FLIP ROBO

Malignant Comments Classifier Project

Submitted by:

NISHANT POKHRIYAL

ACKNOWLEDGMENT

I would like to express my special thanks of gratitude to FlipRobo who gave me this opportunity to do this wonderful project on Malignant Comments Classifier Project, which also helped me in doing a lot of research about how to catch which comments are bad and which are good, I leaned about stopwords and worlcloud and I came to know about a lot of other new things I am really thankful to them.

I am highly indebted to FlipRobo Technologies for their guidance and constant supervision as well as for providing necessary information regarding the project & also for their support in completing the project.

INTRODUCTION

- ## Business Problem Framing

The proliferation of social media enables people to express their opinions widely online. However, at the same time, this has resulted in the emergence of conflict and hate, making online environments uninviting for users. So we have to create a machine learning model which can predict the bad comments.

- ## Conceptual Background of the Domain Problem

Online hate, described as abusive language, aggression, cyberbullying, hatefulness and many others has been identified as a major threat on online social media platforms. Social media platforms are the most prominent grounds for such toxic behaviour.

Our goal is to build a prototype of online hate and abuse comment classifier which can used to classify hate and offensive comments so that it can be controlled and restricted from spreading hatred and cyberbullying.

- ## Review of Literature

Internet comments are bastions of hatred and vitriol. While online anonymity has provided a new outlet for aggression and hate speech, machine learning can be used to fight it.

We have saved the data in a Dataframe, We have done the required exploratory data analysis such as we have counted the length of comments and reduced the extra length by converting email, phone numbers, web address into shorter form. Plotted the data of offensive words by using wordcloud.

We have build several machine learning models to predict the bad comments. The best performing model is Logistic Regression with the accuracy of approximately 95.55%. Other models accuracy are (Decision Tree classifier :94%),(Random Forest Classifier : 95.51%),(Ada Booster Classifier : 94.9%) and (KNeighbors Classifier : 91.7%).

- Motivation for the Problem Undertaken

There has been a remarkable increase in the cases of cyberbullying and trolls on various social media platforms. Many celebrities and influences are facing backlashes from people and have to come across hateful and offensive comments. This can take a toll on anyone and affect them mentally leading to depression, mental illness, self-hatred and suicidal thoughts.

Analytical Problem Framing

- Data Sources and their formats

  The data set contains the training set, which has approximately 1,59,000 samples and the test set which contains nearly 1,53,000 samples. All the data samples contain 8 fields which includes ‘Id’, ‘Comments’, ‘Malignant’, ‘Highly malignant’, ‘Rude’, ‘Threat’, ‘Abuse’ and ‘Loathe’.

  The label can be either 0 or 1, where 0 denotes a NO while 1 denotes a YES. There are various comments which have multiple labels. The first attribute is a unique ID associated with each comment.

  The data set includes:

- **Malignant:** It is the Label column, which includes values 0 and 1, denoting if the comment is malignant or not.
- **Highly Malignant:** It denotes comments that are highly malignant and hurtful.
- **Rude:** It denotes comments that are very rude and offensive.
- **Threat:** It contains indication of the comments that are giving any threat to someone.
- **Abuse:** It is for comments that are abusive in nature.
- **Loathe:** It describes the comments which are hateful and loathing in nature.
- **ID:** It includes unique Ids associated with each comment text given.
  Comment text: This column contains the comments extracted from various social media platforms.

	id	comment_text	malignant	highly_malignant	rude	threat	abuse	loathe
0	0000997932d777bf	Explanation\nWhy the edits made under my usern...	0	0	0	0	0	0
1	000103f0d9cfb60f	D'aww! He matches this background colour I'm s...	0	0	0	0	0	0
2	000113f07ec002fd	Hey man, I'm really not trying to edit war. It...	0	0	0	0	0	0
3	0001b41b1c6bb37e	"\nMore\nI can't make any real suggestions on ...	0	0	0	0	0	0
4	0001d958c54c6e35	You, sir, are my hero. Any chance you remember...	0	0	0	0	0	0
...	...	...	...	...	...	...	...	...
159566	ffe987279560d7ff	":::::And for the second time of asking, when ...	0	0	0	0	0	0
159567	ffea4adeee384e90	You should be ashamed of yourself \n\nThat is ...	0	0	0	0	0	0
159568	ffee36eab5c267c9	Spitzer \n\nUmm, theres no actual article for ...	0	0	0	0	0	0
159569	fff125370e4aaaf3	And it looks like it was actually you who put ...	0	0	0	0	0	0
159570	fff46fc426af1f9a	"\nAnd ... I really don't think you understand...	0	0	0	0	0	0

