Reviews Rating Prediction

Submitted By:

Aman Saxena

ACKNOWLEDGMENT

- https://scikit-learn.org/stable/ - For the libraries used in the project.
- Rest project is done by myself only.

INTRODUCTION

- Business Problem Framing

We have a client who has a website where people write different reviews for technical products. Now they are adding a new feature to their website i.e. The reviewer will have to add stars(rating) as well with the review. The rating is out 5 stars and it only has 5 options available 1 star, 2 stars, 3 stars, 4 stars, 5 stars. Now they want to predict ratings for the reviews which were written in the past and they don't have a rating. So, we have to build an application which can predict the rating by seeing the review. Our goal is to build a model which will predict the ratings of the reviews.

.

- Conceptual Background of the Domain Problem

For more understanding we can simply correlate it with the project of to find the type of the comment.

- Motivation for the Problem Undertaken

In this project we have to build the model that will predict the Rating of the reviews share by the users.

Analytical Problem Framing

- Mathematical/ Analytical Modeling of the Problem

First of all we scrape the reviews and ratings of different electronic items from different websites and store them in the excel file.

After that data preprocessing is done in which we cleaned the comments in different ways i.e. convert all messages to lower case, Replace emailaddresses with 'emailaddress', Replace Urls with 'webaddress', Replace money symbols with 'moneysymbols', Replace 10 digit phone numbers, Replace numbers with 'number', Remove Punctuatuations, Remove White space between terms with single space, Remove leading and trailing white space, Remove Stopwords . After that Lematization is done . And then the new column is added ‘clean length’ after the punctuations and stopwords were removed.

Then we converted the text into the vectors using TF-IDF. Then we divide the dataset into dependent and independent variables. After that dataset is trained with different models and prediction is made and found that the Random forest classifier is giving the best accuracy.

. **Data Preprocessing Done**

we cleaned the comments in different ways i.e. convert all messages to lower case, Replace emailaddresses with 'emailaddress', Replace Urls with 'webaddress', Replace money symbols with 'moneysymbols', Replace 10 digit phone numbers,

Replace numbers with 'number', Remove Punctuatuations, Remove White space between terms with single space, Remove leading and trailing white space, Remove Stopwords . After that Lematization is done . And then the new column is added ‘clean length’ after the punctuations and stopwords were removed.

Then we converted the text into the vectors using TF-IDF. Then we divide the dataset into dependent and independent variables. After that dataset is trained with different models and prediction is made and found that the Random forest classifier is giving the best accuracy.

- **Hardware and Software Requirements and Tools Used**

- import numpy as np
- import pandas as pd
- import seaborn as sns
- import matplotlib.pyplot as plt
- import warnings
- warnings.filterwarnings('ignore')
- from nltk.stem import WordNetLemmatizer
- import nltk
- from nltk.corpus import stopwords
- import string

Model/s Development and Evaluation

- Identification of possible problem-solving approaches (methods)

Since the problem is of NLP and we have to predict the Rating of the Reviews

Testing of Identified Approaches (Algorithms)

- ➢ Logistic Regression
- ➢ MultinomialNB
- ➢ Random forest Classifier
- ➢ Decision Tree Classifier

- Run and Evaluate selected models
  - ➢ Logistic Regression

Logistic Regression

```
[1]: LR=LogisticRegression()
     LR.fit(x_train,y_train)
     pred=LR.predict(x_test)
     accuracy=accuracy_score(y_test,pred)
     print(accuracy*100)
     print(confusion_matrix(y_test,pred))
     print(classification_report(y_test,pred))
```

```
78.58694078471117
[[ 578    0    4   14  152]
 [  50   38    4   13   42]
 [  28    0  102   50  239]
 [  17    0    5  392  745]
 [  15    0   11   90 4318]]
              precision    recall  f1-score   support

           1       0.84      0.77      0.81       748
           2       1.00      0.26      0.41       147
           3       0.81      0.24      0.37       419
           4       0.70      0.34      0.46      1159
           5       0.79      0.97      0.87      4434

    accuracy                           0.79      6907
   macro avg       0.83      0.52      0.58      6907
weighted avg       0.78      0.79      0.75      6907
```

- MultinomialNB

MultinomialNB

```
naive=MultinomialNB()
naive.fit(x_train,y_train)
y_pred=naive.predict(x_test)
accu=accuracy_score(y_test,y_pred)
print(accu)
```

```
0.7211524540321413
```

- Random Forest Classifier

Random Forest Classifier

```
from sklearn.ensemble import RandomForestClassifier
RF=RandomForestClassifier()
RF.fit(x_train,y_train)
pred=RF.predict(x_test)
print('Accuracy ',accuracy_score(y_test,pred)*100)
print(confusion_matrix(y_test,pred))
print(classification_report(y_test,pred))
```

```
Accuracy  87.05660923700593
[[ 616    2    2    0  128]
 [  35   78    2    1   31]
 [  18    2  225   15  159]
 [  10    1    5  706  437]
 [   9    0    5   32 4388]]
              precision    recall  f1-score   support

           1       0.90      0.82      0.86       748
           2       0.94      0.53      0.68       147
           3       0.94      0.54      0.68       419
           4       0.94      0.61      0.74      1159
           5       0.85      0.99      0.92      4434

    accuracy                           0.87      6907
   macro avg       0.91      0.70      0.77      6907
weighted avg       0.88      0.87      0.86      6907
```

➢ Decision Tree Classifier

Decision Tree Classifier

```
1 from sklearn.tree import DecisionTreeClassifier
2 DT=DecisionTreeClassifier()
3 DT.fit(x_train,y_train)
4 pred=DT.predict(x_test)
5 print('Accuracy ',accuracy_score(y_test,pred)*100)
6 print(confusion_matrix(y_test,pred))
7 print(classification_report(y_test,pred))
```

```
Accuracy  84.50846966845229
[[ 611   13   17   16   91]
 [  36   80    7    4   20]
 [  26    4  236   30  123]
 [  19    3   17  783  337]
 [  49   18   36  204 4127]]
              precision    recall  f1-score   support

           1       0.82      0.82      0.82       748
           2       0.68      0.54      0.60       147
           3       0.75      0.56      0.64       419
           4       0.76      0.68      0.71      1159
           5       0.88      0.93      0.90      4434

    accuracy                           0.85      6907
   macro avg       0.78      0.71      0.74      6907
weighted avg       0.84      0.85      0.84      6907
```

- Key Metrics for success in solving problem under consideration

accuracy_score → for calculating the accuracy

CONCLUSION

First of all we scrape the reviews and ratings of different electronic items from different websites and store them in the excel file.

After that data preprocessing is done in which we cleaned the comments in different ways i.e. convert all messages to lower case, Replace emailaddresses with 'emailaddress', Replace Urls with 'webaddress', Replace money symbols with 'moneysymbols', Replace 10 digit phone numbers, Replace numbers with 'number', Remove Punctuatuations, Remove White space between terms with single space, Remove leading and trailing white space, Remove Stopwords . After that Lematization is done . And then the new column is added ‘clean length’ after the punctuations and stopwords were removed.

Then we converted the text into the vectors using TF-IDF. Then we divide the dataset into dependent and independent variables. After that dataset is trained with different models and prediction is made and found that the Random forest classifier is giving the best accuracy.