular country occurs and '0' if it is absent. Here, we perform One hot encoding on categorical values for both data sets for this purpose.

4. Split Data into Train and Test

At this crucial juncture we set aside 10% of data for Test and the rest of the data we keep for training. We split the data in such a way that every time we run our program we get the split at the same position of the dataset by using the parameter random_state= fixed number (Say 40).

2. Feature Engineering

We perform 2 operations here:

- Correlating Features
- Feature Scaling
- Feature Selection using Random Forest Classifier or extraction

I. Correlating Features

Below we see the heat map of the correlation of various features in our dataset. We need to observe negative and positive correlation among the features. Negative correlation implies increase in one feature decreases the other whereas positive correlation implies increase in one feature increases the other. We would like to observe this phenomenon through the heat map below with respect to the output label (y) of whether a customer made a deposit or not.

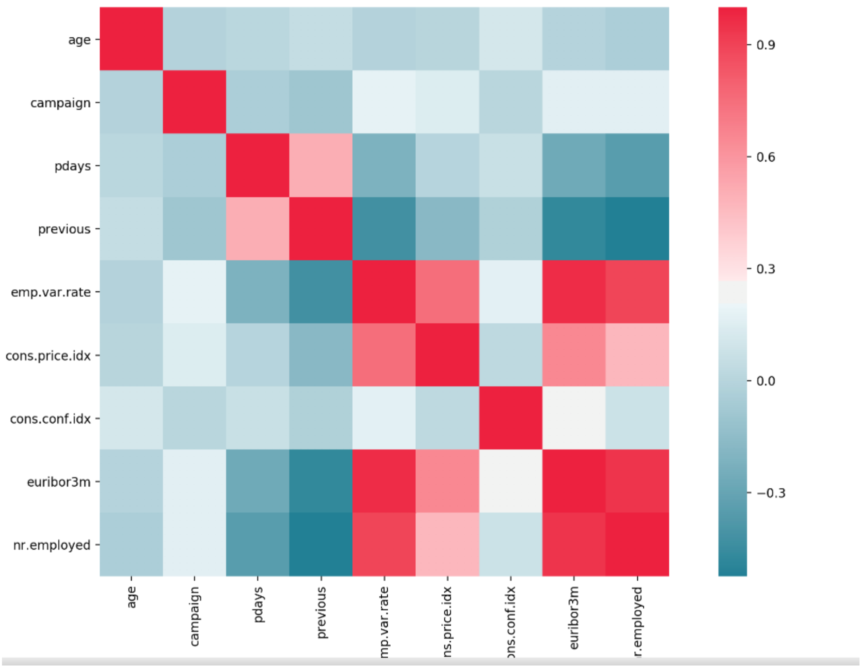


Figure 4: Correlation Heat Map among the features of our dataset

II. Feature Scaling

The raw values of data vary through different ranges which could be problematic in many optimization techniques which look to optimize an objective function. For example, the gradient descent algorithm in machine learning does not converge fast without normalization.

Hence, we perform feature scaling by two methods:

- **Normalization:**
  We normalize the numerical data in the range of 0 to 1 using the min-max scaler from sci-kit learn utility. The formula for the Normalization is given below:

$$x' = \frac{x - \min(x)}{\max(x) - \min(x)}$$

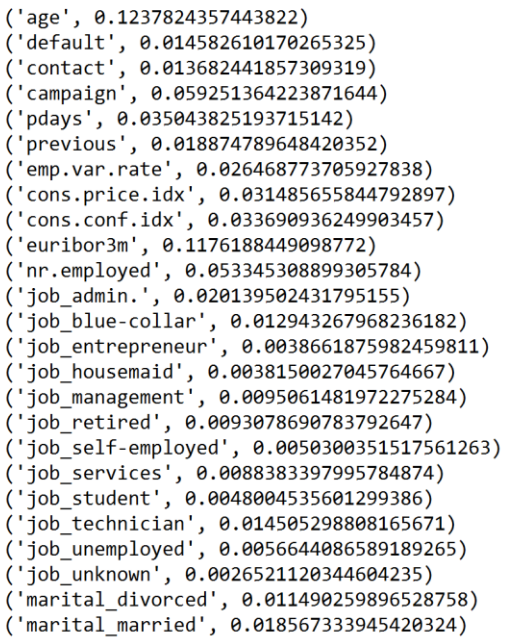
- **Standardization:**

  We transform our datasets to have zero-mean and unit variance. We use Standard-Scaler from sci-kit learn utility to achieve this. The formula for this is given below.

