Machine Learning Engineer Nanodegree

Capstone Project Report

Deepika Kothapalli

28/06/18

Project Overview:

Banks contain huge information about the customers. They can use this data for several useful purposes. In order to have a good relationship with the customers and to market several other schemes. They will select a means of communication like phone call or SMS etc.
Sometimes Banks can use TV and other means to inform or market their offers and schemes. But customers may not pay a good interest in that. So, they use telephone as a medium where they can directly speak with the customer about the offers and clarify about certain things and know whether they are interested or not.
In this way they can get to know about the genuine feedback what they think of the certain offer. Through direct contact with the customer it is also possible to convince the customer about their ideas. This type of marketing product or service is called direct marketing which came into existence in 1960's.

Problem Statement:

- In this I want to determine whether the customer subscribe to the campaign or not. I want to classify the customers subscribed and unsubscribed to the campaign.
- There are certain features based on which I want to classify the data points like age, type of job, marital status, education,

loan, housing, number of days before the bank contacted the customer etc.

- I decided to use several supervised learning classification algorithms like Decision trees, logistic regression etc. I will find the best model among those using many performance metrics through which I can get good results.
- This project consists of several phases like Data Exploration, Data preprocessing, application of various classification algorithms, finding the best model etc.

Features and Description:

- age: age of the customer.
- Present: occupation of the customer ('admin', 'bluecollar', 'entrepreneur', 'housemaid', 'management', 'retired', 'self-employed', 'services', 'student', 'technician', 'unemployed', 'unknown')
- marital: Marital status of the customer ('single', 'married', 'divorced', 'unknown')
- default : whether the customer has a credit in default ('yes', 'no', 'unknown')
- balance: current balance in the account.
- housing: whether the customer has housing loan ('yes', 'no', 'unknown')
- loan: whether the customer has personal loan ('yes', 'no', 'unknown')
- contact : contact communication type ('cellular','telephone')
- day : days before which the customer is contacted
- month : last contact month of year ('jan', 'feb', 'mar', ..., 'nov', 'dec')
- duration : last contact duration, in seconds
- campaign : number of contacts performed during this campaign and for this client (numeric, includes last contact)
- pdays: number of days that passed by after the client was last contacted from a previous campaign

- Previous : number of contacts performed before this campaign and for this client
- poutcome: outcome of the previous contact('success','failure','unknown').

Metrics :

Accuracy : It determines the proportion of correct predicts among all the predictions

$$Accuracy = \frac{(TP + TN)}{(TP + TN + FP + FN)}$$

Precision :

$$Precision = \frac{(TP)}{(TP + FP)}$$

It determines among all the customers that are predicted as subscribed to the campaign who are actually subscribed.

Recall :

$$Recall = \frac{(TP)}{(TP + FN)}$$

Recall determines the proportion of subscribed customers correctly predicted among the actual subscribed customers.

F-Beta score :

F-beta score is the weighted harmonic mean of precision and recall.

$$F - beta = \frac{(1 + \beta 2) * Precision * recall}{\beta 2 * \text{precision} + \text{recall}}$$

F-beta=(1+ β2) *precision *recall/(β2*precision+recall)

Data Exploration:

In this section I have calculated the total number of records, number of customers of the bank subscribed for the campaign, number of the customers who are unsubscribed for the campaign. Finally the percentage of customers subscribed for the campaign among all the customers.

```
Total number of records: 4521
The number of customers subscribed: 521
The number of customers does not subscribe: 4000
The percentage of customers subscribed: 11.5239991152%
```

From the exploration I found that the number of customers subscrib ed for the campaign are very few among all the customers of the bank.

data.describe()

