MovieBuzz

An application to compare movies based on IMDB rating and public opinion from Twitter data

Sumesh Balan, Ismail Vandeliwala

CSE 6339 – Data Science and Computational Journalism
Advisor – Dr. Chengkai Li
05-07-2015

Table of Contents

Introduction

MovieBuzz is an application developed to compare movies based on IMDB rating and public opinion based on Twitter data. IMDB is a popular database of movies, where users can rate a movie based on a scale of 1 – 10. The IMDB movie community is not comparable in size or popularity with social networking platforms such as Twitter, Facebook and LinkedIn etc. The motivation for developing MovieBuzz is based on the intuition that public's opinion about a movie could be different than the ratings provided by small communities such as IMDB, as relatively a very small population uses such websites.

However, designing and implementing efficient and provably correct application is very challenging. We restricted our focus only on the rating value provided by IMDB and available tweets from Twitter via streaming API. We did not set any threshold for the number of tweets, so the total number of tweets analysed varies across a wide range. To the best of our knowledge there is no such system exists today. *Our main contributions include*:

- An efficient and uninterrupted way to download tweets from Twitter
- Classification of tweets as positive or negative using Naive Bayes classifier, using NLTK movie review corpus as the training data
- Integration of IMDB datasets to the MovieBuzz web application, plain text datasets are migrated to MySQL DB
- An interactive web application for comparing 2 – 5 movies and extensible for any number of movies in the future
- Multiple visualizations for plotting the comparison results to user

Data Collection

We downloaded IMDB data sets from the ftp site ftp://ftp.fu-berlin.de/pub/misc/movies/database/. Data is in plain text files in compressed format,

size ~ 150 MB. The IMDB datasets gets updated every two weeks so it is not possible to include the latest releases in the application. Data collection from twitter is challenging, as connections will be terminated if the connection is kept alive for a long duration. We implemented a system to overcome this obstacle, developed a python application with timers along with the MAC-OSX utility *launchd*. We could download ~1.5 million tweets in three weeks time, for 30 different movies. The main limitation of streaming API is that it will provide only current data; there is absolutely no way to download older data other than using the paid REST APIs of Twitter. Because of this limitation, we restricted our initial implementation based on a set of 30 movies, which are released in the late 2014 and early 2015. All source code and documentation of the application is placed at the git repository https://github.com/sumeshnb/6339Project

Data Cleaning

Twitter streaming API provides tweets in JSON format. For our initial implementation, we used the value of column "text" which was relevant to our project. We parsed the files to filter tweets out of the JSON data. People posts tweets usually with URL or hash (#) tag followed by movie name or @ followed by user name. These kinds of data in tweets removed from the text, as they are not relevant for prediction. Also special symbols such as ',', '.', ';', ':', "" and many more were removed from the data.

Tweets Classification

We used Python NLTK Naive Bayes Classifier for classifying the tweets into positive and negative. NLTK has movie corpus data, which has the reviews classified into positive and negative. We used movie corpus data as training data to build classifier. For each of the movie tweets we ran this classifier, classified the tweets into positive and negative and stored the results to CSV files. Also we count the total number of tweets, total number of positive tweets and total number of negative tweets for all the movies and stored those results along with the movie name into a single file. The CSV file with movie names mapped to total tweets, positive tweets and negative tweets is one of the data source for the application, and the other data source is MySQL DB with IMDB dataset.

Normalization of Data

We normalized both the imdb ratings and Twitter positive tweets to a scale of 0 - 100. IMDB rating is multiplied the by 10 to give the rating out of 100. We divided the positive tweets by total number of tweets and then multiplied by 100 to give the rating out of 100.

$$\text{Normalized Imdb Rating} = \text{imdb Rating} * 10$$

$$\text{Twitter Rating} = round(\frac{num\ of\ posittive\ tweets}{number\ of\ total\ tweets} * 100)$$

For ex: Let the imdb rating for a movie be 8.4 then our rating would be 84. If we have 2000 positive tweets out of 3000 tweets, then according to the formula our rating would be 67. In this way we can compare the critics rating with the public opinions.

