Amazon Reviews - Text Classification

Wajahat Ali Mughal
Applied Machine Learning
ID: 001191074

***Abstract*—Text classification is the process of categorizing the text into an organized group. NLP (Natural Language Processing), is being used by the text classifiers to automatically analyze the text and assign it accordingly to the categories based on the content of it. NLP can be utilized to help us comprehend and decipher text, sounds, and discourse of individuals. In this report, the text classification has performed on the dataset of Amazon review using NLP to analyze the reviews and predicts the number of starts according to the given reviews. Furthermore, after analyzing the reviews, we will divide in into two categories, "Luxury Beauty products" and "Prime Pantry products". The ML algorithm I used for the prediction are Random Forest and Decision Tree. After performing different steps on the data which are cleaning, splitting, pre-processing and ML implementation using both algorithm we found that Random Forest performed better. we got 93% in predicting product categories and 88% in review score.**

I. INTRODUCTION AND RELATED WORK

Natural language processing (NLP) refers to the part of computer science explicitly, the part of artificial intelligence which empowers the PCs to understand text and words similarly as people can. [1]

In our review, the given dataset contained 5 columns which are review_id, text, verified, review_score, and product_category. Just review_score column contains floating numbers, while other columns have a blend of text and number values. The principal challenge was to clean and bring together the information with the end goal of ML implementation. Consequently, we utilized a few methods, for example, eliminating stop words, extending constrictions, stemming, lemmatization, tokenization and so on. To clean and coordinate the information prepared of text classification. Random-forest and Decision-tree are the algorithms chosen for execution. This is on the grounds that these two algorithms are among the most well known algorithms utilized mostly for the classification. Decision tree is not difficult to build and is very quick for the classifying the unknown records. whereas, Random forest will beat any model in terms of performance since it contains an uncorrelated tree joined together to perform a specific task. That is why we picked Random Forest and Decision tree algorithms for the task. Our deliberate approach is to resolve the ML problems by exploratory analysis of data, data cleaning, splitting, pre-processing, ML implementation, results, and assessment. We would be utilizing a few matrices like train accuracy, test accuracy and recall to assess the models. Furthermore, for better understanding of the performance of the model, we used some visualizations like confusion matrix and ROC curve.

II. ETHICAL DISCUSSION

However man-made consciousness is changing the way in which organizations work, there are worries about how it might impact our lives. This isn't simply an intellectual or a cultural concern however a reputational risk for organizations, no organization needs to be defaced with information or AI morals embarrassments that influenced organizations like Amazon. For instance, there was critical reaction because of the offer of Rekognition to policing. This was trailed by Amazon's choice to quit giving this innovation to policing a year since they expect the legitimate lawful structure to be set up by then, at that point. [2]

In our review, we talked about text classification by predicting the "Review score" of the Amazon reviews. Furthermore, we categorize the products into two categories which are L̈uxury Beauty productsänd P̈rime Pantry products.̈ This sort of algorithm is exceptionally requesting in the present world because ecommerce market is expanding exponentially.

III. DATASET PREPARATION

Data preparation contains a couple of steps which we are going to perform one by one mainly these step consists of analysis, data cleaning, splitting and pre-processing.

IV. DATA ANALYSIS

Before solving a machine learning problem, it is expected of us to play out the exploratory analyses on the dataset, which not just aides us in the examination of the dataset, for example tracking down patterns, and spotting inconsistencies or exceptions. It also helps us to have better insights into the data by visualizing it through diverse types of graphical representations like graphs, charts, histograms etc.

```
#reading dataset
df = pd.read_csv("processed_reviews_split_RESIT_minimal.csv")
df.head()
```

	review_id	text	verified	review_score	product_category
0	product_review_000000	OMG this is sooo good.. Very Good!!!	True	5.0	prime_pantry
1	product_review_000001	This soap smells pretty good and it seems to w...	True	4.0	prime_pantry
2	product_review_000002	Don't seem to dissolve after quite some time. ...	False	1.0	prime_pantry
3	product_review_000003	these zip bags do as they should. very good. I...	True	5.0	prime_pantry
4	product_review_000004	What. A. Mess.\nI recommend making this in a m...	True	2.0	prime_pantry

