FLIP ROBO

RATING PREDICTION

Submitted by:

JAYASURYA E

Acknowledgment

Recommendation systems are an important unit in today's e-commerce applications, such as targeted advertising, personalized marketing and information retrieval. In recent years, the importance of contextual information has motivated generation of personalized recommendations according to the available contextual information of users. Compared to the traditional systems which mainly utilize product rating history, review-based recommendation hopefully provides more relevant results to product. We introduce a review-based recommendation approach that obtains contextual information by e-commerce product reviews. The proposed approach relates to features obtained by analysing textual reviews using methods developed in Natural Language Processing (NLP) and information retrieval discipline to compute a utility function over a given item. An item utility is a measure that shows how much it is preferred according to review current context. In our system, the context inference is modelled as similarity between the product rating history and the product reviews history. As an example, application we used our method to collect contextual data from customers reviews of e-commerce sector and use it to produce review-based rating prediction. The predicted ratings can generate recommendations that are product-based and should appear at the recommended items list in the product page. A comprehensive review of the literature, theories and models have been carried out to propose the models for customer review and rating prediction.

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. In this paper, a machine learning model is proposed to predict a rating prediction based on data related to the e-commerce (product, rating, review etc.). During the development and evaluation of our model, we will show the code used for each step followed by its output. This will facilitate the reproducibility of our work. In this study, Python programming language with a number of Python packages will be used.

- **Conceptual Background of the Domain Problem:**

  The main objectives of this study are as follows:

  - Apply web scraping tool and prepare a dataset of required features.
  - To apply NLP (Natural Language processing) technique for approaching a textual review.
  - To apply data pre-processing and preparation techniques in order to obtain clean data.

- To build machine learning models able to do rating prediction based on data is collected from the e-commerce sector i.e., product, review and rating.
- To analyse and compare model's performance in order to choose the best model
- To apply an Hyperparameter tuning for best model to obtain a better accuracy.

• Literature Review:

Natural Language Processing (NLP)

Natural Language Processing (NLP) is a way of analysing texts by computerized means. NLP involves gathering of knowledge on how human beings understand and use language. This is done in order to develop appropriate tools and techniques which could make computer systems understand and manipulate natural languages to perform various desired tasks.

NLP as a theoretically motivated range of computational techniques for analysing and representing naturally occurring texts, at one or more levels of linguistic analysis for the purpose of achieving human-like language processing for a range of tasks or applications. The term NLP is normally used to describe the function of software or hardware components in a computer system which analyse or synthesize spoken or written language. The 'natural' epithet is meant to distinguish human speech and writing from more formal languages, such as mathematical notations or programming languages, where vocabulary and syntax are comparatively restricted,

NLP is categorized into the following five areas:

1. Natural Language Understanding
2. Natural Language Generation
3. Speech or Voice recognition
4. Machine Translation
5. Spelling Correction and Grammar Checking

The increase demands for software's that process text of all kinds have tremendously been influenced by the advent of the Internet and World Wide Web. Over a decade, Internet publishing has become a common place activity for private individuals, commercial enterprises, and government organizations, as well as traditional media companies, and the medium of most of these communications and transactions is primarily natural language. Various forms of keyword processing provide access to Web sites as well as organizational principles for retrieving, navigating and browsing web pages within those sites. Search engines and spam filters are now of everyday life and work well enough that their viability as products is not in question. The language is more than transfer of information. Language is a set of resources to enable us to share meanings, but is not best thought of as a means for "encoding" meanings. The foundations of NLP fall within a number of disciplines being: computer and information sciences, linguistics, mathematics, electrical and electronic engineering, psychology, artificial intelligence and robotics, etc. NLP applications comprise a number of fields of studies, such as natural language text processing and summarization, machine translation, user interfaces, multilingual and cross language information retrieval, speech recognition, artificial intelligence and expert systems.

Machine learning:

Machine learning is a form of artificial intelligence which compose available computers with the efficiency to be trained without being veraciously programmed. Machine learning interest on the extensions of computer programs which is capable enough to modify when unprotected to new-fangled data. Machine learning algorithms are broadly classified into three divisions, namely; Supervised learning, Unsupervised learning and Reinforcement learning. Supervised learning is a learning in which we teach or train the machine using data which is well labelled that means some data is already tagged with correct answer. After that, machine is provided with new set of examples so that supervised learning algorithm analyses the training data and produces a correct outcome from labelled data. Unsupervised learning is the training of machine using information that is neither classified nor labelled and allowing the algorithm to act on that information without guidance. Here the task of machine is to group unsorted information according to similarities, patterns and differences without any prior training of data. Unlike, supervised learning, no teacher is provided that means no training will be given to the machine. Therefore, machine is restricted to find the hidden structure in unlabelled data by our-self.