159571 rows × 8 columns

	id	comment_text
0	00001cee341fdb12	Yo bitch Ja Rule is more succesful then you'll...
1	0000247867823ef7	== From RfC == \n\n The title is fine as it is...
2	00013b17ad220c46	" \n\n == Sources == \n\n * Zawe Ashton on Lap...
3	00017563c3f7919a	:If you have a look back at the source, the in...
4	00017695ad8997eb	I don't anonymously edit articles at all.
...	...	...
153159	fffcd0960ee309b5	. \n i totally agree, this stuff is nothing bu...
153160	fffd7a9a6eb32c16	== Throw from out field to home plate. == \n\n...
153161	fffda9e8d6fafa9e	" \n\n == Okinotorishima categories == \n\n I ...
153162	fffe8f1340a79fc2	" \n\n == ""One of the founding nations of the...
153163	ffffce3fb183ee80	" \n :::Stop already. Your bullshit is not wel...

153164 rows × 2 columns

- Data Preprocessing Done

```
tr.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 159571 entries, 0 to 159570
Data columns (total 8 columns):
 #   Column            Non-Null Count   Dtype 
---  ------            --------------   ----- 
 0   id                159571 non-null  object
 1   comment_text      159571 non-null  object
 2   malignant         159571 non-null  int64 
 3   highly_malignant  159571 non-null  int64 
 4   rude              159571 non-null  int64 
 5   threat            159571 non-null  int64 
 6   abuse             159571 non-null  int64 
 7   loathe            159571 non-null  int64 
dtypes: int64(6), object(2)
memory usage: 9.7+ MB
```

No null values are present in the data

```
tr['length'] = tr['comment_text'].str.len()
tr.head()
```

	id	comment_text	malignant	highly_malignant	rude	threat	abuse	loathe	length
0	0000997932d777bf	Explanation\nWhy the edits made under my usern...	0	0	0	0	0	0	264
1	000103f0d9cfb60f	D'aww! He matches this background colour I'm s...	0	0	0	0	0	0	112
2	000113f07ec002fd	Hey man, I'm really not trying to edit war. It...	0	0	0	0	0	0	233
3	0001b41b1c6bb37e	"\nMore\nI can't make any real suggestions on ...	0	0	0	0	0	0	622
4	0001d958c54c6e35	You, sir, are my hero. Any chance you remember...	0	0	0	0	0	0	67

Created length column for length of comments

```
# Convert all messages to lower case
tr['comment_text'] = tr['comment_text'].str.lower()

# Replace email addresses with 'email'
tr['comment_text'] = tr['comment_text'].str.replace(r'^.+@[^\.].*\.[a-z]{2,}$',
                                 'emailaddress')

# Replace URLs with 'webaddress'
tr['comment_text'] = tr['comment_text'].str.replace(r'^http\://[a-zA-Z0-9\-\.]+\.[a-zA-Z]{2,3}(/\S*)?$',
                                  'webaddress')

# Replace money symbols with 'moneysymb' (£ can by typed with ALT key + 156)
tr['comment_text'] = tr['comment_text'].str.replace(r'£|\$', 'dollers')

# Replace 10 digit phone numbers (formats include paranthesis, spaces, no spaces, dashes) with 'phonenumber'
tr['comment_text'] = tr['comment_text'].str.replace(r'^\(?[\d]{3}\)?[\s-]?[\d]{3}[\s-]?[\d]{4}$',
                                  'phonenumber')


# Replace numbers with 'numbr'
tr['comment_text'] = tr['comment_text'].str.replace(r'\d+(\.\d+)?', 'numbr')


tr['comment_text'] = tr['comment_text'].apply(lambda x: ' '.join(
    term for term in x.split() if term not in string.punctuation))

stop_words = set(stopwords.words('english') + ['u', 'ü', 'ur', '4', '2', 'im', 'dont', 'doin', 'ure'])
tr['comment_text'] = tr['comment_text'].apply(lambda x: ' '.join(
    term for term in x.split() if term not in stop_words))

lem=WordNetLemmatizer()
tr['comment_text'] = tr['comment_text'].apply(lambda x: ' '.join(
 lem.lemmatize(t) for t in x.split()))
```

