FOOD REVIEW SENTIMENT ANALYSIS

MACHINE LEARNING MINI PROJECT

ABSTRACT

Making decisions after carefully going through reviews is a common thing, thanks to the internet. These reviews, based on users' personal experiences are used by other users for this purpose. Sentiment analysis of this data can provide us with a lot of interesting results. Sentiment analysis or opinion mining is nothing but classification of emotions in the review into positive, negative and neutral.Sentiment analysis in web embraces the problem of aggregating data in the web and extraction about opinions. Studying the opinions of customers helps to determine the people feeling about a product and how it is received in the marketOpinion mining is a method of information extraction from text processing to improve or develop the business work by review analysis.

This project classifies food reviews into the categories mentioned. The dataset consists of over half a million reviews from which neutral reviews i.e the reviews with a rating of 3 are ignored. This is because, clearly in a binomial classifier, there is no need to consider the third class. Further, NLP techniques such as deduplication, stemming and lemmatization are used to further pre-process the data. The cleaned or pre-processed data is then used. Different models go about the task of classification differently. The reviews are classified using Logistic Regression and Naïve Bayes algorithms in this project. Bernoulli and Multinomial Naïve Bayes form the NB classification while count of words, TF-IDF and TF-IDF with n gram use logistic regression.

INTRODUCTION

Going out for food has always been an important part of people's lives. When an occasion to be celebrated, we usually do it over lunch or dinner. This makes it critical for a person to choose the restaurant and food they wish to have. In years that have gone by, people chose the restaurants based on various factors and the choice was mostly random. It so happened that their choice failed and they had a not-so-tasty dinner on their birthday. But one thing that has changed through time is that this choice of restaurants has been made easier and more convenient for the user. The user can now logon and browse through tonnes of reviews about restaurants in his/her locality.

One thing that is growing at an outrageous rate in today's world is the Internet. It has seen manifold increase in the number of users over the last decade or so and will continue to grow as time passes. The advent and popularity of the internet has led to people connecting better about almost everything happening around them. One of the most wide-spread example of this could be reviews shared on the internet about various restaurants, movies, etc. These reviews on various forums ensure that the modern internet user takes well-informed decisions about what he watches and where he eats his food from.

In today's age of Artificial Intelligence and automation, we can use these reviews, written in natural language to judge if the overall sentiment of the review is positive or negative. An aggregate idea of the measure of the ratings can be obtained by analysing the reviews further using NLP and classification techniques. Although ratings are a good way to review a place/food, it could sometimes fail because of human errors. Thus, sentiment analysis comes in handy for reliable and trust-worthy classification of reviews.

While writing in a review for a restaurant or a particular dish that they serve, there are terms and words that are frequently used by users. As an example, let us consider a really popular dish served by XYZ Food Arena. It can be seen via user reviews that the word tasty (and its derivatives like tasteful, tastiest, etc.) are used quite often in its reviews. Or consider another dish that is popularly not liked by the people; the reviews will then have words like tasteless, bland, etc. The idea is to identify such negative and positive words and to calculate their 'trust factor' in the review which tells us how good or bad people think the dish is.

LITERATURE REVIEW

No.	Name	Author	Publication	Date
1	Study of supervised machine learning approaches for sentiment analysis	1. Sangharshjit S. Kamble 2. Prof. A. R. Itkikar	International Research Journal of Engineering and Technology	April 2018

Description:

The classification process plays the important role in sentiment analysis. For the classification there are two approach are use lexicon based approach and machine learning approach. This work uses the machine learning approach. More specifically, supervised machine learning approach. Three algorithms viz. Naïve Bayes, Support Vector Machine and Maxent Classifier have been used.

The Naive Bayes algorithm is the simplest and most common algorithm used for train classifier. Naive Bayes classification model calculate the probability of a class, that base on how much time words are appear into the documents. The model works with the BOWs feature extraction which ignores the position of the word in the document. It uses Bayes Theorem to predict the probability that a given feature set belongs to a particular label.

The support vector machine uses the linier properties of probability. Text data are ideally suited for SVM classification because of the sparse nature of text, in which few features are irrelevant, but they tend to be correlated with one another and generally organized into linearly separable categories. SVM can construct a nonlinear decision surface in the original feature space by mapping the data instances nonlinearly to an inner product space where the classes can be separated linearly with a hyper plane.

The Maxent Classifier (known as a conditional exponential classifier) is used to find the maximum entropy. It converts labeled feature sets to vectors using encoding. This encoded vector is then used to calculate weights for each feature that can then be combined to determine the most likely label for a feature set. This classifier is parameterized by a set of X{weights}, which is used to combine the joint features that are generated from a feature-set by an X{encoding}. In particular, the encoding maps each C {(feature set, label)} pair to a vector

It was seen that Naïve Bayes provided a better accuracy than SVM and Maxent Classier models. It is further said that this performance is subject to the dataset and the performance of these algorithms could differ under certain conditions.

No.	Name	Author	Publication	Date
2	Identifying Restaurant Features via Sentiment Analysis on Yelp Reviews	1.Boya Yu 2.Jiaxu Zhou 3.Yi Zhang 4.Yunong Cao	Center for Urban Science & Progress New York University	2018

Description:

In this project, an efficient SVM model has been developed for discriminating positive or negative sentiment on Yelp's reviews with an accuracy of 88.906% on the test set. Other than collecting keywords from different cuisines, our model can also be used for automatically generating ratings for tips (short reviews that are not accompanied with ratings) on Yelp by assigning weights to tips using the sentiment score of words and thus giving more reasonable overall ratings for restaurants.

Based on the analysis, it was found out that for most restaurant types, friendly ranks first before all the other positive comments, indicating that service might weight more than taste when people are judging a restaurant. Also for all categories of cuisines except Japanese and Vietnamese, friendly ranks the first. One possible explanation of this coincidence is that when customers decide where to have a meal, they would usually choose the specific kinds of cuisine they prefer. This kind of 'self-selection' would possibly drive the customers to focus more on the service or environment.

In addition, different characteristics are shown for different restaurant categories. Japanese and Vietnamese food received positive feedback because of freshness, while Korean and Thai restaurants received positive reviews for their spicy food. From the point of customers, a more user-satisfying recommendation and a more considerate appraisal of the restaurants can be expected if Yelp includes more features in its overall evaluation of each restaurant using the technique we have developed.

Although the performance of the model is decent, there are still a lot of spaces for improvement. One of the suggestions for future work is to try other classifiers like boosting, random forest or neural networks check whether it may outperform the SVM model. However, since the next step is to differentiate the impact of different words in varied types of restaurants, a linear-based classifier like SVM or logistic regression would be convenient. If ensemble methods like boosting and random forest or neural networks are applied for classification, the solution of this issue would be essential. For the feature selection part, variants of the tf-idf measure may be considered, or a hybrid model featuring more of a word's inherent meanings.

The method was based on a high-accuracy SVM model, calculating word scores and measuring the polarity. The essential features we discovered might not only help customers to choose their favorite cuisine, but also provide restaurants with their advantages and shortages. On the other hand, similar procedures can be reproduced for reviews and comments in other areas like movie reviews and social media posts. Suppose someone would like to find a movie most renowned for its perfect demonstration of 'love', by operating sentiment analysis and polarity detection on IMDB or rotten tomato movie reviews, he would definitely get what he desired. That would be our anticipations in the future: gathering opinions from people, extracting information from opinions and generating suggestions from information.

No.	Name	Author	Publication	Date
3	Sentiment Analysis of Online Food Reviews using Customer Ratings	1.Sasikala P 2.L.Mary Immaculate Sheela	International Journal of Pure and Applied Mathematics	2018

Description:

Sentiment classification is to select and extract the text features. Feature selection in sentiment analysis is collecting the information from reviews in web and performing the following steps. The methodology of this work follows three steps:

Data preparation: The data preparation step will pre-process the data and removes all the non-textual information and tags. Data pre-processing performs cleaning of data by removing the information like review date and name of the reviewer which is not required for sentiment analysis.

Review Analysis: Finding parts of speech (POS) adjectives and counting the presence and frequency.

Sentiment Classification: A collection of nouns (dog, restaurant, cat, etc), verbs (is, are, would, etc) and adjectives (hot, beautiful, fragrant, etc) are used to decide the subjectivity of the sentence. Frequently used features are identified and stored as vector of text. These features are processed and converted to lower case alphabets. Further all the punctuations and stop words are removed. The word frequency matrix is created in the next step. Opinion words with sentiment polarity as positive, negative, neutral are extracted and a score value is provided. Opinion words are mined as summary and displayed as word cloud. The best reviews are those with score of 5 stars and 4 stars, average reviews are those with score of 3 stars and worst reviews are those with score of 2 stars and 1 star.

