İstanbul
Bilgi Üniversitesi
LAUREATE INTERNATIONAL UNIVERSITIES

SOCIAL MEDIA ANALYSIS OF TURKISH POLITICIANS

By
Fikret Efe Doğanay
Ali Çağan Keskin

Supervised
By
Uzay Çetin, Ph.D.

COMPUTER ENGINEERING DEPARTMENT

Contents

List of Figures

List of Tables

1 ABSTRACT

There is no doubt that social media offers great opportunities in terms of rapid data access and analysis. Thus, the new trend of data science is gradually shifting towards analyzing the massive data available on social media platforms. As the number of politicians involved in these platforms increases, multidisciplinary fields emerge such as political data science. In this paper, we aim to analyze Turkish politicians' tweets by using machine learning and network algorithms to develop an online tool to convey the results. We mainly examine the politicians' similarity to others that allows us to come up with a social network based on similarity metrics. In addition to that, sentiment analysis and tweet categorization are included for the further impression of a user. With the help of various data visualization techniques, results are implemented on an online platform, where users can explore outcomes in detail.

Keywords: *Social Media Analysis, Natural Language Processing, Political Data Science, Document Similarity*

2 INTRODUCTION

Social media has become a big part of the individuals' daily life, especially with the increasing usage of mobile phones. Without a doubt, Twitter is one of the most popular platforms among all others and it is mostly used by renowned people. These people can verify their accounts on Twitter, thus, giving other users an indication that their profile is not fake. This is extremely advantageous for famous people, considering that it is not possible to specify their identity on other platforms. Because of that, Twitter has quickly become the most powerful tool for politicians from all around the world. They regularly benefit from Twitter to reach out to their community. Therefore, their tweets are valuable sources for the different types of data analysis.

The goal of this work is to develop a tool that gives information about the tweets of Turkish politicians and create a relational network based on similarity metrics. Furthermore, statistical data analyses of politicians' tweets and their sentiment distribution are included. We utilized various text processing techniques that allow us to clean data. In order to create the network, tweets of a certain user are gathered to a document, and the cosine similarity of that document to others lets us explore the similarity between politicians. Upon completion of our analyses, we developed a website that users can easily access and allowed to analyze and compare Twitter users. A variety of data visualization options are presented on the website to increase the interpretability of our analyses for users.

3 RELATED WORK

Analyses on the Twitter platform has been increasing with the help of the availability of Twitter API. Fujino and Hoshino [2] showed that by applying document similarity algorithms, the similarity between both users and tweets can be calculated. Furthermore, Bagheri and Islam [3] worked on a sentimental analysis of tweets based on the three categories (negative, neutral and positive) and they found out that neutral sentiments are high, therefore the procedure of sentiment analysis needs further improvement. Lastly, Caetano, et. al, [4] worked on classifying the Twitter users' stand on the American presidential election held in 2016. Our project's difference is that we combine all of these different analyses into a tool. Creating a similarity network for these types of problems is not very common, but it results in informative data visualization. Thus, getting insights about a political environment becomes much easier.