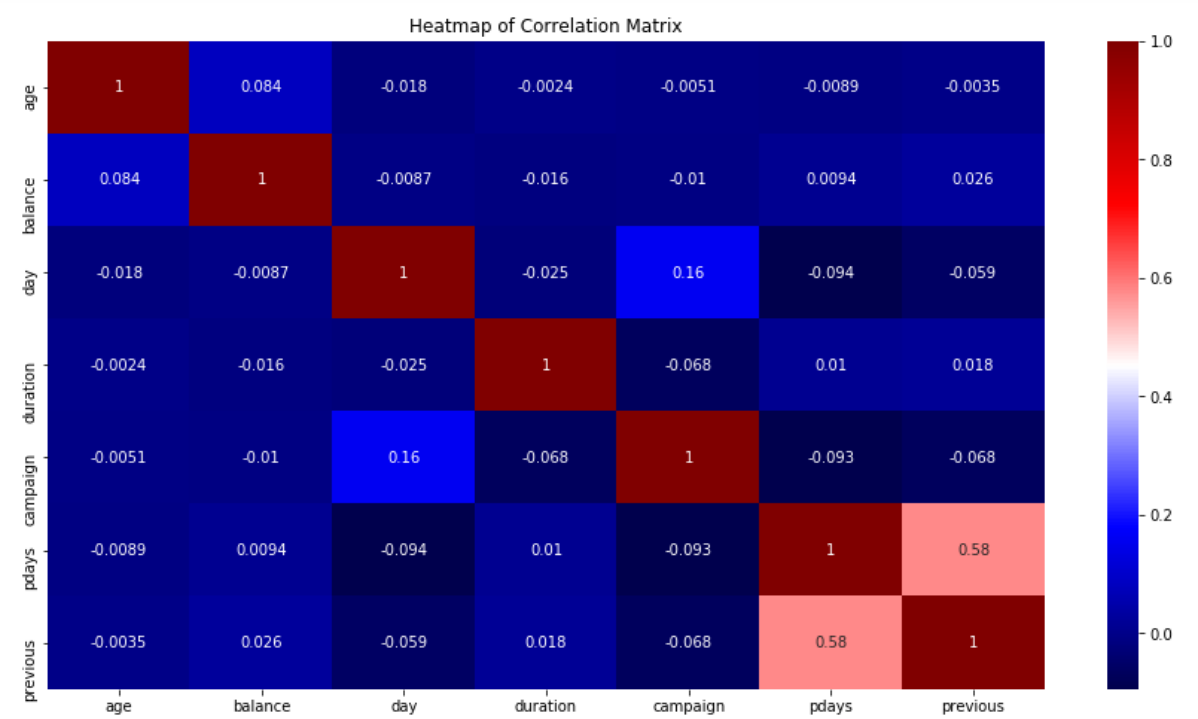
	age	balance	day	duration	campaign	pdays	previous
count	4521.000000	4521.000000	4521.000000	4521.000000	4521.000000	4521.000000	4521.000000
mean	41.170095	1422.657819	15.915284	263.961292	2.793630	39.766645	0.542579
std	10.576211	3009.638142	8.247667	259.856633	3.109807	100.121124	1.693562
min	19.000000	-3313.000000	1.000000	4.000000	1.000000	-1.000000	0.000000
25%	33.000000	69.000000	9.000000	104.000000	1.000000	-1.000000	0.000000
50%	39.000000	444.000000	16.000000	185.000000	2.000000	-1.000000	0.000000
75%	49.000000	1480.000000	21.000000	329.000000	3.000000	-1.000000	0.000000
max	87.000000	71188.000000	31.000000	3025.000000	50.000000	871.000000	25.000000

data.head()

	age	job	marital	education	default	balance	housing	loan	contact	day	month	duration	campaign	pdays	previous	poutcome	y
0	30	unemployed	married	primary	no	1787	no	no	cellular	19	oct	79	1	-1	0	unknown	no
1	33	services	married	secondary	no	4789	yes	yes	cellular	11	may	220	1	339	4	failure	no
2	35	management	single	tertiary	no	1350	yes	no	cellular	16	apr	185	1	330	1	failure	no
3	30	management	married	tertiary	no	1476	yes	yes	unknown	3	jun	199	4	-1	0	unknown	no
4	59	blue-collar	married	secondary	no	0	yes	no	unknown	5	may	226	1	-1	0	unknown	no

Visualization:

In Visualization first I plotted a heat matrix to determine the correlation between features. After plotting the heat matrix I found that the previous and pdays are highly correlated with the highest value 0.58.



Algorithms and Techniques :

I have used four supervised classification algorithms for this model. They are: Logistic Regression, DecisionTreeClassifier, AdaBoostClassifier, GradientBoostingClassifier.

Logistic Regression:

Logistic regression is a supervised learning model used for classification problems. This model is used for binary classification

where the result is either true or false. Hence this model can be applied in the present context and it is the benchmark model.

Logistic regression determines the probability that the data point belongs to a particular class. For example consider the given problem. It is to determine whether the customer subscribes for the campaign or not. If the probability of a person subscribed for the campaign is 0.9. He/she belongs to a class subscribed as it has a high probability. We can set a treshhold limit to determine the class of the data point.

