Predictive Power of Social Media

Vikas Rahar

Submitted for the Degree of Master of Science in

MSc. Data Science & Analytics

Department of Computer Science
Royal Holloway University of London
Egham, Surrey TW20 0EX, UK

September 7, 2020

Declaration

This report has been prepared on the basis of my own work. Where other published and unpublished source materials have been used, these have been acknowledged.

Word Count: 10422

Student Name: Vikas Rahar

Date of Submission: 7 September 2019

Signature:

Acknowledgment

I would like to express my gratitude to my supervisor Colombo Nicolo who has shaped, guided and refined my work through this experiment. His subject expertise, his intuition on the areas to explore and his patience as a teacher played a major part in making this project what it is today.

Abstract

Microblogging on the social network today has become a very popular communication way among the people. These microblogging platforms are rich sources of data for opinions mining and sentiment analysis about the certain entity in very short messages. Some of the existing popular microblogs platforms such as Facebook, Twitter, Instagram and so on. Amongst all these existing microblogging platforms, Twitter attains the maximum amount of attention in the field of the sentimental analysis research study. In the field of technology and research, sentimental and opinion analysis becomes the major issue. On social networking platforms, the number of users is growing day by day, generating a huge amount of data in the form of text, images, audio and video. To find the sentimental analysis labels such as positive, negative or neutral, the sentimental analysis must be performed on texts in the form of messages or tweets.

For this purpose, Kaggle dataset sentiment140 is used for analysis the sentimental labels, emotions and predicting the labels for future tweets. VADER Lexicon is being used for detecting the sentiment labels of the text field of tweet. VADER uses a combination of lexicon, a list of lexical features which are labelled as positive or negative according to their semantic orientation. NRCLex is being used for detecting the emotions of each word. In NRCLex, the words of each tweet text field are categorised amongst of its list which has 8 emotions in it. Next the sentiment orientation is identified using the textblob which a standard NLTK libraray. Once labels are identified for each tweet, I trained and tested the supervised machine algorithms such as Support Vector machines (SVM), Naïve Bayes (Multinominal and Bernoulli) used to compare the precision, accuracy using the evaluation matrix and confusion matrix. These algorithms are saved loaded back using the pickle which are used to predict the sentiment labels of future tweets.

Research Problem

Problem statement

Microblogging is a type of blogging which consists of limited number of words, with words limits restricted by the respective microblogging sites. Users express ideas, thoughts, opinion and sentiments in a smaller number of words, which revolutionary things in the world of technology. To connect with the relatives and rest of the world via the microblogging sites such as Twitter, Facebook, Instagram etc. Here emotions and sentiments come into the play that everyone can share in the time they feel and want to be shared. Sentiment analysis is to determine the opinion of user related to some event or the statement describe the emotion of the user i.e. what the user thinks about it.

The research on sentimental analysis has been going for a long time. In the field of technology and research sentimental analysis and emotions become the major issues. On social networking platforms, the number of users is growing day by day, generating a huge amount of data in the form of text, images, audio and video. To find the sentimental analysis labels such as positive, negative or neutral, the sentimental analysis must be performed on texts in the form of messages or tweets.

Objectives

The objective of the dissertation has been discussed in to following points:

1. Finding/detecting the labels for each tweet text field using the VADER Lexicon.
2. Plotting the word cloud of the most frequent words after finding the labels in complete datasets as well as in both positive and negative tweets.
3. Finding the emotions of each words in the dataset as well in the positive and negative sentiment labels tweets (such as detect the depression, happiness and surprise of the users) and plotting the count of words in each emotion categorically.
4. Detecting the sentiment labels of each model using textblob and with this, comparing different supervised machine algorithms such as SVM, Naïve Bayes and logistic regression, and saving the models using pickle.
5. Predicts the sentiment label of the future tweets.

Research Methodology

The main aim of this dissertation is to detect sentiment labels, emotions and predict the labels of future tweets.

The methodology followed is:

1. For the enormously large dataset, detect the sentiment labels of each tweet using the VADER lexicon.
2. Find the emotions of every words in every tweet using the NRCLex.
3. Predict the sentiment labels of future tweets using the different supervised machine algorithms.
4. Comparing the different models using the evaluation matrix and confusion matrix.

Table of Contents

1 Introduction 9

2 Background Research 12

2.1 VADER Lexicon 12

2.2 NRCLex 12

2.3 TEXTBLOB 12

2.4 TF – IDF Vectoriser 13

2.5 Confusion Matrix 14

2.6 Naives Bayes 15

2.6.1 Bernoulli Model 16

2.6.2 Multinomial Model 16

2.7 SVM Model 17

2.7.1 LinearSVC 18

2.8 Logistic Regression 19

2.9 Pickle 20

3 Methodology 21

3.1 Data Collection 21

3.2 Pre-processing 21

3.2.1 Normalisation 21

3.2.2 Removal of stop words 22

3.3 Sentiment Identification 23

3.4 Emotion Identification 23

3.5 Prediction 23

4 Methodology & Results 27

4.1 Tools for Implementation 27

4.2 Implementation 30

4.3 Results 31

5 Challenges **38**

6 Self Assessment **39**

7 Conclusion **40**

References **41**

Table of Figures

Figure 1.1: Most popular social media sites in 2020 10

Figure 1.2: Twitter Logo 10

Figure 1.3 Sentiment Analysis 11

Figure 1.4: Sample tweets for the most common sentiment class/labels 11

Figure 2.1: Confusion matrix 14

Figure 2.2: SVM separating hyperplane 17

Figure 3.1: List of 179 NLTK stop words 22

Figure 3.2: Architecture for Lexicon classifier system 23

Figure 4.1: NumPy capabilities and application areas 29

Figure 4.2: dataset structure 30

Figure 4.3: Sentiment labels using VADER 31

Figure 4.4: Sentiment labels using Textblob 31

Figure 4.5**:** Bar plot of 25 top words in positive and negative tweets respectively 32

Figure 4.6: Word cloud of frequent words in dataset 32

Figure 4.7: Word clouds of frequent words in the positive tweets 33

Figure 4.8: Word cloud of frequent words in the negative tweets 33

Figure 4.9: Each emotion count in the dataset 34

Figure 4.10: Each emotion count in the positive tweets 34

Figure 4.11: Each emotion count in the positive tweets 35

Figure 4.12: Confusion Matrix for BernoulliNB model 35

Figure 4.13: Confusion Matrix for MultinomialNB 36

Figure 4.14: Confusion Matrix for LinearSVC model 36

Figure 4.15: Confusion Matrix for Logistic Regression model 37

Figure 4.16: Sample inputs used in LR model 37

Chapter 1: Introduction

1.1 Social Networking

Social networking service (also social networking site or social media) is an online platform which people use to build social networks or social relationships with other people who share similar personal or career interests, activities, backgrounds or real-life connections (Obar, Wildman, 2015). Some of the most common social networking sites which are free and easy to access are Twitter, Facebook, Instagram, TikTok, Snapchat, and LinkedIn. People use these platforms of social media to meet new friends, build relationships and even marry. While some take advantage of these social media sites to search and meet their lost friends in their lives. Use of social media sites has made it simpler and quicker to get the latest news around the world and stay updated. It is this social networking that is helping people to make new friends, stay in touch with family, and develop relationships.

Figure 1.1: Most popular social media sites in 2020

Apart from maintaining and updating about social life on social media, social networking sites even help to keep in touch with business associates, clients and co-workers. LinkedIn is the best example for this as it provides the perfect platform to boost your profile as a candidate, build awareness about your brand or product and even help recruit the right people. Hence, LinkedIn is the world's largest professional networking platform and one of the most influential social media networks with more than 575 million users and That's a lot of potential contacts! (How to Use LinkedIn Effectively: Getting the Best from the World's Biggest Networking Site, 2020)

Although social media seems like a new trend but it actual is a natural outcome of many centuries of social media development. During 1900s, letters, telephones and radios were the earliest method of communication. These technologies are still in use today, but the modern versions are much more sophisticated. Technology began to change very rapidly in the 20th century when the super computers were created, and the engineers and scientists began to look for ways to connect these super computers which led to the emergence of network i.e. Internet. The first recognizable social media site, Six Degrees, was created in 1997. It enabled users to upload a profile and make friends with other users. In 1999, Twitter, the first blogging sites became popular, creating a social media sensation that's still popular today (The Complete History of Social Media: Then And Now, 2020).

The success of social networking services can be seen in their dominance in society today, with Facebook having a massive 2.13 billion active monthly users and an average of 1.4 billion daily active users in 2017. According to a study in 2015, 63% of the users of Facebook or Twitter in the USA consider these networks to be their main source of news, with entertainment news being the most seen.A 2015 study shows that 85% of people aged 18 to 34 use social networking sites for their purchase decision making. While over 65% of people aged 55 and over-rely on word of mouth.

1.2 Online Microblogging

Microblogging is an online broadcast medium which exits similar to blogging. Microblogging is distinct from blogging, as the content in both total and actual size files is usually smaller. Microblogging allows users to exchange small pieces of content such as links to media, individual images or short messages. Often those tiny messages are called micro-posts.
Microblogging is progressing steadily into the mainstream. For example, in 2016 millions of people used Twitter, one of the most common microblogging services, to microblog about the Brexit process.

52.2% of website traffic worldwide is generated by mobile and since audience use mobile phones as their immediate source of information, hence microblogging is essential. Since, microblogging usually appeals to the mobile browsing community, some of the famous microblogging platforms include:

- Twitter: It provides the platform to share short posts, GIFs, article links, videos and more.
- Pinterest: Companies on Pinterest link to products, articles and other useful information for audiences. Descriptions allow for quick content connections.
- Instagram: A visual form of microblogging, Instagram allows organizations to share stories and snaps as part of an online narrative.
- Tumblr: Tumblr is another highly popular microblogging platform. You can tag specific topics to attract attention from targeted audiences.

1.3 Twitter

Twitter nowadays is one of the popular social media as of 2018, twitter had over 321 million monthly active users. Twitter allows us to read and write short sentences of length 140 characters called tweets. The Tweets size was doubled to 280 characters for no-CJK languages in November 2017(Twitter, 2019). The 140 characters limit remains the same for most of the accounts while tweeting audio and video. Over twitter user's tweet about a variety of topics from movie reviews to political reviews or their feelings about the on-going or current situation within the country like US Election, Brexit, BlackLiveMatter, COVID-19.

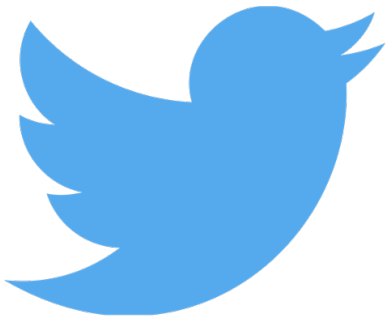


Figure 1.2: Twitter Logo

1.4 Sentimental Analysis

Sentiment analysis is the task of finding the user's opinion and affinity of users towards a specific topic of interest. User stay connected to their community on social media by sharing their day-to-day activities, addressing social issues, and a range of topics. Users tweets/reviews in any format without following any rules. Millions of users on social platforms give their opinions on different topics on a daily basis.

