Azerbaijan Journal of High Performance Computing, Vol 4, Issue 1, 2021, pp. 113-125
https://doi.org/10.32010/26166127.2021.4.1.113.125

Sentiment Analysis of Customer Reviews

Syed Rashiq Nazar[1], Tapalina Bhattasali[2]

St. Xavier's College (Autonomous), Kolkata
[1]syedrashiqnazar@gmail.com, [2]tapalina@sxccal.edu

*Correspondence:
Syed Rashiq Nazar,
St. Xavier's College
(Autonomous), Kolkata,
syedrashiqnazar@gmail.
com

Abstract

Sentiment analysis is a process in which we classify text data as positive, negative, or neutral or into some other category, which helps understand the sentiment behind the data. Mainly machine learning and natural language processing methods are combined in this process. One can find customer sentiment in reviews, tweets, comments, etc. A company needs to evaluate the sentiment behind the reviews of its product. Customer sentiment can be a valuable asset to the company. This ultimately helps the company make better decisions regarding its product marketing and improving product quality. This paper focuses on the sentiment analysis of customer reviews from Amazon. The reviews contain textual feedback along with a rating system. The aim is to build a supervised machine learning model to classify the review as positive or negative. As reviews are in the text format, there is a need to vectorize the text to numerical format for the computer to process the data. To do this, we use the Bag-of-words model and the TF-IDF (Term Frequency-Inverse Document Frequency) model. These two models are related to each other, and the aim is to find which model performs better in our case. The problem in our case is a binary classification problem; the logistic regression algorithm is used. Finally, the performance of the model is calculated using a metric called the F1 score.

Keywords: Sentiment Analysis, Customer Review, Bag of Words, TF-IDF, Supervised Machine Learning

1. Introduction

Sentiment analysis (Gupta, 2018) is a process in which we classify text data as positive, negative, or neutral or into some other category, which helps understand the sentiment behind the data (Monkeylearn, n.d.). Mainly machine learning and natural language processing methods are combined in this process. One can find customer sentiment in reviews, tweets, comments, etc. A company needs to evaluate the sentiment behind the reviews of its product. Customer sentiment can be a valuable asset to the company. This ultimately helps the company make better decisions (Algorithmia, 2018) regarding its product marketing and improving product quality.

There is a dataset (Kaggle, n.d.) of customer reviews based on food products. It contains a text review as well as a rating score of 1-5. The task is to classify reviews into two categories- positive or negative. A supervised machine learning model is considered here to achieve this task.

The aim is to design a simple model using a supervised machine learning algorithm that can classify reviews only into two categories, positive or negative (Pang, B., Lee, L., & Vaithyanathan, S., 2002, Liu, B, 2012). Unbiased reviews are not taken into account since they are not very crucial for a company interested in this type of sentiment analysis. It is assumed that the reviews in text format are grammatically correct. No attempt is made for grammatical correction of the reviews.

The dataset on which the work (Kaggle, n.d.) is done contains reviews on food products and is thus domain-specific. The supervised machine learning model created from the dataset is thus applicable to reviews based on food products. The technique demonstrated (Haddi, E., Liu, X., & Shi, Y., 2013; Mishne, G., & Glance, N. S., 2006, March) using Bag-of-words, TF-IDF, and the logistic regression algorithm can be used to create models from any dataset. The success of the technique used in this work is thus applicable to any domain. The process can successfully analyze customer feedback (Pandey, P., & Soni, N., 2019, February; Yi, S., & Liu, X., 2020) and help a company or an organization. Sentiment analysis (Yi, S., & Liu, X., 2020) will help the company or an organization understand what the public thinks about its products or services.

2. Background of Work

2.1 Bag-of-Words

The computer is unable to understand the raw text. Hence the text has to be represented using some vectors. Bag-of-Words (BoW) model is used to represent a sentence as a collection of vectors.

Example:

Review 1: The food is very good and tasty

Review 2: The food is not good and is bitter

Review 3: The food is delicious and good

The vocabulary consists of these 10 words: ‘The’, ‘food’, ‘is’, ‘very’, ‘good’, ‘and’, ‘tasty’, ‘not’, ‘bitter’, ‘delicious’.

