SSRG International Journal of Computer Science and Engineering
ISSN: 2348–8387 / https://doi.org/10.14445/23488387/IJCSE-V10I6P104

Volume 10 Issue 6, 22-35, June 2023

Original Article

Conducting Sentiment Analysis on Twitter Tweets to Predict the Outcomes of the Upcoming Karnataka State Elections

Prajwal Madhusudhana Reddy

Greenwood High International School, Karnataka, India

Received: 25 April 2023 Revised: 01 June 2023 Accepted: 12 June 2023 Published: 30 June 2023

Abstract *- This research paper aims to predict how Twitter tweets from a specific politician correlate to their winning a seat in a state election. To understand this effect, sentiment analysis has been conducted on tweets by politicians in Karnataka to help predict who will win in the upcoming 2023 Karnataka Legislative Assembly election. Though previous research has already been done in this area, most studies have only focussed on the sentiment analysis of tweets. This paper goes further as it also looks at other factors, including the number of retweets and comments a tweet garners, which measures the tweet's engagement. A model has been created that weighs each factor to help predict who will win an election for a particular constituency. Through this model, a 72.7% accuracy has been achieved. However, the Twitter API severely limited the quantity and quality of data collected. These results can be expanded to help predict elections for other states. They could potentially help understand the effect of positive and negative sentiment on the winnability of a political candidate.*

Keywords *- Election, Karnataka, Sentiment analysis, Twint, Twitter.*

1. Introduction

Often, politicians use various forums to let their voters and the general public know what they are currently doing, to express their thoughts and opinions, advance their political beliefs and party, recent news and developments, major achievements for the politician, and recent political activity.[8] This takes the form of various mediums, including newspaper campaigns, speeches, and rallies. However, with the rise of the internet, another medium has come up: social media sites, such as Twitter.

Twitter is a good source of public opinions to analyze as anyone can share their thoughts there since many politicians utilize the platform to discuss their party. Their message spreads through the politician's followers throughout the internet, which spreads the message further.[11] Moreover, there is a great depth of information as Twitter makes it very easy to share one's thoughts for free without restriction on who can post. There is now a large volume of publicly available data which can be utilized for research analysis. For example, approximately 12.5% of all Indian politicians in the 2014 Indian general election had an account[11], representing a significant number of the total politicians.

Sentiment analysis aims, in short, to answer the question of *'What do people feel about a certain topic?'* It combines computer science and linguistics to help systematically analyze human behavior through the analysis of human language. Making use of extensive data collection, it is a part of the field of Natural Language Processing.[11] Sentiment analysis, therefore, can be conducted on tweets from Twitter to measure public opinion of the data, engagement, and the likelihood that the public supports one politician over another.

Though ample research has been done on Twitter data, the vast majority of those is in English. Many regional politicians use the local language when dealing with the specific state, so there is still a large amount of data that can be utilized that has not been used.

Therefore, using natural language processing and sentiment analysis models, the aim is to find out if there is a noticeable correlation between online discussions regarding local Kannadiga politicians and their eventual election results in the upcoming 2023 Karnataka Legislative Assembly elections. The data will be sourced from Twitter and include Kannada politicians' tweets.

To do this, the following sources of information will be examined: the sentiment polarity of the politician's tweets and the engagement resulting from the tweets. Hopefully, these factors will help ensure a thorough analysis of the link between public perception and election outcomes. A model can be created based on these factors.

The results of this analysis are significant since they will help predict election results based on online discussions, which will aid in public relations campaigns for not only political parties but also big brands and other organizations. If the model proves successful for the

Karnataka elections, it can be generalized to other states and perhaps to national elections.

2. Literature Review, Research Aim, and Hypothesis

2.1. Literature Review

Several studies have previously been conducted using sentiment analysis on Twitter tweets. This could be to test various new sentiment analysis models as Twitter has content on all types of topics, and many users use it to express their opinions as well. *Sentiment Analysis of Twitter Data* by *Apoorv Agarwal et al.*, for example, introduces 'POS-specific prior polarity features' as well as a 'tree kernel' in their analysis of Twitter data. Sentiment analysis could be used in other fields, such as market research, as companies can understand how consumers react to various products.[1]

However, significant previous work has also been done in the election prediction domain. For example, *Sentiment Analysis to Predict Election Results Using Python* by *Sayyada et al.* used sentiment analysis to create a word cloud to count the number of negative, neutral, and positive tweets for the 2016 US Presidential Election.[5] Similarly, *On Using Twitter to Monitor Political Sentiment and Predict Election Results* by *Smeaton et al.* focused on the Irish general elections[2], and *Sentiment Analysis to Predict Election Results Using Python* by *Sayyada et al.* looked at both the 2012 US elections and 2013 Karnataka elections[16]. However, there are two new aspects of election sentiment analysis used by this research. Firstly, it focuses solely on a local election in Karnataka to ensure focus. In doing so, it analyzes Kannada tweets as well. Secondly, it looks at various other factors, in addition to the sentiment analysis, including the engagement of the tweets. This provides more accuracy and a more holistic view of the context of the tweets.

2.2. Research Aim and Hypothesis

This research paper aims to ascertain whether conducting sentiment analysis on public tweets of certain politicians can help predict who will win a local election. This could help public relations campaigns for the different stakeholders involved and allow easier decision-making by increasing certainty in elections.

The prediction is that with a well-trained model with carefully selected parameters, the model will be able to accurately predict the results of an election with certainty, showing that there is indeed a correlation between positive sentiments expressed and high engagement with winnability.