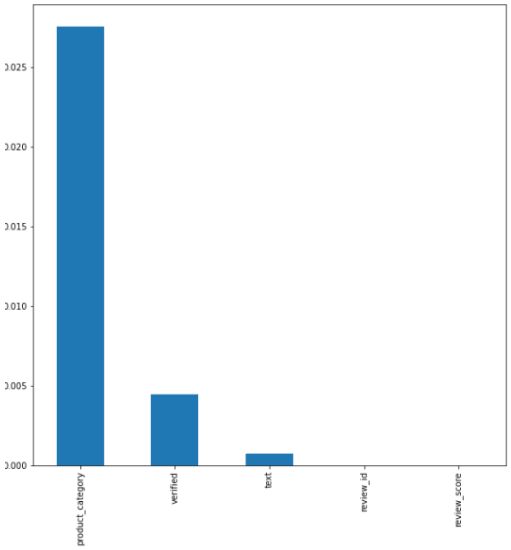
Importing dataset.

As we can see in the above image, we have first imported the dataset and try to view first five records. This dataset has 5 columns which are review_id text, verified and review_id. Review_id representing the id number of the reviews which every record got unique id.

```
#checking null values in percentage.
percent_missing = df.isnull().sum() * 100 / len(df)
missing_value_df = pd.DataFrame({'column_name': df.columns,'percent_missing': percent_missing})
```

Checking null values.

In the above figure, I have manage to find the number of null values in all columns of each record and try to plot the diagram accordingly. which is given below.



percentage of null values per columns.

After finding the null value, we will now review total number of records have given specific review under different scores.

```
review_score
 5.0    18681
 3.0    10000
 4.0    10000
 2.0     9000
 1.0     8500
-1.0     8000
dtype: int64
```

reviews score count.

V. Data Cleaning

As the name suggests, it is a process of cleaning data by identifying the errors and unwanted records which may have null values in it or unexpected values. These records need to be handled carefully, otherwise these records will affect the performance of the model in the process of prediction.

Data cleaning contains a couple of steps in first step I removed the white spaces and unwanted spaces.

```
def remove_spaces(text):
    text=text.strip()
    text=text.split()
    return ' '.join(text)
```

removing unwanted spaces

After removing unwanted spaces, we are going to remove contractions. It is important to remove it because people often writes short words like abbreviation to improve the performance of the text standardization. The code of that is given below.

```
contraction = {'cause':'because', 'aint': 'am not', 'aren\'t': 'are not'}

def mapping_replacer(x,dic):
    for words in dic.keys():
        if ' ' + words + ' ' in x:
            x=x.replace(' '+ words +' ' ,' '+dic[words]+' ' )
    return x
```

removing contractions

Next step in data cleaning I adopted by removing punctuations, brackets, numbers, etc. For better implementation of ML model. The soce I used is given as follows.

```
# updating text to lowercase, removing text between square brackets,
def clean_text(text):
    text = str(text).lower()
    text = re.sub('\[.*?\]', '', text)
    text = re.sub('https?://\S+|www\.\S+', '', text)
    text = re.sub('<.*?>+', '', text)
    text = re.sub('_', '', text)
    text = re.sub('\n', '', text)
    text = re.sub('\w*\d\w*', '', text)
    text = re.sub('\'','', text)
    return text
```

removing brackets

In the next step I performed tokenization which divide the text into smaller parts which is called a token in NLP.

```
def tokenise(text):
    words = word_tokenize(text)
    return words
```

Tokenisation

Tokenization will lead us to the next step, which is removing regular expressions from the test for better performance

```
import re
df['text'] = df['text'].map(lambda x: re.sub(r'\W+', ' ', str(x)))
df['text'] = df['text'].replace(r'\W+', ' ', regex=True)
```

```
df['text']=df['text'].apply(lambda x: mapping_replacer(x, contracti
```

```
df['text']=df['text'].apply(lambda x:clean_text(x))
```

```
df['text']=df['text'].apply(lambda x: remove_stopword(x))
```

```
df['text']=df['text'].apply(lambda x: lexicon_normalization(x))
```