Reinforcement learning is an area of Machine Learning. Reinforcement, it is about taking suitable action to maximize reward in a particular situation. It is employed by various software and machines to find the best possible behaviour or path it should take in a specific situation. Reinforcement learning differs from the supervised learning in a way that in supervised learning the training data has the answer key with it so the model is trained with the correct answer itself whereas in reinforcement learning, there is no answer but the reinforcement agent decides what to do to perform the given task. In the absence of training dataset, it is bound to learn from its experience. Machine learning has many applications out of which one of the applications is prediction of e-commerce. The e-commerce market is one of the most competitive in terms of and same tends to be vary significantly based on lots of factor, forecasting review-based rating prediction is an important modules in decision making for both the retailers and customers, finding rating prediction strategies and determining suitable times hence it becomes one of the prime fields to apply the concepts of machine learning to optimize and predict the rating with high accuracy. The study on e-commerce trend is felt important to support the decisions in urban planning. The e-commerce is an unstable stochastic process. Customer's decisions are based on the market trends to reap maximum returns. Developers are interested to know the future trends for their decision making. To accurately estimate review-based rating strategy and future trends, large amount of data that influences the data is collected from the Indian online shoppers is required for analysis, modelling and forecasting. The factors that affect the e-commerce sector have to be studied and model to be build based on review we want find rating range of the customer. It is inferred that establishing a simple linear mathematical relationship for these time-series data is found not viable for forecasting. Hence it became imperative to establish a non-linear model which can well fit the data characteristic to analyse and forecast future trends. As the e-commerce is fast developing sector, the analysis and forecast of e-commerce sector using mathematical modelling and other scientific techniques is an immediate urgent need for decision making by all those concerned. The increase in population as well as the industrial activity is attributed to various factors, the most prominent being the recent spurt in the knowledge sector viz. Information Technology (IT) and Information technology enabled services. Demand for e-commerce started of showing an upward trend and the e-commerce activity started booming. Retailer started investing in Indian online shopping Industry. The need for predicting the trend in review-based ratings was felt by all in the online shopping industry. Therefore, in this paper, we present various important features to use while predicting rating based on their review with good accuracy. Predictive modelling largely overlaps with the field of machine learning. There are two types of predictive models. They are Classification models, that predict class membership, and Regression models that predict a number. These models are then made up of algorithms. We can use Classification, using various features to have accuracy score, classification report and confusion matrix. While using features in a classification model some feature engineering is required for better prediction. Often a set of features multinomialnb classifier, sgd classifier, k-neighbors classifier and linear support vector classifier is used for making better model fit. So, it directs to the best application of classification models in addition to other techniques to optimize the result.

• Motivation of the problem undertaken:

Our client has decided to check the review-based rating prediction on their e-commerce sector. The e-commerce sector is decided to use NLP techniques and Machine Learning algorithm to predict rating based on no rating is given to review. In such cases, we have to build an application which can predict the rating by seeing the review.

For this e-commerce sector wants to know:

• Which variables are important to predict the rating?

• How do these variables describe the rating based on their review of the product?

• Analytical Problem Framing

Mathematical/ Analytical Modelling of the Problem Statistical Analysis:

Once it comes time to analyse the data, there are an array of statistical model's analysts may choose to utilize. The most common techniques will fall into the following two groups:

1. Supervised learning, including regression and classification models.
2. Unsupervised learning, incuding clustering algorithms and association rules.

- **Classification Model:**

  The main goal of the Classification algorithm is to identify the category of a given dataset, and these algorithms are mainly used to predict the output for the categorical data.

  Classification algorithms can be better understood using the below diagram. In the below diagram, there are two classes, class A and Class B. These classes have features that are similar to each other and dissimilar to other classes.

  The algorithm which implements the classification on a dataset is known as a classifier. There are two types of Classifications:
  1. **Binary Classifier:** If the classification problem has only two possible outcomes, then it is called as Binary Classifier.
     Eg: YES or NO, MALE or FEMALE, SPAM or NOT SPAM, CAT or DOG, etc.
  2. **Multi-class Classifier:** If a classification problem has more than two outcomes, then it is called as Multi-class Classifier.
     Eg: Classifications of types of crops, Classification of types of music.

  The most traditional regression model is Multinomial NB Classifier, SGD Classifier, K-Neighbors classifier and Linear support vector classifier.
  There are 4 main components of an analytics model, namely:
  1. Data Component
  2. Algorithm Component
  3. Real World Component
  4. Ethical Component

• Data Preparation

In this study, we will use a data which is collected from the e-commerce sector has decided to check the review-based rating prediction. Our client is decided to use NLP technique and Machine learning algorithm to predict rating based on no rating is given to review. In such cases, we have to build an application which can predict the rating by seeing the review. The data is provided in the rating_prediction.xlsx file.

• Data Description

The dataset contains 20050 records (rows) and 3 features (columns).

Here, we will provide a brief description of dataset features. Since the number of features is 3, we will attach the original data description file to this study for more information about the dataset. Now, we will mention the feature name of product, rating and review.

- ## Data Pre-processing:

1.Rating Context Representation:
Each product will be represented as a set of his review. User representation composed of 5 strings, for each possible rating value. Each string is a concatenation of the user's reviews with the corresponding rating value.

2.Review Context Representation:
Similar to rating representation, each product will be represented as a set of his reviews. Review representation composed of textual context for each possible rating value.

3.Preproccesing:
In this study, we deal with textual dataset (corpuse) which presented in natural-language form (human language), which present difficulties as described in Stanford Handbook. Therefore, text normalization is required in order to reduce language diversity including transformation to canonical form for further processing. We achieve this by performing textual normalization as presented in algorithm.
The main steps are:
1. Convert characters into lower cases.
2. Remove punctuation.
3. Remove numbers.
4. Remove stop words, described in Onix.
5. Replace slang terms, described in Twitter Dictionary.
6. Word stemming to its root, described in Stanford Handbook.

There are some column features which is indicated with string values so we are replacing such feature into unique functions. They are product, review and rating.

The new column feature is added to indicate the length of text in review feature.

The Rating value counts:
a. The rating 5 has 11671 counts
b. The rating 4 has 4095 counts
c. The rating 1 has 2202 counts
d. The rating 3 has 1439 counts
e. The rating 2 has 643 counts

There are no null values in the dataset. Since, we can see string values in the dataset, so we proceed with further steps.