Evaluation

We took the random sample of 500 tweets of different movies and classified them into positive and negative manually. We then compared those tweets with the results of the classifier to see how accurate our classifier is.

Let's check some of the wrongly classified tweets:

1) *just saw the movie did cry for paul walker go watch it*: negative

It was classified as negative but we know that people cried for that movie because Paul Walker is dead and they miss him. So it should have been classified as positive.

2) *get hard was funny*: negative

3) *get hard is very funny*: positive

Above both the example should have been classified positive but the word "hard" plays the role in classifying one of them as negative. "Very funny" overrule the word "hard" to classify as positive but in the first one the word "hard" overrules the word "funny".

4) *the movie was so cuteee i almost cried*: negative

Again this was classified as negative. Word "Cuteee" should have classified it as positive but we all know that word is not spelled correctly. We need more such of test data to make our classifier classify such data as positive.

5) *insurgent is so good*: negative

Classifier does not know that word insurgent is a movie and its weight does not matter while classifying as positive or negative.

6) *this prince in cinderella is not charming at all*: positive

It should have been classified as negative. The word "charming" increased the chances of classifying it as positive.

Predicted	**Actual**		
	Positive	Negative	
Positive	179 (TP)	14 (FP)	193
Negative	75 (FN)	32 (TN)	107
	254	46	300

Accuracy = ((TP +TN)/300)*100 = ((179 + 32)/300)*100 = 70.33%

Sensitivity or TP rate or Recall is the percentage of positive tweets that were correctly identified as positive.

Sensitivity = (179/254)*100 = 70.47%

Specificity or TN rate is the percentage of negative tweets that were correctly identifies as negative.

Specificity = (32/46)*100 = 69.57%

Prevalence is the proportion of tweets found to be positive

Prevalence = (254/300)*100 = 84.67%

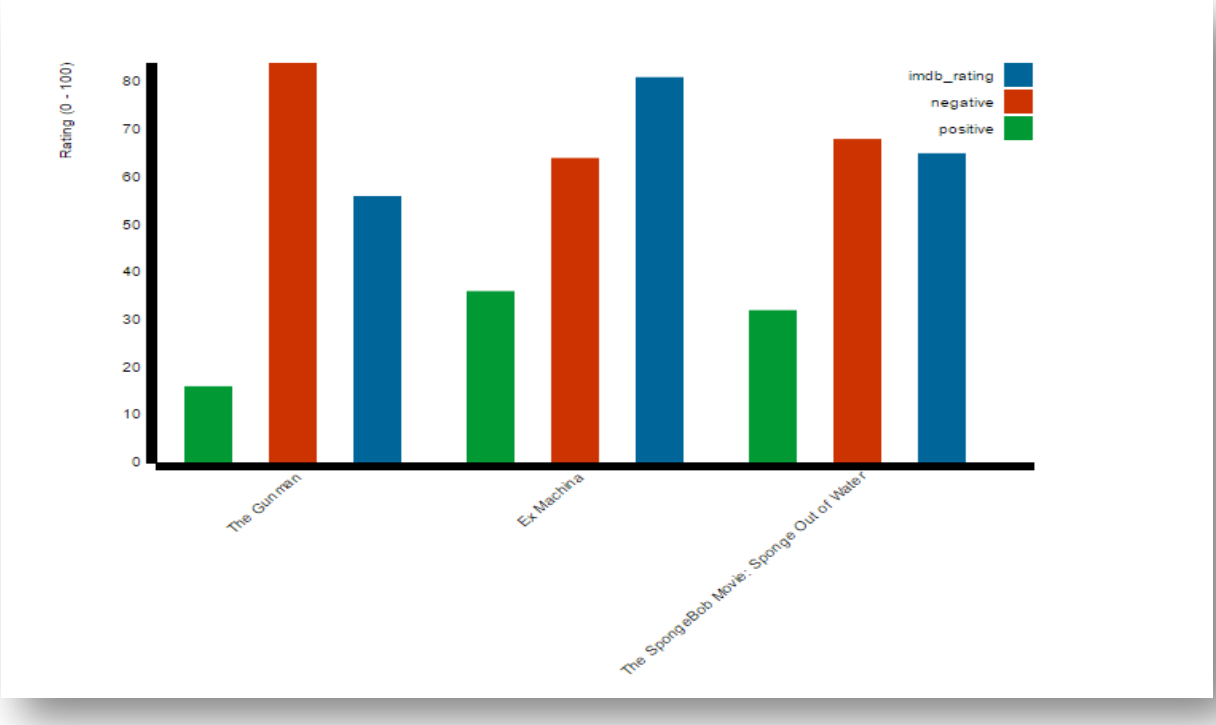
Precision is the proportion of tweets that are relevant

