FLIP ROBO

Malignant Comments Classifier

Submitted by:

Kaushik Veer

ACKNOWLEDGMENT

All thanks to flip robo technologies for providing me the opportunity to work on this project. I leaned a lot from this 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. Although researchers have found that hate is a problem across multiple platforms, there is a lack of models for online hate detection.

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.

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.

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. The problem we sought to solve was the tagging of internet comments that are aggressive towards other users. This means that insults to third parties such as celebrities will be tagged as unoffensive, but “u are an idiot” is clearly offensive.

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

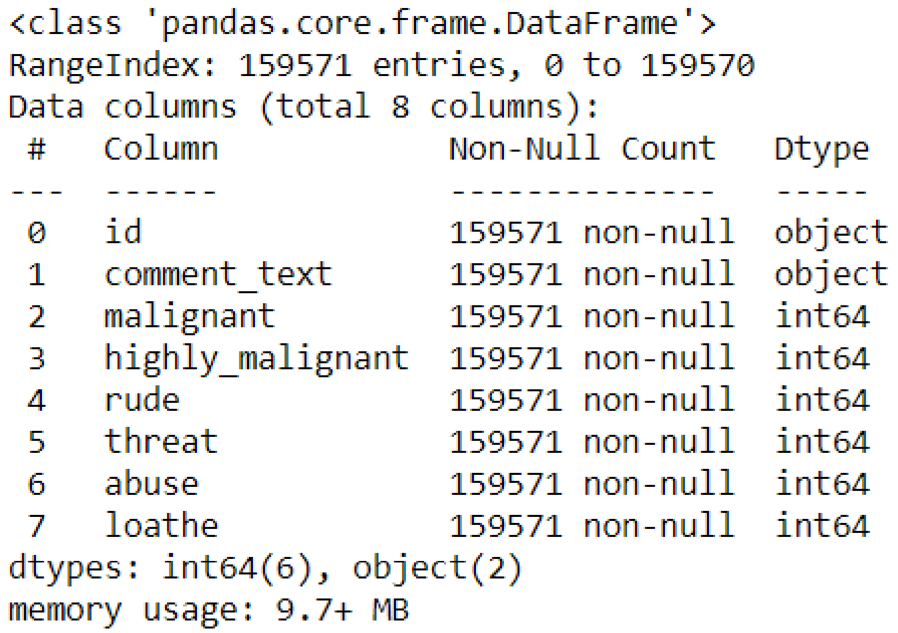
While we were working on this project we need to go through many sources, books and references. We are sharing some of the content which is relevant and useful in the lieu of this particular project.

- ## Motivation for the Problem Undertaken

As we can see this is a highly motivated project. As this is the real time problem of malignant comments which is getting bigger, a single malignant comment can ruin many things. This can take a toll on anyone and affect them mentally leading to depression, mental illness, self-hatred and suicidal thoughts.

