RATING PREDICTIONS FOR ECOMMERCE’s REVIEWS

Group Members:

1. Robin Manchanda (101374905)
2. Bert Aguiar (101377891)
3. Faiz Mohammed Radhanpuri (101369631)
4. Pratiksha Murkey (101373484)

Table of Contents

[Summary: 4](#_Toc84365882)

[The Problem Statement: 4](#_Toc84365883)

[The Dataset 4](#_Toc84365884)

[EDA/Preliminary Analysis Report 5](#_Toc84365885)

[Checking Null Values 5](#_Toc84365886)

[Target Label Distribution 6](#_Toc84365887)

[Univariate analysis 7](#_Toc84365888)

[Feature selected- ['Recommended IND'] 7](#_Toc84365889)

[Feature Selected [‘Age’] 7](#_Toc84365890)

[Feature Selected [‘positive Feedback count’] 8](#_Toc84365891)

[Bivariate analysis: 8](#_Toc84365892)

[Recommend ID and Rating (Target variable) 9](#_Toc84365893)

[Positive Feedback Count and Rating (Target variable) 9](#_Toc84365894)

[Divisions and rating 10](#_Toc84365895)

[Departments and rating 10](#_Toc84365896)

[Word Cloud: 11](#_Toc84365897)

[For Rating 5 11](#_Toc84365898)

[For Rating 1 11](#_Toc84365899)

[Feature Engineering: 12](#_Toc84365900)

[Review\_text\_len 12](#_Toc84365901)

[Review\_text\_len and Rating 12](#_Toc84365902)

[Data Preprocessing 12](#_Toc84365903)

[Strategy to handle Null Values: 13](#_Toc84365904)

[One-hot encoding 13](#_Toc84365905)

[Min-Max Scaling 14](#_Toc84365906)

[Text Preprocessing 14](#_Toc84365907)

[Removing Stop-words: 15](#_Toc84365908)

[Used Regular Expression: 15](#_Toc84365909)

[Lemmatization: 15](#_Toc84365910)

[BOW / TF-IDF: 16](#_Toc84365911)

[Bag Of Words: 16](#_Toc84365912)

[TF-IDF 17](#_Toc84365913)

[Splitting the Dataset: 17](#_Toc84365914)

[Modeling Approaches 19](#_Toc84365915)

[Machine Learning Models and hyperparameter tuning 19](#_Toc84365916)

[Logistic regression 19](#_Toc84365917)

[Support Vector Machines 19](#_Toc84365918)

[Random Forest 20](#_Toc84365919)

[Model Evaluation 21](#_Toc84365920)

[Evaluation Metrics 21](#_Toc84365921)

[Results & Discussion 23](#_Toc84365922)

[Conclusion 23](#_Toc84365923)

[References 23](#_Toc84365924)

|  |  |
| --- | --- |
| Name | Responsibility |
| Robin Manchanda | Modeling Approaches and Evaluation |
| Faiz Mohammed Radhanpuri | Data Preprocessing, PCA |
| Pratiksha Murkey | EDA |
| Bert Aguiar | Data Cleaning, Hyper Parameter Tuning |

The Division of Labour

# Summary:

Capturing ratings from the reviews is important in these times where a few words have the power to start a revolution. A review, once posted, can make, or break a product – it is important to learn from previous reviews and thus direct a customer’s attention accordingly. Evaluating ratings from customer’s reviews can benefit the organization in the following ways:

• Become more competitive

• Attract new customers

• Retain present customers

• Sell more products and services

• Reduce customer servicing

• Make customers more profitable

• Improve marketing messages and campaigns

In this project, we will sift through the reviews in our data set and attempt to gauge what sentiment the customer was expressing by trying to predict the rating a customer would give that product.

# The Problem Statement:

Our project will generate results/insights that may be feed into an e-commerce recommendation system. We are trying to predict Customer Ratings by analyzing customer data and reviews. Recommendation engines then can use this data along with other strategies to make product recommendations to customers. Negative Reviews are to be predicted as matches to the 1-star category and positive reviews as 5-stars.

# The Dataset

To explore the subject further, we chose an interesting dataset with from Kaggle which provided us with real-world data scrapped from a Women’s Clothing E-Commerce Website:

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

The data set consists of the following columns:

1. Clothing ID: Integer variable that refers to the specific piece being reviewed.
2. Age: Positive Integer variable of the reviewer's age.
3. Title: String variable for the title of the review.
4. Review Text: String variable for the review body.
5. Rating: Positive Ordinal Integer variable for the product score by the customer [1 Worst, to 5 Best]
6. Recommended IND: Binary variable stating where the customer recommends the product where 1 is recommended, 0 is not recommended.
7. Positive Feedback Count: Positive Integer documenting the number of other customers who found this review positive.
8. Division Name: Name of the product high-level division.
9. Department Name: Name of the product department name.
10. Class Name: Name of the product class name.

This data set was interesting because the most useful parts of the Data were text. We have to predict the rating using NLP Techniques.

