Sentiment Analysis on Amazon Food Reviews

Venkatesh Ummadisingu
Department of Information and Computer Science
University of California Irvine
vummadis@uci.edu

Vamsi Krishna Balusu
Department of Information and Computer Science
University of California Irvine
balusuv@uci.edu

Ajith Varma Penmatsa
Department of Information and Computer Science
University of California Irvine
apenmats@uci.edu

Manish Kumar Mathukumilli
Department of Information and Computer Science
University of California Irvine
mmathuku@uci.edu

Contents

Abstract

Classification is very important both for humans and machines. Sentiment analysis is the process of classifying a piece of text based on the intuitive information it expresses. It is one of the top applications of Natural Language Processing (NLP). The popularity of sentiment analysis is due to its wide range of real-time applications. Automatically understanding the opinions behind the user-generated text like social media posts and product reviews is of great help for commercial purposes. So many of the leading tech companies are investing in sentiment analysis of the user reviews for their products. In this project, we performed sentiment analysis on Amazon food reviews data set using four different machine learning models – Naive Bayes, Logistic Regression, XgBoost and LSTM Neural Networks and compared the results.

Introduction

In Artificial Intelligence, Natural Language Processing (NLP) is a field which deals with the ways in which computers interpret the human languages. Usage of idioms, metaphors, grammar makes human language irregular and complex for machines to process and understand. NLP combines rule-based modeling of human language with statistical, machine learning and deep learning models. These technologies enable computers to process human voice or text data and understand its meaning.

NLP has wide range of applications like search autocorrect, language translation, sentiment analysis, chat bots, hiring and recruitment - resume parsing, voice assistants, grammar checkers and email filtering.

Sentiment analysis is the process of detecting attitudes, sarcasm, emotions, confusion, suspicion from a given piece of text. In sentiment analysis, the program assigns weighted scores to themes and categories in a sentence and understands if the sentiment behind a piece of text is positive, negative, or neutral.

Below are some of the applications of sentiment analysis

1. **User Sentiment Evolution**
   User opinion of a product/brand is constantly changing and evolving. So, it is essential to monitor its evolution through time and competition from other brands. Thus, discovery of negative trends in users' sentiment is very important.

2. **Operations Analysis**
   There are many aspects which contribute to the customer's opinion of a product. Even if the data is partially organized, it is still very difficult to identify the main factor that contributed to the overall review of the product. Combining sentiment analysis and topic recognition allows us to estimate the purpose behind each review.

3. **Segment Analysis**
   A clear understanding of which user segment is a supporter, or a detractor of a product ensures a successful marketing campaign. This data allows the marketing department to better target disappointed users.

4. **Point of Sale Review**
   This use case is particularly important for companies with big geographical footprint. Granular analysis of user opinions in various geographical areas is necessary to identify failing sale points that requires particular attention.

Our project falls under the first category in the above listed applications where we aim to predict the user sentiments on Amazon food reviews and classify them as positive/negative reviews.

Data set

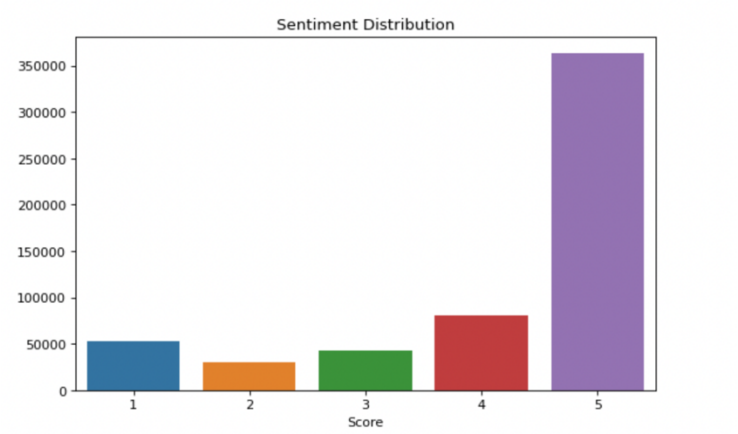
Amazon fine food reviews data set is used in this project to perform sentiment analysis on the reviews provided by the customers for the food products purchased on the Amazon platform. The csv file has 568,455 reviews for various food products. The data span a period of more than 10 years. Reviews include product and user information, ratings, and a plaintext review.