Each word is taken, and the occurrence of each word in three reviews is marked with 1s and 0s. This will give us 3 vectors for 3 reviews.

1 indicates that the word is present in the sentence, and 0 indicates that the word is not present.

Table 1. Bag of Words for 3 Sample Reviews

	The	food	is	very	good	and	tasty	not	bitter	delicious	Length of review
R1	1	1	1	1	1	1	1	0	0	0	7
R2	1	1	1	0	1	1	0	1	1	0	8
R3	1	1	1	0	1	1	0	0	0	1	6

Vector of 1st review: [1 1 1 1 1 1 1 0 0 0]
Vector of 2nd Review: [1 1 1 0 1 1 0 1 1 0]
Vector of 3rd Review: [1 1 1 0 1 1 0 0 0 1]

2.2 Term Frequency-Inverse Document Frequency (TF-IDF)

Term frequency-inverse document frequency is meant to find out the importance of a word in a document

Term Frequency (TF) is a measure of how frequently a term, t, appears in a document, d:

$tf_{t,d} = n_{t,d}$ / number of terms in the document --(i)

Where n is the number of times the term "t" appears in the document "d".

Inverse Document Frequency (IDF) measures the importance of a term.

idf_t = log(no. of documents / number of documents with term t) -----------------(ii)

TF-IDF score for each word is calculated as:

$(tf\_idf)_{t,d} = tf_{t,d} * idf_t$---(iii)

Words with higher TF-IDF score are more important.

Table 2. Calculation of TF and IDF for 3 Sample Reviews

Term	Review1	Review2	Review3	TF-1	TF-2	TF-3	IDF
The	1	1	1	1/7	1/8	1/6	log(3/3)=0
Food	1	1	1	1/7	1/8	1/6	log(3/3)=0
Is	1	1	1	1/7	2/8	1/6	log(3/3)=0
Very	1	0	0	1/7	0	0	log(3/1)=0.48
Good	1	1	1	1/7	1/8	1/6	log(3/3)=0
And	1	1	1	1/7	1/8	1/6	log(3/3)=0
Tasty	1	0	0	1/7	0	0	log(3/1)=0.48
Not	0	1	0	0	1/8	0	log(3/1)=0.48
Bitter	0	1	0	0	1/8	0	log(3/1)=0.48
delicious	0	0	1	0	0	1/6	log(3/1)=0.48

Table 3. TF-IDF Values for Each Review

Term	Review1	Review2	Review3	TF-IDF 1	TF-IDF 2	TF-IDF 3
The	1	1	1	0	0	0
Food	1	1	1	0	0	0
Is	1	1	1	0	0	0
Very	1	0	0	0.07	0	0
Good	1	1	1	0	0	0
And	1	1	1	0	0	0
Tasty	1	0	0	0.07	0	0
Not	0	1	0	0	0.06	0

Bitter	0	1	0	0	0.06	0
delicious	0	0	1	0	0	0.08

2.3 Logistic Regression

Logistic regression is a classification algorithm that is used for binary classification problems. For mapping predicted values to a certain probability, the algorithm uses a sigmoid function. The function can map an actual number into a number between 0 and 1.

Formula of Sigmoid Function: $f(x) = 1/(1+e^{-(x)})$ ---(iv)

The classifier is expected to give us a class based on the probability that is passed as input through the prediction function.

For example, there are 2 classes (1-positive, 0-negative). We have fixed a threshold value above which we classify values into class 1 and below which we classify into class 2.