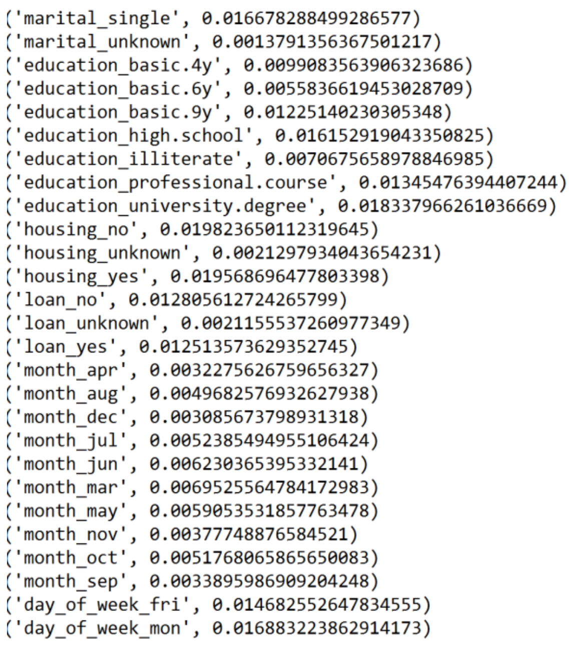
$$x' = \frac{x - \bar{x}}{\sigma}$$

I. Feature Selection using Random Forest Classifier

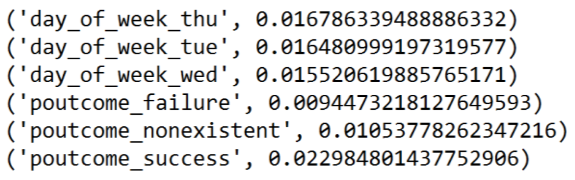
Feature selection helps us make the model better by decreasing the variance of the entire model and hence, preventing any over-fitting phenomenon that might occur. It helps in reducing feature dimensionality and hence, we get the most important-features by not compromising on classification accuracy. Hence, our machine learning model becomes simplified and computational time for training reduces. Therefore, in this section we perform feature selection using an ensemble model known as Random forest classifier. We will describe this classifier in more detail in the classification section. We use the "SelectFromModel" function for thresholding the most important features. Below are some results from "dataset with unknowns".



```
('age', 0.1237824357443822)
('default', 0.014582610170265325)
('contact', 0.013682441857309319)
('campaign', 0.059251364223871644)
('pdays', 0.035043825193715142)
('previous', 0.018874789648420352)
('emp.var.rate', 0.026468773705927838)
('cons.price.idx', 0.031485655844792897)
('cons.conf.idx', 0.033690936249903457)
('euribor3m', 0.1176188449098772)
('nr.employed', 0.053345308899305784)
('job_admin.', 0.020139502431795155)
('job_blue-collar', 0.012943267968236182)
('job_entrepreneur', 0.0038661875982459811)
('job_housemaid', 0.0038150027045764667)
('job_management', 0.0095061481972275284)
('job_retired', 0.0093078690783792647)
('job_self-employed', 0.0050300351517561263)
('job_services', 0.0088383397995784874)
('job_student', 0.0048004535601299386)
('job_technician', 0.014505298808165671)
('job_unemployed', 0.0056644086589189265)
('job_unknown', 0.0026521120344604235)
('marital_divorced', 0.011490259896528758)
('marital_married', 0.018567333945420324)
```

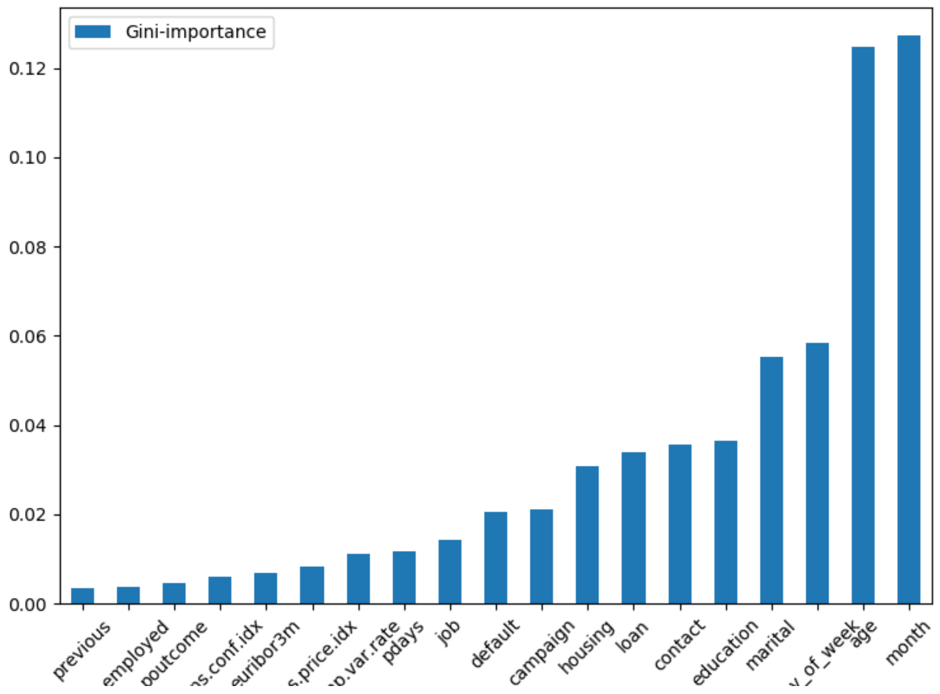


```
('marital_single', 0.016678288499286577)
('marital_unknown', 0.0013791356367501217)
('education_basic.4y', 0.0099083563906323686)
('education_basic.6y', 0.0055836619453028709)
('education_basic.9y', 0.01225140230305348)
('education_high.school', 0.016152919043350825)
('education_illiterate', 0.0070675658978846985)
('education_professional.course', 0.01345476394407244)
('education_university.degree', 0.018337966261036669)
('housing_no', 0.019823650112319645)
('housing_unknown', 0.0021297934043654231)
('housing_yes', 0.019568696477803398)
('loan_no', 0.012805612724265799)
('loan_unknown', 0.0021155537260977349)
('loan_yes', 0.012513573629352745)
('month_apr', 0.0032275626759656327)
('month_aug', 0.0049682576932627938)
('month_dec', 0.003085673798931318)
('month_jul', 0.0052385494955106424)
('month_jun', 0.006230365395332141)
('month_mar', 0.0069525564784172983)
('month_may', 0.0059053531857763478)
('month_nov', 0.00377748876584521)
('month_oct', 0.0051768065865650083)
('month_sep', 0.0033895986909204248)
('day_of_week_fri', 0.014682552647834555)
('day_of_week_mon', 0.016883223862914173)
```

