# REVIEWS- SENTIMENT ANALYSIS

AASHA SURESH

**Background:**

The given dataset contains reviews of various products from a e-commerce company. Reviews include product and user information, ratings, and a plain text review. We have used Sentiment Analysis is the most common text classification tool that analyses an incoming message and tells whether the underlying sentiment is positive, negative.

## Objective:

The objective is to:

- Analyse the reviews based on Helpfulness percentage
- Understanding the frequent words used in positive and negative reviews
- classify the products based on customer review as either positive or negative.

## Data Descripiton:

We have the following columns:

1. Product Id: Unique identifier for the product
2. User Id: unique identifier for the user
3. Profile Name: Profile name of the user
4. Helpfulness Numerator: Number of users who found the review helpful
5. Helpfulness Denominator: Number of users who indicated whether they found the review helpful or not
6. Score: Rating between 1 and 5
7. Time: Timestamp
8. Summary: Summary of the review
9. Text: Review

**Data Pre-processing Steps and EDA:**

The pre-processing of the data included

1. removing duplicate values
2. removing neutral review data points
3. removing data points where helpfulness numerator is not available

***Distribution of the ratings:***

Based on the rating given by each user, we have classified the products as below:

1. **~76 percent** of the reviews in the dataset are **positive reviews** having ratings >3 (4 and 5)
2. **~14 percent** of the reviews in the dataset are **negative reviews** having ratings <3 (1 and 2)
3. Remaining **~8 percent** reviews have neutral rating of 3.

Since a major portion of the reviews are positive, we can say that most of the users have a good experience with their purchases.

2. SINCE WE ARE USING Binary Classification to classify whether the review is positive or negative, we will remove 3-star neutral reviews from the dataset. It is about 7.5% of the total reviews.

3. Remove the data points where helpfulness numerator is more than the helpfulness denominator as it makes no sense. With the remaining dt pts, helpfulness percentage is calculated.

4. With the distribution plot of helpfulness percentage, we infer that most of the people find the product either extremely useful or not useful.

5. Based on the helpfulness percentage, we assign indicators to each review as below:

- helpfulness percentage >= 75- Useful
- helpfulness percentage < 75- Intermediate
- helpfulness percentage <= 40- Not Useful
- helpfulness percentage =0- Not Available

6. 190788 rows are available with 0 helpfulness percentage i.e. is not available for more than 50% of the data. The corresponding data points are also removed.

7. Amongst the remaining dt pts, we assign the sentiment class as positive if the score more than 3 and negative if the score is less than 3

**Algorithm for the Project:**

1. We take the dependent variable as the review text and independent variable as the score.
2. Since we have removed the neutral score data points, we are left with data points with score of 1, 2, 4 and 5. We assign the score as
    - 0 for reviews with 0 and 1 and
    - 1 for reviews with 4 and 5
3. We vectorize the input review text with the help of CountVectorizer/ tf-idf vectorizer

***Count Vectorizer:***

Countvectorizer is a method to convert text to numerical data based on the repetition of words in a sentence. Countvectorizer converts the text to lowercase and uses word-level tokenization.

***TF-IDF:***

tf is the number of times a term appears in a particular document.

tf(t) = No. of times term ‘t’ occurs in a document

Inverse Document Frequency (idf)- idf is a measure of how common or rare a term is across the entire corpus of documents.

df (t) =1 + log e [ n / df(t) ]

where

- n = Total number of documents available
- t = term for which idf value has to be calculated
- df(t) = Number of documents in which the term t appears

term Frequency-Inverse Document Frequency (tf-idf) tf-idf value of a term in a document is the product of its tf and idf. The higher is the value, the more relevant the term is in that document.

4. Stop words doesn’t add much value to the sentence. So, we will remove them. We have used the standard English stop words list available.
5. We then split the data into train and test and apply Logistic Regression.

**Model Evaluation and Technique:**

Since it is an imbalanced dataset, we use auc-roc curve as a performance measure.

ROC curve (receiver operating characteristic curve) is a graph showing the performance of a classification model at all classification thresholds based on two parameters: True Positive Rate and False Positive Rate

$$TPR = \frac{TP}{TP + FN}$$

$$FPR = \frac{FP}{FP + TN}$$

A ROC curve plots TPR vs. FPR at different classification thresholds.