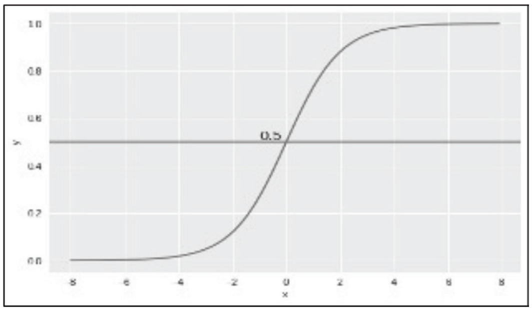


Fig. 1. The sigmoid function curve of logistic regression

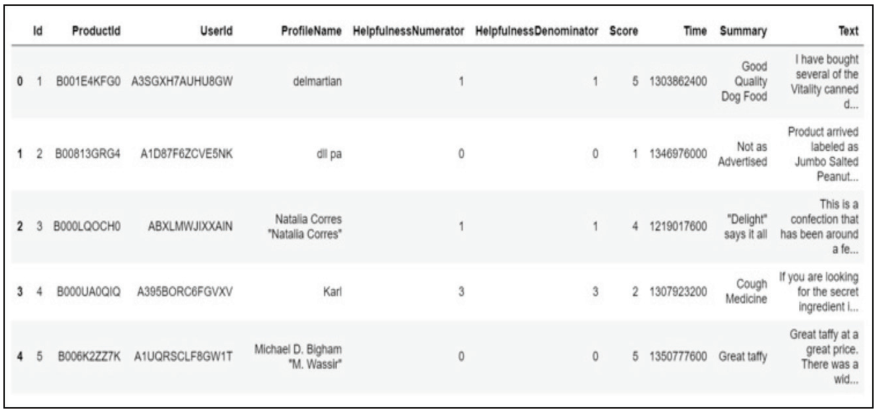
3. Implementation

This research work is implemented using the Python programming language. For visualization, Jupyter Notebook is used. We will work with a dataset in .csv file format that contains reviews on food products. It contains thousands of reviews. There are four steps mentioned below.

1. Read and analyze the input text data and the corresponding response variables (ratings)- perform exploratory data analysis.
2. Perform basic pre-processing to prepare the data for modeling. It will include steps like converting text to lower case, removing special characters, removing stopwords, etc.
3. Apply various ways of Featurizing the reviews text – like bag-of-words (BoW) and TF-IDF (Term frequency-Inverse Document Frequency).
4. Build a machine learning model to classify text as either exhibiting positive or negative sentiment (1 or 0) – Logistic Regression algorithm is to be applied here.

3.1 Exploratory Data Analysis

At first, the head of the dataset is checked to find out the relevant columns of the dataset that are relevant for the analysis.



	Id	ProductId	UserId	ProfileName	HelpfulnessNumerator	HelpfulnessDenominator	Score	Time	Summary	Text
0	1	B001E4KFG0	A3SGXH7AUHU8GW	delmartian	1	1	5	1303862400	Good Quality Dog Food	I have bought several of the Vitality canned d...
1	2	B00813GRG4	A1D87F6ZCVE5NK	dll pa	0	0	1	1346976000	Not as Advertised	Product arrived labeled as Jumbo Salted Peanut...
2	3	B000LQOCH0	ABXLMWJIXXAIN	Natalia Corres "Natalia Corres"	1	1	4	1219017600	"Delight" says it all	This is a confection that has been around a fe...
3	4	B000UA0QIQ	A395BORC6FGVXV	Karl	3	3	2	1307923200	Cough Medicine	If you are looking for the secret ingredient i...
4	5	B006K2ZZ7K	A1UQRSCLF8GW1T	Michael D. Bigham "M. Wassir"	0	0	5	1350777600	Great taffy	Great taffy at a great price. There was a wid...

Fig. 2. Head of the Dataset

We find that two columns are relevant for our sentiment analysis, 'Score' and 'Text". The "score' column contains a score value of 1-5 for a given product. The 'Text' column contains the reviews in text format. Other columns of the dataset are irrelevant for the classification of the reviews.

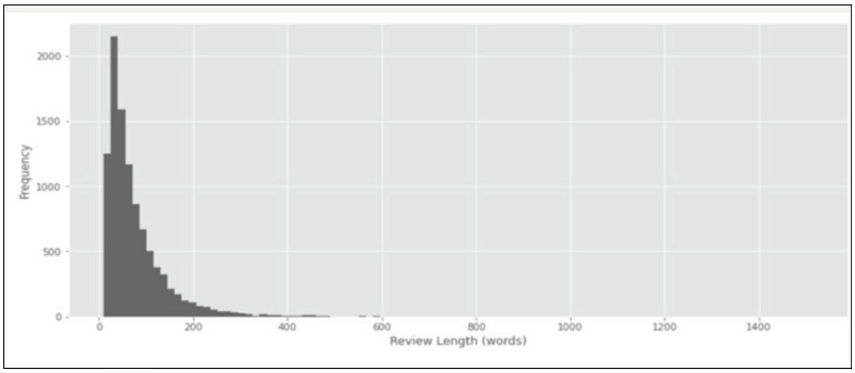


Fig. 3. Distribution of Number of Words per Review

The average number of words per review is found out to be 77.9028. It implies that most of the reviews are big, and the bag-of-words will be large and sparse.