Id	1
ProductId	B001E4KFG0
UserId	A3SGXH7AUHU8GW
ProfileName	delmartian
HelpfulnessNumerator	1
HelpfulnessDenominator	1
Score	5
Time	1303862400
Summary	Good Quality Dog Food
Text	I have bought several of the Vitality canned d...

Data format

```
Number of reviews : 568454
Number of products : 74258
Number of users : 256059
```

Statistics



Data distribution

The data set has 10 features including the text review and score. We are interested only in the text review on which we perform the sentiment analysis. To classify a review as positive or negative, we made use of the score feature and added a new column named “Class” to specify the classification. A review is classified as positive if the rating is either 4 or 5 and as negative if the rating is 1 or 2. We dropped all the reviews with rating equal to 3 as it is neutral in most of the cases.

	Id	ProductId	UserId	ProfileName	HelpfulnessNumerator	HelpfulnessDenominator	Score	Time	Summary	Text	Class
0	1	B001E4KFG0	A3SGXH7AUHU8GW	delmartian	1	1	5	1303862400	Good Quality Dog Food	I have bought several of the Vitality canned d...	Positive
1	2	B00813GRG4	A1D87F6ZCVE5NK	dll pa	0	0	1	1346976000	Not as Advertised	Product arrived labeled as Jumbo Salted Peanut...	Negative

classification

Data preprocessing

Text pre-processing is an important step in any Natural Language Processing tasks. It is the process of removing unnecessary noise from the given piece of text. The data that we get from customers is unstructured and needs to be cleaned up for the machine learning algorithms to perform better. The noise can include duplicates, extra spaces, special characters, numbers, html tags (in case the data is obtained from web scrapping), upper case – lower case combinations, in correct grammar etc.

We performed the following preprocessing steps on the data set:

1) Duplicates

The data set has several duplicate reviews. We filtered the dataset to remove all the duplicate entries.

	UserId	Time	Summary	Text
73790	AR5J8UI46CURR	1199577600	LOACKER QUADRATINI VANILLA WAFERS	DELICIOUS WAFERS. I FIND THAT EUROPEAN WAFERS ...
78444	AR5J8UI46CURR	1199577600	LOACKER QUADRATINI VANILLA WAFERS	DELICIOUS WAFERS. I FIND THAT EUROPEAN WAFERS ...
138276	AR5J8UI46CURR	1199577600	LOACKER QUADRATINI VANILLA WAFERS	DELICIOUS WAFERS. I FIND THAT EUROPEAN WAFERS ...
138316	AR5J8UI46CURR	1199577600	LOACKER QUADRATINI VANILLA WAFERS	DELICIOUS WAFERS. I FIND THAT EUROPEAN WAFERS ...
155048	AR5J8UI46CURR	1199577600	LOACKER QUADRATINI VANILLA WAFERS	DELICIOUS WAFERS. I FIND THAT EUROPEAN WAFERS ...

duplicate entries

After dropping the duplicates, we lost around 30 percent of the original data.

```
print("Percentage of data fltered out: ", 100- (filtered_data['Id'].size * 1.0) / (data['Id'].size  *1.0) * 100)
```

```
Percentage of data fltered out:  30.741098563370315
```

