VISVESVARAYA TECHNOLOGICAL UNIVERSITY
Jnana Sangama, Belagavi - 590 018

Internship / Professional Practice Report

on

Twitter sentiment analysis

Submitted in partial fulfillment for the award of degree of

Bachelor of Engineering
in
Electronics and Communication Engineering

Submitted by

DIPTI G VARA
1RN18EC061

Internship carried out at
JPMorgans Chase and Co

Guide
Dr Uma Keshav
Senior Professor
ECE Dept.

External Guide
Ms Jessica Livon
Recruiter Jp Morgans
and Chase

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING
(Accredited by NBA for the Academic Years 2019-20, 2020-21 and 2021-22)

RNS INSTITUTE OF TECHNOLOGY
(AICTE Approved, VTU Affiliated and NAAC 'A' accredited)
(UG Programs - CSE, ECE, ISE, EIE and EEE have been Accredited by NBA for the Academic Years 2018-19, 2019-20 and 2020-2021)
Channasandra,Dr.Vishnuvardhan Road,Bengaluru-560098

2021-22

VISVESVARAYA TECHNOLOGICAL UNIVERSITY
Jnana Sangama, Belagavi - 590 018

Internship / Professional Practice Report

on

Twitter sentiment analysis

Submitted in partial fulfillment for the award of degree of

Bachelor of Engineering
in
Electronics and Communication Engineering

Submitted by

DIPTI G VARA
1RN18EC061

Internship carried out at
JPMorgans Chase and Co

Guide
Dr Uma Keshav
Senior Professor
ECE Dept.

External Guide
Ms Jessica Livon
Recruiter Jp Morgans Chase

ESTD 2001

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING
(Accredited by NBA for the Academic Years 2019-20, 2020-21 and 2021-22)

RNS INSTITUTE OF TECHNOLOGY
(AICTE Approved, VTU Affiliated and NAAC 'A' Accredited)
(UG Programs - CSE, ECE, ISE, EIE and EEE have been Accredited by NBA for the Academic Years 2019-20, 2020-21 and 2021-2022)
Channasandra,Dr.Vishnuvardhan Road,Bengaluru-560098

2021-22

RNS INSTITUTE OF TECHNOLOGY

(AICTE Approved, VTU Affiliated and NAAC 'A' Accredited)
(UG Programs - CSE, ECE, ISE, EIE and EEE have been Accredited by NBA for the Academic Years 2018-19, 2019-20 and 2020-2021)
Channasandra,Dr.Vishnuvardhan Road,Bengaluru-560098

DEPARTMENT OF ELECTRONICS AND COMMUNICATION ENGINEERING
(Accredited by NBA for the Academic Years 2018-19,2019-20 and 2020-21)

ESTD 2001

CERTIFICATE

This is to certify that the Internship/ Professional Practice work entitled **"Twitter sentiment analysis"** has been successfully carried at **JPMorgan Chase and Co** by **Dipti G Vara** bearing the usn **1RN18EC061**, bonafide student of **RNS Institute of Technology** in partial fulfillment for the award of Bachelor of Engineering in **Electronics and Communication Engineering** from **Visvesvaraya Technological University, Belagavi**, during the year 2020-2021. It is certified that all corrections / suggestions indicated for internal assessment have been incorporated in the report. The Internship report has been approved as it satisfies the academic requirements in aspect of the Internship work prescribed for the award of degree of **Bachelor of Engineering**.

Dr Uma Keshav	Dr. Vipula Singh	Dr. M K Venkatesha
Senior Professor	Head of Department	Principal

External Viva

Name of the examiners	**Signature with date**
1	
2	

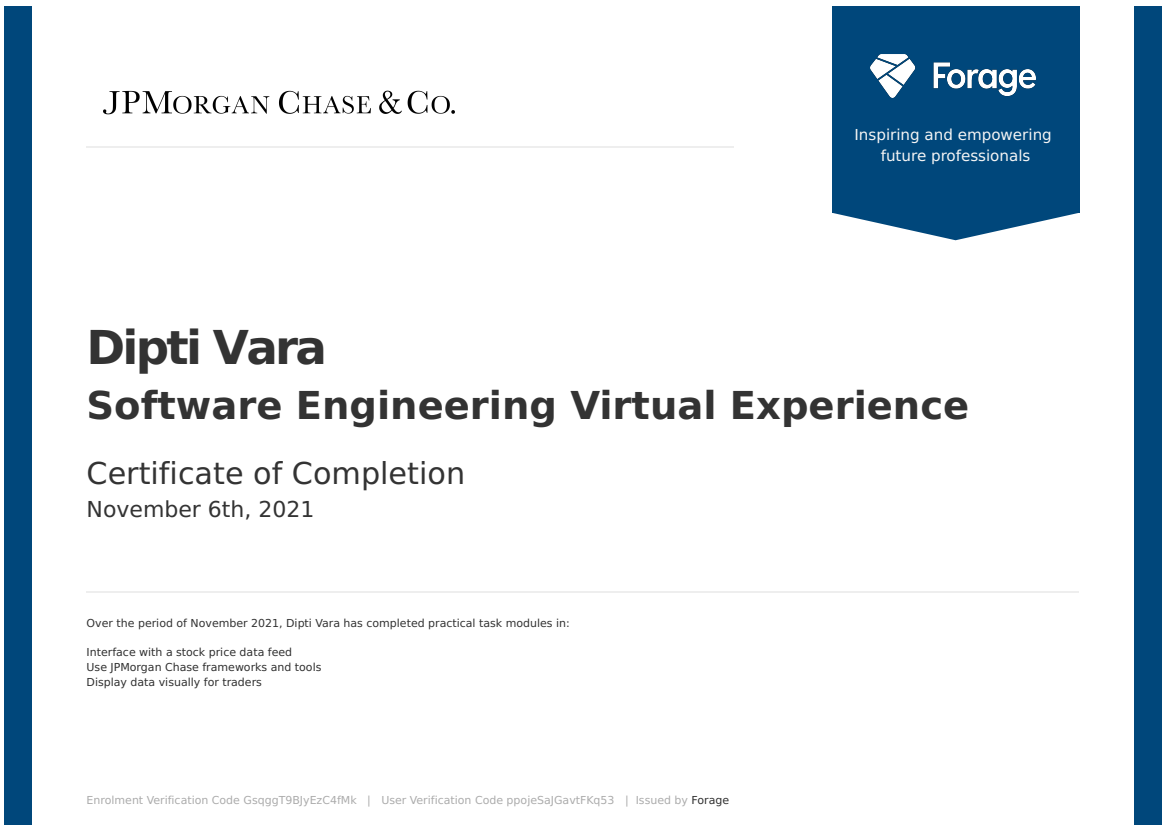

Figure 0.1: internship certificate

Acknowledgement

The joy and satisfaction that accompany the successful completion of any task would be incomplete without thanking those who made it possible. We consider ourselves proud to be a part of RNS Institute of Technology, the institution which molded us in all our endeavors.

I express my gratitude to our Chairman late **Dr. R N Shetty and MD Satish R Shetty**, for providing state of art facilities.

I would like to express my sincere thanks to **Dr. M K Venkatesha**, Principal and **Dr. Vipula Singh**, Professor and Head, Department of ECE, for their valuable guidance and encouragement throughout our program.

I extend my sincere thanks and heartfelt gratitude to my guide **Dr Uma Keshav**, Assistant Professor,Department of ECE, for providing me an invaluable support throughout the period of my Internship.