Visualization of data:

The results obtained have to be visualised in a way that makes understanding and comprehending the results and its inferences can be made in an easier and more convenient way for the user. The results have been visualised as histograms – based on the user ratings. The histogram plots the number of ratings versus the rating stars. The tallest building in the histogram tells the user about the reviews.

Another method used for visualisation is Word Cloud. Word cloud is an image composed of words used in a particular text or subject, in which the size of each word indicates its frequency or importance. Word clouds (also known as text clouds or tag clouds) work in a simple way: the more a specific word appears in a source of textual data (such as a speech, blog post, or database), the bigger and bolder it appears in the word cloud.

To make the final predictions, the prediction model focuses on sentiment polarity like positive or negative instead of scores. Predictive techniques like Naïve Bayes, Regression can be used to test the data.

No	Name	Author	Publication	Date
4	An Effective Machine learning Approach for Sentiment Analysis of Restaurant Reviews	Rabita Karim	BRAC University, Dhaka, Bangladesh	August 2016

Description:

In machine learning, Naive Bayes classifier is a classification technique based on Bayes' Theorem with an assumption of independence among predictors. In simple terms, a Naive Bayes classifier assumes that the presence of a particular feature in a class is unrelated to the presence of any other feature.

The extension of naive Bayes is called Gaussian Naive Bayes which is used for real-valued attributes. Other functions can be used to estimate the distribution of the data, but the Gaussian (or Normal distribution) is the easiest to work in case where only the mean and the standard deviation from training data is need to be estimated.

There are some instinct points behind choosing Gaussian Naive Bayes First of all, Naive Bayes is a classification algorithm suitable for binary and multiclass classification. In our dataset, the sentiments are representing as binary value. If the input variables are real-valued, a Gaussian distribution is assumed. In which case the algorithm will perform better. Our data set is a real valued dataset.

A decision tree is a flow-chart-like structure, where each internal (non-leaf) node denotes a test on an attribute, each branch represents the outcome of a test, and each leaf (or terminal) node holds a class label. The topmost node in a tree is the root node. Algorithms for constructing decision trees usually work top-down, by choosing a variable at each step that best splits the set of items.

In decision trees, for predicting a class label for a record we start from the root of the tree. We compare the values of the root attribute with record's attribute. On the basis of comparison, we follow the branch corresponding to that value and jump to the next node. We continue comparing our record's attribute values with other internal nodes of the tree until we reach a leaf node with predicted class value.

From this thesis work we have come to a decision that Gaussian Classifier is an effective machine learning model for sentiment analysis. It provides a better prediction for sentiment analysis. In the field of sentiment analysis it is a big challenge to analysis the sarcastic review/text. Machine can to detect sarcasm. Future research can focus on sarcastic expressions which are usually difficult to understand, both by the users' and the computer system. One more challenging issue is the detection of spam contents in users' review. Finally the study can be extended to resolve the problem of co-reference resolution.

PROPOSED SYSTEM

Scope of the Project

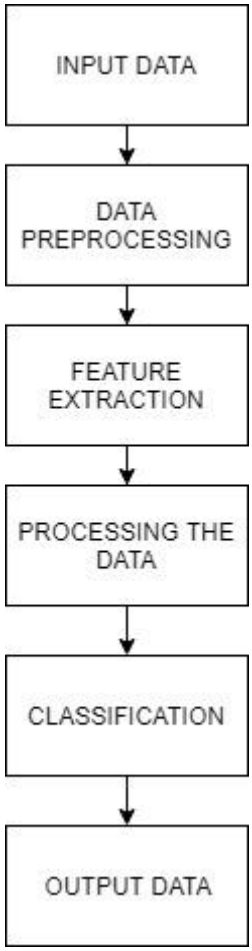
Sentiment analysis is applied at different levels of scope:

- **Sentence level** sentiment analysis obtains the review of the food item at a particular restaurant described in a single sentence.
- **Sub-sentence level** sentiment analysis obtains the review of the food item at a particular restaurant described in a single sentence.

SYSTEM DESIGN

Basic Block Diagram:

Flow Diagram of the analysis model:



Steps:

- Get the input data
- **Data pre-processing -**In dataset we have scores from 1 to 5. 1,2 are considered as negative,3 is considered as neutral and 4,5 are considered as positive. Using data pre-processing we filter out the neutral reviews from our dataset.
- **Feature extraction-** In this we find out the top 20 negative words and top 20 positive words based on the reviews present in out data set.
- **Processing the data -** Based on the algorithm which is selected for classification, the model is trained using the dataset. Aim is to select the dataset and the algorithm which gives us the maximum accuracy of the model.
- **Data Classification -** Data classification refers to the labelling of reviews into one of a number of predefined categories. So, we then distribute the data according to the classes.
- After all the above steps we get the output data.

IMPLEMENTATION DETAILS

Pre-processing of data for Naïve Bayes:

Deduplication:

In computing, data deduplication is a technique for eliminating duplicate copies of repeating data. A related and somewhat synonymous term is single-instance (data) storage. This technique is used to improve storage utilization and can also be applied to network data transfers to reduce the number of bytes that must be sent. In the deduplication process, unique chunks of data, or byte patterns, are identified and stored during a process of analysis. As the analysis continues, other chunks are compared to the stored copy and whenever a match occurs, the redundant chunk is replaced with a small reference that points to the stored chunk. Given that the same byte pattern may occur dozens, hundreds, or even thousands of times (the match frequency is dependent on the chunk size), the amount of data that must be stored or transferred can be greatly reduced.

Text Pre-processing:

- Stemming:
  Stemming is the process of reducing a word to its word stem that affixes to suffixes and prefixes or to the roots of words known as a lemma. Stemming is important in natural language understanding (NLU) and natural language processing (NLP). Recognizing, searching and retrieving more forms of words returns more results. When a form of a word is recognized it can make it possible to return search results that otherwise might have been missed. That additional information retrieved is why stemming is integral to search queries and information retrieval.

- Lemmatization:
  Lemmatization is the grouping together of different forms of the same word. In search queries, lemmatization allows end users to query any version of a base word and get relevant results. Because search engine algorithms use lemmatization, the user is free to query any inflectional form of a word and get relevant results. For example, if the user queries the plural form of a word (routers), the search engine knows to also return relevant content that uses the singular form of the same word (router).

- Stopword removal:
  Stopwords are the English words which does not add much meaning to a sentence. They can safely be ignored without sacrificing the meaning of the sentence. For example, the words like the, he, have etc. Such words are already captured this in corpus named corpus. We first download it to our python environment.

Bag of Words

The bag-of-words model is one of the simplest language models used in NLP. It makes an unigram model of the text by keeping track of the number of occurences of each

word. This can later be used as a features for Text Classifiers. In this bag-of-words model you only take individual words into account and give each word a specific subjectivity score. This subjectivity score can be looked up in a sentiment lexicon. If the total score is negative the text will be classified as negative and if its positive the text will be classified as positive. It is simple to make, but is less accurate because it does not take the word order or grammar into account.

A simple improvement on using unigrams would be to use unigrams + bigrams. That is, not split a sentence after words like “not”,”no”,”very”, “just” etc. It is easy to implement but can give significant improvement to the accuracy. The sentence “This dish is not good” will be interpreted as a positive sentence, unless such a construct is implemented. Another example is that the sentences “This dish is very good” and “This dish is good” will have the same score with a unigram model of the text, but not with an unigram + bigram model.