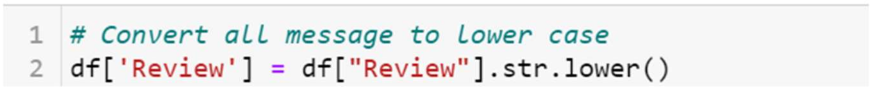
- **Data Cleaning:**

Column	Data types
Product	object
Rating	object
Review	object

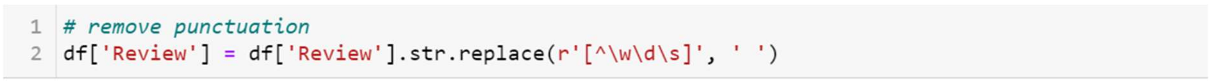
Now, we can proceed with NLP Technique.

Regular Expression:

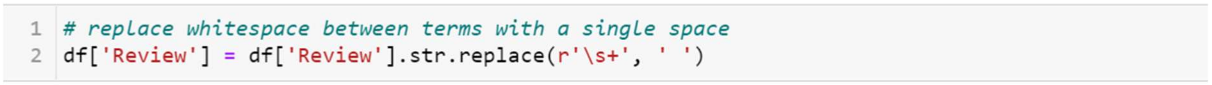
1. To Convert all message to lower case:
2. To Remove Punctuation
3. To Replace whitespace between terms with a single space.
4. To Remove leading and trailing white space
5. To Replace numbers with 'numbr'
6. To Replace Urls with 'webaddress'
7. To Remove Stopwords



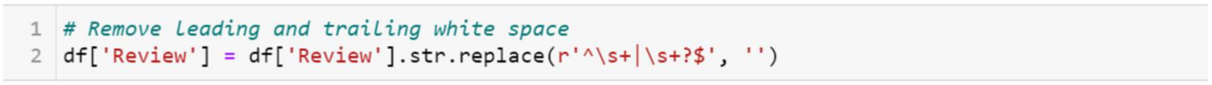
```python
# Convert all message to lower case
df['Review'] = df["Review"].str.lower()
```



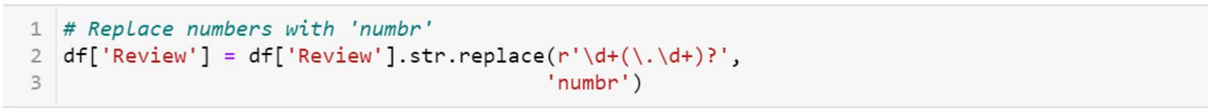
```python
# remove punctuation
df['Review'] = df['Review'].str.replace(r'[^\w\d\s]', ' ')
```



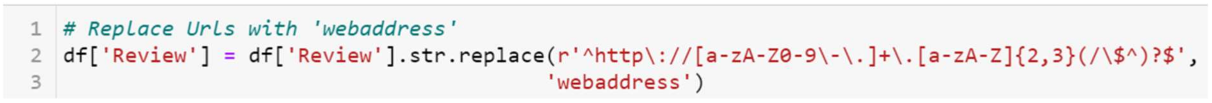
```python
# replace whitespace between terms with a single space
df['Review'] = df['Review'].str.replace(r'\s+', ' ')
```



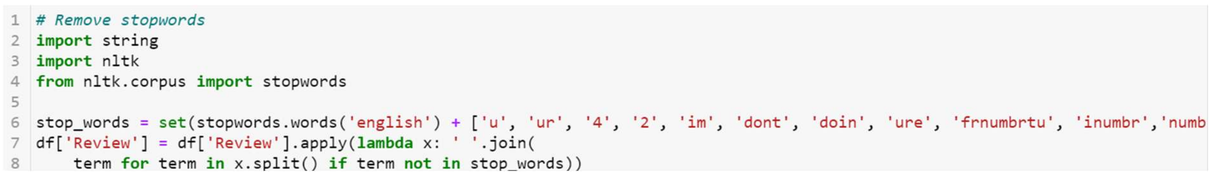
```python
# Remove leading and trailing white space
df['Review'] = df['Review'].str.replace(r'^\s+|\s+?$', '')
```



```python
# Replace numbers with 'numbr'
df['Review'] = df['Review'].str.replace(r'\d+(\.\d+)?',
                                       'numbr')
```



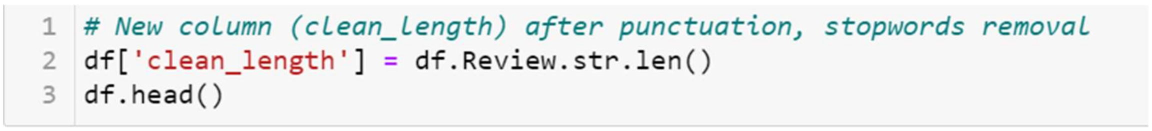
```python
# Replace Urls with 'webaddress'
df['Review'] = df['Review'].str.replace(r'^http\://[a-zA-Z0-9\-\.]+\.[a-zA-Z]{2,3}(/\$^)?$',
                                       'webaddress')
```



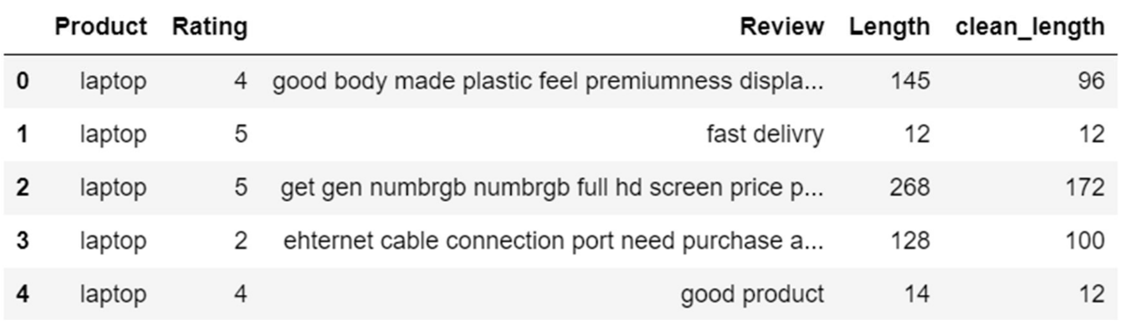
```python
# Remove stopwords
import string
import nltk
from nltk.corpus import stopwords

stop_words = set(stopwords.words('english') + ['u', 'ur', '4', '2', 'im', 'dont', 'doin', 'ure', 'frnumbrtu', 'inumbr','numb
df['Review'] = df['Review'].apply(lambda x: ' '.join(
    term for term in x.split() if term not in stop_words))
```

