Fall
2022

Predicting Outcome of Bank Telemarketing Campaign

GROUP 3

AHMED ALSAADI, ANKITA GOYAL, SIMRAN BHATIA, VENKATA SAI TARUN REDDY PONGULATY

HTTPS://GITHUB.COM/SIMBYY/IE7300-TERM-DEPOSIT-PREDICTION

Table of Contents

Abstract

Marketing and commercialization of any capital industry has become the need of the hour. Every industry aspires to perform good marketing strategies for their firm so that they can make maximum profit. This is the main reason for the rise in demand of data analytics and data science applications across domains. This project intends to explore the nation of Portugal and study their banking strategies. This is accomplished through the example of a multinational Portuguese bank which aims to target customers for term deposits and determine which of their clients will agree to a fixed term deposit when approached by their bank through a phone call. The dataset was obtained from UCI's data repository and contains some strong indicators that help in executing various classification models. Ultimately, several models were chosen to help quantify what clients will be more likely to deposit a fixed term payment as opposed to others. The performance of these models was then compared to a baseline.

Introduction

Direct-to-Customer marketing campaigns help institutes reach different segments of their existing and potential clientele to achieve a specific goal. Telemarketing is used to centralize such campaigns and helps execute them and analyze campaign results all in one place. Availing technology and data-driven decision-making tactics helps companies maximize profit and minimize collateral costs. Longer-term impacts include maximizing customer lifetime value and using metrics and data points available to recommend products or strategies that align a specific client well with business demand.

This project focuses on direct marketing campaigns conducted by a Portuguese banking institution. The marketing campaigns were executed via phone calls. In many scenarios, the same client was contacted multiple times to conclude whether the bank term deposit would be a “yes” or “no”. Hence, this is a classification problem. The goal is to predict whether the client called by the bank in question will subscribe to a term bank deposit or not. Since every call and every marketing campaign have a cost associated with them, in order to maximize profit the class of interest will be “yes”.

There are several classification models that can be employed to solve this problem. For this project, Logistic Regression, Naïve Bayes Classifier, Neural Networks and Support Vector Machine models were chosen.

Data Description

Data resource: https://archive.ics.uci.edu/ml/datasets/bank+marketing

The dataset has 45211 instances and 17 attributes. The meaning of each attribute present in the dataset and its datatype is described below:

	age	job	marital	education	default	balance	housing	loan	contact	day	month	duration	campaign	pdays	previous	poutcome	y
0	58	management	married	tertiary	no	2143	yes	no	unknown	5	may	261	1	-1	0	unknown	no
1	44	technician	single	secondary	no	29	yes	no	unknown	5	may	151	1	-1	0	unknown	no
2	33	entrepreneur	married	secondary	no	2	yes	yes	unknown	5	may	76	1	-1	0	unknown	no
3	47	blue-collar	married	unknown	no	1506	yes	no	unknown	5	may	92	1	-1	0	unknown	no
4	33	unknown	single	unknown	no	1	no	no	unknown	5	may	198	1	-1	0	unknown	no

Figure 1 Preview of raw dataset

- Age-> Numeric; Continuous
- Job-> Categorical-> admin, blue collar, entrepreneur, housemaid, management, retired, self-employed, services, student, technician, unemployed, unknown
- Marital Status->Categorical-> divorced, married, single, unknown
- Education->Categorical-> high school, illiterate, professional course, university degree, unknown, basic 4yr, basic 6yr, basic 9yr
- Default->Categorical-> Do the customers have credit in default? (no, yes, unknown)
- Housing->Categorical, Do the customers have housing loan? (yes, no)
- Loan-> Categorical, Do the customers have personal loan? (yes, no)
- Balance->Numerical; Continuous
- Contact->Categorical, type of contact number (unknown, cellular, telephone)
- Month->Categorical, last contacted month of year
- pOutcome->Categorical, outcome of previous contact (unknown, failure, other, success)
- Duration->Numerical, Continuous
- Campaign->Numerical, Continuous
- pDays->Numerical, Continuous
- Target->Numerical, Continuous
- Previous->Numerical, Continuous

The target variable of the dataset is in the form of yes/no and tells us whether the client has subscribed in the term deposit or not.

#	Column	Non-Null Count	Dtype
0	age	45211 non-null	int64
1	job	45211 non-null	object
2	marital	45211 non-null	object
3	education	45211 non-null	object
4	default	45211 non-null	object
5	balance	45211 non-null	int64
6	housing	45211 non-null	object
7	loan	45211 non-null	object
8	contact	45211 non-null	object
9	day	45211 non-null	int64
10	month	45211 non-null	object
11	duration	45211 non-null	int64
12	campaign	45211 non-null	int64
13	pdays	45211 non-null	int64
14	previous	45211 non-null	int64
15	poutcome	45211 non-null	object
16	y	45211 non-null	object

Figure 2 description of features in dataset

Exploratory Data Analysis and Visualization

Exploratory Data Analysis was performed to get familiar with the data and understand how what variable adds value to the result. It helps in highlighting noise and outliers in the data and aids in understanding relations between different variables.

Column-wise Data Exploration

Boolean Columns

These columns contain only “yes” and “no” values, which can also be interpreted as 0s and 1s. Distribution of either value can be observed in the bar charts below.

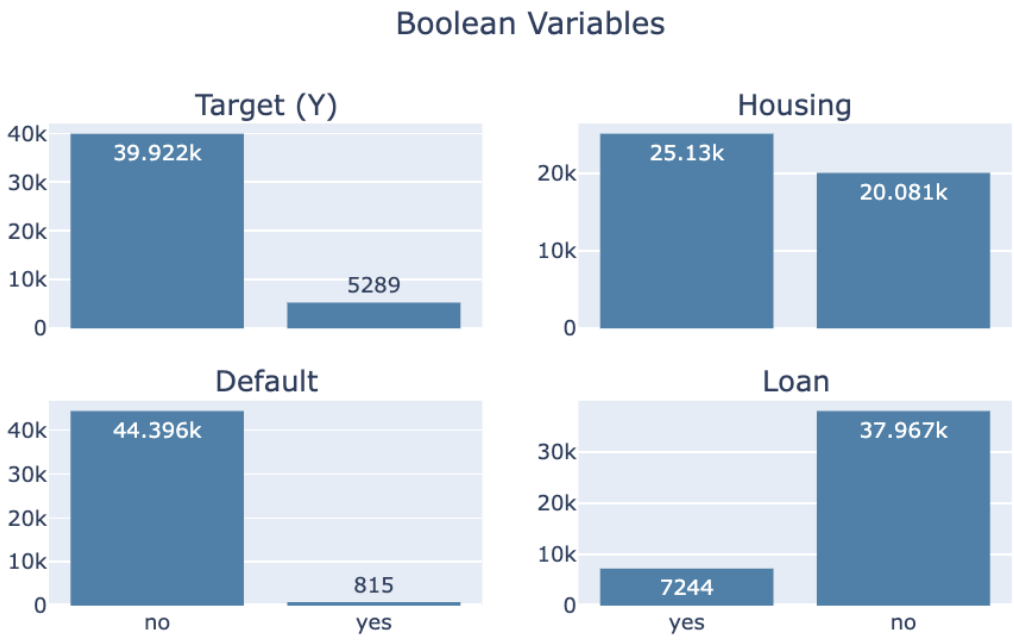


Figure 3 range of boolean columns

Categorical Columns

These columns contain distinct and known values from a finite set of values.

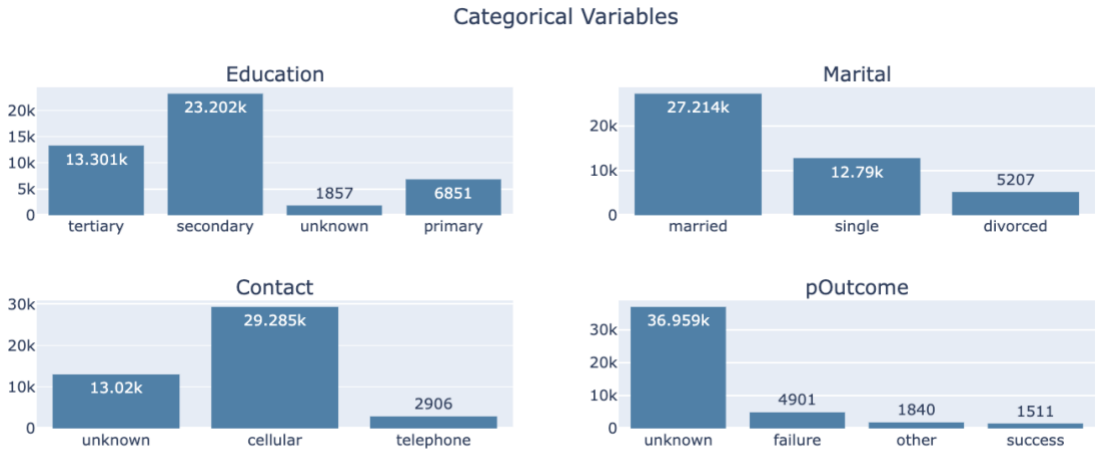


Figure 4 range of categorical columns

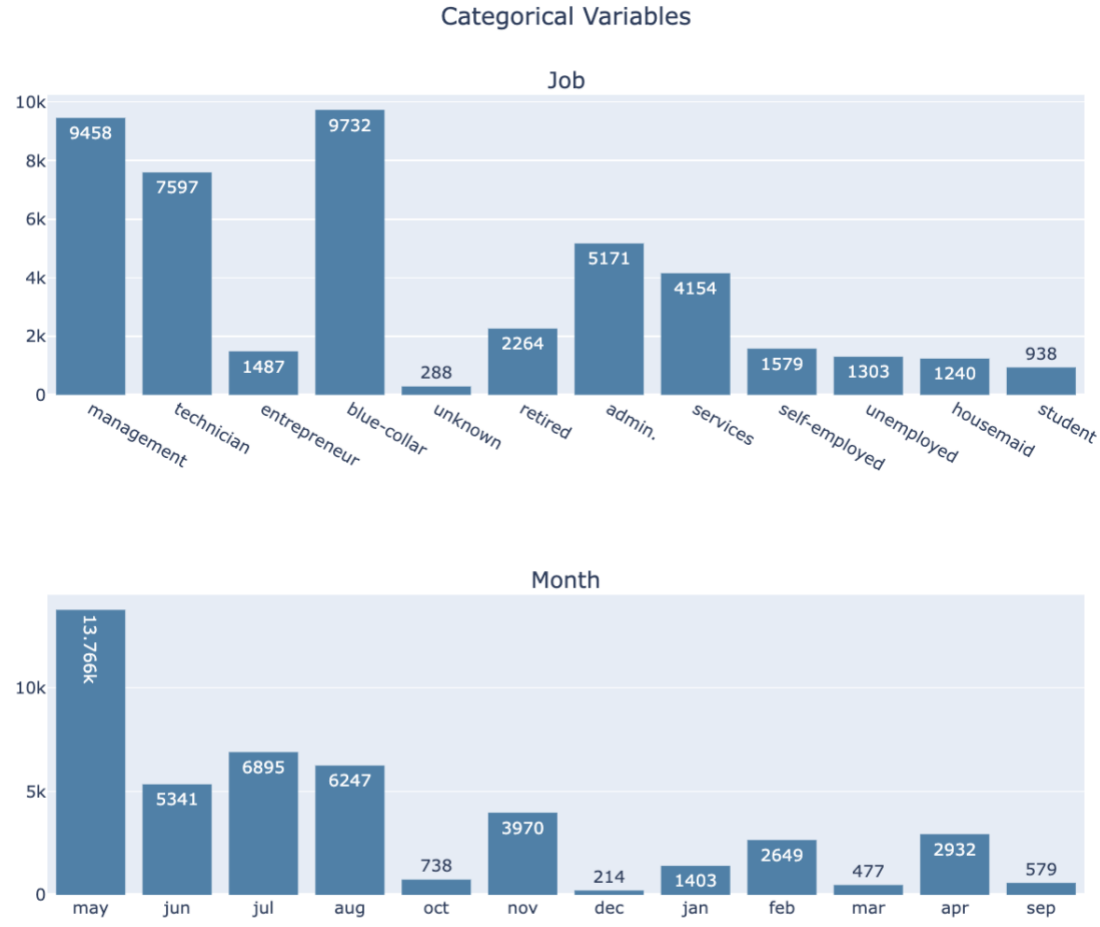


Figure 5 range of categorical columns (continued)

Numerical Columns

Value distribution of numerical columns with continuous data can be observed below.