Bernoulli’s Naïve Bayes:

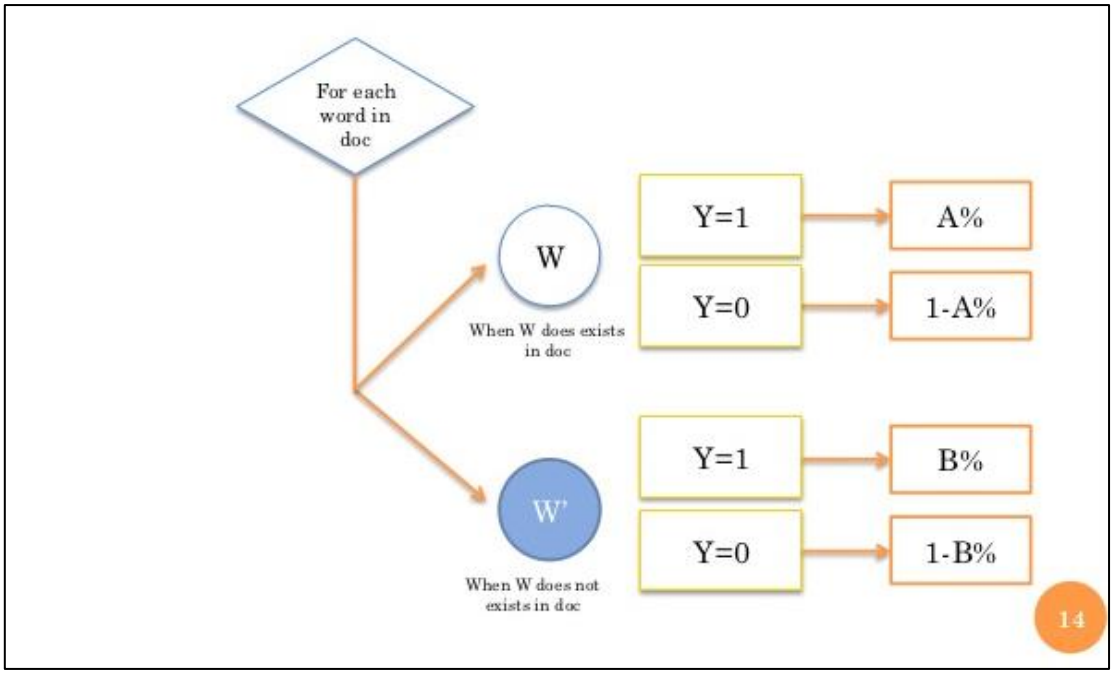
Bernoulli’s Naïve Bayes implements the naive Bayes training and classification algorithms for data that is distributed according to multivariate Bernoulli distributions; i.e., there may be multiple features but each one is assumed to be a binary-valued (Bernoulli, boolean) variable. Therefore, this class requires samples to be represented as binary-valued feature vectors; if handed any other kind of data, a instance Bernoulli’s Naïve Bayes may binarize its input (depending on the binarize parameter).

The decision rule for Bernoulli naive Bayes is based on:

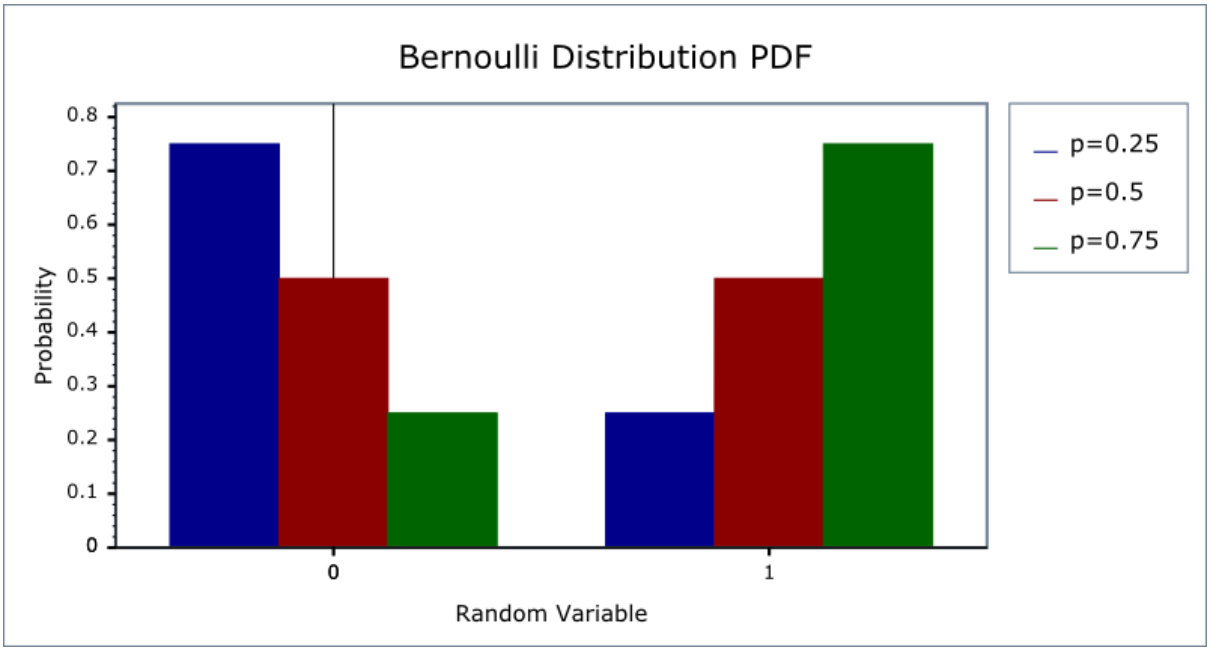
$$P(xi|y)=P(i|y)xi+(1−P(i|y))(1−xi)$$

which differs from multinomial NB’s rule in that it explicitly penalizes the non-occurrence of a feature i that is an indicator for class y, where the multinomial variant would simply ignore a non-occurring feature.

In the case of text classification, word occurrence vectors (rather than word count vectors) may be used to train and use this classifier. Bernoulli’s Naïve Bayes might perform better on some datasets, especially those with shorter documents. It is advisable to evaluate both models, if time permits.



Graph:



Multinomial Naïve Bayes:

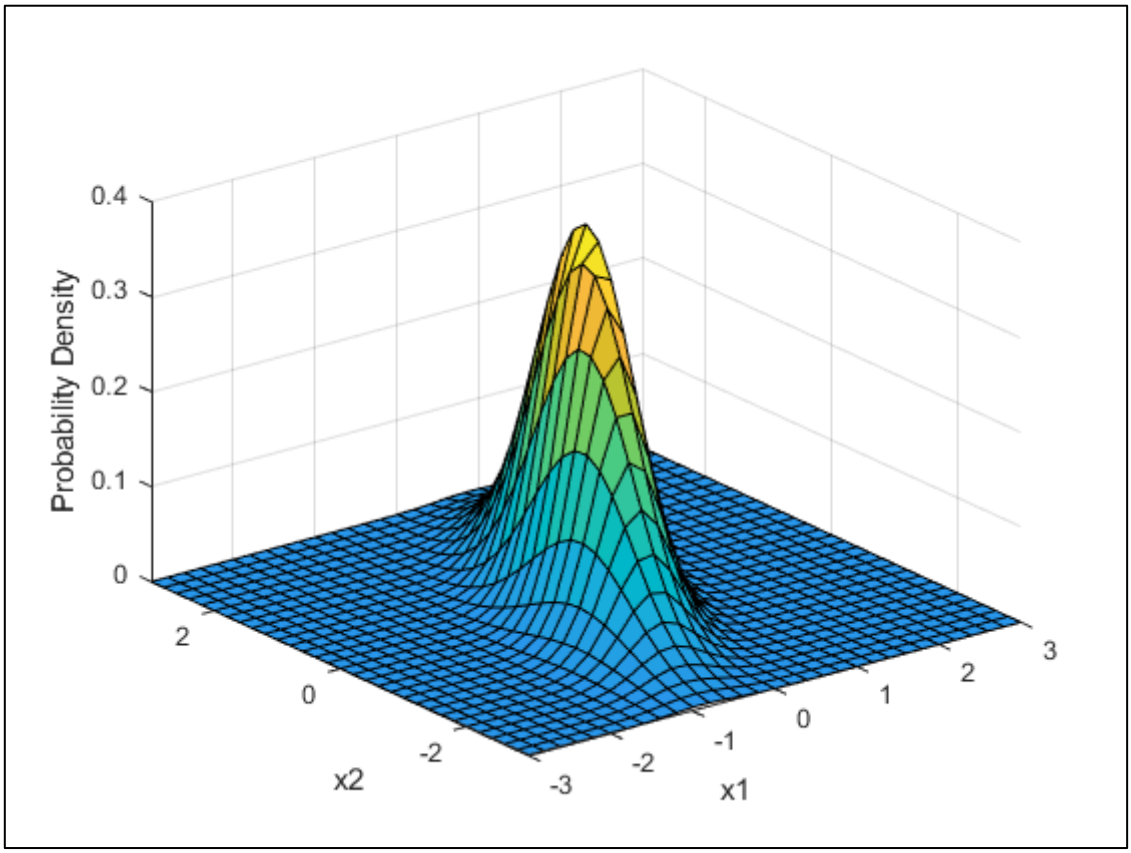
In contrast to the multi-variate Bernoulli event model, the multinomial model captures word frequency information in documents. Consider, for example, the occurrence of numbers in the Reuters newswire articles; our tokenization maps all strings of digits to a common token. Since every news article is dated, and thus has a number, the number token in the multi-variate Bernoulli event model is uninformative. However, news articles about earnings tend to have a lot of numbers compared to general news articles. Thus, capturing frequency

information of this token can help classification. In the multinomial model, a document is an ordered sequence of word events, drawn from the same vocabulary V .