4 TIMELINE

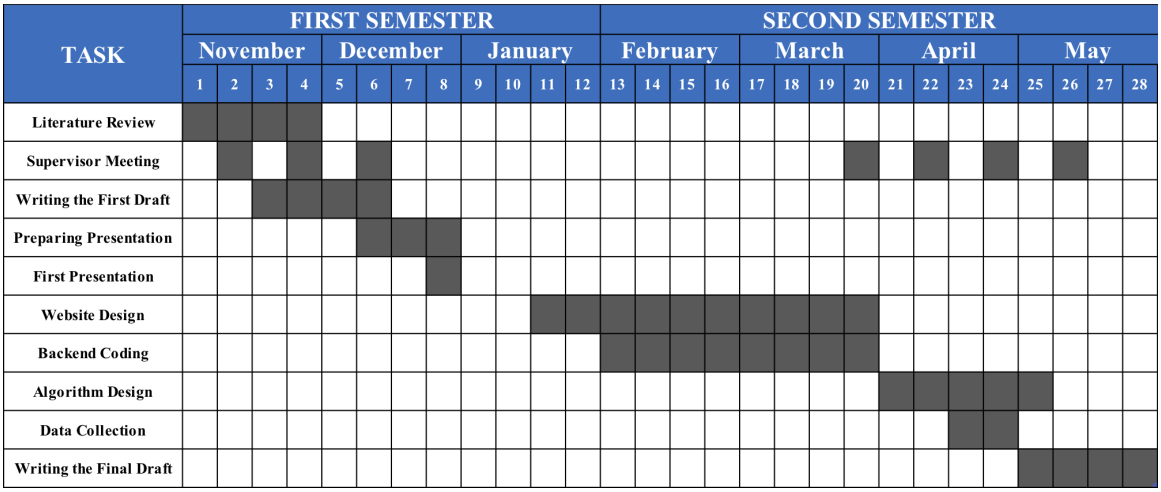

Figure 4.1: Timeline of the project

5 RISK ANALYSIS

Our first intention was to benefit from all the tweets of a certain user. However, Twitter API does not allow its' developers to retrieve more than 200 tweets per request. Therefore, we are currently able to show statistics for the last 200 tweets for the requested user. It is possible to retrieve more than 200 tweets by using third-party libraries, though they are not quite effective in terms of speed.

As stated above, the project's maintainability is completely dependent on the Twitter API and over the last years, the API underwent several changes. Thus, Twitter might also restrict access to the API in the future. In this scenario, we need to use one of the alternative approaches to fetch data, but they are not as efficient as the official API.

6 METHODS

6.1 Data Retrieval

6.1.1 The Twitter API

Twitter provides a specific Application Programming Interface called Twitter API. Researchers generally have free access to this API in order to gather data [1]. We also utilized this service, since it does not require programmers to scrape a whole page, allowing them to get data from Twitter's servers in a faster way. To securely access the data stream, the Tweepy library is used in the Python programming language.

```
import tweepy

auth = tweepy.OAuthHandler(consumer_key, consumer_secret)
auth.set_access_token(access_token, access_token_secret)
```

Figure 6.1: Setup of the Tweepy for Python

6.1.2 External Data Sources

The Twitter API is especially very useful for applications that include real-time data streaming. However, for machine learning tasks, we require huge amounts of data to create an efficient model. Fetching data from the Twitter API would not be enough to fulfill this requirement. Hence, for the sentiment analysis, data retrieved and labeled manually. For the text categorization, however, a library called "GetOldTweets3" is used to automatize the retrieval and labeling process.

6.2 Data Preprocessing

6.2.1 Data Cleaning

Tweets collected via Twitter API generally contain plenty of unwanted information. For our analysis to produce more accurate results, we applied the following techniques to raw data:

I) Filtering: This step involves the removal of textual data that is specific to Twitter such as emojis, multimedia files, and links.

II) Tokenization: Tokenization is about splitting a textual data into a list of smaller tokens that are linguistic units such as words, numbers, or punctuations. This is one of the essential parts of data cleaning and will help us to ease the natural language processing process.

III) Stemming: With stemming, our aim is to consider the morphological roots of some words. This is an important step as it enables us to neglect the effects of suffixes.

IV) Stopword Removal: There are certain words and articles in tweets that do not change the overall meaning, and these need to be removed. Well known examples of these in Turkish are ”de”, ”da”, ”ve”, ”veya”, ”yani”, ”ya” etc,.

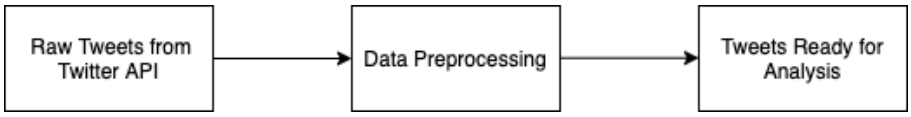

Figure 6.2: Summary of the data preprocessing process

6.3 Statistical Insights

Some of the basic statistical outcomes computed by using the data preparation steps mentioned in 6.2. These outcomes are listed as ”Most N used words”, ”Most N mentioned profiles” and ”Most N used hashtags”. For the hashtag and mention part, the prefix searching of ”#” and ”@” are implemented. Then, two hash tables

are created in which their keys are hashtag (or mentioned profile) and values are the corresponding counts. Finally, these hash tables sorted (in descending order) by value and the first item gives us the result. For the most used word part, a similar process is applied, but instead of searching for a prefix, we just tokenized tweets to extract the words out of them.