3. Methodology

3.1. Tools

The code used to analyze the data was written in the Anaconda Python distribution, with Jupyter Notebook for the development environment. Five standard library modules were used to provide extra capabilities and include *os*, *json*, *re*, *string*, and *pathlib*.

Apart from this, three third-party libraries were used. Firstly, the *nltk* (Natural Language Processing Toolkit) library (https://www.nltk.org/) was used to provide the sentiment analysis model. This model is described further in the Algorithm and Code section. Next, *translate* (https://pypi.org/project/translate/) was used to provide translation services for the Kannada tweets. Lastly, *twint* was used to scrape the tweets from Twitter. This is described further in the Data Collection Section.

3.2. Data Collection

Before any data is analyzed, it first needs to be collected. For the purposes of this paper, the social media network Twitter was chosen as the data source. This was because of the large corpus of data available on Twitter, especially from politicians.

The tweet gathering is done using the popular Python module *twint*, which allows one to gather Twitter posts without needing an API key. This ensures that the data collection is seamless and is not restricted by internal Twitter rate limits.

Two sets of data were collected. The first one covered a total of 11 pairs of training politicians from the time range of May 12th, 2017, to May 5th, 2018, which corresponds to the period exactly one year before the previous 2018 election. The second one covered 5 pairs of test politicians from the time range of Mar 10th, 2023, to Apr 10th, 2023, corresponding to the period of about one month before the upcoming 2023 election. These Twitter tweets, both in English and Kannada, were scraped from several politicians from different constituencies in Karnataka and stored for analysis.

Moreover, the targeted Twitter accounts will include multiple politicians who competed for the same seat in parliament. A wide variety of politicians from different constituencies will be chosen to have as many data points as possible. For the sake of simplicity, only one opposition candidate has been considered for the models. The full list of considered candidates can be seen in Appendix 1.

A Python script has been written that reads a list of accounts from a *settings.json* file and scrapes the tweets from each account one by one. There is also an option to select a time range to scrape the tweets. Though optional, it can be used to differentiate between training and test tweets. 2,174 tweets were collected across all training politicians, and 100 tweets from across the test politicians.

A variety of different parameters are captured from the scraped tweets. This includes all the following:

Table 1. Parameters captured from the scraped tweets

Parameter	Description
ID	This is the tweet ID that can be used to identify a specific tweet uniquely.
Username	This is the Twitter username of the specific politician.
Name	This is the politician's actual name, which may or may not be different from his or her username.
Date	This is the date the tweet was published. This can be used to identify the context in which the tweet was published, e.g., before or after an election.
Tweet	This is the actual content of the tweet and is the primary text that will be analyzed. Sentiment analysis will be conducted on this content.
Language	As politicians from Karnataka usually write their tweets in either Kannada or English, the tweet's language is also stored.
Likes	This is the number of likes a post has garnered.
Replies	This is the number of replies a post has.
Retweets	This indicates the number of times a post has been retweeted.

All the above parameters are taken together as a whole to be input into the models proposed to be used. The parameters above were chosen as they were easy to collect using the Twint API. Some statistics on the training tweets are shown below. The maximum counts all came from Siddaramaiah.

Table 2. Some statistics from the scraped training tweets

Minimum Likes	0	Maximum Likes	17257
Minimum Replies	0	Maximum Replies	1829
Minimum Retweets	0	Maximum Retweets	7901

The code used to analyze the data is made up of two basic parts: data collection and data analysis, the former of which is already described above. The data analysis part is further split into two sections: training and testing. Whereas the training part of the model is to focus on accuracy using data from the previous 2018 elections, the testing part is to focus on predicting the upcoming 2023 elections.

To train the proposed models and test their accuracy, the model is tested with tweets from politicians who ran for the previous Karnataka election in 2018. Since the election result is known, the model's accuracy can be calculated, and any necessary adjustments to improve its accuracy can be made here. If the model's accuracy is at an acceptable level, then the model can be used to help predict the election results for the upcoming 2023 election.

The same model is used for the second part, the testing part. Here, tweets are collected for the upcoming 2023 election, and the same analysis is conducted on them. Since the winners are unknown, a prediction is given on who will win based on the factors described.

In both cases, the model primarily looks at the following factors: the tweet's content, the language, the number of likes, replies, and retweets. The last 3 factors measure the tweet's engagement and, by extension, popularity. For example, Siddaramaiah has the highest engagement scores, as seen in the above table, so it can be inferred that he is extremely popular.

From these, an average score is calculated for each politician based on an equal weightage of each factor. The politician with the higher average score is seen as more likely to win.

3.3. Algorithm and Code

The proposed model looks at various factors (the tweet's content, the language, the number of likes, replies, and retweets) and assigns each candidate an average score. The candidate with a higher score than their opponent is presumed to have a higher chance of winning.

Sentiment analysis is conducted on each tweet for each candidate, and the sentiment polarity is calculated. Before this can be done, however, the text must be translated. If the text is in Kannada, the *translate* module translates this to English so sentiment analysis can be done. This does not reduce the information conveyed as only the sentiment, and not the pure meaning of the text, needs to be conveyed.