Reducing the length of the comments

```
tr['clean_length'] = tr.comment_text.str.len()
tr.head()
```

	id	comment_text	malignant	highly_malignant	rude	threat	abuse	loathe	length	clean_length
0	0000997932d777bf	explanation edits made username hardcore metal...	0	0	0	0	0	0	264	180
1	000103f0d9cfb60f	d'aww! match background colour i'm seemingly s...	0	0	0	0	0	0	112	111
2	000113f07ec002fd	hey man, i'm really trying edit war. guy const...	0	0	0	0	0	0	233	149
3	0001b41b1c6bb37e	can't make real suggestion improvement wondere...	0	0	0	0	0	0	622	397
4	0001d958c54c6e35	you, sir, hero. chance remember page that's on?	0	0	0	0	0	0	67	47

Created Clean Length column.

```
target_data = tr[cols_target]

tr['bad'] =tr[cols_target].sum(axis =1)
print(tr['bad'].value_counts())
tr['bad'] = tr['bad'] > 0
tr['bad'] = tr['bad'].astype(int)
print(tr['bad'].value_counts())
```

```
0    143346
1      6360
3      4209
2      3480
4      1760
5       385
6        31
Name: bad, dtype: int64
0    143346
1     16225
Name: bad, dtype: int64
```

Created separate column for bad comments including all the other columns data.

- Data Inputs- Logic- Output Relationships

  The comment is bad depends on the words used in it, it can be bad because it is malignant, highly malignant, rude, threat , abuse and loathe.

- Hardware and Software Requirements and Tools Used

  1. Jupyter Notebook
  2. Module : Pickle
  3. Packages : Pandas, Numpy, sklearn, matplotlip, seaborn, nltk, wordcloud and ELI5
  4. Dataset : https://github.com/nishantpokhriyal/Internship/blob/main/Malignant-Comments-Classifier-Project/malignant_test_pred.rar

Model/s Development and Evaluation

- Testing of Identified Approaches (Algorithms)

Listing down all the algorithms used for the training and testing.

```
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import cross_val_score,GridSearchCV
from sklearn.tree import DecisionTreeClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.ensemble import RandomForestClassifier,AdaBoostClassifier
```

- Run and Evaluate selected models

```
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score, confusion_matrix, classification_report,roc_curve,roc_auc_score,auc,f1_score
```

Logistic Regression : Logistic regression analysis is used to predict whether comment is bad or not based on the comment_text variable. It is only used to make prediction about categorical variable.

```
# LogisticRegression
LG = LogisticRegression(C=1, max_iter = 3000)

LG.fit(x_train, y_train)

y_pred_train = LG.predict(x_train)
print('Training accuracy is {}'.format(accuracy_score(y_train, y_pred_train)))
y_pred_test = LG.predict(x_test)
print('Test accuracy is {}'.format(accuracy_score(y_test,y_pred_test)))
print(confusion_matrix(y_test,y_pred_test))
print(classification_report(y_test,y_pred_test))
```

```
Training accuracy is 0.9595520103134316
Test accuracy is 0.9553392379679144
[[42729   221]
 [ 1917  3005]]
              precision    recall  f1-score   support

           0       0.96      0.99      0.98     42950
           1       0.93      0.61      0.74      4922

    accuracy                           0.96     47872
   macro avg       0.94      0.80      0.86     47872
weighted avg       0.95      0.96      0.95     47872
```

Decision Tree Classifier : DTC is used because of its ability to using different feature subsets and decision rules at different stages of classification.

```python
# DecisionTreeClassifier
DT = DecisionTreeClassifier()

DT.fit(x_train, y_train)
y_pred_train = DT.predict(x_train)
print('Training accuracy is {}'.format(accuracy_score(y_train, y_pred_train)))
y_pred_test = DT.predict(x_test)
print('Test accuracy is {}'.format(accuracy_score(y_test,y_pred_test)))
print(confusion_matrix(y_test,y_pred_test))
print(classification_report(y_test,y_pred_test))
```

```
Training accuracy is 0.9988898736783678
Test accuracy is 0.9399022393048129
[[41605  1345]
 [ 1532  3390]]
              precision    recall  f1-score   support

           0       0.96      0.97      0.97     42950
           1       0.72      0.69      0.70      4922

    accuracy                           0.94     47872
   macro avg       0.84      0.83      0.83     47872
weighted avg       0.94      0.94      0.94     47872
```