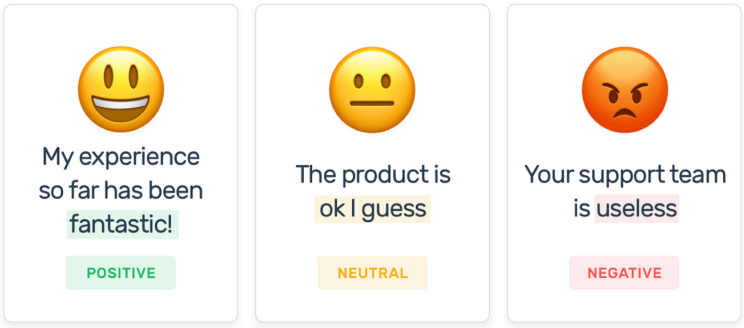


Figure 1.3 Sentiment Analysis

Sentiment Analysis allow businesses to identify customer sentiment towards products, brand or services in online feedback.

When a user wants to buy a product online, the first thing he or she sees is the kind of reviews and opinions the users have written. Twitter has become a forum for people to share their views on specific topics. The sentiment of the tweets of a particular subject has multiple applications, including political views analysis of users, stock market analysis of a company, movies reviews and analysis the mood of people.

These social media platforms are an immense source of information and it is very safe to claim that the process of sentiment analysis needs to be automated, as there is so much effort involved in manually processing this information. Various techniques such as machine learning and natural language processing are used for the automation of this process. This enormous amount of data used in the research purpose of sentiment analysis and opinion mining.

For example, moviemakers interested in the following questions:

- How many people viewed and reacted to their movies?
- How the movie is turned out to be? Good or bad?
- What is audience expectation from their movie?

The sentiment of reviews/tweets are categorized or classified into many categories such as positive, negative, neutral, extremely positive, extremely negative and so on. Below image is the example for the most common sentiments for tweets.

Class	Tweet
positive	@hon1paris: I <3 1D too! #muchlove
negative	The new Transformers suck!! Wasted my time and money!!!
neutral	Well, I guess the govt did what it could. More needed though! I plan to wake up early in the morning #early2bad

Figure 1.4: Sample tweets for the most common sentiment class/labels

Chapter 2: Background Research

2.1 VADER Lexicon

VADER stands for Valence Aware Dictionary and sEntiment Reasoner.

VADER, a simple rule-based model for general sentiment analysis (Hutto, Eric, 2020), a lexicon and rule-based sentiment analysis tool that is specifically attuned to sentiments expressed in social media. It is avaibale in the NLTK package and can be directly applied to unlabelled text data.

In dealing with the social media posts such as VADER not only tells us the score for positivity and negativity but also informs us how positive and negative is a sentiment.

VADER takes into consideration the valence score for the words.
Valence Score, score attributed by observation and experience to the word under consideration, rather than pure logic.

VADER is used by calling SentimentIntensityAnalyzer() method, which takes in a string and returns a dictionary of scores in each of four categories:

- Negative
- Neutral
- Positive
- Compound score

Compound score is calculated by normalizing the scores from above dictionaries (Negetive, Positive, Neutral). The compound score is determined by summing each word's valence scores in the lexicon, modified to the rules, and then standardised to be between -1 (most extreme negative) and +1 (most extreme positive). This is the most useful metric for a given sentence if you want a simple unidimensional measure of feeling

In order to detect the Sentiment of the text:

- If the compound is less than 0, then the sentiment is negative.
- If the compound is more than 0, then the sentiment is positive.
- If the compound is equal to 0, then the sentiment is neutral.

2.2 NRCLex (The Sentiment and Emotion Lexicons Download, 2020)

NRCLex also referred as **The Sentiment and Emotion Lexicons.** Detecting emotions in text has a wide range of applications including identifying anxiety or depression of individuals and measuring well-being or public mood of a community (Maryam Hasan, Elke and Agu, 2014).

The Sentiment and Emotion Lexicons is a collection of lexicons that was entirely created by the experts of the National Research Council of Canada (The Sentiment and Emotion Lexicons Download, 2020). This series of lexicons, developed with a wide range of applications, can be used in a multitude of contexts such as sentiment analysis, product marketing, customer behaviour and even political campaign analysis.

NRCLex measures the emotional effect from a body of text. The dictionary contains approximately 27000 words and is associated with eight basic emotions (fear, anger, anticipation, trust, surprise, sadness, disgust, and joy) and two sentiments (positive and negative)

It uses a list of words that help identify emotions, sentiment, as well as analyzing hashtags, emoticons and word-colour associations. The lexicons contain entries for English words and can be used to analyse English texts. For some lexicons, also provided are translations of the entries in other languages, including French, Arabic, Chinese, and Spanish.

2.3 TEXTBLOB

TEXTBLOB TextBlob is a Python (2 and 3) library for processing textual data. It provides a simple API for diving into common natural language processing (NLP) tasks such as part-of-speech tagging, noun phrase extraction, sentiment analysis, classification, translation, and more. (TextBlob: Simplified Text Processing — TextBlob 0.16.0 documentation, 2020)

Textblob is just like the python strings, can be used, transformed just like the strings in python. In our case, we need to find the sentiment labels (positive or negative) of the tweet text field.

Textblob has its own sentiment function, which returns two properties, subjectivity and polarity. Subjectivity sentence quantifies the amount of personal emotion, opinion, or judgment while objective refers to factual information.

Score:
Polarity is a float which lies in range of [-1,1] where -1 refers to extreme negative and +1 refers to extreme positive. The polarity is reversed by the negation words.
Subjectivity is also a float which lies in range of [0,1]. The higher subjectivity means that the text field of tweet contains personal opinion than factual opinion.

2.4 TF-IDF Vectoriser

"Term frequency–inverse document frequency" (tf–idf) is one of the most commonly used term weighting schemes in today's information retrieval systems. In a larger tweet corpus, some terms would be quite present (such as "the", "a", "in" etc.). If we were to feed the direct data count to the classifier, the frequencies of rare and more interesting terms will be shadowed by these frequent terms.

In order to re-weight the count features into floating point values suitable for usage by a classifier, it is very common to use the tf-idf transform (6.2. Feature extraction — scikit-learn 0.23.2 documentation, 2020). TF-IDF indicates what the importance of the word is in order to understand the document or dataset.

TF stands for term-frequency while IDF stands for inverse document-frequency.

Formula for TF-IDF is:

tf-idf(t,d) = tf(t,d) × idf(t) (6.2. Feature extraction-scikit-learn 0.23.2 documentation, 2020)

2.4.1 Term Frequency (TF)

The number of times a word appears in a document divided by the total number of words in the document. Every document has its own term frequency.

$$tf_{i,j} = \frac{n_{i,j}}{\sum_k n_{i,j}}$$

2.4.2 Inverse Data Frequency (IDF)

The log of the number of documents divided by the number of documents containing the word w. Inverse frequency of data determines the weight of rare words in the corpus in all documents.

$$idf(w) = log(\frac{N}{df_t})$$

Lastly, the TF-IDF is simply the TF multiplied by IDF.

$$w_{i,j} = tf_{i,j} \times \log\left(\frac{N}{df_i}\right)$$

TF-IDF Vectoriser converts a collection of raw documents to a matrix of TF-IDF features. The Vectoriser is usually trained on only the X_train dataset.

Parameters of TF-IDF vectoriser include –

- ngram_range which is the range of number of words in a sequence. [e.g "very expensive" is a 2-gram that is considered as an extra feature separately from "very" and "expensive" when you have a n-gram range of (1,2)]

- max_features specifies the number of features to consider. [Ordered by feature frequency across the corpus].

2.5 Confusion Matrix

Confusion Matrix is also known as Contingency Table or the Error Matrix (Confusion matrix, 2020). In a classification setting, this provides information on how well the algorithm has performed. So, this is a matrix 'kxk' for a classifier with classes k. An entry in row 'i' column 'j' reflects the number of examples which are in true class 'i' but which our classifier placed into class 'j'. The matrix name is derived from the fact that it illustrates uncertainty, that is, mislabelling of examples.

In this case the common names for the groups predicted are positive and negative.

Ways to check if the predictions are right or wrong:

- True Negative (TN): Predicated negative and it is true.
- True Positive (TP): Predicated positive and it is true.
- False Negative (FN): Predicated negative and it is false.
- False Positive (FP): Predicated Positive and it is false.

		True Class	
		Positive	Negative
Prediction Class	Positive	True Positive (TP)	False Positive (FP)
	Negative	False Negative (FN)	True Negative (TN)

Figure 2.1: Confusion matrix

One of the very common representation of the confusion matrix is to code the number as colours in the matrix. The extent of uncertainty between any two groups is then easy to see. Quite often in this case, the diagonal elements are zeroed out, because these are the examples our classifier got right and are typically much larger than the other matrix entries.

2.6 Naïve Bayes

Naive Bayes has been denigrated as “the punching bag of classifiers” (Lewis, 1998). Naïve Bayes is a probabilistic classifier based on the Bayes theorem. Naive Bayes methods are a series of supervised learning algorithms based on applying the theorem of Bayes with the "naive" assumption of conditional independence for each pair of characteristics given the class variable 's value.

Bayes formula:

$$P(y \mid x) = \frac{P(x \mid y) * P(y)}{P(x)}$$

Bayes’ theorem states that for the given class variable y has dependent feature vector x_1 through x_n:

$$P(y \mid x_1, \ldots, x_n) = \frac{P(y)P(x_1, \ldots, x_n \mid y)}{P(x_1, \ldots, x_n)}$$

(1.9. Naive Bayes — scikit-learn 0.23.2 documentation, 2020)

For all the above equations, the relationship is simplified to

$$P(y \mid x_1, \ldots, x_n) = \frac{P(y) \prod_{i=1}^{n} P(x_i \mid y)}{P(x_1, \ldots, x_n)}$$

(1.9. Naive Bayes — scikit-learn 0.23.2 documentation, 2020)

In our case, a tweet ‘y’ is represented by a vector of ‘n’ attributes such as X = (x_1, x_2, …… x_n). Computing P(Y/X) is not trivial and that’s why the Naïve Bayes introduces the assumption that all of the values X_j are independent given the category label c. That is, for i ≠ j, w_i and w_j are conditionally independent given the label y.

Since for the given input P(x_1, x_2, …… x_n) is constant, using the following classification rule:

$$P(y \mid x_1, \ldots, x_n) \propto P(y) \coprod_{i=1}^{n} P(x_i \mid y)$$

$$\Downarrow$$

$$\hat{y} = \arg\max_y P(y) \prod_{i=1}^{n} P(x_i \mid y),$$

(1.9. Naive Bayes — scikit-learn 0.23.2 documentation, 2020)

We can use the estimation Maximum A Posteriori (MAP) for estimating P(y) and P (x_i | y); the former is then the relative class ‘y’ frequency in the training set.

The various naive Bayes classifiers vary mainly in their assumptions about the distribution of P (xi | y).

2.6.1 Bernuolli Model

BernoulliNB 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 variable (Bernoulli, boolean) (1.9. Naive Bayes — scikit-learn 0.23.2 documentation, 2020).

Formula for Bernoulli model is:

$$P(x_i \mid y) = P(i \mid y) * x_i + (1 - P(i \mid y)) * (1-x_i)$$

Thus, the algorithm explicitly penalises a feature 's non-occurrence (word in the message is missing in the vocabulary) whereas the multinomial method uses the smoothing parameter for the missing values. Sklearn Bernoulli algorithm binarizes input values, so no further action is required.