After this, the text is sanitized. This process clears all the 'noise' from the text to create a list of words influencing the text's sentiment. This process transforms all the words in the text to lowercase letters, creates a list of all the remaining lemma forms or root forms of the word, and removes punctuation and stop words (words such as *and*, *is*, and *the* which do not add to the text's meaning). This preprocessing step is critical as it helps remove all words that do not affect the text's meaning and removes any inflectional variations in the text (e.g., *eat* and *ate* should be recognized as the same word) to ensure a consistent analysis.[5] The code is shown below:

```
def sanitize_text(text):
  text = text.lower()
  text = re.sub(r'\n', '', text)
```

```
  translator = str.maketrans('', '', string.punctuation)
  text = text.translate(translator)

  lemmatizer = WordNetLemmatizer()
  words = tokenize.word_tokenize(text)
  words = [lemmatizer.lemmatize(word) for word in
words]

  stop_words = stopwords.words("english")
  filtered_text = [word for word in words if not word in
stop_words]

  return " ".join(filtered_text)
```

After these two processes, the actual analysis occurs. The *SentimentIntensityAnalyzer* class from *nltk. sentiment* is used to return a sentiment score from 0 to 1, where higher scores indicate more positive sentiments. The code is shown below:

```
def sentiment_polarity(text):
  sia = SentimentIntensityAnalyzer()

  score = sia.polarity_scores(sanitize_text(text))
  key =
list(score.keys())[list(score.values()).index(max(list(score.v
alues())[:len(score) - 1]))]

  return score, key
```

Furthermore, the factors are also considered, which are taken directly from the tweets. This includes the number of likes, replies, and retweets for each tweet from each candidate. The reason these factors are weighed in is because this is a direct indicator of the popularity of the tweets and how much engagement the tweet has received. Moreover, previous research has heavily focused on the sentiments of a given tweet or collection of tweets rather than other factors that can also have a bearing on the result. This function is the most important, as seen below:

```
def calculate_score(json_tweet):
  text = json_tweet["Tweet"]

  if json_tweet["Language"] != "en":
     text = translate_text(json_tweet)

  score, key = sentiment_polarity(text)

  sent_pol = (score["pos"] - score["neg"]) if (score["pos"] -
score["neg"]) > 0 else 0  # Ensures The Sentiment Polarity
Is Positive

  likes_score = (json_tweet["Likes"] - minLikes) /
(maxLikes - minLikes)  # Scales The Likes Score To The
Range Of 0.0 to 1.0

  replies_score = (json_tweet["Replies"] - minReplies) /
(maxReplies - minReplies)  # Scales The Replies Score To
The Range Of 0.0 to 1.0
  retweets_score = (json_tweet["Retweets"] -
minRetweets) / (maxRetweets - minRetweets)  # Scales
The Retweets Score To The Range Of 0.0 to 1.0

  # The Final Total Score Is From 0.0 to 100.0
  total_score = 25 * (sent_pol + likes_score + replies_score
+ retweets_score)
  return total_score
```

Each factor is given an equal weightage, and a score is calculated from these parameters. This is done for each tweet for the candidate, ultimately providing a total average score. This is the final score for the candidate and can be compared to that of his or her opposition. The candidate with the higher score is seen as having a higher chance of winning a seat. The complete code for the sentiment analysis is given in Appendix B.

4. Results

For the purposes of the paper, 64 different politician pairs from the different constituencies of Karnataka were considered. Of these 64, 30 pairs were made up of politicians who both have Twitter accounts. The model used eventually scraped tweets from 15 different politicians and was able to get 11 viable pairs of politicians from the same constituency. The summary of results is given below, with the winning politician given first. All figures are to 2 decimal places. The full list of politicians considered can be found in Appendix A, attached below.

From these 11 pairs, 8 were predicted accurately, giving an accuracy of 72.7%. A few indicators of the accuracy of the model were also calculated to place this in context better:

The Average Net Margin Score is the average of all the losing candidates' scores subtracted from all the winning candidates' scores. When this score is high, it shows that the winning candidate's tweets had more engagement and positive sentiments expressed than the losing candidate's. If it is low or negative, then the losing candidate had more engagement and positive sentiments.

The Average Percentage Margin is the average predicted margin of victory based on the analysis scores. It is calculated as follows: (Predicted Score 1 - Predicted Score 2) * 100 / (Predicted Score 1 + Predicted Score 2). This can be used as a measure of accuracy for an individual candidate pair and is essentially a measure of how certain the model is that one politician will win over another, with higher scores indicating more certainty. It is important to note, though, that 100% - APM does not indicate the chances of the model being wrong; the APM just shows the certainty of the predictions. The summary of these indicators is given below.

Table 3. Results of the training tweets analysis

Winning Candidate	Score	Losing Candidate	Score	Correctly Predicted
Lalaji Mendon	19.69	Vinay Kumar Sorake	32.05	✗
GT Devegowda	14.97	Siddaramaiah	13.15	✓
HD Kumaraswamy	2.51	C. P. Yogeeshwara	0.72	✓
G. H. Thippareddy	1.59	K. C. Veerendra	0.75	✓
L. Nagendra	2.19	Vasu	2.02	✓
Dr. C.N. Ashwath Narayan	3.37	Kengal Shreepadha Renu	2.09	✓
Dinesh Gundu Rao	4.81	A. R. Sapthagiri Gowda	2.34	✓
Ramalinga Reddy	1.52	Lallesh Reddy	2.71	✗
Munirathna	1.72	P. Muniraju Gowda	1.48	✓
Suresha BS	3.19	Y. A. Narayanaswamy	5.23	✗
C. Puttarangashetty	1.17	K. R. Mallikarjunappa	0.96	✓

Table 4. Some indicators from the training tweet analysis

Correct Pairs	8
Total Pairs	11
Average Net Margin	-0.61
Average Percent Margin	9.21%
Accuracy	72.7%

The fact that the Average Net Margin is low, yet still fairly close to zero, means that most results were close absolutely, but there were a few results that were off by a large margin, namely that of Lalaji Mendon (19.69) and Vinay Kumar Sorake (32.05).

However, the Average Percent Margin shows that, on average, there was still a significant difference between the winning and losing candidates' scores.

