Predicting the Winner of Delhi Assembly Election, 2015 from Sentiment Analysis on Twitter Data-A BigData Perspective

Lija Mohan and Sudheep Elayidom
Division of Computer Science, Cochin University of Science and Technology, India

Abstract: *Social media is currently a place where people create and share contents at a massive rate. Because of its ease of use, speed and reach, it is fast changing the public discourse in society and setting trends and agendas in different topics including environment, politics technology, entertainment etc. As it is a form of collective wisdom, we decided to investigate its power at predicting real-world outcomes. The objective was to design a Twitter-based sentiment mining. We introduce a keyword-aware user-based collective tweet mining approach to rank the sentiment of each user. To prove the accuracy of this method, we chose an Election Winner Prediction application and observed how the sentiments of people on different political issues at that time got reflected in their votes. A Domain thesaurus is built by collecting keywords related to each issue. Twitter data being huge in size and difficult to process, we use a scalable and efficient Map Reduce programming model-based approach, to classify the tweets. The experiments were designed to predict the winner of Delhi Assembly Elections 2015, by analyzing the sentiments of people on political issues and from this analysis, we accurately predicted that Aam Admi Party has a higher support, compared to Bharathiya Janatha Party (BJP), the ruling party. Thus, a Big Data Approach that has widespread applications in today's world, is used for sentiment analysis on Twitter data.*

Keywords: *Election winner prediction, big data, sentiment analysis, tweet mining, map reduce*.

Revived February 26, 2016; accepted June 29, 2017

1. Introduction

Large amount of data characterized by high volume, velocity and variety, which cannot be handled using traditional architecture, is called Big Data [4]. Through proper and meaningful analysis of data, many of the real-world problems can be solved. The authors propose a keyword based technique to extract the sentiment of people. To analyze the efficiency of our algorithm we integrated the algorithm with the interesting 'election winner prediction' (more specifically The Delhi Assembly Elections 2015) application. Tweets are analyzed to obtain the sentiments.

1.1. Delhi Assembly Elections, 2015

In January 2015, the Governor announced that the state assembly elections for 70 constituencies will be held in February 2015, thus setting the stage for the most thrilling Delhi political battle. On February 10th, the Election results were out and the Aam Admy Party (AAP) won the battle, grabbing 67 seats to their credit. Bharathiya Janatha Party (BJP) got the remaining 3 seats and Congress did not obtain any. Even though exit polls predicted the victory of AAP, the margin of victory remained unpredicted by any of these. Most of the exit polls predicted 30 to 35 seats for AAP and 25 to 30 for BJP.

1.2. Role of Social Media in Delhi Election 2015

The Assembly Election conducted in Delhi, was extra ordinary for many reasons. Irrespective of becoming the largest and the most influential parties, BJP and Congress suffered a huge loss in the battle, to an emerging party called Aam Admy, which came into existence in 2013. Even though the failure of Congress was predicted by Exit Polls, the terrible failure of BJP, left everybody astounded. Social media sites like Twitter and Facebook also took part in the political battle for a long period before the elections. A flurry of tweets kept pouring on, on February 7th 2015, the voting day, which reflected the intense political battle that was going to take place. Twitter Analytics revealed that #Delhi Votes was the top trending hashtag on that day. This article analyzes the sentiments from tweets containing hashtags and words like '@AamAdmiParty' ,'#Kejriwal', 'Kejriwal', #Vote4MufflerMan, #PressJhaaduButton, '#KiranBedi', 'KiranBedi', '@BJPDelhiState','#DelhiDecides',"#Delhi Votes.'DelhiVotes'.

2. Background

Sentiment Analysis is an exciting field of research in text mining or natural language processing for the treatment of opinions, sentiments, characteristics,

behavior and subjectivity of text [23]. This is an important information as the sentiment of one person can influence other person as well. This phenomenon takes place in social media as well.

2.1. Motivation and Contribution of our Research

Explanatory power of tweet feeds for predicting the election results is considered ambiguous in nature as studied by Gayo-Avello [8] and O'Connor *et al.* [17]. For example, Pang and Lee [20] proved that, mere volume analysis of political tweets accurately predicted the election result in Germany whereas the sentiment analysis failed to predict the 2008 US Presidential election, as proposed by Gayo-Avello [7]. Hence, authors experimented to investigate on this topic.

Predictive Analysis techniques are usually used by statisticians to identify the winners in elections. But, it does not take into account the current political situations. During elections, each real time issue has high impact on people and thus their votes. Real time sentiment mining is the only solution here. Also, predictive analysis accuracy will be very low to predict the success rate of a newly formed party.

The applications of sentiment analysis include customer behavior identification, feedback analysis, effective marketing strategy identification etc. There exist a lot of articles describing different techniques and methods for sentiment analysis. But, as per literature, ours is the first method which studies the effect of "Trending issues" of social media contents on the sentiment of users. Some of the applications where these "trending issues" should be given critical importance and hence our research has significant impact are listed below:

1. *Feedback analysis of products/services*: Apart from the basic features, products can be effectively analyzed based on several trending issues. For example, Samsung was considered as a leader in the smart phone business. But, the "Samsung's Galaxy Note 7 Battery Exploding Disaster" in 2016 completely downgraded the market value of Samsung. The fact can be evidently proved by analyzing trending tweets related to #TooEmbarrassed, #SamsungDisaster etc.
2. *Trend Analysis*: Trends vary with time. But, the current trends can be accurately predicted by analyzing the trending issues. For example, the terrorist attack that took place in Paris this year has terribly affected their tourism. People expressed their concerns through tweets with hashtags, #parisattack, #prayforparis, #porteouverte etc.
3. *Popularity Prediction*: Political leaders/celebrities' popularity depends on their day to day activities. Hence, it can show changes with respect to different trending issues. For example, the brand value of Brand Pitt declined after the divorce with Angelina Jolie which is evident from the hashtags #Brangelina, #Brexpitt etc.
4. *Demand Prediction*: Demands to different products depends on different other attributes like climate, financial status, diseases etc. For example, increase in the #HeavyRain hashtag makes cab service providers to increase their booking price.
5. *Product/Service Recommendation*: Different people need different things at different time and location. Hence, the exact demand is actually time dependent. For example, the Zika virus outbreak in India recently increased the demand for mosquito coils and repellants as mosquitoes are the major carries of these viruses.