2) Case of alphabets

Some of the reviews have combination of lower-case and upper-case alphabets. We converted entire text to lower case letters.

```
Before converting to lower case:
['DELICIOUS WAFERS. I FIND THAT EUROPEAN WAFERS ARE LESS SWEET ( LESS SUGAR AND CARBOHYDRATES) BUT VERY DELECTABLE.']


After converting to lower case:
['delicious wafers. i find that european wafers are less sweet ( less sugar and carbohydrates) but very delectable.']
```

case conversion

3) HTML tags

The review text has HTML tags which are not relevant in determining the sentiment of the review. They only help the review to render in a certain manner in the browser. So, we removed all kinds of HTML tags from the text as part of pre-processing.

```
Before removing HTML tags:

['product received is as advertised.<br /><br /><a href="http://www.amazon.com/gp/product/b001gvisjm">twizzlers, strawberry, 16-ounce bags (pack of 6)</a>'
 'best price for wasa crispbread, multi grain, 9.7-ounce boxes (pack of 12)is to order through amazon.com "subscribe & save items"'
 'product received is as advertised.  <a href="http://www.amazon.com/gp/product/b002edemly">red vines red original licorice twists, 64-ounce tub</a>']


After removing HTML tags:

['product received is as advertised.twizzlers, strawberry, 16-ounce bags (pack of 6)'
 'best price for wasa crispbread, multi grain, 9.7-ounce boxes (pack of 12)is to order through amazon.com "subscribe & save items"'
 'product received is as advertised.  red vines red original licorice twists, 64-ounce tub']
```

removing html tags

4) Contractions

Contractions are words or combinations of words which are shortened by dropping letters and replacing them by an apostrophe. The computer can't decode the contractions and treats contracted word as a different word from the expanded words. Hence, contractions will also increase the size of the document term matrix when the text is vectorized. So, we expanded all the contractions in the review text.

```
Before expanding:
["i don't know if it's the cactus or the tequila or just the unique combination of ingredients, but the flavour of this hot sauce makes it one of a kind!
 "just as with the original tequila picante gourmet de inclan hot sauce, this salsa has a unique, addictive flavour.  we've been a salsa family for two de

After expanding:
['i do not know if it is the cactus or the tequila or just the unique combination of ingredients, but the flavour of this hot sauce makes it one of a kind
 'just as with the original tequila picante gourmet de inclan hot sauce, this salsa has a unique, addictive flavour.  we have been a salsa family for two
```

expansions

5) Stop words

Stop words are the commonly used words in a language. English has several such words like 'a', 'and' 'the', 'as', 'on', 'which' etc. They can safely be ignored without changing the meaning of the text. So, we removed the stop words from the reviews as they do not provide any information for the machine learning models.

```
Before removing stop words:
['delicious wafers i find that european wafers are less sweet  less sugar and carbohydrates but very delectable']

After removing stop words:
['delicious wafers find european wafers less sweet less sugar carbohydrates delectable']
```

stop words

6) Non alphabetic characters

Reviews contain many non-alphabetic characters including extra spaces, numbers and special characters which should be removed to obtain more meaningful results. We cleaned the text to

remove all non-alphabetic characters before passing the data to the machine learning models.

7) Stemming and Lemmatization

Reviews contain different forms of the same word, such as organize, organizes, and organizing. Stemming and Lemmatization are the processes used to reduce a word to its root word. The root word is called as “stem” in case of stemming and “lemma” in case of lemmatization.

In stemming, a given word is chopped off at the tail to obtain its root word and the algorithms don’t know the meaning of the word in the language it belongs to. In lemmatization, on the other hand, the algorithms have this knowledge. For example, lemmatization algorithm knows that the word “better” is derived from “good”. So, as lemmatization is more accurate, we preferred lemmatization for the reviews.

We used WordNetLemmatizer from NLTK library. WordNet is a large and publicly available lexical database for the English language aiming to establish structured semantic relationships between words.

```
Before lemmatization:
['healthy dog food good digestion also good small puppies dog eats required amount every feeding'
 'dog come outside training look cupboard waiting treat use clicker training method comes knows something special']

After lemmatization:
['healthy dog food good digestion also good small puppy dog eats required amount every feeding'
 'dog come outside training look cupboard waiting treat use clicker training method come know something special']
```

lemmatization

8) Input and output variables

Text review of the product is the input ‘x’ and class of the review (positive or negative) is the output ‘y’.