I wish to express my heartfelt gratitude to my Reviewer **Dr Uma Keshav**, Assistant Professor,Department of ECE, place for his valuable guidance, suggestions and cheerful encouragement during the entire period of my internship.

I would like to sincerely thank all those people who have been supporting in the part of my internship at *JP Morgan* in collaboration with *RNSIT* Centre of Excellence in Automation.

Finally, I take this opportunity to extend my earnest gratitude and respect to my parents, teaching and non-teaching staff of the department, the library staff and all my friends who have directly or indirectly supported me during the period of my Internship

Dipti G Vara

Executive Summary

The time I spent at JPMorgan Chase and Co as an intern from september 2021 to october 2021 was memorable one. I had so many experiences and opportunities which helped me to discover my potential. Internship in JPMorgans Chase and Co helped me to increase my skills and knowledge, understanding of particular job in an industry, gaining an insight into the way organization operates and challenges they face and networking. Internship also provided an opportunity for me to get to grips with working-meeting deadlines and working in team. It claries whether this type of work is really for me. I am feeling too much glad to work as an intern at JP Morgan Chase and Co. For interns working hours are from 4pm to evening 6.30. During this period of my internship, initially I was trained on data analytics with several tasks and application based programs later I worked on the project TWITTER SENTIMENTS ANALYSIS. I have learnt how projects will be done in real scenario.
some of the tasks done during my internship include:
Interface with a stock price data feed
Interface with a stock price data feed and set up your system for analysis of the data
Display data visually for traders,
Use Perspective to create the chart for the trader's dashboard

Table of Contents

List of Figures

Acronyms

EDM : Educational Data Mining

DMT : Data Mining Techniques

ML : Machine Learning

CSV : Comma Separated Values

KNN : K Nearest Neighbours

SVM : Support Vector Machines

NB : Naïve Bayes

RFA : Random Forest Algorithm

CV : Cross Validation

Poly : Polynomial

Chapter 1

Introduction

1.1 About company

JPMorgan

JP Morgan is a global leader in financial services ,offering solutions to the world's most important corporations,government and institutions in more than 100 countries .As announced in early 2018, JPMorgan chase will deploy 1.75 billion dollars in philanthropic capital around the world in 2023. They also lead volunteer services activities for employees in local communities by utilizing their many resources, including those that stem from access to capital, economies of scale, global reach expertise. Business has an important role to play in advancing the transition to a low carbon economy. JPMorgan Chase is committed to helping clients navigate the challenges and help capitalize on the long term economic opportunities and environmental benefits of progressing towards a low carbon world. They are applying their capital ,data ,a expertise and other resources to help address climate change and promote long tersm, innovative solutions for a sustainable future. In october 202, JPMorgan Chase announced the commitment to align key sectors of their financial portfolio with the goals of paris agreement.

1.2 Products and Servives

1.2.1 Business

The company is a leading global financial services firm with assets of 2.6 trillion dollars and operations worldwide. Our rich history spans over 200 years. We are a leader in investment banking, financial services for consumers and small business, commercial banking, financial transactions processing and asset management

1.2.2 Business model

ESG at J.P. Morgan
The firm continues to align its Environmental, Social and Governance (ESG) values with strategic business opportunities for our clients, employees and communities. Within the Corporate Investment Bank, recent deals – from green and social bonds

to holistic financing solutions – underscore the firm's expanding role in supporting the global shift towards a greener economy in both emerging and developed markets. On top of cybersecurity's critical role in protecting systems, networks, programs and data, it is equally as important to investors, who typically examine data protection and information security policies to assess a firm's cybersecurity risks. While cybersecurity has mainly been viewed as a technology issue, it is now also regarded as a key environmental, social and governance (ESG) concern, falling under the "Social" pillar.

ESG frameworks are a tangible means of evaluating corporate behavior; by incorporating cybersecurity, a new dimension is added, giving insight into cyber behaviors and risks which form a critical part of the bigger ESG picture. J.P. Morgan Global Research takes a closer look at the current cyber risks and why cybersecurity is fast becoming a core consideration in ESG frameworks.

2020 was a challenging year for global organizations, with the adjusted average total cost of a data breach reaching 4 million dollars per company. This was compounded by remote work forces, increasing the average total cost of a data breach by nearly 137,000 dollars. In a booming digital economy, cybersecurity is no longer just a software industry concern. It is becoming a major topic for company management, global investors and players from all industries with exposure to cyber technology and customers' private information. A far broader demographic is becoming increasingly concerned with cybersecurity's social impact as well as technological implications.

Chapter 2

Project Introduction

2.1 Introduction

Twitter is a popular social networking website where members create and interact with messages known as "tweets". This serves as a mean for individuals to express their thoughts or feelings about different subjects. Various different parties such as consumers and marketers have done sentiment analysis on such tweets to gather insights into products or to conduct market analysis. Furthermore, with the recent advancements in machine learning algorithms, we are able improve the accuracy of our sentiment analysis predictions.

In this report, we will attempt to conduct sentiment analysis on "tweets" using various different machine learning algorithms. We attempt to classify the polarity of the tweet where it is either positive or negative. If the tweet has both positive and negative elements, the more dominant sentiment should be picked as the final label. We use the data set from Kaggle which was crawled and labeled positive/negative. The data provided comes with emoticons, usernames and hashtags which are required to be processed and converted into a standard form. We also need to extract useful features from the text such unigrams and bigrams which is a form of representation of the "tweet".

We use various machine learning algorithms to conduct sentiment analysis using the extracted features. However, just relying on individual models did not give a high accuracy so we pick the top few models to generate a model ensemble. Ensembling is a form of meta learning algorithm technique where we combine different classifiers in order to improve the prediction accuracy. Finally, we report our experimental results and findings at the end.

2.2 Features

- Term presence and their frequency
- Part of speech information

- Negations
- Opinion words and phrases

2.3 Prerequisite

- There are some general library requirements for the project and some which are specific to individual methods. The general requirements are as follows.
- The sofwares used :
    1. Python
- Libraries used :
    1. numpy
    2. nltk
    3. matplotlib
    4. Scipy
- The library requirements specific to some methods are:
    1. keras with TensorFlow backend for Logistic Regression, MLP, RNN (LSTM), and CNN.
    2. xgboost for XGBoost.

2.4 Description of each component

2.4.0.1 Python

Python is an interpreted, high-level , general purpose programming language. It is one of the programming languages that has notable use of the signifificant whitespace. The main goal of python is to increase the code readability. Its a dynamically typed and garbage collected language. The biggest advantage of the programming language is the vast majority of its preloaded libraries that make the code minimalistic and reduce the size.

2.4.0.2 Tensorflow

TensorFlow is a free and open-source software library for machine learning. It can be used across a range of tasks but has a particular focus on training and inference of deep neural networks. Tensorflow is a symbolic math library based on dataflow and differentiable programming.TensorFlow, an interface for expressing machine learning algorithms, is utilized for implementing ML systems into fabrication over a bunch of areas of computer science, including sentiment analysis, voice recognition, geographic information extraction, computer vision, text summarization, information retrieval, computational drug discovery and flaw detection to pursue research . In the proposed model, the whole Sequential CNN architecture (consists of several layers) uses TensorFlow at backend. It is also used to reshape the data (image) in the data processing.

