****

**Project Report – Women’s E-Commerce Clothing Review Analysis**

Course: ALY 6020 Predictive Analytics

**Instructor: Dr. Marco Montes De Oca**

Submitted By,

Ashlesha Kshirsagar

Shruti Avinash Pawar

26 June 2020

**Abstract**

Recent increase in the base of ecommerce vendors has made the women clothing ecommerce industry highly competitive. Thus, customer reviews received on the websites for the products are used for text mining and building ML models. Our objective is to work on such data to build classification models from features extracted using the review text. We have therefore selected a data set from Kaggle known as Women’s Ecommerce Clothing reviews

**Content**

1.Introduction………………………………………………………………………………….......3

2. Exploratory Data Analysis......................................................................................................

3. Natural Language Processing for feature extraction…………………………………

4 Model Building ………………………………………………………….6

4.1.K-nearest Neighbors classifier:

4.2. Logistic Regression Model

4.3. Random forest

4.4 Adaptive Boosting

4.5. Gaussian Naïve Bayes

4.6 Comparison of Model accuracies

5.Model Tuning

5.1Tuning of RF

5.2Tuning Multinomial Naive Bayes

6.Conclusion

7.Challenges faced during the implementation of the project

8.Reference

**Figure/Table Index**

Figure 1: Recommended Yes/No plot with bars representing count of reviewers

Figure 2: Recommended Yes/No plot with bars representing count of ratings

Figure 3: Bar Chart representing Department recommending ‘Yes’

Figure 4: Rating Distribution among different Class names.

Figure 5: Frequency Count based on Division and Department

Figure 6: Frequency Count of different Class names.

Figure 7: Bar Chart Representing the Number of Reviews per Age range

Figure 8: Percentage of positive feedback by percent reviews

Figure 9: Distribution of Review Text

Figure 10: Bar plot for count of Exclamations in reviews

Figure 11: Distribution of polarity

Figure 12: Words used by reviewers who recommended the products

Figure 13: Words used by reviewers who did not recommended the products

Figure 14: Roc Curve – Logistic regression

Figure 15: Roc Curve – Random Forest

Figure 16: Roc Curve – AdaBoost

Figure 17: Roc Curve – Gaussian Naïve Bayes

Figure 18 : Roc Curve – Random Forest

Figure 19: Roc Curve – Multinomial navie bayes

Table 1 : Comaprision of different Model accuracies

Table 2 : Comparing accuracies for Random Forest and Naïve bayes model

**1.Introduction:**

Understanding customer sentiments are of great importance in retailing strategies today. Not only will it give companies an insight as to how customers perceive their products and services, but it will also provide them with an idea on how to improve their offers (Abien Fred ,2018). With this project we are trying to analyze reviews of clothes purchased by women and build a model to predict if the product will be recommended or not based on the reviews received. In this week we have performed 2 important steps which are exploratory data analysis and NLP post which we have built a logistic regression model.

EDA is the practice of representing the data utilizing statistical and visualization techniques to bring important aspects of that data into focus for further analysis. This involves looking at your data set from many angles, describing it, and summarizing it without making any assumptions about its contents. This is a significant step to take before diving into machine learning or statistical modelling, to make sure the data are really what they are claimed to be and that there are no apparent problems.**We have explored different visualizations to understand the data ( Chloe, 2017).**

Machine learning for NLP and text analytics involves a set of statistical techniques for identifying parts of speech, entities, sentiment, and other aspects of text. It involves analyzing the review text to understand what is expressed by the person writing it. The sentiment in the text is calculated using polarity which tags the comment as positive or negative (Sarkar, 2018).

Project link: <https://github.com/ashlesha10/Women-s-E-Commerce-Clothing-Reviews/tree/Final>

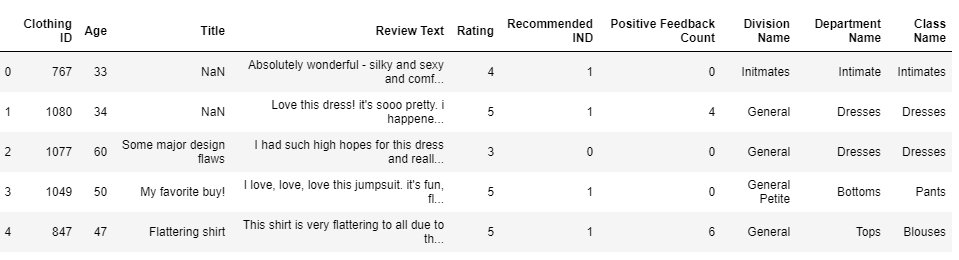
Dataset link: <https://www.kaggle.com/nicapotato/womens-ecommerce-clothing-reviews>



**2. Exploratory Data Analysis**

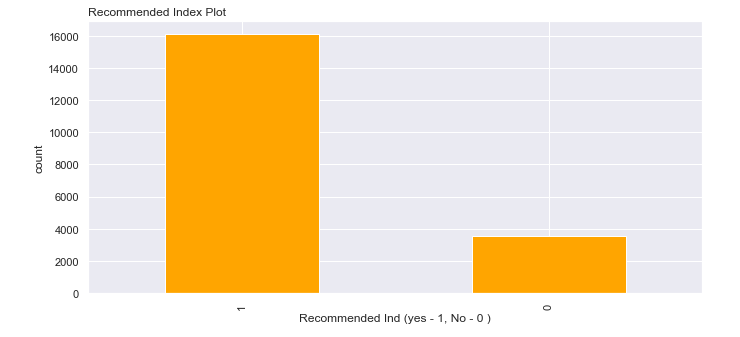
The data consists of 23486 rows and 10 variable columns such as age, rating, recommended IND, review text, etc. The tool used to process and analyze this acquired data is ‘python’ and the packages used are NumPy, Pandas, Scikit learn, Matplotlib, Seaborn, Textblob, and NLTK.