9) Splitting the data into training and testing sets

We have split the data in the ratio 80:20 for training and testing.

Models

1) Naive Bayes

Naive Bayes is a supervised learning and probabilistic machine learning algorithm. It classifies the output based on bayes theorem. Bayes theorem helps us to determine the probability of a hypothesis given the prior knowledge of conditions related to the hypothesis.

Below is the equation for Bayes theorem

$$P(A/B\ ) = \frac{P(B/A) * P(A)}{P(B)}$$

where,

- $P(A/B)$ *is the conditional probability i.e., the probability of event "A" happening given that that event "B" happened. This is also called as posterior probability.*
- $P(B/A)$ *is the conditional probability i.e., the probability of event "B" happening given that that event "A" happened.*
- $P(A)$ *is the probability that the event "A" happened. This is also called as prior probability of "A".*
- $P(B)$ *is the probability that the event "B" happened.*

Naive Bayes returns the class for which the posterior probability is maximum. It can therefore be used to perform sentiment analysis on text where the algorithm calculates the conditional probabilities for a review to be positive and negative based on prior knowledge of probability of words occurring in positive and negative reviews.

$$output\ class\ \hat{c} = \ argmax_{c\ =\ \{positive,\ \ negative\}}\ P(c/review)$$

Using bayes theorem,

$$P(c/review) = \frac{P(review/c) * P(c)}{P(review)}$$

Ignoring the denominator in the above equations,

$$output\ class\ \hat{c} = argmax_{c\ =\ \{positive,\ \ negative\}}\ P(review/c) * P(c)$$

Without loss of generalization,

$P(review/c) = \ P(w_1, w_2, w_3, \ldots, w_n/c) * P(c)$ where w_1, w_2, …. , w_n are the words in the review.

Naive bayes assumes that each of the features i.e., words in the review are *independent* of each other.

$$P(w_1, w_2, w_3, \dots, w_n/c) = \prod_{i=1}^{n} P(w_i/c)$$

$$output\ class\ \hat{c} = argmax_{\text{c = \{positive, negative\}}}\ P(c) * \prod_{i=1}^{n} P(w_i/c)$$

here

$$P(c) = \frac{Number\ of\ reviews\ in\ class}{Total\ number\ of\ reviews}$$

and

$$P(w_i/c\ ) = \frac{Frequence\ of\ word\ w_i\ in\ class\ c\ reviews}{Total\ number\ of\ words\ in\ class\ c\ reviews}$$

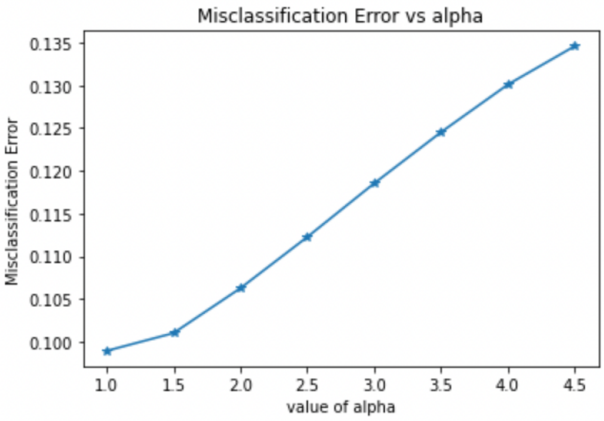
While calculating $P(w_i/c\ )$, since the word w_i may not be present in the class 'c' reviews, it results in $P(w_i/c\ )$ to be zero. To avoid this problem of zero probabilities, we use *Laplacian Smoothing*.

$$P(w_i/c\ ) = \frac{Frequence\ of\ word\ w_i\ in\ class\ c\ reviews + a}{Total\ number\ of\ words\ in\ class\ c\ reviews + a * K}$$

where,

- α *represents the smoothing parameter*
- *K represents the dimension of the set of words*