It is used for both research and production at Google, TensorFlow is Google Brain's second-generation system. Version 1.0.0 was released on February 11, While the reference implementation runs on single devices, TensorFlow can run on multiple CPUs and GPUs (with optional CUDA and SYCL extensions for general-purpose computing on graphics processing units).Tensor Flow is available on 64-bit Linux, macOS, Windows, and mobile computing platforms including Android and iOS. Its flexible architecture allows for the easy deployment of computation across a variety of platforms (CPUs, GPUs, TPUs), and from desktops to clusters of servers to mobile and edge devices. The name Tensor Flow derives from the operations that such neural networks perform on multidimensional data arrays, which are referred to as tensors.

During the Google I/O Conference in June 2016, Jeff Dean stated that 1,500 repositories on GitHub mentioned TensorFlow, of which only 5 were from Google.Unlike other numerical libraries intended for use in Deep Learning like Theano, TensorFlow was designed for use both in research and development and in production systems, not least RankBrain in Google search and the fun DeepDream project.It can run on single CPU systems, GPUs as well as mobile devices and large scale distributed systems of hundreds of machines.

2.4.0.3 keras

Keras is an open-source software library that provides a Python interface for artificial neural networks. Keras acts as an interface for the TensorFlow library. Up until version 2.3 Keras supported multiple backends, including TensorFlow, Microsoft Cognitive Toolkit, Theano, and PlaidML. Keras gives fundamental reflections and building units for creation and transportation of ML arrangements with high iteration velocity. It takes full advantage of the scalability and cross-platform capabilities of TensorFlow.

The core data structures of Keras are layers and models . All the layers used in the CNN model are implemented using Keras. Along with the conversion of the class vector to the binary class matrix in data processing, it helps to compile the overall model.

Keras is an API designed for human beings, not machines. Keras follows best practices for reducing cognitive load: it offers consistent & simple APIs, it minimizes the number of user actions required for common use cases, and it provides clear & actionable error messages. It also has extensive documentation and developer guides.Keras contains numerous implementations of commonly used neural network building blocks such as layers, objectives, activation functions, optimizers, and a host of tools to make working with image and text data easier to simplify the coding necessary for writing deep neural network code.The code is hosted on GitHub, and community support forums include the GitHub issues page, and a Slack channel. Keras is a minimalist Python library for deep learning that can run on top of Theano or Tensor Flow.It was developed to make implementing deep learning models as fast and easy as possible for research and development.It runs on Python 2.7 or 3.5 and can seamlessly execute on GPUs and CPUs given the underlying frameworks. It is released under the permissive MIT license.

- Keras was developed and maintained by François Chollet, a Google engineer using four guiding principles:

    1. Modularity: A model can be understood as a sequence or a graph alone. All the concerns of a deep learning model are discrete components that can be combined in arbitraryways.
    2. Minimalism: The library provides just enough to achieve an outcome, no frills and maximizing readability.
    3. Extensibility: New components are intentionally easy to add and use within the framework, intended for researchers to trial and explore new ideas.
    4. Python: No separate model files with custom file formats. Everything is native Python. Keras is designed for minimalism and modularity allowing you to very quickly define deep learning models and run them on top of a Theano or TensorFlow backend.

2.4.0.4 numpy

NumPy is a library for the Python programming language, adding support for large, multi-dimensional arrays and matrices, along with a large collection of high- level

mathematical functions to operate on these arrays.

2.4.0.5 Scipy

The SciPy library is one of the core packages that make up the SciPy stack. It provides many user-friendly and efficient numerical routines, such as routines for numerical integration, interpolation, optimization, linear algebra, and statistics.SciPy is a community-driven project. Development happens on GitHub.

Chapter 3

Project design and development

3.1 overview

Applying sentiment analysis on Twitter is the upcoming trend with researchers recognizing the scientific trials and its potential applications. The challenges unique to this problem area are largely attributed to the dominantly IJCSI International Journal of Computer Science Issues, Vol. 9, Issue 4, No 3, July 2012 ISSN (Online): 1694-0814 www.IJCSI.org 372 Copyright (c) 2012 International Journal of Computer Science Issues. All Rights Reserved. informal tone of the micro blogging. Pak and Paroubek [5] rationale the use of microblogging and more particularly Twitter as a corpus for sentiment analysis. They cited: Microblogging platforms are used by different people to express their opinion about different topics, thus it is a valuable source of people's opinions. Twitter contains an enormous number of text posts and it grows every day. The collected corpus can be arbitrarily large. Twitter's audience varies from regular users to celebrities, company representatives, politicians, and even country presidents. Therefore, it is possible to collect text posts of users from different social and interests groups. Twitter's audience is represented by users from many countries. Parikh and Movassate implemented two Naive Bayes unigram models, a Naive Bayes bigram model and a Maximum Entropy model to classify tweets. They found that the Naive Bayes classifiers worked much better than the Maximum Entropy model could. Go et al. proposed a solution by using distant supervision, in which their training data consisted of tweets with emoticons. This approach was initially introduced by Read. The emoticons served as noisy labels. They build models using Naive Bayes, MaxEnt and Support Vector Machines (SVM). Their feature space consisted of unigrams, bigrams and POS. The reported that SVM outperformed other models and that unigram were more effective as features. Pak and Paroubek have done similar work but classify the tweets as objective, positive and negative. In order to collect a corpus of objective posts, they retrieved text messages from Twitter accounts of popular newspapers and magazines, such as "New York Times", "Washington Posts" etc. Their classifier is based on the multinomial Naïve Bayes classifier that uses N-gram and POS-tags as features. Barbosa et al. too classified tweets as objective or subjective and then the subjective tweets were classified as positive or negative. The feature space used included features of tweets like retweets, hashtags, link, punctuation and exclamation

marks in conjunction with features like prior polarity of words and POS of words. Mining for entity opinions in Twitter, Batra and Rao used a dataset of tweets spanning two months starting from June 2009. The dataset has roughly 60 million tweets.

The entity was extracted using the Stanford NER, user tags and URLs were used to augment the entities found. A corpus of 200,000 product reviews that had been labeled as positive or negative was used to train the model. Using this corpus the model computed the probability that a given unigram or bigram was being used in a positive context and the probability that it was being used in a negative context. Bifet and Frank used Twitter streaming data provided by Firehouse, which gave all messages from every user in real-time. They experimented with three fast incremental methods that were well-suited to deal with data streams: multinomial naive Bayes, stochastic gradient descent, and the Hoeffding tree. They concluded that the SGD-based model, used with an appropriate learning rate was the best. Agarwal et al. approached the task of mining sentiment from twitter, as a 3-way task of classifying sentiment into positive, negative and neutral classes. They experimented with three types of models: unigram model, a feature based model and a tree kernel based model. For the tree kernel based model they designed a new tree representation for tweets. The feature based model that uses 100 features and the unigram model uses over 10,000 features. They concluded features that combine prior polarity of words with their parts-of-speech tags are most important for the classification task. The tree kernel based model outperformed the other two.

The Sentiment Analysis tasks can be done at several levels of granularity, namely, word level, phrase or sentence level, document level and feature level . As Twitter allows its users to share short pieces of information known as “tweets” (limited to 140 characters), the word level granularity aptly suits its setting. Survey through the literature substantiates that the methods of automatically annotating sentiment at the word level fall into the following two categories:
Corpus-based approaches.
Dictionary based approaches.

3.2 Methodology and Implementation.

Sentiment analysis (a.k.a opinion mining) is the automated process of identifying and extracting the subjective information that underlies a text. This can be either an

opinion, a judgment, or a feeling about a particular topic or subject. The most common type of sentiment analysis is called 'polarity detection' and consists in classifying a statement as 'positive', 'negative' or 'neutral'.

