CROWD SOURCE MOVIE RATINGS BASED ON TWITTER DATA ANALYTICS

A Project

Presented to the faculty of the Department of Computer Science
California State University, Sacramento

Submitted in partial satisfaction of the requirements for the degree of

MASTER OF SCIENCE

in

Computer Science

By

Priya Holikatti

SPRING
2015
CROWD SOURCE MOVIE RATINGS BASED ON TWITTER DATA ANALYTICS

A Project

By

Priya Holikatti

Approved by:

__________________________________, Committee Chair
Du Zhang, Ph.D.

__________________________________, Second Reader
Meiliu Lu, Ph.D.

____________________________
Date

iii
Student:  Priya Holikatti

I certify that this student has met the requirements for format contained in the University format manual, and that this project is suitable for shelving in the Library and credit is to be awarded for the project.

__________________________, Graduate Coordinator       ___________________  
Jinsong Ouyang, Ph.D.        Date  

Department of Computer Science
Abstract

CROWD SOURCE MOVIE RATINGS BASED ON TWITTER DATA ANALYTICS

by

Priya Holikatti

Twitter is an online social networking and micro blogging site that has millions of users today. Twitter users ‘tweet’ at least twice a day on an average, thus generating a massive amount of data. This project tries to harness the data generated by Twitter into some meaningful information. The objective of this project is to give an accurate rating for a movie based on twitter chatter.

A user can search for any movie of his choice. The application uses the REST API provided by Twitter to get the most recent tweets about it. These tweets are classified into positive, negative or neutral tweets using the Naïve Bayes Classification. The retrieved tweets are then stored in the database along with the sentiment associated with it. The rating of the movie is the result of a mathematical formula based on the number of positive, negative and neutral tweets stored in the database.
The expected conclusion is that the rating will be more accurate than the rating for a movie when compared to ratings on IMDB, Rotten Tomato or other such websites, since it is designed to reflect the opinion of the people at the time of query.

_____________________, Committee Chair
Du Zhang, Ph.D.

________________________
Date
ACKNOWLEDGEMENTS

Firstly I would like to express my sincerest gratitude to Dr. Du Zhang for his continued guidance and advice throughout the course of this project. His guidance helped me all the time in rethinking ideas and implementing them in the project. Without his support this project would not have been a success.

I would like to thank Dr. Meiliu Lu for taking an interest in this project, and graciously taking the time to review it.

I would also like to thank all the professors and other faculty members of Department of Computer Science who have helped me and guided me towards becoming a graduate.

I would like to thank all my friends and family for their continuous support that has helped me achieve this personal goal.
# TABLE OF CONTENTS

<table>
<thead>
<tr>
<th>Chapter</th>
<th>Title</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>INTRODUCTION</td>
<td>1</td>
</tr>
<tr>
<td>1.1</td>
<td>Overview</td>
<td>1</td>
</tr>
<tr>
<td>1.2</td>
<td>More about Twitter</td>
<td>2</td>
</tr>
<tr>
<td>1.3</td>
<td>Scope</td>
<td>3</td>
</tr>
<tr>
<td>2.</td>
<td>SYSTEM DEVELOPMENT</td>
<td>4</td>
</tr>
<tr>
<td>2.1</td>
<td>Overview</td>
<td>4</td>
</tr>
<tr>
<td>2.2</td>
<td>Data Collection</td>
<td>4</td>
</tr>
<tr>
<td>2.3</td>
<td>Classification</td>
<td>10</td>
</tr>
<tr>
<td>2.3.1</td>
<td>Naïve Bayes Text Classification</td>
<td>10</td>
</tr>
</tbody>
</table>

Acknowledgements ........................................................................................................ vii

List of Tables ................................................................................................................ x

List of Figures ............................................................................................................. xi
# LIST OF TABLES

<table>
<thead>
<tr>
<th>Tables</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Table 3.1: Number of user reviews used – 1</td>
<td>20</td>
</tr>
<tr>
<td>Table 3.2: Number of user reviews used – 2</td>
<td>22</td>
</tr>
</tbody>
</table>
# LIST OF FIGURES

<table>
<thead>
<tr>
<th>Figures</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>Figure 2.1: Twitter profile used to authenticate the application</td>
<td>5</td>
</tr>
<tr>
<td>Figure 2.2: Auth token for secure requests</td>
<td>5</td>
</tr>
<tr>
<td>Figure 2.3: Code snippet for search API</td>
<td>6</td>
</tr>
<tr>
<td>Figure 2.4: Part of Twitter JSON object</td>
<td>9</td>
</tr>
<tr>
<td>Figure 2.5: Twitter object fields stored in the database</td>
<td>10</td>
</tr>
<tr>
<td>Figure 2.6: Bayer’s rule</td>
<td>12</td>
</tr>
<tr>
<td>Figure 2.7: Probability that a tweet belongs to category</td>
<td>12</td>
</tr>
<tr>
<td>Figure 2.8: Movie tweets stored in the database</td>
<td>14</td>
</tr>
<tr>
<td>Figure 2.9: Stop words</td>
<td>15</td>
</tr>
<tr>
<td>Figure 2.10: Training data collection</td>
<td>16</td>
</tr>
<tr>
<td>Figure 2.11: Query used to search movie name in MongoDB</td>
<td>17</td>
</tr>
<tr>
<td>Figure 3.1: Twitter Rating versus IMDB and Rotten Tomatoes – 1</td>
<td>19</td>
</tr>
<tr>
<td>Figure 3.2: Twitter Rating versus IMDB and Rotten Tomatoes – 2</td>
<td>21</td>
</tr>
<tr>
<td>Figure 3.3: Number of User Reviews used in Twitter Rating, IMDB and Rotten Tomatoes</td>
<td>22</td>
</tr>
<tr>
<td>Figure 3.4: Rating for “Birdman” using 300-tweets training data</td>
<td>24</td>
</tr>
<tr>
<td>Figure 3.5: Rating for “Birdman” using 500-tweets training data</td>
<td>25</td>
</tr>
</tbody>
</table>
Figure 3.6: An example of tweet posted by a different application .......................................... 26
Figure 3.7: Spam Tweets from different users .......................................................................... 27
Chapter 1