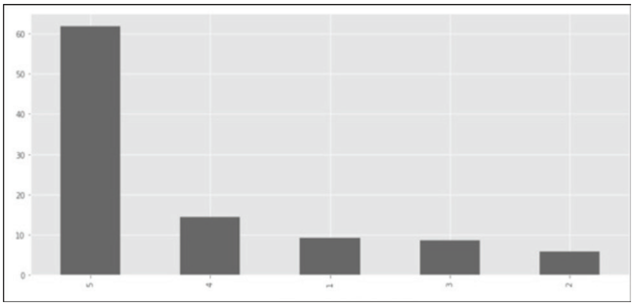


Fig. 4. Distribution of Ratings

The distribution of ratings in figure 4 is skewed with a large number of 5s, and a very few 4s, 3s, 2s,1s. This may affect the training of the data set.

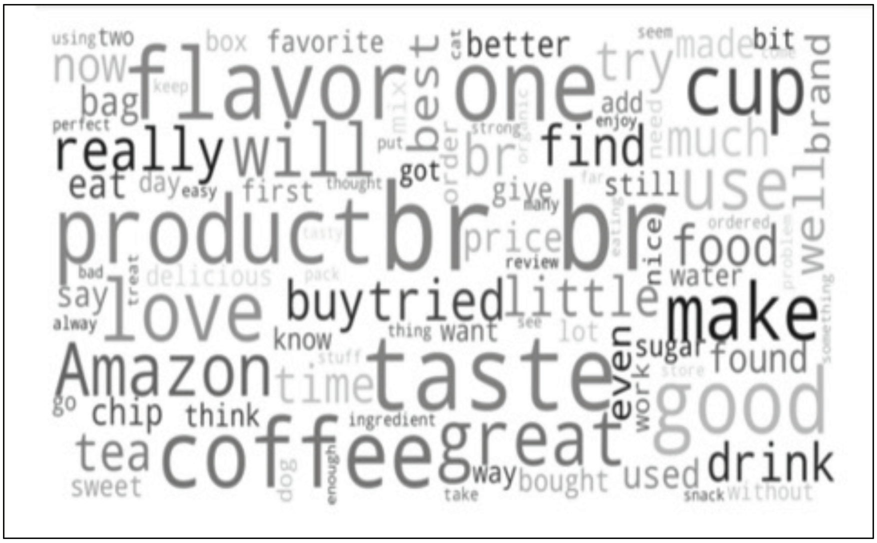


Fig. 5. Text Visualization using Wordcloud

Visualization of text data is as important as visualization of numerical data. Here word cloud comes in handy. The size of a word in a word cloud indicates the frequency of that word in the dataset being analyzed. It is clear from the word cloud that the reviews are related to food items as the words ‘taste’, ‘coffee’, ‘drink’, ‘water’, ‘snack’ indicate. We also find positive words like ‘good’ , ‘great’ and negative words like ‘bad’ and ‘little’. There is also the presence of words that do not contribute to a sentiment, for example- ‘br’ (line break), ‘will’, ‘something’. Hence the aim should be to remove such words.

3.2 Pre-Processing of Data

In this step, the data is prepared for further processing. The following steps are performed here.

Standardizing the Ratings for Sentiment Analysis

For sentiment analysis, we have converted all of the ratings into binary values using the following rule:

- Ratings of 4 or 5 will get mapped to 1 and will be related to positive reviews.
- Ratings of 1 or 2 will get mapped to 0 and will be related to negative reviews.
- Ratings of 3 will get removed since they will represent neutral reviews.