We assume that the lengths of documents are independent of class. We again make a similar naive Bayes assumption: that the probability of each word event in a document is independent of the word's context and position in the document. Thus, each document di is drawn from a multinomial distribution of words with as many independent trials as the length of di. This yields the familiar "bag of words" representation for documents. Define Nit to be the count of the number of times word wt occurs in document di. Then, the probability of a document given its class from Equation 1 is simply the multinomial distribution:

$$\mathrm{P}(d_i|c_j;\theta) = \mathrm{P}(|d_i|)|d_i|! \prod_{t=1}^{|V|} \frac{\mathrm{P}(w_t|c_j;\theta)^{N_{it}}}{N_{it}!}.$$

The parameters of the generative component for each class are the probabilities for each word, written$_{\theta wt|cj}$ = P(wt|cj; θ), where 0 ≤ θwt |cj ≤ 1 and t θwt |cj = 1.



Multinomial Distribution

Pre-processing for Logistic Regression:

In dataset we have scores from 1 to 5. 1,2 are considered as negative,3 is considered as neutral and 4,5 are considered as positive. Using data pre-processing we filter out the neutral reviews from our dataset.

We first calculate the helpful percent by using the helpfulness numerator and helpfulness denominator. Formula is given as:

% helpfulness=helpfulness numerator/helpfulness denominator

Now we calculate the % Upvote. It depends upon the % helpfulness attribute in the dataset.The following table is used:

Upvote	% Helpfulness
0	Empty
0.2	0-20%
0.4	20-40%%
0.6	40-60%
0.8	60-80%
1	80-100%

Logistic Regression:

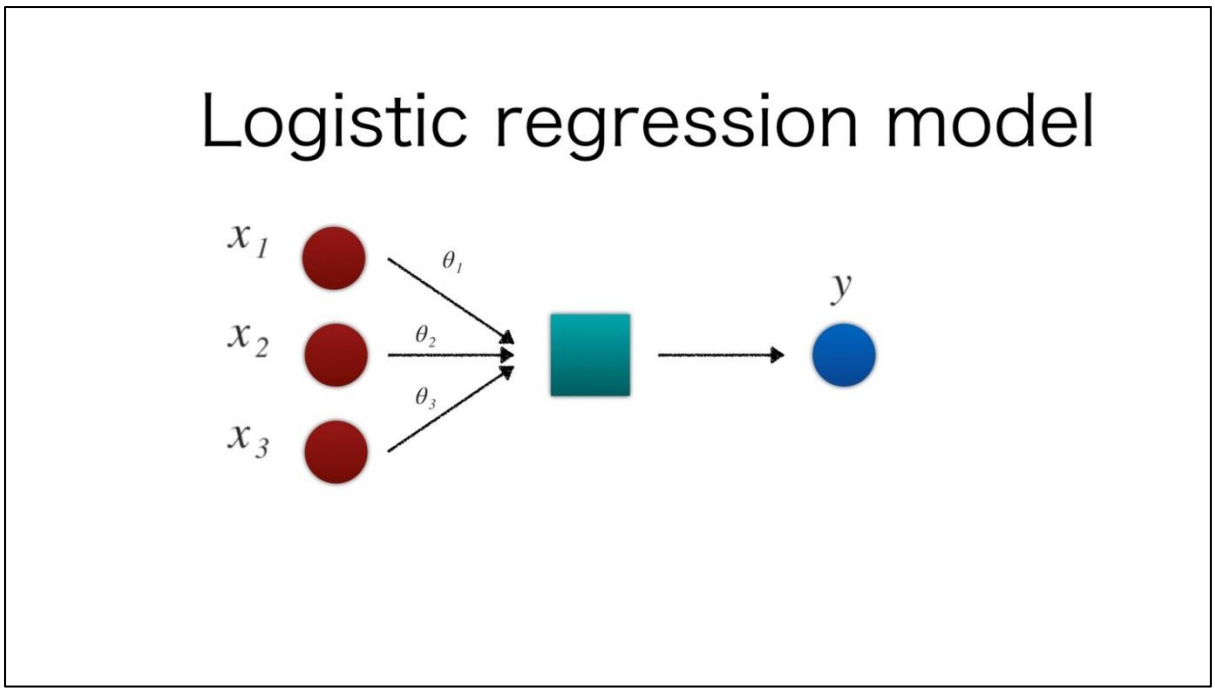
Logistic Regression was used in the biological sciences in early twentieth century. It was then used in many social science applications. Logistic Regression is used when the dependent variable(target) is categorical.

For example,

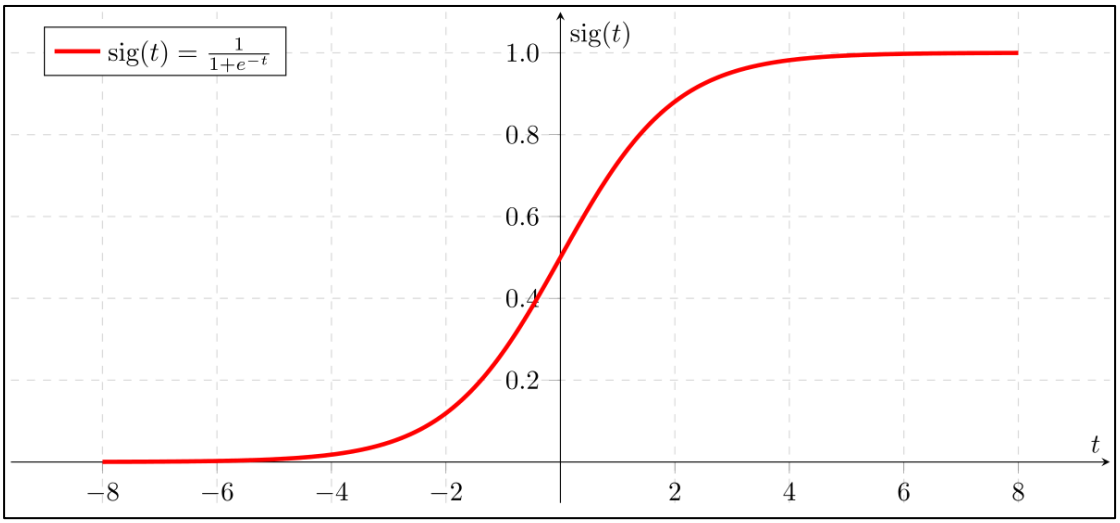
- To predict whether an email is spam (1) or (0)
- Whether the tumor is malignant (1) or not (0)

Consider a scenario where we need to classify whether an email is spam or not. If we use linear regression for this problem, there is a need for setting up a threshold based on which classification can be done. Say if the actual class is malignant, predicted continuous value 0.4 and the threshold value is 0.5, the data point will be classified as not malignant which can lead to serious consequence in real time.

From this example, it can be inferred that linear regression is not suitable for classification problem. Linear regression is unbounded, and this brings logistic regression into picture. Their value strictly ranges from 0 to 1.


Logistic regression model
x_1
x_2
x_3
θ_1
θ_2
θ_3
y

Graph:


$\text{sig}(t) = \frac{1}{1+e^{-t}}$
$\text{sig}(t)$
1.0
0.8
0.6
0.4
0.2
t
−8
−6
−4
−2
2
4
6
8

TF-ID:

TF-IDF (term frequency–inverse document frequency) vectorises words by taking into account the frequency of a word in a given document and the frequency between documents.

Mathematically, the importance increases proportionally to the number of times a word appears in the document but is offset by the frequency of the word in the corpus. To break it down:

Term Frequency: More frequent terms in a target article, the higher the score

$$\mathrm{tf}(t,\ d) = N_{\mathrm{term}}$$

Inverse Document Frequency**:** The more common in other articles, the lower the score

$$\mathrm{idf}(t,\ D) = 1 + \log\left(\frac{1 + N_{\mathrm{Documents}}}{1 + N_{\mathrm{Documents\ that\ contain\ term}}}\right)$$

$$\text{tf-idf}(t,\ d,\ D) = \mathrm{tf}(t,\ d) \cdot \mathrm{idf}(t,\ D)$$

So if a word is common in most documents it is suppressed and rare words are given more influence showing they are highly specific for a particular document.

N-Gram:

Basically, an N-gram model predicts the occurrence of a word based on the occurrence of its N – 1 previous words. So here we are answering the question – how far back in the history of a sequence of words should we go to predict the next word? For instance, a bigram model (N = 2) predicts the occurrence of a word given only its previous word (as N – 1 = 1 in this case). Similarly, a trigram model (N = 3) predicts the occurrence of a word based on its previous two words (as N – 1 = 2 in this case).

First of all, it can help in deciding which N-grams can be chunked together to form single entities.