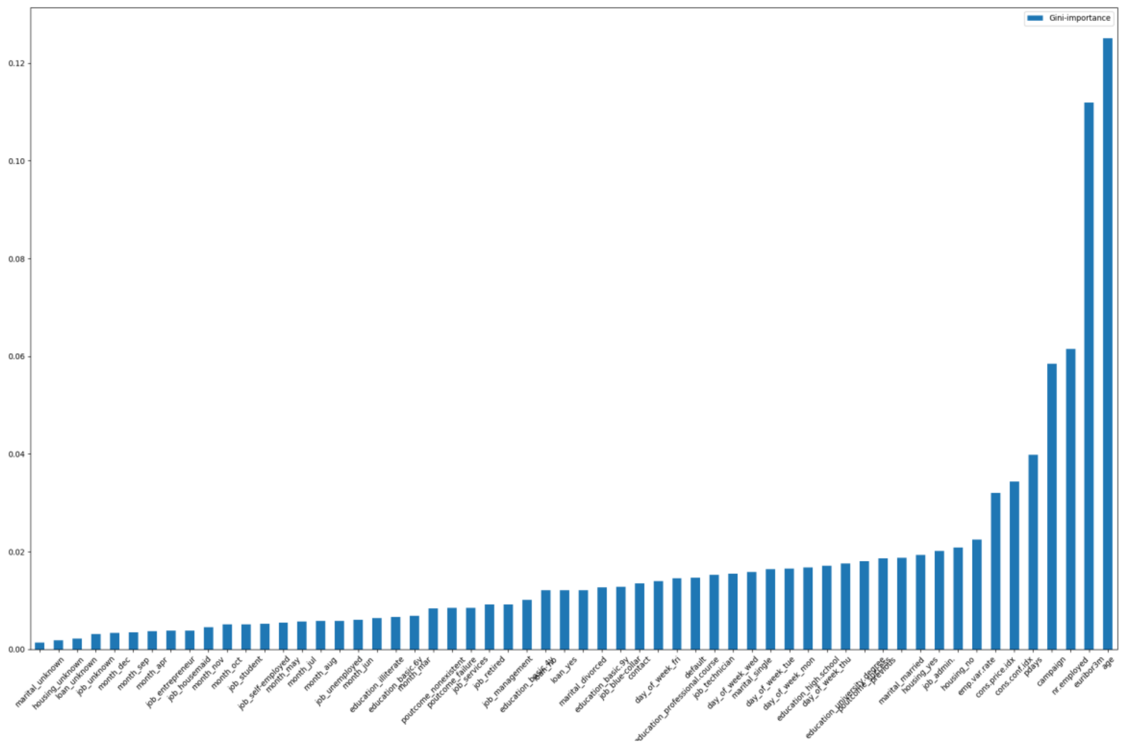


```
('day_of_week_thu', 0.016786339488886332)
('day_of_week_tue', 0.016480999197319577)
('day_of_week_wed', 0.015520619885765171)
('poutcome_failure', 0.0094473218127649593)
('poutcome_nonexistent', 0.01053778262347216)
('poutcome_success', 0.022984801437752906)
```

Figure 5: The Feature importance % values for the "Dataset with unknown"



<u>Figure 6: Final Selected Features for our “Dataset without unknowns” based on importance</u>



<u>Figure 7: Final Selected Features for our “Dataset with unknowns” based on importance</u>

The above figure, gives feature importance which add up to 100% when summed up. From this we use the "SelectFromModel" and a decide to threshold by which we can drop unimportant features and hence, achieve good feature selection.

3. Cross-validation

Cross validation is mainly done in order to generalize the model and avoid overfitting. In this case, we divide the training set into a training subset and a validation set and perform Stratified k-fold cross-validation. Stratifield k-fold cross-validation ensure we have training and validation sets of equal data associated to specific labels throughout the process. In this method we use K=5, predict accuracy on the train set and use this knowledge on the validation test. The Dataset is always shuffled to bring in the flavour of randomness and unpredictability. Then we average across this validation accuracy. Using this knowledge, we get the best parameters like C & Gamma for SVM and "K" value for the K-nearest neighbours. Based on our cross-validation we get the "K" value as 6 for KNN and C= 50 & Gamma= 0.3 for SVM. With these best parameters we perform classification on the test set and evaluate our model.