The first step of the review analysis process was data cleaning and EDA. The data was extracted from a csv file to a pandas data frame and below was the output in python:



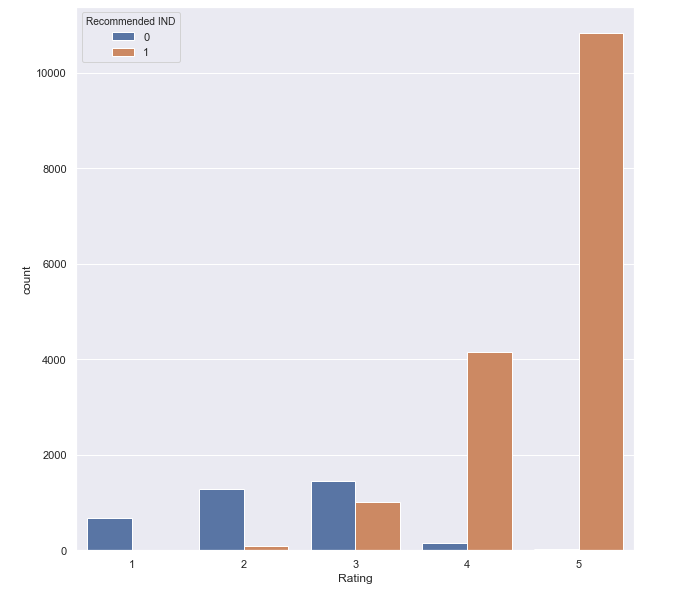
We used ‘isnull function’ and found that there are 3810 null values in Title and 845 null values in review text. These null values will add no value to the analysis; thus, we have excluded those from the data frame. This leaves 19662 rows available in the data frame for analysis. Below is the snip of the code and output:

Further, to understand the data variables, we performed Exploratory Data Analysis and Visualization of the data. A bar plot was created using matplotlib and below was the output:



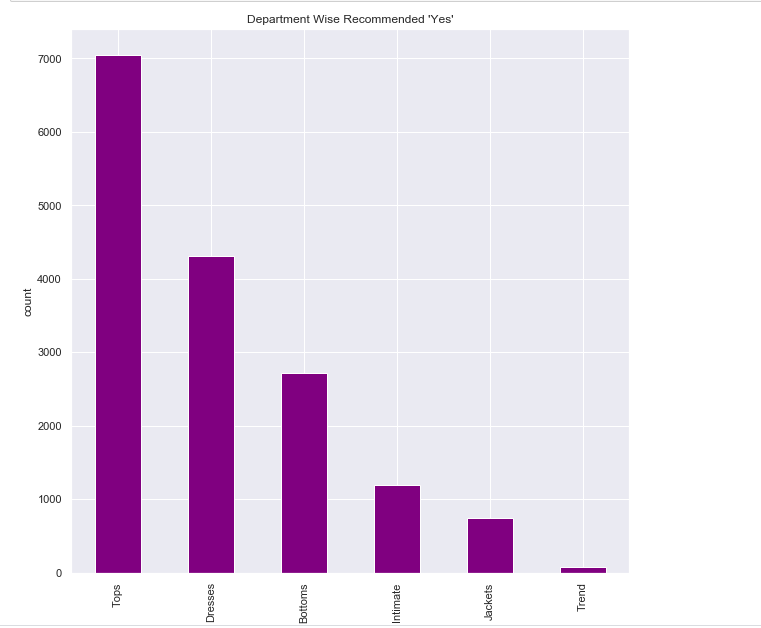
**Figure 1:** Recommended Yes/No plot with bars representing count of reviewers

Figure 1 shows that there are more than 15000 reviewers who have recommended the product and about 4000 have not recommended the product.



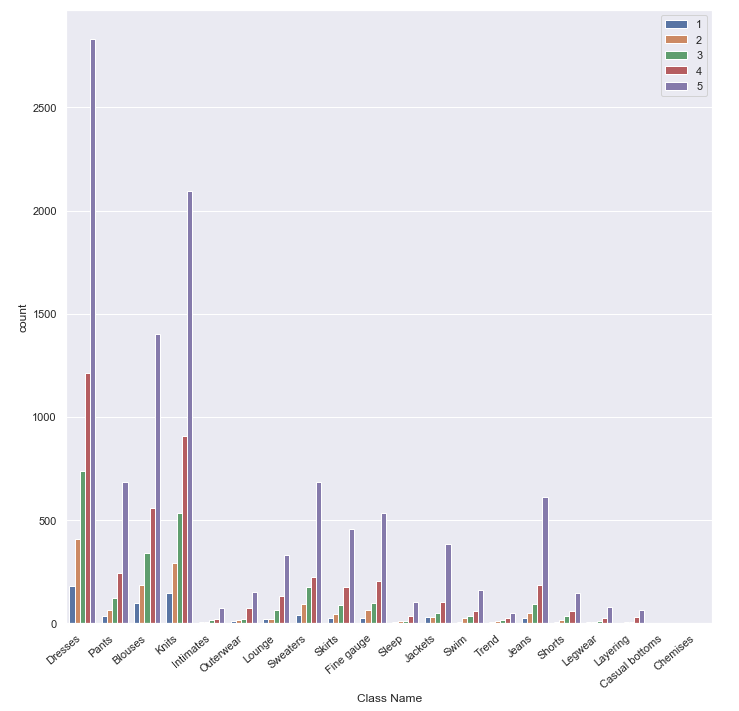
**Figure 2:** Recommended Yes/No plot with bars representing count of ratings

In figure 2, we see that the majority of the reviews were positive with 77% at 4 or higher and 89% of reviews over 3. 3 would normally be the cutoff for an acceptable review while 2 or lower would be considered poor. However, it is interesting to see the high amount of non-recommended product recommendations at 3 with some even present at 4.



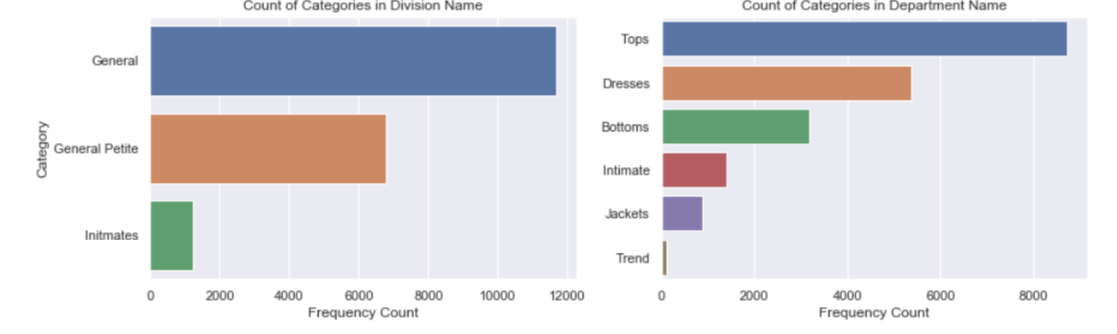
**Figure 3:** Bar Chart representing Department recommending ‘Yes’

From figure 3 we can see that maximum recommendations are given to the Tops, Dresses than another department.



**Figure 4:** Rating Distribution among different Class names.

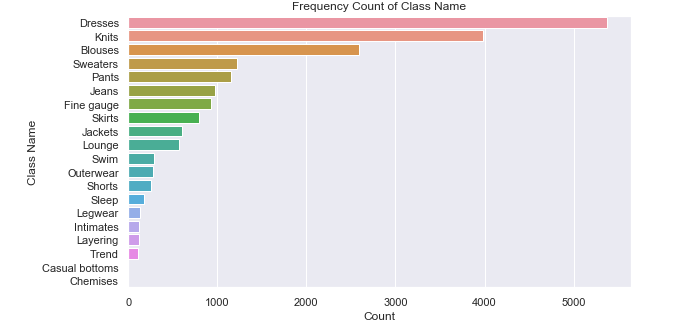
Figure 4 helps us to understnd the distribution of rating for each class.We can see that the maximum ratings given to the Dresses , Pants, blouses, knits.There could be probability that the maximum purchase happened in this classes as well.