Analytical Problem Framing

- ## Mathematical/ Analytical Modelling of the Problem



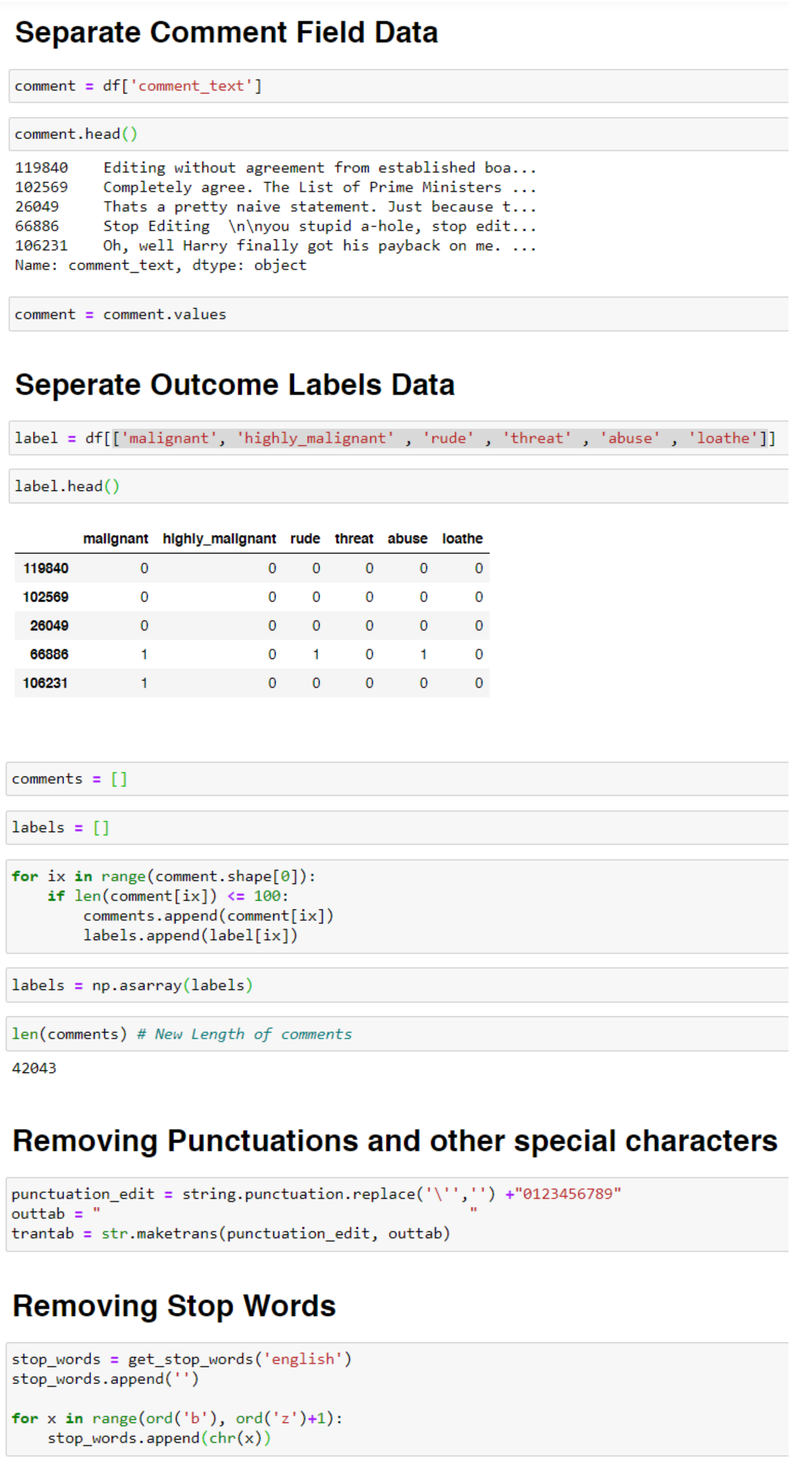
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 159571 entries, 0 to 159570
Data columns (total 8 columns):
 #   Column            Non-Null Count   Dtype 
---  ------            --------------   ----- 
 0   id                159571 non-null  object
 1   comment_text      159571 non-null  object
 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
```

- ## Data Sources and their formats

There are 8 columns in the dataset provided to you. The description of each of the column is given below:

- **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.

- Data Pre-processing Done



Separate Comment Field Data

```
comment = df['comment_text']
```

```
comment.head()
```

```
119840    Editing without agreement from established boa...
102569    Completely agree. The List of Prime Ministers ...
26049     Thats a pretty naive statement. Just because t...
66886     Stop Editing  \n\nyou stupid a-hole, stop edit...
106231    Oh, well Harry finally got his payback on me. ...
Name: comment_text, dtype: object
```

```
comment = comment.values
```

Seperate Outcome Labels Data

```
label = df[['malignant', 'highly_malignant' , 'rude' , 'threat' , 'abuse' , 'loathe']]
```

```
label.head()
```

	malignant	highly_malignant	rude	threat	abuse	loathe
119840	0	0	0	0	0	0
102569	0	0	0	0	0	0
26049	0	0	0	0	0	0
66886	1	0	1	0	1	0
106231	1	0	0	0	0	0

```
comments = []
```

```
labels = []
```

```
for ix in range(comment.shape[0]):
    if len(comment[ix]) <= 100:
        comments.append(comment[ix])
        labels.append(label[ix])
```