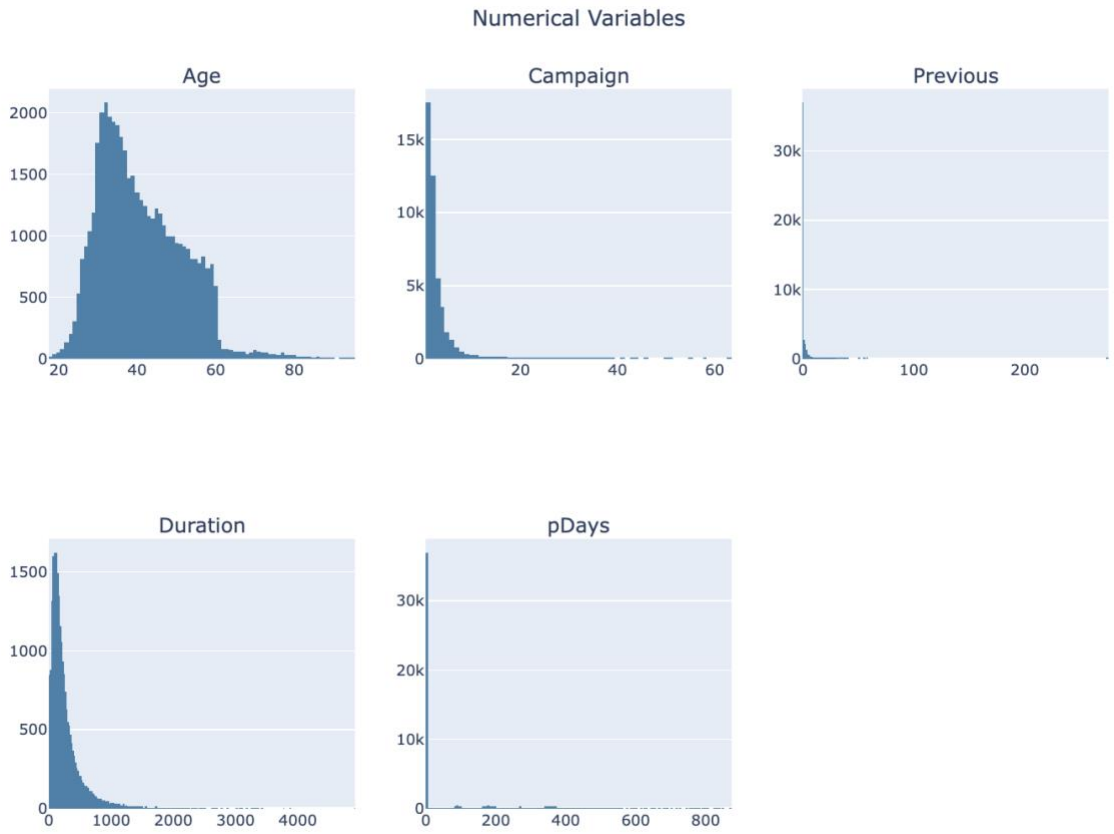


Figure 6 range of numerical columns

Correlation Across Variables - Heatmap

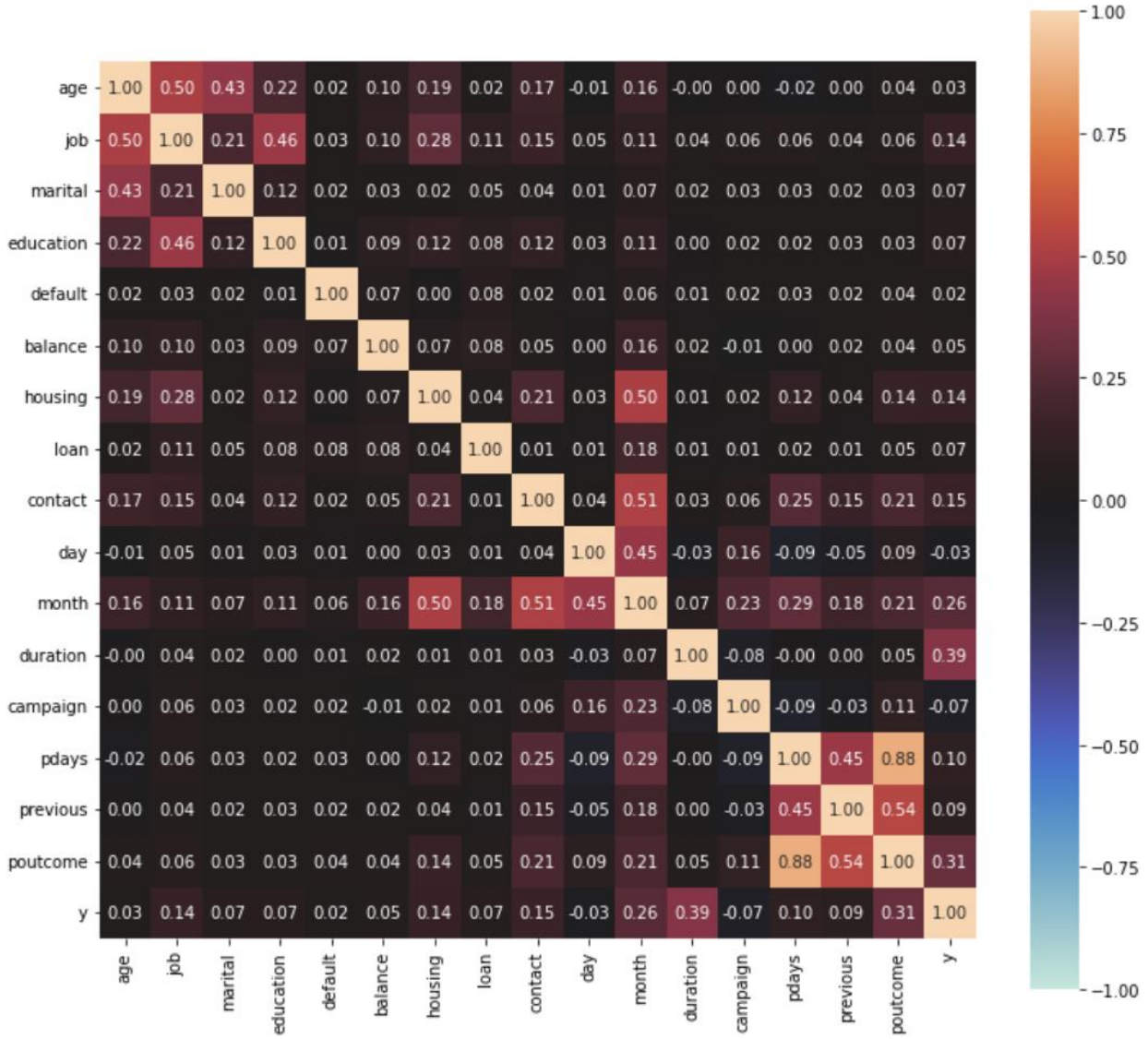


Figure 7 heatmap depicting correlation of features

Figure 8 heatmap depicting correlation of variables

Takeaway – No feature is highly correlated. As a result, no columns were dropped from the dataset in this step.

Outlier Visualization - Boxplots

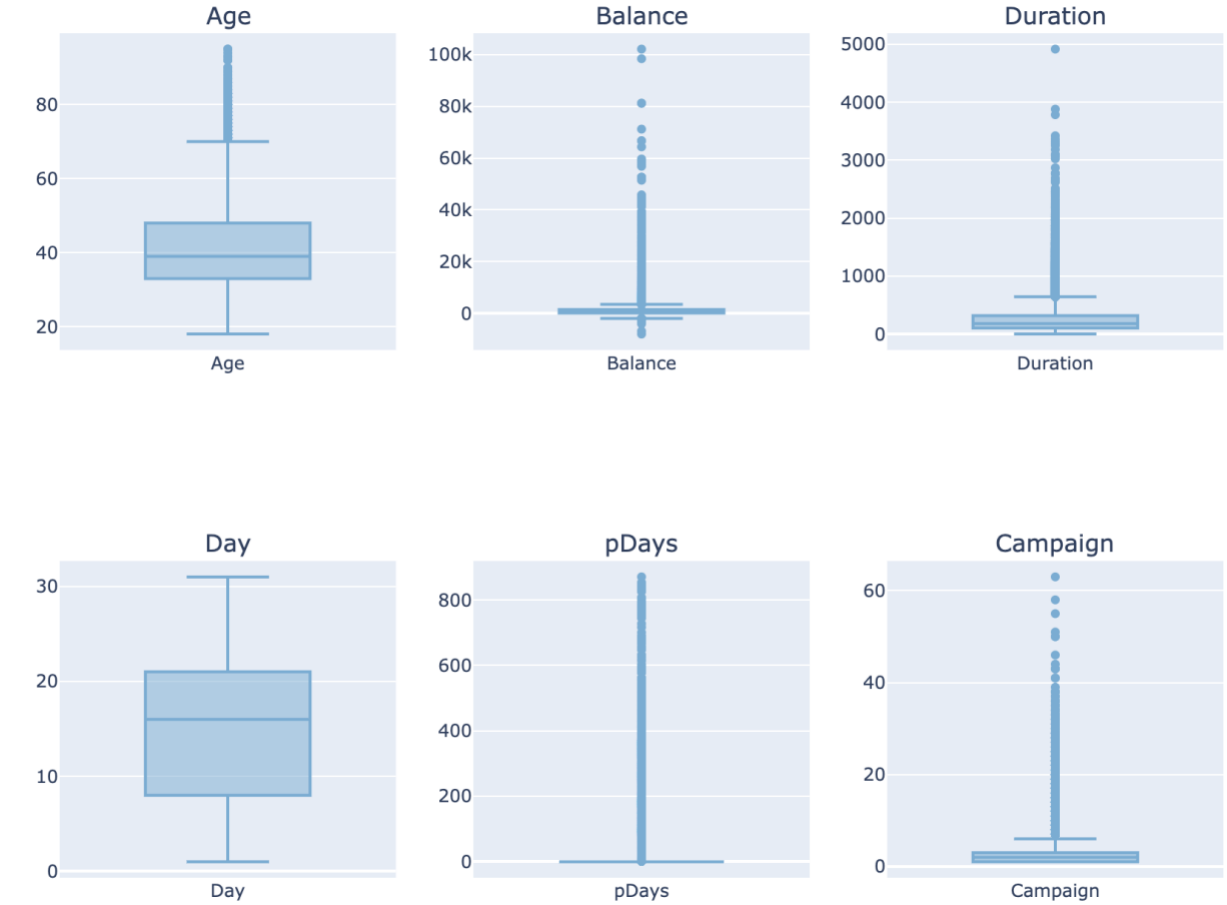


Figure 9 boxplot of numerical features

Takeaway- As can be seen, many of the numeric features are right skewed and this resulted also in having many outliers. Hence, exponential function is used to reduce the skewedness.

Pair Plots

Figure 10 pair plot of numerical variables

Takeaway- The data doesn't appear to be linearly separable.

Feature Engineering

Exponential Function

After applying exponential function to reduce the skewedness in this dataset, transformation can be observed through following visualizations:

Box Plot

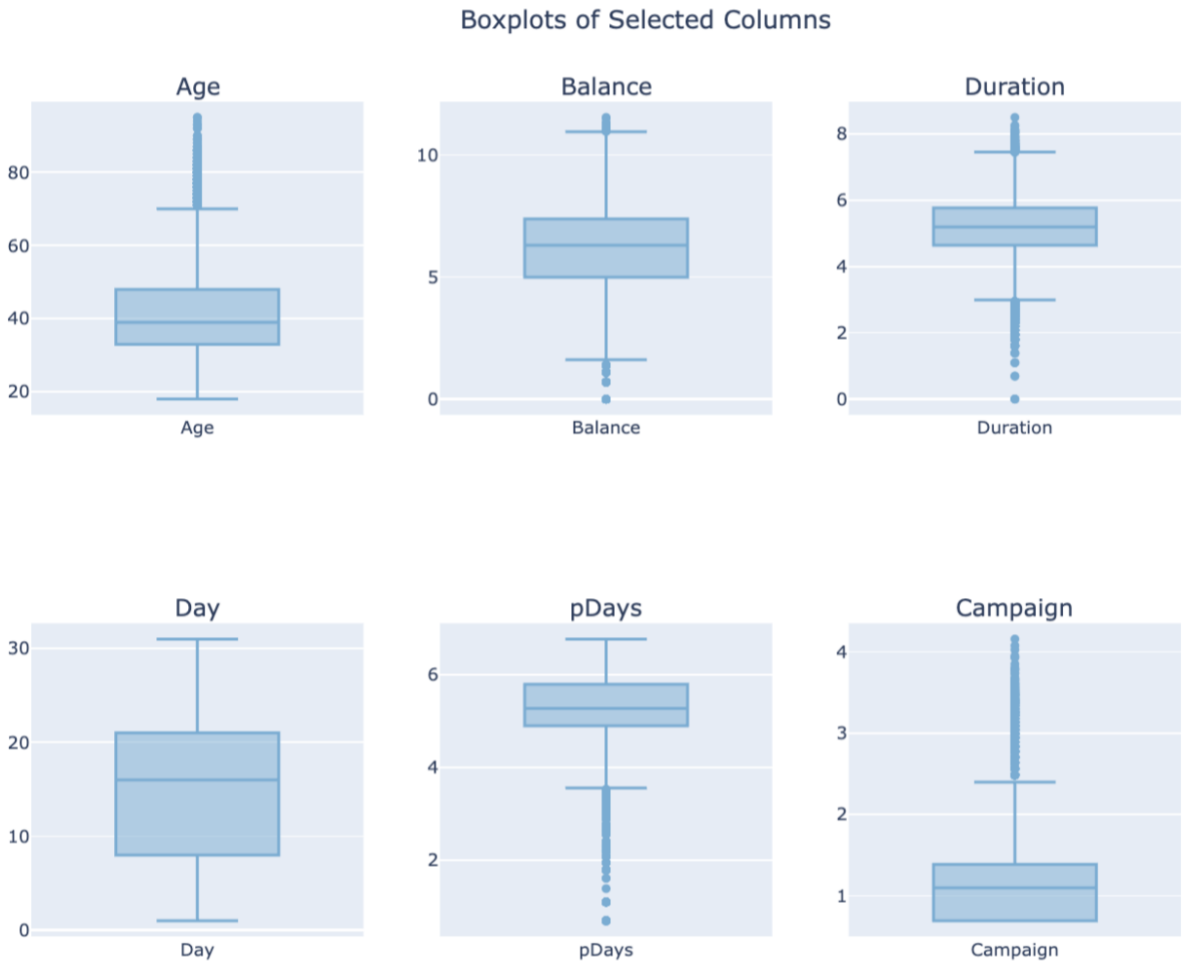


Figure 11 updated box plot depicting corrected ranges of numerical columns

Pair Plots

Figure 12 updated pair plot showing distribution of variables

Heat Map

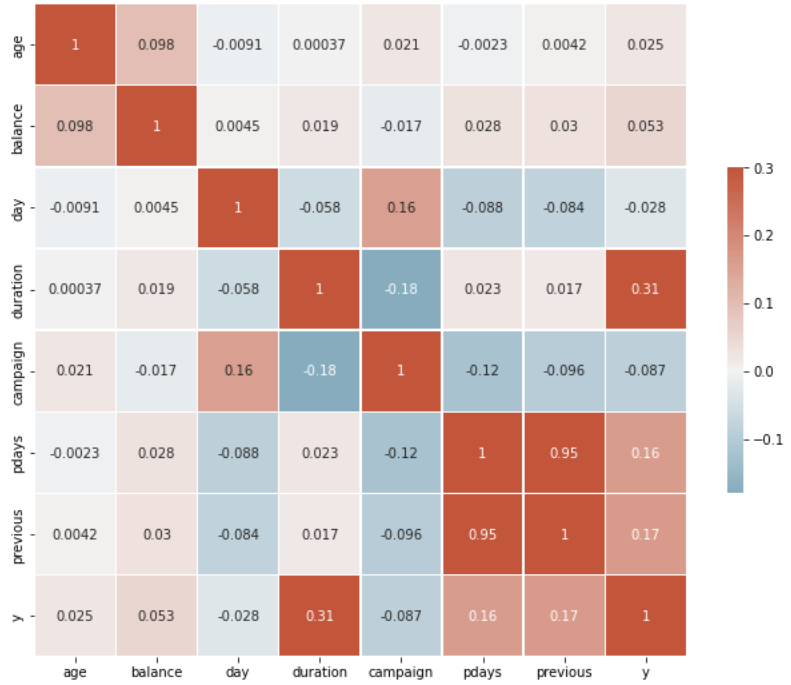


Figure 13 updated heatmap showing correlation of columns

Feature Selection for Dimensional Reduction

Using ExtraTreesClassifier(), following variable importance scores were observed:

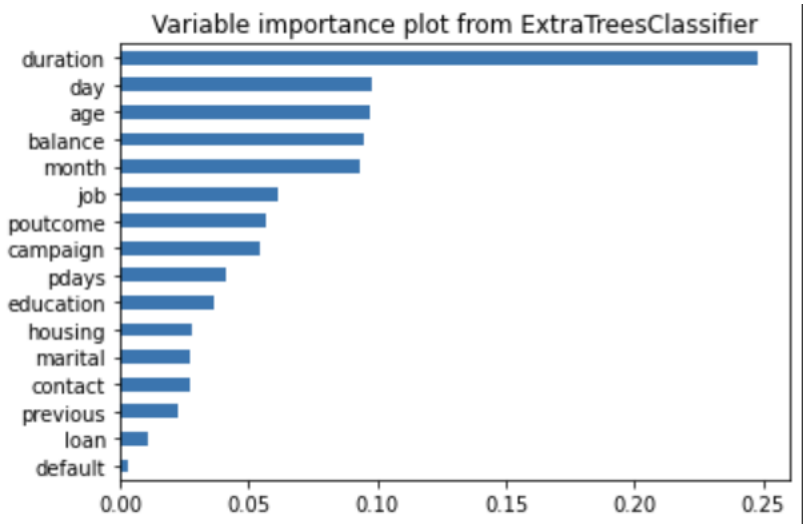


Figure 14 variable importance plot through ExtraTreesClassifier

Using RandomForestClassifier(), following variable importance scores were observed:

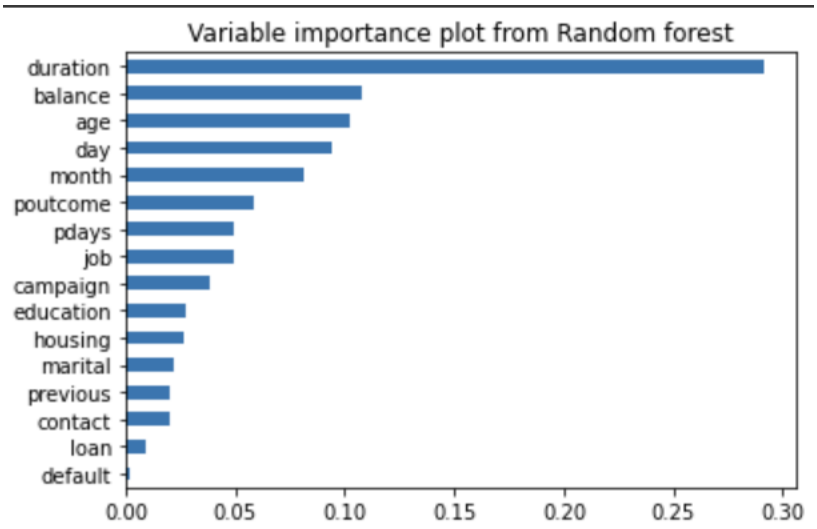


Figure 15 variable importance scores through RandomForest

	Specs	Score
12	pdays	4724.360211
14	poutcome	2661.225090
13	previous	1437.698352
10	duration	872.764749
1	job	392.140782
5	housing	388.949715
7	contact	300.161142
6	loan	176.516137
8	day	159.200398
3	education	90.617723

Figure 16 variable importance scores for feature selection

From the above analysis, columns 'marital', 'default', 'loan', 'contact', 'day', 'pdays', 'previous' columns were deemed less useful and therefore removed from the dataset.

Standardization and Dummy Columns

All numerical columns were standardized because they are varying in ranges. The following formula of standardization was employed:

Equation for standardization:

$$z = \frac{x - \mu}{\sigma}$$

Here, mean $\mu = \frac{1}{N} . \Sigma_{i=1}^{n} . (x_i)$ and standard deviation $\sigma = \sqrt{\frac{1}{N} . \sum_{i=1}^{N} (x_i - \mu)^2}$

Dummy columns were added for all categorical columns to one-hot encode their values.

The updated dataset has 34 columns:

```
Index(['age', 'balance', 'duration', 'campaign', 'housing', 'job_blue-collar',
       'job_entrepreneur', 'job_housemaid', 'job_management', 'job_retired',
       'job_self-employed', 'job_services', 'job_student', 'job_technician',
       'job_unemployed', 'job_unknown', 'education_secondary',
       'education_tertiary', 'education_unknown', 'month_aug', 'month_dec',
       'month_feb', 'month_jan', 'month_jul', 'month_jun', 'month_mar',
       'month_may', 'month_nov', 'month_oct', 'month_sep', 'poutcome_other',
       'poutcome_success', 'poutcome_unknown', 'y'],
      dtype='object')
```

Figure 17 updated columns after removing unimportant columns

Over-sampling and Under-sampling

Depending on the model implemented, over-sampled and under-sampled datasets were used to remedy obstacles imposed by an unbalanced dataset.

The division of classes before under/over-sampling:

```
df_temp = df["y"].unique()
df['y'].value_counts()
```

```
0    39922
1     5289
Name: y, dtype: int64
```

Figure 18 '1' is the class of interest, ie 'yes' and '0' is the other class, 'no'

After implementing under-sampling, the dataset is balanced among both classes.

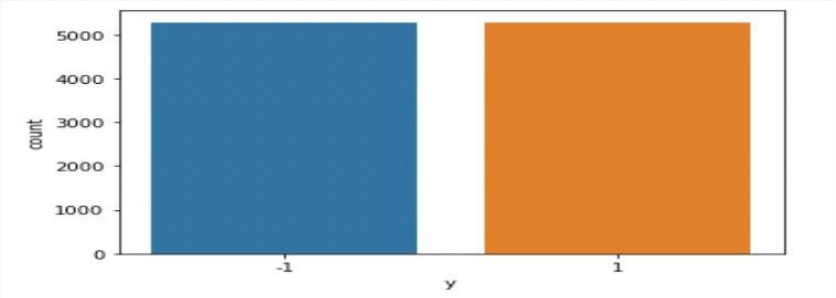


Figure 19 '1' is the class of interest, ie 'yes' and '-1' is the other class, ie 'no'

Model Implementation

Logistic Regression

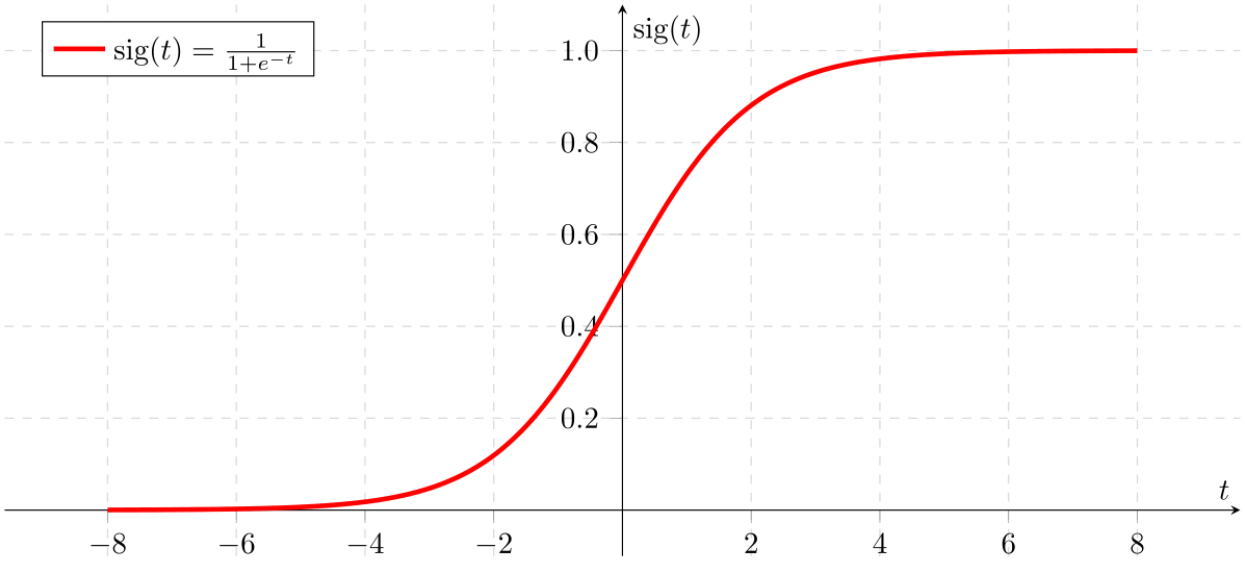


Figure 20 depiction of sigmoid function

Logistic regression is a statistical model used for classification. The model calculates the probability of an event on the basis of provided dataset. The provided data is assumed to contain only independent variables. The outcome is probability of test data belonging to either class and is therefore bounded between 0 and 1. Natural logarithm of odds is applied on the data.

$$logit(\pi) = \frac{1}{(1 + e^{-\pi})}$$

$$\ln\left(\frac{\pi}{1-\pi}\right) = \beta_0 + \beta_1.x_1 \ldots + \beta_k.x_k$$

In this equation, $logit(\pi)$ is the response variable and x is the independent variable. β parameter is the coefficient and will be obtained using Maximum Likelihood Estimation (MLE). Once the optimal coefficient has been determined through MLE, conditional probability of each observation is calculated to yield the predicted probability.

For this dataset, there are two possible values for the outcome variable, hence binary classification model is employed. This means a probability below 0.5 will correspond to class 0 and above 0.5 will point towards class 1.

Performance

Performance of Logistic Regression model was evaluated by running the model using k-means validation (with k=5), and various learning rates of 0.1, 0.0001 and 0.00001. The performance

on each k^{th} version of the dataset was then averaged to evaluate accuracy, precision, recall and f1 score on test and training data, both. Best performance was achieved with **learning rate = 0.0001.**

Learning Rate = 0.1

Data	Accuracy	Precision	Recall	f1_score	Misclassified
train	0.7016	0.4882	0.8984	0.6326	6664.8
test	0.6346	0.2140	0.9078	0.3136	3305.4