Removing Regular Expressions

Pre Processing

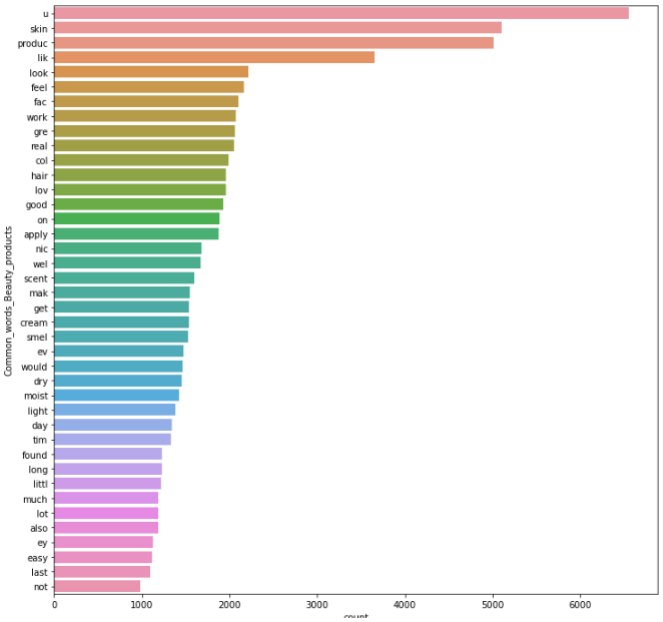
Pre-processing is a mandatory process before implementing the Machine learning model. It helps to get the best performance by processing the data before implementation.

Let us first count the number of common words for the L̈uxury Beauty Productsänd P̈rime Pantry Products.

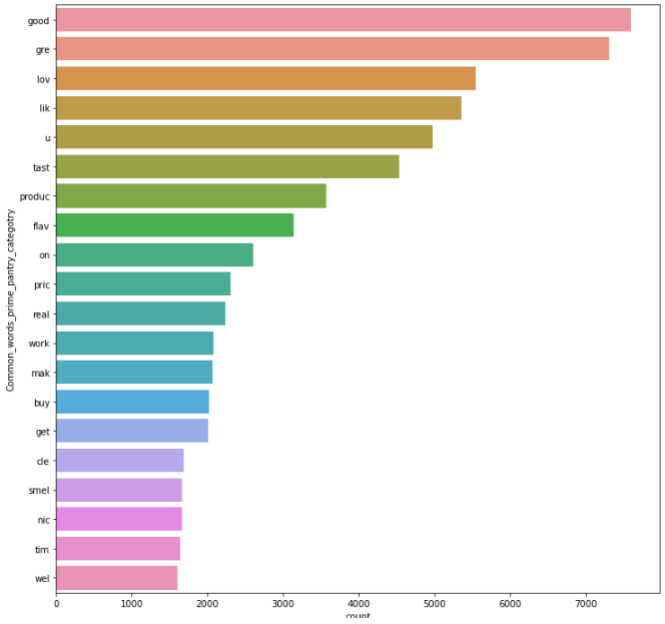
```
beauty=df[df['product_category']=='luxury_beauty']
```

```
top = Counter([item for sublist in beauty['text'] for item in sublist])
temp = pd.DataFrame(top.most_common(40))
temp.columns = ['Common_words_Beauty_products','count']
temp.style.background_gradient(cmap='Oranges')
```

Code for getting common words



Common words for Beauty Products



Common words for Prime Pantry Products

SPLITTING AND PRE-PROCESSING

Now we are going to divide the dataset into 2 parts, one will be used for training and other one will be for testing. and just before that, we will preprocess the data and by encoding of categorical features. This is the process of converting features into numerical data for the processing.

```
a=df.select_dtypes(include='object')
```

```
from sklearn import preprocessing
le = preprocessing.LabelEncoder()
df['product_category']=le.fit_transform(df['product_category'])
```

preprocessing

METHODS

In this stage, we will choose the ML models. The algorithms I choose for the task are Random forest and Decision tree for the prediction by performing training and testing on the dataset. After performing both models and finding the accuracy, pick one which would serve the best prediction accuracy and give great outcomes.

DECISION TREES

Decision Trees are a non-parametric supervised learning method utilized for classification and regression. The objective is to make a model that predicts the value of the target variable by learning clear decision principles inferred from the data features. A tree should be visible as a piecewise constant approximation.Decision trees are a profoundly successful construction inside which you can spread out choices and examine the potential results of picking those choices. They likewise assist you with shaping a fair image of the dangers and prizes associated with every conceivable action [3].