**Figure 5:** Frequency Count based on Division and Department

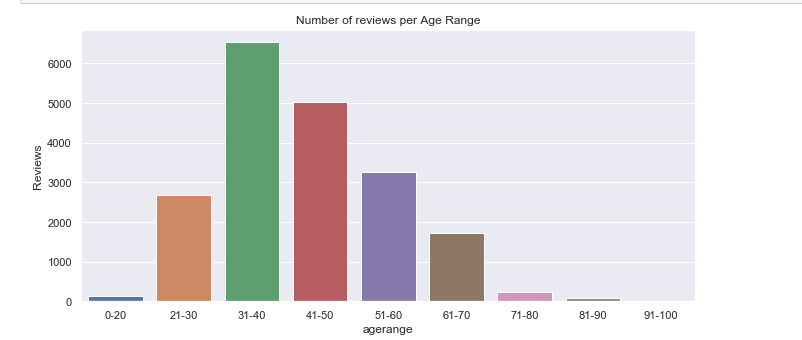
.

Above figure shows the distribution in three categories: General, Petite, and Intimates. This offers some insight into the clothing sizes of the customers leaving reviews. It is notable to observe that Tops and Dresses are the most commonly reviewed products.



**Figure 6:** Frequency Count of different Class names.

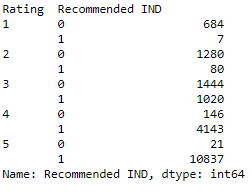
In figure 6 we can see the frequency count for different class names. Exploring the class variable suggests that the most popular clothing types are: Petite and Anthro, Dresses, Blouses, and Cut and Sew Knits. The distribution of reviews is fairly constant, suggesting that there are not negative nor positive outliers.



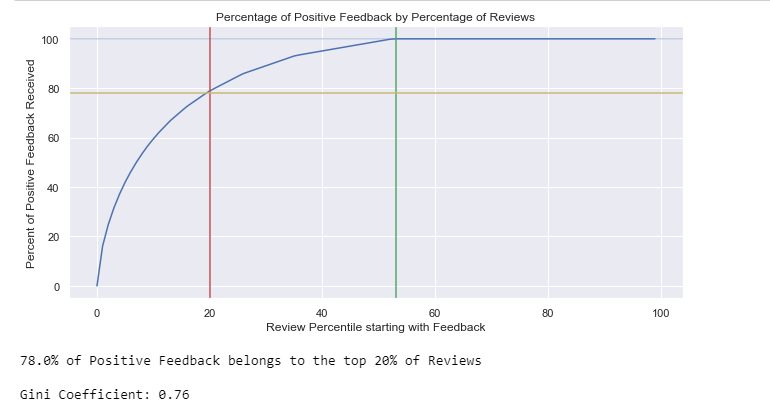
**Figure 7:** Bar Chart Representing the Number of Reviews per Age range

.

From figure 7,it is seen that majority of the reviews are given by women in the age class of 31 to 40 followed by 41 to 50 and 51 to 60. It is very surprising that the age class 21 to 30 which is supposedly considered to be active on social media, have given lesser reviews as compared to the other age classes. . We now took a rating wise count of recommendations and found the below output:



A high number of reviews with high product ratings is justified however 7 people who have rated the product 1 and 80 who have rated 2 have recommended the product.



**Figure 8**: Percentage of positive feedback by percent reviews

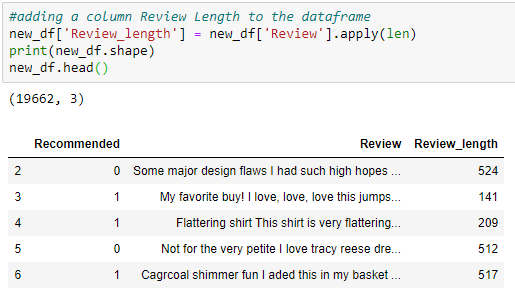
A **Gini coefficient** can be used to evaluate the **performance** of a classifier. From figure 7, we can say that the 78 % of positive feedback belongs to the top 20 % of reviews.

**3.Natural Language Processing for feature extraction:**

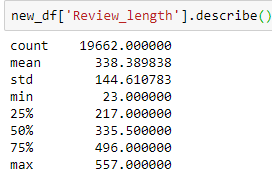
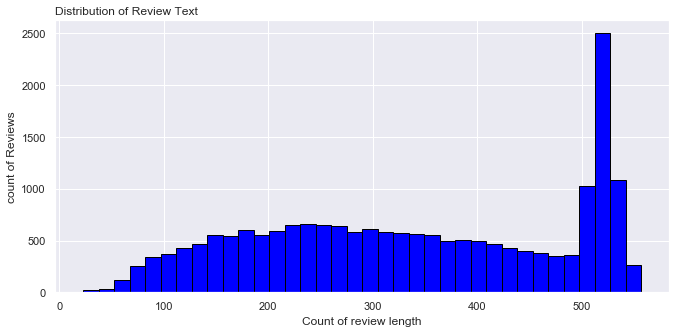
To extract features, we created a new dataframe with variables ‘recommended Ind’ which is our target variable and columns named asTitle and review text are merged into one to collate all the text comments in one variable. The dataframe looks like the snip below.



Further, we added a feature ‘ review\_length’ to the dataframe. The code provided below calculated the character length of the review. The reason to choose this feature is to evaluate the efforts taken by a reviewer to write about the product and its impact on our target variable ‘ recommended’

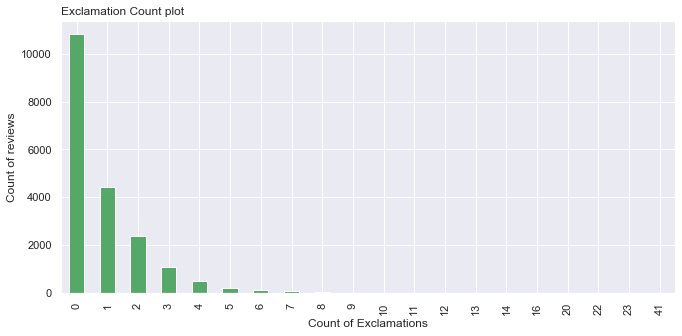


If we take a descriptive summary of the Review\_Length variable, it is observed that on an average, a reviewer uses 335 characters and a maximum of 557 character to write reviews about the product.