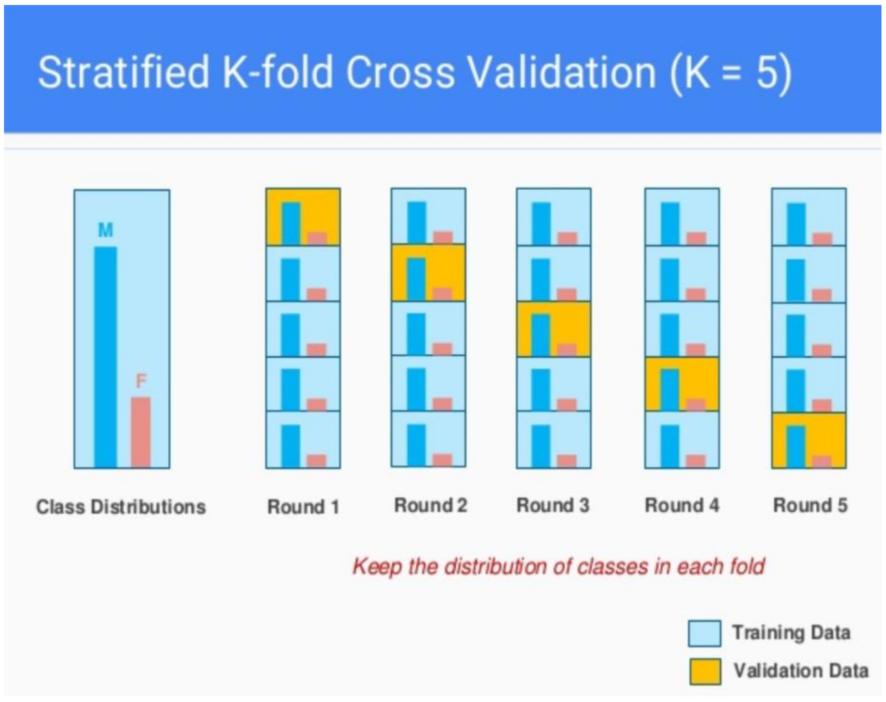

Stratified K-fold Cross Validation (K = 5)
M
F
Class Distributions
Round 1
Round 2
Round 3
Round 4
Round 5
Keep the distribution of classes in each fold
Training Data
Validation Data

Figure 8: Illustration of the Stratified K-fold Cross-Validation process

4. Training and Classification

We use the following classifiers for training:

1. Naive Bayes Classifier

Naive Bayes technique is derived from the Bayes theorem and makes predictions based on probability. It assumes feature independence and is scalable to a large extent. It also works very well with small amount of data samples and has faster convergence than some ML algorithms. The Naïve Bayes mathematical equation is demonstrated below.

$$\underbrace{P(c\mid x)}_{\text{Posterior Probability}} = \frac{\overbrace{P(x\mid c)}^{\text{Likelihood}}\,\overbrace{P(c)}^{\text{Class Prior Probability}}}{\underbrace{P(x)}_{\text{Predictor Prior Probability}}}$$

$$P(c\mid \mathrm{X}) = P(x_1\mid c)\times P(x_2\mid c)\times\cdots\times P(x_n\mid c)\times P(c)$$

Figure 9: Naïve Bayes Mathematical Depiction

2. SVM Classifier

The support vector machine classifier is a parametric or discriminative classifier. The SVM produces the best hyperplane by maximizing the margin to get best separation of data into their respective classes. Here we use the rbf kernel or radial basis function kernel and use the parameters C and Gamma for classification

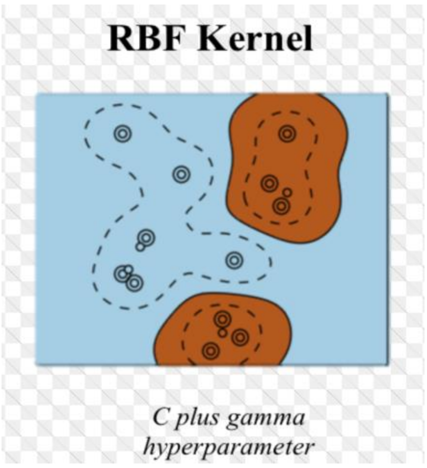


Figure 10: SVM using RBF kernel

3. KNN

K nearest neighbour classifier is a non-parametric based classification algorithm. It decides which data point goes to which class based on the influential decision of majority vote from its neighbours in the feature space. Here “k” was found to be 6 after cross-validation.

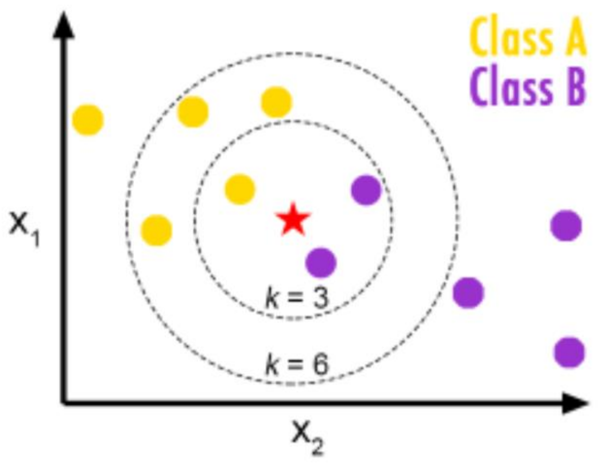


Figure 11: KNN illustration for K=3 & K=6 in feature space

4. Decision tree

Decision tree model is built upon observing a particular data instance and assigning it a class label based on predictive analysis. The data instance observations are known as branches and the classes are known as leaves. They are used heavily in data mining and machine learning algorithms.

