FLIP ROBO

PROJECT REPORT

ON

MALIGNANT COMMENTS CLASSIFIER

Submitted by

MANISHA

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. 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.

Conceptual Background of the Domain Problem:

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.

Review of Literature:

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.

Motivation for the Problem Undertaken:

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. So, we fit the model and predict the test data.

Analytical Problem Framing

Mathematical/ Analytical Modeling of the Problem:

We use Statistical techniques and analytics modeling in our projects, such as:

- describe() : use to calculate the statistical values that are mean, standard deviation, quantile deviation, minimum and maximum values.
- corr(): use to calculate the relation between feature variable with the target variable.
- skew(): use to check whether the skewness is present in the continuous data or not.

Data Source and their formats:

The data set of the Malignant Comments Classifier as show in the fig:

```
train_data=pd.read_csv('C:\\Users\\Dell\\Desktop\\Manisha\\Project6\\train.csv')
train_data.head()
```

	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

There are 159571 rows and 8 columns. 'malignant' column is our target variable and others are feature variable. There are 6 columns of numerical data and 2 columns of object type.

Data Pre-processing

There are no null values in the dataset. Apply feature engineering technique on comment_text. Convert object data into numerical data. There are more object data so we do not consider Standard Scaler() in the dataset.

Data inputs-Logic-Output Relationships

There are 3 columns having higher relation (greater than 50%) and left of all are lower relation in independent and another independent variable. Malignant column is the target variable.

Hardware and Software Requirements and Tools used

Hardware:

- Memory 16GB minimum
- Hard Drive SSD is preferred 500GB
- Processor intel i5 minimum
- Operating system Windows 10

Software:

- Jupyter notebook (Python)

Libraries:

pandas (used to create the data and read the data)

numpy (used with the mathematical function)

seaborn (used to create a different types of graphs)

matplotlib (used to plot the graph)

accuracy_score (used to calculate accuracy score for train and test)

classification_report (to display precision, f1 score)

confusion_matrix (form the matrix)

Model/s Development and Evaluation

Identification of possible problem:

We approach to both statistical and analytical problem

- ❖ Plot a bar graph for nominal data and distribution graph for continuous data
- ❖ describe () use to calculate mean, standard deviation, minimum, maximum and quantile deviation
- ❖ corr() used to calculate the correlation of input variable with the output variable.
- ❖ skew() used to calculate the skewness of the data

Testing of Identified Approaches:

Here we work on the classification problem so the machine learning models are:

- Logistic Regression
- K Neighbors Classification
- Random Forest Classification
- Decision Tree Classification

Run and Evaluate selected models:

➢ Logistic Regression

```
x_train,x_test,y_train,y_test = train_test_split(x,y, test_size=0.30,random_state=43)
```

```
lr.fit(x_train,y_train)
y_pred1 = lr.predict(x_test)
accuracy = accuracy_score(y_test,y_pred1)*100
print("accuracy score:",accuracy)
```

```
accuracy score: 89.82912767379679
```

```
cm= confusion_matrix(y_test,y_pred1)
print(cm)
```

```
[[43003     0]
 [ 4869     0]]
```

```
clr=classification_report(y_test,y_pred1)
print(clr)
```

```
              precision    recall  f1-score   support

           0       0.90      1.00      0.95     43003
           1       0.00      0.00      0.00      4869

    accuracy                           0.90     47872
   macro avg       0.45      0.50      0.47     47872
weighted avg       0.81      0.90      0.85     47872
```

The logistic regression of precision is 90, recall is 100 and f1-score is 95.

➢ K Neighbors Classification

```
x_train,x_test,y_train,y_test = train_test_split(x,y,test_size=0.30,random_state=19)
```

```
knn.fit(x_train,y_train)
y_pred2 = knn.predict(x_test)
accuracy = accuracy_score(y_test,y_pred2)*100
print("accuracy score:",accuracy)
```

```
accuracy score: 89.29436831550802
```

```
cm= confusion_matrix(y_test,y_pred2)
print(cm)
```

```
[[42398   684]
 [ 4441   349]]
```

```
clr=classification_report(y_test,y_pred2)
print(clr)
```

```
              precision    recall  f1-score   support

           0       0.91      0.98      0.94     43082
           1       0.34      0.07      0.12      4790

    accuracy                           0.89     47872
   macro avg       0.62      0.53      0.53     47872
weighted avg       0.85      0.89      0.86     47872
```

The K Neighbors classification of precision is 91, recall is 98 and f1-score is 94.

- Random Forest Classification

```
x_train,x_test,y_train,y_test = train_test_split(x,y, test_size=0.30,random_state=84)
```

```
rfc.fit(x_train,y_train)
y_pred4 = rfc.predict(x_test)
accuracy = accuracy_score(y_test,y_pred4)*100
print("accuracy score:",accuracy)
```

```
accuracy score: 90.2615307486631
```

```
cm= confusion_matrix(y_test,y_pred4)
print(cm)
```

```
[[42787   301]
 [ 4361   423]]
```

```
clr=classification_report(y_test,y_pred4)
print(clr)
```

```
              precision    recall  f1-score   support

           0       0.91      0.99      0.95     43088
           1       0.58      0.09      0.15      4784

    accuracy                           0.90     47872
   macro avg       0.75      0.54      0.55     47872
weighted avg       0.88      0.90      0.87     47872
```

The Random Forest classification of precision is 95, recall is 94 and f1-score is 94.

- Decision Tree Classification

```
x_train,x_test,y_train,y_test = train_test_split(x,y,test_size=0.30,random_state=35)
```

```
clf.fit(x_train,y_train)
y_pred3 = clf.predict(x_test)
accuracy = accuracy_score(y_test,y_pred3)*100
print("accuracy score:",accuracy)
```

```
accuracy score: 84.70295788770053
```

```
cm= confusion_matrix(y_test,y_pred3)
print(cm)
```

```
[[39275  3705]
 [ 3618  1274]]
```

```
clr=classification_report(y_test,y_pred3)
print(clr)
```

```
              precision    recall  f1-score   support

           0       0.92      0.91      0.91     42980
           1       0.26      0.26      0.26      4892

    accuracy                           0.85     47872
   macro avg       0.59      0.59      0.59     47872
weighted avg       0.85      0.85      0.85     47872
```

The Decision Tree classification of precision is 89, recall is 90 and f1-score is 89.

The Random forest classification gives better precision score, recall and f1-score. The total of True Negative and False Negative in the confusion matrix is also less in the same model.

Visualisation:

On visualising the continuous data we see the data is right skewed and the target variable is imbalance because zero is in a large number and one is rare in number. Id and Comment_text are object type.

Interpretation of the Results:

On our analysis basis we go through various models and then we conclude better model on the basis of precision and f1-score. That will predict the test data on training the train data.

Conclusion

Key Findings and Conclusions of the Study:

On study we see two data set one is train and another one is test but for test we have to predict the target values. The zero values are very high in numbers and value one have less in number of all the malignant, highly_malignant, rude, threat, abuse and loathe columns.

Learning Outcomes of the Study in respect of Data Science:

Here we use feature engineering technique on date and drop unused column from dataset. Visualise the continuous data and here we see the data is skewed. The target variable ‘Result’ is imbalanced. In analysis we do describe the statistical values, correlation, outliers and skewness. Fit some classification models and find the better one i.e. Random Forest Model. Calculate confusion metrics and classification report.