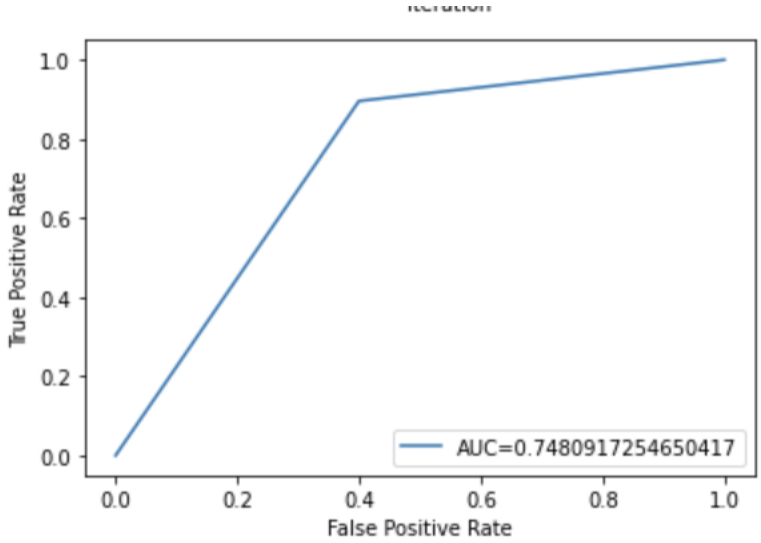


Figure 21 training data, k=1, learning rate=0.1

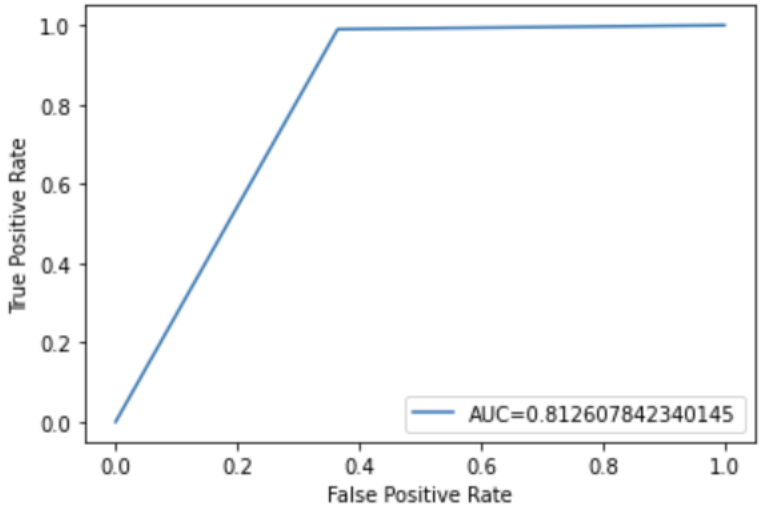


Figure 22 testing data, k=1, learning rate=0.1

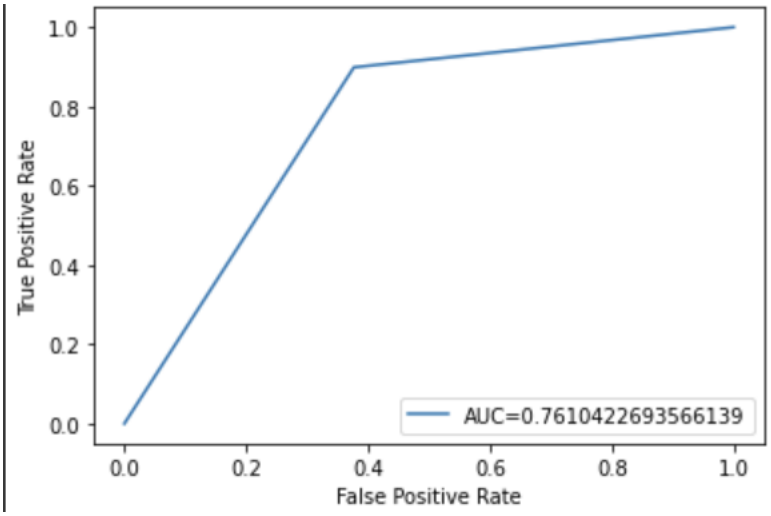


Figure 23 training data, learning rate = 0.1, k=2

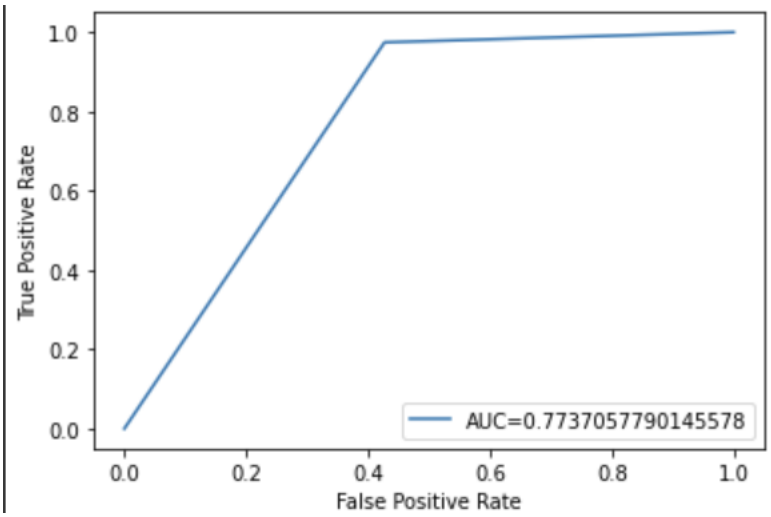


Figure 24 testing data, learning rate = 0.1, k=2

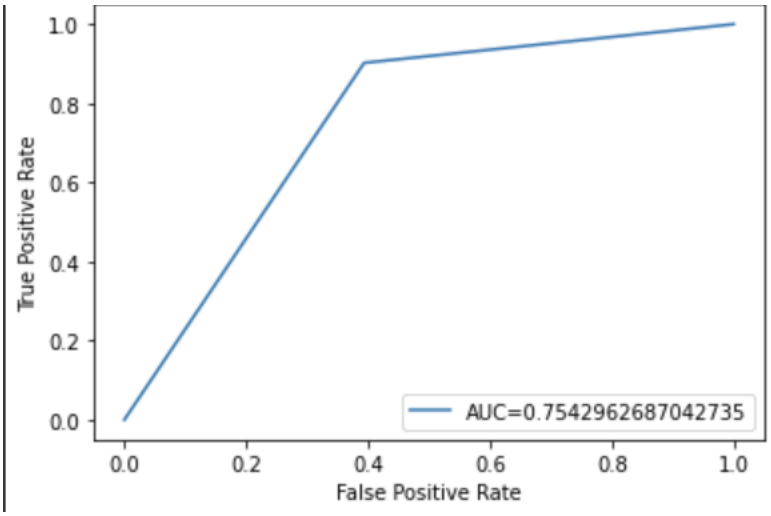


Figure 25 training data, learning rate=0.1, k=3

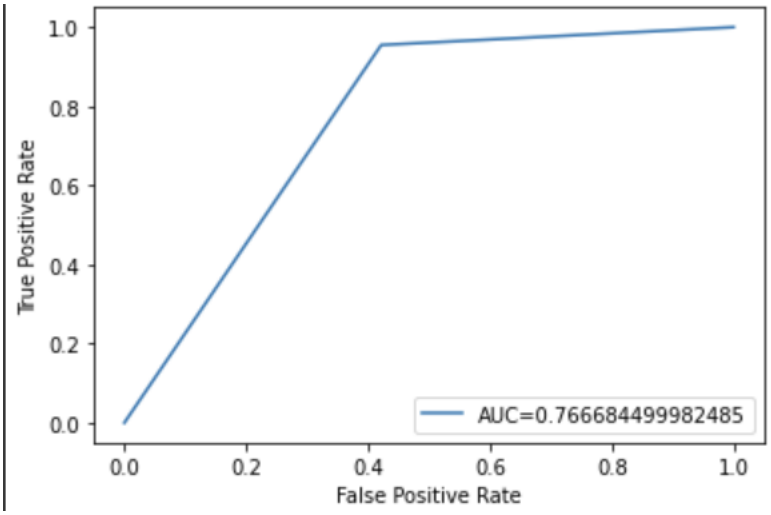


Figure 26 testing data, learning rate=0.1, k=3

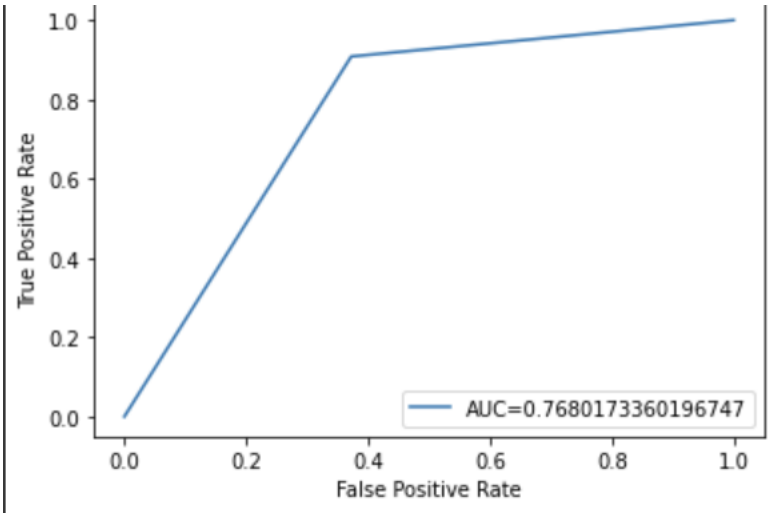


Figure 27 training data, learning rate=0.1, k=4

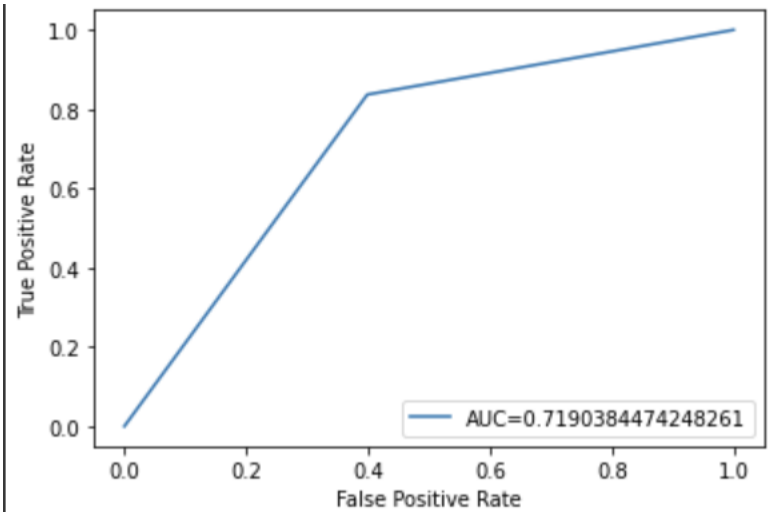


Figure 28 testing data, learning rate=0.1, k=4

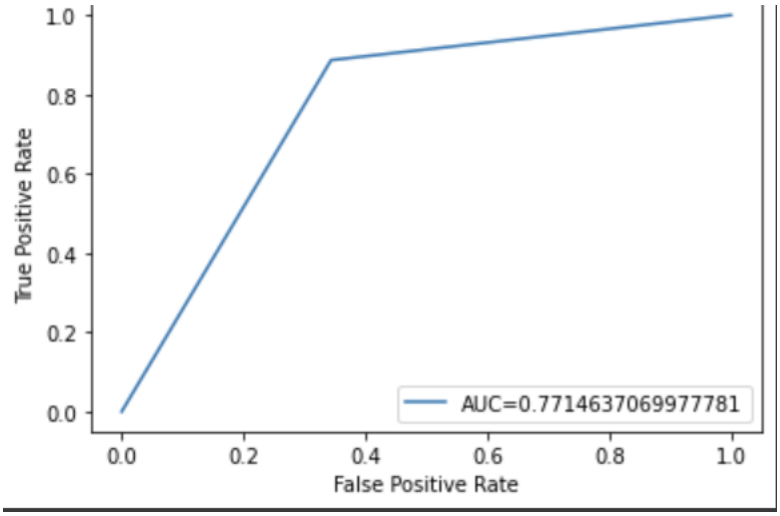


Figure 29 training data, learning rate=0.1, k=5

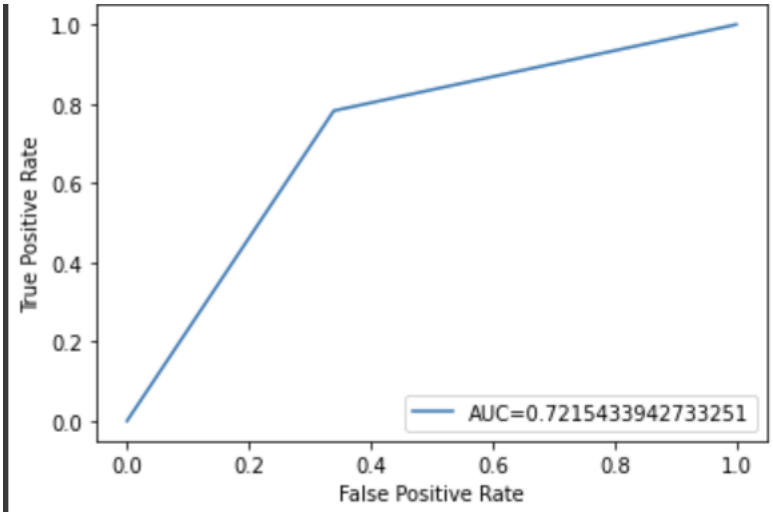
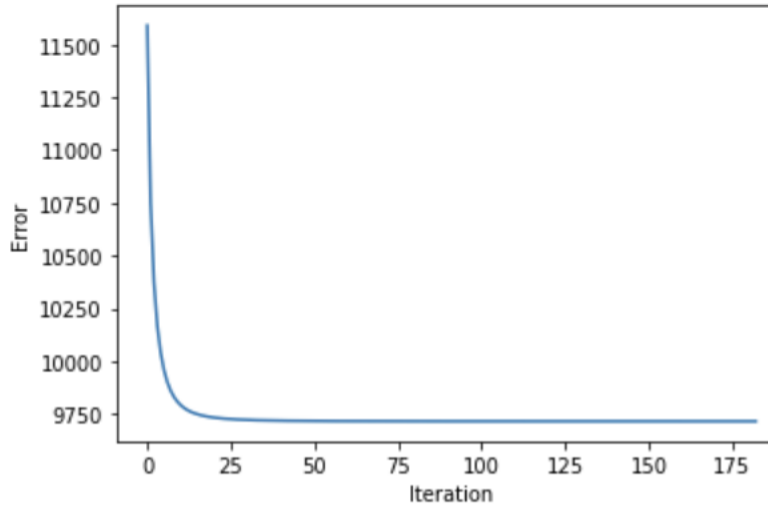
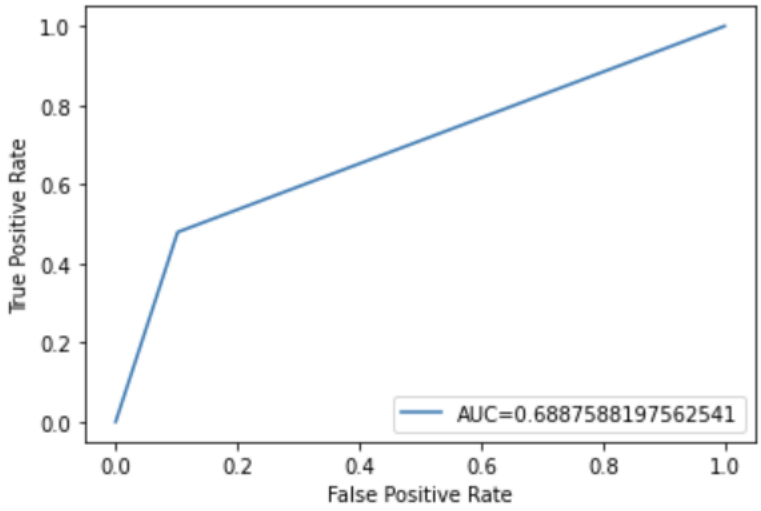


Figure 30 testing data, learning rate=0.1, k=5

Learning Rate = 0.0001

Data	Accuracy	Precision	Recall	f1_score	Misclassified
train	0.7984	0.6842	0.5416	0.6040	4495.0
test	0.8480	0.3718	0.6128	0.4166	1371.0





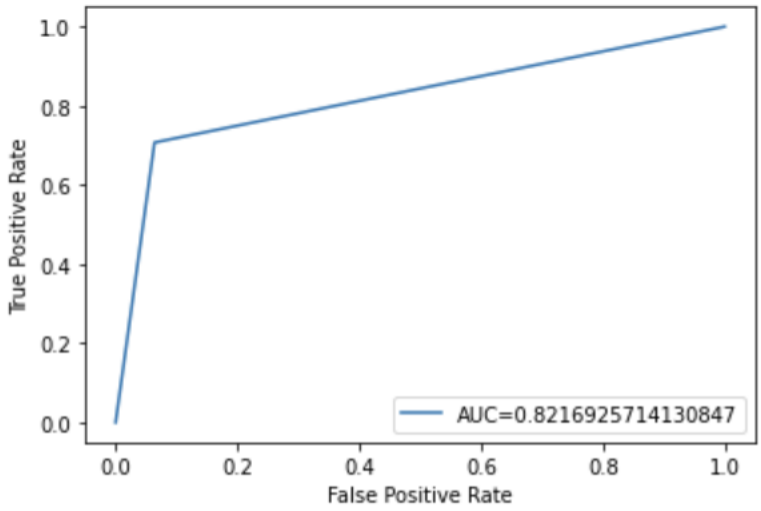
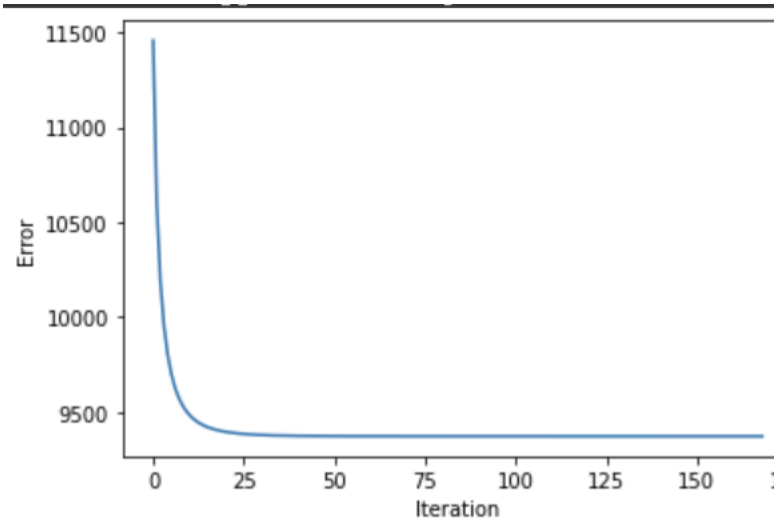
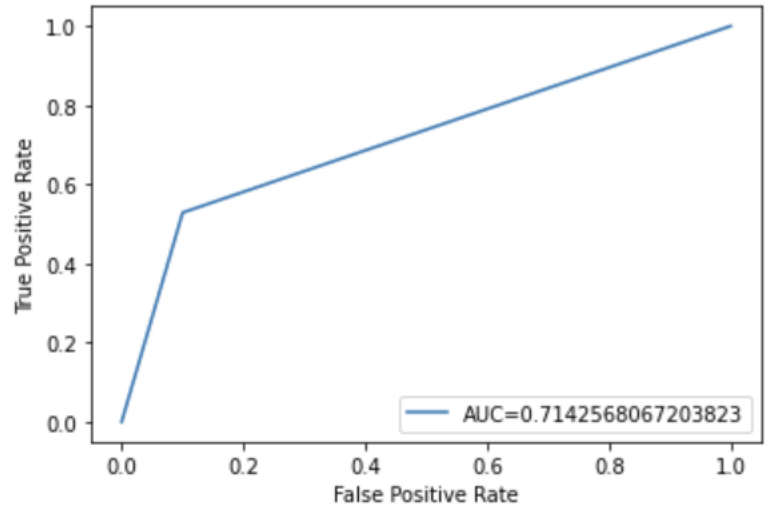