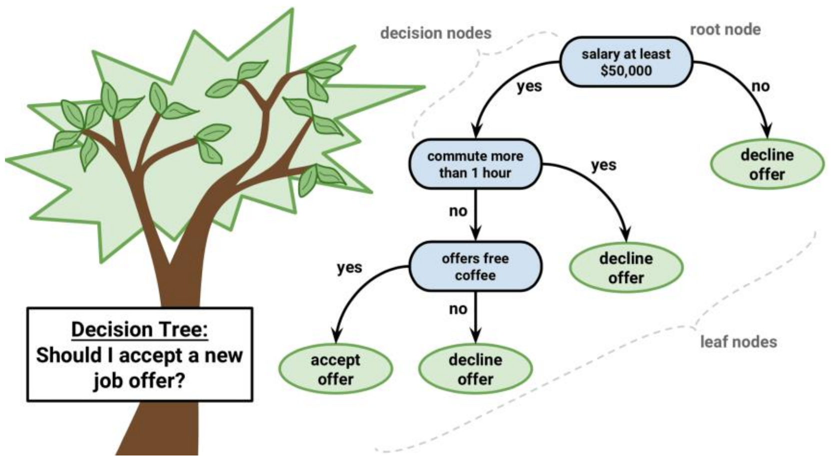


Figure 12: Illustration of a Decision Tree

5. Random Forest Classifier

The Random Forest Classifier is an ensemble learning model that is heavily used for edge detection in image processing to feature selection and Training classification. It is a supervised learning algorithm. Random forests go about their business by fabricating many decision trees during training and use this knowledge to predict or classify the data into their respective class labels attached to individual trees. The random forests improve the decision trees which have a bad property of over-fitting.

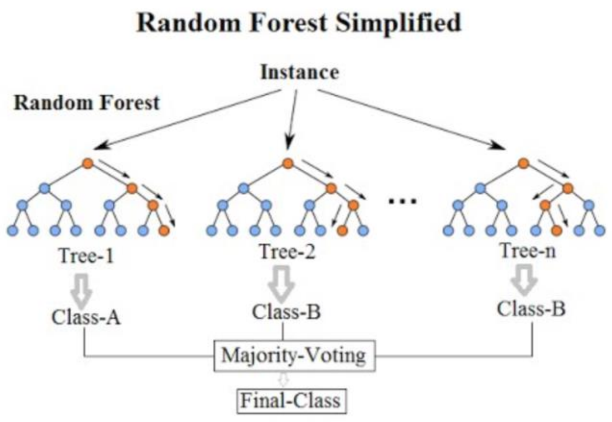


Figure 13: Random Forest Illustration

6. Logistic Regression

Logistic regression is used mainly in medical fields. It classifies data by probabilistically analysing and calculating the binary yield based on one or more features and hence, is known as a binary model.

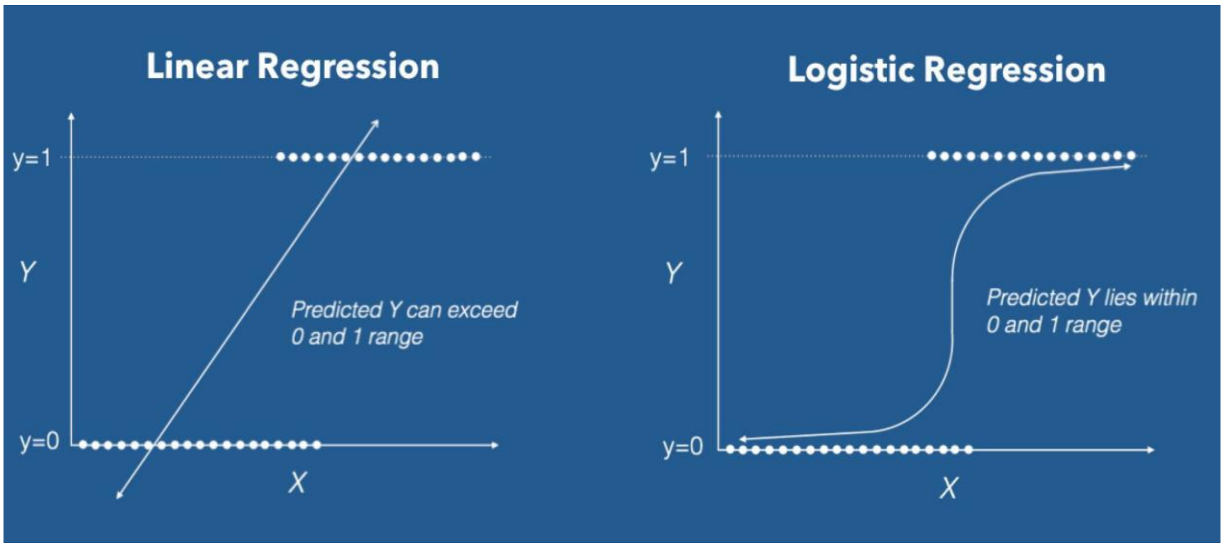

Linear Regression
y=1
Y
y=0
X
Predicted Y can exceed 0 and 1 range
Logistic Regression
y=1
Y
y=0
X
Predicted Y lies within 0 and 1 range

Figure 14: Comparison between Logistic and Linear Regression

5. Model Evaluation Metrics

We use the F1-score and ROC-AUC curve to measure performance of our models. They are explained below.

The F1 score is given by the harmonic mean of precision and recall and measures accuracy of the test data.

$$F_1 = \frac{2}{\frac{1}{\mathrm{recall}} + \frac{1}{\mathrm{precision}}} = 2 \cdot \frac{\mathrm{precision} \cdot \mathrm{recall}}{\mathrm{precision} + \mathrm{recall}}.$$

A Receiver Operating Characteristic (ROC) Curve is a tool by which we can compare the performance of the classifier at various values of thresholds. It is plotted between true positive and false positive. True positive and False positive are given below:

True Positive Rate (**TPR**) is a synonym for recall and is therefore defined as follows:

$$TPR = \frac{TP}{TP + FN}$$

False Positive Rate (**FPR**) is defined as follows:

$$FPR = \frac{FP}{FP + TN}$$

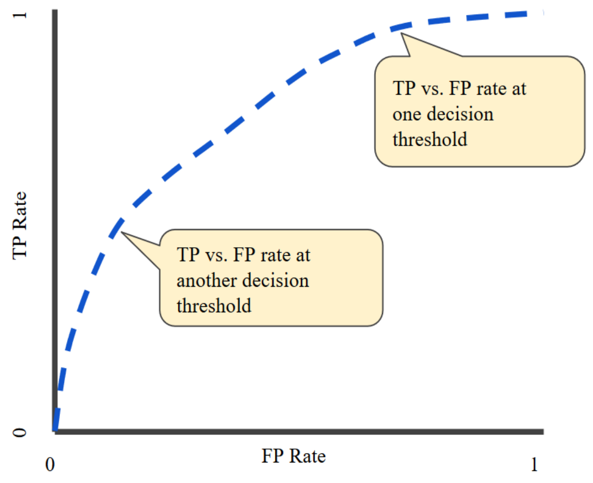


Figure 15: ROC for Different Thresholds

The Area under the Curve or AUC is the area under the ROC curve and fives the cumulative performance of the classifier at different thresholds.

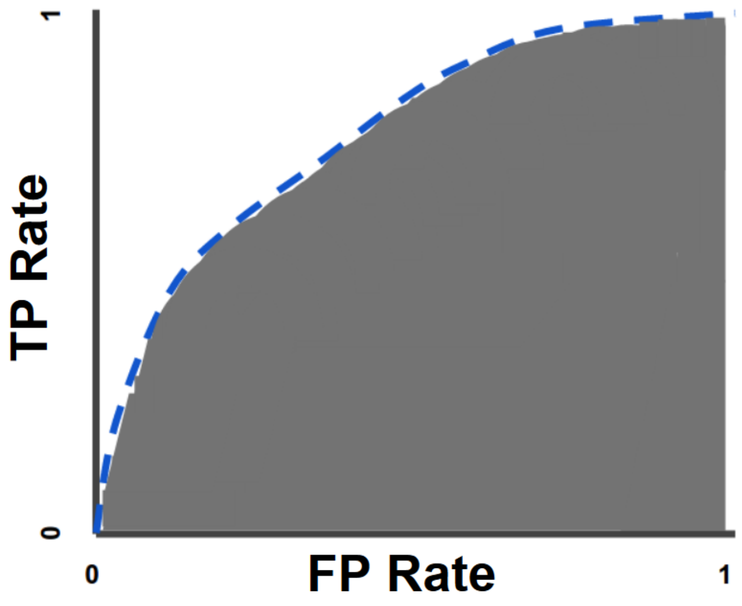
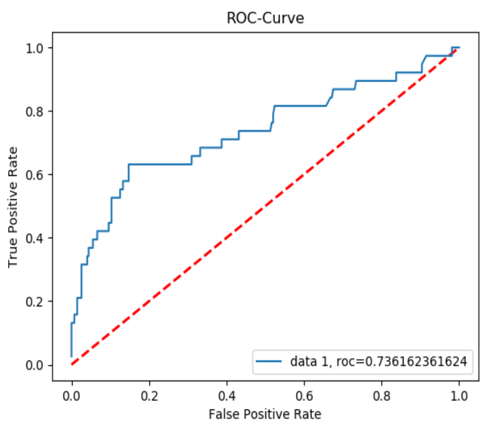


Figure 16: Illustration of Area Under the Curve (AUC)

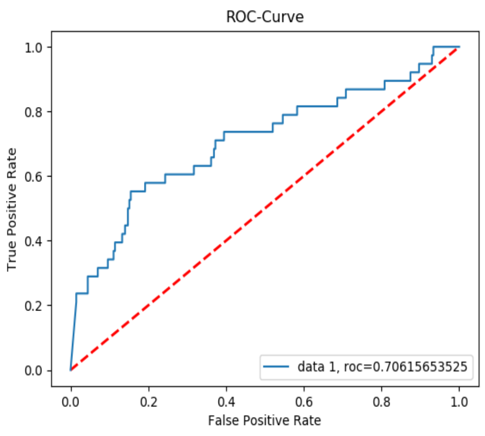
III. Experimental Results

For Dataset without unknowns (Removed during pre-processing):

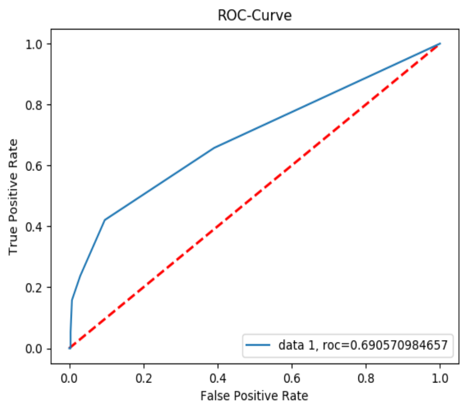
Classifier	Precision	Recall	F1 Score	ROC_AUC Score
Naïve Bayes	0.84	0.81	0.82	0.7061
SVM-RBF	0.87	0.89	0.86	0.7540
Logistic Regression	0.88	0.90	0.87	0.7419
KNN	0.88	0.89	0.86	0.6905
Decision Tree	0.82	0.81	0.82	0.5443
Random Forest	0.85	0.88	0.86	0.7361