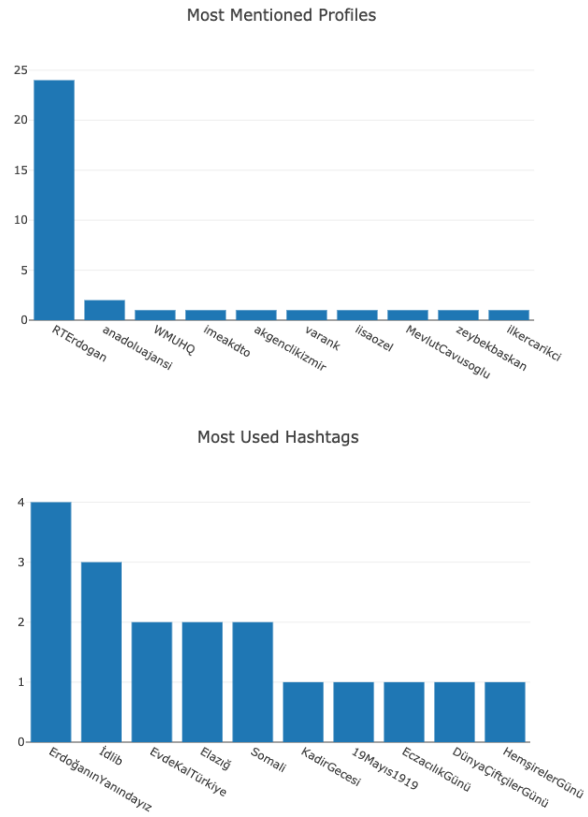

Figure 6.3: Most mentioned profiles and most used hashtags in Binali Yildirim's Tweets

6.4 Data Organization and Network Analysis

6.4.1 Term Frequency - Inverse Document Frequency

Finally, after all the data preprocessing steps, textual data needs to be converted into a form which can be represented with numbers. The underlying reason

for this is that textual data cannot be directly fed into any mathematical estimation models. To achieve this, we benefited from the Term Frequency and Inverse Document Frequency algorithm.

In order to apply tf-idf algorithm, we will first extract the words from documents that composed of tweets of a certain politician. First, we will give the brief definition of term frequency tf(k,j). tf(k,j) gives the appearance frequency of the term t_k [2]. If the appearance number of the term t_k in the document D_j is denoted as n_{kj}, the formula of the t(k,j) can be achieved as:

$$tf(k,j) = \frac{n_{kj}}{\sum_l n_{lj}} \tag{6.1}$$

Next, the equation of document frequency df(k) will be provided [2]. df(k) is the appearance frequency of the term t_k. When the term t_k is appeared in the document D_j at least once, we count the appearance number of document related term t_k at least one time. The total number of document in which the term t_k exists will be called as the appearance number of document regarding the all document in the data set and it will be specified as $|d_k|$ [2]. Furthermore, if the total number of whole document set is denoted as $|D|$, then we can give the document frequency formula as shown below:

$$df(k) = \frac{|d_k|}{|D|} \tag{6.2}$$

whereas the term t_k will become less important as df(k) becomes larger. To make it proportional to the significance of terms, the logarithm operation is used on inversed df(k) which is also called the inverse document frequency idf(k) [2]. The expression of idf(k) is as follows:

$$idf(k) = \log \frac{1}{df(k)} = \log \frac{|d_k|}{|D|} \tag{6.3}$$

As a result of the adjustment above, the term t_k and idf(k) will be positively correlated. Finally, the weight argument tfidf(k,j) of the term t_k in the document

D_j is described as the multiplication of term frequency tf(k,j) and the document frequency df(k) [2], which is stated below:

$$w_k^j = tfidf(k,j) = tf(k,j)idf(k) \tag{6.4}$$

This parameter will be used as a determiner factor for weighting terms.