AdaBoostClassifier :

AdaBoostClassifier is one of the ensemble methods that critically attempts to build a strong classifier considerably from a number of weak classifiers. This algorithm is designed for binary classification.

The classical problems that the systems designed using machine learning suffer is The Dimensionality Curse i.e., the potential risk of having a tremendous amount of input features and hence it is quite obvious that the evaluation of every feature can potentially reduce not only the classifier training and execution speed but may also result in the reduction of predictive capability of the system.

Besides the application of the algorithms like Support Vector Machines, Neural Networks etc. , the AdaBoost Algorithm extracts the features that potentially aid in the optimization of the predictive capability of the system and further reducing the dimensionality and eventually improving the computation(execution) time of the model as the irrelevant attributes are not considered by the algorithm.

DecisionTreeClassifier:

DecisionTreeClassifier is one of the supervised learning models used for classification. This model splits the data based on the features. Each node represents the decision on features based on which the data has to be splitted. The leaves of the decision tree tells the final output .

A decision tree splits the data based on the decisions at each node. If the tree split is high. It may lead to overfitting which can be controlled by pruning the tree.

GradientBoostClassifier :

GradientBoostClassifier is one of the ensemble methods. Sometimes normal algorithms cannot determine the parameters well. Hence they cannot be able to predict the data points well. So the performance may be poor.

But GradientBoostClassifier recursively determines the best parameters that contribute highly to prediction. This helps in accurately predicting the outcomes. Hence, this is one of the best models for prediction.

GradientBoosting uses loss function for optimization.

Benchmark model :

Logistic regression is used as benchmark model. F-Beta score and accuracy score of benchmark model is used as reference and other model will be judged to perform better if their f-beta score and accuracy score will be greater than Logistic regression model. Accuracy, f-beta score get better results in the ensemble learning models.

```
Accuracy score for logistic regression : 0.892817679558
f-score for logistic regression :0.663871260199
```

Data Preprocessing :

- As there is skewness in the data I have normalized the numeric data by first applying logarithmic transformation and then scaled the data using MinmaxScaler
- I have splitted the data into 80% training data and 20% testing data using train_test_split()
- As many values are categorical values, so we need to bring them to a scale.
- I have performed one hot encoding technique to convert the categorical data into numeric data

Code :

```
features_final = pd.get_dummies(features_log_minmax_transform)
target = target.map(lambda x : 0 if x == 'no' else 1)
features_final.head()
```

Output :

age	balance	day	duration	campaign	pdays	previous	job_admin.	job_blue-collar	job_entrepreneur	...	month_jun	month_mar	month_may	montl
0.295798	0.670258	0.830482	0.432841	0.000000	0.000000	0.000000	0	0	0	...	0	0	0	
0.358144	0.758455	0.646241	0.591474	0.000000	0.860896	0.493981	0	0	0	...	0	0	1	
0.396723	0.645175	0.771866	0.564558	0.000000	0.856934	0.212746	0	0	0	...	0	0	0	
0.295798	0.653156	0.250000	0.575887	0.282921	0.000000	0.000000	0	0	0	...	1	0	0	
0.741502	0.000000	0.396241	0.595656	0.000000	0.000000	0.000000	0	1	0	...	0	0	1	

Implementation :

Logistic Regression :

Logistic regression determines the probability that the data point belongs to a particular class.

For example consider the given problem. It is to determine whether the customer subscribes for the campaign or not. If the probability of a person subscribed for the campaign is 0.9. He/she belongs to a class subscribed as it has a high probability.

I will import this model form sklearn.linear_model. I have random_state as a parameter to this classifier.

Code:

```
from sklearn import linear_model
l =  linear_model.LogisticRegression(random_state=20)
l_fit = l.fit(X_train,y_train)
l_pred = l_fit.predict(X_test)
score=accuracy_scorer(y_test,l_pred)
f_score = fbeta_scorer(y_test,l_pred)
print "Accuracy score for DecisionTreeClassifier :
{}".format(dec_score)
print "f-score for DecisionTreeClassifier :{}".format(dec_f_score)
```

Output:

```
Accuracy score for logistic regression : 0.892817679558
f-score for logistic regression :0.663871260199
```

Initial Model Evaluation:-

AdaBoostClassifier :

AdaBoostClassifier is one of the ensemble methods. This model is mainly used for boosting the machine learning models. As some algorithms cannot determine the parameters well, they cannot accurately predict the outcomes.

This classifier is used to boost such kind of models.