Word occurrence vectors (rather than word count vectors) can be used to train and use this classifier in text classification.

BernoulliNB instance created and called with the alpha, which is a smoothing parameter. BernoulliNB is implemented as class BernoulliNB in module naïve_bayes.

2.6.2 Mutlinomial Model

Finally the most general generalisation of the Bernoulli distribution is across both the number of trials and the number of outcomes, called the multinomial distribution. For the text classification, the most used considered as the best choice is multinomial Naïve Bayes Model.

MultinomialNB implements the naive Bayes algorithm for multinomially distributed data, and is one of the two classic naive Bayes variants used in text classification (where the data are typically represented as word vector counts, although tf-idf vectors are also known to work well in practice) (1.9. Naive Bayes — scikit-learn 0.23.2 documentation, 2020)

For each class y, the distribution is parametrized by vectors $\theta_y = (\theta_{y1}, \ldots, \theta_{yn})$ where n is number of features and probability of P(xi | y) is θ_{yi} for feature i.

In this case distribution of probabilities for each event bases on the formula:

$$\hat{\theta}_{yi} = \frac{N_{yi} + \alpha}{N_y + \alpha n}$$

Where Ny is the total number of features of the event y, Nyi is the count of each feature, N is the number of features and α is the smoothing Laplace parameter. The smoothing parameter accounts for features not present in learning samples and prevents zero probabilities in further computations. (1.9. Naive Bayes — scikit-learn 0.23.2 documentation, 2020)
When α = 1, it is called Laplace Smoothing, while α < 1, it is called Lidstone Smoothing.

MultinomialNB instance created and called with the alpha, which is a smoothing parameter. MultinomialNB is implemented as class MultinomialNB in module naïve_bayes.

2.7 SVM model

Support Vector Machines (SVMs) (Corinna Cortes and Vladimir Vapnik,1995) are a system for efficiently training a linear learning machine to separate a given set of examples into positive and negative classes. The purpose of linear classification is to search in a feature space for a hyperplane dividing all entities into two class.

An important feature of SVMs is that they train the learning machine subject to specific constraints but result in 'sparse' dual representations of the algorithm, requiring only the dot products of the data (Nello, John, 1999).

The basic concept of SVMs is to search a single hyperplane that has the maximum distance from the points closest to it in the function space.

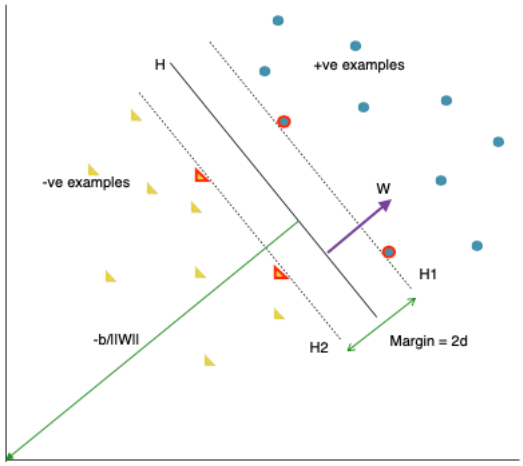


Figure 2.2: SVM separating hyperplane

In the above figure, the data points which are actually situated on H1 and H2 defines the separating hyperplane H and are called Support vectors. Once the SVM is trained, only one required are them. The non-support vectors, the remaining data points and not required anymore and can be ignored during the predication. Any new point can be predicted by measuring which side of H1 and H2 the new point would fall on. The expected mark is+1 if the new point falls outside of H1, where the positive examples are clustered. The expected mark is-1 if the new point falls outside of H2, where the negative examples live.

In figure 2.2:

Define the hyperplanes H such that: (MIT,2020)

$w \bullet x_i + b \geq +1$ when $y_i = +1$
$w \bullet x_i + b \leq -1$ when $y_i = -1$

H1 and H2 are the planes:

H1: $w \bullet x_i + b = +1$
H2: $w \bullet x_i + b = -1$

The points on the planes H1 and H2 are the tips of the Support Vectors
The plane H0 is the median in between, where $w \bullet xi + b = 0$

D – the shortest distance to the closet point

For the linearly sample, for an optimization problem search for a hyperplane that can be written down as:

$$\frac{1}{2}\|\omega\|^2 \to \min_{\omega,b}$$
$$y_i\left(\omega^T x_i + b\right) \geq 1, j = 1,\ldots,m,$$

(Barhan and Shakhomirov, 2020)

Where $1/||\omega||$ is the gap between the hyperplane and the points of both the first and second class, nearest to it, $y\ (\omega^T x + b)$ is the product of point class value and its position relative to the hyperplane.

The algorithm can commit errors in the linearly separable sample on the training entities. The error is to minimize this new optimization problem:

$$\frac{1}{2}\|\omega\|^2 + C\sum_{i=1}^{l} e_i \to \min_{\omega,b}$$
$$y_i\left(\omega^T x_i + b\right) \geq 1, j = 1,\ldots,k,$$
$$e_i \geq 0, i = 1,\ldots,k,$$

(Barhan and Shakhomirov, 2020)

For the sample k, values of error are given by e_i. Constant C is used to find a balance between maximising the distance on the training set and reducing the overall error.

2.7.1 LinearSVC()

LinearSVC is faster to implement Support Vector classification for a linear kernel. LinearSVC does not accept the kernel parameter, as this is assumed to be linear[21]. Linear SVC is similar to SVC with parameter kernel='linear', but implemented in terms of liblinear rather than libsvm, so it has more flexibility in the choice of penalties and loss functions and should scale better to large numbers of samples.
This class supports both dense and sparse data, and a one-vs-the-rest scheme is used to handle multi class support.

The primal problem can be equivalently formulated as (1.4. Support Vector Machines — scikit-learn 0.23.2 documentation, 2020)

$$\min_{w,b} \frac{1}{2} w^T w + C\sum_{i=1} \max(0, y_i(w^T \phi(x_i) + b)),$$

Where we make use of the hinge loss. The above form is directly optimized by LinearSVC, but unlike the dual form, this one does not require inter sample internal products, so the popular kernel trick cannot be applied. This is why only the linear kernel is supported

by LinearSVC (φ is the identity function) (1.4. Support Vector Machines — scikit-learn 0.23.2 documentation, 2020)

Linear SVC is implemented as class LinearSVC in module svm.

2.8 Logistic Regression

Logistic regression (LR) is a generalization of linear regression that models the probability of events' occurrence as a linear function of a set of predictor variables (Kantardzic, 2011). This model captures a vector of variables and calculates coefficients or weights for input variable and then predicts the class of tweet described as a word vector.

Logistic regression creates characteristic weights that are usually interpretable, making it particularly useful when you need to be able to clarify the reasons for a decision. This interpretability also comes in handy— with lenders for example having to explain their loan decisions.

The label default falls into one of two categories, Yes or No. Rather than modelling this label Y directly, logistic regression models the probability that Y is Yes. Let us encode Yes as 1 and No as 0.
Namely, logistic regression models p(X) := P(Y = 1 | X), where X = (X1,...,Xp)', by

$$p(X)=\sigma(\beta 0+\beta 1X1+\cdots+\beta pXp)$$

where σ is the logistic function (usually called the sigmoid function in the context of Neural Networks)

$$\sigma(x) = \frac{1}{1 + e^{-x}} = \frac{e^x}{1 + e^x}$$

LogisticRegression implementation can fit binary, this implementation can fit binary, One-vs-Rest, or multinomial logistic regression with optional ℓ1, ℓ2 or Elastic-Net regularization (1.1. Linear Models — scikit-learn 0.23.2 documentation, 2020)

In the optimization problem, binary class ℓ2 penalized logistic regression Minimizes the (1.1. Linear Models — scikit-learn 0.23.2 documentation, 2020)

$$\min_{w,c} \frac{1}{2} w^T w + C \sum_{i=1}^{n} \log(\exp(-y_i(X_i^T w + c)) + 1).$$

Optimization problem solved by ℓ1 regularized logistic regression: (1.1. Linear Models — scikit-learn 0.23.2 documentation, 2020)

$$\min_{w,c} \|w\|_1 + C \sum_{i=1}^{n} \log(\exp(-y_i(X_i^T w + c)) + 1).$$

Logistic regression is implemented as class LogisticRegression in module linear_model.

2.9 Pickle

The pickle module implements binary protocols for serializing and de-serializing a Python object structure.(pickle — Python object serialization — Python 3.8.5 documentation, 2020) Pickling is the method by which a Python object hierarchy is transformed into a byte stream, and unpickling is the opposite procedure by transforming a byte stream back into an object hierarchy.

Serialisation refers to the process of converting an object to a byte stream in memory that can be stored on a disk or sent over a network. This character stream can then be extracted later and de-serialized back to an object in Python.

Pickle uses the data format which us specific to the Python. By design, a reasonably compact binary representation is used in the pickle data Format. If needed optimum size characteristics you can compress pickled data efficiently.

Few pickle functions used in this project:

- Pickle.dump(obj, file, protocol=None, *, fix_imports=True, buffer_callback=None):
  Writes the object obj pickled representation to the file of the open file object.

- Pickle.dump(obj, protocol=None, *, fix_imports=True, buffer_callback=None):
  Return the object object's pickled representation as byte object, rather than writing it on disk.

- pickle.load(file, *, fix_imports=True, encoding="ASCII", errors="strict", buffers=None):
  Read the pickled representation of an object from the open file object file, and return the stated reconstituted object hierarchy.

- pickle.load(data, *, fix_imports=True, encoding="ASCII", errors="strict", buffers=None):
  Returns the reconstituted object hierarchy of an object's pickled representation data. Data must be a bytes-like object.

Chapter 3: Methodology

In this dissertation on “Predictive Power of Social Media”, I have selected a dataset with 1.6 million tweets from Twitter (available via Kaggle), the work is structured as follows: First find the score and sentiments labels (i.e positive – (0,1], negative – [-1,0]) for each tweet available in text field of the dataset using the VADEX lexicon. Then find the most frequent words in the whole dataset as well as each of sentiment labels and visualises them using the bar plot and word clouds. After that, I have compared some working model machine learning algorithms (such as TF-IDF vectoriser, SVMs, BernoulliNB. MultinomialNB, LogisticRegression) for accuracy and predication, for these comparisons I detected the sentiment label using the TEXTBLOB NLTK library and saving these models. At last the labels are predicted for the filtered input tweets with the help of these saved models. The input passed can be an array of text or tweet dataset.

3.1 Data Collection

A dataset of 16,00,000 tweets in English was found on Kaggle after my research. The dataset being used is the Sentiment140 dataset. It contains 1,600,000 tweets extracted using the Twitter API.

The dataset contains the following 6 fields:

- sentiment: the polarity of the tweet (0 = negative, 4 = positive)
- ids: The id of the tweet (1467810369)
- date: the date of the tweet (Sat May 16 23:58:44 UTC 2009)
- flag: The query (lyx). If there is no query, then this value is NO_QUERY.
- user: the user that tweeted (_theSpecialOne_)
- text: the text of the tweet (is upset that he can’t update his Facebook…)

We required only the text field, so the rest of the column fields were disregarded. The tweet text column is raw.