Random Forest Classifier : Random forest is used because It builds decision trees on different samples and takes their majority vote for classification.

```python
#RandomForestClassifier
RF = RandomForestClassifier()

RF.fit(x_train, y_train)
y_pred_train = RF.predict(x_train)
print('Training accuracy is {}'.format(accuracy_score(y_train, y_pred_train)))
y_pred_test = RF.predict(x_test)
print('Test accuracy is {}'.format(accuracy_score(y_test,y_pred_test)))
print(confusion_matrix(y_test,y_pred_test))
print(classification_report(y_test,y_pred_test))
```

```
Training accuracy is 0.9988898736783678
Test accuracy is 0.9550885695187166
[[42394   556]
 [ 1594  3328]]
              precision    recall  f1-score   support

           0       0.96      0.99      0.98     42950
           1       0.86      0.68      0.76      4922

    accuracy                           0.96     47872
   macro avg       0.91      0.83      0.87     47872
weighted avg       0.95      0.96      0.95     47872
```

Ada Boost Classifier : Ada Boost is used because it helps in combining multiple "weak classifiers" into a single "strong classifier".

```
#AdaBoostClassifier
ada=AdaBoostClassifier(n_estimators=100)
ada.fit(x_train, y_train)
y_pred_train = ada.predict(x_train)
print('Training accuracy is {}'.format(accuracy_score(y_train, y_pred_train)))
y_pred_test = ada.predict(x_test)
print('Test accuracy is {}'.format(accuracy_score(y_test,y_pred_test)))
print(confusion_matrix(y_test,y_pred_test))
print(classification_report(y_test,y_pred_test))
```

```
Training accuracy is 0.951118631321677
Test accuracy is 0.9490307486631016
[[42553   397]
 [ 2043  2879]]
              precision    recall  f1-score   support

           0       0.95      0.99      0.97     42950
           1       0.88      0.58      0.70      4922

    accuracy                           0.95     47872
   macro avg       0.92      0.79      0.84     47872
weighted avg       0.95      0.95      0.94     47872
```

KNeighbors Classifier : It classifies the data point on how its neighbor is classified.

```
]: #KNeighborsClassifier
knn=KNeighborsClassifier(n_neighbors=9)
knn.fit(x_train, y_train)
y_pred_train = knn.predict(x_train)
print('Training accuracy is {}'.format(accuracy_score(y_train, y_pred_train)))
y_pred_test = knn.predict(x_test)
print('Test accuracy is {}'.format(accuracy_score(y_test,y_pred_test)))
print(confusion_matrix(y_test,y_pred_test))
print(classification_report(y_test,y_pred_test))
```

```
Training accuracy is 0.922300110117369
Test accuracy is 0.9173629679144385
[[42809   141]
 [ 3815  1107]]
              precision    recall  f1-score   support

           0       0.92      1.00      0.96     42950
           1       0.89      0.22      0.36      4922

    accuracy                           0.92     47872
   macro avg       0.90      0.61      0.66     47872
weighted avg       0.91      0.92      0.89     47872
```

- Key Metrics for success in solving problem under consideration

  Accuracy Score :

  Accuracy score is used to measure the model performance in terms of measuring the ratio of sum of true positive and true negatives out of all the predictions made.

  Confusion Matrix :

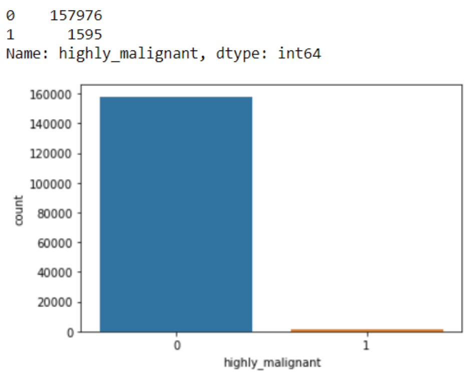
  A confusion matrix is a table that is used to define the performance of a classification algorithm. A confusion matrix visualizes and summarizes the performance of a classification algorithm.