Precision = (179/193)*100 = 92.75%

Experiments

Experiment – 1: Compared 3 movies

- Gunman
  - Imdb Rating: 56, Twitter Positive Rating: 17, Twitter Negative Rating: 83
- Ex Machina
  - Imdb Rating: 81, Twitter Positive Rating: 38, Twitter Negative Rating: 62
- SpongeBob Movie: Sponge Out of water
  - Imdb Rating: 65, Twitter Positive Rating: 32, Twitter Positive Rating: 68

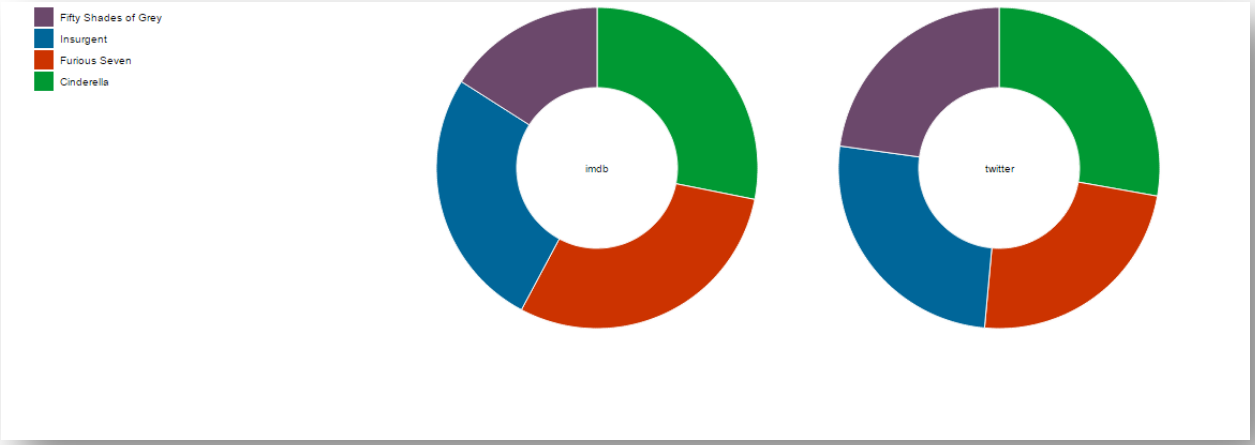


Comparison between normalized ratings for imdb and twitter - positive and negative comments

Conclusion: Even if the imdb rating is high, it does not mean the movie is popular among people.

Experiment -2: Compared 4 movies:

- Cinderella - Imdb Rating: 74, Positive Tweets: 127599
- Furious Seven - Imdb Rating: 78, Positive Tweets : 143930
- Insurgent - Imdb Rating: 69 , Positive Tweets : 44585
- Fifty Shades of Grey - Imdb Rating: 42, Positive Tweets : 78461