4.1. Predictions

In order to fully test the accuracy and reliability of the model, predictions have been made on 5 candidate pairs for a total of ten candidates. These tweets were collected from Mar 10th, 2023, to Apr 10th, 2023, which is about a month before the 2023 elections and were analyzed using the data as it was at the time of collection. 10 tweets were collected per candidate for a total of 100 test tweets. These predictions can be tested against the results of the 2023 election. The predicted winning candidate is given first in the table below.

Table 5. The results of the test tweets analysis

Constituency	Candidate 1	Score	Candidate 2	Score	Confidence
Basavanagudi	Ravi Subramanya (BJP)	19.04	UB Venkatesh (INC)	2.88	73.76%
Channapatna	H. D. Kumaraswamy (JDS)	33.89	C. P. Yogeeshwara (BJP)	7.44	63.99%
Kanakapura	D. K. Shivakumar (INC)	26.88	R. Ashoka (BJP)	2.42	83.48%
Varuna	Siddaramaiah (INC)	25.08	V. Somanna (BJP)	8.82	47.97%
Vijay Nagar	H. Ravindra (BJP)	1.04	M. Krishnappa (INC)	0.93	5.84%

Table 6. Some indicators from the test tweets

Total Pairs	5
Average Predicted Net Margin	6.06
Average Predicted Percent Margin	55.01%

As an example of the validity of the results, Ravi Subramanya was predicted as more likely to win than U. B. Venkatesh, with scores of 19.0 and 2.88, respectively. This is likely accurate as Ravi Subramanya won the 2018 election seat with 2 times more votes than his opponent.

Some statistics on the testing tweets are shown below. The maximum counts all came from Siddaramaiah.

Table 7. Some statistics from the test tweets

Minimum Likes	0	**Maximum Likes**	3383
Minimum Replies	0	**Maximum Replies**	678
Minimum Retweets	0	**Maximum Retweets**	843

Comparing the test data against the results of the 2023 election, we can see that the model achieves 80% accuracy. The table of predictions compared to the actual results is given.

Table 8. The results of the 2023 election predictions

Constituency	Predicted Winner	Actual Result	Correctly Predicted
Basavanagudi	Ravi Subramanya (BJP)	Ravi Subramanya (BJP)	✔
Channapatna	H. D. Kumaraswamy (JDS)	H. D. Kumaraswamy (JDS)	✔
Kanakapura	D. K. Shivakumar (INC)	D. K. Shivakumar (INC)	✔
Varuna	Siddaramaiah (INC)	Siddaramaiah (INC)	✔
Vijay Nagar	H. Ravindra (BJP)	M. Krishnappa (INC)	✘

It can be seen here that the wrong prediction came from the Vijay Nagar constituency. However, the confidence was low for the respective candidate pair. At the same time, it was higher for the other candidate pairs, which indicates a close result for this constituency supported by the close scores for the two candidates from Vijay Nagar.

5. Discussion

The model used was able to achieve a relatively high accuracy of 72.7% when tested with the results of the previous 2018 Karnataka State Elections. Yet, due to limitations in the Twitter API, the total number of tweets collected and the number of politicians' tweets collected remain low.

However, the model proves promising for future applications. Overall, this model seems promising for future elections as positive sentiments and tweet engagement can be seen to correlate to a higher chance of winning, proving the hypothesis. This can be applied to Tweets from right before the upcoming 2023 elections to help predict which politician will win from each constituency. For this, a better data collection method needs to be found.

5.1. Limitations

Due to the nature of this research, there were many limitations involved. Since the tweets were collected from Twitter, which has a severely limited API, an external dependency was required to collect the tweets. Yet, Twint still proved to be slow and unresponsive at times, which hindered the data collection phase. Not all tweets were collected from within the dates given, and sometimes the API did not even connect. Currently, Twint is no longer supported, and its GitHub repository, at https://github.com/twintproject/twint, was archived on Mar 30th, 2023.

Moreover, due to the vast nature of elections, not all factors could be considered for the model used. This includes the volume of tweets per candidate, the demographics of the commenters on Twitter, and the exact topics spoken about in the tweet. Similarly, not all factors that could be analyzed were analyzed, such as the topic being discussed in the tweet and the date of the tweet, which measures relevance. Additionally, elections are inherently difficult to predict, owing to humans' complex and unpredictable nature and the possible skew of demographics on Twitter.[6]

Lastly, the fact that not all candidates in the Karnataka legislative assembly are on Twitter or have tweets from the selected date range limits the number of data points that can be gathered. Out of the 64 politician pairs considered, only 30 of the pairs were viable, with both having accounts. Out of these, many were too recent to be used or did not have any tweets, further reducing the volume of tweets that could be collected. Not all politicians are on Twitter or are active. From the 64 pairs of politicians considered for analysis (out of the 224 possible seats), only 11 pairs had tweets that could be analyzed.

These same limitations applied to the prediction pairs as well. As the Twint API was down at the time of data collection, the tweets had to be collected manually. Therefore, the number of predictions that could be made

and the number of tweets per prediction were restricted. Moreover, not all politicians had accounts or sufficient tweets that could be analyzed as well, further restricting the amount of data that could be collected.

5.2. Scope For Further Improvement

Further research can be conducted on the same topic to increase the paper's scope and accuracy. A machine learning model, such as a neural network, can be used that selects the most optimal weights and factors automatically, increasing accuracy considerably. Similarly, different factors should also be included in the scope of this paper, including the date the tweet was published (which will help measure relevance), the content of the tweet, and the sentiment of comments on the tweet.