INTRODUCTION

1.1 Overview

Say a person is planning to go out to a movie on Saturday night and he wants to see reviews about the movie before investing his time and money on it. Now, assume the movie he wants to see has been released to the public just a day before. When he pulls up IMDB [1] website, the movie has a rating of 8.3 out of 10, implying that it’s an above average movie. For good measure, he also decides to check the reviews for it on Rotten Tomatoes [2] where it looks like the movie has a rating of 60%, implying that it is an average movie.

There may be many reasons why the ratings from these popular websites are contradicting each other. Some obvious ones are –

1. The time at which user checked the review: In the scenario described, the user is checking for reviews immediately after the movie has been released. A person, who saw the movie on Friday night, has not yet had an opportunity to access his IMDB account and write a review about the movie.

2. The kind of people who reviewed: Usually, IMDB only reflects the opinions of the few users who make a conscious effort to find the websites, create an account with them (which might not always be free) and then rate it out of 5 stars or 10 points, whichever method of measurement is supported by the website.
Wouldn’t it be nice if there were a single place on the Internet where you could see what the general public is saying about a movie at that point in time? There are many social platforms where people like to voice their opinion about issues that they care about. A ridiculous amount of data is generated when someone writes, blogs, comments or posts about topics. Twitter is among the more popular social networking websites where a user is allowed to post a comment in less than 140 characters. According to Statisticbrain.com, the average number of tweets per day is about 58 million. It is not wrong to assume that if someone went to a movie, they would come out and send a quick tweet from their cell phone about it for their friends and family. The more passionate moviegoers might vote or review the movie on IMDB or Rotten Tomatoes at their leisure, but the average person is more likely to send out a quick update on social media.

The idea is to harness this data from Twitter, which is easily available, and use it to construct some meaningful data for moviegoers.

1.2 More about Twitter

Twitter is a social networking website that allows you to send small bursts of information out into the world. Each such post is called a “tweet”. Twitter users can tag other users in their posts using a @ sign, and users can “follow” each other in order to receive updates about their latest posts. Tweets may include URLs pointing to other articles or photographs, and / or hash tags (#) that can be used to reference the tweet itself. Popular tweets can be re-tweeted by the followers of the original user. These posts, or topics
referred to in the posts can become a twitter “trend” if the hash tagged keyword is popular on Twitter at that particular time.

Twitter has half a billion user accounts and around 288 million monthly active Twitter users [3]. One can only imagine the sheer volume of the data generated. Users ranging from politicians to pop stars are using it massively to post content on the Web. Users comment on topics of interest, express their opinions and emotions to the world. Twitter has become a strong and popular social medium.

1.3 Scope

Twitter has introduced a Rate Limit on the volume of Tweets that can be downloaded using their APIs. Therefore, the scope of this project had to be reduced down to a limited amount of data that is available for querying. Also, requirements of the application needed the user to be able to query any movie of his/her choice. This implied the usage of a REST API provided by Twitter, which allows you to query with a search parameter live. This web application can be used by anyone who wants to see recent movie ratings. The result would be the most recent opinions of people about the movie, and a mathematical rating of the movie out of 10, 10 being the highest.

The system development of this application is explained in detail under Chapter 2. Performance of the application is discussed under Chapter 3. This is divided into 3 sections – correctness with respect to other popular websites, correctness with respect to feature set in training data and other issues with twitter data. Chapter 4 discusses related work, future enhancements and Chapter 5 concludes the report.
Chapter 2

SYSTEM DEVELOPMENT

2.1 Overview

This application can be viewed as an integration of three main modules:

1. Data collection.
2. Data Classification.

Data collection and classification happen in parallel. Rating module is invoked every time a user does “Search” operation so that the rating is calculated based on existing data in the database and new data returned by the Twitter API.