3.2.1 Pre-processing

Raw tweets scraped from twitter generally result in a noisy dataset. This is due to the casual nature of people's usage of social media. Tweets have certain special characteristics such as retweets, emoticons, user mentions, etc. which have to be suitably extracted. Therefore, raw twitter data has to be normalized to create a dataset which can be easily learned by various classifiers. We have applied an extensive number of pre-processing steps to standardize the dataset and reduce its size. We first do some general pre-processing on tweets which is as follows.

- Convert the tweet to lower case.
- Replace 2 or more dots (.) with space.
- Strip spaces and quotes (" and ') from the ends of tweet.
- Replace 2 or more spaces with a single space. We handle special twitter features as follows.

3.2.1.1 URL.

Users often share hyperlinks to other webpages in their tweets. Any particular URL is not important for text classification as it would lead to very sparse features. Therefore, we replace all the URLs in tweets with the word URL. The regular expression used to match URLs is ((www[§]+)—(https?://[§]+)).

3.2.1.2 User Mention

Every twitter user has a handle associated with them. Users often mention other users in their tweets by @handle. We replace all user mentions with the word USERMENTION. The regular expression used to match user mention is @[§]+.

3.2.1.3 Emoticon

Users often use a number of different emoticons in their tweet to convey different emotions. It is impossible to exhaustively match all the different emoticons used on

Emoticon(s)	Type	Regex	Replacement
:), :), :-), (:, (:, (-:, :')	Smile	(:\s?\)|:-\)|\(\s?:|\(-:|:\'\))	EMO_POS
:D, : D, :-D, xD, x-D, XD, X-D	Laugh	(:\s?D|:-D|x-?D|X-?D)	EMO_POS
;-), ;), ;-D, ;D, (;, (-;	Wink	(:\s?\(|:-\(|\)\s?:|\)-:)	EMO_POS
<3, :*	Love	(<3|:*)	EMO_POS
:-(, : (, :(,):,)-:	Sad	(:\s?\(|:-\(|\)\s?:|\)-:)	EMO_NEG
:,(, :'(, :"(	Cry	(:,\(|:\'\(|:"\()	EMO_NEG

Figure 3.1: list of emotions

social media as the number is ever increasing. However, we match some common emoticons which are used very frequently. We replace the matched emoticons with either EMOPOS or EMONEG depending on whether it is conveying a positive or a negative emotion. A list of all emoticons matched by our method is given below.

3.2.1.4 Hashtag

Hashtags are non-spaced phrases prefixed by the hash symbol () which is frequently used by users to mention a trending topic on twitter. We replace all the hashtags with the words with the hash symbol. For example, hello is replaced by hello. The regular expression used to match hashtags is hashtag(§+).

3.2.1.5 Retweet

Retweets are tweets which have already been sent by someone else and are shared by other users. Retweets begin with the letters RT. We remove RT from the tweets as it is not an important feature for text classification. The regular expression used to match retweets is .
After applying tweet level pre-processing, we processed individual words of tweets as follows.

- Strip any punctuation ['"?!,.():;] from the word.
- Convert 2 or more letter repetitions to 2 letters. Some people send tweets like I am sooooo happpppy adding multiple characters to emphasize on certain words. This is done to handle such tweets by converting them to I am soo happy.
- Remove - and '. This is done to handle words like t-shirt and their's by converting them to the more general form tshirt and theirs.

Raw	misses Swimming Class. http://plurk.com/p/12nt0b
Normalized	misses swimming class URL
Raw	@98PXYRochester HEYYYYYYYYY!! its Fer from Chile again
Normalized	USER_MENTION heyy its fer from chile again
Raw	Sometimes, You gotta hate #Windows updates.
Normalized	sometimes you gotta hate windows updates
Raw	@Santiago_Steph hii come talk to me i got candy :)
Normalized	USER_MENTION hii come talk to me i got candy EMO_POS
Raw	@bolly47 oh no :'(r.i.p. your bella
Normalized	USER_MENTION oh no EMO_NEG r.i.p your bella

Figure 3.2: Example tweets from the dataset and their normalized versions.

- Check if the word is valid and accept it only if it is. We define a valid word as a word which begins with an alphabet with successive characters being alphabets, numbers or one of dot (.) and underscore.

3.2.2 Feature Extraction

We extract two types of features from our dataset, namely unigrams and bigrams. We create a frequency distribution of the unigrams and bigrams present in the dataset and choose top N unigrams and bigrams for our analysis.

3.2.2.1 Unigrams

Probably the simplest and the most commonly used features for text classification is the presence of single words or tokens in the the text. We extract single words from the training dataset and create a frequency distribution of these words. A total of 181232 unique words are extracted from the dataset. Out of these words, most of the words at end of frequency spectrum are noise and occur very few times to influence classification. We, therefore, only use top N words from these to create our vocabulary where N is 15000 for sparse vector classification and 90000 for dense vector classification. The frequency distribution of top 20 words in our vocabulary is shown in figure 1. We can observe in figure 2 that the frequency distribution follows Zipf's law which states that in a large sample of words, the frequency of a word is inversely proportional to its rank in the frequency table. This can be seen by the fact that a linear trendline with a negative slope fits the plot of log (Frequency) vs. log (Rank). The equation of the trendline shown in figure 2 is log(Frequency) = 0:78 log(Rank) + 13:31.

$$idf(t) = \log\left(\frac{1+n_d}{1+df(d,t)}\right)+1$$

Figure 3.3: inverse document frequency

3.2.2.2 Bigrams

Bigrams are word pairs in the dataset which occur in succession in the corpus. These features are a good way to model negation in natural language like in the phrase – This is not good. A total of 1954953 unique bigrams were extracted from the dataset. Out of these, most of the bigrams at end of frequency spectrum are noise and occur very few times to influence classification. We therefore use only top 10000 bigrams from these to create our vocabulary.

3.2.3 Feature representation

After extracting the unigrams and bigrams, we represent each tweet as a feature vector in either sparse vector representation or dense vector representation depending on the classification method.

3.2.3.1 Sparse Vector Representation

Depending on whether or not we are using bigram features, the sparse vector representation of each tweet is either of length 15000 (when considering only unigrams) or 25000 (when considering unigrams and bigrams). Each unigram (and bigram) is given a unique index depending on its rank. The feature vector for a tweet has a positive value at the indices of unigrams (and bigrams) which are present in that tweet and zero elsewhere which is why the vector is sparse. The positive value at the indices of unigrams (and bigrams) depends on the feature type we specify which is one of presence and frequency.

- presence In the case of presence feature type, the feature vector has a 1 at indices of unigrams (and bigrams) present in a tweet and 0 elsewhere.
- frequency In the case of frequency feature type, the feature vector has a positive integer at indices of unigrams (and bigrams) which is the frequency of that

$$\hat{c} = \underset{c}{argmax}\ \mathrm{P(c|t)}$$

$$\mathrm{P(c|t)} \propto \mathrm{P(c)} \prod_{i=1}^{n} \mathrm{P}(f_i|\mathrm{c})$$

Figure 3.4: Equation

unigram (or bigram) in the tweet and 0 elsewhere. A matrix of such term-frequency vectors is constructed for the entire training dataset and then each term frequency is scaled by the inverse-document-frequency of the term (idf) to assign higher values to important terms. The inverse-document-frequency of a term t is defined as.