**Figure 9:** Distribution of Review Text

We can see from figure 9 that the distribution of the review length is skewed with more number of reviewers writing lengthy reviews. The next variable that we have added is the count of exclamations in the review. Exclamations are used as an expression of excitement, admiration or surprise. Thus we have selected this variable to consider the impact of these expressions on our target variable. In figure 9 below, it is seen that more than 10000 reviewers have not used exclamation and have used just words to write their reviews. There are 4000+ reviewers who have used 1 exclamation to express their words.

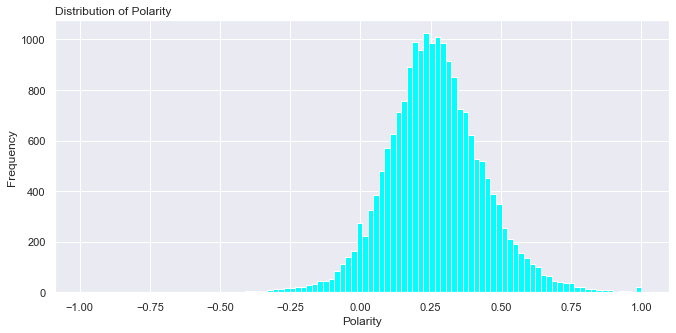


**Figure 10:** Bar plot for count of Exclamations in reviews

The third feature that is added to the data is Polarity. Polarity is a value that falls between -1 and 1, with 1 being a strongly positive statement and -1 being strongly negative statement. (Jain, 2018). When we calculated the polarity, we saw reviews that had words like ‘not’,’cheap’, ‘issues’ etc had a negative polarity whereas reviews with word ‘love’ had a polarity of 0.5 or more. Refer to the snip of the data frame with the polarity feature added on page 9.

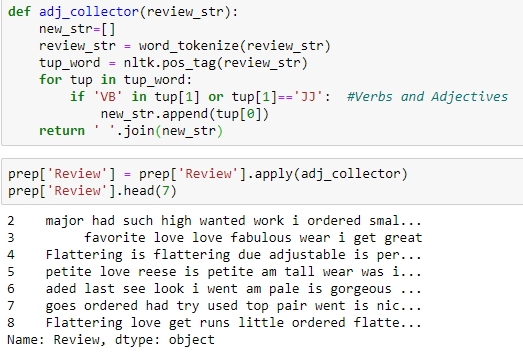


In figure 11 below, we can see that the distribution looks like a normal distribution with most of the reviews on the positive end.

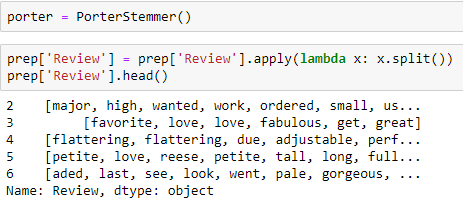


**Figure 11:** Distribution of polarity

To further extract numeric features from the review text, we removed all the punctuations from the review column. We then used the below function to collect adjectives and verbs from the review text.



In the above snip, if we notice, the adjective collector has picked certain words like ‘was’, ‘I’, ‘had’,etc that we need to remove from the text. These words are called as stop words and we have removed the same from the review text. Now, the review text variable contains only adjectives and verbs. We have also removed punctuations from the text. The next step performed is stemming. Stemming is a process carried out to reduce redundancies by tagging all the related words to the base word. For example, programmer, programs, programming will all be tagged to the base word program. ("Python | Stemming words with NLTK - GeeksforGeeks", 2020). Below is the code and output of stemming:



To visualize the words used in negative reviews and positive reviews, we used wordcloud from python. The data frame was split into two subsets with Recommended 0 as negative and 1 as positive. The wordcloud appears for both positive and negative reviews appear as follows:



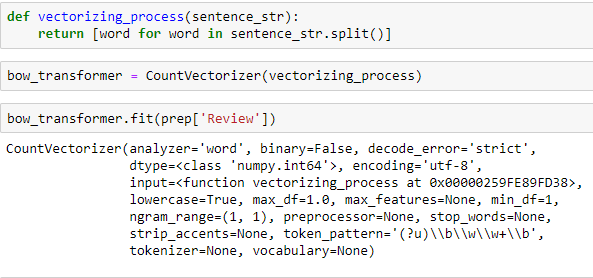
**Figure 12:** Words used by reviewers who recommended the products

From figure 12, we can identify that the reviewers who recommended the product have frequently used words like ‘love’, ‘great’, ‘perfect’, etc.

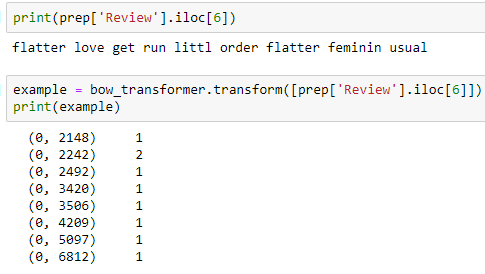
****

**Figure 13:** Words used by reviewers who did not recommended the products

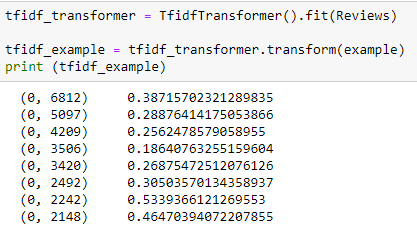
Figure 13 emphasizes on words ‘return’, ‘disappoint’, ‘fit’, etc. We can also see words like ‘love’ and ‘flatter’. This could be possibly because the reviewer liked the product and gave good reviews however did not recommend the product.



Turing the entire process to a numeric front, we start vectorizing the words. Bag of words is a phrase that denotes a bunch of words in a set. Here, we will refer the set as a review. With the help of bow transformer, we will be creating a corpus and then we are tokenizing the text and building a vocabulary of words in total. Hence, we see if we lock the 6th review, we get the details of a corpus. In the snip on page 12, we see what is happening in the 6th index of the data. The words are converted to index and their frequency is updated next to that index. Index of flatter is 2242 and it is repeated 2 times.



We are creating an example of the locked review and vectorizing the string set based on bash numbers, the right-side column denotes the frequency in the string set. Furthermore, we are applying the transformer in the review’s column. and after updating the values, we are using sparse matrix to store elements.



TF-IDF, Term Frequency Inverse Document Frequency, simply states that a particular word occurring in a statement water review upon the number of words in a string set multiplied by that particular word occurring in the entire document set that we have; basically, the collection of reviews. This Transformer actually converts a vectorized value of the corpus into a bash number giving it a unique value. The statistical explanation is explained as follows:

**A picture containing knife

Description automatically generated** (Scott, 2019).

This bash numbers are unique to every word in the review and can be looked up using a bow transformer vocabulary. Once the bash number is identified using the TF-IDF method, we assign the transformer to the entire ‘review’ labeled column.

**A screenshot of a social media post

Description automatically generated**

Once, this is done, we will be merging the sparse matrix with other variables to form a matrix containing all bash values of these vectorizing corpus.