2.2 Data collection

Twitter data can be accessed through the public APIs provided by Twitter. These APIs can be accessed only by authenticated requests, which must be signed with valid Twitter user credentials. I created a special Twitter account (Figure 2.1) for this purpose, with valid OAuth keys to communicate with the Twitter Search API. Figure 2.2 shows the auth token requests from my application to the Twitter API.
Spring Social project [4] is a Java Library that enables applications to establish connections with social networks such as Facebook and Twitter. It has frameworks such as Spring Social Core, Spring Social Facebook, Spring Social Twitter and Spring Social LinkedIn. I have used the Spring Social Twitter extension to invoke the Twitter Search API for my application. The application is written using Java MVC Framework.

```java
TwitterTemplate twitter = new TwitterTemplate(
    "YxG95YYQ0Y0A8hBd8GkrCR",
    "7SZZOaDrpw8t8eLz5ZrXXNzxVULCukV5rMuxPww1McDoYUGvG",
    "2994077576-LUqPdPj0sjbff1ad9eHL97PGxU4OwrJew2GzJEH",
    "PfKeQ2j31Piy9EvIbck0p4Lw1UpeHsTNqw0AU4hsM7a7r";
```

**Figure 2.1:** Twitter Profile used to authenticate the application

**Figure 2.2:** Auth token for secure requests
Twitter Search API [5] is part of Twitter’s v1.1 REST API. It allows queries against the indices of recent or popular Tweets. It behaves similar to the Search feature available in Twitter mobile or web clients. It allows you to search for tweets based on parameters that are passed. The maximum number of characters that you can search for are 500 including operators. Language of the tweets can be restricted to English. Another interesting feature of the API is that it gives you an option to retrieve tweets generated before a given date, but this is not useful for the movie rating application.

In this application, I decided to search for “recent” tweets, so only the most recent tweets are grabbed. This conforms to the initial assumption that movie ratings tend to be more accurate based on the opinion of the crowd at that time. I also decided to restrict the language parameter to query only for “English” language tweets. This is done to simplify the process of evaluating the correct classification of the tweets. Figure 2.3 shown here is a small snippet of code used to achieve this.

```java
SearchParameters parameters = new SearchParameters(movieName);
parameters.count(100);
parameters.lang("en");
parameters.resultType(ResultType.RECENT);
SearchResults results =
twitter.searchOperations().search(parameters);
```

**Figure 2.3: Code snippet for search API**

Twitter returns about 100 tweets or less, depending on how recently there has been a search on the text entered by the user – the movie name, in the case of this application. The API returns tweet from a user as a JSON object. A sample twitter object [4] that is
returned is shown in the Figure 2.4. As it can be observed, this is a lot of information and we only need a few fields for the purpose of this application.

```json
{
    "statuses": [
        {
            "coordinates": null,
            "favorited": false,
            "truncated": false,
            "created_at": "Mon Sep 24 03:35:21 +0000 2012",
            "id_str": "250075927172759552",
            "entities": {
                "hashtags": [
                    {
                        "text": "freebandnames",
                        "indices": [20, 34]
                    }
                ],
                "hashtags": [20, 34]
            },
            "in_reply_to_user_id_str": null,
            "contributors": null,
            "text": "Thee Namaste Nerdz. #FreeBandNames",
            "metadata": {
                "iso_language_code": "pl",
                "result_type": "recent"
            },
            "retweet_count": 0,
            "in_reply_to_status_id_str": null,
            "in_reply_to_status_id_str": null
        }
    ]
}
```
"max_id": 250126199840518145,
"since_id": 24012619984051000,
"refresh_url":
"?since_id=250126199840518145&q=%23freebandnames&result_type=mixed&include_entities=1",
"next_results":
"?max_id=249279667666817023&q=%23freebandnames&count=4&include_entities=1&result_type=mixed",
"count": 4,
"completed_in": 0.035,
"since_id_str": "24012619984051000",
"query": "%23freebandnames",
"max_id_str": "250126199840518145"
}
}

**Figure 2.4: Part of Twitter JSON object**

The data this application is interested in are - Created time of the tweet, tweet ID, the actual text about the search string, username and language code. The most important data required for this application is the tweet text itself, the other parameters are specified only to narrow down the search results to the kind of tweets that we want to capture. This is a very crucial step since we can only get a limited number of tweets per call to the API and we want to make sure we get the relevant data.

The code snippet displayed in Figure 2.5 shows the relevant fields being stored in the database.
Figure 2.5: Twitter object fields stored in the database.

2.3 Classification

Apart from the tweets obtained from Twitter, the application also stores the sentiment associated with the tweet in the database. Sentiment of a tweet is categorized as positive, negative or neutral. This application uses the Naïve Bayes classification algorithm.

2.3.1. Naïve Bayes Text Classification

Naive Bayes is a technique used for constructing classifiers. Here, classifiers are similar to models that assign class labels to problem instances. The problem instances belong to a finite set of class labels and the feature values in the problem define which class the problem instance gets categorized into. A major advantage of using the algorithm for this application is that it only requires a small amount of training data to estimate the parameters necessary for classification.

Naïve Bayes classifier is not one algorithm for a classifier but a family of algorithms, all based on one principle: that the value of a particular feature is independent of the value of any other feature, given the class variable. For each known class value, the algorithm
1. Calculate probabilities for each attribute, conditional on the class value.

2. Use the product rule to obtain a joint conditional probability for the attributes.

3. Use Bayes rule to derive conditional probabilities for the class variable.

Once this has been done for all class values, the class with highest probability is displayed as the output.