The new feature is added to show the cleaned length of review feature after removal of punctuation, leading and trailing white space, numbers, urls and stop words.



```python
# New column (clean_length) after punctuation, stopwords removal
df['clean_length'] = df.Review.str.len()
df.head()
```

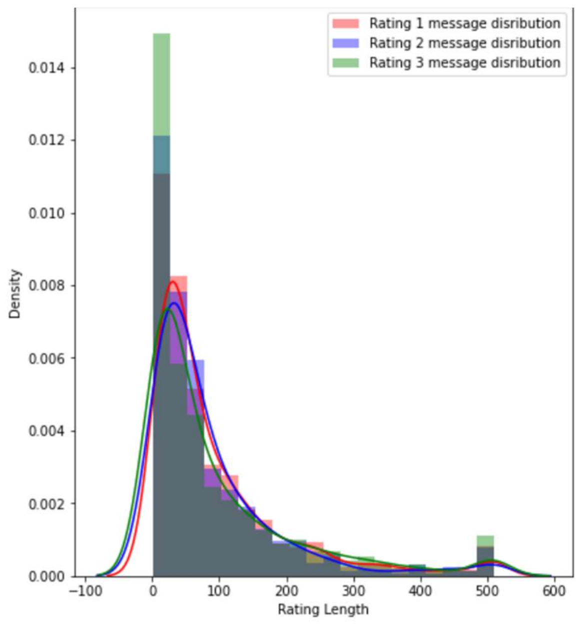


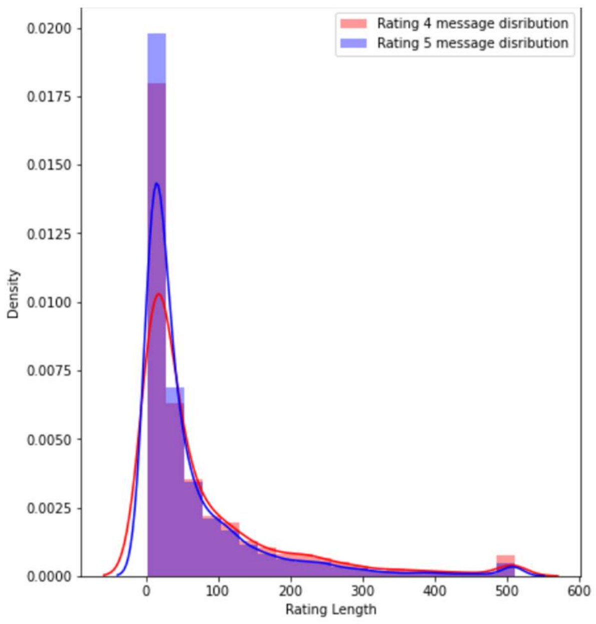
	Product	Rating	Review	Length	clean_length
0	laptop	4	good body made plastic feel premiumness displa...	145	96
1	laptop	5	fast delivry	12	12
2	laptop	5	get gen numbrgb numbrgb full hd screen price p...	268	172
3	laptop	2	ehternet cable connection port need purchase a...	128	100
4	laptop	4	good product	14	12

Total length removal of review feature:

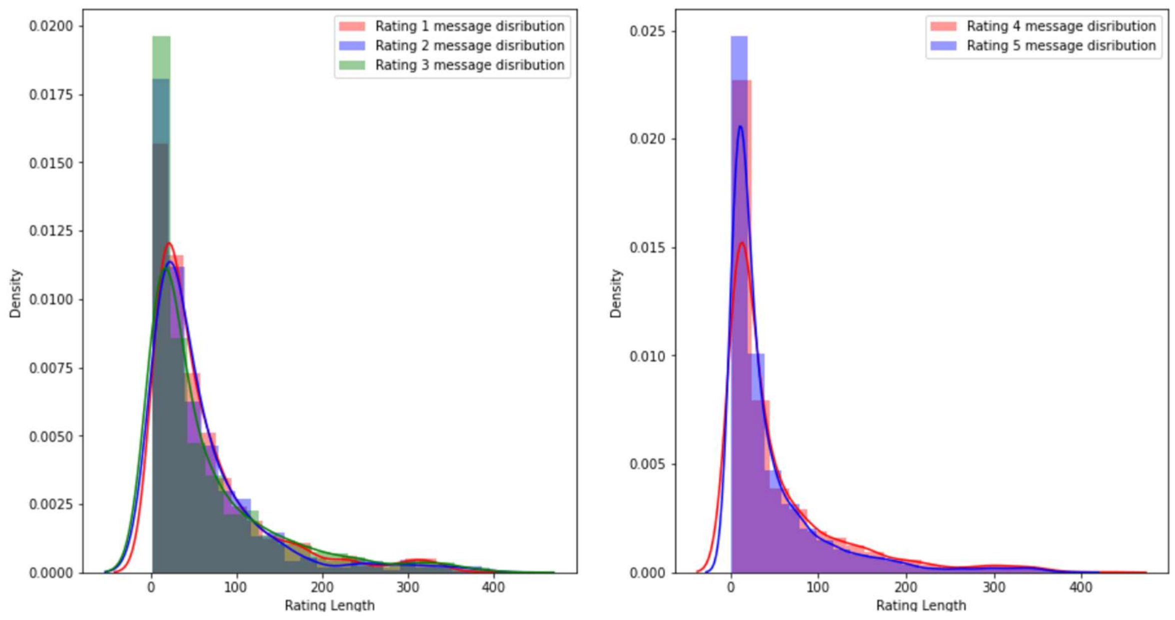
1. Original Length: 1445748
2. Cleaned Length: 999563