According to the creators of the dataset:

"Our approach was unique because our training data was automatically created, as opposed to having human’s manual annotate tweets. In our approach, we assume that any tweet with positive emoticons, like :), were positive, and tweets with negative emotions, like :(, were negative. We used the Twitter Search API to collect these tweets by using keyword search" (A., Bhayani, R. and Huang, L., 2009. Twitter sentiment classification using distant supervision. *CS224N Project Report, Stanford, 1(2009), p.12*.)

3.2 Preprocessing

Text Pre-processing is traditionally an important step in the processing of natural language (NLP) functions. It transforms text into a more digestible form, so that algorithms that learn machine can perform better.

3.2.1 Normalisation

As tweets may be in the user language, we need to clean up all irrelevant text field info.
The following things which can be irrelevant to the data are:

- Lower casting: Each tweet field should be converted into lowercase.

- URL's: URL's does not make any sense as it simply distracts the result.
  urlSyntax = r"((http://)[^]*|(https://)[^]*|(www\.)[^]*)"

- Username: For the cleaning purpose, it is necessary to removal of username as it simply distracts the result.
  userSyntax = '@[^\s]+'

- Non-alphabets: Removing characters except Digits and Alphabets with a space.
  alphaSyntax = "[^a-zA-Z0-9]"

- Repeated characters: If the character is repeated more than two times, which can be comprising a new word, but the context is the same, so we have to eliminate the word and make it real. For example, Hey can be written as Heyyyy.
  sequenceSyntax = r"(.)\1\1+"
  sequenceReplaceSyntax = r"\1\1"

- Stemming/Lemmatization: The aim of stemming and lemmatizationis to reduce inflectional forms and sometimes derivationally related forms of a word to a common base form.

  Stemming is the process of removing the ends of words. This often includes the removal or derivational affixes. For example, adjustable to adjust, formality to formal.

  Lemmatization is the process of converting a word to its base form. For example, was to be or better to good.

For the above-mentioned steps, I have built my own user-defined function, we take each tweet text as the parameter.

3.2.2 Removal of stop words

Stop words are words like "the," "a," "an," "in" and so on. The words being referred to as the stop words have nothing to do with the meaning of the statement for sentiment analysis, meaning these words need to be discarded from the tweet texts for further analysis on sentiment as they do not provide any information to our model that is removing the stop words from our corpus.

The stop words are being removed using the standard NLTK library which has the function import stopwords to perform text processing task for NLP.

Now each of the tweet text field is cleaned. Next step is to detect the sentiment orientation of each tweet text field.

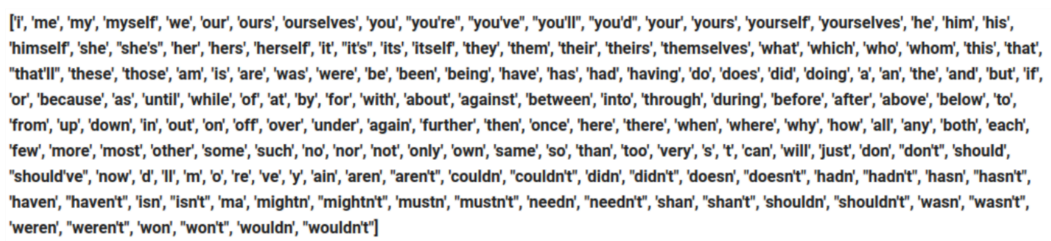

['i', 'me', 'my', 'myself', 'we', 'our', 'ours', 'ourselves', 'you', "you're", "you've", "you'll", "you'd", 'your', 'yours', 'yourself', 'yourselves', 'he', 'him', 'his', 'himself', 'she', "she's", 'her', 'hers', 'herself', 'it', "it's", 'its', 'itself', 'they', 'them', 'their', 'theirs', 'themselves', 'what', 'which', 'who', 'whom', 'this', 'that', "that'll", 'these', 'those', 'am', 'is', 'are', 'was', 'were', 'be', 'been', 'being', 'have', 'has', 'had', 'having', 'do', 'does', 'did', 'doing', 'a', 'an', 'the', 'and', 'but', 'if', 'or', 'because', 'as', 'until', 'while', 'of', 'at', 'by', 'for', 'with', 'about', 'against', 'between', 'into', 'through', 'during', 'before', 'after', 'above', 'below', 'to', 'from', 'up', 'down', 'in', 'out', 'on', 'off', 'over', 'under', 'again', 'further', 'then', 'once', 'here', 'there', 'when', 'where', 'why', 'how', 'all', 'any', 'both', 'each', 'few', 'more', 'most', 'other', 'some', 'such', 'no', 'nor', 'not', 'only', 'own', 'same', 'so', 'than', 'too', 'very', 's', 't', 'can', 'will', 'just', 'don', "don't", 'should', "should've", 'now', 'd', 'll', 'm', 'o', 're', 've', 'y', 'ain', 'aren', "aren't", 'couldn', "couldn't", 'didn', "didn't", 'doesn', "doesn't", 'hadn', "hadn't", 'hasn', "hasn't", 'haven', "haven't", 'isn', "isn't", 'ma', 'mightn', "mightn't", 'mustn', "mustn't", 'needn', "needn't", 'shan', "shan't", 'shouldn', "shouldn't", 'wasn', "wasn't", 'weren', "weren't", 'won', "won't", 'wouldn', "wouldn't"]

Figure 3.1: List of 179 NLTK stop words

3.3 Sentiment Identification

The text field of all tweets is cleaned and filtered. Now I have to find the sentiments labels (i.e positive or negative) for each tweet which is detected by VADER lexicon.

The lexicon-based Sentiment Analysis system accepts single tweets or a set of tweets, and outputs a predicted classification per tweet. The predicted classification is determined by running each tweet through three main stages: pre-processing, analysis and classification, as shown in Figure 3.2

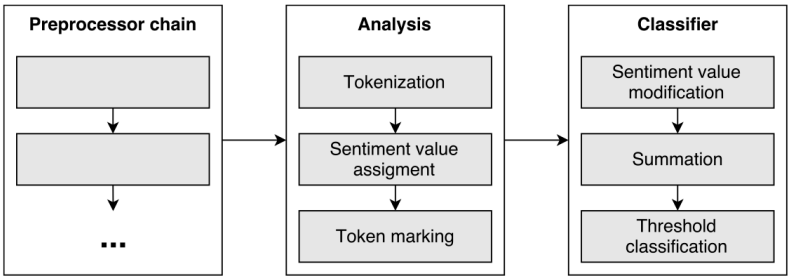


Figure 3.2: Architecture for Lexicon classifier system

For Identifying the sentiment orientation (positive and negative) of each tweet, VADER lexicon (refer 2.1) is being used.
In order to determine the sentiment for the selected dataset, Sentiment140 with which I am working on is assumed to be associated with the positive and negative sentiments of the tweets. While detecting these sentiments:
If the compound score (refer 2.1) is less than or equal to 0, then the sentiment orientation is negative else the sentiment orientation is positive.

Now, sentiment labels for each tweet text field is detected.

3.4 Emotion Identification

It can be observed that the text field of all tweets is cleaned and filtered and for each tweet each and every word has some associated emotions with it.

Emotions being identified are Fear, Anger, Anticipation, Trust, Surprise, Sadness, Disgust and Joy.

Lexicon file (NRC Emotion Lexicon) is used for identifying these emotions within the filtered tweets. NRCLex identifies the emotions of each word, such that it categories in one of these 8 categories of emotions.

3.5 Prediction

Prediction refers to the output of an algorithm after being trained on a dataset and when predicting the probability of a particular outcome applied to new data such as whether the future tweets would be positive or negative. For each record in the new data, the algorithm will produce probable values for an unknown variable, allowing the model builder to determine what that value will most likely be.

For the prediction following steps need to be followed:

3.5.1 Sentiment identification using TEXTBLOB

VADER lexicon is not being used here because the output of VADER cannot be used for training the classifiers.

Here, Textblob which is a standard NLTK library, being used for identifying the sentiment analysis of each tweet text field. Textblob is being used on the filtered and pre-processed text field.

Steps followed while identifying the sentiment labels are:

Importing textblob
Import nltk
From textblob import TEXTBLOB

A user defined function is used that calculates subjectivity, polarity and score for polarity. Sentiment labels identified on the basis of this score. If score is less than or equal to zero (<= 0), then sentiment label is negative otherwise sentiment label is positive.

This step identifies sentiment labels using the standard NLTK library textblob.

3.5.2 Data Splitting

Before building the models for machine learning, we will split the data into one set that will be used to develop models, pre-process the predictors, and explore relationships among the predictors and the response (the training set) and another that will be the final arbiter of the predictor set/model combination performance (the test set). To partition the data, the splitting of the original data set will be done in a *stratified* manner by making random splits in each of the outcome classes. This will keep the proportion of stroke patients approximately the same.

Data splitting is done using train_test_split() that splits the given dataset into two groups referred as training dataset and testing dataset. There are a number of ways to split the data into training and testing sets. The most common approach is to use some version of random sampling. Completely random sampling is a straightforward strategy to implement and usually protects the process from being biased towards any characteristic of the data.

- Training Data: The model is trained on 70% of the dataset
- Test Data: The dataset upon which the model would be tested for prediction and accuracy. It contains 30% of data from the dataset.

3.5.3 Training Dataset

Once the data is being split, train the machine learning models on the training dataset. The training set is used to develop models and feature sets; they are the substrate for estimating parameters, comparing models, and all of the other activities required to reach a final model.

Machine Learning models being used on the training dataset are: Naive Bayes (Bernoulli and Multinomial), Support Vector Machine (Linear SVC) and Logistics Regression (LR).

3.5.4 Test Models

Once the models are trained on the training dataset, the models are tested on the testing dataset. The test set is used only at the conclusion of these activities for estimating a final, unbiased assessment of the model's performance. It is critical that the test set not be used prior to this point. Looking at the test sets results would bias the outcomes since the testing data will have become part of the model development process.

Along with accuracy, prediction and f-score for each model is compared as well as with each model executing, confusion matrix is also being plotted.

3.5.5 Performance Measured

To determine the accuracy of above models it is necessary to measure from which accuracy can be obtained. There are two measurements on which accuracy will depend:

- Precision
- Recall
- Accuracy

Collection of M documents, Mp denotes the number of documents which belongs to true positive class and Mn denotes the number of documents which belongs to the true negative class. While classifying, True Positive (TP) documents had rightly classified whereas False Positive (FP) documents are wrongly classified, Similarly False Negative (FN) documents are wrongly classified and True Positive (TN) documents are rightly classified.

Precision: is the ratio of documents of rightly classified under positive predication class to the documents under positive Prediction class.

$$\text{Precision} = \frac{TP}{TP + FP}$$

Recall: is the ratio of documents of rightly classified under positive prediction class to the documents that are positive in the negative prediction class.

$$\text{Recall} = \frac{TP}{TP + FN}$$

Accuracy: In order to check which n-gram feature will give better results for these three models, we have to find the accuracy of classifiers. Accuracy for any prediction model can be given as:

$$\text{Accuracy} = \frac{TP+TN}{TP+TN+FP+FN}$$

3.5.6 Save the models

As the models are trained and tested. These trained and tested datasets are needed to be used in the predicting the input tweet sentiments labels. Before using them again later, these models need to be saved.

For this purpose, pickle is used. Pickle is used for serializing and de-serializing object structures.

For each model, pickle creates respective new files for each model using open() method, serialize them using the dump() method and stores them into their respective file and at last close them using close() function.