3. Literature Review

Sentiment Analysis is a part of Natural Language Processing, which identifies the opinion, mood, plans, interests etc of people by analyzing the available text. Sentiment Analysis on social media like Facebook and micro blogging sites like Twitter, are becoming more trending, as it explores, every possible subjective information about a person or a product. Customer behavior identification [5] can be accurately achieved through sentiment analysis.

The application of Twitter analysis in election result prediction is being studied by Depanshu 2015, Kim and Hovy [12], Metaxas *et al.* [15], Pan and Lee [19] etc., Existing literature explores capability of tweet analysis for forecasting election results in countries like USA by Liu and Hu [14], Dutch by Agrawal *et al.* [1], Pakistan by Gimpel *et al.* [9], Korea by Song *et al.* [22]), Germany by Gayo-Avello [8], Arabic [2] and Singapore by O'Connor *et al*. [17]).

As per our knowledge, ours is the first attempt to extend the capability of tweet analysis to predict election winner in an Indian Context. Compared to other countries, ours is the largest democratic country with multiple parties participating in the election. Existence of multiple regional parties is another hurdle. Evaluating the entire economic characteristics will not be successful in identifying such diverse requirements. Also, election is such a long process in India and politicians try to campaign based on almost all the major issues occurring during that time span. Hence, we try to prove that this paper not only uses a new dataset from Indian context, but also explores the predictive power of twitter in a relatively complex and diverse political setting with respect to prior studies.

Sentiment analysis is considered as a subset of Natural Language Processing but at different levels of granularity. At the higher level, a document classification by Pang and Lee [20] can be performed, then to sentence level (Gimple *et al.* [9]) and more specifically at the phrase level (Agarwal *et al.* [1]). Twitter analysis is a form of sentiment analysis, because tweets are mere reflection of a common man.

There exists several works in literature to implement Twitter Analysis, making use of the existing models like Fuzzy [6], Naïve Bayes, Decision Trees, Support Vector Machines etc. To implement the feature space, existing methods like unigram, bigram and ngram methods are experimented but unigram [10] is found to be most efficient.

Compared to the existing methods [27] already experimented in literature, we used Hadoop Hadoop Distributed File System (HDFS) to store large scale twitter data and analyze the resulting data using Map Reduce and Hive, which supports Structured Query Language (SQL) like query to do the processing. Apart from analyzing positive and negative sentiment, we tried to be more specific by introducing different sentiments like fear, joy, anger and sadness and how each candidate is associated with it. The paper identified each different scams and issues that remained trending during the election period, and identified the sentiment of common man in each of these issues.

4. Research Design

In this article, we utilized a keyword based approach to classify the tweets into different sentiment domains. As a first step, we predefined some sentiment domains like anger, joy, fear, sadness and surprise (represented in oval shape). Corresponding to each domain we identified some keywords, short forms (generally used in micro blogging sites) and emoticons as well. The subset of sample keywords is given inside flower brackets in figure. This domain thesaurus is built based on the Tweet Natural Language Processing (NLP) sentiment dataset provided by Carnegie Mellon University [26]. Similarly, we build the domain thesaurus for each issue identified as in Figure1.

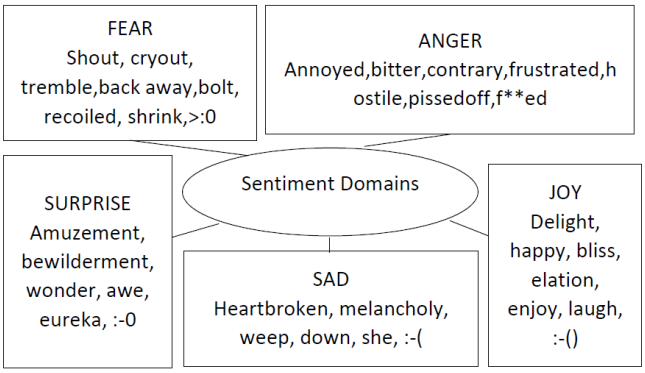

Figure 1. Sample Sentiment domain thesaurus for Tweet Classification

4.1. Extract Tweet Lexicons and Compute Sentiment Domain Similarity

Extract all lexicons from a single tweet. Compare it with the keywords associated with each sentiment domain separately [16]. The simple and direct approach is to use the Jaccard's coefficient as a similarity metric. Find the sentiment domain that has largest number of candidates matching with tweet lexicons.

Let T denotes a tweet and SD denotes the sentiment domain of T. Let L_t denotes the set of lexicons present inside a tweet and L_{si} denotes the lexicons corresponding to a particular sentiment s_i.

$$SD(T) = max \{ |L_t \cap L_{si}| / |L_t \cup L_{si}| \} \text{ where } i \text{ ranges from 1 to n.} \quad (1)$$

This method is fairly simple and direct, but does not consider negative words at all. Also, predicting the sentiment from a single tweet may not be accurate in identifying the overall support for a party. Hence, to obtain a better accurate similarity measure, we experimented with a method to collectively analyze tweets from a particular user by combining vector space model and cosine similarity measures.

In our keyword based similarity finding approach, the lexicon set from the tweets and lexicons belonging to each sentiment will be represented using an *mxn* matrix where each cell is filled with the weight associated with it. The weight corresponds to the frequency of occurrence of a lexicon in a tweet and m is the number of lexicons in a tweet and n is the total number of distinct lexicons considered by the entire domain thesaurus the overall process is illustrated in Figure 2.

