RATINGS PREDICTION

Submitted by:

SAYAN KUMAR BHUNIA

ACKNOWLEDGMENT

This includes mentioning of all the references, research papers, data sources, professionals and other resources that helped me and guided me in completion of the project.

- https://towardsdatascience.com/

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

➢ Motivation for the Problem Undertaken

I have to scrape data. More the data better the model In this section I need to scrape the reviews of different laptops, Phones, Headphones, smart watches, Professional Cameras, Printers, Monitors, Home theater, Router from different e commerce websites.

Analytical Problem Framing

➢ Mathematical/ Analytical Modelling of the Problem

We shall build a supervised Classification model to predict the price of a car based on given features.

Now, when we talk about building a supervised Classification catering to certain use-case, following three things come into our minds:

- **Data** appropriate to the business requirement or use-case we are trying to solve
- A **Classification model** which we think (or, rather assess) to be the best for our solution.
- **Optimize** the chosen model to ensure best performance.

➢ Data Sources and their formats

I have to scrape at least 20000 rows of data. You can scrape more data as well, it's up to you. more the data better the model

```
]: train_df=pd.read_csv('review_ratings_prediction.csv')
   train_df
```

]:

	Unnamed: 0	reviews.rating	reviews.text
0	0	5.0	This keyboard is very easy to type on, but the...
1	1	4.0	It's thin and light. I can type pretty easily ...
2	2	4.0	I love the new design the keys are spaced well...
3	3	5.0	Attached easily and firmly. Has a nice feel. A...
4	4	5.0	Our original keyboard was okay, but did not ha...
...	...	...	...
7294	7294	4.0	This remote was a decent replacement for my lo...
7295	7295	4.0	This looks good on my celling. Flushes very we...
7296	7296	4.0	Great Plate for projector mount, very easy to ...
7297	7297	5.0	This mount worked very well for me when I used...
7298	7298	4.0	It's a nice solid mounting plate check and mak...

7299 rows × 3 columns

➢ Data Pre-processing

Following steps have been performed on the data.

- **checking missing values**-

- If there is any missing value present in your data set then for a better and correct accuracy you have to impute it.
- If missing data present in object type column, then you have to take most frequent value for your missing data.
- If missing data present in int or float type column then use mean/median for missing value.

In the following case no missing value present:

```
[47]: train_df.isnull().sum()

t[47]: reviews.rating    164
       reviews.text        5
       dtype: int64
```

- **Stemming**- Stemming is a natural language processing technique that lowers inflection in words to their root forms, hence aiding in the preprocessing of text, words, and documents for text normalization.
  PorterStemmer() is a module in NLTK that implements the Porter Stemming technique.

```
In [167]: port_stem=PorterStemmer()

In [168]: def stemming(comment_text):
              stemmed_comment_text = re.sub('[^a-zA-z]',' ',comment_text)##will re
              stemmed_comment_text = stemmed_comment_text.lower() ##converting all
              stemmed_comment_text = stemmed_comment_text.split()
              #stemmed_comment_text = [port_stem(word) for word in stemmed_comment
              stemmed_comment_text = ' '.join(stemmed_comment_text)
              return stemmed_comment_text
```

- **TfidfVectorizer-**. Tf-idf stands for Term Frequency Inverse Document Frequency of records. It can be defined as the calculation of how relevant a word in a series or corpus is to a text. The meaning increases proportionally to the number of times in the text a word appears but is compensated by the word frequency in the corpus (data-set).

- **Feature scaling-** Feature Scaling ensures that all features will get equal importance in supervised classifier models. Standard scaler was used to scale all features in the data.

- **Reducing dimension of the data-** Sklearn's pca can be used to apply principal component analysis on the data. This helped in finding the vectors of maximal variance in the data.
  In this case iam not using it.

- **Outliers detection-** In simple words, an outlier is an observation that diverges from an overall pattern on a sample.

  In the following case no need to detect outliers.