The following codes are used.

```
# Mapping the ratings
amazon_reviews['Sentiment_rating'] = np.where(amazon_reviews.Score > 3,1,0)
# Removing neutral reviews
```

amazon_reviews = amazon_reviews[amazon_reviews.Score != 3]
# Printing the counts of each class
amazon_reviews['Sentiment_rating'].value_counts()

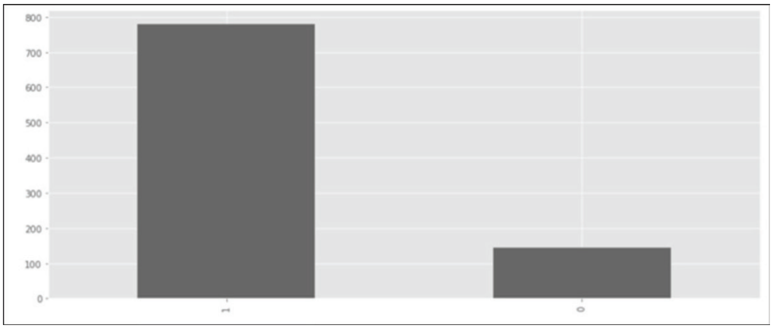


Fig. 6. Distribution of Sentiment Rating After Conversion To Binary

Conversion to Lower Case

The following code is used for converting reviews to lower case.

amazon_reviews['reviews_text_new'] = amazon_reviews['Text'].str.lower()

The conversion to lower case is done because it reduces the number of unique tokens significantly, as is shown below.

Number of unique tokens before conversion to lower case: 27504

Number of unique tokens after conversion to lower case: 22470

Removal of Special Characters

The special characters are removed from the reviews to simplify the work but based on the results later, this decision may be revisited.

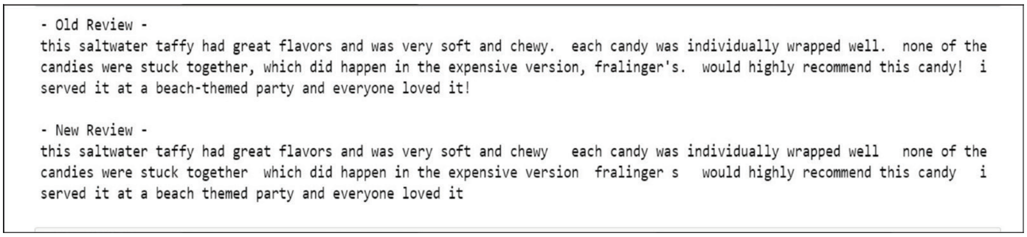
The following code removes special characters:

review_backup = amazon_reviews['reviews_text_new'].copy()

amazon_reviews['reviews_text_new'] = amazon_reviews['reviews_text_new'].str.replace(r'[^A- Za-z0-9]+', ' ')

The regular expression [^A-Za-z0-9] + can be decomposed as follows.

• [ ]: match any character inside the list defined by the square brackets, including the space character ' '
• ^: start of the line
• A-Z: accept english upper case characters from A to Z
• a-z: accept english lower case characters from a to z
• 0-9: accept single number characters from 0 to 9



```
- Old Review -
this saltwater taffy had great flavors and was very soft and chewy.  each candy was individually wrapped well.  none of the
candies were stuck together, which did happen in the expensive version, fralinger's.  would highly recommend this candy!  i
served it at a beach-themed party and everyone loved it!

- New Review -
this saltwater taffy had great flavors and was very soft and chewy   each candy was individually wrapped well   none of the
candies were stuck together  which did happen in the expensive version  fralinger s   would highly recommend this candy   i
served it at a beach themed party and everyone loved it
```

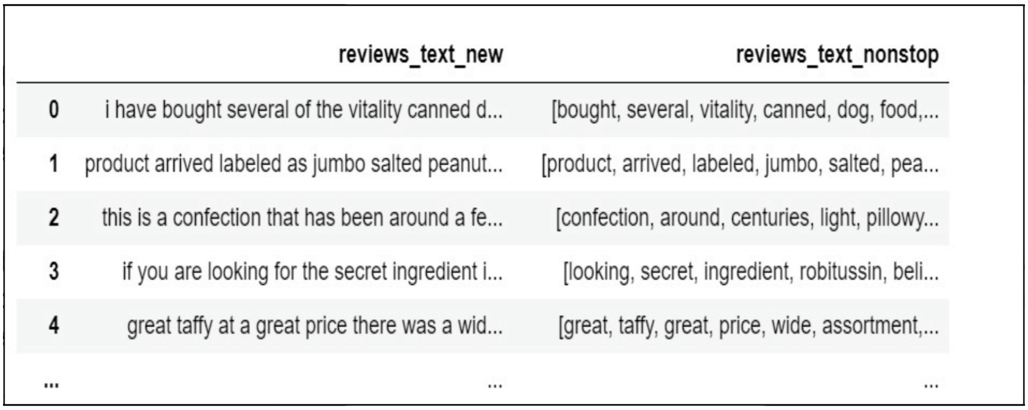
Fig. 7. Removal of Special Characters from a Review