We determined the optimal value of alpha for Laplacian Smoothing by using *10 - fold cross validation* of Naive Bayes algorithm for alpha in the range one to five. The optimal value turned to be one.




```
Optimal value of alpha is:  1.0
```

There are 3 types of Naive Bayes models

a) **Gaussian Naive Bayes:** It is used for continuous inputs where each input feature follows a normal probability distribution.
b) **Bernoulli Naïve Bayes:** It is used when the input features are binary.
c) **Multinomial Naïve Bayes:** It is used when the inputs represent frequencies.

As we are using frequencies of words to calculate the probabilities, we chose Multinomial Naïve Bayes model. To compute the frequencies of words in the reviews, we used *CountVectorizer* from Python's sklearn library.

2) Logistic Regression

Logistic regression is the process of modeling the probability of a discrete outcome based on input data. Most common logistic regression models result in a binary outcome such as positive/negative, 0/1, and so on. It can also model scenarios where there are more than two possible discrete outcomes. It is a majorly used in solving classification problems, where you are trying to determine if an input fits best into a particular class.

Logistic regression describes and estimates the relationship between one dependent binary variable and independent variables. It is a special case of linear regression where the target variable is categorical in nature. It predicts the probability of occurrence of a binary event using a logistic function.

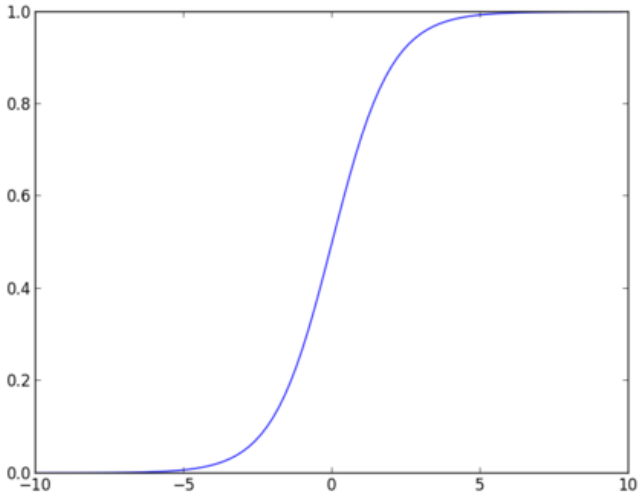
Logistic Regression has the following properties

- Dependent variable follows Bernoulli Distribution.
- Follows Maximum Likelihood Estimation approach.

The sigmoid function, also known as the logistic function, gives an 'S' shaped curve that can take any real-valued number and map it into a value between 0 and 1. If the curve approaches positive infinity, y will be predicted as 1, and if the curve approaches to negative infinity, y will be predicted as 0. If the output of the sigmoid function is more than 0.5, we can classify the outcome as 1 and if it is less than 0.5, we can classify it as 0.

Logistic regression uses a logistic function defined below to model a binary target outcome.

$$Logistic\ function = \frac{1}{1 + e^{-x}}$$



Below are the different types of Logistic Regression

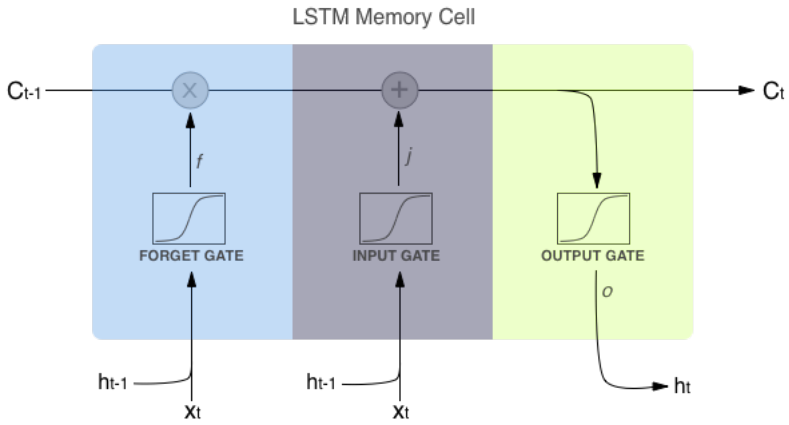
- **Binary Logistic Regression:** The output variable has two possible outcomes.
- **Multinomial Logistic Regression:** The output variable can be classified into three or more nominal categories such as predicting types of wine.
- **Ordinal Logistic Regression:** The output variable can be classified into three or more ordinal categories such product rating from 1 to 5.