➢ Software Requirements and library Used

```
: import pandas
  import pandas as pd
  import numpy as np
  import seaborn as sns
  import matplotlib.pyplot as plt
  import re
  from nltk.corpus import stopwords
  from nltk.stem.porter import PorterStemmer
  from sklearn.feature_extraction.text import TfidfVectorizer
  import warnings
  warnings.filterwarnings('ignore')
```

```
: !pip install nltk
```

```
Requirement already satisfied: nltk in c:\users\sayan\anaconda3\lib\site-packages (3.6.5)
Requirement already satisfied: click in c:\users\sayan\anaconda3\lib\site-packages (from nltk) (8.0.3)
Requirement already satisfied: joblib in c:\users\sayan\anaconda3\lib\site-packages (from nltk) (1.1.0)
Requirement already satisfied: regex>=2021.8.3 in c:\users\sayan\anaconda3\lib\site-packages (from nltk
Requirement already satisfied: tqdm in c:\users\sayan\anaconda3\lib\site-packages (from nltk) (4.62.3)
Requirement already satisfied: colorama in c:\users\sayan\anaconda3\lib\site-packages (from click->nltk
```

```
: import nltk
  nltk.download()
```

```
showing info https://raw.githubusercontent.com/nltk/nltk_data/gh-pages/index.xml
```

```
: True
```

- **NumPy**
  NumPy is a popular Python library for multi-dimensional array and matrix processing because it can be used to perform a great variety of mathematical operations. Its capability to handle linear algebra, Fourier transform, and more, makes NumPy ideal for machine learning and artificial intelligence (AI) projects, allowing users to manipulate the matrix to easily improve machine learning

performance. NumPy is faster and easier to use than most other Python libraries.

- **Scikit-learn**
  Scikit-learn is a very popular machine learning library that is built on NumPy and SciPy. It supports most of the classic supervised and unsupervised learning algorithms, and it can also be used for data mining, modelling, and analysis.

- **Seaborn**
  Seaborn is another open-source Python library, one that is based on Matplotlib (which focuses on plotting and data visualization) but features Pandas' data structures. Seaborn is often used in ML projects because it can generate plots of learning data. Of all the Python libraries, it produces the most aesthetically pleasing graphs and plots, making it an effective choice if you'll also use it for marketing and data analysis.

- **Pandas**
  Pandas is another Python library that is built on top of NumPy, responsible for preparing high-level data sets for machine learning and training. It relies on two types of data structures, one-dimensional (series) and two-dimensional (Data Frame). This allows Pandas to be applicable in a variety of industries including finance, engineering, and statistics. Unlike the slow-moving animals themselves, the Pandas library is quick, compliant, and flexible.

➢ Class imbalance problem

The first challenge we hit upon exploring the data, is class imbalanced problem. Imbalance data will lead to a bad accuracy of a model. To achieve better accuracy, we'll balance the data by using Smote Over Sampling or under sampling Method .But in this project data is balanced so I am not using it.

Model/s Development and Evaluation

➢ Run and evaluate selected models

Let's select our classification model for this project:

- Decission Tree Classifier
- RandomForestClassifier
- Logistic regression
- SVM
- KNN Classifier

➢ Testing of Identified Approaches (Algorithms)

```
|: import sklearn
   from sklearn.linear_model import LogisticRegression
   from sklearn.neighbors import KNeighborsClassifier
   from sklearn.ensemble import RandomForestClassifier
   from sklearn.svm import SVC
   from sklearn.tree import DecisionTreeClassifier
   from sklearn.metrics import accuracy_score,confusion_matrix,classification_report
   from sklearn.model_selection import train_test_split
```

```
|: x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=.30,stratify=y,random_state=20)
```

```
|: rf=RandomForestClassifier(n_estimators=100,random_state=50)
   svm=SVC()
   lg=LogisticRegression()
   dt=DecisionTreeClassifier()
   knn=KNeighborsClassifier()
   model=[rf,svm,lg,dt,knn]
   for m in model:
       m.fit(x_train,y_train)
       predm=m.predict(x_train)
       print('Training accuracy is {}'.format(accuracy_score(y_train,predm)))
       predm=m.predict(x_test)
       print('accuracy_score of',m ,'is:')
       print(accuracy_score(y_test,predm))
       print(confusion_matrix(y_test,predm))
```

- Key Metrics for success in solving problem under consideration

Selection of a model requires evaluation and evaluation requires a good metric. This is indeed important. If we optimize a model based on incorrect metric, then, our model might not be suitable for the business goals.

We have a number of metrics, for example, accuracy, recall, precision, F1 score, area under receiver operating characteristic curve, to choose from.

CONCLUSION

Key aspects of building successful classifier are:

- Selecting correct data according to the purpose or problem statement.
- Proper processing and understanding of the data
- Selecting the model and optimizing the model.

I have used tfidfvectorizer for convert object data type into int datatype as machine doesn't understand object type data.

I have used three Classification model and found Random Forest Classifier to be the best fit. It is giving good accuracy.