Moreover, the scope can be increased to cover other states in South India to increase the number of data points gathered. Importantly, other regional languages can also be analyzed. This is crucial as regional languages are often underrepresented in sentiment analysis.[13] This may also help increase the generalizability of the results. The new social media website Koo may be used, specifically targeting Indians using local languages.

It was noted during tweet collection that more macro topics, such as national news and infrastructure development, were more positively received, so this angle could be explored further. In this way, the content of the tweets should be analyzed. It could be factored in if a certain subject correlates to higher engagement or negative publicity for the opposition.

6. Conclusion

The sentiment analysis of tweets from local politicians is a promising avenue to understand public opinion better and predict election results.[4] This paper innovates on previous attempts by looking at tweets from local politicians from Karnataka, including those in Kannada. Furthermore, it analyzed the engagement of the tweets as well to provide a more holistic view of the sentiments expressed.

Moreover, this method provides a cheaper alternative to using exit polls as this data can be collected online automatically and well in advance of the elections, providing clear benefits over existing methods. Suppose tweets from other states in India can be analyzed, and the new social media website Koo (specifically targeted towards Indians using local languages) is utilized. In that case, the results can be generalized to allow the model to predict election results from all over India.

Appendix 1

The table below gives the full list of candidates whose data was considered for collection from the 2018 elections. This includes the winning candidate and their party, the opposition candidate and their party, the constituency fought for, and if both candidates had a Twitter account.

A dash is given in the 'Username' column if the considered candidate does not have a Twitter account. A cross is given in the 'Twitter Status' column if at least one candidate in that row does not have a Twitter account. A tick mark implies that the candidate pair has complete data and has been input into the model.

Table 9. The complete list of candidates considered for the test tweets

Constituency	Winning Candidate			Opposition Candidate 1			Twitter Status
	Name	Username	Party	Name	Username	Party	
Kudachi	P. Rajeev	@PRajeevBJP	BJP	Amit Shama Ghatage	-	INC	✗
Gulbarga Rural	Basawaraj Mattimud	@b_mattimadu	BJP	Vijaykumar G. Ramakrishna	@VijayKumarGRam2	INC	✓
Kaup	Lalaji Mendon	@lalajibjp	BJP	Vinay Kumar Sorake	@VKSorake	INC	✓
Shanti Nagar	N. A. Haris	@mlanaharis	INC	K. Vasudevamurthy	-	BJP	✗
Gokak	Ramesh Jarkiholi	Ramesh Jarkiholi	INC	Ashok Pujari	-	BJP	✗
Ron	Kalakappa Bandi	@kalakappaGBandi	BJP	Gurupadagouda Patil	-	INC	✗
Bhatkal	Sunil Biliya Naik	@sunilnaikbhatkl	BJP	M. S. Vaidya	-	INC	✗

C. V. Raman Nagar	S. Raghu	@mla_raghu	BJP	R. Sampath Raj	@SampathRajR	INC	✔
Chamundeshwari	GT Devegowda	@GTDevegowda	JDS	Siddaramaiah	@siddaramaiah	INC	✔
Sakleshpur	H. K. Kumaraswamy	-	JDS	Somashekar Jayaraj	-	BJP	✘
Belgaum Uttar	Anil S Benake	@BenakeAnil	BJP	Fairoz Nuruddin Saith	-	INC	✘
Kampli	J.N. Ganesh	@J_N_Ganesh	INC	T H Suresh Babu	@THsureshbabu01	BJP	✔
Belthangady	Harish Poonja	@HPoonja	BJP	K. Vasantha Bangera	-	INC	✘
Padmanabhanagar	R. Ashoka	@RAshokaBJP	BJP	V. K. Gopal	-	JDS	✘
Bangalore South	M. Krishnappa	@MLAMkrishnappa	BJP	R. K. Ramesh	@RKRames76169218	INC	✔
Channapatna	H.D. Kumaraswamy	@hd_kumaraswamy	JDS	C. P. Yogeeshwara	@CPYogeeshwara	BJP	✔
Chitradurga	G. H. Thippareddy	@BJP_GHT	BJP	K. C. Veerendra	@Puppy_JDS_CTA	JDS	✔
Athani	Mahesh Kumathalli	@kumathalliM	INC	Laxman Savadi	@LaxmanSavadi	BJP	✔
Basavanagudi	L. A. Ravi Subramanya	@Ravi_LA	BJP	K. Bagegowda	-	JDS	✘
Nanjangud	Harshavardhan B.	@NanjangudMLA	BJP	Kalale N. Keshavamurthy	-	INC	✘
Chamaraja	L. Nagendra	@NimmaNagendra	BJP	Vasu	@MlaVasu	INC	✔
Yelahanka	S. R. Vishwanath	@SRVishwanathBJP	BJP	A. M. Hanumanthegowda	-	JDS	✘
Malleshwaram	Dr. C.N. Ashwath Narayan	@drashwathcn	BJP	Kengal Shreepadha Renu	@KengalShreepada	INC	✔
Gandhi Nagar	Dinesh Gundu Rao	@dineshgrao	INC	A. R. Sapthagiri Gowda	@sapthagirigowda	BJP	✔
B.T.M. Layout	Ramalinga Reddy	@RLR_BTM	INC	Lallesh Reddy	@LalleshReddyGlr	BJP	✔
Jayanagar	Soumya Reddy	@Sowmyareddyr	INC	B. N. Prahlad	@BjpPrahlad	BJP	✔
Mangalore City North	Bharath Shetty	@bharathshetty_y	BJP	Mohiuddin Bava	@mohiuddinbava55	INC	✔