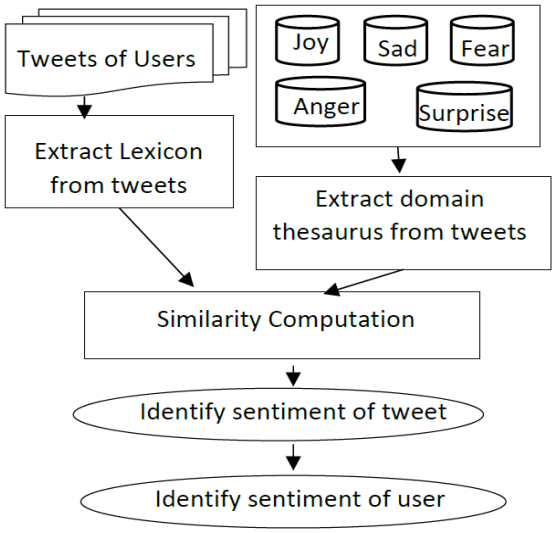

Figure 2. Overall Data Flow in tweet sentiment Classification.

Let W_T denotes the weight vector corresponding to a tweet, T. Therefore, $W_T = [w_1, w_2, w_3, \ldots . w_m]$ where w_i corresponds to the weight or the frequency of occurrence of a lexicon in a user's entire tweet collection and m is the total number of lexicons in T. To improve the comparison accuracy the weights can be updated using Analytic Hierarchy Process (AHP) proposed by Tumasjan [24]. AHP computes weights by analyzing the relative importance between two adjacent keywords; i.e., a pair-wise similarity comparison approach is followed.

Let P_T be the pair-wise similarity matrix generated from tweet L_T. Here $P_T = (p_{ij})_m$ where p_{ij} represents

pair-wise importance. P_{ij} is obtained using the following rule:

1. $p_{ij} = 1$ $\quad i=j=1,2,3,\ldots m$
2. $p_{ij} = 1/p_{ji}$ $\quad i,j=1,2,3,\ldots m \ \& \ i \neq j$
3. $p_{ij} = p_{ik}/p_{jk}$ $\quad i,j,k=1,2,3,\ldots m \ \& \ i \neq j$

After updating all cells in the matrix, we compute the weight by the following function:

$$w_i = 1/m \sum_{j=1}^{m} \{ p_{ij} / \sum_{k=1}^{m} p_{kj} \} \quad \ldots\ldots \tag{2}$$

where, p_{ij} is the relative importance between lexicons i and j.

A user's previous sentiment keywords can be identified by applying Term Frequency/Inverse document Frequency (TF/IDF) measure. Term Frequency/Inverse Document Frequency measure is widely used in Information Retrieval applications to weigh the importance of a term to the entire documents in the collection.

Before applying TF-IDF, collect all tweets from a particular user, u. The tweets are transformed into vector space based on the sentiment thesaurus. Here, Term Frequency will be the frequency of occurrence of each keyword, k in all his previous tweets.

$$TF = N_k / \sum_n N_k \tag{3}$$

Inverse Document Frequency is obtained by dividing total tweets, T by tweets containing the keyword, k.

$$IDF = log (|T| / |t : k \, \varepsilon \, t|) \tag{4}$$

Therefore, the weight of each keyword, k is obtained by:

$$W_k = N_k / \sum_n N_k \ X \ log (|T| / |t : k \, \varepsilon \, t|) \tag{5}$$

To find the similarity between the user's current tweet T, to his previous collection of tweets PT, cosine based sentiment similarity metrics is utilized.

$$sim(T,PT_i) = cos(W_T, W_{PTi}) = W_T \, . \, W_{PTi} \, / \, ||W_T|| \ X \ ||W_{PTi}|| \tag{6}$$

where W represents the weighted vector space model that we developed by applying Equation (2).

Only tweets which satisfy a minimum threshold, δ will be considered as similar ones. Others will be discarded. Now, we obtained all the tweets of a user which described his view on a particular issue. Next, we need to identify his sentiment/opinion on that issue. For that, we follow a weighted average approach.

$$p_s = r^* + k \sum_{PKj \sum Q} SIM(P_K , T_K). (r_j - r^*) \tag{7}$$

$$k = 1 / \sum_{PKj \sum Q} SIM(P_K , T_K) \tag{8}$$

Here, k is a normalizing factor which is the inverse of the sum of all similarities. Q is the set of all qualified keywords from Equation (3). 'r' is the rank of the keyword for a user and r^* is the average of all ranks.

Similarly, for each user, we identify his sentiment domain by applying the above mentioned procedure. The same approach is adopted to find whether a tweet reflects the positive or negative sentiment of a person.

We extracted trending issues by counting the frequency of occurrence of hashtags. Corresponding to each such topic, we identified the sentiment of people to know how much they supported different parties. The above sentiment matching method is summarized in Algorithm 1.

Algorithm 1 illustrates the basic working of our keyword based sentiment matching procedure. The inputs needed are the lexicons extracted from tweets T_k , the sentiment domains identified SD, a threshold δ for filtering less matching tweets, and a constant k which acts as the normalization factor. Output will be the highest probable set of sentiment domains to which a user belongs to. Q stores the qualified keywords of all tweets from a user and 'count' is used to obtain the cardinality of the set Q. 'r' is the ranking parameter which gets updated as algorithm iterates. Lines 3 to 8 analyzes the previous tweets from user and extract some matching qualified sentiment keywords which is stored inside Q. Line 9 to 18 finds the similarity measure and filters some irrelevant keywords which has score less than a threshold, δ. Also, it finds the sentiment score for each user based on Equation (4). Line 19 and 20 sorts the result based on the score and returns the highest matching sentiment of a user. The similarity of a person's tweets to his preferences obtained from his collection of tweets can be calculated by Algorithm 2.