Consider an example: a fruit may be considered to be an apple if it is red, round, and about 3" in diameter. A Naive Bayes classifier considers each of these features to contribute independently to the probability that this fruit is an apple, regardless of any possible correlations between the color, roundness and diameter features. It calculates the probability of the fruit being red, the fruit being round and the fruit being 3” in diameter. A joint probability is then calculated – the probability that the fruit is an apple, knowing the probability that it is red in color. It calculates conditional probability for the other classes as well – fruit being round and an apple, and fruit being 3” diameter and an apple.

2.3.2. Probabilities of Classification

Despite their naive design and apparently oversimplified assumptions, naive Bayes classifiers have worked quite well in many complex real-world situations. Many commercial text classification and sentiment analysis tools today have implemented some form of Bayes classification because of their proved efficiency.

According to the Bayes rule [9], the probability that an input belongs to a particular class can be calculated as shown in Figure 2.6.
Here, C is the class that a given input can belong to based on A, and A is the Attribute. With respect to the Twitter rating application, C will be the category – positive, negative or neutral and A will be the feature set. The output returned by the Bayes classifier in the application is shown in Figure 2.7. It first returns the category as Positive and then gives a probability of 0.78943694, indicating a fairly high change of this tweet reporting a positive opinion of the movie. The feature set attribute shows the features that were retained after processing the tweet for stop words. Hence the tweet “Insurgent was pretty cool #film #insurgent #DivergentSeries” is classified as positive, which is fairly accurate.

Classification [category=positive, probability=0.78943694, featureset=[Insurgent, was, pretty, cool, #film, #insurgent, #DivergentSeries]]

Classification [category=neutral, probability=0.01942467, featureset=[RT, @Divergent, Theo, James, in, 3D?, YES,, PLEASE,, #Insurgent, tix;, http://t.co/q2omrs0X2M, http://t.co/j3WPP9E9Ew]]

Classification [category=negative, probability=0.083147789, featureset=[@pechrush, n, me, bout, to, watch, #Insurgent, We're, the, only, losers, here,, everyone, else, is, watching, that, cool, movie, fast7!, http://t.co/cqXRxmAoN3]]

Figure 2.6: Bayes’s rule

Figure 2.7: Probability that a tweet belongs to category.
The second tweet in the figure is a retweet, indicated by “RT” at the beginning of the tweet. The feature set is not good enough for the classifier because it is returning a category as neutral and probability of 0.01942467. The third tweet is an example of a negatively classified tweet with a probability of 0.083147789. It has a low probability but it is correctly classified. This implies that the feature set was not rich enough for the classifier to categorize with a high probability but it was good enough to categorize correctly.

2.3.3. Implementation for this application

Multinomial Naïve Bayes Text classification algorithm [6] is used by this application to determine the category of a tweet in this application. The Classifier takes twitter text as input and compares it with the training data. It then determines the probability of the tweet being classified as positive, negative and neutral, and then multiplies all the probabilities and scales it against the probability of evidence. Output of the classifier is a category associated with the tweet.

The classified tweets are then stored in the database as shown in the database view below in Figure 2.8.

```json
{"_id" : NumberLong("579728496248913920"),  "_class" : "com.movie.rating.twitter.model.Movie",  "text" : "I love Benedict Cumberbatch in this movie he definitely deserved his Oscar nomination. #TheImitationGame",  "name": "#theimitationgame",}
"fromUser": "AgentSulaw", "languageCode": "en", "createdAt": ISODate("2015-03-22T19:36:58Z"), "classifier": "positive" } 

{"_id" : NumberLong("579727938209255424"), "_class" : "com.movie.rating.twitter.model.Movie", "text" : "RT @glaad: Congrats to #TheImitationGame for picking up a trophy at the #glaadawards!", "name": "#theimitationgame", "fromUser" : "sonjanuttall", "languageCode": "en", "createdAt" : ISODate("2015-03-22T19:34:45Z"), "classifier": "neutral" }

Figure 2.8: Movie tweets stored in the database

The Classification algorithm also gets rid of all words like “a, an, the, their ….etc” which can appear more than once in any sentence and has no sentiment associated with it. This is called feature selection. It removes all the noise around the data and keeps only relevant data. The stop words used by this application are shown in Figure 2.9.

stopwords = {"a", "about", "above", "after", "again", "against", "all", "am", "an", "and", "any", "are", "as", "at", "be", "because", "been", "before", "being", "below", "between", "both", "but", "by", "can", "did", "do", "does", "doing", "don", "down", "during", "each", "few", "for", "from", "further", "get", "had", "has", "have", "having", "he", "her", "here", "hers", "herself", "him", "himself", "his", "how", "i", "if", "im", "i'm", "in", "into", "is", "it", "its", "itself", "just", "me", "more", "most", "my", "myself", "no", "nor", "not", "now", "of", "off", "on", "once", "only", "or", "other", "our", "ours", "ourselves", "out", "over", "own", "rt", "s", "same", "she", "should", "so", "some", "such", "t", "than", "that", "the", "their", "theirs", "them", "themselves", "then", "there", "these", "they", "this", "those", "through", "to", "too", "under", "until", "up", "us",}
Consider, for example, the tweet: "I love Benedict Cumberbatch in this movie he definitely deserved his Oscar nomination. #TheImitationGame". In this, "I, in, this, he, his" will be ignored. This process is also called as stemming a sentence. This is one of the techniques used to increase the efficiency of Naïve Bayes algorithm.

2.3.4 Training Data

The data for training the classifier was collected by using the Twitter API. Live data was collected about some movies and exported into a csv file. The most challenging part in the building of this application was to develop an appropriate training set for rating movies based on Twitter user data. The tweets were extracted based on the movie name as a search string. This implied that all the tweets extracted to develop the training data had to be specifically written for a movie. These had to be preprocessed to make sure all spam, movie name, and other proper nouns like names of Actors were removed so that they don’t confuse the classifier. If an extracted movie name appeared in the positive training set, and a user searched for the rating of that movie, then it is more probable for that movie to be categorized as positive. This had to be avoided.

About 2000 tweets were collected for training the classifier. This initial data retrieved from the application was raw and had to be manually categorized as Positive, Negative
and Neutral tweets. Currently, the application is using 500 positive, 500 negative and 500 neutral tweets for training. This data was imported into the application as separate files and stored in the database. The incoming tweets are then classified based on this training data and stored in the database. A small portion of data from the positive training data collection is shown in Figure 2.10

Figure 2.10: Training data collection

2.3.5 Movie Rating

Rating is determined by calculating the average of positive tweets over total tweets. MongoDB is queried for total number of tweets and how they are classified for a particular movie. This query returns all the tweets stored in the database for that string. This was designed so the rating is as accurate as it can get. Twitter API has a rate limit which limits data collection to 100 tweets per request. The calculated rating gets more
accurate when the size of the dataset increases. The query runs after the new tweets are inserted so that it can collect both the new results as well as existing results in the database. Query used is shown in Figure 2.11.