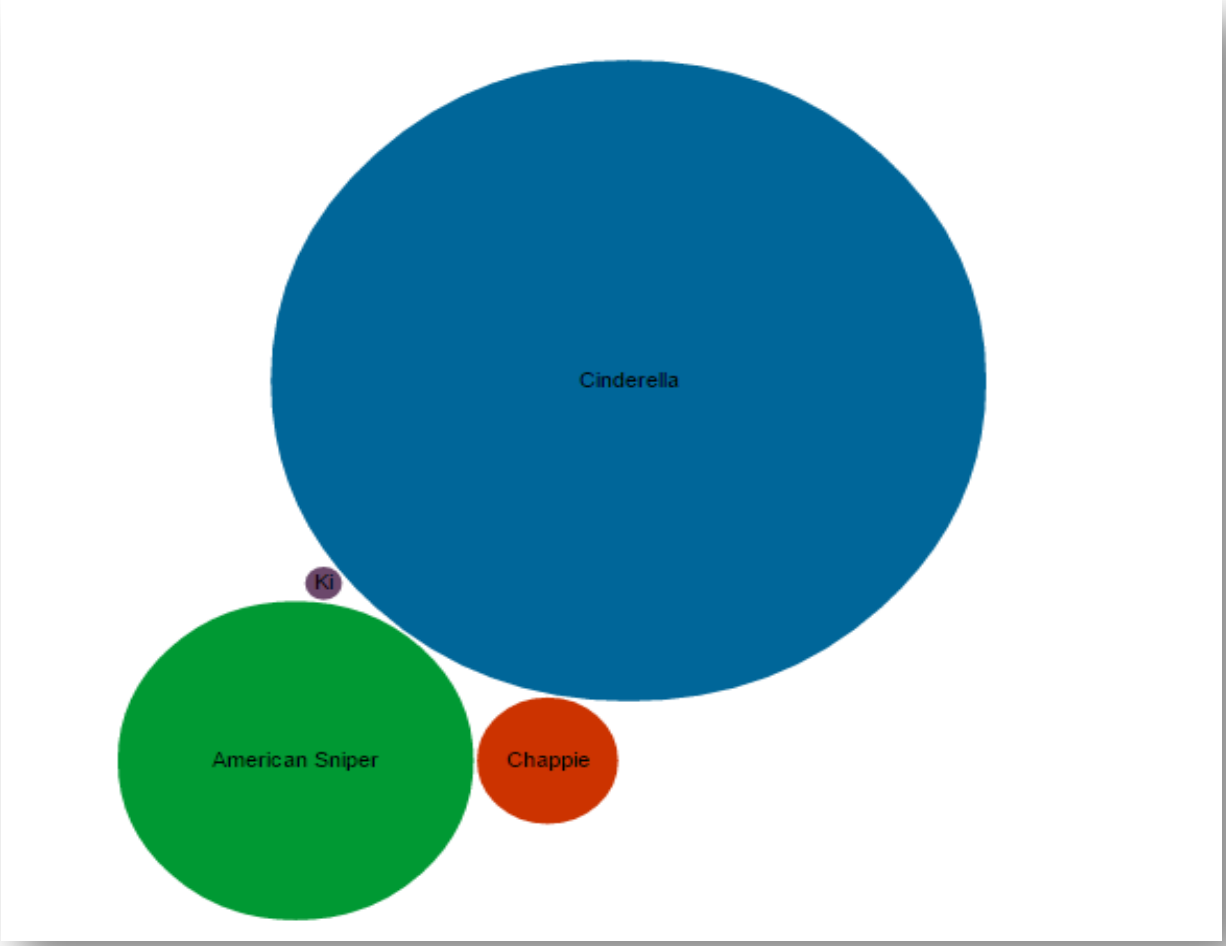


Comparison between imdb ratings and normalized positive number of tweets

Conclusion: If the number of positive tweets is high then those movies are considered better among the general public irrespective of imdb rating.

Experiment -3: Compared 4 movies

- Cinderella - Total Tweets: 186116
- American Sniper -Total Tweets: 38225
- Chappie - Total Tweets: 4960
- Kill me three times - Total Tweets: 864



Comparison of total number of tweets for selected movies

Conclusion: If there is more number of total tweets, the movie is more popular in general public.

Web-Application

The model of the movie comparison discussed above is implemented as a single page web-application (SPA). We used python based webserver Flask in the backend and HTML front end for presentation.

User Interaction:

1) User selects movie and click 'Add to compare' button.

2) If number of movies selected is greater than 5 then

    a) Program gives the notification message "Cannot add more movie"

3) Else

    a) Program fetches imdb data from imdb database.