**A screenshot of a cell phone

Description automatically generated**

Finally, we have all the categorical values of the Review column in numerical form.

Now, we will be merging these values into the original data set. A screenshot of a social media post

Description automatically generated

Now, as we have merged these data sets together. Now we split the data into two sets. One containing the ‘recommendations’ column and the other containing the entire data set.

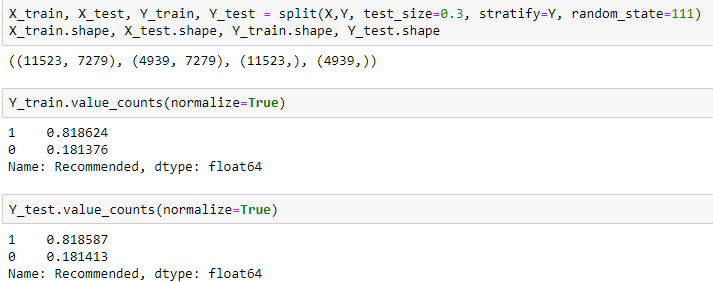
**A screenshot of a cell phone

Description automatically generated**

**A screenshot of a cell phone

Description automatically generated**

We now have the entire data set apart from the recommended column. After describing this, we get the mean and maximum value of the data set, which can be comparable to the previous value.



We are now dividing the data set into two columns the train and the test data set.

**4. Model Building**

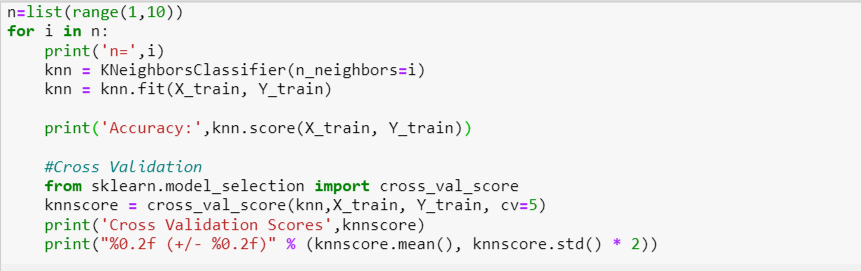
We divided our data set into train and test dataset in the 70-30 ratio respectively. We build different classification models on the dataset.

1. **K-nearest neighbors**
2. Logistic regression
3. Random Forest
4. Adaptive Boosting
5. Navie bayes – Gaussian

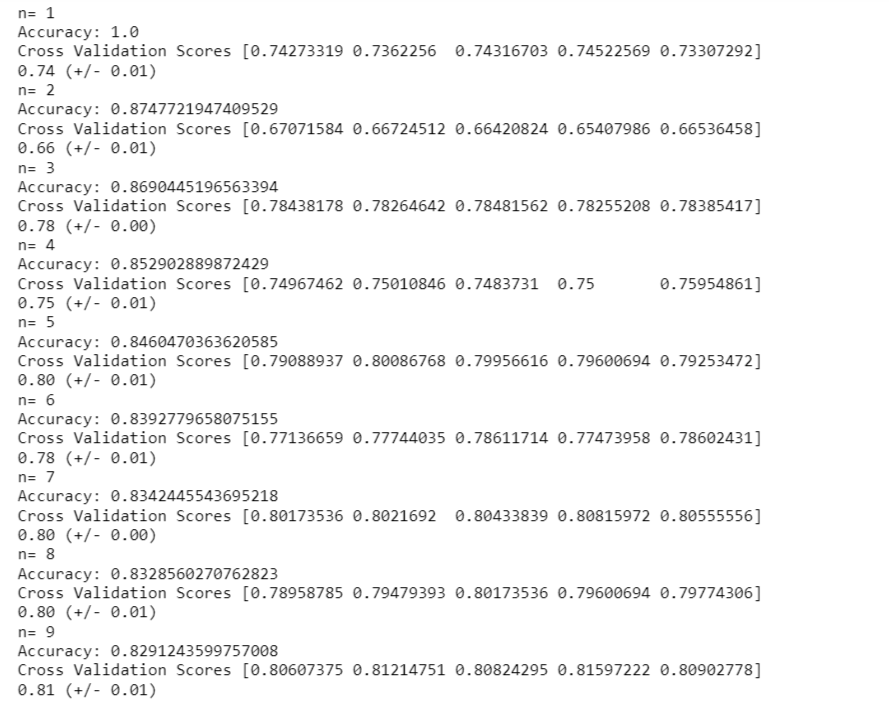
**4.1.K-nearest Neighbors classifier:**

The k-nearest neighbors’ classifier (k-NN) is a classical supervised method based on statistical data, which maps any feature vector X to the pattern class that appears most frequently among the k-nearest neighbors. The performance of a classifier depends on the interrelationship between sample size, number of features, and classifier complexity. Estimating the accuracy of a classifier is important not only to predict its future prediction accuracy, but also for choosing a good classifier from a given set. (Max Kuhn and Kjell )In training phases of those supervised classifiers, accuracy estimation is helpful for adjusting partial parameters until achieving expected accuracy rate. The accuracy of a classifier is the probability of correctly classifying a random or selected instance. Usually, the absolute accuracy is unknown, cannot be calculated, and must be estimated from given datasets.

we define our classifer, in this case KNN, fit it to our training data and evaluate its accuracy. We’ll be using an arbitrary K but we will see later on how cross validation can be used to find its optimal value.

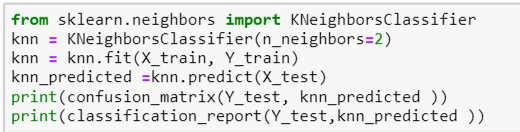


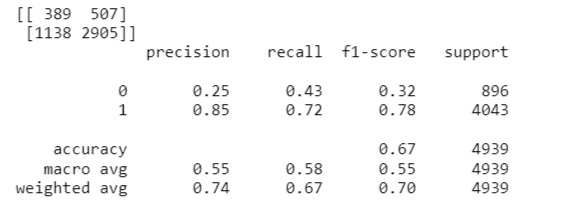
Cross-Validation is used for evaluating predictive models by partitioning the original sample into a training set to train the model, and a test set to evaluate it.