3.2.3.2 Dense vector representation

For dense vector representation we use a vocabulary of unigrams of size 90000 i.e. the top 90000 words in the dataset. We assign an integer index to each word depending on its rank (starting from 1) which means that the most common word is assigned the number 1, the second most common word is assigned the number 2 and so on. Each tweet is then represented by a vector of these indices which is a dense vector.

3.2.4 Classifiers

3.2.4.1 Naive Bayes

Naive Bayes is a simple model which can be used for text classification. In this model, the class ‘c is assigned to a tweet t, where

In the formula above, fi represents the i-th feature of total n features. P(c) and P(fi—c) can be obtained through maximum likelihood estimates.

3.2.4.2 Maximum Entropy

Maximum Entropy Classifier model is based on the Principle of Maximum Entropy. The main idea behind it is to choose the most uniform probabilistic model that maximizes the entropy, with given constraints. Unlike Naive Bayes, it does not assume that features are conditionally independent of each other. So, we can add features like bigrams without worrying about feature overlap. In a binary classification problem

like the one we are addressing, it is the same as using Logistic Regression to find a distribution over the classes. The model is represented by

3.2.4.3 Decision Tree

Decision trees are a classifier model in which each node of the tree represents a test on the attribute of the data set, and its children represent the outcomes. The leaf nodes represents the final classes of the data points. It is a supervised classifier model which uses data with known labels to form the decision tree and then the model is applied on the test data. For each node in the tree the best test condition or decision has to be taken. We use the GINI factor to decide the best split. For a given node t, GINI(t) = 1 P j [p(jjt)]2, where p(jjt) is the relative frequency of class j at node t, and GINIsplit = Pk i=1 ni n GINI(i) (ni = number of records at child i, n = number of records at node p)indicates the quality of the split. We choose a split that minimizes the GINI factor.

3.2.4.4 Random forest

Random Forest is an ensemble learning algorithm for classification and regression. Random Forest generates a multitude of decision trees classifies based on the aggregated decision of those trees. For a set of tweets x1; x2; : : : xn and their respective sentiment labels y1; y2; : : :n bagging repeatedly selects a random sample (Xb, Yb) with replacement. Each classification tree fb is trained using a different random sample (Xb, Yb) where b ranges from 1 : : :B. Finally, a majority vote is taken of predictions of these B trees

3.2.4.5 XGBoost

Xgboost is a form of gradient boosting algorithm which produces a prediction model that is an ensemble of weak prediction decision trees. We use the ensemble of K models by adding their outputs in the following manner where F is the space of trees, xi is the input and yi is the final output. We attempt to minimize the following loss function

3.2.4.6 SVM

SVM, also known as support vector machines, is a non-probabilistic binary linear classifier. For a training set of points (xi; yi) where x is the feature vector and y is

the class, we want to find the

3.2.4.7 Multi-Layer Perceptron

MLP or Multilayer perceptron is a class of feed-forward neural networks, which has atleast three layers of neurons. Each neuron uses a non-linear activation function, and learns with supervision using backpropagation algorithm. It performs well in complex classification problems such as sentiment analysis by learning non-linear models.

3.2.4.8 Convolutional Neural Networks

Convolutional Neural Networks or CNNs are a type of neural networks which involve layers called convolution layers which can interpret spacial data. A convolution layers has a number of filters or kernels which it learns to extract specific types of features from the data. The kernel is a 2D window which is slided over the input data performing the convolution operation. We use temporal convolution in our experiments which is suitable for analyzing sequential data like tweets.

3.2.4.9 Recurrent Neural Networks

Recurrent Neural Network are a network of neuron-like nodes, each with a directed (one-way) connection to every other node. In RNN, hidden state denoted by ht acts as memory of the network and learns contextual information which is important for classification of natural language. The output at each step is calculated based on the memory ht at time t and current input xt. The main feature of an RNN is its hidden state, which captures sequential dependence in information. We used Long Term Short Memory (LSTM) networks in our experiments which is a special kind of RNN capable of remembering information over a long period of time.

3.3 Experiments

3.3.1 Baseline

For a baseline, we use a simple positive and negative word counting method to assign sentiment to a given tweet. We use the Opinion Dataset of positive and negative words to classify tweets. In cases when the number of positive and negative words are equal, we assign positive sentiment. Using this baseline model, we achieve a classification

Algorithms	Presence		Frequency	
	Unigrams	Unigrams+Bigrams	Unigrams	Unigrams+Bigrams
Naive Bayes	78.16	**79.68**	77.52	79.38
Max Entropy	79.96	**81.52**	79.7	81.5
Decision Tree	**68.1**	68.01	67.82	67.78
Random Forest	76.54	**77.21**	76.16	77.14
XGBoost	77.56	**78.72**	77.42	78.32
SVM	79.54	81.11	79.83	**81.55**
MLP	80.1	**81.7**	80.15	81.35

Figure 3.5: Comparison of various classifiers which use sparse vector representation

accuracy of 63.48 percent on Kaggle public leaderboard.

3.3.2 Naive Bayes

We used MultinomialNB from sklearn.naive-bayes package of scikit-learn for Naive Bayes classification. We used Laplace smoothed version of Naive Bayes with the smoothing parameter set to its default value of 1. We used sparse vector representation for classification and ran experiments using both presence and frequency feature types. We found that presence features outperform frequency features because Naive Bayes is essentially built to work better on integer features rather than floats. We also observed that addition of bigram features improves the accuracy. We obtain a best validation accuracy of 79.68 percent using Naive Bayes with presence of unigrams and bigrams. A comparison of accuracies obtained on the validation set using different features is shown in table.

3.3.3 Maximum Entropy

The nltk library provides several text analysis tools. We use the MaxentClassifier to perform sentiment analysis on the given tweets. Unigrams, bigrams and a combination of both were given as input features to the classifier. The Improved Iterative Scaling algorithm for training provided better results than Generalised Iterative Scaling. Feature combination of unigrams and bigrams, gave better accuracy of 80.98For a binary classification problem, Logistic Regression is essentially the same as Maximum Entropy. So, we implemented a sequential Logistic Regression model using keras, with sigmoid activation function, binary cross-entropy loss and Adam's optimizer achieving better performance than nltk. Using frequency and presence features we get almost the same accuracies, but the performance is slightly better when we use unigrams and bigrams together. The best accuracy achieved was 81.52 percent A comparison

of accuracies obtained on the validation set using different features is shown in table.

3.3.4 Decision Tree

We use the DecisionTreeClassifier from sklearn.tree package provided by scikit-learn to build our model. GINI is used to evaluate the split at every node and the best split is chosen always. The model performed slightly better using the presence feature compared to frequency. Also using unigrams with or without bigrams didn't make any significant improvements. The best accuracy achieved using decision trees was 68.1 percent A comparison of accuracies obtained on the validation set using different features is shown in the table.

3.3.5 Random forest

We implemented random forest algorithm by using RandomForestClassifier from sklearn.ensemble provided by scikit-learn. We experimented using 10 estimators (trees) using both presence and frequency features. presence features performed better than frequency though the improvement was not substantial. A comparison of accuracies obtained on the validation set using different features is shown in table 5.

3.3.6 XGBoost

We also attempted tackling the problem with XGboost classifier. We set max tree depth to 25 where it refers to the maximum depth of a tree and is used to control over-fitting as a high value might result in the model learning relations that are tied to the training data. Since XGboost is an algorithm that utilises an ensemble of weaker trees, it is important to tune the number of estimators that is used. We realised that setting this value to 400 gave the best result. The best result was 0.78.72 which came from the configuration of presence with Unigrams + Bigrams.