We have selected logistic regression model to do sentiment analysis because the target variable is binary (positive or negative) and Logistic Regression is a very popular and proven classification model especially when the output is binary. In our code we have used logistic regression implementation from scikit-learn. Scikit-learn has a well-optimized version of logistic regression implementation, which supports multiclass classification.

In the code, we used *CountVectorizer* to vectorize the text data. We have given maximum number of iterations as 1000 while training the model. The default value for maximum iterations is 100 but we have increased it to 1000 since the solver(lbfgs) took approximately 1000 iterations to converge.

3) Long Short-Term Memory

Long short-term memory (LSTM) is a type of recurrent neural network (RNN). LSTM can process not only single data point, but entire sequences or window of data.



As shown in the above figure, a LSTM unit consists of an input gate, output gate, forget gate and a cell. The cell is used to remember data over a period. Also, LSTM can deal with vanishing gradient problem which RNN does not handle.

For Sentimental Analysis, we get the context of every sentence and LSTM suits the use case as it follows the same principle and processes entire sequence of words at a time.

We preprocess the data using a method called embedding which maps discrete words in text reviews to continuous values. We used *Tokenizer API* from Python's *Keras* library to convert the text to 2D vector as the suitable input for LSTM.

We have 3 layers in our LSTM model.

- The first layer is an embedding layer. Here 140 is the maximum number of words in the review text.
- The second layer is LSTM with 100 hidden nodes.
- The last layer is the dense layer for the which argument passed is one. So all the reviews with output in the range 0 to 0.5 are considered as negative reviews and in the range 0.5 to 1 are considered positive.

Layer (type)	Output Shape	Param #
embedding (Embedding)	(None, 140, 32)	208032
lstm (LSTM)	(None, 100)	53200
dense (Dense)	(None, 1)	101

layers in LSTM

We have taken the batch size as 64, and anything more than that value is giving GPU issues. We have taken number of epochs as 3 as it turned out to be optimal value when validation loss is considered.

4) XGBOOST

XGBOOST (eXtreme Gradient Boosting) is an advanced example of Ensemble Learning Method which implements improved version of Gradient Boosting. It is used widely for its performance, speed and efficiency. This creates new weak learners (decision trees) measures its predictions and then sequentially combines them to create new strong learners with improved predictions.

For incorrect predictions, Gradient boosting adjusts weights using gradient (a direction in the loss function) using an algorithm called Gradient Descent, which iteratively optimizes the loss of the model by updating weights.

$$w = w - \eta \nabla w$$
$$\nabla w = \frac{\partial L}{\partial w} \; where \; L \; is \; loss$$

w- represents the weight vector and η is the learning rate
L- difference between the predicted and actual value

Learning rate(η), also known as shrinkage, slows down the learning in gradient boosting model. η ranges from 0 to 1. Higher values of η results in less correction for the new decision trees created

Gradient descent algorithm is used to minimize the loss for each new decision tree created. Existing trees in the model remain untouched and thus slow down the rate of overfitting. The output of the new tree is combined with the output of existing trees until the loss is minimized below a threshold or specified limit of trees is reached.