6.4.2 Measuring the Document Similarity

By using the mathematical operations mentioned above, a document can be expressed as a numerical vector as follows [2]:

$$\vec{D_j} = (w_j^1, w_j^2, ...w_j^m) \tag{6.5}$$

where m is the dimension of the vector space, which is the number of terms present in the entire data. Thus the similarity between two documents $(\vec{D_i}, \vec{D_j})$, which are also the tweets of politicians, can be calculated with the help of cosine similarity formula given below:

$$\text{sim}(\text{D}_i, D_j) = \frac{\vec{D_i}.\vec{D_j}}{\left|\vec{D_i}\right|\left|\vec{D_j}\right|} = \frac{\sum_{k=1}^{m}(w_i^k.w_j^k)}{\sqrt{\sum_{k=1}^{m}(w_i^k)^2}.\sqrt{\sum_{k=1}^{m}(w_j^k)^2}} \tag{6.6}$$

6.4.3 Creating the Network

As the last step, calculated cosine similarities are used to create politicians' network. Our approach is simple and straightforward: politicians whose similarities are high compared to others, will have a darker edge to each other. To ensure this, in our network, each node represents the politician and each edge color represents the cosine similarity of one to the other. To illustrate, in Figure 6.4.3 below, the edge connecting "by" and "rterdogan" is very dark which means their tweets are highly similar.

Visualizing a complete network with high number of nodes is a difficult task. For a network having more than 8 nodes, certain edges which their similarity are lower than the average similarity of the network are removed. This makes the visualization much more understandable.

For instance, in Figure 6.4.3, there are no edges connected to ”herkesicinchp” and ”akparti” accounts. This is because the similarities of them fall below the average.