```
labels = np.asarray(labels)
```

```
len(comments) # New Length of comments
```

```
42043
```

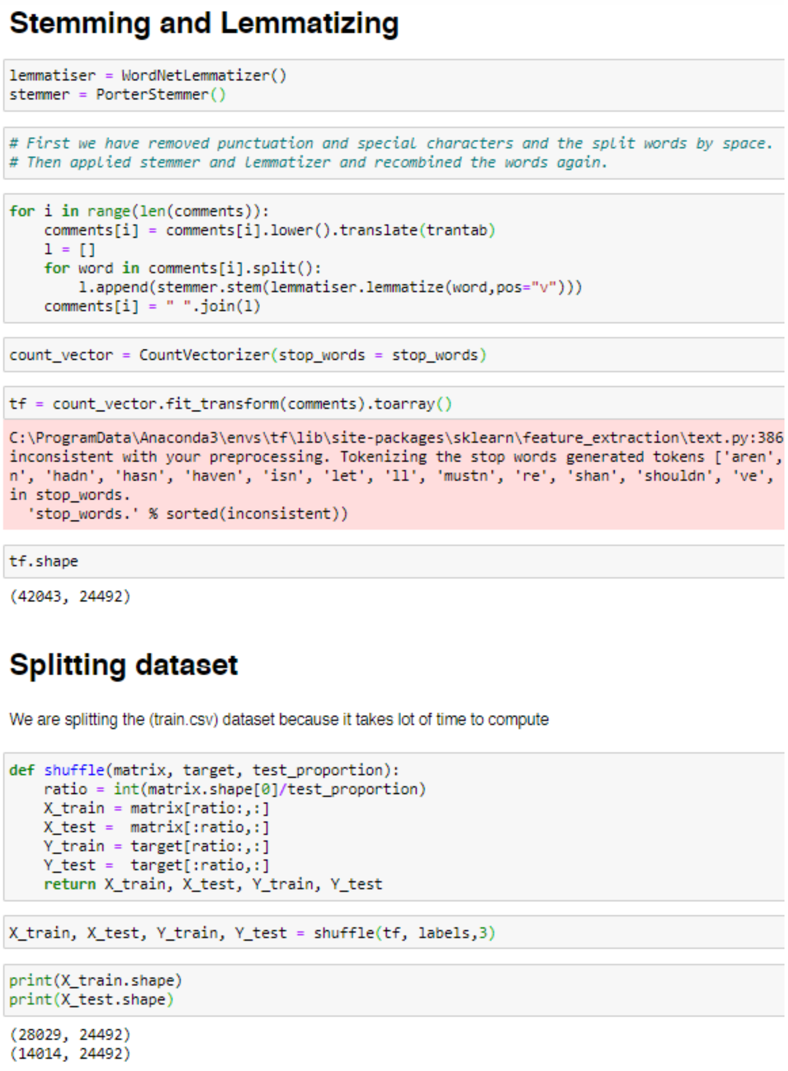
Removing Punctuations and other special characters

```
punctuation_edit = string.punctuation.replace('\'','') +"0123456789"
outtab = "                                 "
trantab = str.maketrans(punctuation_edit, outtab)
```

Removing Stop Words

```
stop_words = get_stop_words('english')
stop_words.append('')

for x in range(ord('b'), ord('z')+1):
    stop_words.append(chr(x))
```



Stemming and Lemmatizing

```
lemmatiser = WordNetLemmatizer()
stemmer = PorterStemmer()
```

```
# First we have removed punctuation and special characters and the split words by space.
# Then applied stemmer and lemmatizer and recombined the words again.
```

```
for i in range(len(comments)):
    comments[i] = comments[i].lower().translate(trantab)
    l = []
    for word in comments[i].split():
        l.append(stemmer.stem(lemmatiser.lemmatize(word,pos="v")))
    comments[i] = " ".join(l)
```