```java
public List<Movie> getMovieTweets(String movieName) {
    List<Movie> movies = mongoOperation.find
        (Query.query(where("name") .is(movieName)), Movie.class);

    return movies;
}
```

**Figure 2.11: Query used to search movie name in MongoDB**

Rating out of 10 is calculated using the following formula -

Rate = (Number of positive tweets/(Total number of tweets – Neutral Tweets) * 100)

It is a simple formula that summarizes the polarity of the tweets by calculating the average of positive.
Chapter 3

PERFORMANCE EVALUATION

3.1 Overview

The performance of this application is evaluated by comparing its results with results from popular movie rating websites like IMDB and Rotten Tomatoes.

IMDB is an online database of information related to films, television, programs etc. The site allows registered users to rate any film on a scale of 1 to 10, apart from writing reviews about it. The site displays a weighted mean of user ratings and displays it next to the movie title. This site has 6.05 Daily Pageviews per Visitor [7] and an average of 15 million people visit the website per month.

Rotten Tomatoes is a website devoted to film reviews and news. It offers two types of scores for movies – Tomatometer critic aggregate score and Audience score. The Critic aggregate score reflects reviews and ratings from various newspaper writers or from people who belong to film critic associations. The Audience score is calculated based on user’s reviews and ratings. For the purpose of validating this application, only Audience score is considered. Registration is free but the site requests permission to view user’s social network profiles. It gets 3.60 Daily Pageviews per Visitor and an average of 13 million people visit the website per month. [8]
3.1.1 Correctness of the rating: volume of dataset

The rating as calculated by this application is being compared with IMDB rating and Rotten Tomatoes rating for the same movie. Figure 3.1 shows a graph comparing ratings for 8 random recently released movies. The graph has movies on the X – axis and ratings out of 10 on the Y – axis. It shows that the Twitter ratings from the application follow the same high points and low points as that of the other two websites. Table 3.1 shows the number of user reviews used to map the graph in Figure 3.1. The ratings collected show that the Twitter rating application is following similar trends as shown in IMDB and Rotten Tomatoes.
<table>
<thead>
<tr>
<th>Movie Name</th>
<th>Twitter Reviews</th>
<th>IMDB Reviews</th>
<th>RT Reviews</th>
</tr>
</thead>
<tbody>
<tr>
<td>#theimitationgame</td>
<td>499</td>
<td>216,597</td>
<td>86,411</td>
</tr>
<tr>
<td>#chappie</td>
<td>251</td>
<td>30,053</td>
<td>35,016</td>
</tr>
<tr>
<td>#birdman</td>
<td>398</td>
<td>226,332</td>
<td>80,236</td>
</tr>
<tr>
<td>#whiplash</td>
<td>300</td>
<td>182,570</td>
<td>53,662</td>
</tr>
<tr>
<td>#insurgent</td>
<td>682</td>
<td>20,836</td>
<td>50,176</td>
</tr>
<tr>
<td>#cinderella</td>
<td>299</td>
<td>20,873</td>
<td>45,679</td>
</tr>
<tr>
<td>#intothewoods</td>
<td>400</td>
<td>56,836</td>
<td>88,403</td>
</tr>
<tr>
<td>#thetheoryofeverything</td>
<td>429</td>
<td>139,889</td>
<td>67,150</td>
</tr>
</tbody>
</table>

Table 3.1: Number of user reviews used - 1

Ratings from Twitter application are observed to be of a higher value. This can be attributed to the fact that the number of users rating a movie on the other websites is almost 100x times than what the Twitter application is using. A small experiment was conducted with varying number of tweets used by Twitter application to confirm this theory.

Table 3.2 shows a higher number of Tweets that were used to rate the same movies and compared with IMDB and Rotten Tomatoes. The ratings observed with this data are displayed in Figure 3.2. It was observed that the ratings for some movies changed by a small fraction. This change indicated that the trend was even closer to what was seen in IMDB for some movies but further away for some others. For the movie “Chappie”, Twitter ratings using 251 tweets had a rating of 9, but using 999 tweets has a rating of 9.1 out of 10. This movie has a rating of 7.3 on IMDB and 6.3 on Rotten Tomatoes. Thus the
rating is going away from the trend. Whereas, for the movie “Insurgent” the rating went from 9.1 to 8.9 when a larger number of tweets were used. The rating for this movie is 7 on IMDB and 6.8 on Rotten Tomatoes. Thus the rating from application is closer to the trend. Similarly, the rating for movie “Into the woods” changed from 8.8 to 8.5 when a larger number of tweets were used. Again, this was closer to the trend seen in IMDB and Rotten Tomatoes. This shows that a large number of tweets collected may represent a more accurate rating of the movie.

![Figure 3.2: Twitter Rating versus IMDB and Rotten Tomatoes – 2](image-url)
<table>
<thead>
<tr>
<th>Movie Name</th>
<th>Twitter Reviews</th>
<th>IMDB Reviews</th>
<th>RT Reviews</th>
</tr>
</thead>
<tbody>
<tr>
<td>#theimitationgame</td>
<td>2027</td>
<td>216,597</td>
<td>86,411</td>
</tr>
<tr>
<td>#chappie</td>
<td>1999</td>
<td>30,053</td>
<td>35,016</td>
</tr>
<tr>
<td>#birdman</td>
<td>2003</td>
<td>226,332</td>
<td>80,236</td>
</tr>
<tr>
<td>#whiplash</td>
<td>2120</td>
<td>182,570</td>
<td>53,662</td>
</tr>
<tr>
<td>#insurgent</td>
<td>2102</td>
<td>20,836</td>
<td>50,176</td>
</tr>
<tr>
<td>#cinderella</td>
<td>2008</td>
<td>20,873</td>
<td>45,679</td>
</tr>
<tr>
<td>#intothewoods</td>
<td>1997</td>
<td>56,836</td>
<td>88,403</td>
</tr>
<tr>
<td>#thetheoryofeverything</td>
<td>2109</td>
<td>139,889</td>
<td>67,150</td>
</tr>
</tbody>
</table>

**Table 3.2: Number of user reviews used - 2**

Figure 3.3 shows another graph mapping the number of user reviews and ratings used for the same movies. This shows the large difference in the dataset used by the different applications.

**Figure 3.3: Number of Users Reviews used in Twitter Rating, IMDB and Rotten Tomatoes**
Currently the application can retrieve close to 100 tweets per API call, depending on how recently the movie was tweeted about. It has to be stressed that this is not enough data to determine the opinion of the Crowd for rating a movie. This is especially the case when dealing with Twitter data retrieval based on keyword search. The search results invariably include advertisements, retweets, and spam. These have to be filtered out or categorized separately in order to not interfere with the sentiment classification. Since Twitter has a rate limit on data available for use, the workaround for this issue in the application is to store the tweets every time it is retrieved, and calculate a new rating when the movie name is searched again. This results in an increase of data used to review movies, every time the application is used.