In k-fold cross-validation, the original sample is randomly partitioned into k equal size subsamples.scikit-learn comes in handy with its Cross\_val\_score  method. We specify that we are performing 5 folds with the CV =5 parameter and that our scoring metric should be accuracy since we are in a classification setting. From output we got the maximum accuracy at K = 2 which is 87.47%.and accuracy got decreased afterwards due to under-fitting

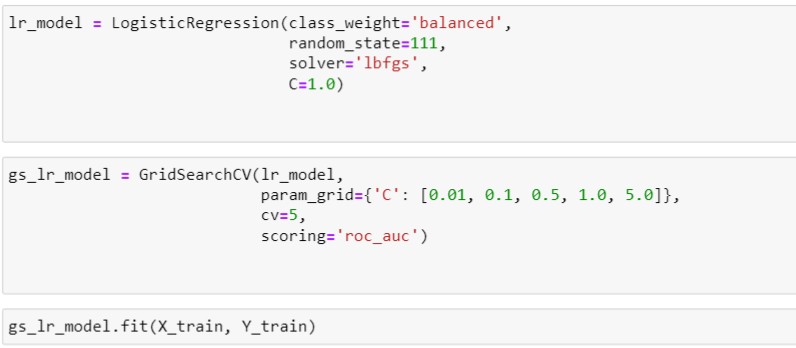
We calculated accuracy for K = 2 and predicted the output as below





**4.2. Logistic Regression Model**

Logistic regression is a predictive analysis.it is used to explain the relationship between one dependent binary variable and one or more nominal, ordinal, interval or ratio-level independent variables. it's usually recommended to choose the learning rate that better minimizes the cost function.we check the accuracy for different learning rate.



The best Learning rate we got is 0.1.



Using the learning rate 0.1 we again run the model and below are the results we obtained.The model accuracy is 70% and AUC is 0.78.

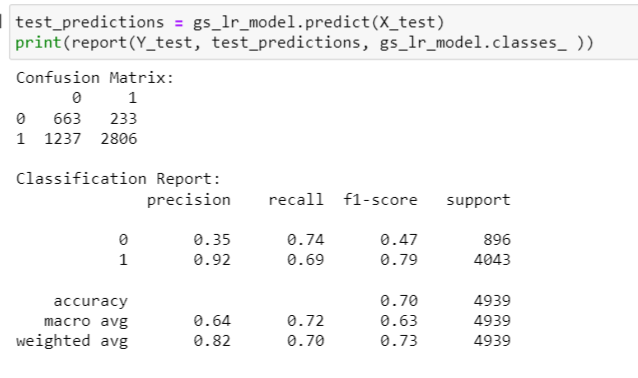
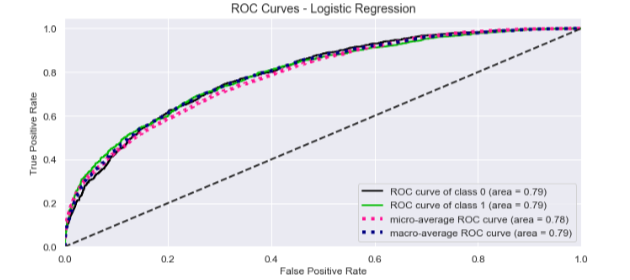
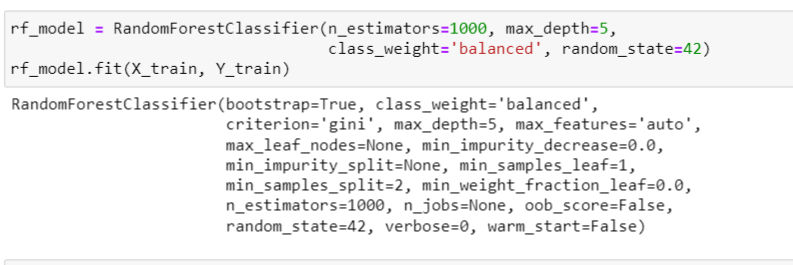
 

Figure 14: Roc Curve – Logistic regression

**4.3. Random forest**

Random forest is a group of decision trees. These decision trees are fit on different training dataset and the process of picking the dataset is known as sampling with replacement. The output of the random forest model is an average of all the group decision tree output which reduces the bias. However, deciding the optimum number of trees is an important task as higher number of trees would result into higher sample repetitions which also have the potential of causing bias. The number of trees is a hyper parameter in Random forest model, and it can be tuned using cross validation to determine the number. (Brownlee, 2020)

We continued with the train and test dataset to fit a Random forest model at three different n\_estimators (tree size) 100, 500 and 1000. 

After checking the accuracies for different estimators values, we got maximum accuracy of 76% for 1000 estimator value.

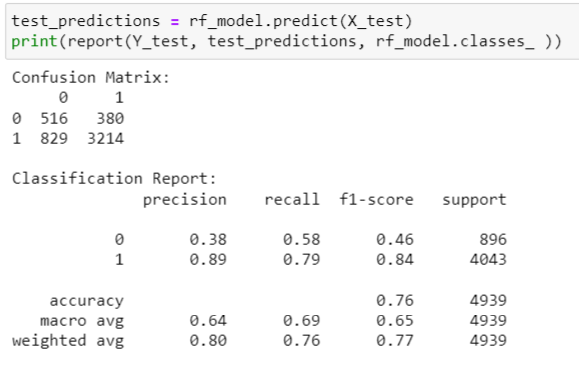
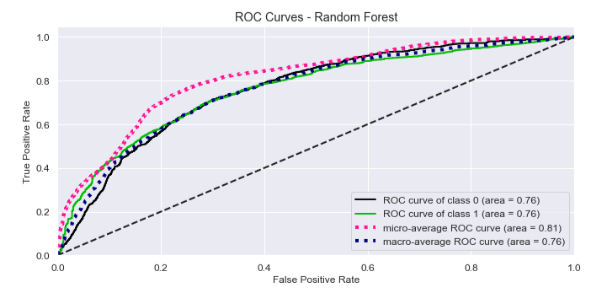
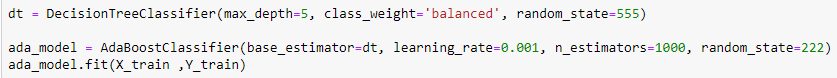
 

Figure 15: Roc Curve – Random Forest

**4.4 Adaptive Boosting**

Ada boost models are built on decision stumps (decision trees with just binary classification of two classes). Ada boost algorithm is another ensemble technique like random forest which creates a strong classifier learning from multiple weak classifiers (Brownlee, 2016). We have used the below code to fit Ada model on the data



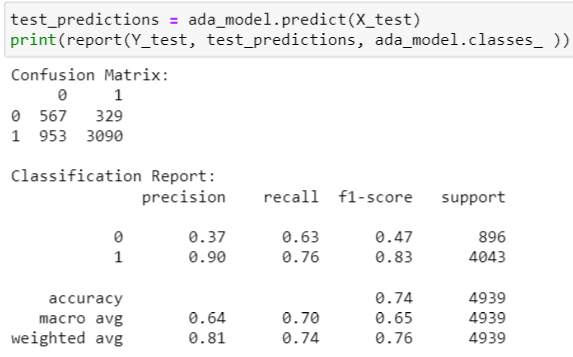
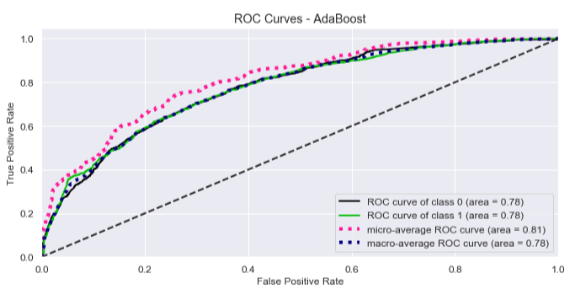
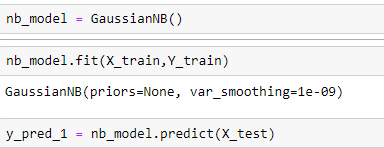
 