Belgaum Dakshin	Abhay Patil	@iamabhaypatil	BJP	M. D. Lakshminarayana	-	INC	✗
Bagalkot	Veerabhadrayya Charantimath	@V_Charantimath	BJP	H. Y. Meti	@HYMeti1	INC	✔
Chittapur	Priyank M. Kharge	@PriyankKharge	INC	Valmiki Nayak	-	BJP	✗
Bhalki	Eshwara Khandre	@eshwar_khandre	INC	D. K. Sidram	@DKSidram	BJP	✔
Raichur	Dr. Shivaraj Patil	-	BJP	Syed Yaseen	-	INC	✗
Dharwad	Amrut Desai	@amrutdesaibjp	BJP	Vinay Kulkarni	-	INC	✗
Bellary City	G. Somashekara Reddy	@GSReddyBJP	BJP	Anil Lad	-	INC	✗
Davanagere North	S. A. Ravindranath	-	BJP	Shamanur Mallikarjun	-	INC	✗
Davanagere South	Shamanur Shivashankarappa	@DvgShamanurS	INC	Yashavantha Rao Jadhav	-	BJP	✗
Tumkur City	G. B. Jyothi Ganesh	@Mlajyothiganesh	BJP	N. Govindaraju	@Govindaraju_JDS	JDS	✔
Rajarajeshwari nagar	Munirathna	@MunirathnaMLA	INC	P. Muniraju Gowda	@tulasimuniraju1	BJP	✔
Shiggaon	Basavaraj Bommai	@BSBommai	BJP	Sayed Azeempeer Khadri	-	INC	✗
Hebbal	Suresha BS	@INCHebbal	INC	Y. A. Narayanaswamy	@YAN_MLC	BJP	✔
Bommanahalli	M Satish Reddy	@msrbommanahalli	BJP	Sushma Rajagopala Reddy	-	INC	✗
Hoskote	M. T. B. Nagaraj	@MTB_Nagaraj	INC	Sharath Kumar Bachegowda	-	BJP	✗
Devanahalli	Narayanaswamy L. N.	-	JDS	Venkataswamy	-	INC	✗
Mandya	M. Srinivas	-	JDS	P. Ravikumar	-	INC	✗
Shrirangapattana	Ravindra Srikantaiah	-	JDS	A. B. Ramesha Bandisiddegowda	@bandisiddegowda	INC	✗
Mangalore	U. T. Khader	@utkhader	INC	Santhosh Kumar Rai Boliyaru	@RaiBoliyar	BJP	✔

Madikeri	Appachu Ranjan	@ravimlapa	BJP	B A Jivijaya	-	JDS	✗
Chamarajanagar	C. Puttarangashetty	@ucpshetty	INC	K. R. Mallikarjunappa	@KRMalikarjunapa	BJP	✓
Doddaballapur	T. Venkataramanaiah	@Nimma_MLA	INC	B. Munegowda	-	JDS	✗
Nelamangala	Dr. K. Srinavasamurthy	-	JDS	R. Narayanaswamy	-	INC	✗
Kanakapura	D. K. Shivakumar	@DKShivakumar	INC	Narayana Gowda	@narayanagowdakc	JDS	✓
Nippani	Shashikala Jolle	@ShashikalaJolle	BJP	Kakaso Pandurang Patil	-	INC	✗
Chikkodi-Sadalga	Ganesh Hukkeri	@HukkeriGanesh	INC	Annasaheb Jolle	@annasahebsjolle	BJP	✓
Kagwad	Srimant Patil	@shrimantpatil_	INC	Raju Kage	@rajukage	BJP	✓
Khanapur	Anjali Nimbalkar	@DrAnjaliTai	INC	Vithal Halagekar	-	BJP	✗
Terdal	Siddu Savadi	@siddusavadi_bjp	BJP	Umashree	-	INC	✗
Bijapur City	Basangouda Patil Yatnal	@BasanagoudaBJP	BJP	Abdul Hameed Mushrif	@HamidMushrif	INC	✓
Jevargi	Ajay Singh	@Dr_Ajay_Singh	INC	Doddappa Gouda S. Patil Naribol	-	BJP	✗
Bidar	Rahim Khan	@RahimKhan_MLA	INC	Surayakanth Nagmarpalli	-	BJP	✗
Bidar South	Bandeppa Kashempur	@kashempur	JDS	Dr. Shailendra Beldale	@SBeldale	BJP	✓
Shivajinagar	R. Roshan Baig	@RRoshanBaigOff1	INC	Katta Subramanya Naidu	@KSNBJP	BJP	✓
Govindraj Nagar	V. Somanna	@VSOMANNA_BJP	BJP	Priya Krishna	@Priyakrishna_K	INC	✓
Hassan	Preetham J. Gowda	@nimmapreetham	BJP	H. S. Prakash	-	JDS	✗
Belur	K. S. Lingesha	@lingeshksmla	JDS	H. K. Suresh	@hksureshbjp	BJP	✓