EDA/Preliminary Analysis Report

EDA is primarily used to see what data can reveal beyond the formal modeling or hypothesis testing task. It provides a better understanding of data set’s variables and the relationships between them. The main purpose of EDA is to get a better look at the data before making any assumptions. It also takes advantage of several quantitative methods to describe the data.

### Checking Null Values

A picture containing diagram

Description automatically generated

Observations:

* Most of the missing values are in the Title and Review Text field.

### Target Label Distribution

* X-Axis – Rating (1 Star – 5 Stars): Target Label
* Y-Axis – (Count)

Chart

Description automatically generated with low confidence

Observations:

* The dataset is highly Imbalanced.
* It can be seen from the plot that the count of examples for rating 5 is more than 12000 (more than 50%), whereas only 1000 examples are there for a one-star rating.

## Univariate analysis

### Feature selected- ['Recommended IND']

Chart

Description automatically generated

Observations:

* Around 80% of the products are recommended by customers.

### Feature Selected [‘Age’]

Chart, histogram

Description automatically generated

Chart, box and whisker chart

Description automatically generated

Observations:

* Most of the customers are around 40 years old.
* 75% of the total customers are less than 52.

### Feature Selected [‘positive Feedback count’]

A picture containing histogram

Description automatically generated

Observations:

* Most of the products have zero positive feedback count.

## Bivariate analysis:

### Recommend ID and Rating (Target variable)

A picture containing bar chart

Description automatically generated

Observations:

* This plot shows that the product with higher recommendations tends to have more ratings.

### Positive Feedback Count and Rating (Target variable)

Chart, bar chart

Description automatically generated

Observations:

* It can be clearly seen from the plot that the products with higher positive feedback counts have mostly higher ratings

### Divisions and rating

Chart, bar chart

Description automatically generated

Observations:

* The mean rating of every division is approximately the same.

### Departments and rating

A picture containing calendar

Description automatically generated

Observations:

* The mean rating of every department is approximately the same.
* There is a huge difference in the mean rating of Bottoms and Trend that shows people rate more Bottoms than Trend.

## Word Cloud:

A **Word Cloud** is a collection or cluster of words depicted in different sizes. The **bigger and bolder** the word appears, the **more often it is selected/voted for** by an audience member.

We implemented the same for our target label rating from one star to five stars, below is the representation of the same.

For Rating 5

Text

Description automatically generated

### For Rating 1

Text

Description automatically generated

# Feature Engineering:

We created two features, Review\_text\_len and tile\_len, from the features ‘review’ and ‘title’ to look for any correlation of their lengths with the target label.

## Review\_text\_len

Chart, histogram

Description automatically generated

## Review\_text\_len and Rating

Chart, box and whisker chart

Description automatically generated

# Data Preprocessing

As we have seen, the dataset contains various features with data types such as categorical, numerical, and text. The ML model, however, can only understand numerical data.

So, we need to convert these features into those that can be understood by the model. The following steps have been taken to tackle this issue:

* Handling Null Values
* One hot encoding
* Min Max Scaling

Text Preprocessing:

* Removing stop words
* Regular Expression
* Lemmatization
* Merging Title and Review Text columns
* BOW / TF-IDF

## Strategy to handle Null Values:

* As we have seen that around 12-13% of data is null in this dataset which is a very large number to drop.
* We did EDA and found that most of the null index is the same for review text and title.
* Hence, we merge the review text and title column (explained in feature engineering) and dropped only those rows which have null values for both review text and title.
* By adopting this strategy, we saved almost 10% of the data.

## One-hot encoding

* The categorical features ('Division\_Name', 'Department\_Name', 'Class\_Name') in this dataset are nominal, so one-hot encoding is used.
* After encoding these becomes 29 features in total.

**Table

Description automatically generated**

# Min-Max Scaling

* The features like ‘Age' and 'Positive\_Feedback\_Count' have their ranges too higher than the encoded features.
* This can negatively affect accuracy and training time, so min-max scaling is used to tackle this problem.

Table

Description automatically generated

# Text Preprocessing

* The main data features on which we are performing our model is in the form of text which contains strings, symbols, repetitive words, stop words, etc. which are not that useful for our test model, so we performed the data cleaning techniques on these features, 'Title' and 'Review\_Text'.
* The text needs to be converted to be in such a form that it is easily executable.
* We performed the following processes on our data.

Text

Description automatically generated

## Removing Stop-words:

* We used the 'nltk' library to remove all the stop-words from the data and storing the data in an empty list.

## Used Regular Expression:

* To get every single word from the data we used the regular expressions re.sub() method and removed all the Symbols, URLS, Punctuations, Next\_line character from the available string text and converted it to lower case so that we will have a similar kind of available data to do further processing.
* We split the available string into a list of words so that each word can be used further.