It can also help to make spelling error corrections. For instance, the sentence “drink cofee” could be corrected to “drink coffee” if you knew that the word “coffee” had a high probability of occurrence after the word “drink” and also the overlap of letters between “cofee” and “coffee” is high.

Word Count:

Consider a Corpus C of D documents {d1,d2…..dD} and N unique tokens extracted out of the corpus C. The N tokens will form our dictionary and the size of the Count Vector matrix

M will be given by D X N. Each row in the matrix M contains the frequency of tokens in document D(i).

Let us understand this using a simple example.

D1: He is a lazy boy. She is also lazy.

D2: Neeraj is a lazy person.

The dictionary created may be a list of unique tokens(words) in the corpus =['He','She','lazy','boy','Neeraj','person']

Here, D=2, N=6

The count matrix M of size 2 X 6 will be represented as –

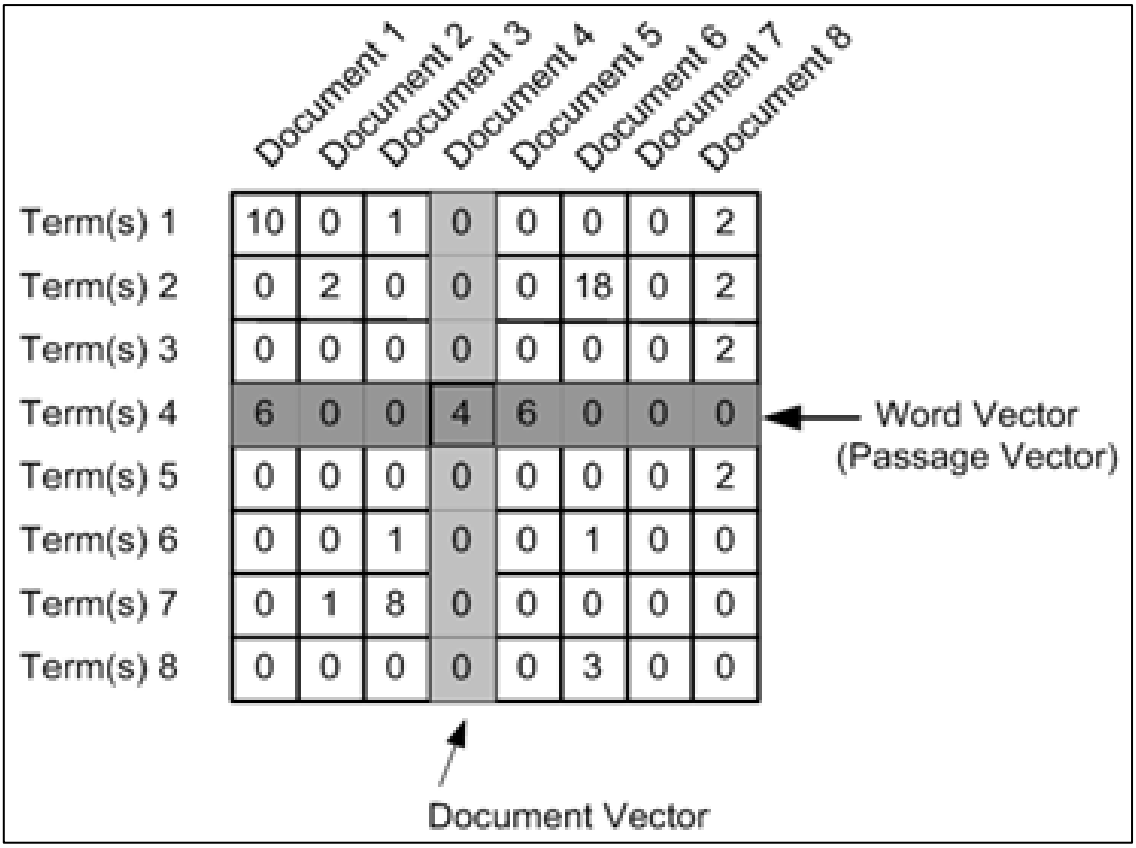
	He	She	lazy	boy	Neeraj	person
D1	1	1	2	1	0	0
D2	0	0	1	0	1	1

Now, a column can also be understood as word vector for the corresponding word in the matrix M. For example, the word vector for 'lazy' in the above matrix is [2,1] and so on.Here, the *rows* correspond to the *documents* in the corpus and the *columns* correspond to the *tokens* in the dictionary. The second row in the above matrix may be read as – D2 contains 'lazy': once, 'Neeraj': once and 'person' once.

Now there may be quite a few variations while preparing the above matrix M. The variations will be generally in-

1. The way dictionary is prepared.
   Why? Because in real world applications we might have a corpus which contains millions of documents. And with millions of document, we can extract hundreds of millions of unique words. So basically, the matrix that will be prepared like above will be a very sparse one and inefficient for any computation. So an alternative to using every unique word as a dictionary element would be to pick say top 10,000 words based on frequency and then prepare a dictionary.
2. The way count is taken for each word.
   We may either take the frequency (number of times a word has appeared in the document) or the presence(has the word appeared in the document?) to be the entry in the count matrix M. But generally, frequency method is preferred over the latter.

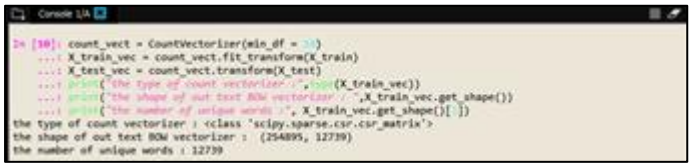
Below is a representational image of the matrix M for easy understanding.


Document 1
Document 2
Document 3
Document 4
Document 5
Document 6
Document 7
Document 8
Term(s) 1
Term(s) 2
Term(s) 3
Term(s) 4
Term(s) 5
Term(s) 6
Term(s) 7
Term(s) 8
10 0 1 0 0 0 0 2
0 2 0 0 0 18 0 2
0 0 0 0 0 0 0 2
6 0 0 4 6 0 0 0
0 0 0 0 0 0 0 2
0 0 1 0 0 1 0 0
0 1 8 0 0 0 0 0
0 0 0 0 0 3 0 0
Word Vector
(Passage Vector)
Document Vector

RESULT ANALYSIS

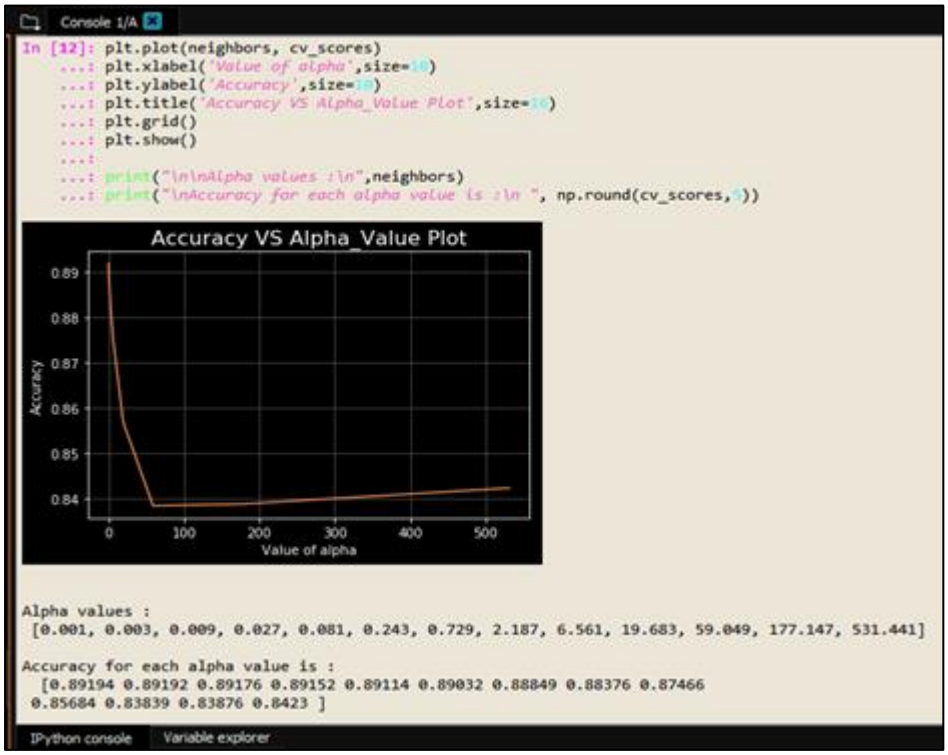
1] NAÏVE BAYES CLASSIFIER

BAG OF WORDS (BOW)