Algorithm 1: Keyword Similarity Computation Algorithm for Sentiment Classification

Input: Lexicons extracted from tweets, T_k
Sentiment Domains SD ={sd_1, sd_2, …, sd_N} , The threshold δ in the filtering phase, The number K
Output: Identified sentiments with the Top-K highest similarity {tsd_1,tsd_2, …, tsd_K}

```
1: for each sentiment sd_i Ɛ SD
    2: Q=ϕ, count=0, r=0
    3: for each feature word R_j of sentiment sd_i
        4: process the tweet into a probable keyword set PK_j
        5: if PK_j ∩ T_K ≠ ϕ then
            6: insert PK_j into Q
        7: end if
    8: end for
    9: for each keyword set PK_j ∑ Q
        10: similarity_score = SIM(T_K, PK_j )
        11: if similarity_score ≤ δ then
            12: remove PK_j from Q
            13: else count = count +1, r=r+r_j
        14: end if
    15: end for
    16: r* = r / count
    17: p_s = r* + k ∑ PKj ∑Q SIM(P_K , T_K). (r_j – r*)
18: end for
19: sort the sentiment domains according to the personalized sentiment matching, p_s
```

20: return the sentiment domains with the Top-K highest matching {tsd_1,tsd_2, …, tsd_K}

Thus the over all design for predicting the sentiment of each tweet in Delhi Elections can be summarized by Algorithm 3.

4.2. Building A Data Set and Pre-Processing

After integrating with twitter, you need to specify keywords-like '@AamAdmiParty' ,'#Kejriwal', 'Kejriwal', #Vote4MufflerMan, #PressJhaaduButton, '#KiranBedi', 'KiranBedi', '@BJPDelhiState','#DelhiDecides',"#Delhi Votes, 'DelhiVotes' for which you want the tweets. We collected 650932 tweets starting 1st November, 2014 to 5th February, 2015 (two days before the election date) for both the parties. Then, remove duplicate tweets as some are retweeted.

Before feeding the data to the sentiment analysis, a preprocessing/data cleaning stage is necessary. Since tweets are limited to 140 characters, there will be a lot of noise in the form of short forms, emoticons, hashtags etc. To remove these noises, we adopted the syntactic normalization of tweets, proposed by Turney [25].

5. Implementation Details

Twitter exposes the Twitter API that enables developers to collect the Tweets. The service is made free, but requires the user to register and get authorized for the service. Download real time live data of twitter to Hadoop Distributed File System directly, by using Apache Flume tool. Instead of downloading entire tweets, preferences can be given in the form of keywords.

Algorithm 2: Similarity Computation

Input: The preference keyword set extracted from current tweet, T_k

The preference keyword set extracted from previous collection of tweets P_K

Output: The similarity of T_k and P_K

1: for each keyword ki in the sentiment domain thesaurus
2: if ki Ɛ T_k then
3: get W_{Ti} by formula (1)
4: else set $W_{Ti} = 0$
5: end if
6: if ki Ɛ P_K then
7: get W_{Pi} by formula (2)
8: else set $W_{Pi} = 0$
9: end if
10: end for
11: get similarity_score by formula (6)
12: return similarity_score

Algorithm 3: Delhi Election Prediction

1. *Download live Twitter data with keywords matching aap, bjp, modi, kejriwal, etc.*
2. *Store the twitter data to hadoop hdfs.*
3. *Extract needed information like tweets, retweeted, favorites etc from downloaded data and store it into a separate table using Hive.*
4. *Data preprocessing is done according to algorithm 4 to normalize the tweets and develop it into Standard English language.*
5. *Apply Similarity Algorithm 1 to identify the sentiment domain of a user.*
6. *Predict the winner based on maximum user support.*

The TwitterAgent.sources.Twitter.keywords are updated with necessary keywords like like '@AamAdmiParty' ,'#Kejriwal', 'Kejriwal', #Vote4MufflerMan, #PressJhaaduButton, '#KiranBedi', 'KiranBedi', '@BJPDelhiState','#DelhiDecides',"#Delhi Votes, 'DelhiVotes'. The resultant matching tweets are downloaded and stored inside Hadoop Distributed file system. From HDFS, the tweets can be directly processed using a map reduce program. But this will consume lot of time as the tweet data set contains a lot of unwanted data. To ease the process, in our implementation we extracted the tweet text and tweet id and created a table and stored using Hive warehousing tool. The advantage is that, Hive allows SQL like queries, which reduces the coding cost a lot. To find the similarity measure, we extracted data from hive table and used map reduce programming model to classify the tweets into different sentiment domain. Figure 3 demonstrates this.

We tested the application in Amazon Web Service. A Hadoop 2.6.2 cluster of 10 nodes are formed with one namenode, one secondary namenode and eight datanodes. We selected General purpose t2.large instance for namenode and t2.micro for secondary namenode and datanodes.

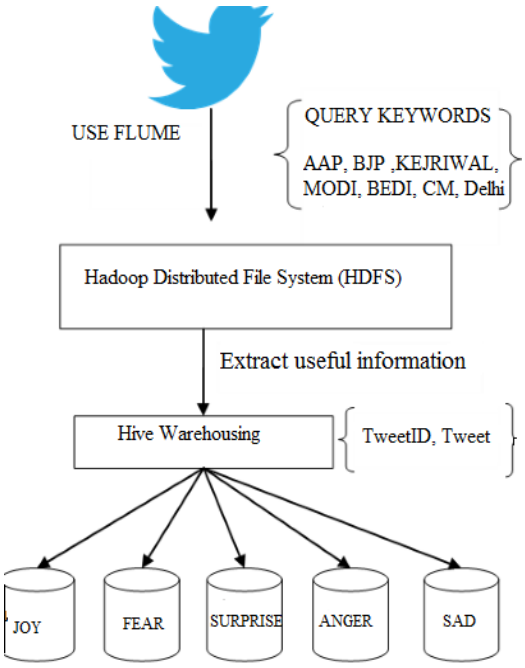

Figure 3. Information flow in the classification of tweets.

5.1. Map Reduce Implementation to Find the Keyword Similarity Measure

Tweet Analysis has two stages. The cumulative tweets from a user are analyzed offline. Then, the exact similarity of newly arriving tweets to user's previous interests are calculated in the Online Analysis stage.