3.3.7 SVM

We utilise the SVM classifier available in sklearn. We set the C term to be 0.1. C term is the penalty parameter of the error term. In other words, this influences the misclassification on the objective function. We run SVM with both Unigram as well Unigram + Bigram. We also run the configurations with frequency and presence. The best result was 81.55 which came the configuration of frequency and Unigram +

Bigram.

3.3.8 Multi-Layer Perceptron

The output from the neural network gives the probability Pr(positivejtweet) i.e. the probability of the tweets sentiment being positive. At the prediction step, we round off the probability values to convert them to class labels 0 (negative) and 1 (positive). The architecture of the model is shown in figure . Red hidden layers represent layers with sigmoid non-linearity. We trained our model using binary cross entropy loss with the weight update scheme being the one defined by Adam et. al. We also conducted experiments using SGD + Momentum weight updates and found out that it takes too long to converge. We ran our model upto 20 epochs after which it began to overfit. We used sparse vector representation of tweets for training. We found that the presence of bigrams features significantly improved the accuracy.

3.3.9 Convolutional Neural Networks

We used keras with TensorFlow backend to implement the Convolutional Neural Network model. We used the dense vector representation of the tweets to train our CNN models. We used a vocabulary of top 90000 words from the training dataset. We represent each word in our vocabulary with an integer index from 1 : : : 90000 where the integer index represents the rank of the word in the dataset. The integer index 0 is reserved for the special padding word. Further each of these 90000+1 words is represented by a 200 dimensional vector. The first layer of our models is the embedding layer which is a matrix of shape (v+1)d where v is vocabulary size (=90000) and d is the dimension of each word vector (=200). We initialize the embedding layer with random weights from N(0; 0:01). Each row of this embedding matrix represents represents the 200 dimensional word vector for a word in the vocabulary. For words in our vocabulary which match GloVe word vectors provided by the Stanford NLP group, we seed the corresponding row of the embedding matrix from GloVe vectors. Each tweet i.e. its dense vector representation is padded with 0s at the end until its length is equal to max-length which is a parameter we tweak in our experiments. We trained our model using binary cross entropy loss with the weight update scheme being the one defined by Adam et. al. We also conducted experiments using SGD + Momentum weight updates and found out that it takes longer (100 epochs) to converge compared to validation accuracy equivalent to Adam. We ran our model upto 10 epochs. Using the Adam weight update scheme, the model converges very fast (4

epochs) and begins to over fit badly after that. We, therefore,use models from 3rd or 4th epoch for our results. We tried four different CNN architectures which are as follows.

- 1-Conv-NN: As the name suggests, this is an architecture with 1 convolution layer. We perform temporal convolution with a kernel size of 3 and zero padding. After the convolution layer, we apply relu activation function (which is defined as f(x) = max(0; x)) and then perform Global Max Pooling over time to reduce the dimensionality of the data. We pass the output of the Global Max Pool layer to a fully-connected layer which then outputs a single value which is passed through sigmoid activation function to convert it into a probability value

- 2-Conv-NN: In this architecture we increased the vocabulary from 80000 to 90000. We also increased the dropout after embedding layer to 0.4 and that after the fully connected layer to 0.5 to further regularize the network and thus prevent overfitting. We changed the number of filters in the first convolution layer to 600 and added another convolution layer with 300 filters after the first convolution layer. We also replaced the Global MaxPool layer with a Flatten layer as we believed some features of the input tweets got lost while max pooling. We also increased the number of units in the fully-connected layer to 600.

- 3-Conv-NN: In this architecture we added another convolution layer with 150 filters after the second convolution layer.

- 4-Conv-NN: In this architecture we added another convolution layer with 75 filters after the third convolution layer. We also increased max-length of the tweet to 40 going by the fact that the length of largest tweet in our pre-processed dataset is about 40 words.

3.3.10 Recurrent Neural Networks

We used neural networks with LSTM layers in our experiments. We used a vocabulary of top 20000 words from the training dataset. We used the dense vector representation for training our models. We pad or truncate each dense vector representation to make it equal to max-length which is a parameter we tweak in our experiments. The first layer of our network is the Embedding layer which as described in section 4.9 We test two different types of LSTM models. We experimented with both Adam optimizer and SGD with momentum for training our networks and find the Adam worked better and converges faster. We trained our model using $\mathrm{mean}_{s}quared_{e}rrorandbinary_{c}ross_{e}ntropyloss.Wefoundthatbinary_{c}ross_{e}ntropyworkedbetterthann$

3.3.11 Ensemble

In a quest to further improve accuracy, we developed a simple ensemble model. We first extract 600 dimensional feature vectors for each tweet from the penultimate layer of our best performing 4-Conv-NN model. Each tweet is now represented by a 600 dimensional feature vector. We use these features to classify the tweets using a linear SVM model with C=1. We classify the tweets using this SVM model. We then take the majority vote of predictions from the following 5 models.

- LSTM-NN
- 4-Conv-NN
- 4-Conv-NN features + SVM

- 4-Conv-NN with $\max_length = 203 - Conv - NN$

Chapter 4

Result Analysis

4.1 Analysis

The obtained results are:
figure 4.1 shows the frequencies obtained for top 20 bigrams

figure 4.2 below shows the output obtained for frequencies that follow Zipf's law.
figure 4.3 shows the comparison of accuracies of various models.