Removal of Stopwords

Stopwords are those words that do not add meaning to a sentence. They can be ignored without affecting the meaning of the sentence. Example- the, he, have, etc.

The following code removes stopwords-

```
#Removal of stopwords from text
stop_words = set(eng_stop_words)
def stopwords_removal(stop_words, sentence):
return [word for word in nltk.word_tokenize(sentence) if word not in stop_words]
amazon_reviews['reviews_text_nonstop'] = amazon_reviews['reviews_text_new'].apply(lambda row: stopwords_removal(stop_words, row))
amazon_reviews[['reviews_text_new','reviews_text_nonstop']]
```



	reviews_text_new	reviews_text_nonstop
0	i have bought several of the vitality canned d...	[bought, several, vitality, canned, dog, food,...
1	product arrived labeled as jumbo salted peanut...	[product, arrived, labeled, jumbo, salted, pea...
2	this is a confection that has been around a fe...	[confection, around, centuries, light, pillowy...
3	if you are looking for the secret ingredient i...	[looking, secret, ingredient, robitussin, beli...
4	great taffy at a great price there was a wid...	[great, taffy, great, price, wide, assortment,...
...	...	...

Fig. 8. Reviews after Removal of Stopwords

3.3. Feature Extraction from Text

It is not possible for machine learning algorithms to directly work on raw text. Here comes the need to create features from the text data, which is later used to train the model using an algorithm.

Creating Bag-of-Words

The following codes are used to create a bag-of-words from the pre-processed reviews:

```
### Creating a python object of the class CountVectorizer
bow_counts = CountVectorizer(tokenizer = word_tokenize, # type of tokenization
 ngram_range=(1,1)) # number of n-grams
bow_data = bow_counts.fit_transform(amazon_reviews['reviews_text_new1'])
```

Creating TF-IDF (Term Frequency- Inverse Document Frequency) model

The following codes create TF-IDF model from the pre-processed reviews.

```
from sklearn.feature_extraction.text import TfidfVectorizer
### Creating a python object of the class CountVectorizer
tfidf_counts = TfidfVectorizer(tokenizer= word_tokenize, # type of tokenization
 ngram_range=(1,1)) # number of n-grams
tfidf_data = tfidf_counts.fit_transform(amazon_reviews['reviews_text_new1'])
```

3.4. Applying Logistic Regression Algorithm

We are only interested in classifying the reviews into 2 categories- Positive or Negative. Hence it can be called a binary classification problem. Logistic regression is suited to perform binary classification problems. Hence this algorithm is chosen to build a supervised machine learning model.

The "sentiment_rating" values created earlier by mapping the ratings of 1-5 into either 0 or 1 is the target variable (dependent variable) in our case. The independent variables are either the elements of the feature matrix created from bag-of-words or the TF-IDF model.

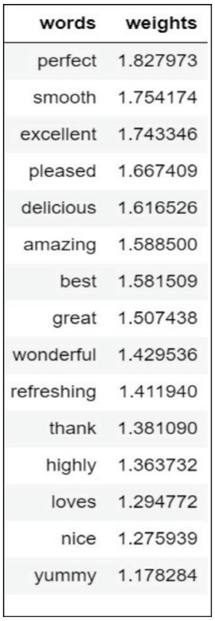
The dataset is first divided into training and testing sets. The size of the test set is 20%, wheres the size of the training set is 80%.