- MapReduce Stage 1 (Offline): Mapper takes the collective tweets categorized by different users as input and outputs the probable keywords which match his interests. Reducer collects the probable keywords and calculates an average preference weight for each user. Map input < i, j, r_{ij}, Q_{ij} > on *i* such that the records with the same *i* are shuffled to the same node in the form of <j, rij, Q_{ij}>. $Reduce_i$ takes <j, rij, Q_{ij}> as the input and emit <i, j, r_{ij}, Pk_{ij} , $r_i^{\wedge}$> for each input of Map_i. The output of this Reduce stage will be used as the input for Stage 2 Map to calculate the similarity.
- MapReduce Stage 2 (Online): After getting each user's sentiment about different issues, Mapper will compare each new tweet to his previous interests. Reducer will return the similarity score of each user to different sentiment domains. Map <i, j, r_{ij}, Pk_{ij} , $r_i^{\wedge}$> on *i*, and tuples with the same *i* are shuffled to the same node in the form of < j, r_{ij}, Pk_{ij} , $r_i^{\wedge}$>. Apart from this input, Reducer takes <T_K> as the input, and calculate the similarity_score, <i, j, sim_score>. This stage makes use of TF-IDF metric to quantify the sentiment of each user.
- Map Reduce Stage 3 (Online): The sentiments of each user are ranked according to the similarity score calculated. The details of input and outputs to different map reduce stages are depicted in Figure 4.

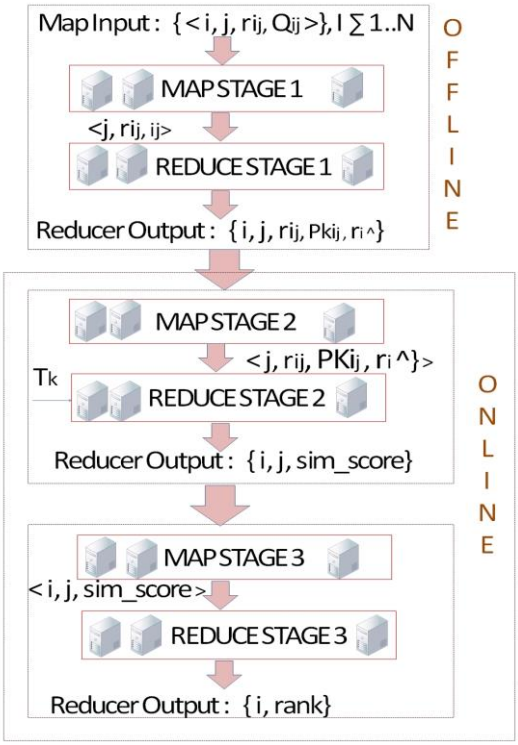

Figure 4. MapReduce implementation stages.

6. Results and Discussion

Some of the important results obtained after performing a lot of analysis on twitter data are given below.

6.1. Accuracy and Efficiency Comparison

We compared our method with two existing methods, Jaccard's Similairty Metric (JSM), Dirichlet-multinomial regression (DMR) [18] and Pearson Correlation Coefficient (PCC) to categorize the sentiments. The metrics chosen to evaluate the accuracy are Mean Absolute Error (MAE) [11], Mean Average Precision (MAP) [21] and Discounted Cumulative Gain (DCG) [13]. To analyze scalability, speedup is used as the metric [3]. We denote our method using the terminology, Domain Similarity Matching (DSM) which indicates domain based similarity matching.

6.1.1. Comparison of JSM, DMR, PCC and DSM in MAE

MAE is a statistical measure used to analyze prediction accuracy. Normalized Mean Absolute Error (NMAE) indicates the normalized value of MAE. Lower values of MAE or NMAE indicates higher accuracy of similarity matching. Figure 5 illustrates the MAE/NMAE values for Jaccard's Similairty Metric (JSM), Dirichlet-multinomial regression (DMR), Pearson Correlation Coefficient (PCC) and DSM respectively. The graph shows a minimum of 25% decrease in MAE and 36% decrease in NMAE for our approach, compared to the existing approaches.

6.1.2. Comparison of JSM, DMR, PCC and DSM in MAP

The objective of our algorithm to is to predict the most matching sentiment of a user. Our algorithm ranks the sentiments based on a similarity score and return top k sentiments. From that we chose the sentiment with the highest rank. MAP is a metric to measure the quality of ranking function. Higher MAP value indicates higher precision in ranking. We analyzed MAP for k = 2, 3 and 5. The MAP value obtained for different methods is plotted in Figure 6. Our method clearly shows a minimum increase of 17% compared to others.

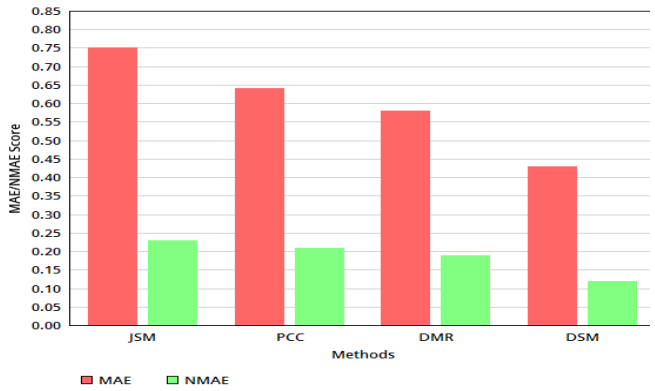

Figure 5. MAE/NMAE Comparison.

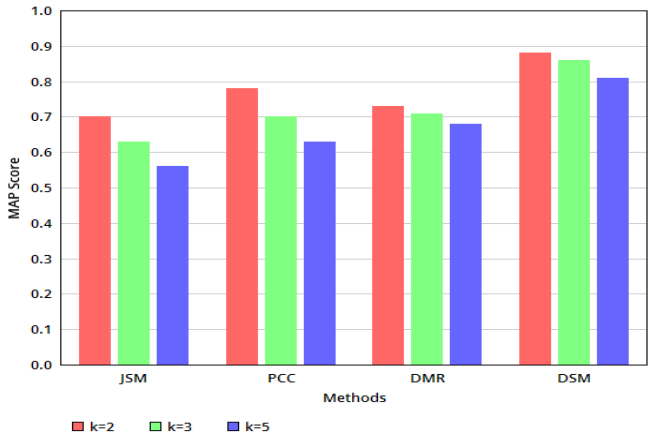

Figure 6. Comparison on MAP.