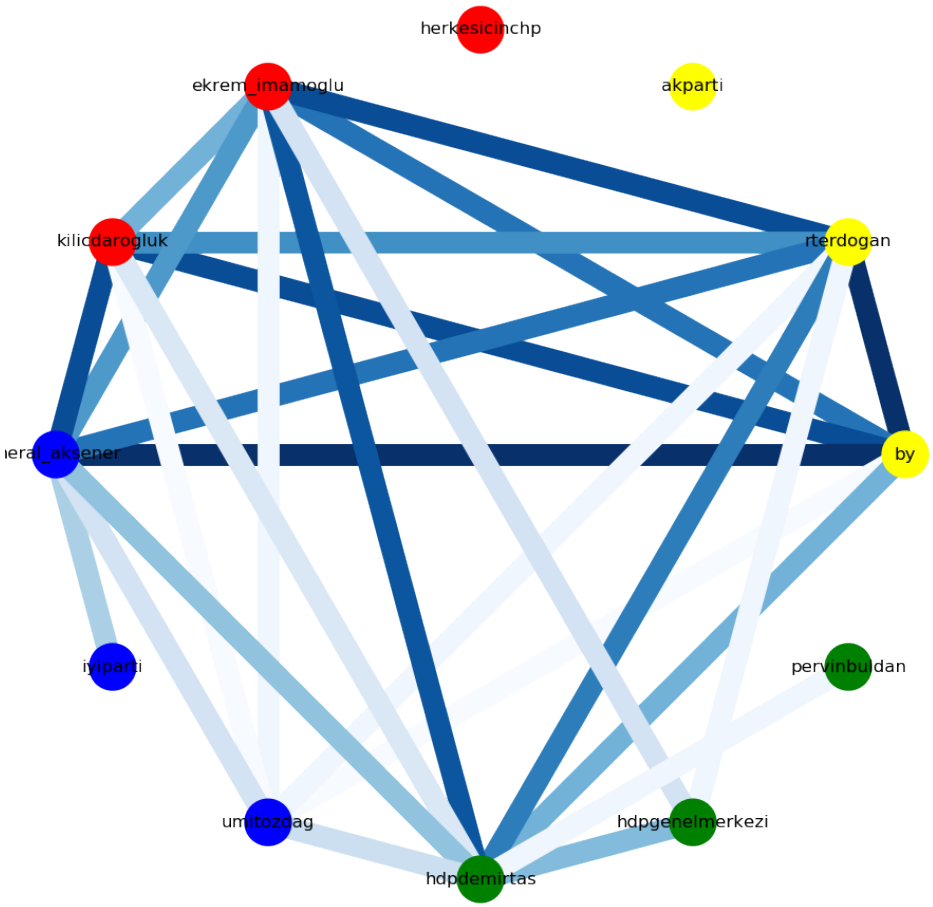

Figure 6.4: Sample similarity network between AKP (Yellow), CHP (Red) Iyi Party (Blue), and HDP (Green)

6.5 Sentiment Analysis

6.5.1 Overview

Sentiment analysis is about discovering the overall sentiment of a sentence. In general, the two most commonly used classes are positive and negative. It must also be noted that as number of the classes increases, it becomes difficult to distinguish sentiments for both humans and computers. Different from the network analysis, this part involves the use of supervised machine learning models. Therefore, we needed high amounts of labeled data for our models to be accurate.

6.5.2 Details of the Data

As mentioned in 6.1.2, data size needs to be as big as possible for efficient machine learning models. For sentiment analysis, we have manually collected and labeled data that consists of 15.000 tweets in Turkish and their corresponding sentiment labels. The data is equally distributed with positive and negative sentiments.

6.5.3 Classification, Training, and Testing

Steps explained in 6.2 are applied to get the data suitable for the machine learning phase. Before applying any classification algorithm, the data is split as the training and test data with 80% to 20% ratio respectively. Having this, since our task is a binary classification problem, we applied the logistic regression algorithm. The training data is used to create the model, whereas the testing data is used to evaluate the performance of that model.

6.5.4 Results

After the training and testing stage, the performance of the model was evaluated and the trained model was able to predict the sentiment of a tweet with 69.5% accuracy.

Sentiment	Precision	Recall	F1 Score
Negative	0.74	0.63	0.68
Positive	0.66	0.77	0.71

Table 1: Performance of the sentiment classification metrics

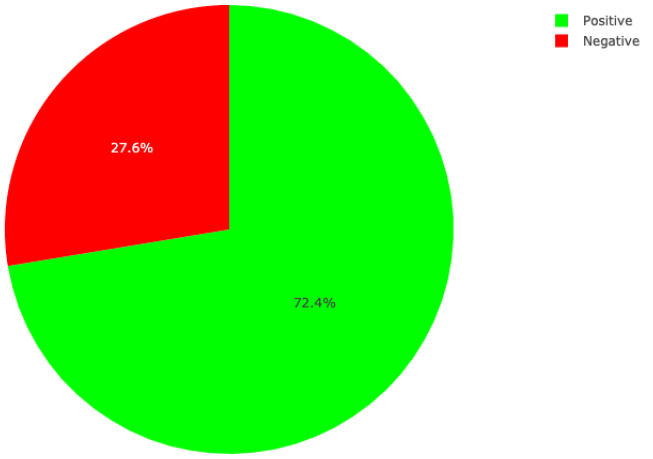

Figure 6.5: Sentiment analysis of Ekrem Imamoglu's tweets

6.6 Tweet Categorization

6.6.1 Overview

Similar to sentiment analysis, tweet categorization is a supervised classification task. In order to find out the category of a tweet, 4 different classes have been predefined, that are health, politics, economy and other. The "other" class is introduced since a tweet does not necessarily give information in the context of health, politics or economy.

6.6.2 Details of the Data

With the help of the GetOldTweets library in Python, we have created our own data by collecting tweets which include a hashtag of one of the three categories. Then, corresponding labels are assigned to these tweets. A total of 60.000 tweets were collected with equal distribution of the three classes.