The following code is used:

```
    X_train_bow, X_test_bow, y_train_bow, y_test_bow = train_test_split(bow_data,
# Features
     amazon_reviews['Sentiment_rating']# Target variable
     test_size = 0.2, # 20% test size
     random_state = 0) # random state for replication
```

Then the model is trained using logistic regression algorithm and the class(0 or 1) is predicted for the test set using the following code:

```
#Applying logistic regression
### Training the model
lr_model_all = LogisticRegression() # Logistic regression
lr_model_all.fit(X_train_bow, y_train_bow) # Fitting a logistic regression model
## Predicting the output
```



words	weights
perfect	1.827973
smooth	1.754174
excellent	1.743346
pleased	1.667409
delicious	1.616526
amazing	1.588500
best	1.581509
great	1.507438
wonderful	1.429536
refreshing	1.411940
thank	1.381090
highly	1.363732
loves	1.294772
nice	1.275939
yummy	1.178284

Fig. 9. Top 15 Features for Positive Reviews

test_pred_lr_all = lr_model_all.predict(X_test_bow) # Class prediction

4. Results and Performance Score

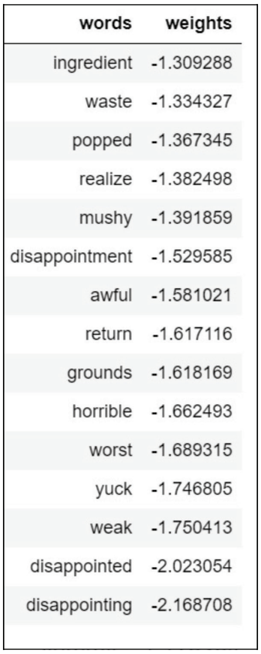
4.1 Display Top 15 Positive and Negative Words

To find out the top 15 features for positive reviews, we have used the following codes.

lr_weights = pd.DataFrame(list(zip(bow_counts.get_feature_names(), # get all feature names
lr_model_all.coef_[0])), # get the logistic regression coefficients
columns= ['words','weights']) # defining the colunm names
lr_weights.sort_values(['weights'], ascending = False)[:15]
# top-15 features for positive reviews

To find out the top 15 features for positive reviews, we have used the following code.

lr_weights.sort_values(['weights'], ascending = False)[-15:]
# top-15 features for negative reviews

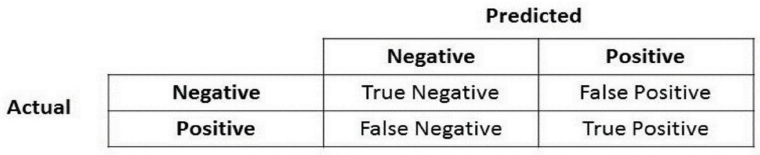


words	weights
ingredient	-1.309288
waste	-1.334327
popped	-1.367345
realize	-1.382498
mushy	-1.391859
disappointment	-1.529585
awful	-1.581021
return	-1.617116
grounds	-1.618169
horrible	-1.662493
worst	-1.689315
yuck	-1.746805
weak	-1.750413
disappointed	-2.023054
disappointing	-2.168708

Fig. 10. Top 15 Features for Negative Reviews

So the output clearly shows that words like 'perfect' , 'smooth', excellent', 'delicious' , etc., have contributed to the weight of positive reviews. On the other hand, words like 'waste', 'disappointment', 'yuck' have contributed to negative reviews. These words have almost correctly been predicted as positive or negative, and it shows that our machine learning model has good accuracy in its result.

4.2 Performance Score



		Predicted	
		Negative	Positive
Actual	Negative	True Negative	False Positive
	Positive	False Negative	True Positive

Fig. 11. Confusion Matrix

Where,
TN=True Negative
FP=False positive
FN=False Negative
TP=True Positive
Accuracy=(TN+TP/ TN+FP+FN+TP)---(v)

However, this metric may be misleading if the high cost is associated with false positive or false negative values. We are interested in a metric that balances both false-positive and false-negative results in such a case. Generally, for sentiment analysis, both false positive and false negative values are of high cost for an organization.

Two other metrics are precision and recall.
Precision=TP / (TP+FP) ---(vi)
Recall= TP/ (TP + FN) --(vii)

If the cost of False Positive is high, Precision value is taken into account.

On the other hand, if the cost of False Negative is high, we take into account the value of Recall to find out the performance of our model.

Another metric called the F1 score creates a balance between the costs of false positive and false negative to measure the performance of a model.

F1 score = 2*(Precision*Recall) / (Precision + recall) ------------------------------(viii)

Hence, to measure the performance of our model, we use the F1 score.