Figure 16: Roc Curve – AdaBoost

To check how Ada boost will work on our data set, we kept the tree size to 1000 and learning rate at 0.001. The code took a long time to run as the learning rate was low. The prediction on the test dataset gave an accuracy of 74% and AUC was 0.81 as seen in the figure above

**4.5. Gaussian Naïve Bayes**

The next model we chose to fit is Gaussian Naïve Bayes. This is because the predictors in the Gaussian Naïve Bayes take up a continuous value and the feature values extracted from NLP also take a continuous value (Gandhi,2018). We have used below code to fit Naïve Bayes model:



The predictions determined from the model had an accuracy of only 35%. Looking at the low accuracy, we realized that although the values of the features extracted were continuous, they were in the form of a sparse matrix due to which there were a lot if zeros in our Dataframe. The model must have interpreted these zeros as discrete variables and Gaussian works best on continuous variables.

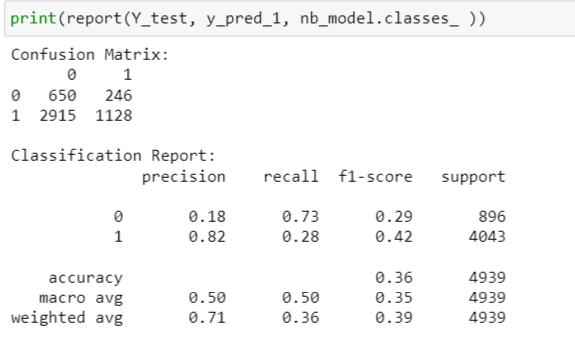
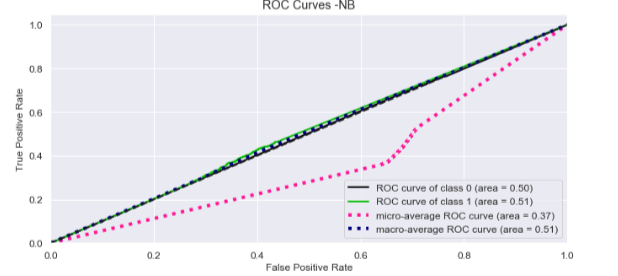
 

Figure 17: Roc Curve – Gaussian Naïve Bayes

**4.6 Comparison of Model accuracies**

We have compared the accuracies of different models. Random Forest and Adaptive Boosting has higher accuracies as compared to other models.

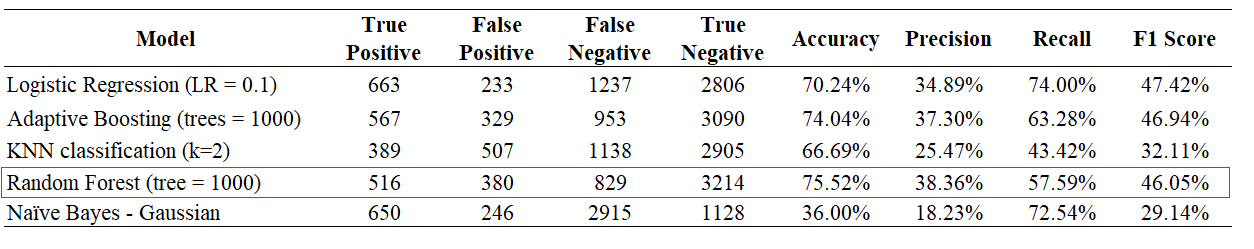


Table 1 : Comaprision of different Model accuracies

**5.Model Tuning**

Random forest model had the highest accuracy, it had precision and recall similar to other models and an AUC of 0.81. Thus, we selected this model to increase the accuracy. We also apply multinomial naïve bayes and check the accuracy of the model.

**5.1Tuning of RF**

While building the model we tried increasing and decreasing the tree size and tree depths which are hyperparameters of the model. With trial and test such as 100 tree size or 500 tree size, we found that these changes were only boosting the accuracy by 1 or 2%. Thus, we decided to add more features to the training data. Earlier the dataset only consisted of features related to the text such as review text length, polarity and the TF-IDF sparse matrix. In the new Dataframe, two features age and rating were added. Age would give weightage to the reviewer and rating would account for how satisfied the reviewer is with the product. By adding just these two features to the earlier Dataframe, we run the same Random Forest model on the new training data with a tree size 500 and maximum dept of 5. The accuracy drastically increased from 76% to 93% with AUC at 0.96.

[**Complete code for Random Forest Tuning can be found here:**](https://github.com/ashlesha10/Women-s-E-Commerce-Clothing-Reviews/blob/master/Final%20Code/ALY6020_FinalProhec_Ashlesh_Shruti_Part2.ipynb)

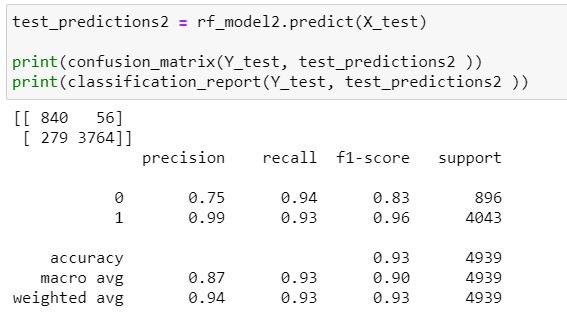
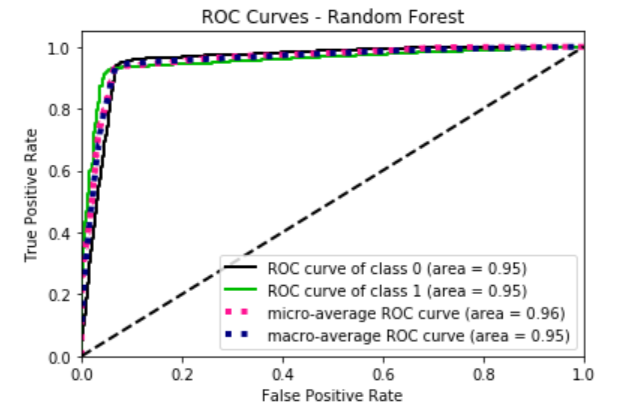
 

Figure 18 : Roc Curve – Random Forest

**5.2.Tuning Multinomial Naive Bayes**

The multinomial Naive Bayes classifier is suitable for classification with discrete features. Feature engineering is a critical step when applying Naive Bayes classifier. We only use Review Text and Recommended IND as a predictor and target class variable and dropped all the other features in the data. Review Text Contains review from customers and will be used as a predictor variable. And Recommended IND Contains recommendation from customers, whether the product recommended or not, will be used as the target variable.So from below roc curve and table showing the results, we can say that Model has an excellent performance with accuracy equal to 89%. Recall for "Recommended" values as positive target class have value equal to 0.95, this means that "Recommended" review text has 95% predicted correctly. Model have RUC curve that closer to 1, and this means that Model has a good performance in True Positive Rate.