6.6.3 Classification, Training, and Testing

A 4:1 ratio of train/test split is applied to data. Before any training, a threshold predefined to introduce the ”Other” class, which is not present in the initial data. The idea is that, if the accuracy of a tweet is less than this threshold and if it is classified as one of the three classes, the category is changed to ”Other” class. The gaussian naive Bayes algorithm is implemented in the training phase due to the data's multiclass structure.

6.6.4 Results

Obviously, it is harder to obtain a higher accuracy score in a multiclass classification compared to the binary one. However, our tweet categorization model outperformed the sentiment analysis model mentioned in 6.5.4 with an accuracy of 77.3%. This is most probably due to the larger data size and the difficulty of determining the sentiment of a tweet.

Sentiment	Precision	Recall	F1 Score
Politics	0.75	0.69	0.72
Economy	0.76	0.81	0.79
Health	0.81	0.80	0.81

Table 2: Performance of the tweet categorization metrics

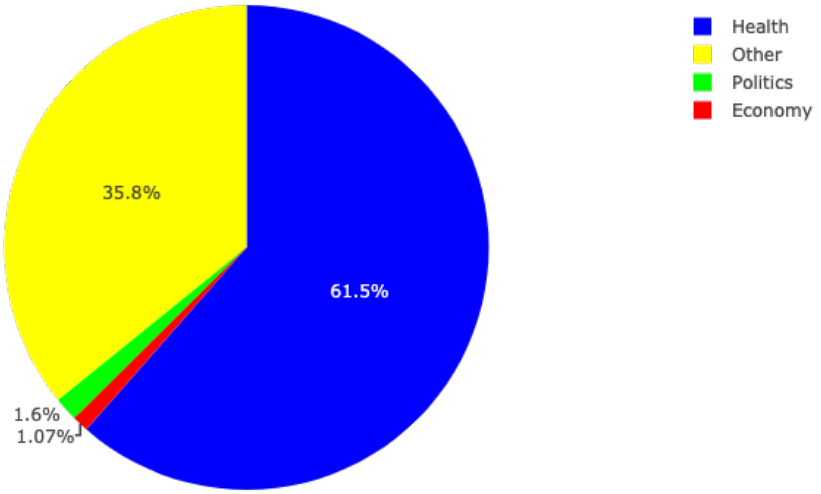

Figure 6.6: Categorization of the tweets of Dr. Fahrettin Koca (The Minister of Health in Turkey)

7 DEVELOPMENT OF THE WEBSITE

7.1 Architecture

We added our works to a website to enable users to explore Twitter profiles. To achieve this task, we have mainly used Flask for the backend, Dash for dynamic data visualization page, and Twitter API for real-time data streaming. For the frontend, we used HTML, CSS, Bootstrap, Plotly, React, and JavaScript technologies. Currently, the website is only available on local but it will be deployed on a host as soon as some additional services are implemented. Currently, there is no need to implement a database for our application since it is based on real-time data retrieval.