3.1.2 Correctness of the rating: feature set

Grabbing twitter chatter about movies and classifying them as positive, negative and neutral generated the training set for the Naïve Bayes classifier. A total of 500 tweets were classified for each category. Initially, 300 tweets per category were used to rate movies. The rating for “Birdman” movie with 300 tweets in training data is shown in Figure 3.4
The training data was then increased to 500 tweets per category. The rating for the same movie “Birdman” with a larger training data is shown in Figure 3.5.
Figure 3.5: Rating for “Birdman” using 500-tweets training data

The same pattern was observed for many other movies. Rating for “The Imitation Game” was 9.3 for 300-tweet training dataset and 500-tweet training dataset. Rating for “Whiplash” also remained 9.5 with both training datasets.

As it can be seen, the training data volume is not a big factor. The algorithm is designed to classify an input with many features with a relatively smaller attribute set in the training data. Thus, properly trained Naïve Bayes classifiers are staggeringly accurate for categorizing tweets.

3.2. Other issues and workarounds

Another major issue that needs to be addressed is the presence of spam accounts or user accounts set up to promote movies. Figure 3.6 shows an example of one such tweet. This
tweet was posted by another application on behalf of a Twitter user. Twitter API does not
provide a way to exclude such tweets. Most such tweets are categorized as Neutral
because of the presence of such data in the training set. But there is no specific format for
such tweets, and the Classifier gets confused when the feature set in the tweet has a
higher probability of appearing in a different category.

Another example of spam results returned when searching for the movie “Kingsmen” is
shown in Figure 3.3. Logic can be implemented to ignore all the tweets starting with
“RT” indicating that it is a retweet. But, it doesn’t make sense to avoid RT completely,
since it might be coming from a user who just thinks that another user has expressed his
opinions about a movie and the user wishes to emphasize the same. He could be
retweeting it to imply that he feels the same way. This makes dealing with Spam accounts
retweeting movie coupons / advertisements much harder. They usually contain words like
“thrilling” “winning” etc, which are in the positive training set for classification.
Language used by Twitter users is another aspect that needs to be considered. Twitter restricts tweets to 140 characters. This results in users getting creative when they have a lot more than 140 characters to write about. Twitter slang, and usage of abbreviations for common phrases are dependent on many things. Some of them include the location tweets are originating from, the local culture, or current pop culture references. The training data does not catch these. The algorithm is designed to work with a relatively small set of training data with respect to a large feature set. The training data used in this application captures a small set of slang or abbreviations, but not all. This results in reduced performance in categorizing tweets, in turn affecting the performance of the

Figure 3.7: Spam Tweets from different users
application. But from the evaluation conducted, this affects the application only because the dataset used is small. In case of large datasets, this issue will not affect the performance of the application.
Other works in this domain include retrieving twitter data with a hashtag of “#imdb”, and imdb data to provide a live database of ratings, in the paper titled “MovieTweetings: a Movie Rating Dataset Collected From Twitter” by Simon Dooms et al [10]. No other work has been done on using twitter data exclusively for rating movies.

In future enhancements, more categories can be introduced to classify tweets – extremely positive, mildly positive, extremely negative, mildly negative, neutral, and irrelevant. This can be used to improve the rating formula to make it more accurate. A weight can be associated with each category and then calculate the average. For example, if extremely positive has a weight of 5 and mild has a weight of 3, average rating can be calculated as follows:

\[
\text{Rating} = \frac{(\text{Extreme P} \times 5) + (\text{Mild P} \times 3)}{(\text{Total} - \text{neutral} - \text{irrelevant}) \times 8}
\]

Where Extreme P is extremely positive and Mild P is mildly positive.

More number of classes implies increase in number of attributes for the classifier to compare the input with. This should show some difference in the outcome of the classifier and hence difference in the rating shown by the application.
Chapter 5

CONCLUSION

Twitter data effectively manages to capture the opinions and emotions of the crowd and Twitter APIs make it fairly easy to gather this information and analyze it. This web application indeed manages to use this massive amount of data to provide a meaningful and useful result. Because of the rate limit introduced by Twitter, this is currently an academic implementation of the idea to use twitter data for rating a movie. If all the tweets containing the search string for a movie name can be captured and analyzed, a more accurate conclusion can be drawn.

The training data used by the classification algorithm has only 500 tweets per category, this is a relatively small amount compared to the amount of input data that is being categorized. The results have been encouraging with the small set; I expect the result to be even more impressive with a larger selection of natural language slang in the training data.

As the application is used frequently, the dataset grows with it. This results in the rating of a movie always being up to date with public opinion. The best use case is to search for recent movies. Simply because that is when the crowd seems to be tweeting most about movies.
Twitter JSON Object