Reason of Choosing:- These are easy to implement and really fast in terms of classifying unknown records.

RANDOM FOREST

Random forest model is quite famous in ML supervised learning. This model can be utilized for both regression and classification problems. It is conceptually based on ensemble learning, which is the act of solving a complex problem by combining multiple classifiers to improve the performance of the model. The performance of the random forest is depends on the number of decisions trees [4]. Reason for Choosing:- Random Forest will beat any singular model since it is made out of uncorrelated trees joined together to play out a task of predicting with high accuracy.

EXPERIMENTS AND EVALUATION

In this task, we are going to implement the models we have chosen for the implementation.

Before implementation, we are going to use the bag of words technique because text is a bit messy, and it requires well-defined input and output for modelling the ML algorithm. Bag of words technique will be used to obtain certain features

from text to any no. possible for the model implementation. we use the bag of words technique and limit the maximum features to 2500 and then converted the data frame into vectors from data frames.

```
vectorizer2 = CountVectorizer(max_features=2500)
X2 = vectorizer2.fit_transform(df['text'])
b2=vectorizer2.get_feature_names()
```

```
#defining dataframe which is converted into vector which also contains other features
df2 = pd.DataFrame(X2.toarray(), columns=vectorizer2.get_feature_names())
```

bag of words for review score.

It will lead us to next step where we are going to define inputs and outputs as x and y, X = df1 and y = df[['review_score']. for scaling the features we are using

from sklearn.preprocessing import StandarScaler
scaler = StandardScaler()
x=scaler.fit_transform(x)

Decision Tree for Review Score

After scaling the data, it is then split for training and testing. and after we have implemented Decision Tree Classification

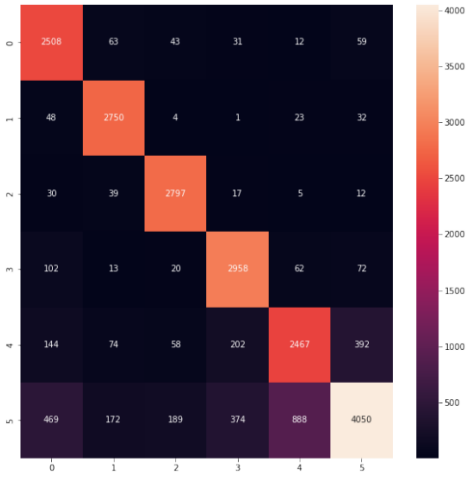
```
from sklearn.preprocessing import StandardScaler
scaler = StandardScaler()
X2 = scaler.fit_transform(X2)
```

```
X_train2, X_test2, y_train2, y_test2 = train_test_split(X2,y2, test_size=0.33, random_state=42)
```

```
dt2 =DecisionTreeClassifier(random_state=1024)
dt2.fit(X_train2, y_train2)
y_predict_dt2 = dt2.predict(X_test2)
# confusion_matrix
cm = confusion_matrix(y_test2, y_predict_dt2)
plt.figure(figsize=(10,10))
sns.heatmap(cm, annot=True, fmt="d")
```

Decision Tree implementation

After Performing the decision tree on the review score we have implemented the result and try to view it using confusion matrix graph. As we can see we got accuracy of 82% and precision is around 82% and recall is 86%. Based on these result, we have implemented the confusion tree after observing it as we can see I got 4050 time 5 stars and 2 stars are around 2797 times respectively.



Decision Tree confusion Matrix

```
print('accuracy',accuracy_score((y_test2), y_predict_dt2))
print('precision',precision_score(y_test2, y_predict_dt2, average='macro'))
print('recall',recall_score(y_test2, y_predict_dt2, average='macro'))
```

```
accuracy 0.82766761095373
precision 0.8265300688783562
recall 0.8609054743556918
```