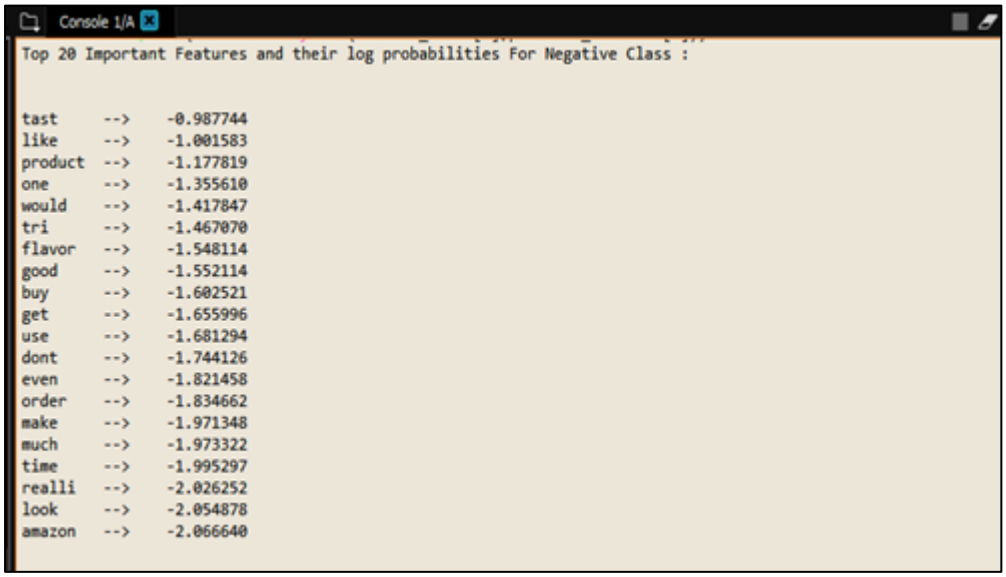


A] BERNOULLI NAÏVE BAYES CLASSIFIER

The optimal value of alpha is 0.001.



The Test Accuracy of the Bernoulli naive Bayes classifier for alpha = 0.001 is 89.365714%



```
Console 1/A

Top 20 Important Features and their log probabilities For Positive Class :


like      -->    -1.175476
tast      -->    -1.204468
love      -->    -1.260625
good      -->    -1.265811
great     -->    -1.295596
flavor    -->    -1.423290
one       -->    -1.467773
use       -->    -1.479067
tri       -->    -1.520718
product   -->    -1.537382
make      -->    -1.657849
get       -->    -1.684409
buy       -->    -1.893541
time      -->    -1.902954
would     -->    -1.928464
realli    -->    -1.967278
best      -->    -1.989593
amazon    -->    -1.997007
find      -->    -2.011305
price     -->    -2.015592
```

Evaluating Accuracy , F1-Score , Precision , Recall , TPR , FPR , TNR , FNR:

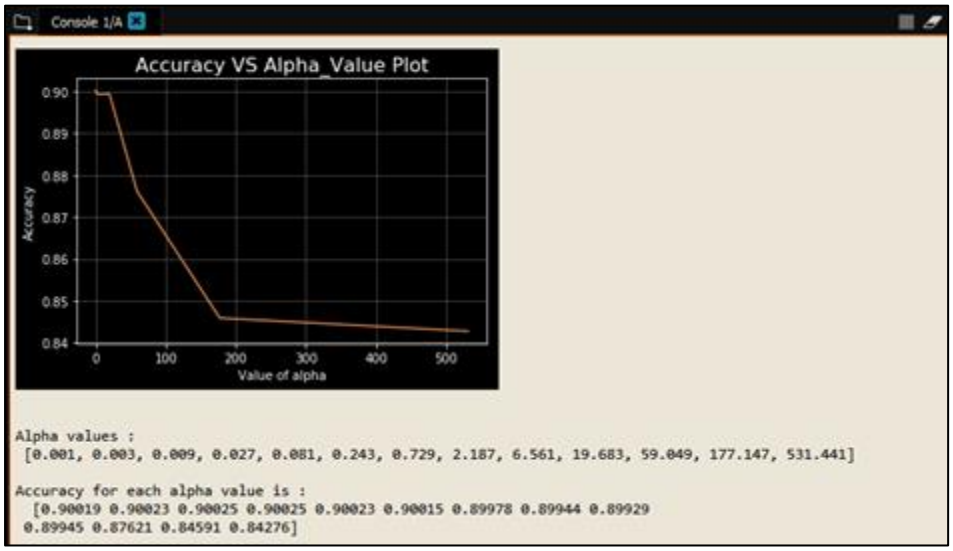
```
Console 1/A

TPR of the Bernoulli naive Bayes classifier for alpha = 0.001 is :  0.937157
FPR of the Bernoulli naive Bayes classifier for alpha = 0.001 is :  0.340881
TNR of the Bernoulli naive Bayes classifier for alpha = 0.001 is :  0.659119
FNR of the Bernoulli naive Bayes classifier for alpha = 0.001 is :  0.062843
```

```
Console 1/A

The Test Accuracy of the Bernoulli naive Bayes classifier for alpha = 0.001 is 89.365714%

The Test Precision of the Bernoulli naive Bayes classifier for alpha = 0.001 is 0.937157

The Test Recall of the Bernoulli naive Bayes classifier for alpha = 0.001 is 0.936801

The Test F1-Score of the Bernoulli naive Bayes classifier for alpha = 0.001 is 0.936979
```

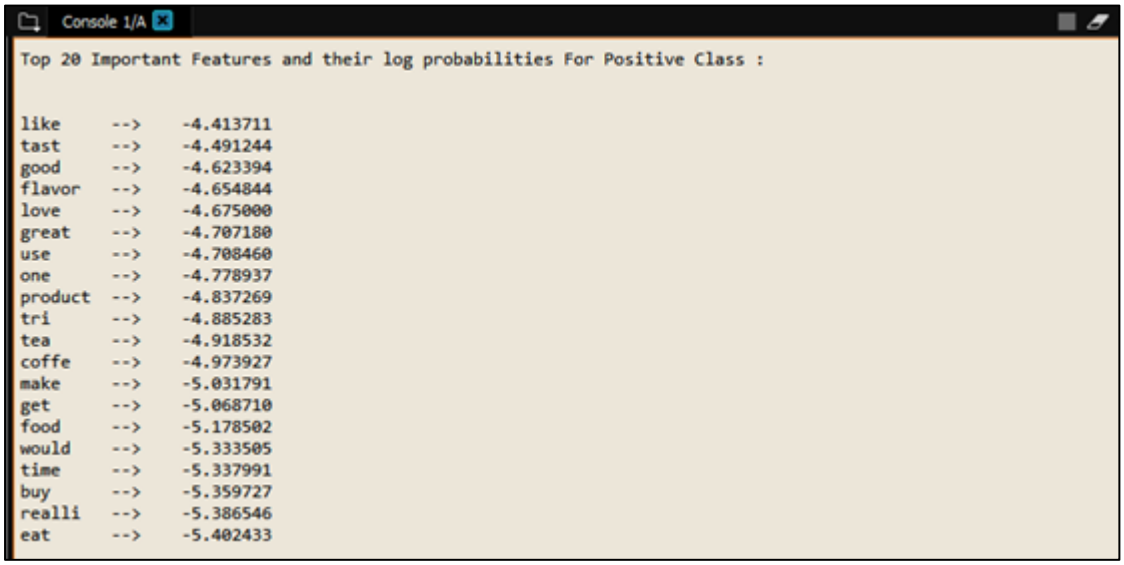
Confusion Matrix:

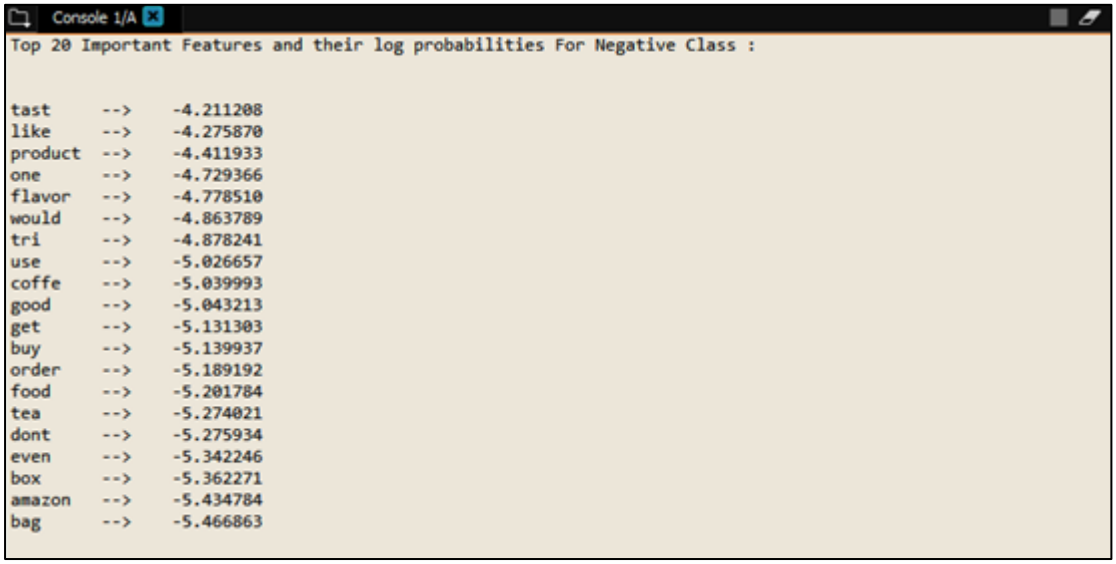
B] MULTINOMIAL NAÏVE BAYES CLASSIFIER

The optimal value of alpha is 0.009.