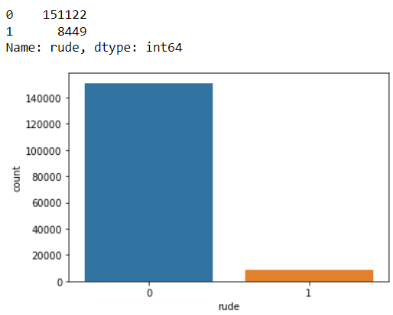
  Classification Report :

  A Classification report is used to measure the quality of predictions from a classification algorithm. How many predictions are True and how many are False.

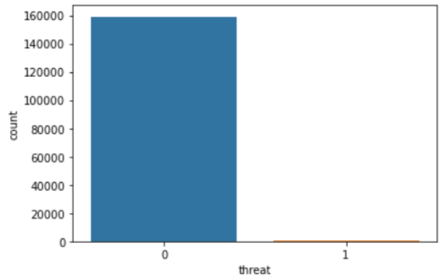
- Visualizations

  Count Plot : Used to count yes and no in different variables.

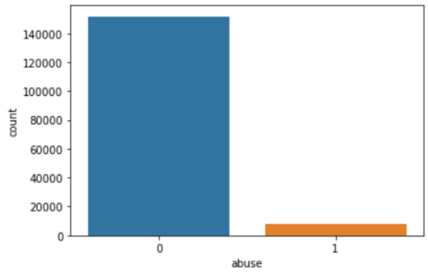




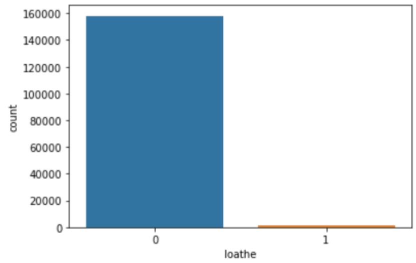
```
0    159093
1       478
Name: threat, dtype: int64
```



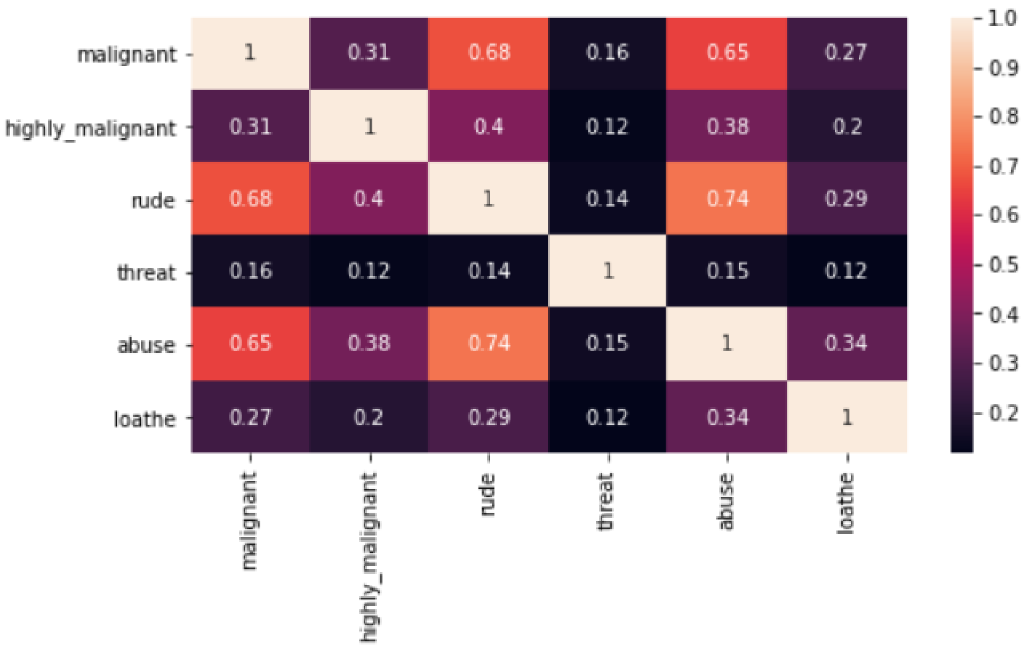

```
0    151694
1      7877
Name: abuse, dtype: int64
```




```
0    158166
1      1405
Name: loathe, dtype: int64
```