AUC stands for "Area under the ROC Curve." That is, AUC measures the entire two-dimensional area underneath the entire ROC curve (think integral calculus) from (0,0) to (1,1).

AUC ranges in value from 0 to 1. A model whose predictions are 100% wrong has an AUC of 0.0; one whose predictions are 100% correct has an AUC of 1.0. Our model has an AUC of 0.87 which makes it good.

Then, with the help of feature names and coefficients we find the important positive and negative words

**Inferences from the Project:**

***Results for Count Vectorizer:***

Confusion Matrix:
[[10058  2342]
 [ 4109 74522]]

Accuracy: 0.9291340312640749
AUC Score: 0.8794361443672589

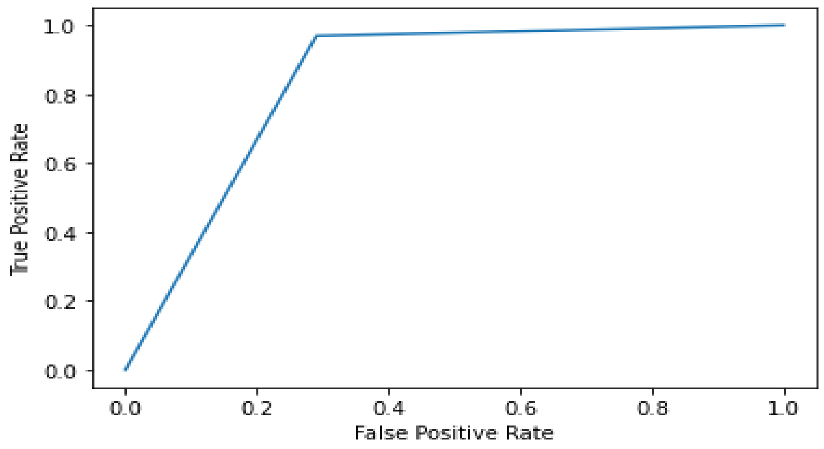
Top 20 Positive words:

| | Word | Coefficient |
|---|---|---|
| 80797 | pleasantly | 2.793921 |
| 5868 | addicting | 2.644562 |
| 111795 | welcome | 2.306464 |

| | | |
|---|---|---|
| 55152 | hooked | 2.248143 |
| 94884 | skeptical | 2.208209 |
| 87434 | relieved | 2.171142 |
| 88168 | resist | 2.150239 |
| 91120 | satisfies | 1.963483 |
| 85436 | ramune | 1.888694 |
| 39857 | duplicates | 1.842211 |
| 65753 | ma | 1.815885 |
| 102287 | tastey | 1.811711 |
| 102296 | tastiest | 1.777777 |
| 61184 | kenzi | 1.767496 |
| 39282 | drawback | 1.764087 |
| 35754 | delighted | 1.758003 |
| 35704 | delicious | 1.757579 |
| 102577 | tearing | 1.757513 |
| 79319 | perruche | 1.745793 |
| 63559 | lends | 1.742436 |

Top 20 Negative words:

| | Word | Coefficient |
|---|---|---|
| 86896 | reformulate | -2.037041 |
| 30346 | commodity | -2.048226 |
| 56355 | ick | -2.066733 |
| 38182 | dissapointing | -2.122054 |
| 11033 | awful | -2.128572 |
| 5765 | actuality | -2.138916 |
| 103075 | terrible | -2.154825 |
| 54890 | holle | -2.158104 |
| 111532 | weakest | -2.160330 |
| 90069 | ruins | -2.164818 |
| 21338 | blech | -2.182839 |
| 107160 | unappealing | -2.336489 |
| 25005 | cancelled | -2.354654 |
| 107650 | undrinkable | -2.360014 |
| 35052 | deceptive | -2.387436 |
| 37628 | disappointment | -2.436964 |
| 68046 | mediocre | -2.450064 |
| 86622 | redeeming | -2.556448 |
| 37625 | disappointing | -2.715056 |
| 113466 | worst | -2.748449 |




***Results for tf-idf vectorizer:***

Confusion Matrix:
[[ 9425  1703]
 [ 5046 74857]]