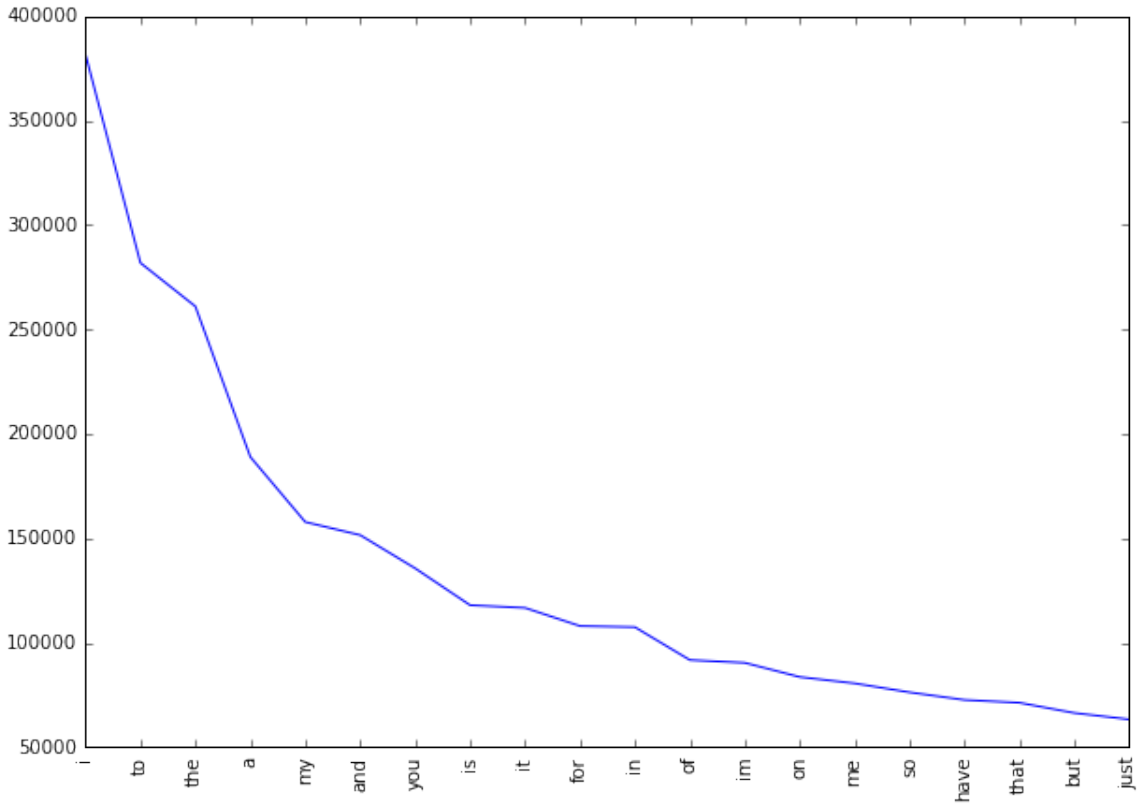

Figure 4.1: Frequencies of top 20 bigrams

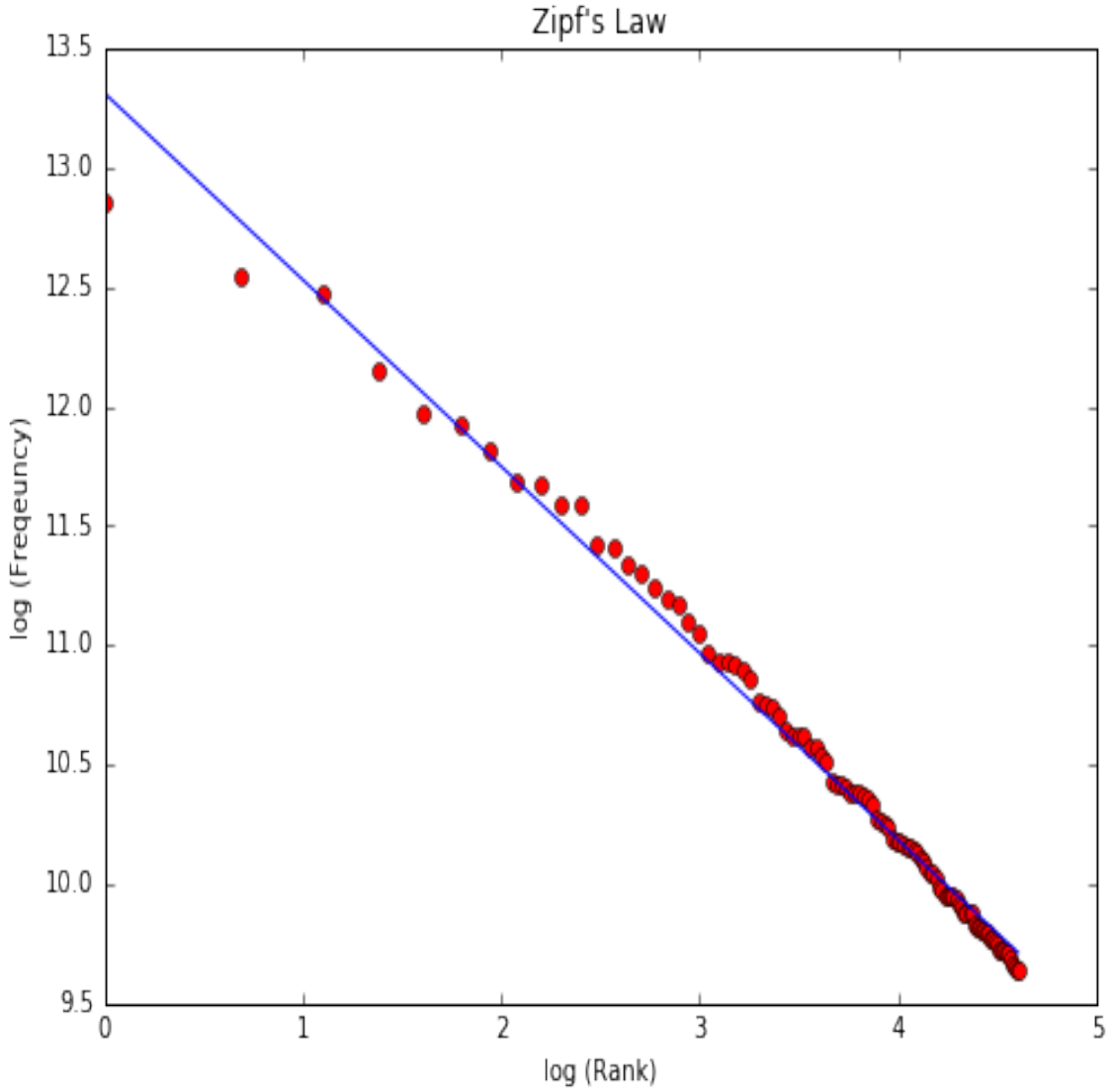

Figure 4.2: Unigrams frequencies follow Zipf's Law.