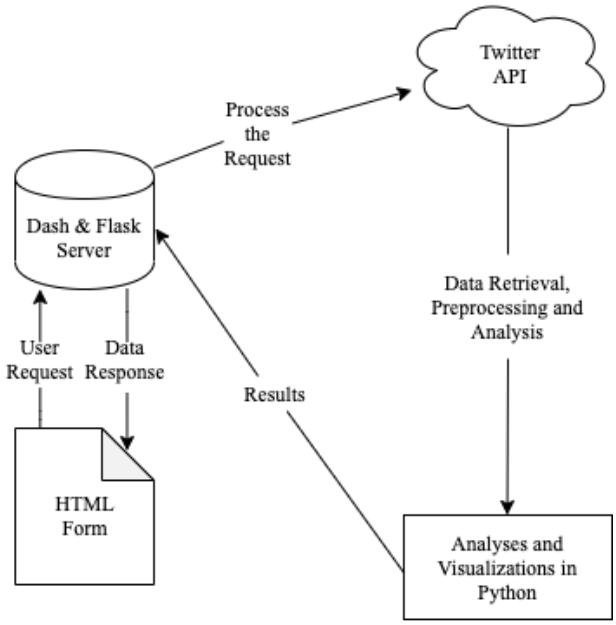

Figure 7.1: Main architecture of the website

7.2 User Actions

Although this project is based on the political environment of Turkey, the website appeals to everyone who wants to analyze Twitter profiles. Basically, as soon as a user enters a Twitter profile, he has access to features mentioned in 6.3, 6.5, and 6.6 only if the requested Twitter profile exists and open to public. He also has an option whether to include retweets of that profile or not. Conditions stated above apply for the network analysis as well, that is, profiles to be added must be valid Twitter accounts and publicly available. Furthermore, users are allowed to download the produced visualizations.

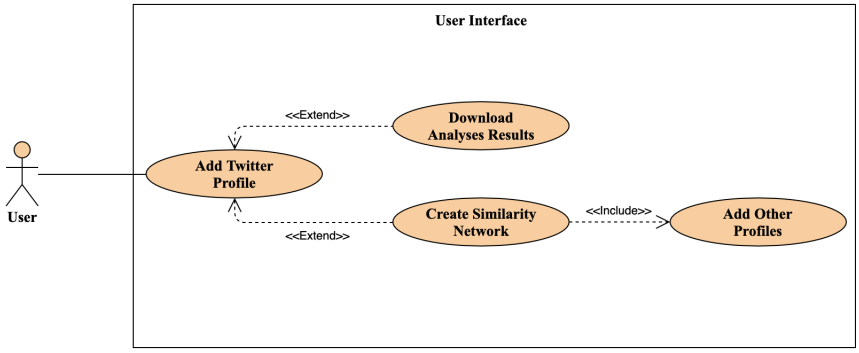

Figure 7.2: Use case diagram of the website

7.3 Test Cases

There are several important parts needs to be handled for the website's implementation. These parts can be listed as:

I) Entering a non-existent/private profile: If user enters a profile which does not exist on Twitter or it's private, he/she gets a warning message.

II) No tweets: If user enters a profile which exists on Twitter but has no

tweets at all, he/she gets a warning message indicating ”Profile has not tweeted yet, there is nothing to be analyzed.”.

III) No 10 hashtag/word/mention: If user enters a profile which has less than 10 of either hashtag, word, or mention, the histogram gets updated accordingly and there will be no warning message.

IV) Adding the same profile for the network analysis: If user enters the same profile again in the network analysis section, he/she gets a warning message indicating ”The profile was already added.”.

7.4 Design

7.4.1 Overview

As stated in 7.1, the website is built by using the Python programming language. The application's server runs on Flask framework and there are three distinct pages as ”Home”, ”About” and ”Analysis”. In the home and the about page, the general overview of the website and our background are included. The analysis page is developed with Dash framework and Flask redirects to it if user goes to that page. All of these pages' frontends are designed by using the Bootstrap framework.

7.4.2 Details of the Coding

Apart from the CSS and HTML files, the application has mainly four core Python classes as:

I) app.py: This is where the core functionalities of the Flask server are implemented. Page redirections, port changes and the listeners of the components are also included here.

II) twitterapi.py: This class is created to connect to the Twitter API based on the requests coming from app.py.

III) analysis.py: This class includes all the implementations of the data preprocessing, analyses, network creation, machine learning and natural language processing algorithms.

IV) interface.py: This is where the Python data types combined with the HTML structure can be found. None of the functions regarding server or algorithm is implemented here.