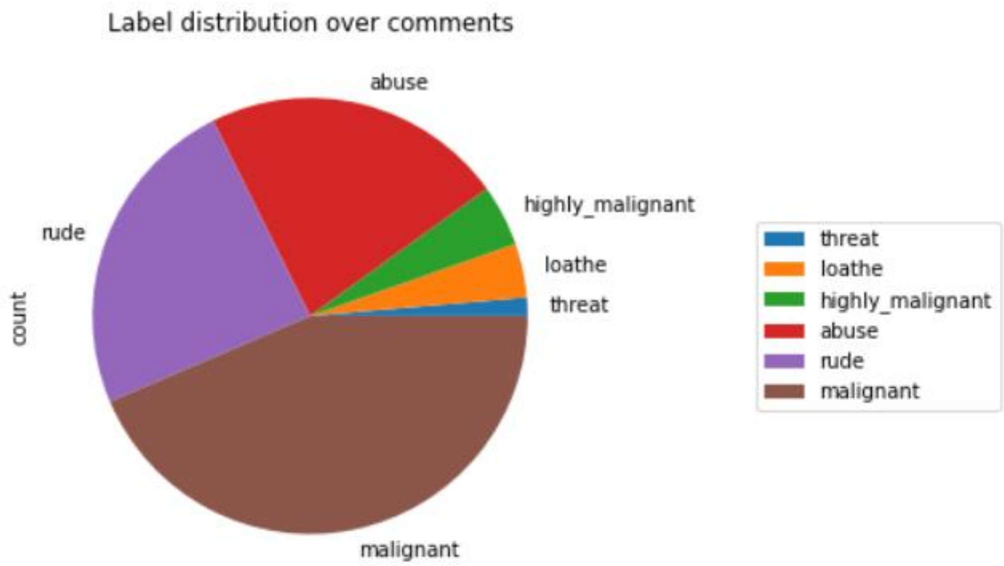


Heatmap : It is used to see the correlation between different variables.



Some of the variables like malignant and rude, malignant and abuse, rude and abuse are highly correlated and Threat is not much correlated with any of the variables.

Pie Plot : Used to see the which how many comments are malignant, highly malignant, threat, rude, abuse and loathe.



- Interpretation of the Results

  On the basis of my analysis I found that most of the comments are good and only few comments are bad.

```
0    143346
1     16225
```

0 represents good comments and 1 represents bad comments.

```
Test accuracy is 0.9553392379679144
```

Our model has more than 95% of accuracy.

We have used our model on test data to predict the bad comments.

Below is the result :

	id	comment_text	prediction
0	00001cee341fdb12	Yo bitch Ja Rule is more succesful then you'll...	0
1	0000247867823ef7	== From RfC == \n\n The title is fine as it is...	0
2	00013b17ad220c46	" \n\n == Sources == \n\n * Zawe Ashton on Lap...	0
3	00017563c3f7919a	:If you have a look back at the source, the in...	0
4	00017695ad8997eb	I don't anonymously edit articles at all.	0
...	...	...	...
153159	fffcd0960ee309b5	. \n i totally agree, this stuff is nothing bu...	0
153160	fffd7a9a6eb32c16	== Throw from out field to home plate. == \n\n...	0
153161	fffda9e8d6fafa9e	" \n\n == Okinotorishima categories == \n\n I ...	0
153162	fffe8f1340a79fc2	" \n\n == ""One of the founding nations of the...	0
153163	ffffce3fb183ee80	" \n :::Stop already. Your bullshit is not wel...	0

153164 rows × 3 columns

CONCLUSION

- Key Findings and Conclusions of the Study

  I have found that people are doing more number of malignant comments followed by rude and abuse, people do very less loathe and threat comments.

- Learning Outcomes of the Study in respect of Data Science

  I've got to know so much new things during this project like how to use the wordcloud and nltk package. I was able to understand this by the help of visualization libraries like matplotlib and seaborn by creating plots of different variables. I have created different types of classification models like Logistic Regression, Decision Tree Classifier, Random Booster classifier, Ada Booster Classifier and KNeighbors Classfier to predict the prices by training the data and the best performing model was Logistic Regression with 95.5 % of accuracy score.