Figure 31 learning rate=0.0001 a. training, k=1, b. testing, k=1, c. training, k=2





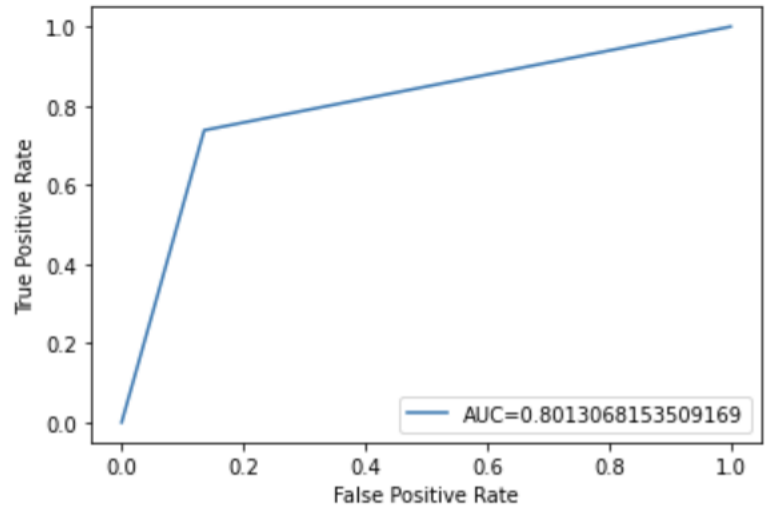
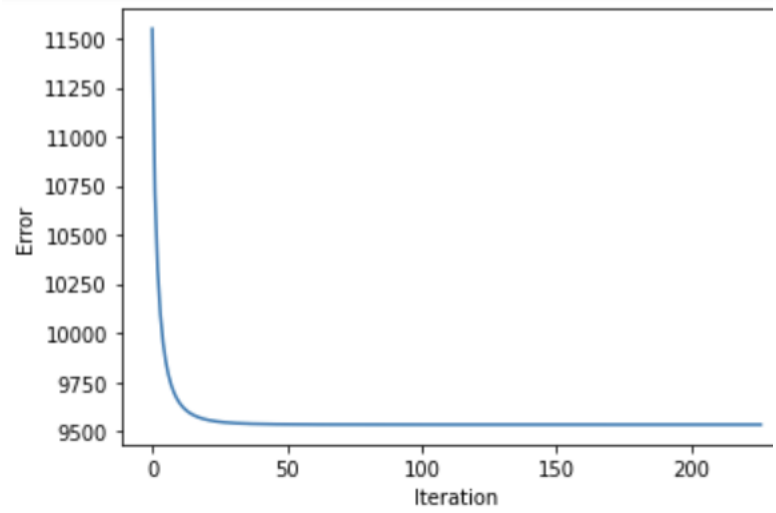
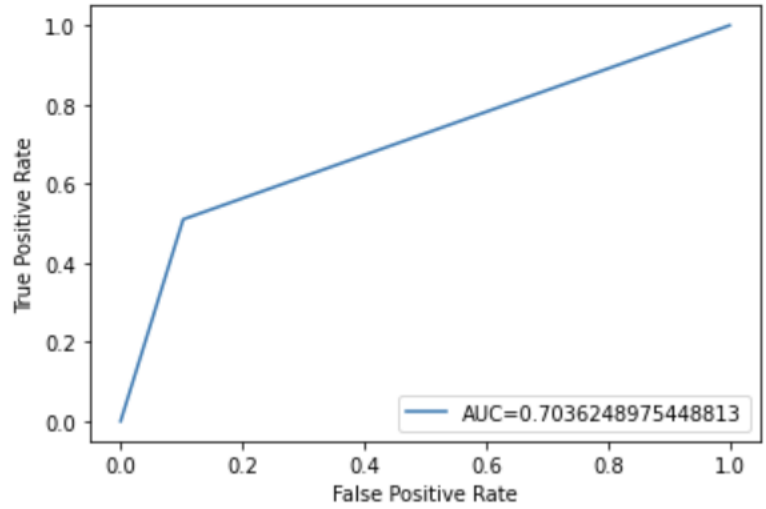


Figure 32 learning rate=0.0001, a. testing, k=2, b. training, k=3, c. testing, k=3





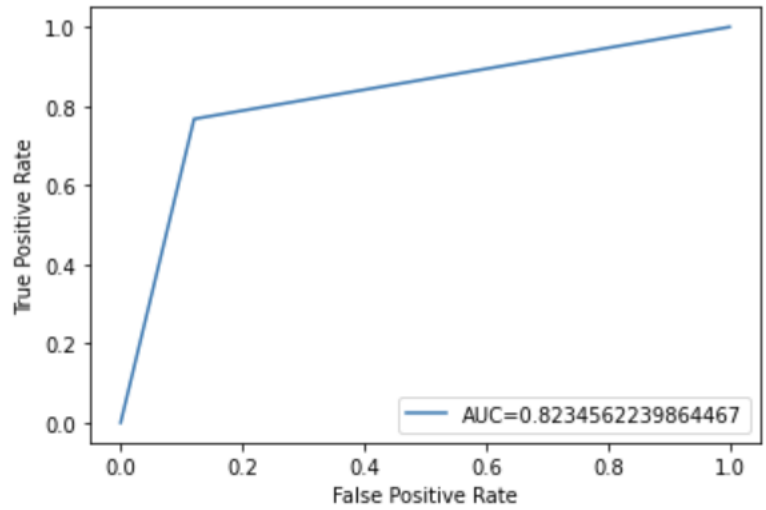
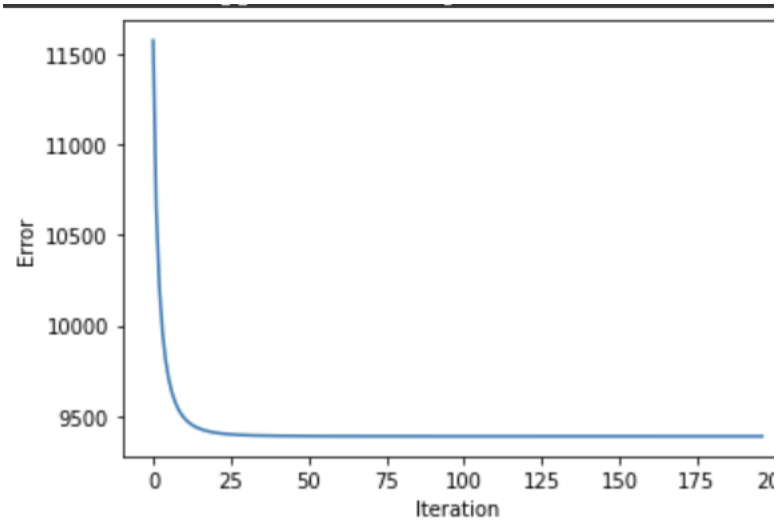
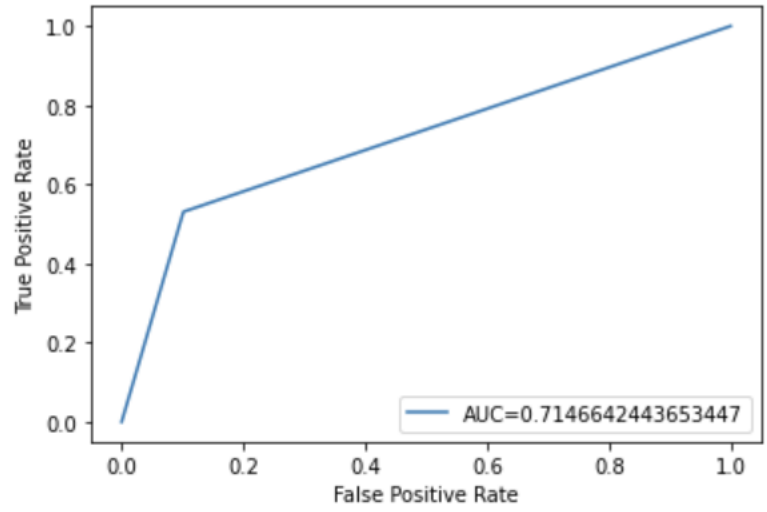


Figure 33 learning rate=0.0001, a. training, k=4, b. testing, k=4, c. training, k=5





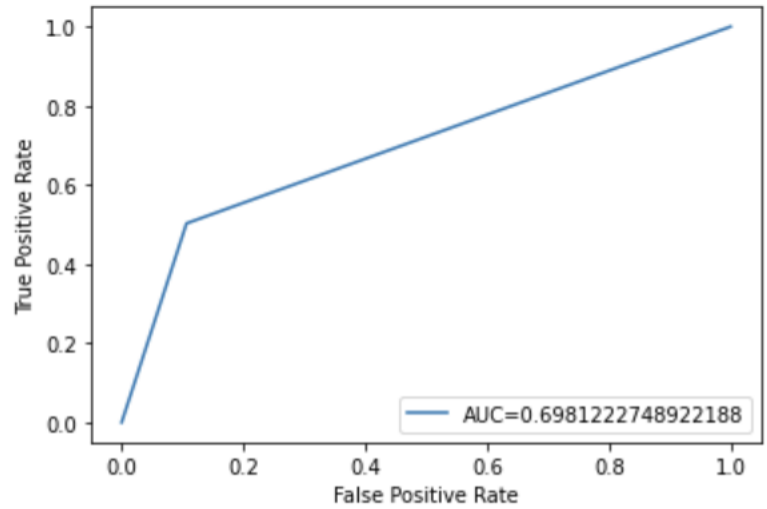
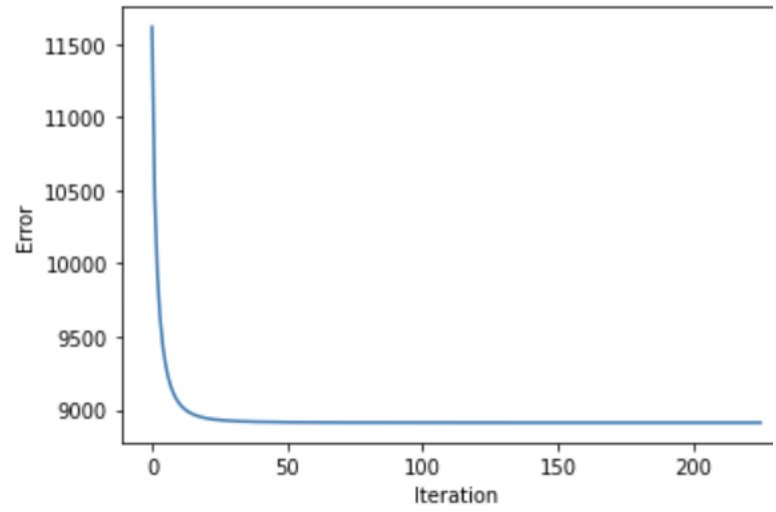
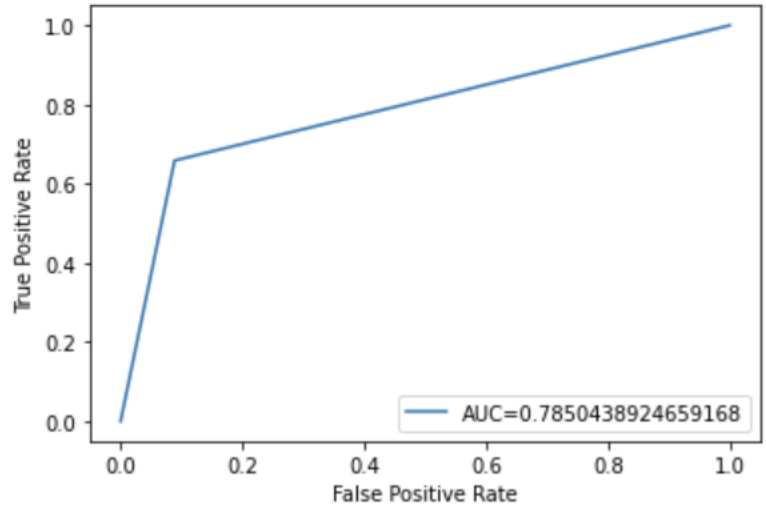


Figure 34 learning rate=0.0001, a. testing, k=5, b. training, k=6, c. testing, k=6