3.5.7 Load the models

While predicting using the predict() user defined function, function has a parameter which calls the saving models. Therefor these models need to load before predicting the labels for each input tweet.
The process of loading the models back is done by the pickle().

For this process, pickle loads the saved files of each model using open() method, de-serialize them using the load() method and output the labels for the called model and once output returned, closes the model file using close() function.

3.5.8 Predict

Here, a user defined function predict(), which takes 3 values, Vectoriser, model to be used for prediction and input filtered text or input filtered dataset. This function returns the pandas data frame with the data and columns name “text” and “sentiment”.

The input tweet text whose sentiment have to predict must be pre-processed already before passing as input argument.

For the input tweets, predication of sentiment labels (positive or negative) is predicted and shows as the output.

4.1 Tools used for Implementation

4.1.1 Sklearn

Scikit-learn (also known as sklearn) is an open source library/python module for machine learning which is built on top of SciPy (scikit-learn, 2020). It includes various features such as classification, regression, and clustering algorithms support vector machines, random forests, gradient boosting, k-means and works within scientific Python Packages such as NumPy, SciPy, and matplotlib.

It aims to provide simple and efficient tools for predictive data analysis that are accessible and reusable to everybody. Hence it is distributed under the 3-Clause BSD license (scikit-learn, 2020).

4.1.2 Warnings

warnings

Warning messages usually appear in situations to alert the developer of some condition in the program that aren't necessarily exceptions. Usually a warning is issued when the program uses some obsolete module (warnings — Warning control — Python 3.8.5 documentation, 2020). A warning in a program is distinct from an error because even if the warning message will be reported, the program will keep executing. You can, therefore, ignore the warning each time the code is executed. Programmers use warn() function to issue warnings that is defined within this module. The determine whether to issue a warning is controlled by filterwarnings().

Ignoring warning messages may abstruse real errors or outputs and may even negatively impact the program unless considered. While supressing warnings might be a quick fix for R&D work but should not be used in a production system. Hence, it is recommended to fix the warning messages in the software.

4.1.3 Pandas

Pandas is an open source software library for data manipulation and analysis for python programming language. It offers high performance, data structures and operations for manipulating datasets and time-series.

It aims to be the fundamental high-level building block for doing practical, real world data analysis in Python. Additionally, it has the broader goal of becoming the most powerful and flexible open source data analysis / manipulation tool available in any language.

Pandas are well suited for different kinds of dataframes:

- Tabular data as in SQL table or Excel spreadsheet or JSON

- Time series data (ordered/unordered)
- Matrix data with row and column labels
- Observational / Statistical datasets

Pandas allow various data manipulation operation on the dataset like merging, reshaping, selecting, data cleaning and data wrangling.

Pandas Library features include:

- Easy handling of missing data
- Mutability of data frame and data alignment
- Group by engine allowing split-apply-combine operations on data sets.
- Label-based slicing, fancy indexing, and subsetting of large data sets
- Data set merging and joining
- **Time series**-specific functionality: date range generation and frequency conversion, moving window statistics, date shifting and lagging (Package overview — pandas 1.1.1 documentation, 2020).

4.1.4 NumPy

NumPy is a library for the python programming language needed for scientific computation on multi-dimensional arrays and matrices. At the core of the NumPy package, is the *ndarray* object. This encapsulates *n*-dimensional arrays of homogeneous data types, with many operations being performed in compiled code for performance (What is NumPy? — NumPy v1.19 Manual, 2020).

NumPy's API is the starting point when libraries are written to exploit innovative hardware, create specialized array types, or add capabilities beyond what NumPy provides.

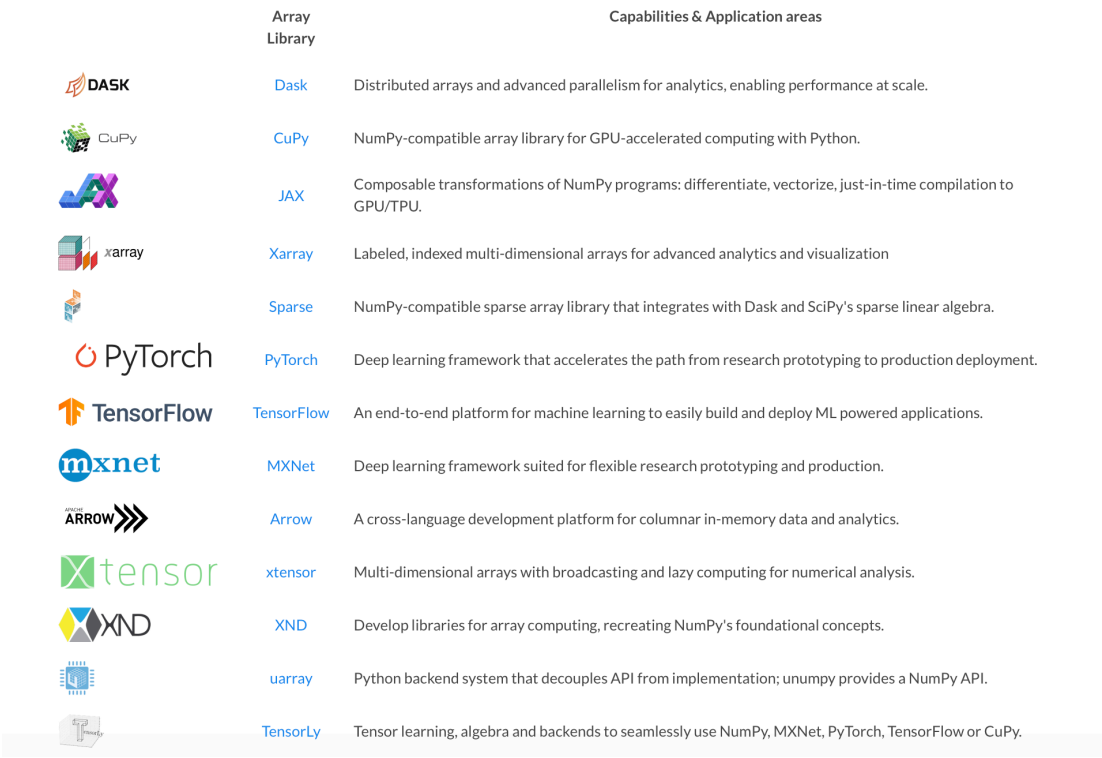
NumPy provides:

- Powerful and high-performance N-dimensional array object, and tools for working with these arrays.
- Tools for integrating C/C++ and Fortran code
- Numerical Computational Tools; NumPy offers useful comprehensive mathematical functions, linear algebra, Fourier transform, and random number capabilities
- Supports a wide range of hardware and computing platforms, and plays well with distributed, GPU, and sparse array libraries (NumPy, 2020).

4.1.5 NLTK

The Natural Language Toolkit, or commonly known as NLTK is an open source Python package for natural language processing NLTK includes graphical demonstrations and sample data. It provides easy-to-use interfaces to over 50 corpora and lexical resources such as WordNet, working along with text processing libraries for classification, tokenisation, stemming, tagging,

parsing, and semantic reasoning, wrappers for industrial-strength NLP libraries (Natural Language Toolkit — NLTK 3.5 documentation, 2020).



	Array Library	Capabilities & Application areas
DASK	Dask	Distributed arrays and advanced parallelism for analytics, enabling performance at scale.
CuPy	CuPy	NumPy-compatible array library for GPU-accelerated computing with Python.
	JAX	Composable transformations of NumPy programs: differentiate, vectorize, just-in-time compilation to GPU/TPU.
xarray	Xarray	Labeled, indexed multi-dimensional arrays for advanced analytics and visualization
	Sparse	NumPy-compatible sparse array library that integrates with Dask and SciPy's sparse linear algebra.
PyTorch	PyTorch	Deep learning framework that accelerates the path from research prototyping to production deployment.
TensorFlow	TensorFlow	An end-to-end platform for machine learning to easily build and deploy ML powered applications.
mxnet	MXNet	Deep learning framework suited for flexible research prototyping and production.
ARROW	Arrow	A cross-language development platform for columnar in-memory data and analytics.
xtensor	xtensor	Multi-dimensional arrays with broadcasting and lazy computing for numerical analysis.
XND	XND	Develop libraries for array computing, recreating NumPy's foundational concepts.
	uarray	Python backend system that decouples API from implementation; unumpy provides a NumPy API.
	TensorLy	Tensor learning, algebra and backends to seamlessly use NumPy, MXNet, PyTorch, TensorFlow or CuPy.

Figure 4.1: NumPy capabilities and application areas

4.1.6 Seaborn

seaborn

Seaborn is a Python data visualisation library built on matplotlib and integrated with pandas data structures. This Python library provides a high-level interface for making informative statistical graphics (seaborn: statistical data visualization — seaborn 0.10.1 documentation, 2020).

Seaborn aims to make visualization a central part of exploring and understanding data. Its dataset-oriented plotting functions operate on dataframes and arrays containing whole datasets and internally perform the necessary semantic mapping and statistical aggregation to produce informative plots (An introduction to seaborn — seaborn 0.10.1 documentation, 2020).

Functionalities that Seaborn offer:

- Provides with an API for understanding the relationship along various variables.
- Estimation and plotting for linear regression models for different kinds of dependent variables.
- Structuring of multi-plot grids that helps build complex visualisations.
- Visualising univariate or bivariate distributions for comparing these distributions between subsets of data.
- Provides tools for colour palettes that reveal patterns in the data.

4.1.7 Matplotlib

Matplotlib is a comprehensive plotting library for the Python programming language. It aims at creating static, animated, and interactive visualisations (Matplotlib: Python plotting — Matplotlib 3.3.1 documentation, 2020).

Matplotlib features include:

- CREATE
  Quality plots with few lines of code.
  Interactive figures that can zoom, pan, update etc.

- CUSTOMISE
  Control of line styles, font properties, axes properties
  Export and embed to a number of file formats and interactive environments.

- EXTEND
  Tailored functionality provided by third party packages

matplotlib.pyplot is an interface to matplotlib module which provides a MATLAB interface. Matplotlib is designed to be usable as MATLAB, with the ability to use Python, and the advantage of being free and open source (Matplotlib, 2020).

4.2 Implementation

4.2.1 Data collection

The dataset is being used in this dissertation is downloaded from Kaggle, sentiment140.

	0	1	2	3	4	5
0	0	1467810369	Mon Apr 06 22:19:45 PDT 2009	NO_QUERY	_TheSpecialOne_	@switchfoot http://twitpic.com/2y1zl - Awww, t...
1	0	1467810672	Mon Apr 06 22:19:49 PDT 2009	NO_QUERY	scotthamilton	is upset that he can't update his Facebook by ...
2	0	1467810917	Mon Apr 06 22:19:53 PDT 2009	NO_QUERY	mattycus	@Kenichan I dived many times for the ball. Man...
3	0	1467811184	Mon Apr 06 22:19:57 PDT 2009	NO_QUERY	ElleCTF	my whole body feels itchy and like its on fire
4	0	1467811193	Mon Apr 06 22:19:57 PDT 2009	NO_QUERY	Karoli	@nationwideclass no, it's not behaving at all....