Appendix 2

This appendix gives the complete code used in the sentiment analysis of the training tweets. For the prediction code (which is a modified version of the below code), the code to scrape the politicians' tweets, or the complete list of tweets used as well as the *settings.json* file, visit the GitHub page at https://github.com/PrajwalMReddy/KannadaSentimentAnalysis.

```
# Imports

import os
import json
from datetime import datetime

import re
import string

from nltk import WordNetLemmatizer, tokenize
from nltk.corpus import stopwords
from nltk.sentiment import SentimentIntensityAnalyzer
from translate import Translator

# Global Variables

minLikes = 10000000
maxLikes = 0

minReplies = 10000000
maxReplies = 0

minRetweets = 10000000
maxRetweets = 0

# Helper Data Class

class CandidatePair:
  def __init__(self, candidate1, candidate2, predicted1, predicted2):
    self.candidate1 = candidate1
    self.candidate2 = candidate2
    self.predicted1 = predicted1
    self.predicted2 = predicted2

  def __repr__(self):
    return f"{self.candidate1}: {self.predicted1} | {self.candidate2}: {self.predicted2} {'✔' if self.is_correct_prediction() else '✘'}"

  def is_correct_prediction(self):
    if self.predicted1 > self.predicted2:
      return True
    else:
      return False

  def margin_of_victory(self):
    return self.predicted1 - self.predicted2

  def percent_margin_of_victory(self):
    total = self.predicted1 + self.predicted2
    return ((self.predicted1 - self.predicted2) / total) * 100


# Utility Functions

# Utility Method To Find All Politicians Whose Tweets Are Available
def get_politicians():
  path = "tweets-train"

  with open("settings.json", 'r') as file:
    settings = json.load(file)
    politicians = [politician["Name"] for politician in settings["Candidates"] if os.path.isdir(os.path.join(path, politician["Name"]))]

  for politician in politicians:
    yield politician

# Utility Method To Find All The Available Tweets From A Given Politician
def get_politicians_tweets(politician):
  global maxLikes
  global minLikes

  global maxReplies
  global minReplies

  global maxRetweets
  global minRetweets

  path = f"tweets-train/{politician}"
  tweets = []

  for filename in os.listdir(path):
    file_path = os.path.join(path, filename)

    if os.path.isfile(file_path):
      with open(file_path, 'r', encoding="utf-8") as file:
        data = json.load(file)

        if data["Likes"] > maxLikes:
          maxLikes = data["Likes"]
        elif data["Likes"] < minLikes:
          minLikes = data["Likes"]

        if data["Replies"] > maxReplies:
          maxReplies = data["Replies"]
        elif data["Replies"] < minReplies:
          minReplies = data["Replies"]

        if data["Retweets"] > maxRetweets:
          maxRetweets = data["Retweets"]
        elif data["Retweets"] < minRetweets:
          minRetweets = data["Retweets"]

        tweets.append(data)

  return tweets
```

```
def store_results(results):
  with open("./results.json", "w+") as file:
    json.dump(results, file)

# Functions For Sentiment Analysis

def sanitize_text(text):
  text = text.lower()
  text = re.sub(r'\n', '', text)

  translator = str.maketrans('', '', string.punctuation)
  text = text.translate(translator)

  lemmatizer = WordNetLemmatizer()
  words = tokenize.word_tokenize(text)
  words = [lemmatizer.lemmatize(word) for word in
words]

  stop_words = stopwords.words("english")
  filtered_text = [word for word in words if not word in
stop_words]

  return " ".join(filtered_text)

def sentiment_polarity(text):
  sia = SentimentIntensityAnalyzer()

  score = sia.polarity_scores(sanitize_text(text))
  key =
list(score.keys())[list(score.values()).index(max(list(score.v
alues())[:len(score) - 1]))]

  return score, key

def translate_text(json_tweet):
  if json_tweet["Language"] != "kn":
    return json_tweet["Tweet"]

  translator = Translator(from_lang='kn', to_lang="en")
  translation = translator.translate(json_tweet["Tweet"])

  return translation

# Core Functions

def calculate_score(json_tweet):
  text = json_tweet["Tweet"]

  if json_tweet["Language"] != "en":
    text = translate_text(json_tweet)

  score, key = sentiment_polarity(text)

  sent_pol = (score["pos"] - score["neg"]) if (score["pos"] -
score["neg"]) > 0 else 0  # Ensures The Sentiment Polarity
Is Positive
  likes_score = (json_tweet["Likes"] - minLikes) /
(maxLikes - minLikes)  # Scales The Likes Score To The
Range Of 0.0 to 1.0
  replies_score = (json_tweet["Replies"] - minReplies) /
(maxReplies - minReplies)  # Scales The Replies Score To
The Range Of 0.0 to 1.0
  retweets_score = (json_tweet["Retweets"] -
minRetweets) / (maxRetweets - minRetweets)  # Scales
The Retweets Score To The Range Of 0.0 to 1.0

  # The Final Total Score Is From 0.0 to 100.0
  total_score = 25 * (sent_pol + likes_score + replies_score
+ retweets_score)
  return total_score

def analyze_politician(politician):
  total_score = 0
  tweet_count = 0
  all_politicians_tweets =
get_politicians_tweets(politician)

  for tweet in all_politicians_tweets:
    total_score += calculate_score(tweet)
    tweet_count += 1

  avg_score = total_score / tweet_count if tweet_count != 0
else 0

  return avg_score

def analyze():
  scores = {}
  all_politicians = get_politicians()

  while True:
    try:
      politician = next(all_politicians)
      scores.update({ politician:
analyze_politician(politician) })

    except StopIteration:
      break

  store_results(scores)

def conclude():
  with open("./settings.json", 'r', encoding="utf-8") as file:
    settings = json.load(file)

  with open("./results.json", 'r', encoding="utf-8") as file:
    results = json.load(file)

  candidates = settings["Candidates"]
  candidatePairs = []

  for i in range(0, len(candidates), 2):
    if candidates[i]["Name"] in results.keys() and
candidates[i + 1]["Name"] in results.keys():
      candidatePairs.append(
        CandidatePair(candidates[i]["Name"],
candidates[i + 1]["Name"], results[candidates[i]["Name"]],
results[candidates[i + 1]["Name"]])
      )
```

```
  correct_predictions = 0
  total_predictions = len(candidatePairs)

  total_margin = 0
  total_percent_margin = 0

  print("All Predicted Pairs:")
  for pair in candidatePairs:
    print(pair)

    total_margin += pair.margin_of_victory()
    total_percent_margin +=
pair.percent_margin_of_victory()

    if pair.is_correct_prediction():
      correct_predictions += 1

  print(
    f"\nResults:\n"
    f"Total Correct Pairs: {correct_predictions}\n"
    f"Total Pairs: {total_predictions}\n"
    f"Average Net Margin: {(total_margin /
total_predictions) if total_predictions != 0 else 0}\n"
    f"Average Percent Margin: {(total_percent_margin /
total_predictions) if total_predictions != 0 else 0}%\n"
    f"Accuracy: {(correct_predictions / total_predictions)
* 100 if total_predictions != 0 else 0}%"
  )

# Code Entry Point

def main():
  analyze()
  conclude()

if __name__ == '__main__':
  main()
```