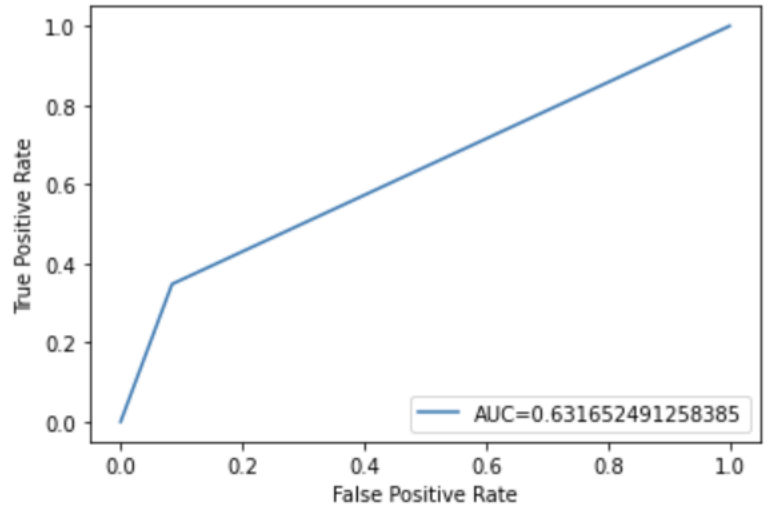
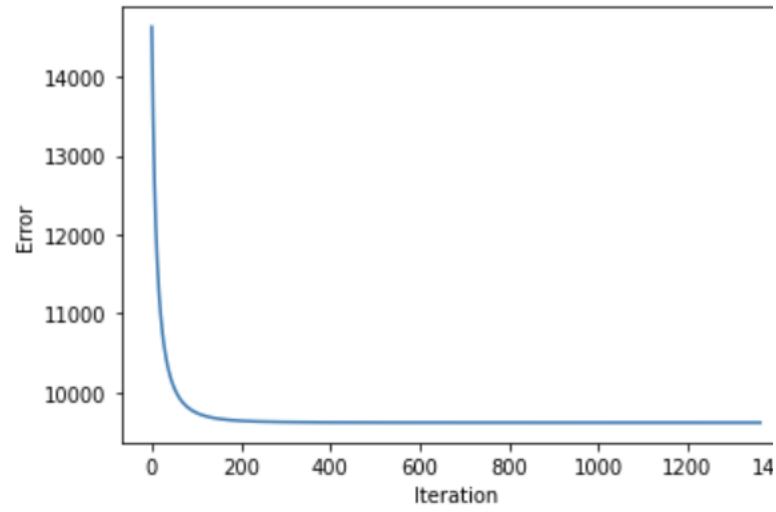
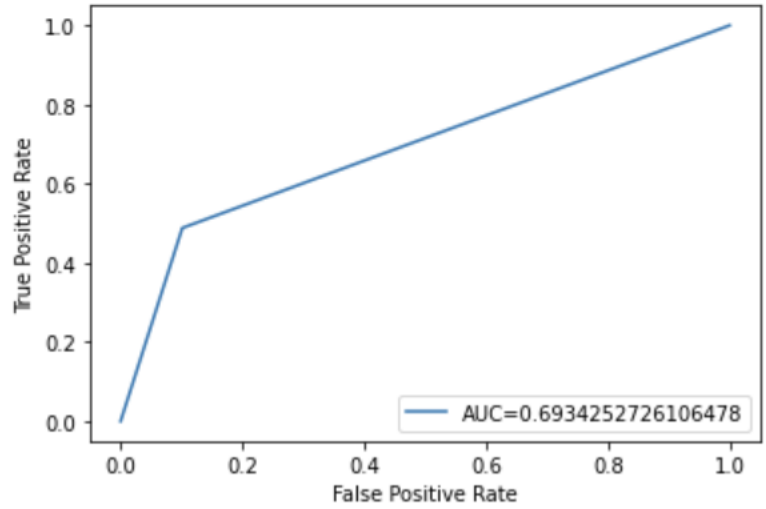


Figure 35 learning rate=0.0001, a. training, k=7, b. testing, k=7, c. training, k=8

Learning Rate = 0.00001

Data	Accuracy	Precision	Recall	f1_score	Misclassified
train	0.7988	0.6850	0.5436	0.6058	4480.0
test	0.8480	0.3692	0.6086	0.4142	1376.0





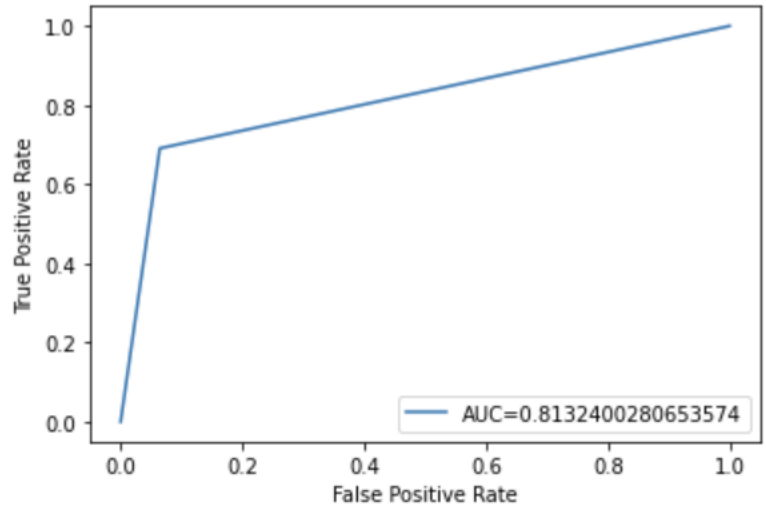
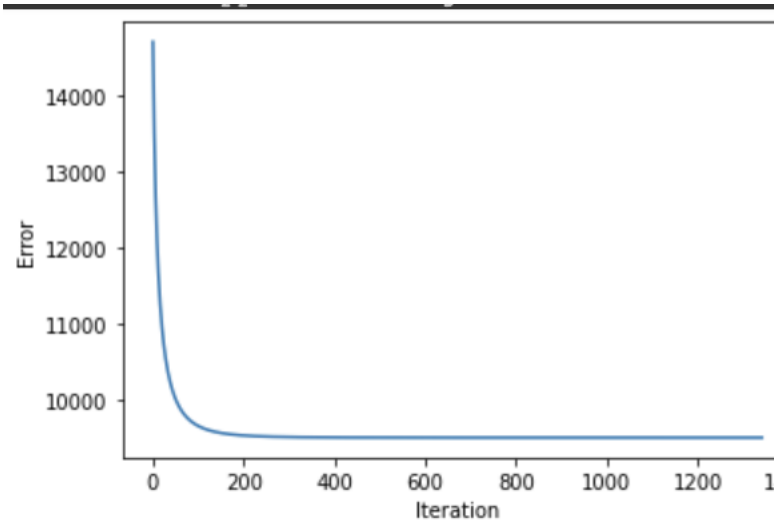
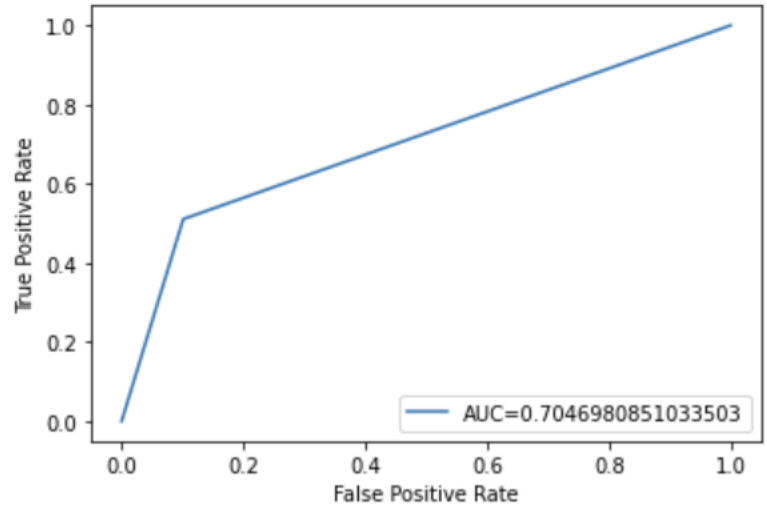


Figure 36 learning rate=0.00001, k=1





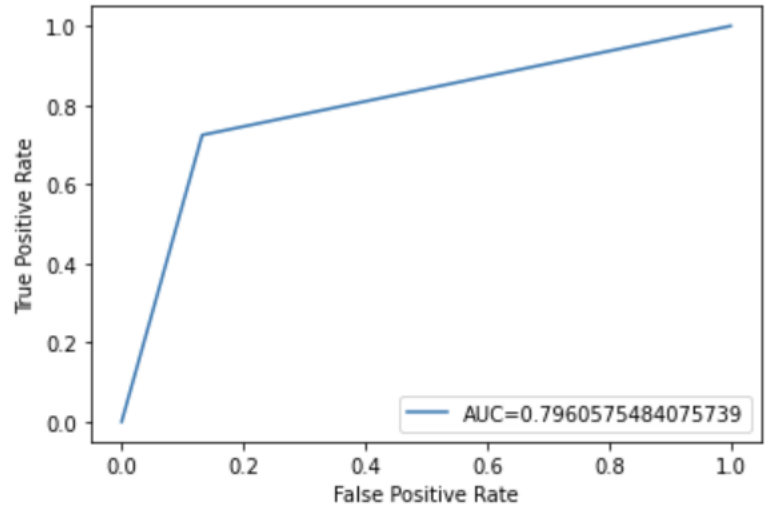
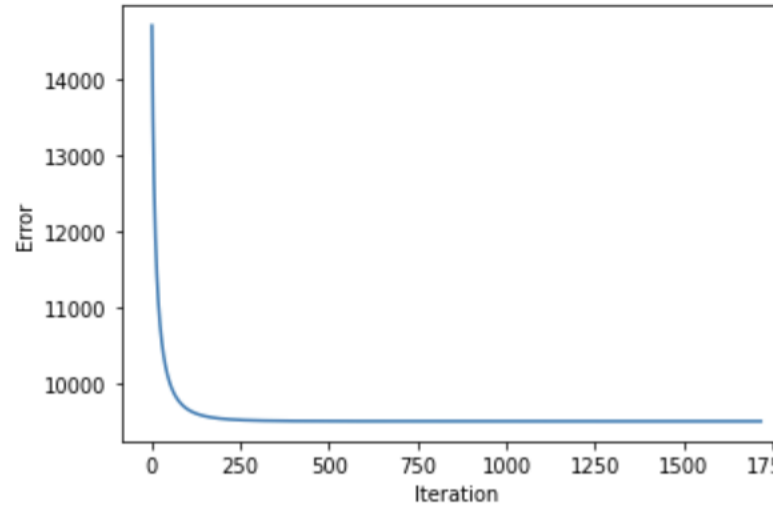
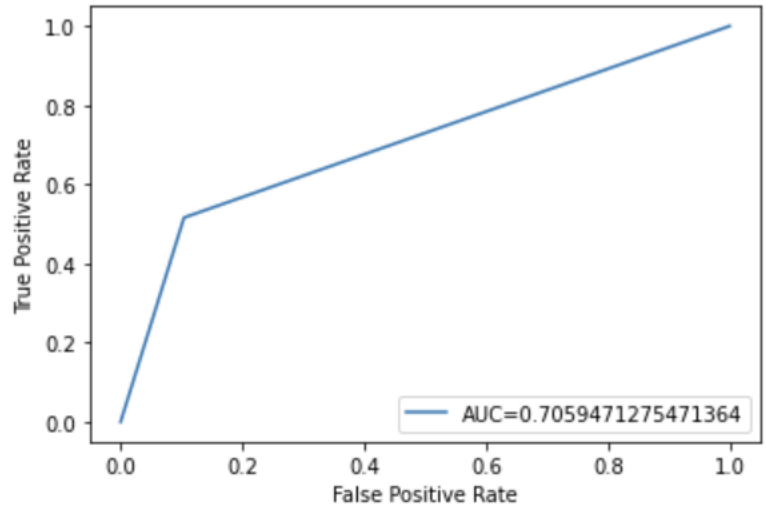


Figure 37 learning rate=0.00001, k=2





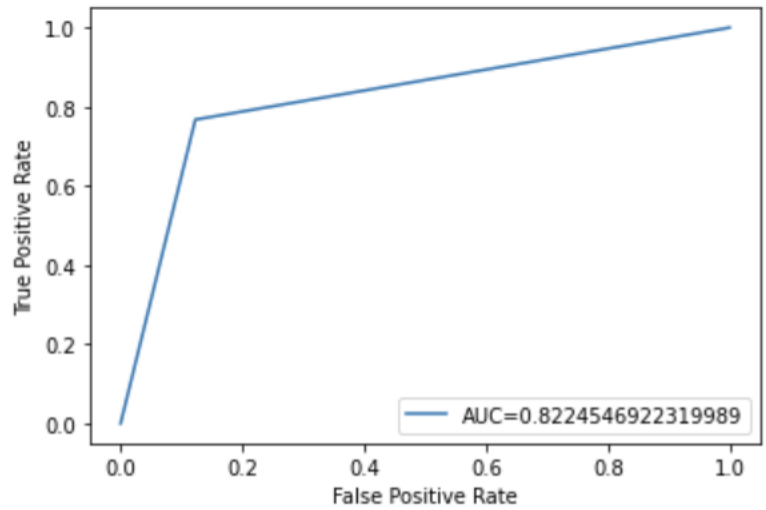
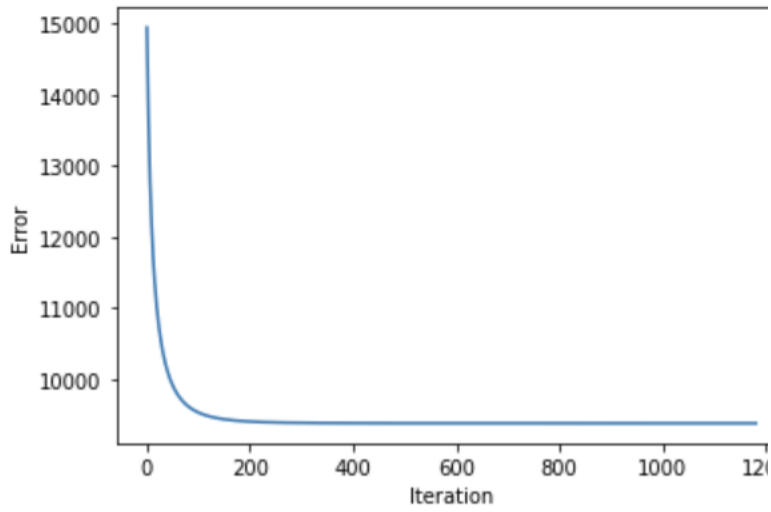
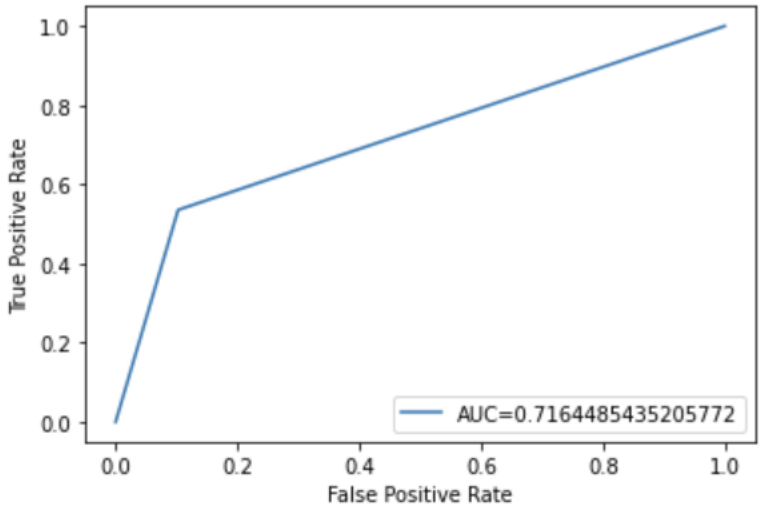