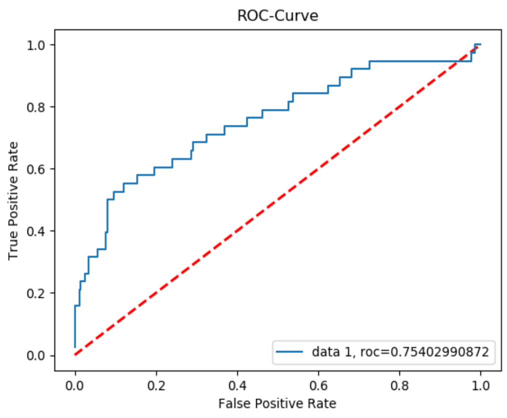
Random forest



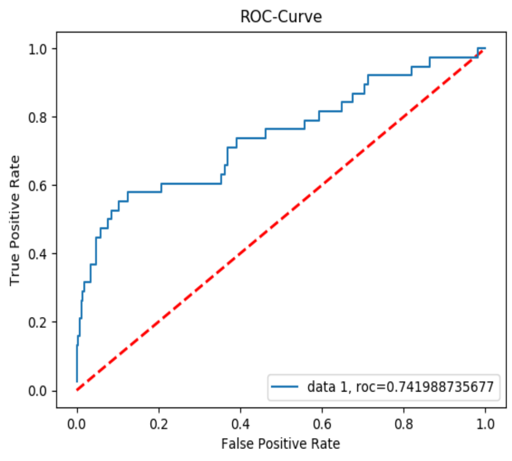
Naïve Bayes



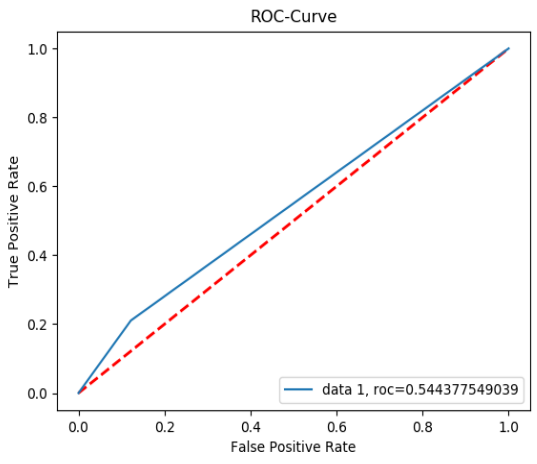
KNN, with K=6



SVM, kernel=RBF, C= 50, Gamma=0.3



Logistic regression

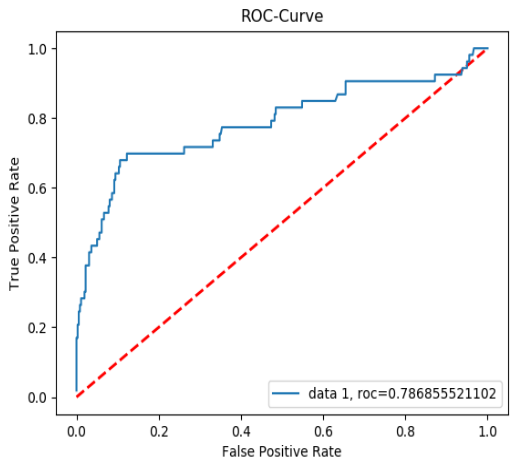


Decision Tree

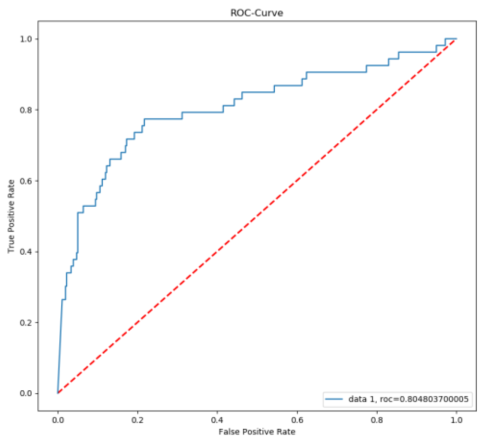
Figure 17: ROC-AUC curves for Different Classifiers for "Dataset without unknowns"

For Dataset with unknowns treated as separate category:

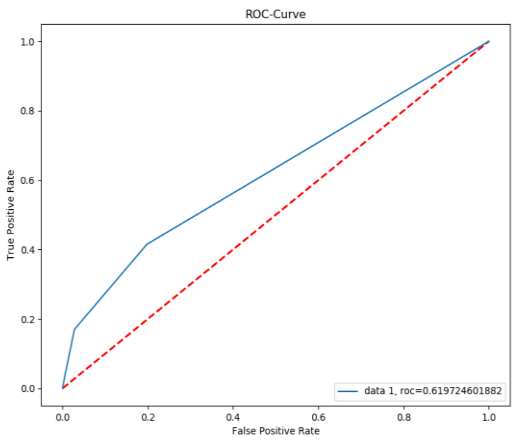
Classifier	Precision	Recall	F1 Score	ROC_AUC Score
Naïve Bayes	0.87	0.86	0.86	0.8048
SVM-RBF	0.90	0.90	0.87	0.8050
Logistic Regression	0.89	0.90	0.87	0.8226
KNN	0.83	0.87	0.84	0.6197
Decision Tree	0.84	0.84	0.84	0.634
Random Forest	0.88	0.90	0.87	0.7865