6.1.3. Comparison of JSM, DMR, PCC and DSM in DCG

DCG is another metric to evaluate the accuracy of ranking functions in information retrieval. The DCG value is directly proportional to the accuracy of ranking. We analyzed DCG value for various values of 'k' and observed a minimum of 4.8% increase in accuracy for our method (Figure 7). Another important observation made from the analysis is that, our method performs well for small values of 'k', and that is what we require for these kinds of prediction applications.

6.1.4. Speedup Comparison

Speedup is a well known metric to evaluate scalability. Speedup (S_u) is defined by the ratio of time required to execute an algorithm sequentially (T_1) to the time required to execute it in 'n' systems (T_n).

$$S_u = T_1 / T_n \quad (9)$$

A system is said to have good scalability, if the value of S_u remains linear for different values of n. We analyzed speedup by running the map reduce code for different input sizes and obtained the values as shown in Figure 8. From the graph it is clear that our method has increased scalability for large input dataset.

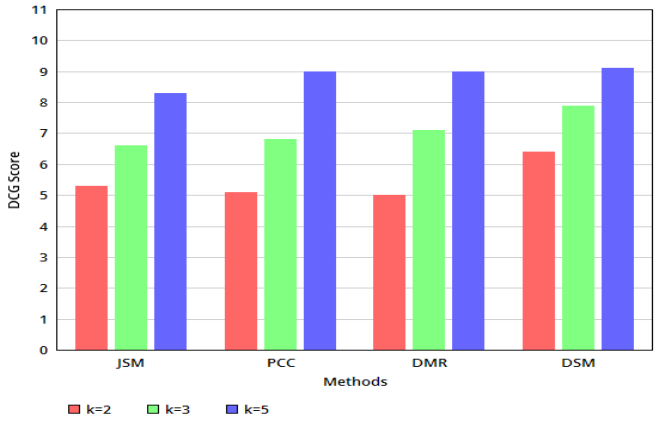

Figure 7. Comparison on DCG.

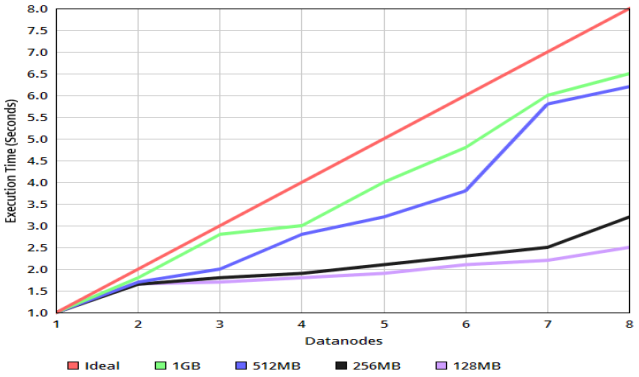

Figure 8. SpeedUp comparison.

6.2. Results Obtained from Twitter Data

The word "is baaraap" occurred more number of times compared to "modisarkar", which shows the support for AAP candidates.

6.2.1. Word Cloud

Wordcloud comparing the frequencies of words in tweets corresponding to BJP and AAP as obtained from Hive are given in Table 1.

Table 1. WordCloud corresponding to AAP and BJP.

AAP	BJP
Isbaraap	Aapfundingsscam
5saalkejriwal	Abkibarbedisarkar
Hawaalaatmidnight	Opportunist
Evm	Kharvaapsi

6.2.2. Sentiment Analysis of Tweets by Emotional Categories

We analyzed the tweets to get an overall view on people's sentiment on different parties. We considered different sentiment categories like fear, joy, sadness, surprise and anger (Figures 9 and 10). Again, to analyze the sentiment category, we took a keyword-based approach.

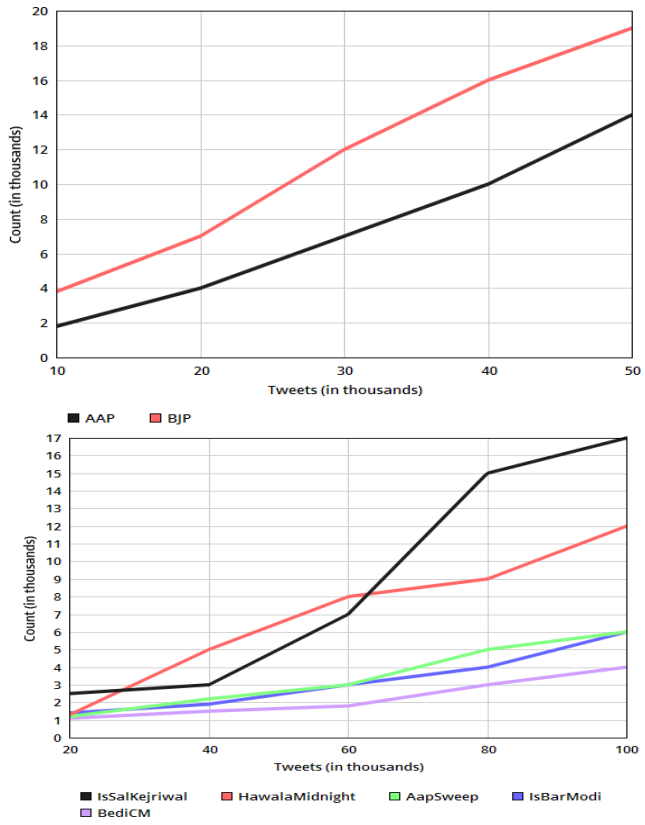

Figure 9. Frequency curve of AAP vs BJP.

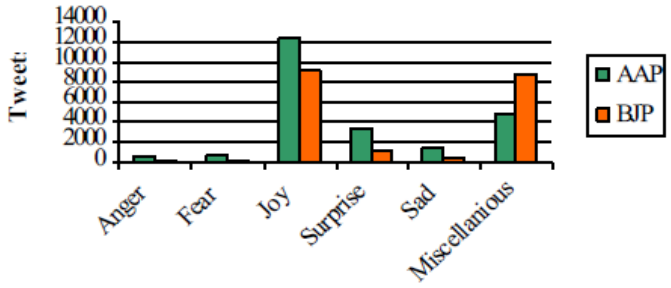

Figure 10. Classification of tweets into different sentiment domain.