[**Complete code for Multinomial Naïve bayes can be found here**](https://github.com/ashlesha10/Women-s-E-Commerce-Clothing-Reviews/blob/Final/Final%20Code/ALY6020_FinalProhec_Ashlesh_Shruti_Part3.ipynb)

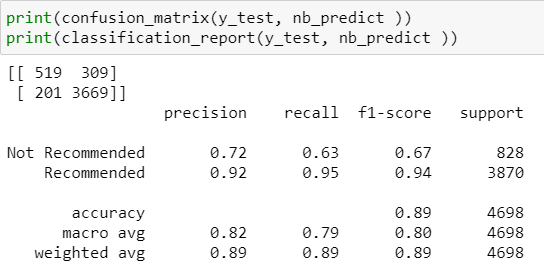
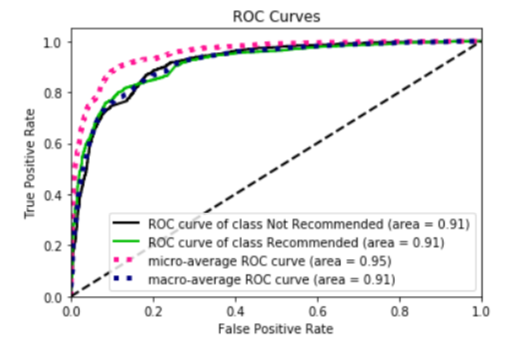
 

Figure 19: Roc Curve – Multinomial navie bayes

**6.Conclusion**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Model** | **True Positive** | **False Positive** | **False Negative** | **True Negative** | **Accuracy** | **Precision** | **Recall** | **F1 Score** |
| Naïve Bayes - Multinomial | 519 | 309 | 201 | 3669 | 89.14% | 72.08% | 62.68% | 67.05% |
| Random Forest | 840 | 56 | 279 | 3764 | 93.22% | 75.07% | 93.75% | 83.37% |

Table 2 : Comparing accuracies for Random Forest and Naïve bayes model

Both the model has ROC curve that closer to 1; this means that model has a good performance in True Positive Rate.

Product recommendation and Product rating are used for different purposes.

* Recommended is a strong indicator for positive sentiment in the review.
* Rating is more convoluted, where rating around 3 are hopeful reviews with constructive criticism of the product.

The females from 20-50 age were more active and bought the stuff online and were more focussed on Tops and Dresses department. And, somewhat focused on Bottoms too but not that much. They were less concentrated on Trend department.

**7.Challenges faced during the implementation of the project**

1.while processing NLP, the biggest challenge we faced in creating an adjective collector. The Functions took a long time to run.

2. Since the data was huge, processing time was relatively high, whenever we made changes or run new code, the Jupyter notebook took a long time to process/run. We also faced memory issues.

3. We have used several libraries for NLP, implementation of algorithms in python. Installing such libraries was a bit challenging initially. We used Conda navigator to install or update the libraries.

4. We have several challenges in model tuning. Starting from finding the best values for K, learning rate, tree size, n estimator. Through trial and errors, we were able to find suitable values for each Model.

5. The processing time for adaptive boosting, KNN was relatively high than other models. How to reduce the time complexity for such an algorithm is our future step of the project.

**8.Reference**

1. Reference: Brownlee, J. (2016). Boosting and AdaBoost for Machine Learning. Retrieved 26 June 2020, from <https://machinelearningmastery.com/boosting-and-adaboost-for-machine-learning/>
2. Gandhi, R. (2018). Naive Bayes Classifier. Retrieved 26 June 2020, from <https://towardsdatascience.com/naive-bayes-classifier-81d512f50a7c>
3. Max Kuhn and Kjell Johnson (2013), Applied Predictive Modeling
4. Ritchie Ng K-nearest Neighbors (KNN) Classification Model Retrieved from <https://www.ritchieng.com/machine-learning-k-nearest-neighbors-knn/>
5. Brownlee, J. (2020). How to Develop a Random Forest Ensemble in Python. Retrieved 12 June 2020, from <https://machinelearningmastery.com/random-forest-ensemble-in-python/>
6. CHLOE MAWER, MARCH 9TH, 2017,The Value of Exploratory Data Analysis,retrived from <https://www.svds.com/value-exploratory-data-analysis/>
7. Abien Fred Agarap,March 2018,Statistical Analysis on E-Commerce Reviews, with Sentiment Classification using Bidirectional Recurrent Neural Network, 10.13140/RG.2.2.16988.08321
8. Sarkar, D. (2018). A Practitioner's Guide to Natural Language Processing (Part I) — Processing & Understanding Text. Retrieved 15 May 2020, from <https://towardsdatascience.com/a-practitioners-guide-to-natural-language-processing-part-i-processing-understanding-text-9f4abfd13e72>
9. Jain, S. (2018). Natural Language Processing for Beginners: Using TextBlob. Retrieved 4 June2020, from <https://www.analyticsvidhya.com/blog/2018/02/natural-language-processing-for-beginners-using-textblob/>
10. Python | Stemming words with NLTK - GeeksforGeeks. (2020). Retrieved 4 June 2020, from <https://www.geeksforgeeks.org/python-stemming-words-with-nltk/>
11. Scott, W. (2019). TF-IDF for Document Ranking from scratch in python on real world dataset. Retrieved 4 June 2020, from <https://towardsdatascience.com/tf-idf-for-document-ranking-from-scratch-in-python-on-real-world-dataset-796d339a4089>
12. Sarkar, D. (2018). A Practitioner's Guide to Natural Language Processing (Part I) — Processing & Understanding Text. Retrieved 15 May 2020, from <https://towardsdatascience.com/a-practitioners-guide-to-natural-language-processing-part-i-processing-understanding-text-9f4abfd13e72>
13. Accuracy, Precision, Recall & F1 Score: Interpretation of Performance Measures - Exsilio Blog. (2016). Retrieved 16 May 2020, from <https://blog.exsilio.com/all/accuracy-precision-recall-f1-score-interpretation-of-performance-measures/>
14. Schoonjans, F. (2020). ROC curve analysis with MedCalc. Retrieved 16 May 2020, from <https://www.medcalc.org/manual/roc-curves.php>