I have imported this model from sklearn. ensemble. I have passed only random_state as parameter to this classifier.

Code :

```
from sklearn.ensemble import AdaBoostClassifier
ada_clf=AdaBoostClassifier(random_state=20)
ada_fit=ada_clf.fit(X_train,y_train)
ada_pred=ada_fit.predict(X_test)
ada_score=accuracy_scorer(y_test,ada_pred)
ada_f_score=fbeta_scorer(y_test,ada_pred)
print "Accuracy score for AdaBoostClassifier : {}".format(ada_score)
print "f-score for AdaBoostClassifier:{}".format(ada_f_score)
```

Output :

```
Accuracy score for AdaBoostClassifier : 0.893922651934
f-score for AdaBoostClassifier:0.605031948882
```

DecisionTreeClassifier :

This model splits the data based on the features. Each node represents the decision on features based on which the data has to be splitted. The leaves of the decision tree tells the final output .

I have imported this model from sklearn.tree. I have passed random_state as parameter to this model.

Code :

```
from sklearn.tree import DecisionTreeClassifier
dec_clf = DecisionTreeClassifier(random_state=100)
dec_fit = dec_clf.fit(X_train,y_train)
dec_pred = dec_fit.predict(X_test)
dec_score = accuracy_scorer(y_test,dec_pred)
dec_f_score = fbeta_scorer(y_test,dec_pred)
print "Accuracy score for DecisionTreeClassifier : {}".format(dec_score)
print "f-score for DecisionTreeClassifier :{}".format(dec_f_score)
```

Output:

```
Accuracy score for DecisionTreeClassifier : 0.850828729282
f-score for DecisionTreeClassifier :0.392341842397
```

GradientBoostingClassifier :

GradientBoostClassifier recursively determines the best parameters that contribute highly to prediction. This helps in accurately predicting the outcomes.

I have imported this model from sklearn.ensemble. I have passed random_state as parameter to this model.

Code :

```
from sklearn.ensemble import GradientBoostingClassifier
gbc_clf = GradientBoostingClassifier(random_state=20)
gbc_fit = gbc_clf.fit(X_train,y_train)
gbc_pred = gbc_clf.predict(X_test)
gbc_score = accuracy_scorer(y_test,gbc_pred)
gbc_f_score = fbeta_scorer(y_test,gbc_pred)
print "Accuracy score for GradientBoostingClassifier : {}".format(gbc_score)
print "f-score for GradientBoostingClassifier:{}".format(gbc_f_score)
```

Output :

```
Accuracy score for GradientBoostingClassifier : 0.891712707182
f-score for GradientBoostingClassifier:0.593955142232
```

Refinement :

First to improve the performance of all the models on the dataset I h ave removed some unnecessary features based on the ranking of RFE . So that It may produce some better results.

I have applied all the four models on the customers data. Benchmark model i.e., logistic regression has an accuracy score of 0.89 anf f-scor e of 0.66.

After that I have applied other algorithms I mentioned above to chec k the performance of those models when compared to the bench m ark model.

Among all the models applied AdaBoostClassifier has the f-score and accuracy score better but less than the logistic regression.

I thought of optimizing the AdaBoostClassifier using GridSearch as it may perform better than the Benchmark model after optimization.

Even after optimization of the AdaBoostClassifier the performance of it han't improved much. It's f-score is only increased at a very small ra te i.e, from 0.60 to 0.61 but it is still less than the benchmark model. So, I chose it as a best model and further optimized it using GridSearc h. Then its performance is increased from 0.66 to 0.86 which is very h igh improvement.

Results :

Model Evaluation and Validation :

After applying different algorithms on the data set the benchmark mo del logistic regression is performing well when compared to all the m odels with an accuracy score of 0.89 and f-score of 0.66.

After that I have tried to optimize the AdaBoostClassifier to check wh ether its score will increase than the benchmark model as it is having the next good accuracy and f-score after logistic regression.

But the score doesn't improved very much for AdaBoostClassifier eve n after optimization using GridSearch.