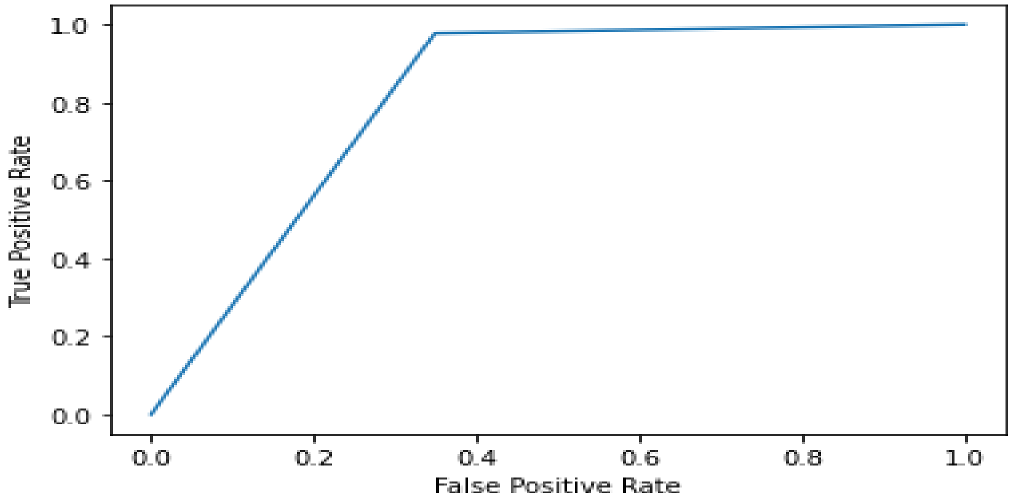
Accuracy: 0.9258604211752041
AUC Score: 0.8919055227711263

Top 20 Positive words:

| | Word | Coefficient |
|---|---|---|
| 51297 | great | 13.814020 |
| 35639 | delicious | 11.739206 |
| 20307 | best | 11.352879 |
| 78885 | perfect | 10.205764 |
| 43220 | excellent | 9.445586 |
| 54306 | highly | 9.241455 |
| 65104 | loves | 8.756859 |
| 65068 | love | 8.168617 |
| 112879 | wonderful | 8.070733 |
| 10992 | awesome | 7.580533 |
| 7821 | amazing | 7.336979 |
| 50507 | good | 7.258296 |
| 72814 | nice | 6.845270 |
| 44750 | favorite | 6.764038 |
| 80604 | pleased | 6.715354 |
| 95446 | smooth | 6.467128 |
| 49852 | glad | 6.464400 |
| 80596 | pleasantly | 6.315828 |
| 55026 | hooked | 6.298172 |
| 114424 | yummy | 6.253712 |

Top 20 Negative words:

| | Word | Coefficient |
|---|---|---|
| 55080 | hoping | -5.462703 |
| 107940 | unpleasant | -5.535360 |
| 113149 | worse | -5.536309 |
| 114316 | yuck | -5.667917 |
| 111043 | waste | -5.895029 |
| 111247 | weak | -6.009485 |
| 21210 | bland | -6.238440 |
| 102035 | tasteless | -6.243006 |
| 37840 | disgusting | -6.382372 |
| 88346 | return | -6.440276 |
| 97952 | stale | -6.496359 |
| 107549 | unfortunately | -6.744928 |
| 103636 | threw | -6.881718 |
| 55165 | horrible | -7.525822 |
| 37554 | disappointed | -7.999876 |
| 37561 | disappointment | -8.581845 |
| 102841 | terrible | -8.698534 |
| 11000 | awful | -8.831384 |
| 37558 | disappointing | -9.340727 |
| 113160 | worst | -11.079343 |




We see that the auc score for tf-idf vectorizer is slightly more than that of count vectorizer and hence the former is preferred.

**Future Possibilities:**

With the help of above words, we can classify a review as positive or negative in the future. This sentiment analysis can help users to identify a product as good or bad from the users who has already bought the same product.

**Conclusion:**

In this project, the focus was on building an automated text classification system which can predict the helpfulness measure of an online review irrespective of the time of posting. The purpose of this was to provide both consumers and manufacturers a wide variety of reviews to choose from by including the most recent yet unvoted reviews in addition to higher voted old dated reviews.