Funding Statement

This research paper was independently produced and written. There was no external funding for this research paper.

Acknowledgements

I would like to thank my mentor, Gurvansh Singh, for all the help and guidance he provided me with while I prepared this research paper.

References

[1] Apoorv Agarwal et al., "Sentiment Analysis of Twitter Data," *Proceedings of the Workshop on Language in Social Media,* pp. 30-38, 2011. [Google Scholar] [Publisher Link]

[2] Adam Bermingham, and Alan F. Smeaton, "On Using Twitter to Monitor Political Sentiment and Predict Election Results," *ACL Anthology,* 2011. [Google Scholar] [Publisher Link]

[3] Anastasia Giachanou, and Fabio Crestani, "Like It or Not: A Survey of Twitter Sentiment Analysis Methods," *ACM Computing Surveys*, vol. 49, no. 2, pp. 1–41, 2016. [CrossRef] [Google Scholar] [Publisher Link]

[4] Bhumika Gupta et al., "Study of Twitter Sentiment Analysis Using Machine Learning Algorithms on Python," *International Journal of Computer Applications,* vol. 165, no. 9, pp. 29–34, 2017. [CrossRef] [Google Scholar] [Publisher Link]

[5] Bhumika Pahwa, S. Taruna, and Neeti Kasliwal, "Sentiment Analysis- Strategy for Text Pre-Processing," *International Journal of Computer Applications,* vol. 180, no. 34, pp. 15–18, 2018. [CrossRef] [Google Scholar] [Publisher Link]

[6] Daniel Gayo-Avello, Panagiotis Metaxas, and Eni Mustafaraj, "Limits of Electoral Predictions Using Twitter," *Proceedings of the International AAAI Conference on Web and Social Media,* vol. 5, no. 1, pp. 490–493, 2021. [CrossRef] [Google Scholar] [Publisher Link]

[7] Francisca Onaolapo Oladipo, Ogunsanya Funmilayo Blessing, and Ezendu Ariwa, "Terrorism Detection Model using Naive Bayes Classifier," *SSRG International Journal of Computer Science and Engineering,* vol. 7, no. 12, pp. 9-15, 2020. [CrossRef] [Publisher Link]

[8] Luca Buccoliero et al., "Twitter and Politics: Evidence from the US Presidential Elections 2016," *Journal of Marketing Communications,* vol. 26, no. 1, pp. 88–114, 2020. [CrossRef] [Google Scholar] [Publisher Link]

[9] Malhar Anjaria, and Ram Mohana Reddy Guddeti, "A Novel Sentiment Analysis of Social Networks Using Supervised Learning," *Social Network Analysis and Mining,* vol. 4, 2014. [CrossRef] [Google Scholar] [Publisher Link]

[10] Payal Khurana Batra et al., "Election Result Prediction using Twitter Sentiments Analysis," *Sixth International Conference on Parallel, Distributed and Grid Computing (PDGC),* pp. 182-185, 2020. [CrossRef] [Google Scholar] [Publisher Link]

[11] Saifuddin Ahmed, Jaeho Cho, and Kokil Jaidka, "Leveling the Playing Field: The Use of Twitter by Politicians during the 2014 Indian General Election Campaign," *Telematics and Informatics,* vol. 34, no. 7, pp. 1377–1386, 2017. [CrossRef] [Google Scholar] [Publisher Link]

[12] S. Rajeswari et al., "Aspect Based Polarity Extraction in Tamil Tweets using Tree-Based Recursive Partitioning Techniques," *International Journal of Engineering Trends and Technology,* vol. 70, no. 12, pp. 421-430, 2022. [CrossRef] [Publisher Link]

[13] V. Rohini, M. Thomas, and C. A. Latha, "Domain Based Sentiment Analysis in Regional Language-Kannada using Machine Learning Algorithm," *IEEE International Conference on Recent Trends in Electronics, Information & Communication Technology,* pp. 503-507, 2016. [CrossRef] [Google Scholar] [Publisher Link]

[14] Yelena Mejova, *Sentiment Analysis: An Overview*, 2009. [Google Scholar] [Publisher Link]

[15] A Go, L Huang, and R Bhayani, Twitter Sentiment Analysis, *Entropy*, 2009. [Google Scholar]

[16] Farha Nausheen, and Sayyada Hajera Begum, "Sentiment Analysis to Predict Election Results using Python," *2nd International Conference on Inventive Systems and Control*, pp. 1259-1262, 2018. [CrossRef] [Google Scholar] [Publisher Link]

[17] Vishal A. Kharde, and Sheetal Sonawane, "Sentiment Analysis of Twitter Data: A Survey of Techniques," *International Journal of Computer Applications,* vol. 139, no. 11, pp. 5–15, 2016. [CrossRef] [Google Scholar] [Publisher Link]