```
count_vector = CountVectorizer(stop_words = stop_words)
```

```
tf = count_vector.fit_transform(comments).toarray()
```

```
C:\ProgramData\Anaconda3\envs\tf\lib\site-packages\sklearn\feature_extraction\text.py:386
inconsistent with your preprocessing. Tokenizing the stop words generated tokens ['aren',
n', 'hadn', 'hasn', 'haven', 'isn', 'let', 'll', 'mustn', 're', 'shan', 'shouldn', 've',
in stop_words.
  'stop_words.' % sorted(inconsistent))
```

```
tf.shape
```

```
(42043, 24492)
```

Splitting dataset

We are splitting the (train.csv) dataset because it takes lot of time to compute

```
def shuffle(matrix, target, test_proportion):
    ratio = int(matrix.shape[0]/test_proportion)
    X_train = matrix[ratio:,:]
    X_test =  matrix[:ratio,:]
    Y_train = target[ratio:,:]
    Y_test =  target[:ratio,:]
    return X_train, X_test, Y_train, Y_test
```

```
X_train, X_test, Y_train, Y_test = shuffle(tf, labels,3)
```

```
print(X_train.shape)
print(X_test.shape)
```

```
(28029, 24492)
(14014, 24492)
```

- Hardware and Software Requirements and Tools Used

Hardware specifications are :-

Ryzen 5

16gb ram

RTX 2070 super graphics

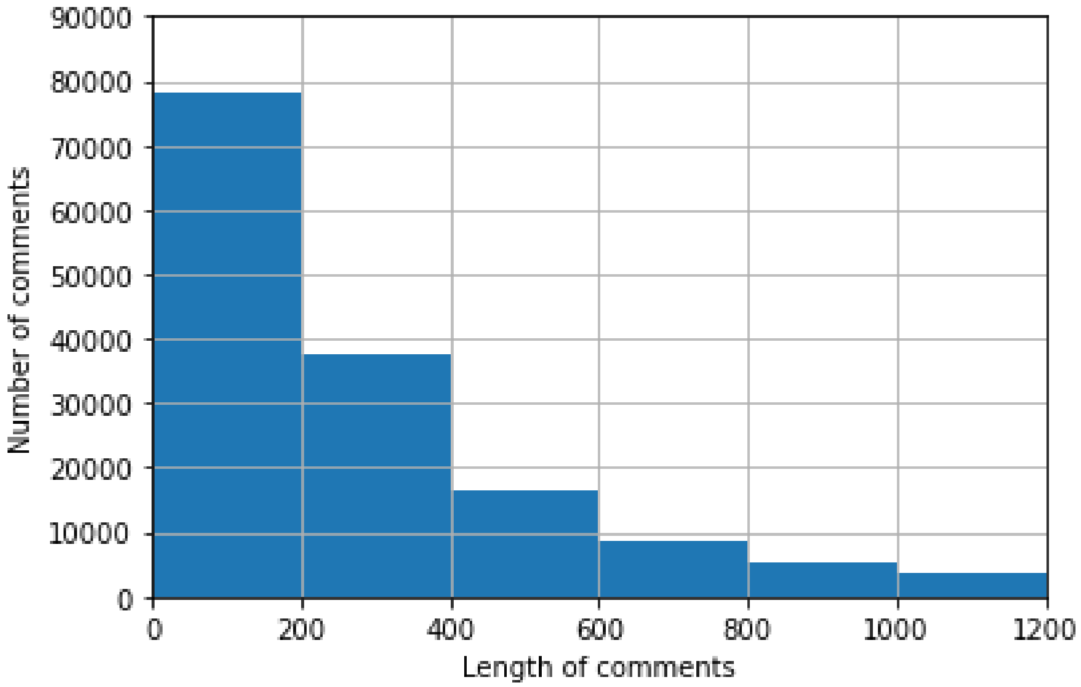
Software used :-

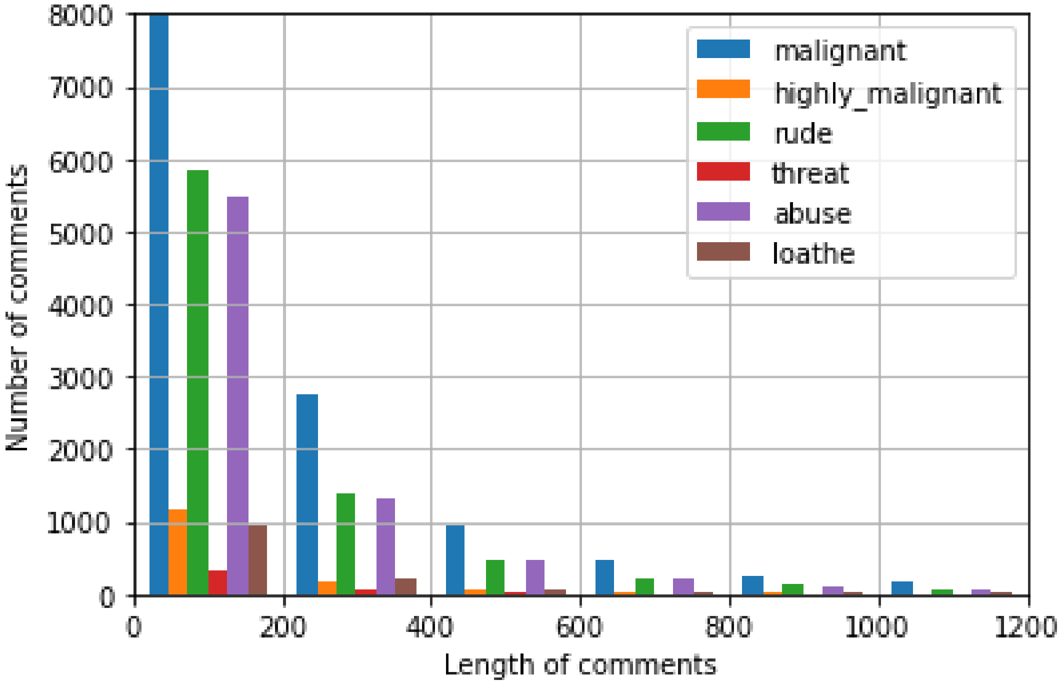
Operating system : windows 10

Jupyter notebook and anaconda navigator- for coding and using data analytics tools and libraries.

Model/s Development and Evaluation

- Identification of possible problem-solving approaches (methods)





- Testing of Identified Approaches (Algorithms)

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

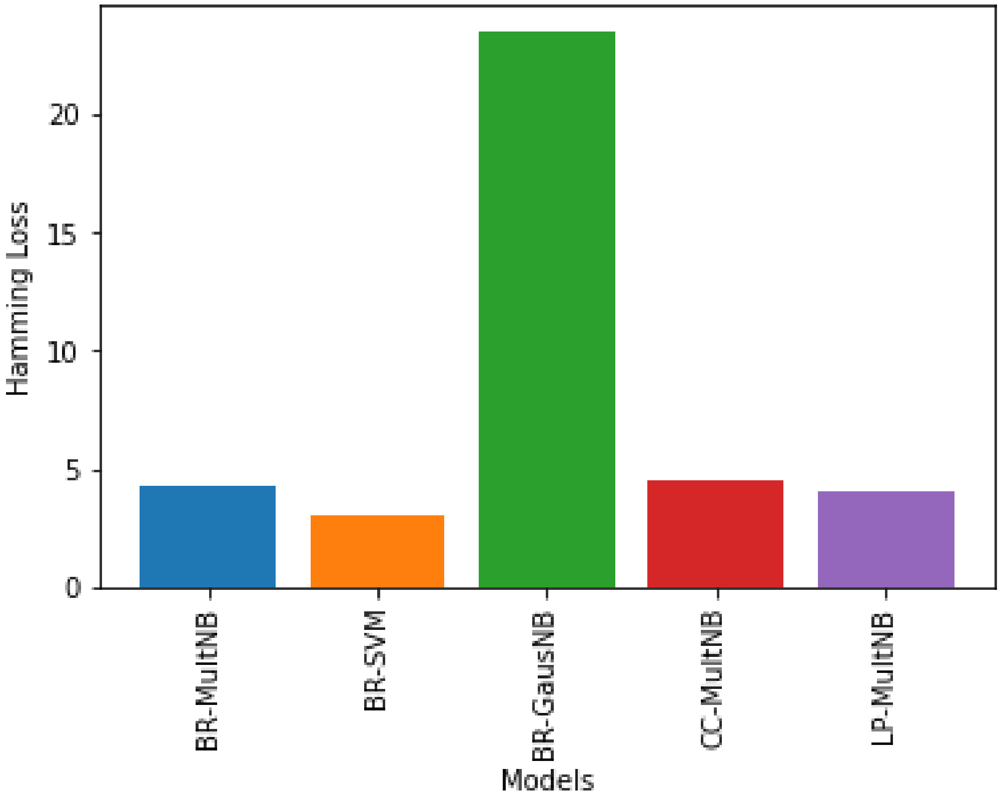
- Run and Evaluate selected models

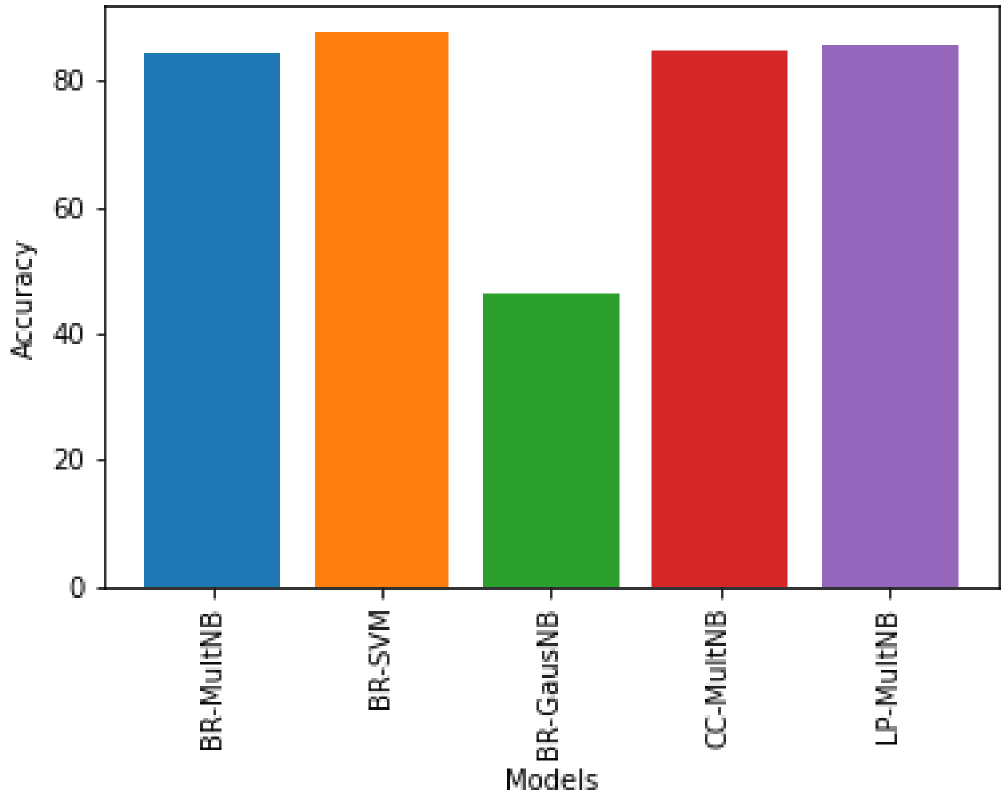
  Describe all the algorithms used along with the snapshot of their code and what were the results observed over different evaluation metrics.

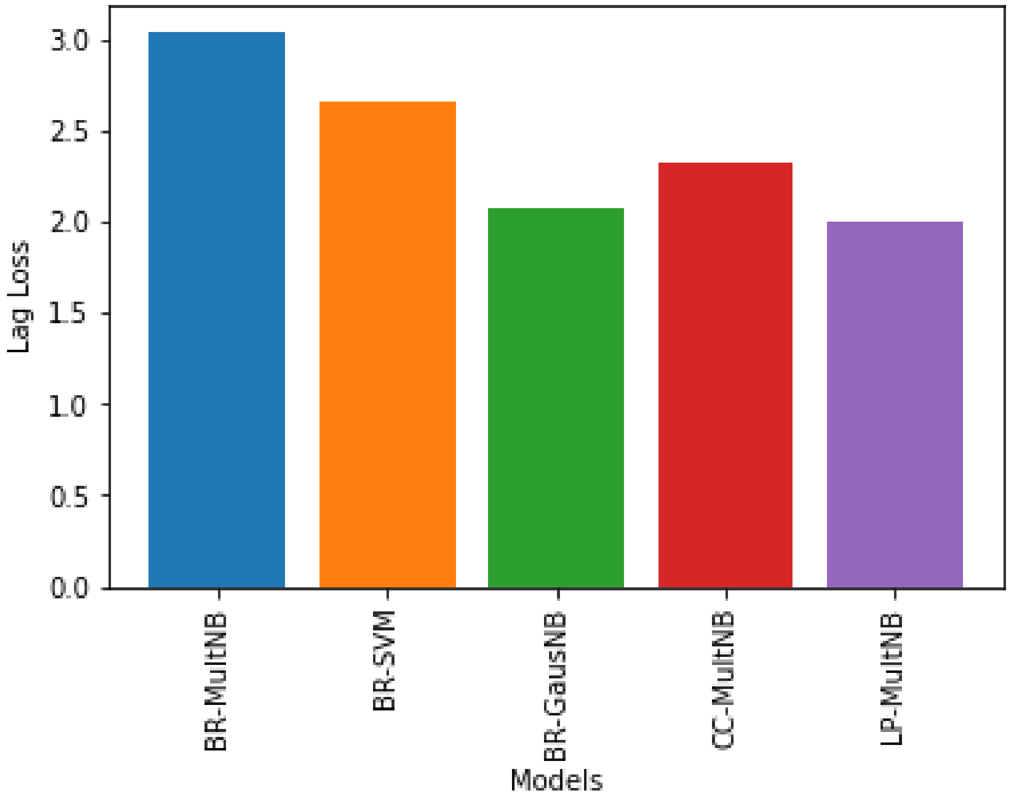
- Key Metrics for success in solving problem under consideration

  What were the key metrics used along with justification for using it? You may also include statistical metrics used if any.