Here we categorized tweets into different segments like positive and negative. For example, "I support AAP" is a positive tweet, where as "I do not support AAP" makes the tweet negative. This implies that the party having higher positive tweets has more support from people Illustrated in Figure 11.

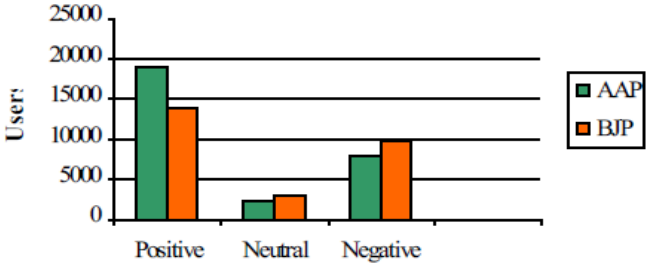

Figure 11. Positive/Negative Sentiment Analysis of Tweets - AAP has greater positive sentiments than BJP.

Similarly we analyzed the sentiment of people corresponding to different trending hash tags like #AAPSweep,#AAPKiDilli,#AAPFundingScam, #HawalaAtMidnight, #SaalEkScamAnek, #KharVapsi etc., To identify positive and negative tweets we followed the same procedure that we mentioned in section IV. Around 479 hash tags were actually found. Out of which the most trending 100 hash tags were selected for analysis. We identified some lexicons, short forms and emoticons corresponding to each sentiment domain and using cosine based similarity measure, identified number of distinct users in support and against different parties Result is given in Figure 12.

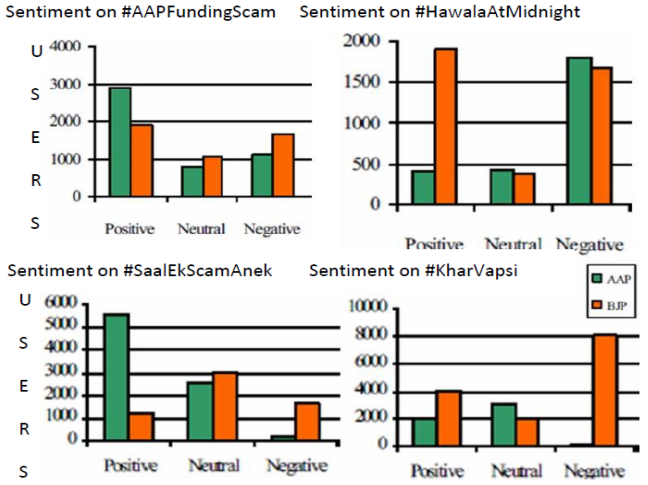

Figure 12. Sentiments of users on different trending issues.

6.2.3. Comparison with Existing Methods

The table below gives an overview of the capabilities of our system compared to existing well known approaches like predictive analytics and Exit polls. Most of the exit polls predicted 20-27 seats for BJP but actually they got only 3 seats. Table 2 compares existing election prediction strategies with the proposed approach.

Table 2. Comparison of election prediction techniques.

Method	Advantages	Drawback
1. Predictive Analytics	-Well established method -Easily applicable to statistical data	-Do not take into account the real-time issues -Sentiment is not taken into consideration -Not suitable to analyze emerging parties or issues
2. Bayesian Analysis	-Good for predicting the winner of each small area	-Do not take into account the real time issues -Sentiments are not considered.
3. Exit polls	-Good for finding real time support of candidates	-Very few people participate in exit polls -poor choice of the samples will lead to error
4. Sentiment Analysis from Social Media	-Accurately identify the real time support -Consider the sentiment of people -Accurate percentage of support can be identified.	-Hardware to analyze huge amount of data

7. Conclusions and Future Scope

The paper was focused on predicting the person who has a higher chance in winning the Delhi Legislative Elections conducted in India in February, 2015. We used twitter data to do the sentiment analysis. Twitter

Data was downloaded using Apache Flume. Sentiment analysis was done using Map Reduce Programming model so that scalability is ensured. Tweets are categorized into different sentiment domain by comparing the lexicons in tweet, and lexicons associated with each sentiment domain. Domain thesaurus is carefully chosen to match micro blogging message patterns. Accurate sentiment domain is identified based on keyword based collective tweet analysis. Apache Hive is used as the data warehousing tool, since it allows simple sql like queries to extract useful information from large amount of twitter data. From the results, it was clear that, the newly formed party named 'Aam Admy Party (AAP)' had higher support among people compared to the well established party like "Bharathiya Janatha Party (BJP)". The same method can be adopted for different applications by changing the domain thesaurus. Product marketing based on customer feedback, spam detection, deal recommendations, collecting people's feedback on political reforms, potential customer identification in agricultural sector etc to list a few.

Acknowledgement

Authors sincerely thank Department of Science and Technology (DST), India for the financial support offered through INSPIRE Research Fellowship under the grant number IF140608 and Amazon Web Service (AWS) in Education Research Grant (No. 651699140108) for utilising AWS resources for free of cost.

References

[1] Agrawal A., Biadsy F., and Mckeown K., "Contextual Phrase-Level Polarity Analysis using Lexical Affect Scoring and Syntactic N-Grams," *in Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics*, Athens, pp. 24-32, 2009.

[2] Al-Kabi M., Al-Ayyoub M., Alsmadi I., and Wahsheh H., "A Prototype for a Standard Arabic Sentiment Analysis Corpus," *The International Arab Journal of Information Technology*, vol. 13, no. 1A, pp. 163-170, 2016.

[3] Amdahl G., "Validity of the Single-Processor Approach to Achieving Large Scale Computing Capabilities," *in Proceedings of Spring Joint Computer Conference*, Atlantic, pp. 483-485, 1967.