Message distribution before data cleaning:





Message distribution after data cleaning:



- ## Hardware and Software Requirements and Tools Used

PYTHON Jupyter Notebook:

Key Features:

An open-source solution that has simple coding processes and syntax so it's fairly easy to learn Integration with other languages such as C/C++, Java, PHP, C#, etc. Advanced analysis processes through machine learning and text mining.

Python is extremely accessible to code in comparison to other popular languages such as Java, and its syntax is relatively easy to learn making this tool popular among users that look for an open-source solution and simple coding processes. In data analysis, Python is used for data crawling, cleaning, modelling, and constructing analysis algorithms based on business scenarios. One of the best features is actually its user-friendliness: programmers don't need to remember the architecture of the system nor handle the memory – Python is considered a high-level language that is not subject to the computer's local processor.

Libraries and Packages used:

Matplotlib:

Matplotlib is a Python library that uses Python Script to write 2-dimensional graphs and plots. Often mathematical or scientific applications require more than single axes in a representation. This library helps us to build multiple plots at a time. You can, however, use Matplotlib to manipulate different characteristics of figures as well.

The task carried out is visualization of dataset i.e., nominal data, ordinal data, continuous data, heatmap display distribution for correlation matrix and null values, boxplot distribution for checking outliers, scatter plot distribution for modelling approach, subplot distribution for analysis and comparison, feature importance and common importance features, line plot for prediction values vs actual values.

Numpy:

Numpy is a popular array – processing package of Python. It provides good support for different dimensional array objects as well as for matrices. Numpy is not only confined to providing arrays only, but it also provides a variety of tools to manage these arrays. It is fast, efficient, and really good for managing matrices and arrays.

The Numpy is used to managing matrices i.e., Accuracy score, Confusion matrix and Classification report i.e., described the values of train test dataset.

Pandas:

Pandas is a python software package. It is a must to learn for data-science and dedicatedly written for Python language. It is a fast, demonstrative, and adjustable platform that offers intuitive data-structures. You can easily manipulate any type of data such as – structured or time-series data with this amazing package.

The Pandas is used to execute a Data frame i.e., test set.csv, train set.csv, predicted values of model approach, conclusion.

Scikit Learn:

Scikit learn is a simple and useful python machine learning library. It is written in python, cython, C, and C++. However, most of it is written in the Python programming language. It is a free machine learning library. It is a flexible python package that can work in complete harmony with other python libraries and packages such as Numpy and Scipy.

Scikit learn library is used to import a pre-processing function i.e., tfidf vectorizer, multinomial nb, sgd classifier, k-neighbors, linear support vector classifier, accuracy score, confusion matrix and classification report, train test split, grid search cv and hyper tuning.

- ## Models Development and Evaluation

In this section, we choose the type of machine learning prediction that is suitable to our problem. We want to determine if this is a regression problem or a classification problem. In this project, we want to predict the rating of e-commerce sector of given information about it. The rating we want to predict, this can be seen by looking at the target variable in our dataset rating:

That means that the prediction type that is appropriate to our problem is multi-class classification. Now, we move to choose the modelling techniques we want to use. There are a lot of techniques available for classification problems like MultinominalNB, SGD Classifier, Linear support vector classifier, k-neighbors classifier. In this project, we will test many modelling techniques, and then choose the technique(s) that yield the best results. The techniques that we will try are:

1. MultinomalNB Classifier

The multinomial Naive Bayes classifier is suitable for classification with discrete features (e.g., word counts for text classification). The multinomial distribution normally requires integer feature counts. However, in practice, fractional counts such as tf-idf may also work.

2. SGD Classifier

SGD Classifier is a linear classifier (SVM, logistic regression) optimized by the SGD. These are two different concepts. While SGD is a optimization method, Logistic Regression or linear Support Vector Machine is a machine learning algorithm/model.

3. K- Neighbors Classifier

KNN works by finding the distances between a query and all the examples in the data, selecting the specified number examples (K) closest to the query, then votes for the most frequent label (in the case of classification) or averages the labels (in the case of regression).

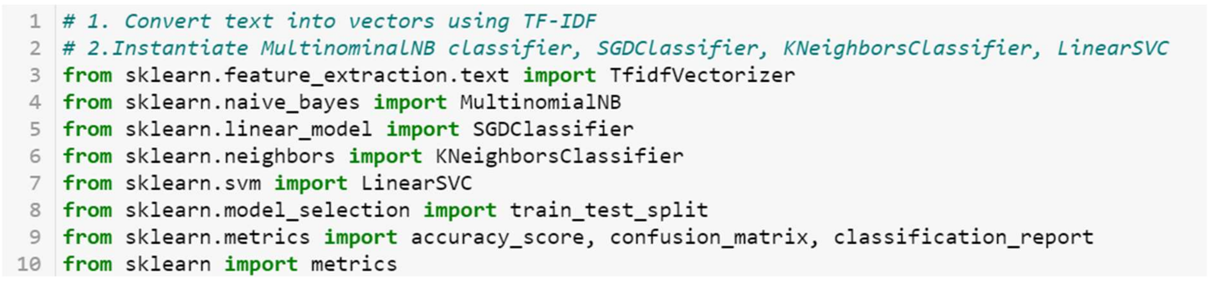
4. Linear Support Vector Classifier

The objective of a Linear SVC (Support Vector Classifier) is to fit to the data you provide, returning a "best fit" hyperplane that divides, or categorizes, your data. From there, after getting the hyperplane, you can then feed some features to your classifier to see what the "predicted" class is.