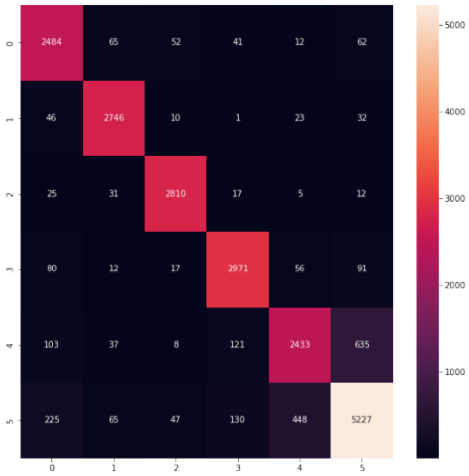
Random Forest For Review Score

We have implemented the random forest

```
rf2 =RandomForestClassifier(random_state=1000)
rf2.fit(X_train2, y_train2)
```

```
y_predict_rf2 = rf2.predict(X_test2)
# confusion_matrix
cm = confusion_matrix(y_test2, y_predict_rf2)
plt.figure(figsize=(10,10))
sns.heatmap(cm, annot=True, fmt="d")
```

As we can see in the above picture we have implemented code we got an accuracy of 88% which is higher than the decision tree. to observe the results, we have implemented the confusion matrix in which we can see that correct predictions for 5 stars is 5227 times and 4 starts are 2473 times.




```
print('accuracy',accuracy_score((y_test2), y_predict_rf2))
print('precision',precision_score(y_test2, y_predict_rf2, average='macro'))
print('recall',recall_score(y_test2, y_predict_rf2, average='macro'))
```

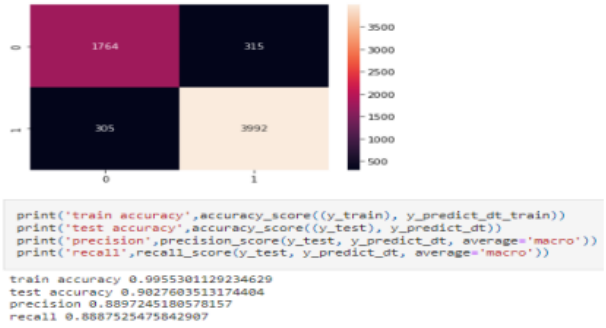
```
accuracy 0.8815391879131256
```

Evaluate

After analyzing both decision tree and Random Forest classifier, we have seen that Random Forest performed better than Decision tree with a rate worth 5% in Random Forest. Subsequently, we chose Random Forest for the classification of our given dataset for both categories Luxury Beauty Products and Prime Pantry Products.

Decision Tree for Product Category

The results we got from the implementation of decision tree on the product categories, the results we got for are accuracy is around 89% and precision is 83% and for both categories luxury Beauty Products and Prime Pantry Products



Decision Tree confusion Matrix for Product Category

Random Forest Tree for Product Category

The results we got from the implementation of Random forest algorithm on the product categories, the results we got for are accuracy is around 93% and precision is 94% and for both categories luxury Beauty Products and Prime Pantry Products

Random Forest confusion Matrix for Product Category

DISCUSSION AND FUTURE WORK

In our review, we used two algorithms, Random Forest and Decision tree. The execution of Random Forest performed admirably good as it gave an improved precision for both sections. As the models were executed on PCs with low equipment. The dataset contained five columns and, 28153 records. For further testing and improving the performance, we can use the high number of records for testing and training the model. As we have only tested 2 ML models, we can use other models such as neural networks for better implementation and better performance.

Conclusion

We have implemented ML Test Classification on Amazon reviews. we are supposed to predict the number of stars associated with the product reviews and also differentiate between the categories we have ”luxury Beauty Products” and ”Prime Pantry Products”. As per the results, we have seen that Random forest performed much better than the decision tree, we got the accuracy of 93% and 88% for Random Forest in both predicting reviews and differentiate categories respectively. Overall performance is 11% better for prediction using Random Forest.

References

[1] NLP Academy 2020. “What is NLP?”, Available at:https://www.nlpacademy.co.uk/what_is_nlp 26/6/2022]

[2] Cem Dilmegani JANUARY 23, 2021. ”Top 9 Ethical Dilemmas of AI and How to Navigate Them”. Available at:https://research.aimultiple.com/ai-ethics/. [Accessed: 9/7/2022]

[3] MindTools, 2022. ”Decision Trees”. Available at:https://www.mindtools.com/dectree.html. [Accessed: 9/7/2022]

[4] Gad, A.F., ”Evaluating Deep Learning Models: The Confusion Matrix, Accuracy, Precision, and Recall”, blog.paperspace.com, Available at: https://blog.paperspace.com/deep-learning-metrics/precision-recall-accuracy/ [Accessed: 9/7/2022]