Figure 4.2: Dataset structure

The dataset contains the following 6 fields:

- sentiment: the polarity of the tweet (0 = negative, 4 = positive)
- ids: The id of the tweet (1467810369)
- date: the date of the tweet (Sat May 16 23:58:44 UTC 2009)
- flag: The query (lyx). If there is no query, then this value is NO_QUERY.
- user: the user that tweeted (_TheSpecialOne_)
- text: the text of the tweet (is upset that he can't update his Facebook…)

4.2.2 Pre-processing using python

Now in this step data collected is pre-processed. Python language is being used for the pre-processing. Stop words, username, URL, non-alphabets, repeated words are removed using the regular expressions. On this filtered tweet text field, VADER lexicon, NRCLex and classifiers algorithm are implemented for the desired result.

4.3 Results

4.3.1 Sentiment Analysis

The first objective of this dissertation is finding the sentiment label (positive or negative) for each tweet text field. This is carried out 2 ways, VADER lexicon and textblob.

```
negative    808640
positive    791360
Name: sentiment, dtype: int64
```

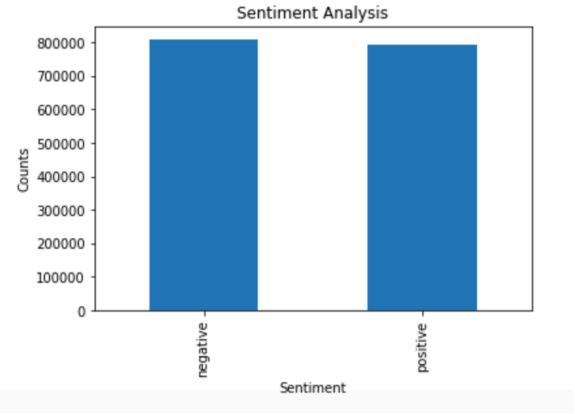


Figure 4.3: Sentiment labels using VADER

```
Positive    1053974
Negative     546026
Name: sentiment, dtype: int64
```

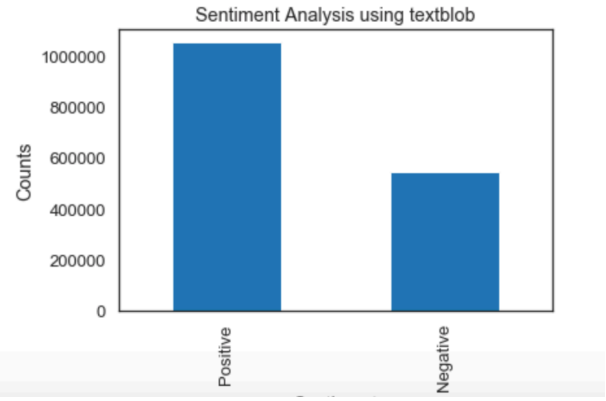


Figure 4.4: Sentiment labels using Textblob.

These above 2 figure demonstrate how many tweets are categorized into positive and negative sentiment label for the VADER and TEXTBLOB respectively.

4.3.2 Top 25 positive and negative words in positive and negative tweets respective

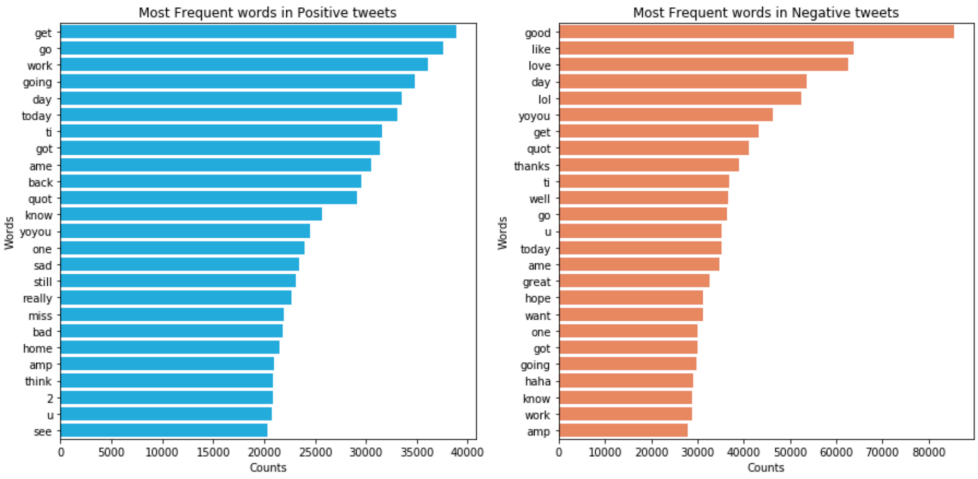


Figure 4.5: Bar plot of 25 top words in positive and negative tweets respectively

This bar plot shows that the most frequent words used in each positive and negative tweet respectively. Above plot gives the few bizarre results such as “good”, “like” are the most frequent words in negative tweets, this is because these words are used in such sentences more and more , and those sentences have the score less than or equal to 0, making such words into the negative tweets.

4.3.3 Word clouds analysis

The below figure 4.6 shows the word cloud of the most repeated words in the dataset after the text is filtered.

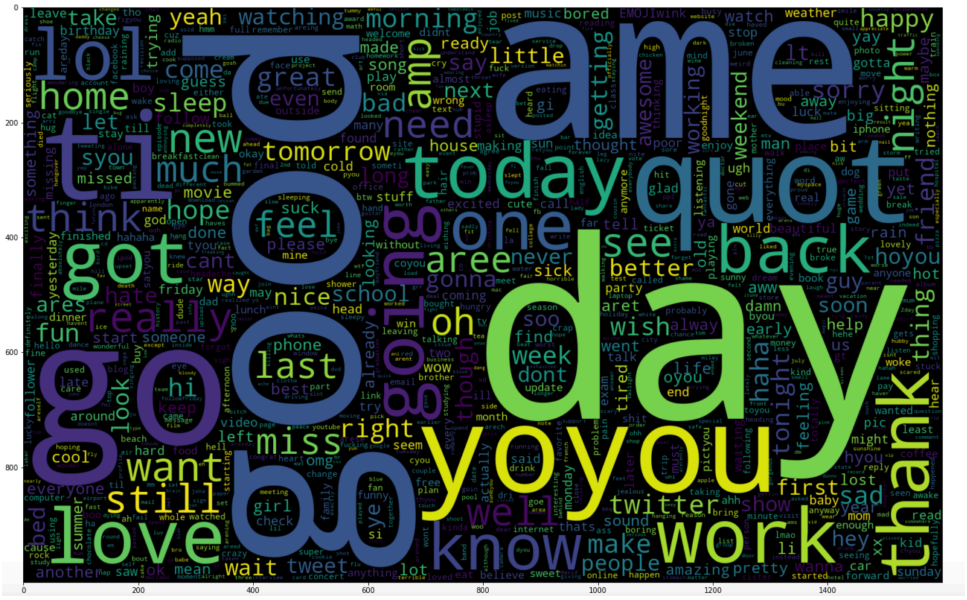


Figure 4.6: Word cloud of frequent words in dataset

The below figure 4.7 shows the word cloud for the most repeated words in positive tweets after the text is being filtered.

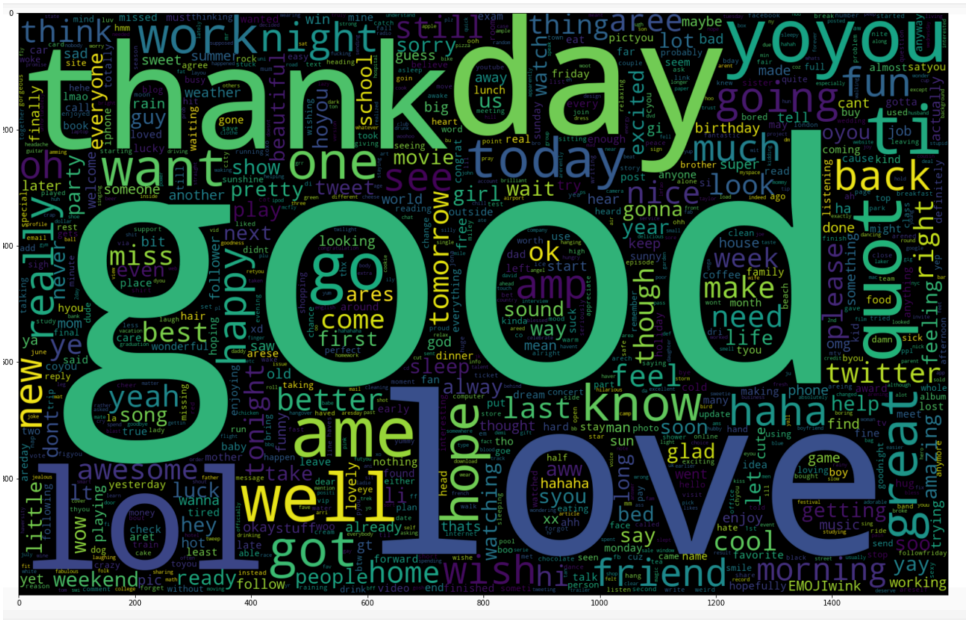


Figure 4.7: Word clouds of frequent words in the positive tweets

The below figure 4.8 shows the word of the most repeated words in the dataset for negative tweets after the text is filtered.

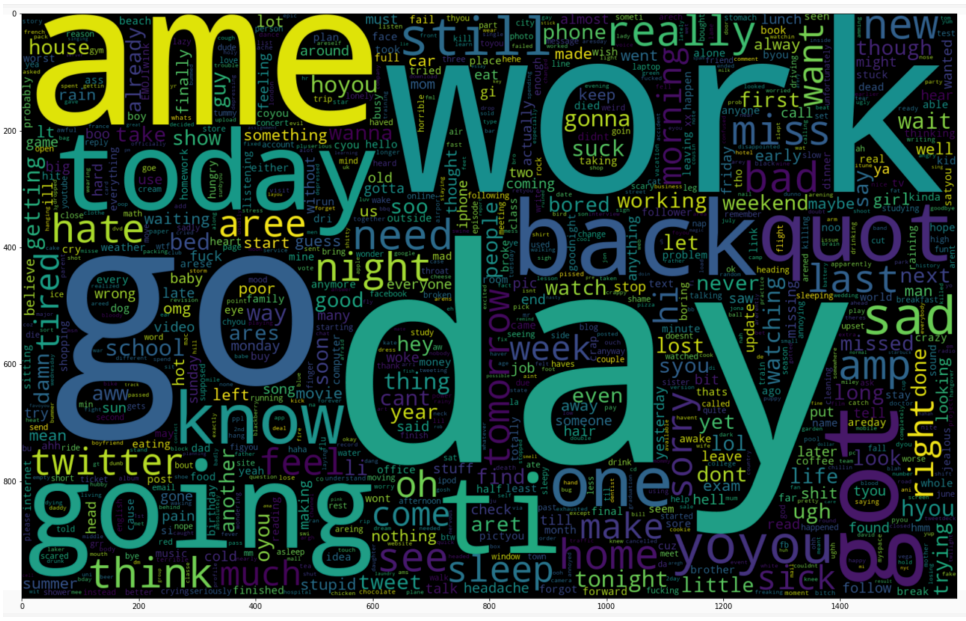


Figure 4.8: Word cloud of frequent words in the negative tweets.

4.3.4 Emotion detection

Emotion detection is being carried out by NRCLex. In NRCLex emotions of each word of tweet text field calculated. Emotions are detected of showing, the words used by people show emotional states of people.