• Model Building and Evaluation

In this part, we will build our prediction model: we will choose algorithms for each of the techniques we mentioned in the previous section. After we build the model, we will evaluate its performance and results.

Importing libraries for metrics and model building:



```python
# 1. Convert text into vectors using TF-IDF
# 2.Instantiate MultinominalNB classifier, SGDClassifier, KNeighborsClassifier, LinearSVC
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.naive_bayes import MultinomialNB
from sklearn.linear_model import SGDClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.svm import LinearSVC
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score, confusion_matrix, classification_report
from sklearn import metrics
```

Modelling Approach:

For each one of the techniques mentioned in the previous section (MultinominalNB, SGD Classifier, Linear support vector classifier, k-neighbors classifier.), we will follow these steps to build a model:

- Choose an algorithm that implements the corresponding technique
- Search for an effective parameter combination for the chosen algorithm
- Create a model using the found parameters

- Train (fit) the model on the training dataset
- Test the model on the test dataset and get the results

Classification Method:

- Using Scikit-Learn, we can build a model.

```
maxAccu=0
BestRS=0
for i in range(1,200):
    x_train, x_test, y_train, y_test = train_test_split(x,y,test_size=.30,random_state=i)
    naive = MultinomialNB()
    naive.fit(x_train, y_train)
    predict_test = naive.predict(x_test)
    acc=accuracy_score(y_test, predict_test)
    if acc>maxAccu:
        maxAccu=acc
        BestRS=i
print("Best acuracy is " ,maxAccu," on Random_state ",BestRS)
```

```
Best acuracy is  0.6408977556109726  on Random_state  135
```

Splitting the Dataset:

As usual for supervised machine learning problems, we need a training dataset to train our model and a test dataset to evaluate the model. So, we will split our dataset randomly into two parts, one for training and the other for testing. For that, we will use another function from Scikit-Learn called train_test_split():

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

Performance Metric:

For evaluating the performance of our models, we will use accuracy score, confusion matrix and classification report. If the predicted value of the element, and the corresponding true value, then for all the element is calculated as:

```
def eval(x):
    mod =x
    mod.fit(x_train, y_train)
    predict_test = mod.predict(x_test)
    print("Accuracy score is ", accuracy_score(y_test, predict_test)*100)
    print("Confusion matrix is", confusion_matrix(y_test,predict_test))
    print("Classification report", metrics.classification_report(y_test, predict_test, digits=2))
```

Model Building	Accuracy Score	Macro average	Weighted average
MultinominalNB	0.64	0.27	0.53
SGD	0.68	0.36	0.59
KNeighbors	0.60	0.33	0.56
LinearSVC	0.66	0.41	0.61

Comparing the Performance metric and Cross validation Score:

Performance Metric	Cross -Validation Score
0.64	0.63
0.68	0.65
0.60	0.57
0.66	0.62

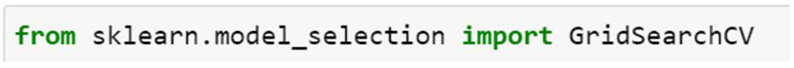
Here we have handled the problem of the overfitting and the underfitting by checking the Accuracy score.
Min difference in Accuracy score and cross validation score is for MultinomialNB model.so this is our best model.

Hyper Parameter Tuning:

Hyperparameters are crucial as they control the overall behaviour of a machine learning model. The ultimate goal is to find an optimal combination of hyperparameters that minimizes a predefined loss function to give better results.

The performance metric for Multinomianl NB Classifier has higher values when compared with other values.so, we are going to perform hyperparameter tuning for this model to get better result.

Importing GridSearchCV:



```
from sklearn.model_selection import GridSearchCV
```

Hyperparameter Tuning for Multinominal NB:

Firstly, we will use GridSearchCV() to search for the best model parameters in a parameter space provided by us. criterion, max features and random state.

```python
from sklearn.naive_bayes import MultinomialNB
parameters={'alpha' : [ 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1.0] ,'fit_prior' : ['True', 'False'],
            'class_prior' : [None, [.1,.9],[.2, .8]]}
mnb=MultinomialNB()
clf=GridSearchCV(mnb,parameters)
clf.fit(x_train,y_train)
print(clf.best_params_)
```

```
{'alpha': 0.2, 'class_prior': None, 'fit_prior': 'True'}
```

We defined the parameter space above using reasonable values for chosen parameters.

```python
mnb=MultinomialNB(alpha = 0.2, fit_prior = 'True', class_prior = None)
mnb.fit(x_train,y_train)
mnb.score(x_train,y_train)
pred_decision=mnb.predict(x_test)
mnbs=accuracy_score(y_test,pred_decision)
print('Accuracy Score:',mnbs*100)
mnbscore=cross_val_score(mnb,x,y,cv=5)
mnbc=mnbscore.mean()
print('Cross Val Score:',mnbc*100)
```

