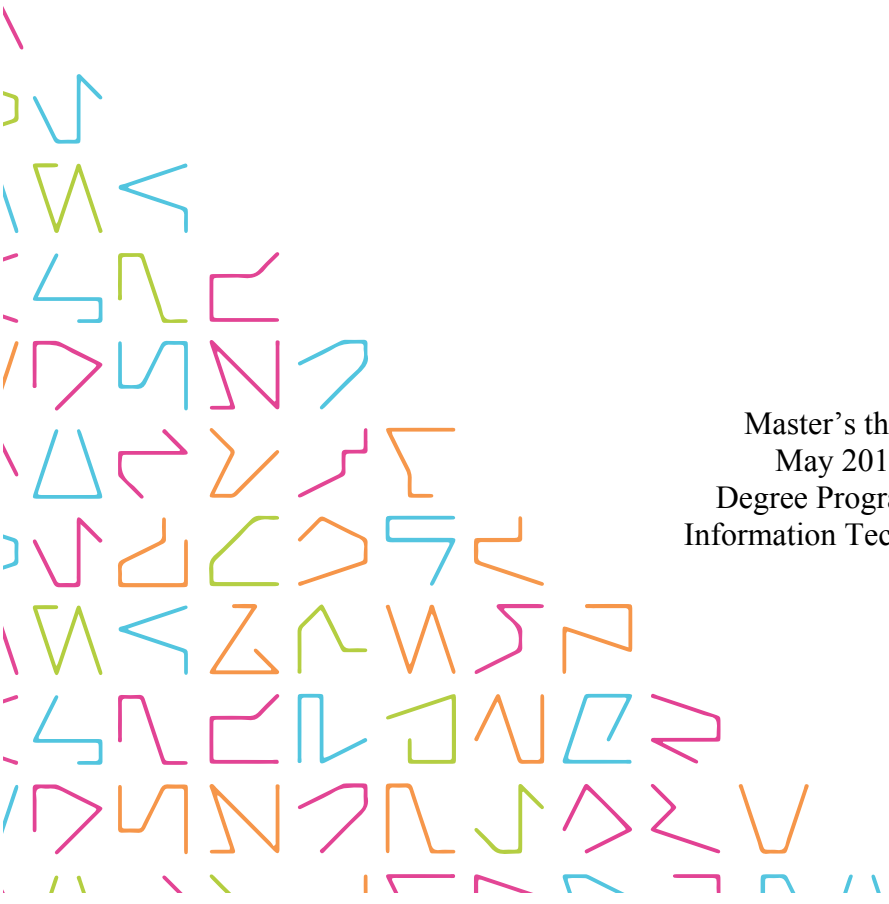
TAMPERE UNIVERSITY
OF APPLIED SCIENCES

Finnish Mood

A Sentiment Analysis Application for Twitter Data

Mojtaba Ahmadi



Master's thesis
May 2017
Degree Programme
Information Technology

ABSTRACT

Tampereen ammattikorkeakoulu
Tampere University of Applied Sciences
Degree Programme in Information Technology

Mojtaba Ahmadi
Finnish Mood
A Sentiment Analysis Application for Twitter Data

Master's thesis 57 pages
June 2017

Sentiment analysis or opinion mining is an emerging and fast-growing field of modern science which aims to understand people's opinion from texts. Sentiment analysis applications are growing in many different fields such as marketing and politics. As an example, sentiment analysis is being used for evaluating the products from customer's reviews.

The purpose of this study was to gather information related to make a sentiment analysis application to estimate how Finns generally feel in a specific time, what is the most important topics for them on that time and how are their attitudes toward each topic, e.g. In 6th of June people in Finland felt positive with average grade 6 out of 10 and the most positive topic was 'weather' with average grade 8 out of 10.

A light working version of the application was created using Twitter data. After the data were collected from Twitter APIs, they were analysed and graded in backend server and the result was published in a web page. The result of each topic was calculated on the scale of 1 to 10 and in three different opinion classification. positive, negative and neutral.

The outcome of this study along with implemented application will be used as parts of the proof of concept (POC) for further developments which could be either in form of a web-service or an independent API, especially by implementing the service in the Finnish language which currently is not available.