Hence, I have finalized that logistic regression is the best model for th is data set and optimized it for having better results.The f-score of the best model i.e., logistic regression has improved a lot after optimizati

on. It increased from 0.66 to 0.86 which is very high

```
GridSearchCV(cv=None, error_score='raise',
       estimator=LogisticRegression(C=1.0, class_weight=None, dual=False, fit_intercept=True,
          intercept_scaling=1, max_iter=100, multi_class='ovr', n_jobs=1,
          penalty='l2', random_state=20, solver='liblinear', tol=0.0001,
          verbose=0, warm_start=False),
       fit_params={}, iid=True, n_jobs=1,
       param_grid={'C': [0.0001, 0.001, 0.01, 0.1]},
       pre_dispatch='2*n_jobs', refit=True,
       scoring=make_scorer(fbeta_score, beta=0.5), verbose=0)
LogisticRegression(C=0.1, class_weight=None, dual=False, fit_intercept=True,
          intercept_scaling=1, max_iter=100, multi_class='ovr', n_jobs=1,
          penalty='l2', random_state=20, solver='liblinear', tol=0.0001,
          verbose=0, warm_start=False)
```

Justification:

The final model that I have chosen is the best model when compared to the other models I have tested as the f-score of the logistic regressi on is very high when compared to the other models. No other model even reached that f-score. AdaboostClassifier has some good f-score nearer to logistic regression with an f-score of 0.60 . But after optimiz ation its score han't gone beyond 0.61. Whereas after optimization f-score of logistic has become 0.86 which is very high.

Unoptimized model :

```
Accuracy score for logistic regression : 0.892817679558
f-score for logistic regression :0.663871260199
```

Optimized model :

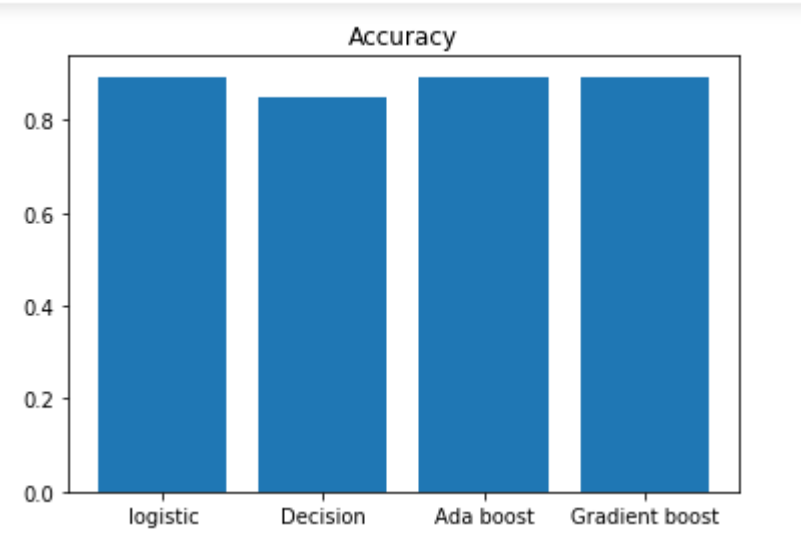
```
The final accuracy of the best model is 0.889502762431
The final f-score of the best model is 0.868386243386
```

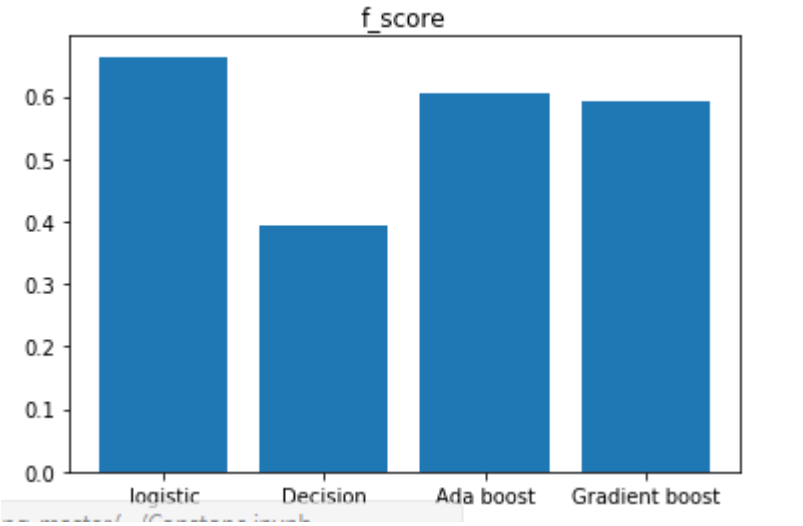
Free-Form Visualization :

I have visualised the accuracy and f-score of all the models and deter mine the best model before and after the Optimization. Before and af ter the optimization f-score and accuracy score of the bench mark mo del is high when compared to the other models with an accuracy scor