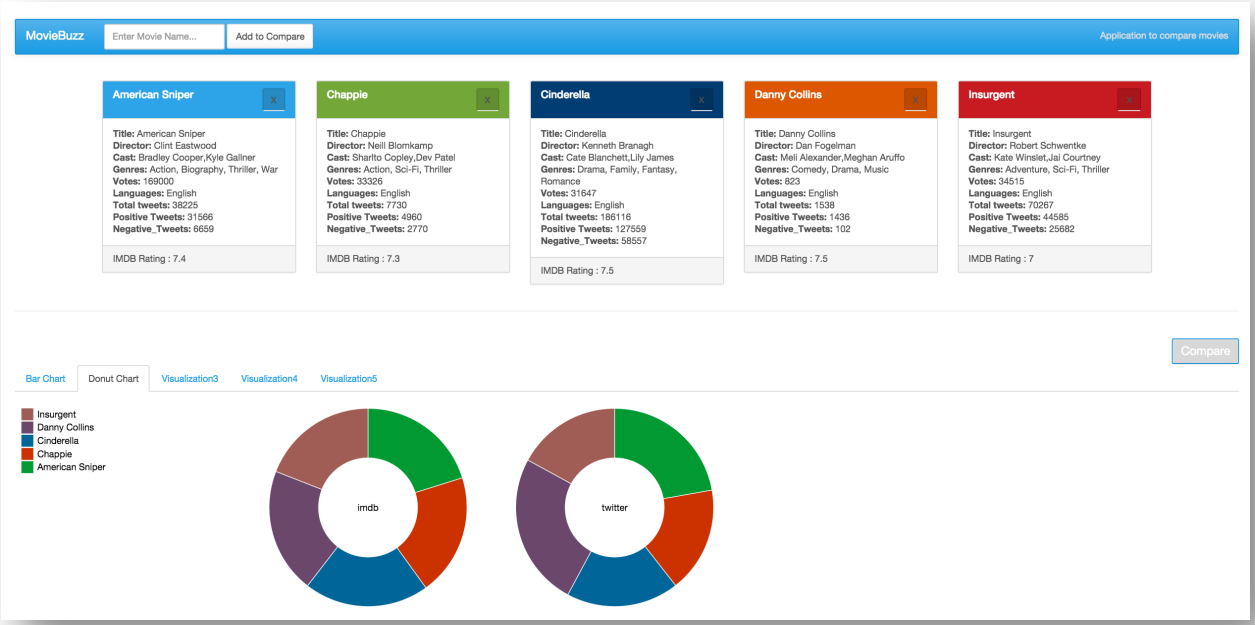
    b) Program fetches twitter data from the file.

    c) Normalize the imdb and twitter data.

    c) Display movie details and twitter statistics on web page

4) User clicks "Compare" button

5) User can choose any one of the visualization tabs to see the corresponding visualization



A screen shot of the application showing comparison of 5 movies

The details of various technology used is enumerated in the below table

Language/Framework	Brief explanation
HTML	Presentation of data in front-end
JavaScript	Scripting in front end
JQuery	JavaScript frame work for DOM manipulation
JQuery-UI	UI elements in HTML
Bootstrap	Style sheet for presentation in HTML
D3JS	Visualization of various charts
Flask	Python web framework
IMDBPy	Python package to query IMDB dataset
Tweepy	Python package for accessing Twitter streaming API
MySQL	Database to store IMDB dataset

Conclusions and Future Work

It is found that deriving the rating of movies based on multiple sources will increase the accuracy of rating. In the initial implementation, only IMDB and Twitter are considered. Similarly data from other social media websites can be included in arriving at conclusion, this will be a future enhancement to the current system. In the current implementation, only comparison of ratings are done, but in future we can come up with a single combined rating by analyzing all the sources of data. The current implementation is limited to 30 movies, but it is possible to extend the framework for all the movies that will be released from now on.

Bibliography

http://flask.pocoo.org/

http://imdbpy.sourceforge.net/

http://www.tweepy.org/

https://jquery.com/

http://getbootstrap.com/

http://d3js.org/

https://jqueryui.com/