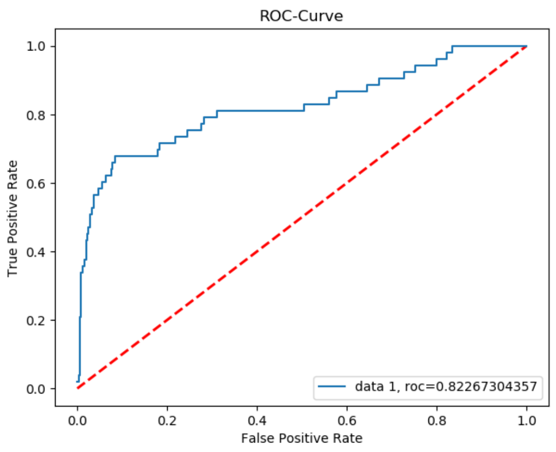
Random Forest



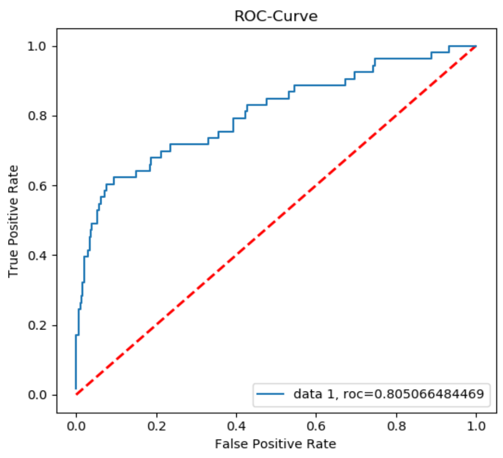
Naïve Bayes



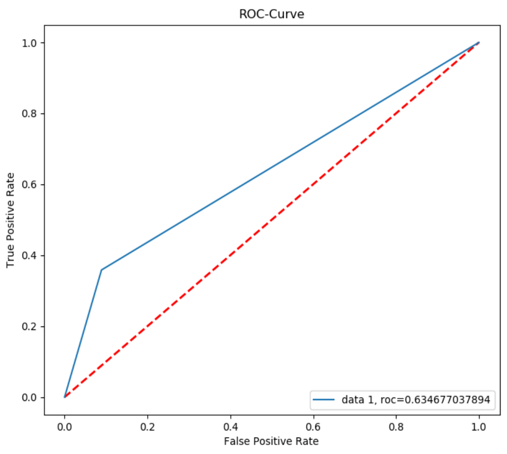
KNN with K=6



Logistic Regression



SVM, Kernel: RBF



Decision Tree

Figure 18: ROC-AUC curves for Different Classifiers for "Dataset with unknowns"

IV. Discussion

We have learnt quite a few things by observing data metrics while building the end-to-end machine learning model to predict customer subscription to deposit in the bank or not.

We now discuss our observations.

For Dataset with unknowns treated as separate category:

Ranking classifier for the dataset with unknowns from descending order according to ROC-AUC accuracy

Classifier	F1 score	ROC-AUC value
Logistic Regression	0.87	0.8226
SVM-RBF	0.87	0.8050
Naïve Bayes	0.86	0.8048
Random Forest	0.87	0.7865
Decision Tree	0.84	0.634
KNN	0.84	0.6197

For Dataset without unknowns:

Ranking classifier for the dataset without unknowns from descending order according to ROC-AUC accuracy

Classifier	F1 score	ROC-AUC value
SVM-RBF	0.86	0.7540
Logistic Regression	0.87	0.7419
Random Forest	0.86	0.7361
Naïve Bayes	0.82	0.7061
KNN	0.86	0.6905
Decision Tree	0.82	0.5443

From the above results and tables where the classifiers were ranked we see that:

1. For Dataset with unknowns treated as separate category, we see that logistic regression performed the best followed by SVM with RBF kernel and KNN performed the worst after which Decision tree performed the next worst possible based on ROC-AUC results.

2. For Dataset without unknowns, we see that SVM performed the best followed by logistic regression and Decision tree performed the worst and KNN did the next worst in terms of classification accuracy.

We see in both cases that logistic and SVM rank top two and KNN and Decision Tree rank at the bottom.

So therefore, our model is pretty close to being consistent.

The reason decision tree has such a low ROC-AUC accuracy is because it over-fits the training set. This was mentioned in the approach section for decision tree and it is also observed that the over-fitting issue is solved by Random forests in both the datasets. Hence, we have proved what we have stated in our description of the classification models. KNN is a lazy learner and does not handle categorical data and "soft" boundaries very well. Hence, this could be the reason for its poor performance.

We assumed our baseline model as the Naïve Bayes Classifier, reason being it was simple to implement and is generally at an average level at making decent accuracies.

We achieved better results than baseline model for logistic regression and SVM-RBF in both data sets and Random forest achieved better results than Naïve Bayes in the dataset without unknowns.

We can say that the data set with unknowns which had more features performed better than the dataset without unknowns where in we removed the default column as there were all "no" values present. Hence, dimensionality reduction gives us poor results. Therefore, we can conclude that reducing features using random forest or PCA would be detrimental to performance.

Our best ROC-AUC accuracy overall is 82.26% and this can be improved by using advanced classifiers like neural networks or the gradient boosting technique.

Data imbalance can be taken care by the library imbalance learn in python. Performing oversampling would naturally give better results than under-sampling in this case, after observing its behaviour with different model environments.

References:

All the figures were taken and material was referred to from links below:

1. **https://www.slideshare.net/markpeng/general-tips-for-participating-kaggle-competitions**

2. **http://blog.echen.me/2011/04/27/choosing-a-machine-learning-classifier/**

3. **http://www.saedsayad.com/naive_bayesian.htm**

4. **https://developers.google.com/machine-learning/crash-course/classification/roc-and-auc**

5. **https://dzone.com/articles/machinex-simplifying-logistic-regression**

6. **https://www.digitalocean.com/community/tutorials/an-introduction-to-machine-learning**

7. **https://machinelearningmastery.com/how-to-get-baseline-results-and-why-they-matter/**