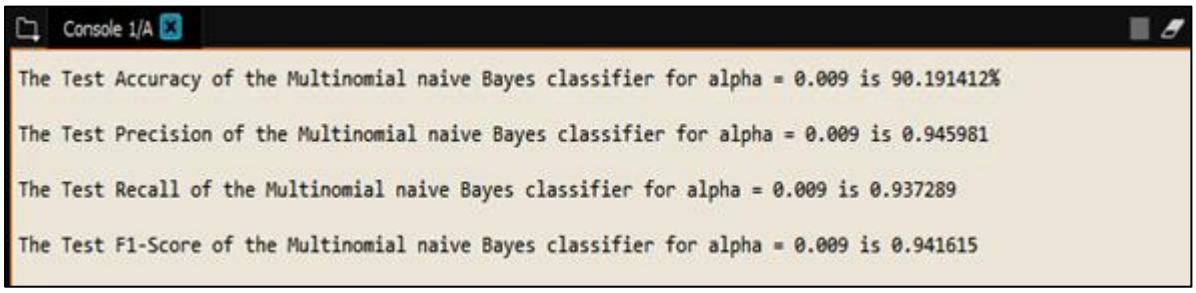


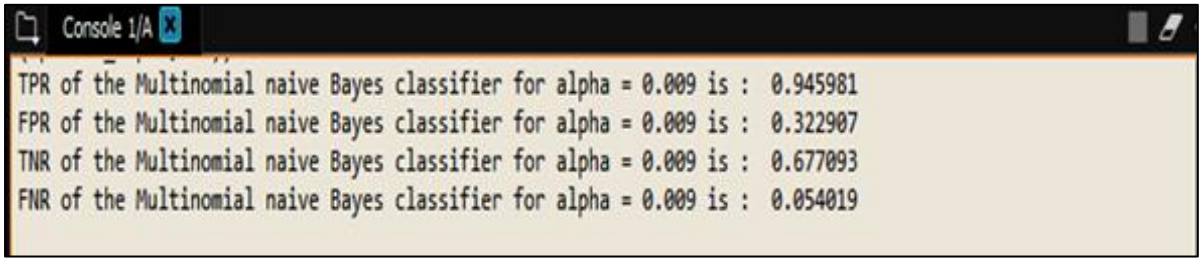
The Test Accuracy of the Multinomial naive Bayes classifier for alpha = 0.009 is 90.191412%



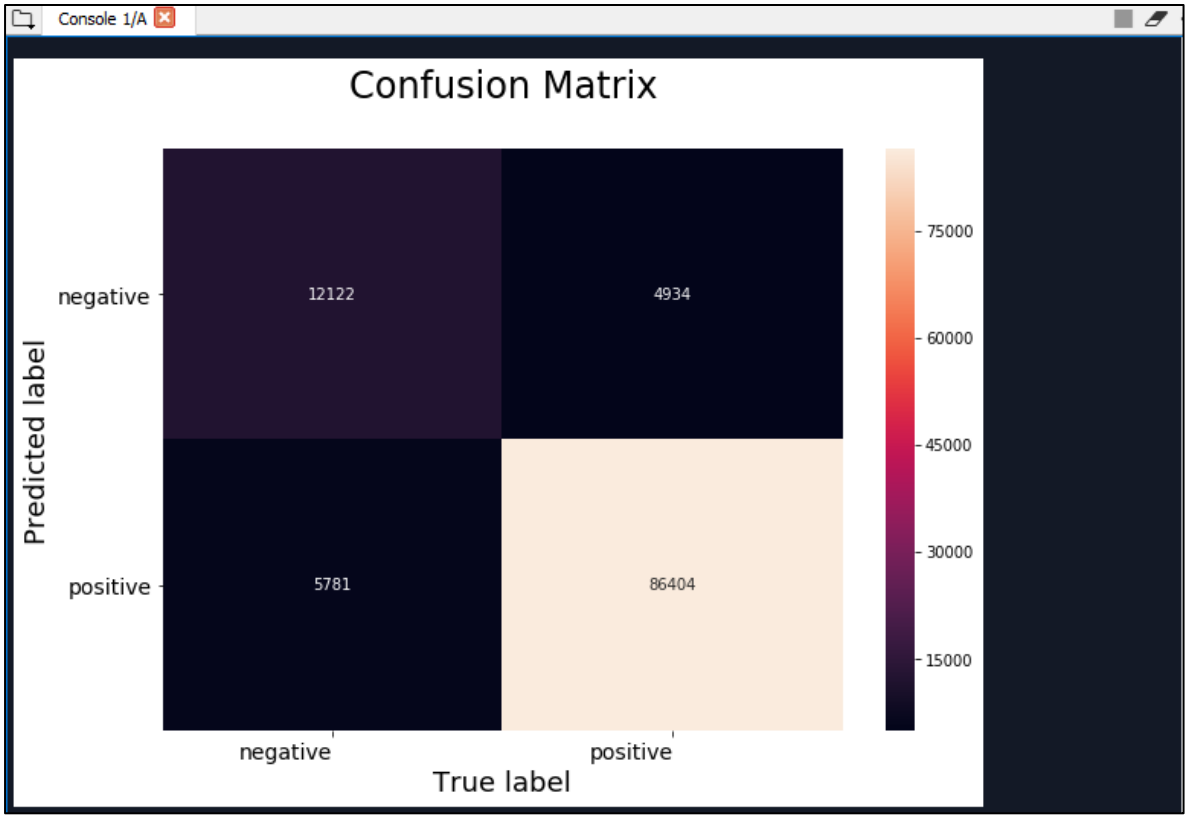


Evaluating Accuracy , F1-Score , Precision , Recall , Confusion_Matrix , TPR , FPR , TNR , FNR:



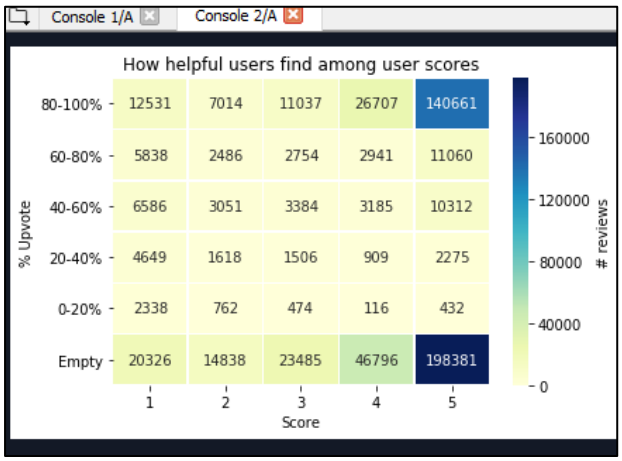


Confusion Matrix:



2] LOGISTIC REGRESSION

HEATMAP:

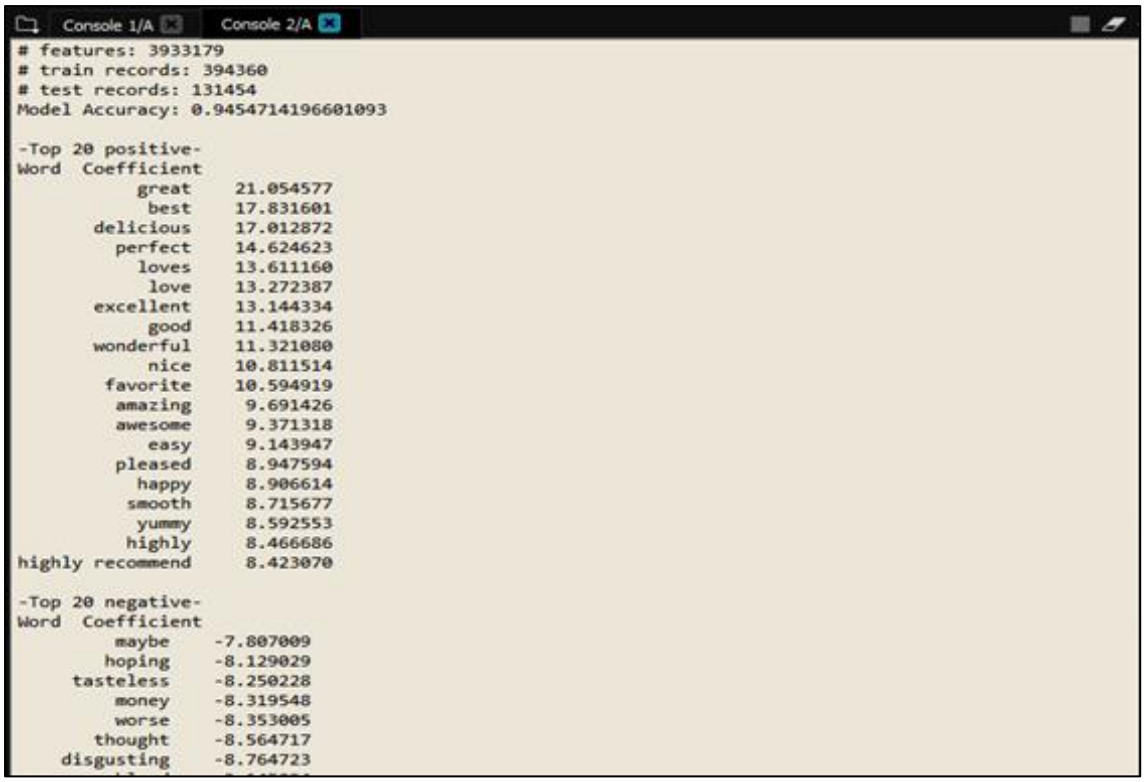


A] Using Word-Count Method (CountVectorizer)

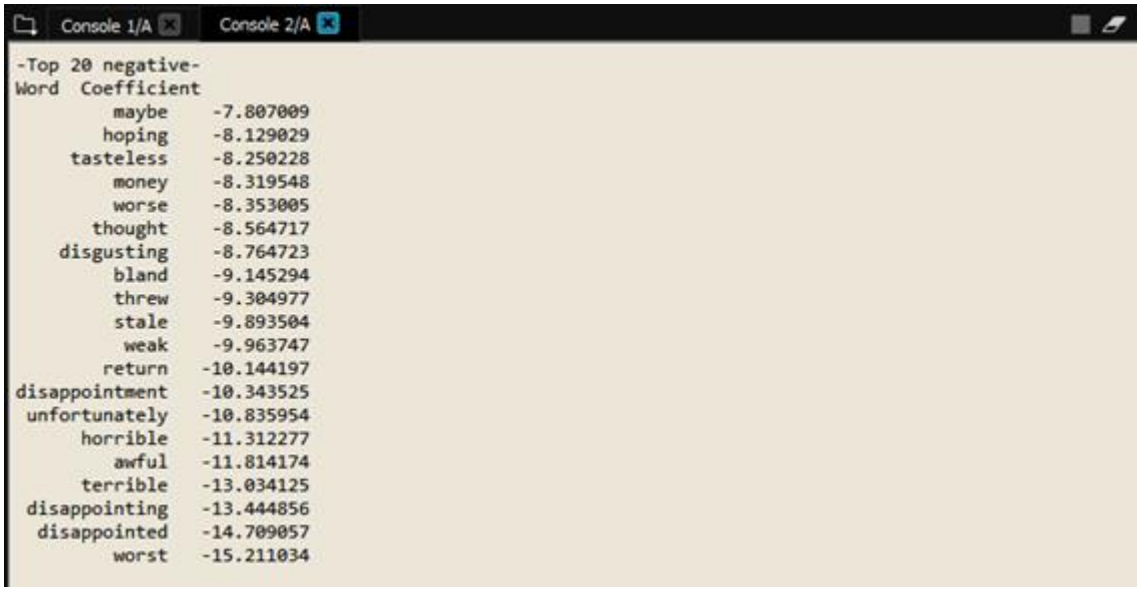
```
Console 1/A  Console 2/A
# features: 114969
# train records: 394360
# test records: 131454
C:\Users\Dhruvin\Anaconda3\lib\site-packages\sklearn\linear_model\logistic.py:433: FutureWarning:
Default solver will be changed to 'lbfgs' in 0.22. Specify a solver to silence this warning.
  FutureWarning)
C:\Users\Dhruvin\Anaconda3\lib\site-packages\sklearn\svm\base.py:922: ConvergenceWarning: Liblinear
failed to converge, increase the number of iterations.
  "the number of iterations.", ConvergenceWarning)
Model Accuracy: 0.938769455474919

-Top 20 positive-
Word  Coefficient
   pleasantly     3.327996
emeraldforest     3.192992
      chedder     3.008159
      easiest     2.560361
      solving     2.449253
      blowout     2.437108
    addicting     2.370756
     heartier     2.369831
   unwrapping     2.294373
     drawback     2.291376
        herrs     2.291095
       hooked     2.289796
       skewed     2.272785
       hahaha     2.260465
     downside     2.249587
   correction     2.234189
        whisk     2.189918
       ration     2.185779
       bertie     2.183490
   dishwasher     2.166738
```

```
Console 1/A  Console 2/A
-Top 20 negative-
Word  Coefficient
   unfinished    -2.465116
     tastless    -2.490250
      weakest    -2.594734
    overrated    -2.602914
    cancelled    -2.692140
  unappealing    -2.729629
      furious    -2.739071
   oversalted    -2.751613
    deceptive    -2.821843
disappointing    -2.822209
  undrinkable    -2.825383
  embarrassed    -2.827563
    redeeming    -2.870237
        worst    -2.891056
dissapointing    -2.915166
          ick    -2.946322
        budda    -2.957805
        280mg    -2.975297
      3095826    -3.026555
 unacceptable    -3.290567
```

B] Using TFIDF+NGRAM



```
Console 1/A   Console 2/A
# features: 3933179
# train records: 394360
# test records: 131454
Model Accuracy: 0.9454714196601093

-Top 20 positive-
Word  Coefficient
           great    21.054577
            best    17.831601
       delicious    17.012872
         perfect    14.624623
           loves    13.611160
            love    13.272387
       excellent    13.144334
            good    11.418326
       wonderful    11.321080
            nice    10.811514
        favorite    10.594919
         amazing     9.691426
         awesome     9.371318
            easy     9.143947
         pleased     8.947594
           happy     8.906614
          smooth     8.715677
           yummy     8.592553
          highly     8.466686
highly recommend     8.423070

-Top 20 negative-
Word  Coefficient
          maybe    -7.807009
         hoping    -8.129029
     tasteless    -8.250228
          money    -8.319548
          worse    -8.353005
        thought    -8.564717
    disgusting    -8.764723
```



```
Console 1/A   Console 2/A
-Top 20 negative-
Word  Coefficient
          maybe    -7.807009
         hoping    -8.129029
     tasteless    -8.250228
          money    -8.319548
          worse    -8.353005
        thought    -8.564717
    disgusting    -8.764723
         bland    -9.145294
         threw    -9.304977
         stale    -9.893504
          weak    -9.963747
        return   -10.144197
disappointment   -10.343525
 unfortunately   -10.835954
      horrible   -11.312277
         awful   -11.814174
      terrible   -13.034125
 disappointing   -13.444856
  disappointed   -14.709057
         worst   -15.211034
```

FUTURE SCOPE

Emotion detection

Emotion detection aims at detecting emotions like, happiness, frustration, anger, sadness, and the like. Many emotion detection systems resort to lexicons (i.e. lists of words and the emotions they convey) or complex machine learning algorithms.

Multilingual sentiment analysis

Multilingual sentiment analysis can be a difficult task. Usually, a lot of preprocessing is needed and that pre-processing makes use of a number of resources. Most of these resources are available online (e.g. sentiment lexicons), but many others have to be created (e.g. translated corpora or noise detection algorithms). The use of the resources available requires a lot of coding experience and can take long to implement.

REFERENCES

1. https://www.irjet.net/archives/V5/i4/IRJET-V5I4673.pdf
2. https://arxiv.org/ftp/arxiv/papers/1709/1709.08698.pdf
3. https://acadpubl.eu/hub/2018-119-15/2/373.pdf
4. http://dspace.bracu.ac.bd/xmlui/bitstream/handle/10361/9023/13101248_CSE.pdf?sequence=1&isAllowed=y
5. https://www.coursera.org/learn/machine-learning
6. Machine Learning A-Z course.
7. www.udemy.com/machinelearning