Keywords: sentiment analysis, opinion mining, Twitter data analysis, Finnish opinion

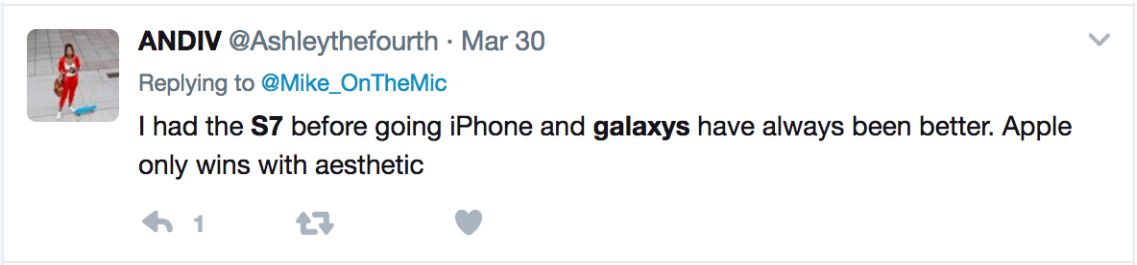
CONTENTS

GLOSSARY

SA	Sentiment Analysis
REST	Representational state transfer
API	Application Program Interface
WOEID	Where On Earth Identification
NLP	Natural Language Processing
BIONLP	Biomedical Natural Language Processing
OP	Opinion Mining
ML	Machine Learning
NB	Naïve Bayes
SVM	Support Vector Machine
RT	Retweet
SMS	Short Message Service
APP	Application
JSON	JavaScript Object Notation
HTTP	Hyper Text Transfer Protocol
HTTPS	Hyper Text Transfer Protocol Secure
OAUTH	Open Authentication
JS	JavaScript
OS	Operating System
MVC	Model View Controller
MVVM	Model-View-View-Model
REST	Representational state transfer

1 INTRODUCTION

Online social media in recent years has become so popular between different group of people for different reasons. Social network categories include different contexts such as the social network for professionals (the LinkedIn), the social network for photographers and designers (the Instagram, the Pinterest) and other group of users. The internet and mobile technologies have affected on rising of social media by providing platforms for information sharing, generating content and interactive communication. User's contribution in social media is very important and it is considered as a valuable resource. These big amounts of data have attracted different groups to understand and use these data for either improving their field or making knowledge to create new fields. Companies are increasingly investing in social media marketing, which is considered as the next generation of digital marketing channels. They use the data shared by users to understand how users feel about their products, which kind of product is popular, how they can improve their product and other upcoming questions. Picture 1 shows a tweet by a user about comparing two phone's brands together. These data are valuable for companies to compete in the market.



PICTURE 1. Tweet by a user about comparing two phone's brand together

Sentiment analysis or opinion mining is the computational study of people's opinions, attitudes, and emotions toward an entity. The input can represent individuals, events or topics which most likely are coming from reviews, tweets, and comments. The two expressions sentiment analysis and opinion mining are interchangeable and refer to same meaning, however, in some resources opinion mining consider slightly different with sentiment analysis. The point that some researcher differs between opinion mining and sentiment analysis is the opinion mining extract the people's opinion from text data,

however sentiment analysis focus on the sentiment of the given text data. In this research, we just are looking for the sentiment of data and its values.

The overall purpose of the thesis is to study and get a deeper comprehension of how sentiment analysis and opinion mining works and then using the knowledge to make an application by utilising different tools and frameworks. In a simple description, the application aims to read the data from Twitter and then analyse the sentiment of collected texts and show the results to users. The final version of this application will be used to analyse data related to Finland and it could be attracted to different user groups such as normal users, companies, business holders, politicians, etc.

1.1 The company and the business

The business idea of this project is owned by Unseen Technologies Oy, an IT company based in Tampere, Finland. The company was established in 2015 and their activities cover different IT fields. Based on the license agreement the outcome of this study, documents, designs, source codes and deployment packages are intellectual property right (IPR) of Unseen Technologies Oy and all version of Licensed Good should include Branding Text (“Powered by Unseen Technologies”) and visible to the end user of Licensed Good in all views.

1.2 The product

The final product consists of web application backend server, web services implementations, data storage implementation, mobile applications and APIs. The figure 1 shows the product features in form of mind map diagram.