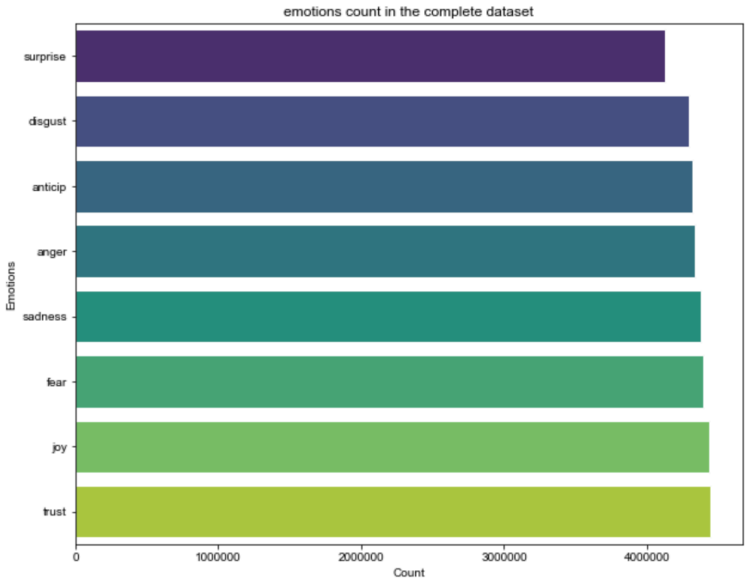


Figure 4.9: Each emotion count in the dataset

The above figure 4.9 shows the total words related to each emotion. This image says that in dataset, most words used in tweets of this dataset are belongs to trust category of emotion and least of surprise category of emotion.

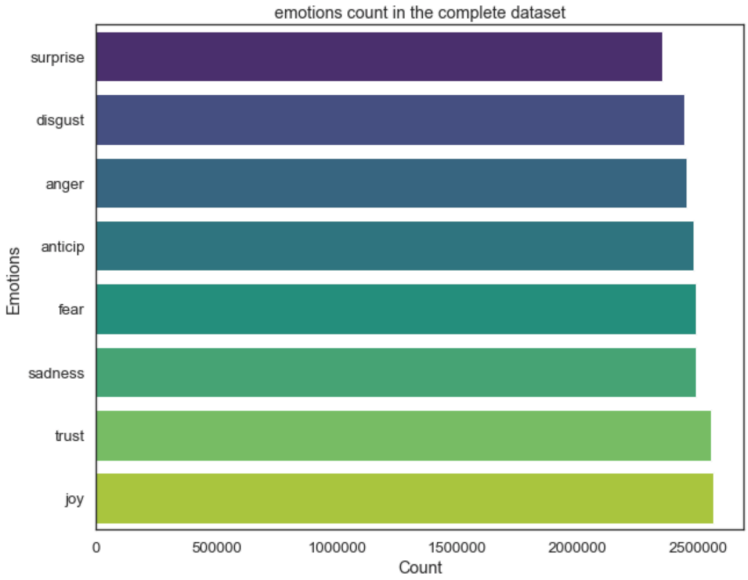


Figure 4.10: Each emotion count in the positive tweets.

The above figure shows that most words sued in positive sentiment tweets falls in joy and trust categories of emotions.

The below figure 4.11 shows the most words used in the negative tweets belong to fear category of the emotions.

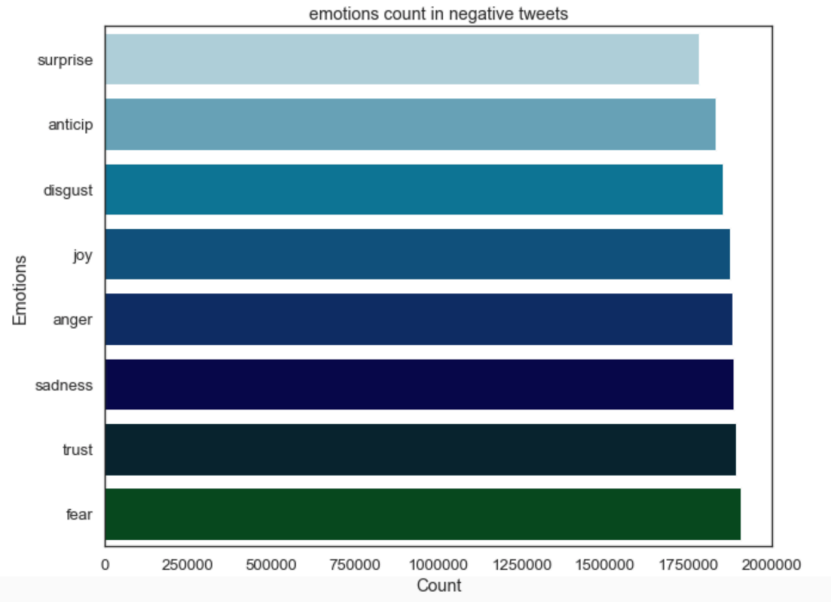


Figure 4.11: Each emotion count in the positive tweets

4.3.5 Performance measurement of classifiers

Different models used for comparison are Bernoulli, Multinomial Naïve bayes, Linear SVC, Logistic regression models.

For each model Evaluation matrix and confusion matrix are determined.

	precision	recall	f1-score	support
Negative	0.99	0.94	0.97	164295
Positive	0.97	1.00	0.98	315705
accuracy			0.98	480000
macro avg	0.98	0.97	0.98	480000
weighted avg	0.98	0.98	0.98	480000

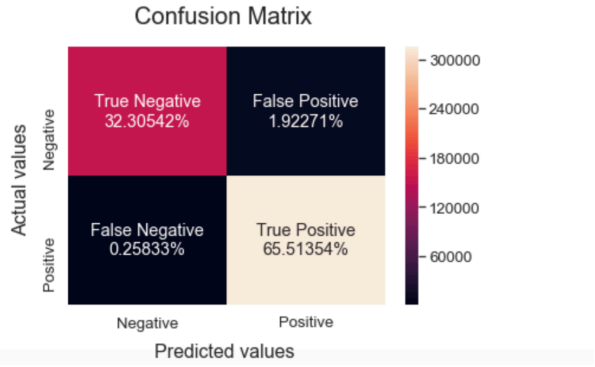


Figure 4.12: BernoulliNB model

The above figure 4.12 shows that accuracy is 98% for Bernoulli model.

	precision	recall	f1-score	support
Negative	1.00	0.26	0.41	164295
Positive	0.72	1.00	0.84	315705
accuracy			0.75	480000
macro avg	0.86	0.63	0.62	480000
weighted avg	0.82	0.75	0.69	480000

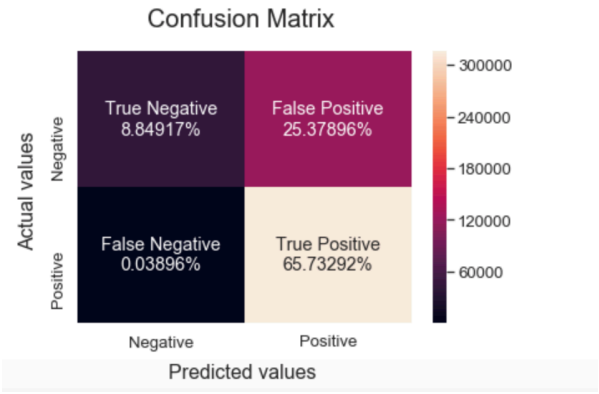


Figure 4.13: MultinomialNB

The above figure 4.13 shows that accuracy is 75% for multinomial Naive Bayes model.

	precision	recall	f1-score	support
Negative	1.00	1.00	1.00	164295
Positive	1.00	1.00	1.00	315705
accuracy			1.00	480000
macro avg	1.00	1.00	1.00	480000
weighted avg	1.00	1.00	1.00	480000

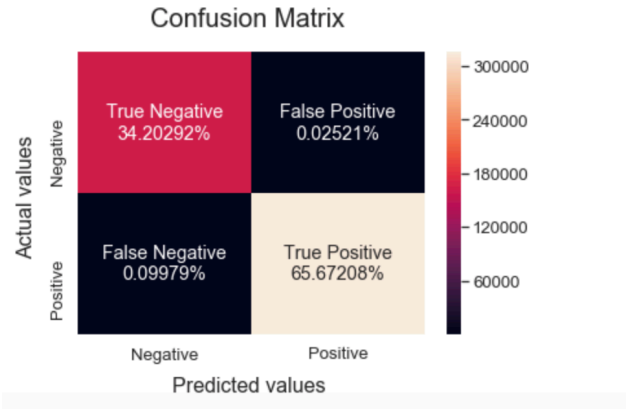


Figure 4.14: LinearSVC model

The above figure shows that accuracy is 100% for LinearSVC model.

	precision	recall	f1-score	support
Negative	0.99	1.00	0.99	164295
Positive	1.00	0.99	1.00	315705
accuracy			0.99	480000
macro avg	0.99	1.00	0.99	480000
weighted avg	1.00	0.99	0.99	480000

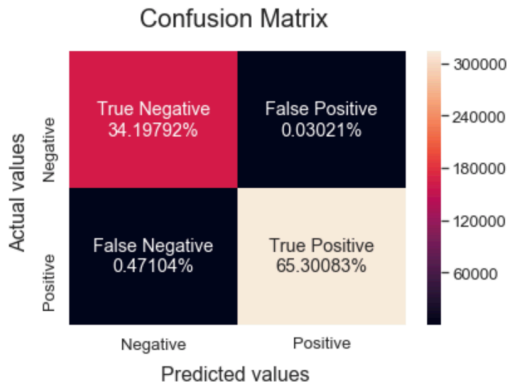


Figure 4.15: Logistic Regression model.

The above figure shows that accuracy is 99% for Logistic regression model.

The conclusion from above confusion matrix and Evaluation matrix is that linearSVC model has most accuracy with 100% and Mutltinomial model has least accuracy with 75% on this dataset with 70:30 spilt on train and test respectively.

4.3.6 Prediction of sentiment labels

For the input tweets, prediction for the sentimental labels (positive or negative) is done. Predication helps in finding the sentiment and emotions of tweets, this can help in finding the early depression, or sadness, fear in the user etc.

	text	sentiment
0	I hate my boring job	Positive
1	lost the not you	Negative
2	I don't like rain!	Negative

Figure 4.16: Sample inputs used in LR model.

In the above figure 4.16, sentiment labels were found out using the LR model.

Chapter 5: Challenges

Once I had the dataset it was unfiltered. While cleaning and filtering the text field of data field, few challenges I faced during the pre-processing of the dataset which are:

- Acronyms: the presence of “idk” or more complicated “omg” acronyms. In future, it can be done.

- Grammatical error: Grammatical errors may influence how a specific form of tweet is processed. On Twitter, urban grammar is being used such as “im gunna” or “im”. In these cases, filtering and understanding the closest possible word may be a big risk.

- Sarcasm: Sarcasm can be used to represent a particulate-large entity implicitly as good, bad or neutral, but it can also be used to improve or weaken a substance and view it as false-positive, true-positive, false-negative, true-negative. Therefor understanding the sarcasm is may be achieved in the future.

- Denoising: Along with numerous formatting and unnecessary phrases, unstructured tweets may influence the pre-processing and sentiment labels.

Chapter 6: Self-assessment

My progress though the execution of this project highlighted several points that intend to use as a steppingstone for my next venture. Here is the SWOT analysis of my work.