Figure 38 learning rate=0.00001, k=3





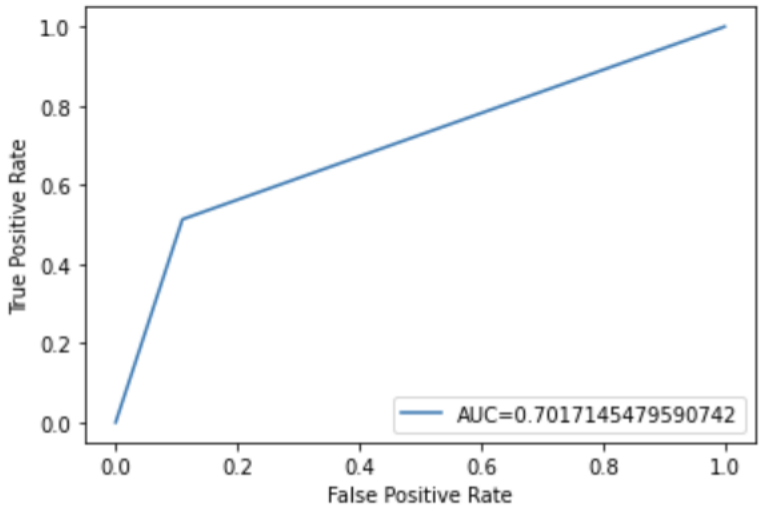
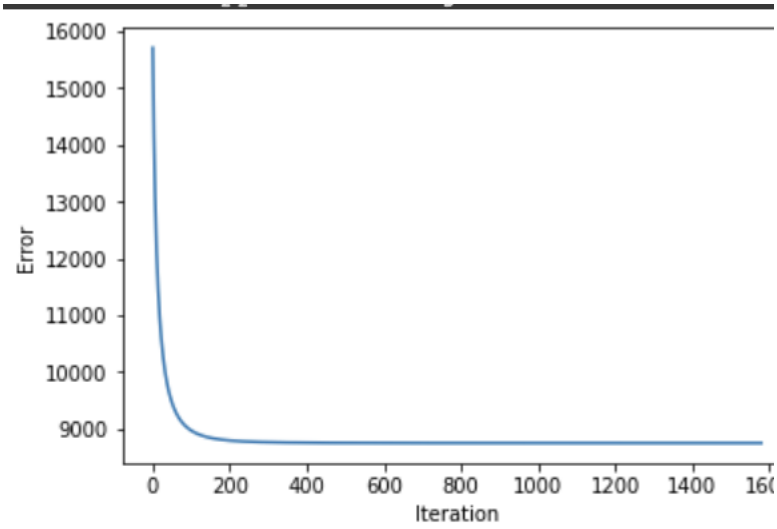
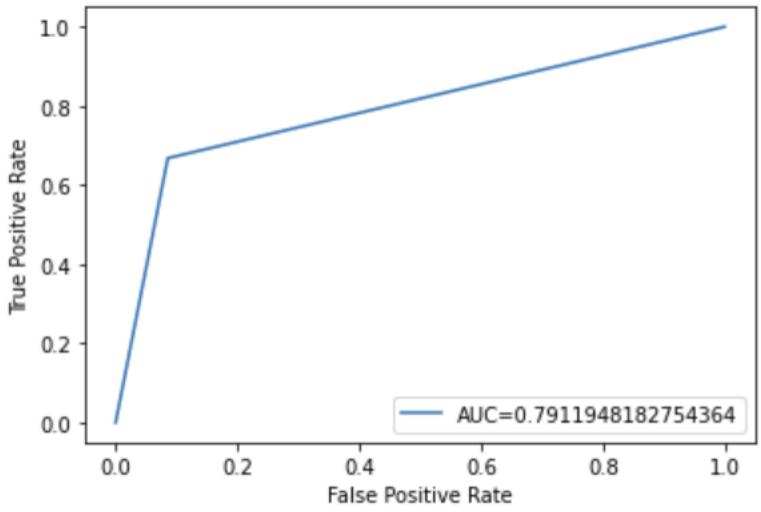


Figure 39 learning rate=0.00001, k=4





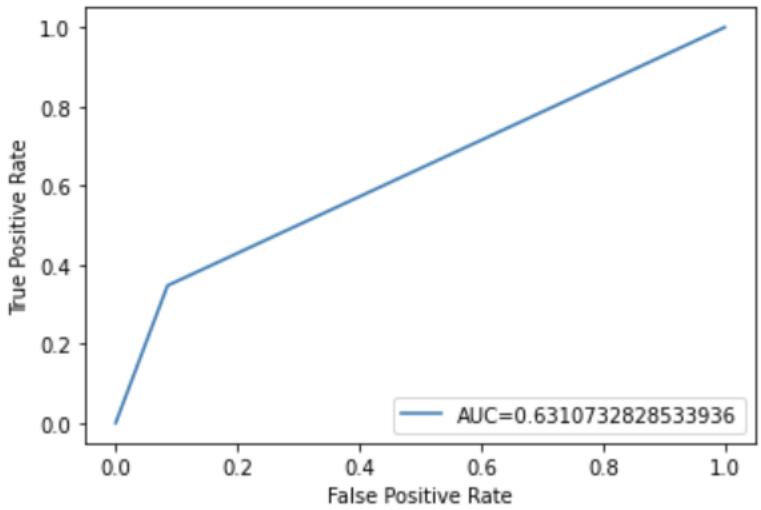


Figure 40 learning rate=0.00001, k=5

Naïve Bayes Model

Naïve Bayes Classification is also a probabilistic model that is used in machine learning, based on Bayes' Theorem. Bayes Theorem states that:

$$P(A|B) = \frac{P(B|A).P(A)}{P(B)}$$

Here, $P(B)$ is the evidence, $P(A)$ is the prior. Using these values, along with $P(B|A)$ help in calculating probability of event A occurring given that event B has occurred already. This classifier relies on a strong assumption that variables are independent. Using Bayes theorem, the probability of test data belonging to either class is determined by:

$$P(y|x_1, x_2 \dots, x_k) \propto P(y).\Pi_{i=1}^{n} P(x_i|y)$$

And then, the class with maximum probability is chosen as the final outcome:

$$y = argmax_y.P(y).\Pi_{i=1}^{n}.P(x_i|y)$$

Performance

The model was executed with over and under-sampled data, the over-sampled dataset performed better.

```
f1_score: 0.465
Accuracy : 0.789
Precision : 0.332
Recall : 0.78
Misclassified 1904
```

Data	Accuracy	Precision	Recall	f1_score	Misclassified
test	0.789	0.332	0.78	0.465	1904.0

Figure 41 performance of naive bayes with over-sampled dataset

Support Vector Machine Model

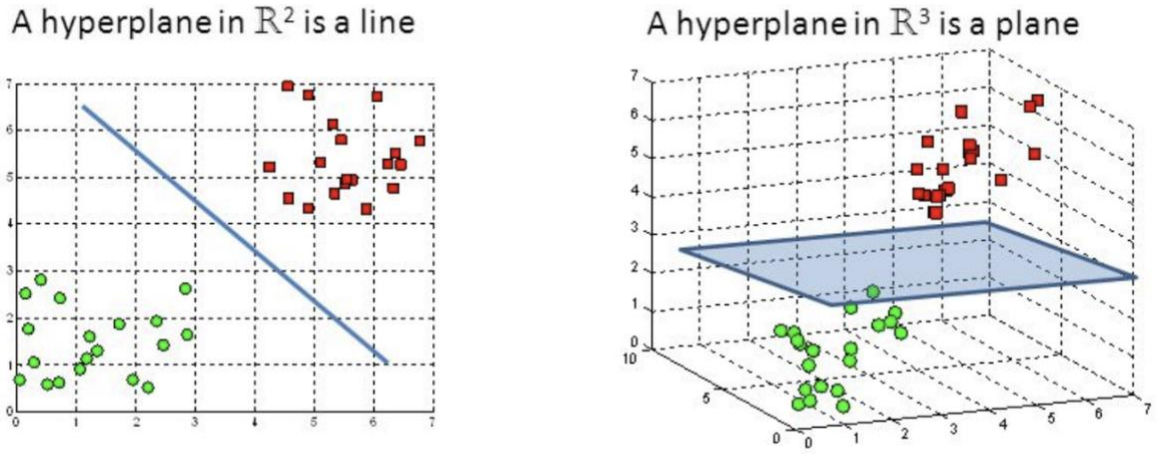


Figure 42 support vector model separating 2 classes

The support vector machine model looks for a decision boundary that can separate the data points into different regions as per their assigned class. In soft margin SVM, which was used for this dataset, a degree of misclassification is permitted to help the model optimize its fitting process. For each model, myriad hyperplanes are possible, but the goal is to find a hyperplane that maximizes the margin separating classes. This equation is used to calculate SVM:

$$L = \Sigma_i \lambda_i - \frac{1}{2} . \Sigma_i \Sigma_j \lambda_i \lambda_j y_i y_j x_i . x_j$$

Performance

20% of the dataset was sampled for practical implementation using Gaussian RBF kernel to capture complex, non-linear relationship.

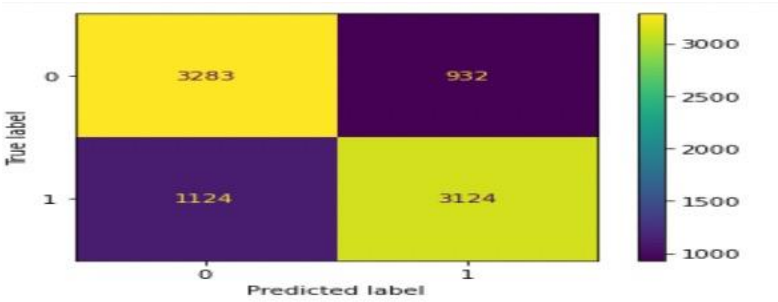


Figure 43 confusion matrix of model performance

Accuracy Score	Precision Score
0.7570601441569184	0.770216965246549

Neural Networks

Neural network model mimics the human nervous system and the relationships between neurons inside one's brain. Multi-layered perceptron (MLP) was implemented in this instance, which is a deep learning model. MLP makes use of the concept of back-propagation, which reinforces information by running a forward propagation and regularizing through a backward propagation in each epoch.

$$y = step(W.x + b)$$

$$W = \begin{matrix} w_1^T \\ \dots \\ w_u^T \end{matrix} = \begin{matrix} w_{1,1} & \dots & w_{1,n} \\ \dots & \dots & \dots \\ w_{u,1} & \dots & w_{u,n} \end{matrix}$$

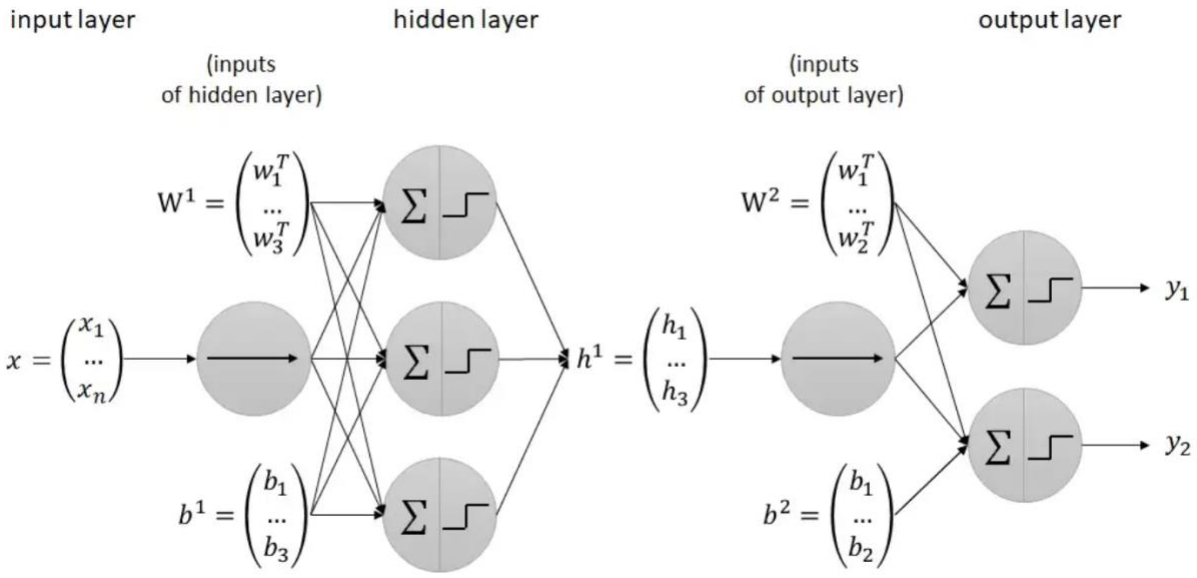


Figure 44 Multi-layer perceptron (hidden layer has 3 LTUs, output layer has 2 LTUs)

Performance

K-means validation was implemented by executing this model with three versions of the original dataset on several learning rates to evaluate its best performance. Original dataset, under-sampled dataset and over-sampled dataset were used with learning rates of 0.01, 0.001 and 0.1. **The best performer was obtained with over-sampled and normalized dataset using learning rate = 0.01.**

Original dataset, without under/over-sampling

Learning rate = 0.001

	precision	recall	f1-score	support
0	0.93	0.97	0.95	9972
1	0.64	0.42	0.51	1331
accuracy			0.90	11303
macro avg	0.79	0.70	0.73	11303
weighted avg	0.89	0.90	0.90	11303

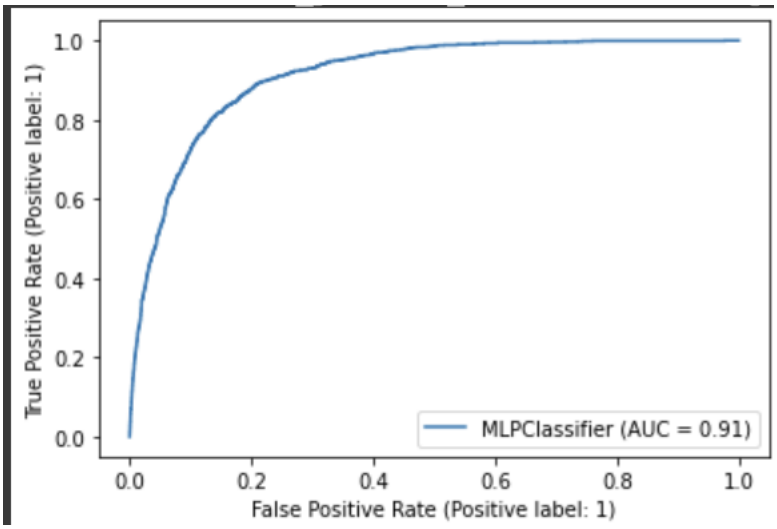


Figure 45 learning rate=0.001

Learning Rate = 0.01

	precision	recall	f1-score	support
0	0.93	0.96	0.95	10074
1	0.56	0.42	0.48	1229
accuracy			0.90	11303
macro avg	0.75	0.69	0.71	11303
weighted avg	0.89	0.90	0.89	11303