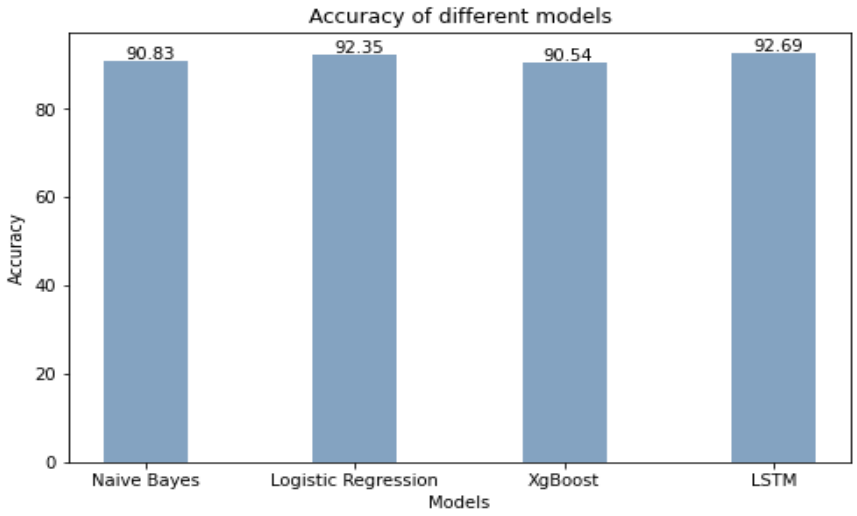
Here we are implementing XGBoost to perform Binary classification on the Amazon food reviews dataset. For example, good review is classified as '1' and bad review is classified as '0'.
We perform required preprocessing on the dataset and then split the data into test and train. Now we have created XGBoost Classifier object with learning rate=0.7. and run the model to classify the test data.

Results

Comparative Analysis

Model	Accuracy	Precision for positives	Precision for negatives	Recall for positives	Recall for negatives	F1 Score	Runtime (sec)
Naive Bayes	0.9083	0.9238	0.7874	0.9713	0.5702	0.9022	276
Logistic Regression	0.9235	0.9410	0.8074	0.9700	0.6738	0.9207	120
LSTM	0.9289	0.9494	0.8040	0.9671	0.7238	0.9273	2124
XGBoost	0.9054	0.9157	0.8122	0.9777	0.5171	0.8965	90

Metrics of various models



Models v/s Accuracy

The table illustrates that

- LSTM model performs better compared to the other algorithms when only accuracy is considered while XgBoost has the least accuracy.
- The highest and lowest accuracies differ only by 2.15% which is very small. So also considering the runtimes of the algorithms, Logistic Regression seems to be the best model.

- Recall for negative reviews is very low compared to positive reviews. This is because the proportion of positive reviews in the training data set is too high.

Conclusion

In this project we have performed sentiment analysis of Amazon food reviews using Naive Bayes, Logistic Regression, LSTM and XGBoost algorithms. We performed comparative analysis among the models to check which model performed better. As shown in results, all the four models have almost same accuracies. LSTM has a slightly higher accuracy than others, but it takes more time to train the model compared to Logistic Regression which could be an important factor if the dataset is huge. The precision and recall for positive reviews in this dataset for all models are close to 1 which indicate that the models predict the positive reviews precisely. Precision and recall for negative reviews are lower compared to positive reviews which shows that negative reviews are not predicted as precisely as positive reviews. So, considering all the metrics including the runtime of the algorithms, Logistic Regression best suits this specific data set.

References

1. Twitter Sentiment Analysis Using Machine Learning for Product Evaluation by Nikhil Yadav; Omkar Kudale; Srishti Gupta; Aditi Rao; Ajitkumar Shitole - IEEE 2020

2. Amazon Fine Food Reviews data set from Kaggle

3. A Complete Sentiment Analysis Project Using Python's Scikit-Learn by Rashida Nasrin Sucky

4. Understanding Logistic Regression in Python by Avinash Navlani

5. Sentiment Analysis explained

6. XGBoost a Deep Dive into Boosting

7. Practical Machine Learning with Python

8. Sentimental analysis using LSTM