[4] Beyer M. and Laney D., "The Importance of 'Big Data': A Definition," Technical Report, Gartner, 2012.

[5] Kang G., Liu J., Tang M., and Liu X., "AWSR: Active Web Service Recommendation Based on Usage History," *in Proceedings of the 19th IEEE International Conference on Web Services*, Honolulu, pp. 186-193, 2012.

[6] Chu A., Kalaba R., and Spingarn K., "A Comparison of Two Methods for Determining the Weights of Belonging to Fuzzy Sets," *Journal of Optimization Theory and Applications*, vol. 27, no. 4, pp. 531-538, 1979.

[7] Gayo-Avello D., "A Meta-Analysis of State-of-the Art Electoral Prediction from Twitter Data," *Social Science Computer Review*, vol. 31, no. 6, pp. 649-679, 2013.

[8] GayoAvello D., "Don't Turn Social Media into another 'LiteraryDigest' Poll," *Communications of the ACM*, vol. 54, no. 10, pp. 121-128, 2011.

[9] Gimpel K., Schneider N., O'Connor B., Das D., Mills D., Eisenstein J., Heilman M., Yogatama D., Flanigan J., and Smith N., "Part-of-Speech Tagging for Twitter: Annotation, Features, and Experiments," *in Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, Portland, pp. 42-47, 2011.

[10] Go A., Bhayani R., and Huang L., "Twitter Sentiment Classification using Distant Supervision," Technical Report, Stanford University, 2009.

[11] Kaufmann M., "Syntactic Normalization of Twitter Messages," *in Proceedings of the International Conference on Natural Language Processing*, Kharagpur, 2010.

[12] Kim S. and Hovy E., "Determining the Sentiment of Opinions," *in Proceedings of the 20th International Conference on Computational Linguistics*, Geneva, 2004.

[13] Lakiotaki K., Matsatsinis N., and Tsoukis A., "Multi-Criteria User Modeling in Recommender Systems," *IEEE Intelligent Systems*, vol. 26, no. 2, pp. 64-76, 2011.

[14] Liu B. and Hu M., "Mining and Summarizing Customer Reviews," *in Proceedings of the 10th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, Washington, pp. 168-177, 2004.

[15] Metaxas P., Mustafaraj E., and Gayo-Avello D., "How (not) to Predict Elections, In Privacy, Security, Risk and Trust (PASSAT)," *in Proceedings of IEEE 3rd International Conference on Social Computing (SocialCom)*, Boston, pp. 165-171, 2011.

[16] Mohan L. and Elayidom S., "Who Will Win Delhi Election-Prediction by HIVE," *in Proceedings of National Conference on Adaptive Techniques in Engineering and Technology*, India, 2015.

[17] O'Connor B., Balasubramanyan R., Routledge B., and Smith N., "From Tweets to Polls: Linking Text Sentiment to Public Opinion Time Series," *in Proceedings of the 4th International*

AAAI Conference on Weblogs and Social Media, Washington, pp. 122- 129, 2010.

[18] Pak A. and Paroubek P., "Twitter as a Corpus for Sentiment Analysis and Opinion Mining," *in Proceedings of the International Conference on Language Resources and Evaluation*, Valletta, pp. 17-23, 2010.

[19] Pan Y. and Lee L., "Performance Analysis for Lattice-Based Speech Indexing Approaches using Words and Subword Units," *IEEE Transactions on Audio, Speech, and Language Processing*, vol. 18, no. 6, pp. 1562-1574, 2010.

[20] Pang B. and Lee L., "A Sentimental Education: Sentiment Analysis using Subjectivity Analysis using Subjectivity Summarization Based on Minimum Cuts," *in Proceedings of the 42nd Annual Meeting on Association for Computational Linguistics*, Barcelona, pp. 271-278, 2004.

[21] Saaty T., "Decision Making with the Analytic Hierarchy Process," *International Journal of Services Sciences*, vol. 1, no. 1, pp. 83-98, 2008.

[22] Song M., Kim M., and Jeong Y., "Analyzing the Political Landscape of 2012 Korean Presidential Election in Twitter," *IEEE Intelligent Systems, Special Issue on Social Intelligence and Technology*, vol. 29, no. 2, pp. 18-26, 2014.

[23] Taboada M., Brooke J., Tifiloski M., Voll K., and Stede M., "Lexicon-Based Methods for Sentiment Analysis," *International Journal of Computer Knowledge and Technology*, vol. 11, pp. 230-239, 2011.

[24] Tumasjan A., Sprenger T., Sandner P., and Welpe I., "Predicting Elections with Twitter: What 140 Characters Reveal about Political Sentiment," *in Proceedings of the 4th International AAAI Conference on Weblogs and Social Media*, Washington, pp. 178-185, 2010.

[25] Turney P., "Thumbs Up or Thumbs Down? Semantic Orientation Applied to Unsupervised Classification of Reviews," *in Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*, Philadelphia, pp. 417-424, 2002.

[26] TweetNLP Dataset provided by Carnegie Mellon University: http://www.ark.cs.cmu.edu/TweetNLP/, Last Visited, 2017.

[27] USQL Sentiment Analysis tool by Microsoft Azure https://msdn.microsoft.com/en-us/azure/data-lake-analytics/u-sql/sentiment-analysis-u-sql, Last Visited, 2017.

Lija Mohan has completed her Ph.D. in Big Data Security from School of Engineering (SOE), Department of Cochin University of Science & Technology (CUSAT) and currently working as Cyber security Specialist at Prevalent AI Pvt Ltd. She took her Masters and Bachelor Degree in Computer Science, both from Mahatma Gandhi University. She has several International publications to her credit and she is the recipient of AWS Research Grant and Inspire Fellowship.

Sudheep Elayidom is working as Professor at the Computer Science Division of Cochin University of Science and Technology, Kerala, India. He received his Masters and Bachelors from Mahatma Gandhi University and Ph.D. from Cochin University of Science and Technology, all in the field of Computer Science. He has delivered keynote addresses, invited seminars, and served as session chair for international conferences and workshops. Also, he has authored several journals, books and conference articles.