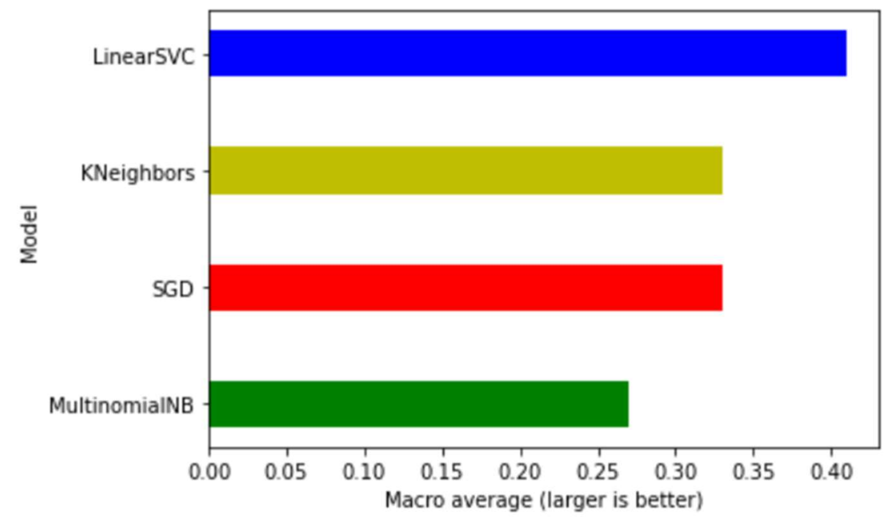
```
Accuracy Score: 66.46716541978387
Cross Val Score: 62.71820448877806
```

We defined the performance model score and cross validation score of hyperparameter tuning for Multinominal NB using chosen parameters. We are getting model accuracy and cross validation has 66.46% & 62.71% respectively. We consider Multinominal NB is our best model for these datasets.

Performance Interpretation:

Macro Average:

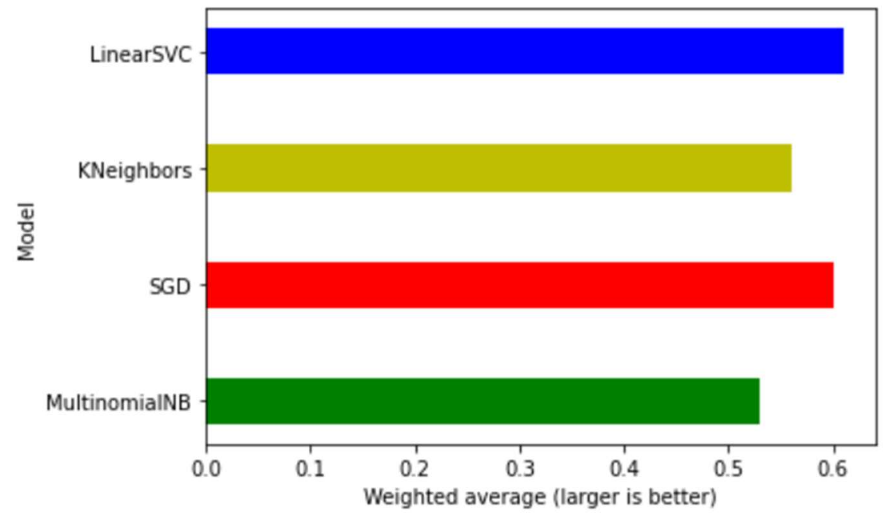
```python
x = ['MultinomialNB','SGD', 'KNeighbors', 'LinearSVC']
y = [0.27, 0.33, 0.33, 0.41]
colors = ["g", "r", "y", "b", "brown"]
fig, js = plt.subplots()
plt.barh(y=range(len(x)), tick_label=x, width=y, height=0.4, color=colors);
js.set(xlabel="Macro average (larger is better)", ylabel="Model");
```



By looking at the table and the graph, we can see that LinearSVC has the larger macro average of 0.41. After that, SGD and KNeighbor comes with an similar macro average score: 0.33 and 0.33 respectively. At last, the MultinomialNB comes with an 0.27.
So, in our experiment, the best model is LinearSVC and the worst model is MultinomialNB. We can see that the difference in macro average between the best model and the worst model is significant; the best model has least of the worst model.

Weighted Average:

```
x = ['MultinomialNB','SGD', 'KNeighbors', 'LinearSVC']
y = [0.53, 0.60, 0.56, 0.61]
colors = ["g", "r", "y", "b", "brown"]
fig, js = plt.subplots()
plt.barh(y=range(len(x)), tick_label=x, width=y, height=0.4, color=colors);
js.set(xlabel="Weighted average (larger is better)", ylabel="Model");
```

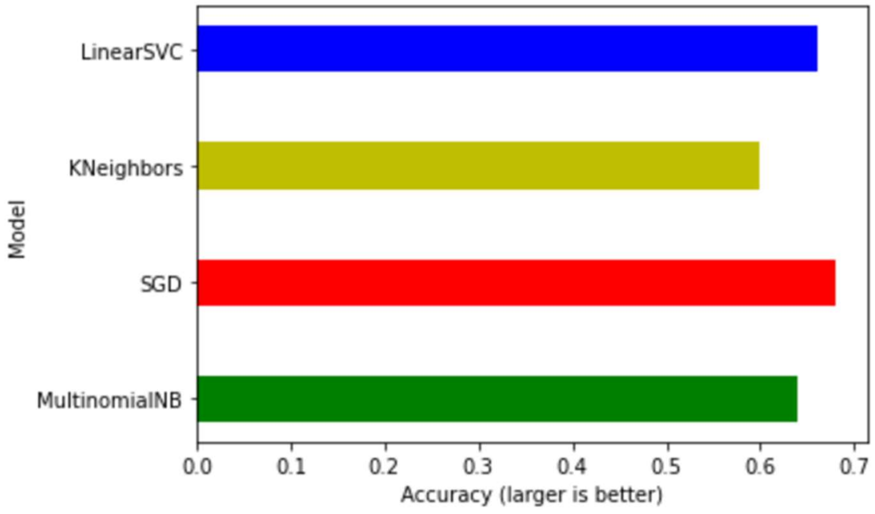


By looking at the table and the graph, we can see that LinearSVC has larger weighted average of 0.61 followed by SGD of 0.6. At last, the KNeighbors

and MultinomialNB comes with a similar weighted average: 0.56 and 0.53 respectively.
So, in our experiment, the best model is LinearSVC and the worst model is MultinomialNB. We can see that the difference in weighted average between the best model and the worst model is significant; the best model has almost least of the worst model.