```json
{
    "statuses": [
        {
            "coordinates": null,
            "favorited": false,
            "truncated": false,
            "created_at": "Mon Sep 24 03:35:21 +0000 2012",
            "id_str": "250075927172759552",
            "entities": {
                "urls": [],
                "hashtags": [
                    {
                        "text": "freebandnames",
                        "indices": [20, 34]
                    }
                ],
                "user_mentions": []
            }
        },
        {
            "in_reply_to_user_id_str": null,
            "contributors": null,
            "text": "Aggressive Ponytail #freebandnames",
            "metadata": {
                "iso_language_code": "en",
                "result_type": "recent"
            },
            "retweet_count": 0,
            "in_reply_to_status_id_str": null,
            "id": 250075927172759552,
            "geo": null,
            "retweeted": false,
            "in_reply_to_user_id": null,
            "place": null,
            "user": {
```
"profile_sidebar_fill_color": "DDEEF6",
"profile_sidebar_border_color": "C0DEED",
"profile_background_tile": false,
"name": "Sean Cummings",
"profile_image_url":

"http://a0.twimg.com/profile_images/2359746665/1v6zfgqo8g0d3mk7ii5s_normal.jpeg",
"created_at": "Mon Apr 26 06:01:55 +0000 2010",
"location": "LA, CA",
"follow_request_sent": null,
"profile_link_color": "0084B4",
"is_translator": false,
"id_str": "137238150",
"entities": {
  "url": {
    "urls": [
      {
        "expanded_url": null,
        "url": "",
        "indices": [0, 0]
      }
    ]
  },
  "description": {
    "urls": [
      
    ]
  }
},
"default_profile": true,
"contributors_enabled": false,
"favourites_count": 0,
"url": null,
"profile_image_url_https":

"https://si0.twimg.com/profile_images/2359746665/1v6zfgqo8g0d3mk7ii5s_normal.jpeg",
"utc_offset": -28800,
"id": 137238150,
"profile_use_background_image": true,
"listed_count": 2,
"profile_text_color": "333333",
"lang": "en",
"followers_count": 70,
"protected": false,
"notifications": null,
"profile_background_image_url_https": "https://si0.twimg.com/images/themes/theme1/bg.png",
"profile_background_color": "C0DEED",
"verified": false,
"geo_enabled": true,
"time_zone": "Pacific Time (US & Canada)",
"description": "Born 330 Live 310",
"default_profile_image": false,
"profile_background_image_url": "http://a0.twimg.com/images/themes/theme1/bg.png",
"statuses_count": 579,
"friends_count": 110,
"following": null,
"show_all_inline_media": false,
"screen_name": "sean_cummings"
},
"in_reply_to_screen_name": null,
"source": "<a>Twitter for Mac</a>",
"in_reply_to_status_id": null
},
{
"coordinates": null,
"favorited": false,
"truncated": false,
"created_at": "Fri Sep 21 23:40:54 +0000 2012",
"id_str": "249292149810667520",
"entities": {
"urls": [
],
"hashtags": [
{
"text": "FreeBandNames",
"indices": [20, 34]
}
],
"user_mentions": [
]
},
"in_reply_to_user_id_str": null,
"contributors": null,
"text": "Thee Namaste Nerdz. #FreeBandNames",
"metadata": {
"iso_language_code": "pl"}
I will sell you some records then, here in Durham, North Carolina. Fun will happen.
"text": "Mexican Heaven, Mexican Hell #freebandnames",
"metadata": {
  "iso_language_code": "en",
  "result_type": "recent"
},
"retweet_count": 0,