## Lemmatization:

* Lemmatization will convert all the available words to their root form.
* E.g.: Words like Happy, Happier, Happiest all will be converted to their root form Happy.
* We performed lemmatization on our list of words after getting the data from Regular Expressions and then we join the output to a string and our clean data was available for 'Review\_Text\_Clean' and 'Title\_Clean'.

A picture containing application

Description automatically generated

## BOW / TF-IDF:

* Our data is still in the form of text after cleaning, we need to convert it to a vector form so that our model can understand the data.
* To convert our text data into the vector form we used 2 techniques Bag of Words and TF-IDF

### Bag Of Words:

* Bag of words creates the vector form for each word available in the 'Review\_text' and values it as 0, 1, 2... based on how many times that word is repeated in the Review\_Text sentence.

Table

Description automatically generated with low confidence

* As you can see the bag of words awards 1 mark to a ‘word count’ each time it encounters a word. Thus, all the words with the same count are treated equally.
* We are not able to assess which word is most impactful in the sentence, as this approach is more indicative of word quantity than it is of word quality.
* We do not get a proper/accurate outcome because of this and so moving forward we used TF-IDF in our attempts to tackle this problem.

### TF-IDF

* TF-IDF is the product of Term Frequency and Inverse Data Frequency.
* It gives a value to a particular word based on its term frequency in the sentence, so that we can prioritize any word and its impact on the rating properly.
* Every word with a different priority will have different values as per the word's TF and IDF so that more precise data can be seen than Bag of Words.

Table

Description automatically generated

* As we can see here every word has different values according to its occurrence in the sentence.
* After performing all these different methods on our data, we got 10,029 features in our dataset to work upon.

# Splitting the Dataset:

* We used train\_test\_split to split our dataset into Training and Test datasets.
* We divided the data into 80-20 split between training and test data.
* We used ‘stratified splitting’ to get accurate results as our data was highly imbalanced.

Feature Extraction

As there is the large number of features, PCA has been tried to reduce the number of features.

A picture containing chart

Description automatically generated

* 2000 principal components are selected as they are giving 90% variance.

Text

Description automatically generated with low confidence

* Thus PCA is not so great (f1-score = 0.45). Hence, we dropped the idea of feature extraction.

# Modeling Approaches

## Machine Learning Models and hyperparameter tuning

In this study, we use several machine learning algorithms, such as Logistic Regression, Support Vector Machine with Gaussian Kernel, Support Vector Machine with Linear Kernel, and Random Forest, to predict the ratings. Moreover, we train each model for both BOW and TF-IDF.

### Logistic regression

Logistic regression is a linear classifier, and its output gives the probability of the prediction being equal to 1 (y=1).

A picture containing diagram

Description automatically generated

If the number of features is much larger than the number of examples, this model might become susceptible to overfitting. L2 regularization is used to reduce overfitting.

Text

Description automatically generated

### Support Vector Machines

With SVM, we can use the strong mathematical foundations behind it to explore the problem. It is a non-linear classifier and can use the kernel trick. In SVM, we do not require to tune a lot of parameters, so it is a turn-key model which can be used as a baseline model. SVM, using the kernel trick can transform a lower-dimensional feature set to a higher dimensional feature set, such that it classifies linearly in higher dimensions which then becomes non-linear in lower dimensions. SVM with a Gaussian kernel can even handle an infinite number of features.



A picture containing text

Description automatically generated

### Random Forest

It is based on the ‘ensembling’ method, where we train a lot of trees (n\_estimator) and output is taken by using the output of each tree.

Diagram

Description automatically generated

* Random forest is very good at learning non-linear boundaries but is prone to overfitting. However, we can employ various techniques to reduce overfitting, such as limiting the depth of the tree, minimum number of samples required to split a node, etc.

Text

Description automatically generated

* Grid search CV is used to find the best hyperparameters.
* When the n\_estimators are set to 50 the model is not doing well on both train and test data i.e., model is underfitting.
* When the n\_estimators are more than 500, the model performs well on the training data but does poorly generalizing for test data i.e., model is overfitting.
* When the n\_estimators are set to 250 the model performs well for both train and test data i.e., it shows optimal fit.

# Model Evaluation

### Evaluation Metrics

As the data is highly imbalanced, we have used multiple performance measures, such as accuracy, precision, recall, f1-score, etc., to assess the performance of the models:



A picture containing text

Description automatically generated

Text

Description automatically generated with medium confidence

# Results & Discussion

For better understanding and comparison, we created a table with the index as model name and features as evaluation metrics.

Text

Description automatically generated with medium confidence

Note: Some abbreviations are used to make the table more concise:

* bow: Bag of Words
* tfidf: TF-IDF
* lr: Logistic regression
* rf: Random Forest

# Conclusion

* Random Forest classifier with TF-IDF vectorizer outperforms the other classifiers.
* Stratified split provides better results than non-stratified.
* Feature extraction with PCA didn’t provide good results.

# References

1. The data is acquired from the following source: Kaggle: <https://www.kaggle.com/nicapotato/womens-ecommerce-clothing-reviews>