Accuracy Score:

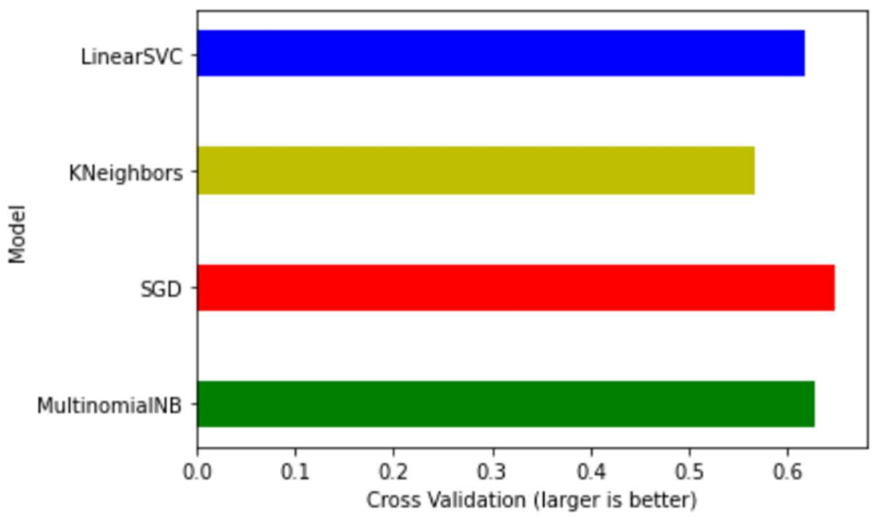
```
x = ['MultinomialNB','SGD', 'KNeighbors', 'LinearSVC']
y = [0.64, 0.68, 0.60, 0.66]
colors = ["g", "r", "y", "b", "brown"]
fig, js = plt.subplots()
plt.barh(y=range(len(x)), tick_label=x, width=y, height=0.4, color=colors);
js.set(xlabel="Accuracy (larger is better)", ylabel="Model");
```



By looking at the table and the graph, we can see that SGD has the larger accuracy of 0.68 followed by LinearSVC of 0.66. At last, the MultinomialNB and KNeighbors comes with an similar accuracy score: 0.64 and 0.60 respectively.
So, in our experiment, the best model is SGD and the worst model is KNeighbors. We can see that the difference in Accuracy score between the best model and the worst model is significant; the best model has least of the worst model.

Cross validation Score:

```
x = ['MultinomialNB','SGD', 'KNeighbors', 'LinearSVC']
y = [0.627, 0.648, 0.567, 0.618]
colors = ["g", "r", "y", "b", "brown"]
fig, js = plt.subplots()
plt.barh(y=range(len(x)), tick_label=x, width=y, height=0.4, color=colors);
js.set(xlabel="Cross Validation (larger is better)", ylabel="Model");
```



By looking at the table and the graph, we can see that SGD has the larger cross validation score of 0.648. After that MultinomialNB and LinearSVC with a similar cross validation score: 0.627 and 0.618 respectively. At last, the KNeighbors comes with an 0.567.
So, in our experiment, the best model is SGD and the worst model is KNeighbors. We can see that the difference in Cross Validation score between the best model and the worst model is significant; the best model has least of the worst model.
Finally, Min difference in Accuracy score and cross validation score is for MultinomialNB model. We compared and came to know that our best model is MNBClassifier, then the worst model is Linear SVC.

Conclusion:

In this paper, we built several classification models to predict the review-based ratings of some product features. We evaluated and compared each model to determine the one with highest performance. We also looked at how some models rank the features according to their importance. In this paper, we followed the NLP technique and Machine learning algorithm process starting with getting the data, then cleaning and pre-processing the data, followed by exploring the data and building models, then evaluating the results.

As a recommendation, we advise to use this model (or a version of it trained with more recent data) by e-commerce sector who want to get an idea about product. The model can be used also with datasets that covered areas provided that they contain the same features. We also suggest that people take into consideration the features that were deemed as most important as seen in the previous section; this might help them estimate the review-based rating prediction is better.

Learning Outcomes of the Study in respect of Data Science:

- Obtain, clean/process, and transform data.
- Analyze and interpret data using an ethically responsible approach.
- Use appropriate models of analysis, assess the quality of input, derive insight from results, and investigate potential issues.
- Apply computing theory, languages, and algorithms, as well as mathematical and statistical models, and the principles of optimization to appropriately formulate and use data analyses
- Formulate and use appropriate models of data analysis to solve hidden solutions to business-related challenges

Limitations of this work and Scope for Future Work:

There are many things that can be tried to improve the models predictions. We can create and add more variables, try different models with different subset of features and/or rows, etc. Some of the ideas are listed below:

- Combine the applicants with 1,2,3 or more dependents and make a new feature as discussed in the EDA part.
- Make independent vs independent variable visualizations to discover some more patterns.
- Arrive at the EMI using a better formula which may include interest rates as well.
- Try neural network using TensorFlow or PyTorch.

******