"in_reply_to_status_id_str": null,
"id": 249289491129438208,
"geo": null,
"retweeted": false,
"in_reply_to_user_id": null,
"place": null,
"user": {
  "profile_sidebar_fill_color": "99CC33",
  "profile_sidebar_border_color": "829D5E",
  "profile_background_tile": false,
  "name": "Thomas John Wakeman",
  "profile_image_url": "http://a0.twimg.com/profile_images/2219333930/Froggystyle_normal.png",
  "created_at": "Tue Sep 01 21:21:35 +0000 2009",
  "location": "Kingston New York",
  "follow_request_sent": null,
  "profile_link_color": "D02B55",
  "is_translator": false,
  "id_str": "70789458",
  "entities": {
    "url": {
      "urls": [
        { "expanded_url": null, "url": "", "indices": [0, 0] }
      ]
    }
  }
}
null}
"truncated": false,
"created_at": "Fri Sep 21 22:51:18 +0000 2012",
"id_str": "249279667666817024",
"entities": {
  "urls": [
  ],
  "hashtags": [
    {
      "text": "freebandnames",
      "indices": [
        20,
        34
      ]
    }
  ],
  "user_mentions": [
  ]
},
"in_reply_to_user_id_str": null,
"contributors": null,
"text": "The Foolish Mortals #freebandnames",
"metadata": {
  "iso_language_code": "en",
  "result_type": "recent"
},
"retweet_count": 0,
"in_reply_to_status_id_str": null,
"id": 249279667666817024,
"geo": null,
"retweeted": false,
"in_reply_to_user_id": null,
"place": null,
"user": {
  "profile_sidebar_fill_color": "BFAC83",
  "profile_sidebar_border_color": "615A44",
  "profile_background_tile": true,
  "name": "Marty Elmer",
  "profile_image_url": "http://a0.twimg.com/profile_images/1629790393/shrinker_2000_trans_normal.png",
  "created_at": "Mon May 04 00:05:00 +0000 2009",
  "location": "Wisconsin, USA",
  "follow_request_sent": null,
  "profile_link_color": "3B2A26",
  "is_translator": false,
  "id_str": "37539828",
  "profile_text_color": "333333",
  "profile_use_background_image": false,
  "profile_image_url_https": "https://a0.twimg.com/profile_images/1629790393/shrinker_2000_trans_normal.png"
}
"entities": {
  "url": {
    "urls": [
      {
        "expanded_url": null,
        "url": "http://www.omnitarian.me",
        "indices": [
          0,
          24
        ]
      }
    ]
  },
  "description": {
    "urls": [
      
    ]
  }
},
  "default_profile": false,
  "contributors_enabled": false,
  "favourites_count": 647,
  "url": "http://www.omnitarian.me",
  "profile_image_url_https": "https://si0.twimg.com/profile_images/1629790393/shrinker_2000_trans_normal.png",
  "utc_offset": -21600,
  "id": 37539828,
  "profile_use_background_image": true,
  "listed_count": 52,
  "profile_text_color": "000000",
  "lang": "en",
  "followers_count": 608,
  "protected": false,
  "notifications": null,
  "profile_background_image_url_https": "https://si0.twimg.com/profile_background_images/106455659/rect6056-9.png",
  "profile_background_color": "EEE3C4",
  "verified": false,
  "geo_enabled": false,
  "time_zone": "Central Time (US & Canada)",
  "description": "Cartoonist, Illustrator, and T-Shirt connoisseur",
  "default_profile_image": false,
  "profile_background_image_url": "http://a0.twimg.com/profile_background_images/106455659/rect6056-9.png",

BIBLIOGRAPHY


[February 2015]


[January 2015]


http://www.sussex.ac.uk/Users/christ/crs/ml/lec02b.html

[February 2015]


Collected From Twitter. Ghent University, Ghent, Belgium, 2013.