- Visualizations




Accuracy
0
20
40
60
80
BR-MultNB
BR-SVM
BR-GausNB
CC-MultNB
LP-MultNB
Models


Lag Loss
0.0
0.5
1.0
1.5
2.0
2.5
3.0
BR-MultNB
BR-SVM
BR-GausNB
CC-MultNB
LP-MultNB
Models

- Interpretation of the Results

The best models work on this dataset is Label Powerset with MultinomialNB which has a Hamming Loss of 4.0, Accuracy of 85.47, Lag Loss of 2.0 and Binary Relevance with SVM which has a Hamming Loss of 3.02, Accuracy of 87.57 and Lag Loss of 2.66

CONCLUSION

- Key Findings and Conclusions of the Study

  In the above dataset every features of the dataset plays an important role to understand the data also in visualization and applying models and algorithms.

- Learning Outcomes of the Study in respect of Data Science

  Visualization is very helpful and it plays a vital role to understand the data into graphical form. So, that we can understand what the data is trying to say.

  Data cleaning is also import part it helps me to clear the comments by removing stop words, punctuations, special characters, stemming and lemmatizing.

  I used five types of algorithm like Binary Relevance with MultinomialNB, Binary Relevance with SVM classifier, Binary Relevance with GausseanNB, Classifier chain with MultinomialNB and Label Powerset with MultinomialNB.

  The problem I faced while on this dataset is it takes more than an hour for output. I overcome this situation with the help of towardsdatascience.com by running the code using small chunk of dataset to complete the project.

- <u>Limitations of this work and Scope for Future Work</u>
- We can see the data was not in its finest form with some more analysis we can still improve the accuracy of the model.

- The goal was to achieve 100% accuracy in predicting the house prices but we ended up with the accuracy of 85%.

- Due to lack of experience I was not able to reach my goal but with working on more such projects and gaining more experience will help me to grow and develop more valuable skills to reach my goal in future.