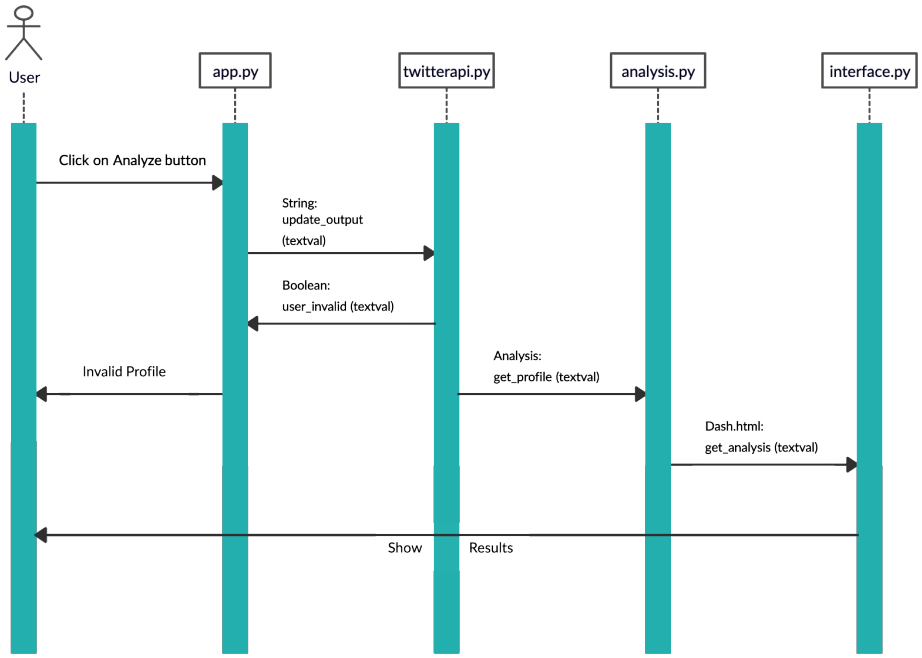

Figure 7.3: UML Sequence Diagram

8 CONCLUSION

Social media is increasingly becoming popular amongst all different groups of society. People who frequently use social media platforms are part of the newly emerged notion of the social media network. Politicians benefit from this new trend of social media interaction, in order to convey their message in a rapid and reliable fashion. Therefore, their data is suitable for analyzing the political environment. This project aims to gain insight about the political context in Turkey, merely based on the raw data found on Twitter. It is based on the three main approaches that can be listed as network analysis, sentiment analysis and the tweet categorization. Out of these, the network analysis produces us a visual feedback to examine the tweets of politicians from different political parties. As it can be seen from the Figure 6.4.3, with some exceptions, politicians who are within the same party have darker edge colors. Another interpretation is that except from ”hdpgenelmerkezi” and ”iyiparti”, the similarities of ”herkesicinchp” and ”akparti” are very low, thus they have no edges at all. This leads us to consider that the similarity between a political party and a politician might get low compared to the similarity between a politician and politician. Apart from the network analysis, the tweet categorization algorithm performs consistently. A good illustration of this is provided in Figure 6.6.4, where the tweets of Dr. Fahrettin Koca are mostly about health with 61.5%.

9 FUTURE WORK

Finding a high quality of data is the integral part of this project. Especially, when it comes to social media platforms, this gets even harder because of the violation of grammar. Hence, one of the efficient ways to increase this project's efficiency is implementing algorithms with big and high quality of labelled data. As manual labeling is an exhaustive task, we plan to overcome this by using crowdsourcing. Moreover, we believe that this project can be extended in a way that it includes the analyses of other languages. Unfortunately, there is nothing can be done to increase the speed of tweet retrieval, because it's based on the Twitter API. However, if a satisfactory progress is made, this work will be launched with AWS (Amazon Web Services) so that it helps our algorithms to compute faster. Finally, we will get opinions of sociologists and political scientists to evaluate our conclusions.

10 REFERENCES

[1] Pfeffer, J., Mayer, K. Morstatter, F. Tampering with Twitter's Sample API. EPJ Data Sci. 7, 50 (2018) doi:10.1140/epjds/s13688-018-0178-0

[2] I. Fujino, Y. Hoshino "Finding similar tweets and similar users by applying document similarity to twitter streaming data" Information and Communication vol. 6 no. 2 pp. 22-30 2013.

[3] Bagheri, Hamid Islam, Md Johirul. (2017). Sentiment analysis of twitter data.

[4] Caetano, J., Lima, H., Santos, M. et al. Using sentiment analysis to define twitter political users' classes and their homophily during the 2016 American presidential election. J Internet Serv Appl 9, 18 (2018) doi:10.1186/s13174-018-0089-0