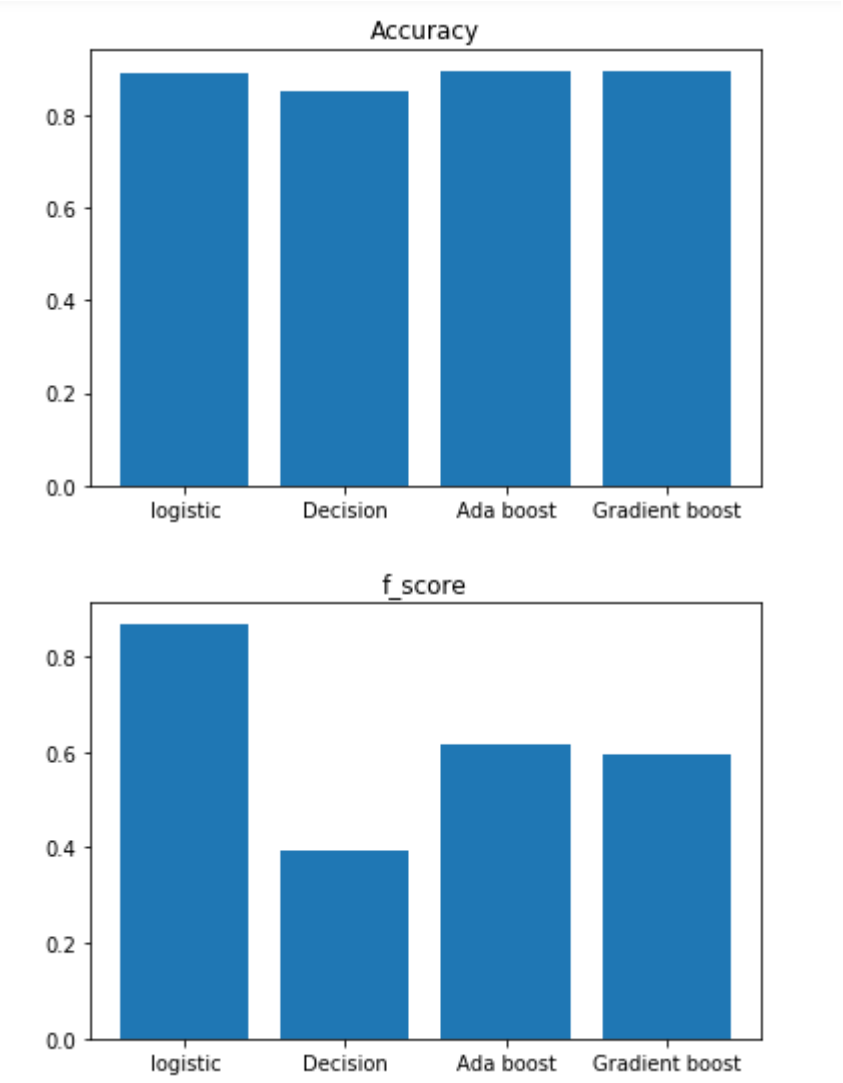
e of 0.89 , f-score- 0.66 before optimization and accuracy score -0.89, f-score -0.86 after optimization

Before optimization :





After optimization :



Reflection:

Initially I load the data from the bank.csv file using read_csv() . After t hat I started data exploration. I find the total number of records, num ber of customers who subscribed for the campaign, the customers wh o are not subscribed and the percentage of customers subscribed for the campaign.

After that I have plotted to histograms to determine the skewness of the data. After finding that the data is skewed, I started normalizing t he data. I first applied logarithmic transformation and then performe d minmaxscaling to scale down the data.

As there are more categorical features in the data, I have applied one hot encoding technique to convert the categorical data to numeric.

After that I removed some of the features that are not found essentia l for prediction by ranking the features using RFE.

After preprocessing I have splitted the data into 80% training and 20 % testing set and applied all the best algorithms on the data set.

I have used some evaluation metrics like Accuracy and f-score to fin d the best algorithm.

Finally I optimized the model using Grid search Which improved the p erformance of the best model logistic regression.

Improvements:

We can apply several more algorithms on this dataset So that we can obtain a more wise model which produces best results. We can also use some review metrics like log-loss to determine how quickly the model will be able to tune.

References:

https://matplotlib.org/tutorials/introductory/sample_plots.html

https://machinelearningmastery.com/gentle-introduction-gradientboosting-algorithm-machine-learning/

http://scikitlearn.org/stable/modules/generated/sklearn.feature_selection.RFE.html