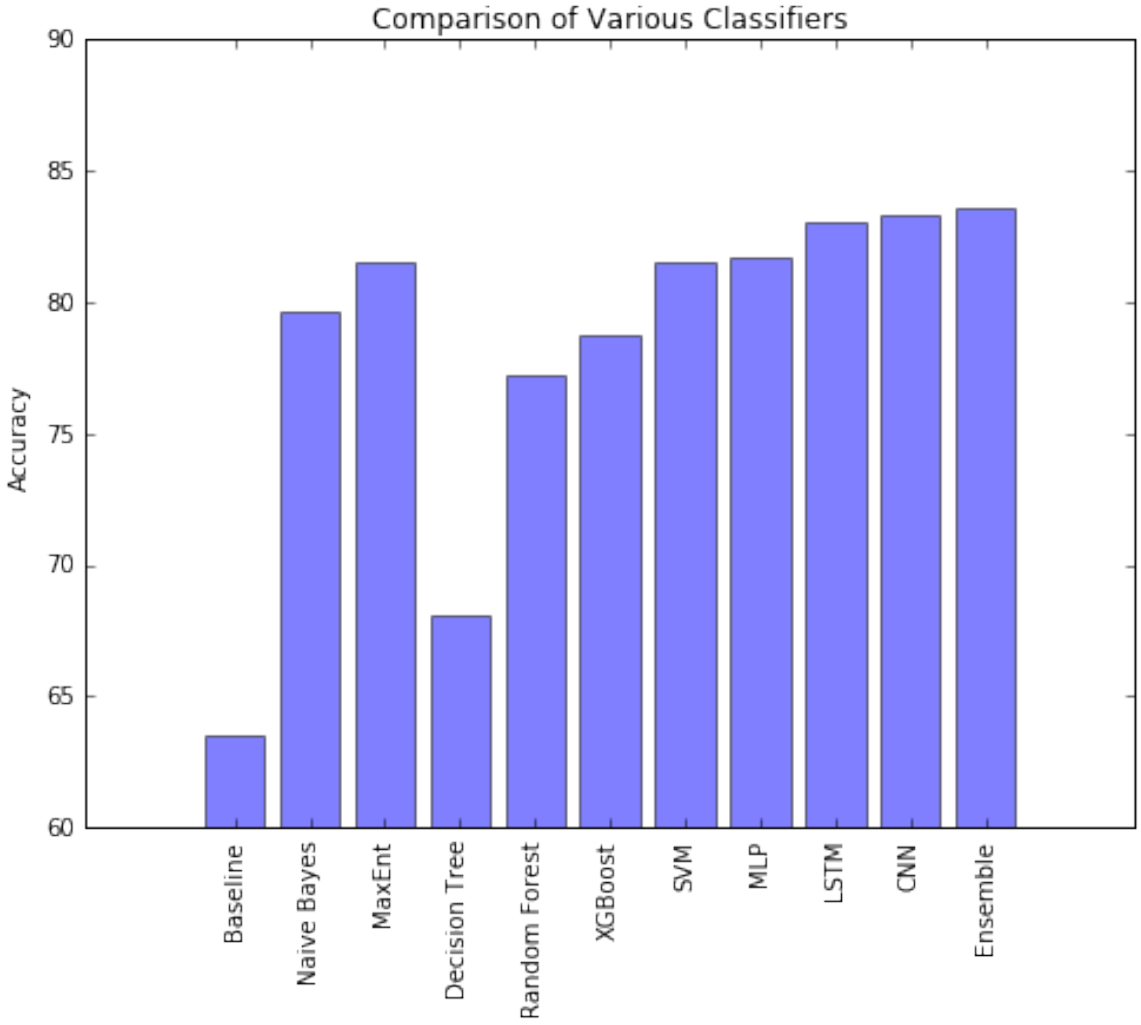

Figure 4.3: Comparison of accuracies of various models

Chapter 5

Conclusion

5.1 project conclusion

The provided tweets were a mixture of words, emoticons, URLs, hastags, user mentions, and symbols. Before training the we pre-process the tweets to make it suitable for feeding into models. We implemented several machine learning algorithms like Naive Bayes, Maximum Entropy, Decision Tree, Random Forest, XGBoost, SVM, Multi-Layer Perceptron, Recurrent Neural networks and Convolutional Neural Networks to classify the polarity of the tweet. We used two types of features namely unigrams and bigrams for classification and observes that augmenting the feature vector with bigrams improved the accuracy. Once the feature has been extracted it was represented as either a sparse vector or a dense vector. It has been observed that presence in the sparse vector representation recorded a better performance than frequency.

Neural methods performed better than other classifiers in general. Our best LSTM model achieved an accuracy of 0.83 on Kaggle while the best CNN model achieved 0.8334. The model which used features from our best CNN model and classifies using SVM performed slightly better than only CNN. We finally used an ensemble method taking a majority vote over the predictions of 5 of our best models achieving an accuracy of 0.8358.

5.2 Future scope

- Handling emotion ranges: We can improve and train our models to handle a range of sentiments. Tweets don't always have positive or negative sentiment. At times they may have no sentiment i.e. neutral. Sentiment can also have gradations like the sentence, This is good, is positive but the sentence, This is extraordinary. is somewhat more positive than the first. We can therefore classify the sentiment in ranges, say from -2 to +2.
- Using symbols: During our pre-processing, we discard most of the symbols like commas, full-stops, and exclamation mark. These symbols may be helpful in

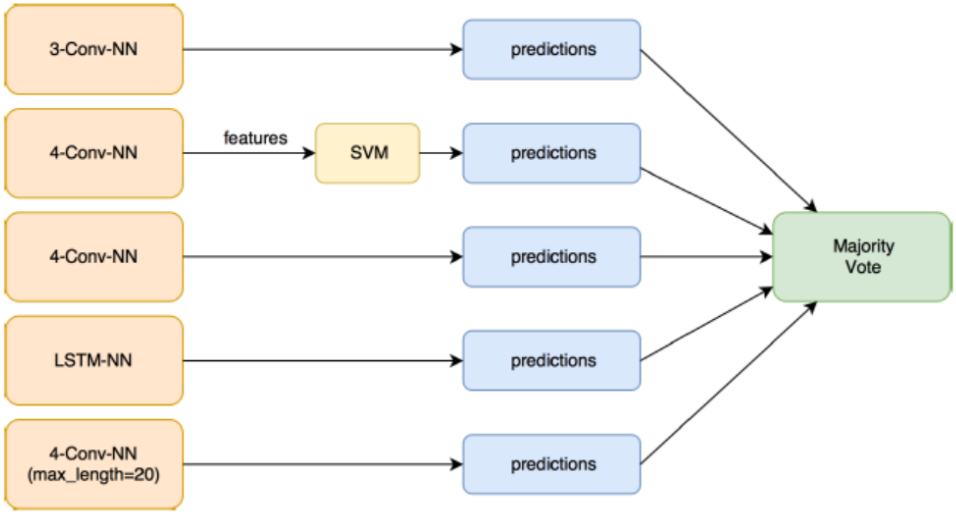

Figure 5.1: flowchart of majority voting ensemble

assigning sentiment to a sentence.

References

[1] A.Pak and P. Paroubek, "Twitter as a Corpus for Sentiment Analysis and Opinion Mining". In Proceedings of the Seventh Conference on International Language Resources and Evaluation, 2010, pp.1320-1326

[2] R. Parikh and M. Movassate,"Sentiment Analysis of User- Generated Twitter Updates using Various Classification Techniques",CS224N Final Report, 2009

[3] Go, R. Bhayani, L.Huang. "Twitter Sentiment Classification Using Distant Supervision". Stanford University, Technical Paper,2009

[4] L. Barbosa, J. Feng. "Robust Sentiment Detection on Twitterfrom Biased and Noisy Data". COLING 2010: Poster Volume,pp. 36-44

[5] Bifet and E. Frank, "Sentiment Knowledge Discovery inTwitter Streaming Data", In Proceedings of the 13th InternationalConference on Discovery Science, Berlin, Germany: Springer,2010, pp. 1-15.

[6] Agarwal, B. Xie, I. Vovsha, O. Rambow, R. Passonneau, "Sentiment Analysis of Twitter Data", In Proceedings of the ACL 2011Workshop on Languages in Social Media,2011 , pp. 30-38

[7] Dmitry Davidov, Ari Rappoport." Enhanced Sentiment Learning Using Twitter Hashtags and Smileys". Coling 2010: Poster Volumepages 241249, Beijing, August 2010

[8] Po-Wei Liang, Bi-Ru Dai, "Opinion Mining on Social Media Data", IEEE 14th International Conference on Mobile Data Management,Milan, Italy, June 3 - 6, 2013, pp 91-96, ISBN: 978-1-494673-6068-5, http://doi.ieeecomputersociety.org/10.1109/MDM.2013.

[9] Bollegala, D., Weir, D., Carroll, J.. Cross-Domain Sentiment Classification using a Sentiment Sensitive Thesaurus. Knowledge and Data Engineering, IEEE Transactions on, 25(8), 1719-1731,2013

[10] P. D. Turney, "Thumbs up or thumbs down?: semantic orientation applied to unsupervised classification of reviews," in Proceedings of the 40th annual meeting on association for computational linguistics, pp. 417–424, Association for Computational Linguistics, 2002

[11] Li, S., Xue, Y., Wang, Z., Zhou, G..“Active learning for cross-domainsentiment classification”. In Proceedings of the Twenty-Thirdinternational joint conference on Artificial Intelligence (pp. 2127-2133).AAAI Press,2013

[12] Taboada, M., Brooke, J., Tofiloski, M., Voll, K., Stede, M..“Lexicon based methods for sentiment analysis”. Computational linguistics, 2011:37(2), 267-307.