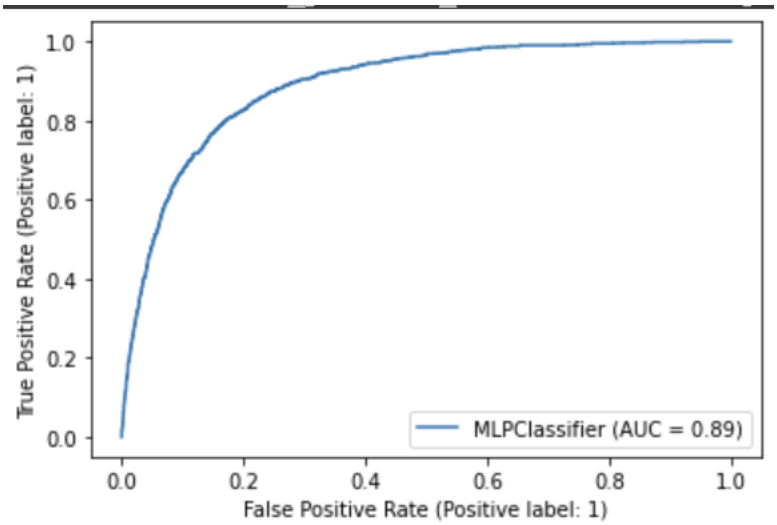


Figure 46 learning rate = 0.01

Learning rate = 0.001

	precision	recall	f1-score	support
0	0.87	0.88	0.87	2681
1	0.74	0.72	0.73	1286
accuracy			0.83	3967
macro avg	0.81	0.80	0.80	3967
weighted avg	0.83	0.83	0.83	3967

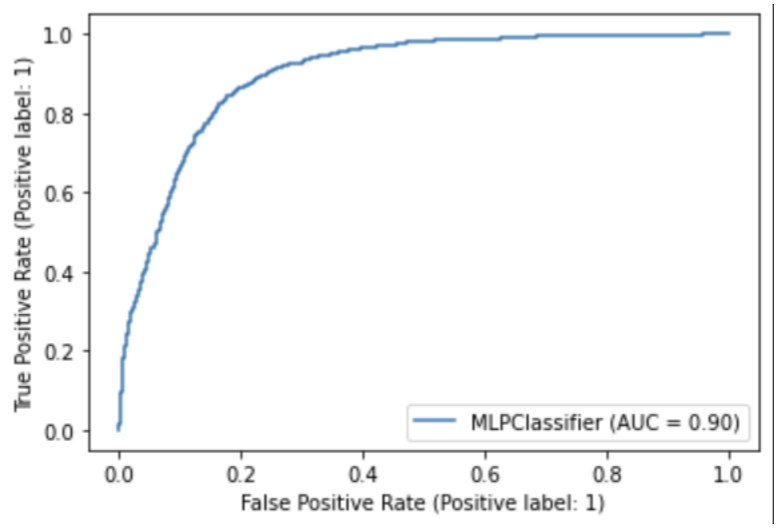


Figure 47 learning rate=0.01

Under-sampled and normalized dataset

Learning rate = 0.01

	precision	recall	f1-score	support
0	0.87	0.87	0.87	2641
1	0.74	0.74	0.74	1326
accuracy			0.83	3967
macro avg	0.80	0.81	0.80	3967
weighted avg	0.83	0.83	0.83	3967

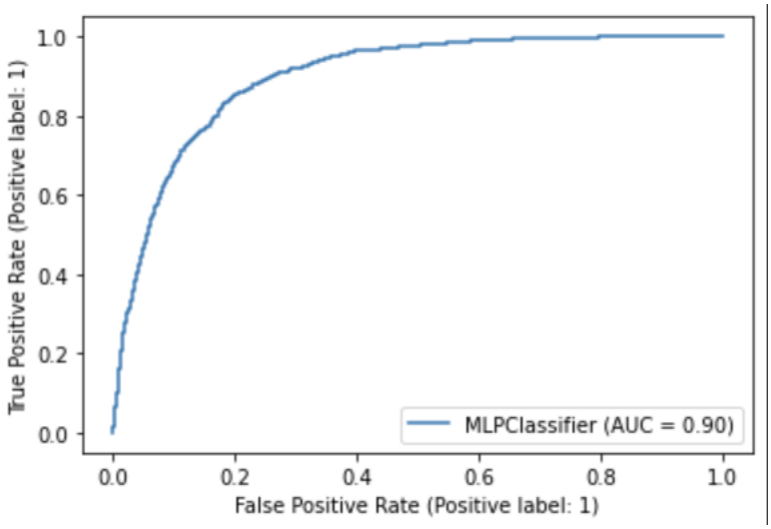


Figure 48 learning rate = 0.01

Learning rate = 0.1

	precision	recall	f1-score	support
0	0.85	0.86	0.86	2645
1	0.71	0.71	0.71	1322
accuracy			0.81	3967
macro avg	0.78	0.78	0.78	3967
weighted avg	0.81	0.81	0.81	3967

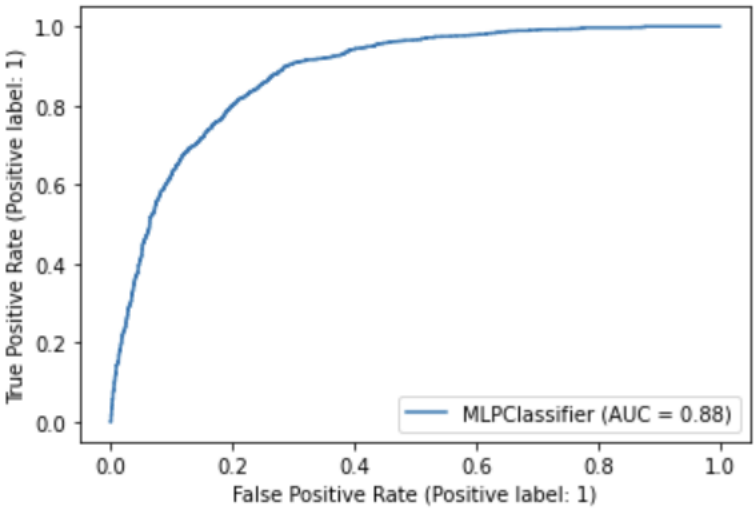


Figure 49 learning rate=0.1

Learning rate = 0.001

	precision	recall	f1-score	support
0	0.88	0.90	0.89	9952
1	0.78	0.75	0.77	5019
accuracy			0.85	14971
macro avg	0.83	0.82	0.83	14971
weighted avg	0.85	0.85	0.85	14971

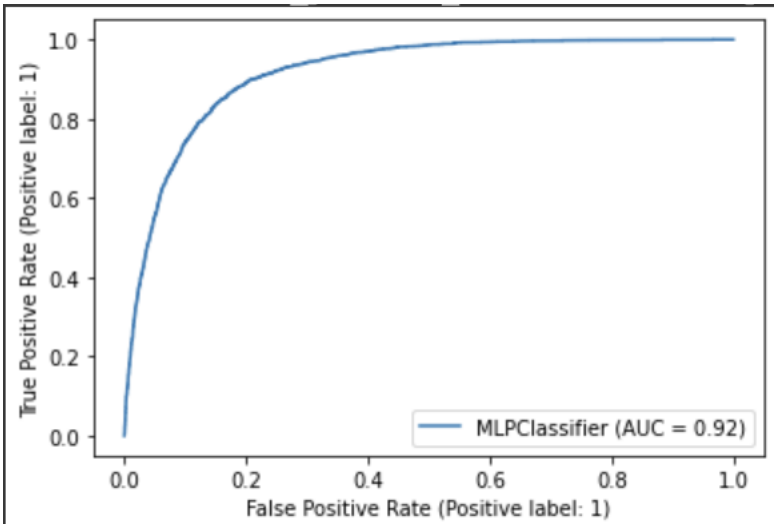


Figure 50 learning rate=0.001

Over-sampled and normalized dataset

Learning rate = 0.01

	precision	recall	f1-score	support
0	0.89	0.90	0.90	9954
1	0.80	0.78	0.79	5017
accuracy			0.86	14971
macro avg	0.84	0.84	0.84	14971
weighted avg	0.86	0.86	0.86	14971

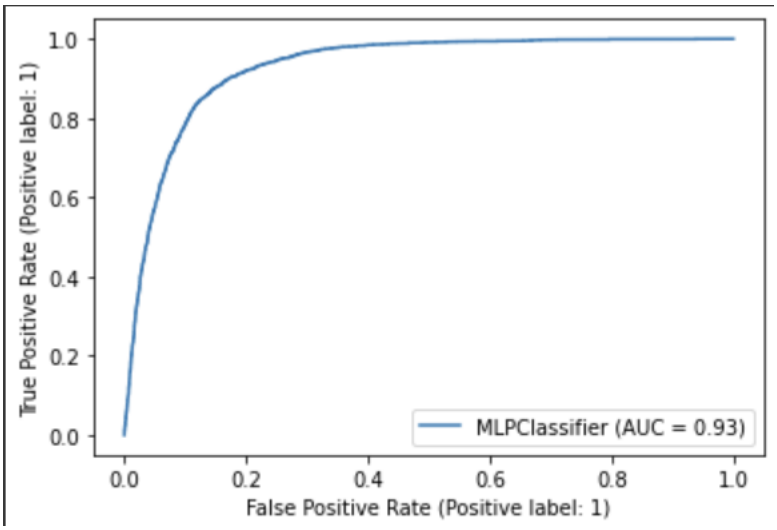


Figure 51 learning rate = 0.01

Learning rate = 0.1

	precision	recall	f1-score	support
0	0.90	0.86	0.88	9984
1	0.74	0.82	0.78	4987
accuracy			0.85	14971
macro avg	0.82	0.84	0.83	14971
weighted avg	0.85	0.85	0.85	14971

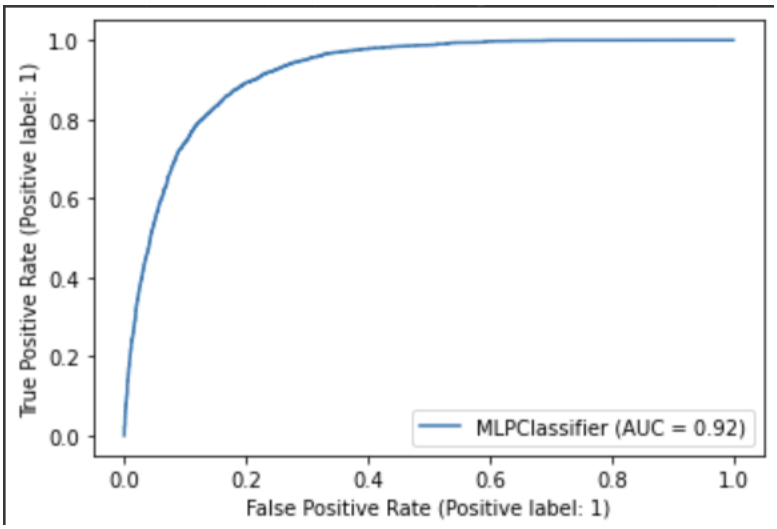
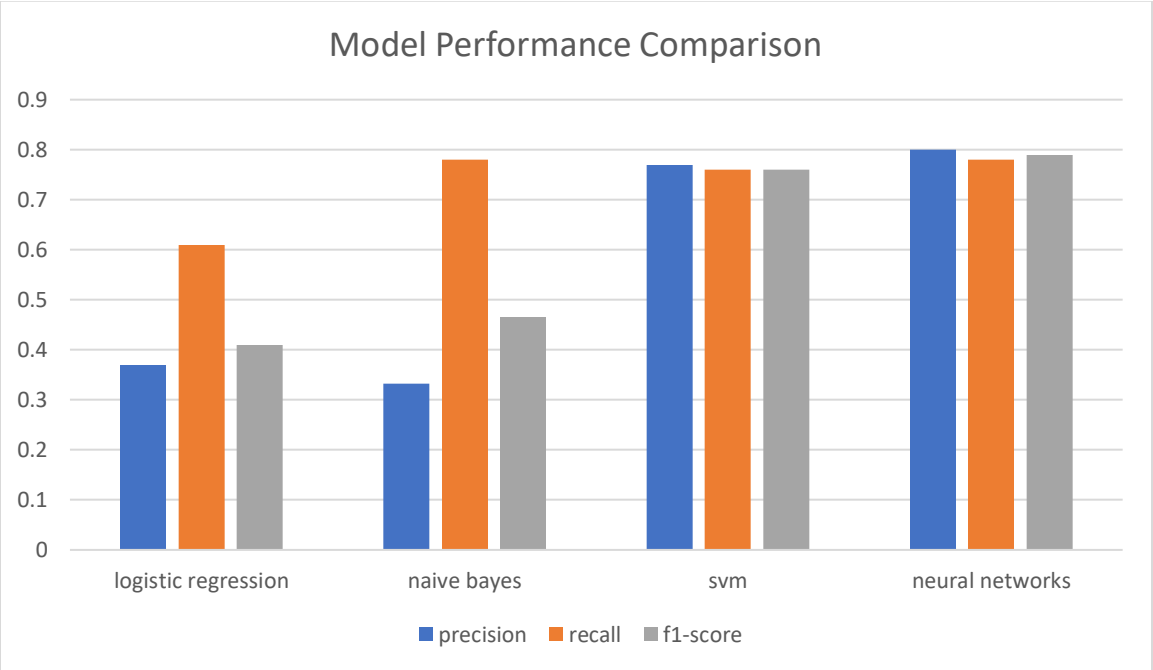


Figure 52 learning rate=0.1

Results

Precision (class=1)			
Logistic Regression	**Naïve Bayes**	**Support Vector Machine**	**Neural Networks**
0.3718	0.332	0.7728	0.80
Recall (class=1)			
Logistic Regression	**Naïve Bayes**	**Support Vector Machine**	**Neural Networks**
0.6128	0.78	0.7636	0.78
F1-score (class=1)			
Logistic Regression	**Naïve Bayes**	**Support Vector Machine**	**Neural Networks**
0.4166	0.465	0.7682	0.79



Discussion

The scope of this project can be expanded by the following recommendations:

- The code can be improved to optimize time complexity and space complexity.
- Performance of implemented models can be improved by further optimizing hyperparameters and feature selection. For example, SMO can be implemented in SVM model.
- More data can be introduced to reinforce the models' learnings or improve their results.