We find out the F1 score of the model created using bag-of-words and for TF-IDF. The results are as follows:

For Bag of Words and logistic regression:
F1 score: 0.9507467429297743
For TF-IDF and logistic regression:
F1 Score: 0.9353566009104705

From the above values, it is clear that for our sentiment analysis of customer reviews of food items, the bag-of-words with logistic regression performs better than the TF-IDF model with logistic regression.

5. Drawbacks & Limitations

The limitations of the work can be listed in the below-mentioned points.

• The reviews were not checked for any grammatical errors.
• Stemming/Lemmatization of the words was not performed. If done, it could

further reduce the number of unique tokens.

• Special characters can have particular implications in forming a sentiment. They may be used as emojis by the reviewer. However, for simplicity, these special characters were not taken into account.

• The feature matrix was created using a single word. Bigrams or trigrams can be used to add context to the features. It will lead to better results.

6. Conclusion

The task was to build a supervised machine learning model that can classify reviews into two categories- positive (1) or negative (0). Exploratory data analysis was performed first to understand the dataset that was available to us. From it, we could understand the columns in the dataset that are relevant for sentiment analysis. Irrelevant columns of the dataset were not considered for sentiment analysis. The distribution of ratings was found to be skewed towards ratings of score 5. However, this skewness did not significantly affect the performance of the model. Then for creating vectors from the text review, we used Bag-of-Words. The reason for choosing Bag-of-Words is that it uses a simple method and structure to represent the vectors and is easy and fast to implement. Another method used for creating vectors from the text reviews is TF-IDF (Term Frequency-Inverse Document Frequency). The reason for choosing TF-IDF is that the method considers the importance of a word to a document and is more logical. Then the logistic regression algorithm was applied to the vectors created from the Bag-of-Words or TF-IDF. The logistic regression algorithm is a classification algorithm used to solve binary classification problems. Our problem was to classify reviews only into two categories – either positive (1) or negative (0). So it can be called a binary classification problem. Hence Logistic regression algorithm was used as it can effectively solve a binary classification problem. The top 15 words that contributed to positive reviews and the top 15 words that contributed to negative reviews were found out, and they were almost relevant and accurate. In the end, we have considered a metric named F1 score to find the performance of our model. The F1 score was used to find the model's performance because it considers both high-cost false positive and high-cost false-negative results. The F1 score of above 90% indicated a very good performance of our model. The bag-of-words model even outperformed the TF-IDF model.

References

Algorithmia (2018) “Introduction to sentiment analysis: What is sentiment analysis?” Available Online At: https://algorithmia.com/blog/introduction-sentiment-analysis

Gupta, S. (2018). Sentiment analysis: concept, analysis and applications. Toward Data Science.

Haddi, E., Liu, X., & Shi, Y. (2013). The role of text pre-processing in sentiment analysis. *Procedia Computer Science, 17,* 26-32.

Kaggle (n.d.) Sentiment Analysis. Available Online At: https://www.kaggle.com/snap/amazon-fine-food-reviews

Liu, B. (2012). Sentiment analysis and opinion mining. *Synthesis lectures on human* *language technologies, 5*(1), 1-167.

Mishne, G., & Glance, N. S. (2006, March). Predicting movie sales from blogger sentiment. In *AAAI spring symposium: computational approaches to analyzing weblogs* (pp. 155-158).

Monkeylearn (n.d.) Sentiment Analysis: A Definitive Guide. Available Online At: https://monkeylearn.com/sentiment-analysis/

Pandey, P., & Soni, N. (2019, February). Sentiment Analysis on Customer Feedback Data: Amazon Product Reviews. In *2019 International Conference on Machine Learning, Big Data, Cloud and Parallel Computing (COMITCon)* (pp. 320-322). IEEE.

Pang, B., Lee, L., & Vaithyanathan, S. (2002). Thumbs up? Sentiment classification using machine learning techniques. arXiv preprint cs/0205070.

Yi, S., & Liu, X. (2020). Machine learning based customer sentiment analysis for recommending shoppers, shops based on customers' review. *Complex & Intelligent Systems, 6*(3), 621-634.

Submitted: 25.09.2020
Accepted: 26.05.2021