6.1 Strengths

Being organised and staying flexible: Staying connected with the coordinator all time times during this pandemics time. I used the sticky notes and remainder in my laptop to study items, weekly tasks. Being open to all ideas and eager to experiment with new methods has made the project much richer than anything else.

Using the right tool for the right job: Sharing the material and research with my supervisor made through outlook and continuous meetings eased my work, gave me a better perspective to my views towards my execution process for my dissertation topic. T Being open to all suggestions and willing to try out new approaches made the project much richer than otherwise

6.2 Weakness

Inexperience in certain areas: Extracting dataset from Twitter API was difficult in real time as it only allowed certain tweets to be downloaded at once. Because of which my initial project plan to find the sentiment for Brexit data became impossible and finally was able to find dataset on Kaggle to implement the same overview I had initial imagined for sentiment analysis.

Scoping the project: Next time round, I would attempt to pin down the extent of the project and manage the timeline a bit better. The workload became a bit bottom-heavy this time round due to the exploratory nature of the project.

6.3 Opportunities

Diving into machine learning Algorithm: As my first full project with a dataset, this has brought home to me how enjoyable it can be working with machine learning algorithms. The experienced guidance of my supervisor was a vital part of this revelation, as I saw him link up various ideas together seamlessly and saw how it was supposed to be done. Through VADER and NRCLex which was new and exciting gave me a fresh perspective towards my project goal.

6.4 Threats

The main obstacles that would prevent me from exploring my opportunities would be the lack of a suitable mentor, a resource and paucity on time.

Chapter 7: Conclusion

In this dissertation, I have done the detection of sentimental labels and emotions, comparative analysis on supervised classifiers like Bernoulli, Multinomial Naïve Bayes, Support vector machines and logistic regression and predicting sentimental labels of the input tweet. Sentimental analysis needs to be done as texts in the form of messages or posts to find out if the sentimental label is positive, neutral and negative. I have collected the dataset from Kaggle.

First the dataset file was loaded, and only text field was needed of tweets. Then in the pre-processing, that data is being cleaned, filtered and removed stop words. Now on this cleaned and filtered data, using standard NLTK libraries VADER lexicon is applied and scores of each tweet is calculated, and based on this score the sentiment labels is detected(if score is in range of [-1,0] then tweet has negative sentiment label otherwise tweet has positive sentiment label). Now few observations are done using the word cloud here.

For detecting the emotions of each word of tweet, NRCLex is being used. NRCLEx has 8 categories of emotions such as surprise, disgust, anticipate, anger, sadness, fear, joy and trust. Here few observations were made such as words belongs to the which category most for the complete dataset as well as in each of positive and negative sentiment labels of tweet. Emotion detections help to find the early depression, sadness or loneliness in the users.

In last different classifiers such as Bernoulli, Multinomial Naïve Bayes, Support vector machines and logistic regression are compared by Evaluation matrix and confusion matrix. For this comparing the models, the sentiment label of each tweet is determined/detected by the TEXTBLOB. Once the sentiment labels found, the classifiers are trained, tested and compared by accuracy and precision. The support vector machines algorithm has the highest precisions while Multinomial Naïve Bayes has the least precision. For reusing these models for the prediction, these classifiers are saved using the pickle and loaded back using the pickle as well. Now the tweets are passed as input for the prediction of sentimental labels. In this way sentimental labels are predicted.

References

About Facebook. 2020. Company Info | About Facebook. [online] Available at: <https://about.fb.com/company-info/> [Accessed 7 September 2020].

Barhan, A. and Shakhomirov, A., 2020. Methods For Sentiment Analysis Of Twitter Messages. [online] Semanticscholar.org. Available at: <https://www.semanticscholar.org/paper/Methods-for-Sentiment-Analysis-of-Twitter-Messages-Barhan-Shakhomirov/654926f241eaf84175ef9f9943fe2aa19d859601> [Accessed 6 September 2020].

Bird, S., Klein, E. and Loper, E., 2009. Natural Language Processing With Python. Sebastopol: O'Reilly Media, Inc.

Blog.twitter.com. 2020. Tweeting Made Easier. [online] Available at: <https://blog.twitter.com/official/en_us/topics/product/2017/tweetingmadeeasier.html> [Accessed 7 September 2020].

C.J Hutto; Eric Gilbert; 2020. [online] Available at: <http://comp.social.gatech.edu/papers/icwsm14.vader.hutto.pdf> [Accessed 6 September 2020].

Corinna Cortes and Vladimir Vapnik. Support-vector networks. Machine Learning, 20(3):273–297, September 1995. ISSN 0885-6125. doi: 10.1007/BF00994018. Available at: <http://link.springer.com/10.1007/BF00994018. >

Docs.python.org. 2020. Pickle — Python Object Serialization — Python 3.8.5 Documentation. [online]

Available at: <https://docs.python.org/3/library/pickle.html> [Accessed 6 September 2020].
Docs.python.org. 2020. Warnings — Warning Control — Python 3.8.5 Documentation. [online] Available at: <https://docs.python.org/3/library/warnings.html> [Accessed 6 September 2020].

eBrandz Blog. 2020. FACEBOOK PREFERRED BY YOUNGSTERS. OLDER LOT STILL PREFER “WORD OF MOUTH” AS THE MOST TRUSTED SOURCE OF MEDIUM - Ebrandz Blog. [online] Available at: <http://blogs.ebrandz.com/facebook-preferred-by-youngsters-older-lot-still-prefer-word-of-mouth-as-the-most-trusted-source-of-medium/> [Accessed 7 September 2020].

En.wikipedia.org. 2020. Confusion Matrix. [online] Available at: <https://en.wikipedia.org/wiki/Confusion_matrix> [Accessed 5 September 2020].

En.wikipedia.org. 2020. Matplotlib. [online] Available at: <https://en.wikipedia.org/wiki/Matplotlib> [Accessed 6 September 2020].

En.wikipedia.org. 2020. Social Networking Service. [online] Available at: <https://en.wikipedia.org/wiki/Social_networking_service> [Accessed 7 September 2020].

Eu.usatoday.com. 2020. [online] Available at: <https://eu.usatoday.com/story/tech/news/2017/10/26/twitter-overcounted-active-users-since-2014-shares-surge/801968001/> [Accessed 7 September 2020].

GeeksforGeeks. 2020. Warnings In Python - Geeksforgeeks. [online] Available at: <https://www.geeksforgeeks.org/warnings-in-python/> [Accessed 6 September 2020].

Fredriksen, V., Jahren, B. and Gambäck, B., 2020. Utilizing Large Twitter Corpora to Create Sentiment Lexica.

Johnson, M., 2020. 12.4 Test Set Results | Feature Engineering And Selection: A Practical Approach For Predictive Models. [online] Bookdown.org. Available at: <https://bookdown.org/max/FES/test-set-results.html> [Accessed 7 September 2020].

Lewis, D. D. (1998). Naive (Bayes) at forty: The independence assumption in information retrieval. Proceedings of ECML '98.

M. Kantardzic, 2011. Data mining concepts, models, method and algorithms.

Maryam Hasan, M., Elke, E. and Agu, E., 2014. [online] Web.cs.wpi.edu. Available at: <http://web.cs.wpi.edu/~emmanuel/publications/PDFs/C30.pdf> [Accessed 6 September 2020].

Matplotlib.org. 2020. Matplotlib: Python Plotting — Matplotlib 3.3.1 Documentation. [online] Available at: <https://matplotlib.org> [Accessed 6 September 2020].

Nello Cristianini and John Shawe-Taylor. An introduction to support Vector Machines: and other kernel-based learning methods. Cambridge University Press, January 1999. ISBN 0-521-78019-5. Avaiable at: http://dl.acm.org/citation.cfm?id=345662.

Nltk.org. 2020. Natural Language Toolkit — NLTK 3.5 Documentation. [online] Available at: <https://www.nltk.org> [Accessed 6 September 2020].

Numpy.org. 2020. Numpy. [online] Available at: <https://numpy.org> [Accessed 6 September 2020].

Numpy.org. 2020. What Is Numpy? — Numpy V1.19 Manual. [online] Available at: <https://numpy.org/doc/stable/user/whatisnumpy.html> [Accessed 6 September 2020].

Obar, J. and Wildman, S., 2015. Social Media Definition and the Governance Challenge: An Introduction to the Special Issue. SSRN Electronic Journal,.

Pandas.pydata.org. 2020. Package Overview — Pandas 1.1.1 Documentation. [online] Available at: <https://pandas.pydata.org/docs/getting_started/overview.html> [Accessed 6 September 2020].
Pdfs.semanticscholar.org. 2020. [online] Available at: <https://pdfs.semanticscholar.org/6549/26f241eaf84175ef9f9943fe2aa19d859601.pdf> [Accessed 6 September 2020].

PyPI. 2020. Scikit-Learn. [online] Available at: <https://pypi.org/project/scikit-learn/> [Accessed 6 September 2020].

Scikit-learn.org. 2020. 1.1. Linear Models — Scikit-Learn 0.23.2 Documentation. [online] Available at: <https://scikit-learn.org/stable/modules/linear_model.html> [Accessed 6 September 2020].

Scikit-learn.org. 2020. 1.4. Support Vector Machines — Scikit-Learn 0.23.2 Documentation. [online] Available at: <https://scikit-learn.org/stable/modules/svm.html#svm-mathematical-formulation> [Accessed 6 September 2020].

Scikit-learn.org. 2020. 1.9. Naive Bayes — Scikit-Learn 0.23.2 Documentation. [online] Available at: <https://scikit-learn.org/stable/modules/naive_bayes.html> [Accessed 5 September 2020].

Scikit-learn.org. 2020. 6.2. Feature Extraction — Scikit-Learn 0.23.2 Documentation. [online] Available at: <https://scikit-learn.org/stable/modules/feature_extraction.html> [Accessed 5 September 2020].

Scikit-learn.org. 2020. Scikit-Learn: Machine Learning In Python — Scikit-Learn 0.23.2 Documentation. [online] Available at: <https://scikit-learn.org/stable/> [Accessed 6 September 2020].

Seaborn.pydata.org. 2020. An Introduction To Seaborn — Seaborn 0.10.1 Documentation. [online] Available at: <https://seaborn.pydata.org/introduction.html> [Accessed 6 September 2020].

Seaborn.pydata.org. 2020. Seaborn: Statistical Data Visualization — Seaborn 0.10.1 Documentation. [online] Available at: <https://seaborn.pydata.org> [Accessed 6 September 2020].

Sentiment.nrc.ca. 2020. The Sentiment And Emotion Lexicons Download. [online] Available at: <http://sentiment.nrc.ca/lexicons-for-research/> [Accessed 6 September 2020].

Sentiment.nrc.ca. 2020. The Sentiment And Emotion Lexicons Download. [online] Available at: <http://sentiment.nrc.ca/lexicons-for-research/> [Accessed 6 September 2020].

Textblob.readthedocs.io. 2020. Textblob: Simplified Text Processing — Textblob 0.16.0 Documentation. [online] Available at: <https://textblob.readthedocs.io/en/dev/> [Accessed 6 September 2020].

Web.mit.edu. 2020. [online] Available at: <http://web.mit.edu/6.034/wwwbob/svm-notes-long-08.pdf> [Accessed 6 September 2020].