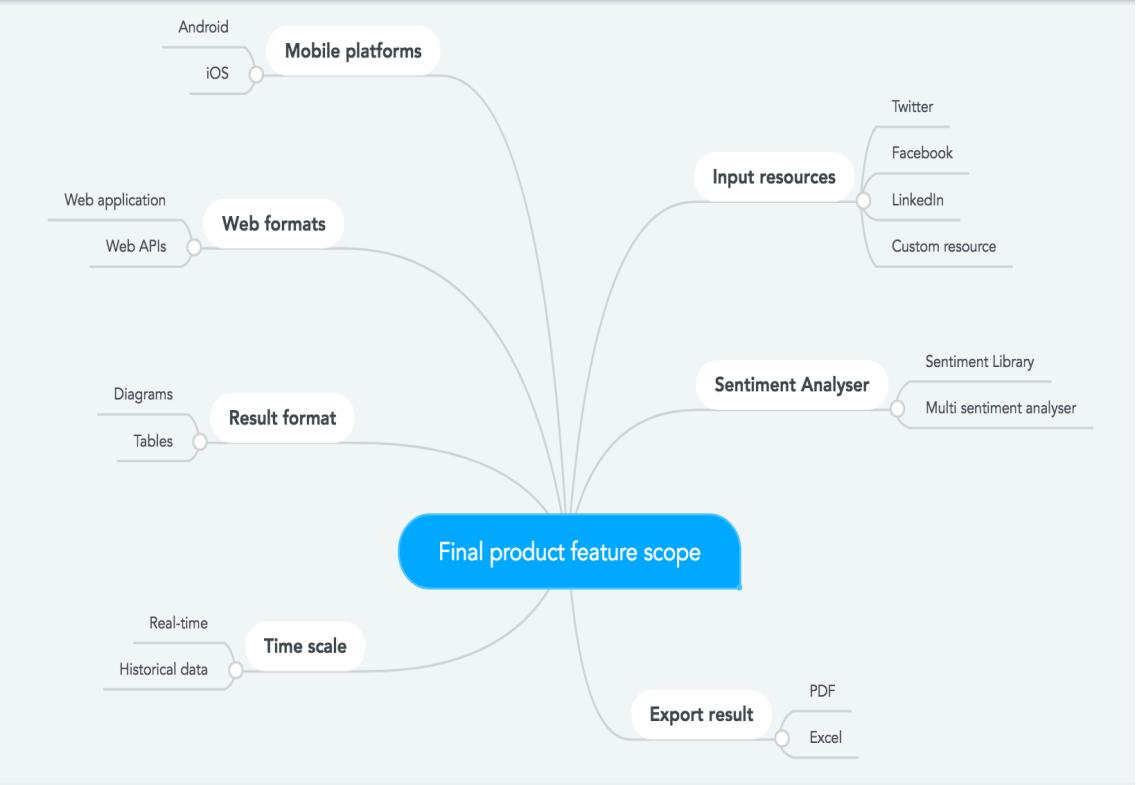


FIGURE 1. The final application product features

Input resource involves different resources for gathering data which includes Twitter, Facebook, LinkedIn data. Additionally, it is possible to gather data from some other resources such as microblogs or other social medias. Gathering data from other resources depends on the context of data. As an example, it is possible to collect the review data about a special product from Tumblr which is a kind of microblog website. The free version of application just get the data from Twitter and covering other resources is considered as a feature for paid users.

Timescale refers to the period which we want to know user's sentiment. In the free version of the application, the data is only available for real-time data and users can see the results in available formats, however, in paid version, it is possible to access previous data in different formats such as a diagram, e.g. How the sentiment of a topic or keywords has changed during last six months.

Sentiment analyser refers to technologies which are used to get the sentiments and sentiment scores. In order to get sentiment and sentiment score of a keyword, we can use 3 different choices or a mixture of different choices. These three choices include

implementing our own sentiment analyser libraries, using other available libraries and using third party services. Using own provided library is a good option if we are going to provide the service for the Finnish language since this service at the time of writing the thesis is not available. However, it requires in-depth knowledge related to working with data mining algorithms. This study just covers very basic introduction about implementing a sentiment analyser, however implementing the library is out of the scope of this study. Based on the backend implementation technology, there are open source and non-open source libraries which could be utilised for our application. As an example, Sentiment library is an open source library available for Node.js applications. Finally, we can use third party services to get sentiment and sentiment score for texts. These services are mostly available in form of APIs, first, we need to post our text to API and then get the result from that service. Bitext is a third-party sentiment analyser which provides sentiment analysis service through APIs.

Mobile platform and **web format** are related to how we present the results and how we export the results. The result of sentiment analysis could be shown as a web application or mobile application. Moreover, the result could be provided as web services like APIs. The result could be shown in form of different type of charts and diagrams. Additionally, the result could be exported in form of PDF or Excel files to the users. Making the decision about which items would be free or for paid users is part of the business plan of the Unseen company and because of that, we can't provide a clear distinction based on paid and non-paid item in this study.

Since the output of the research is going to be used by the Unseen company, the business side of application mostly is relevant to Unseen company business plan. Thus, most of the study concentrate on the implementation of developed application and then required knowledge to provide the final version of the real application. In each chapter depend on the context the available services in the market and how to get benefit from them will be explained.

The implementation of the application is divided to three tier. In fact, based on three-tier architecture, the product's implementation could be divided into three major categories. Presentation tier, logic tier, and data tier. The presentation tier is mostly concern with the way we present data to final users. In logic tier, we mostly are dealing with

implement the application in the backend and finally, the data tier is dealing with database side.

1.3 The scope of work

The development work includes design and implementation of working product with minimum features. The research side of the project, on the other hand, covers all knowledge required to implement desired features of the working project. Since the whole product had a fairly big scope, it was not possible to cover all product's features in this study and at this stage, the scope of work was diminished to a light subset of product's features. The scope of research for this study, based on the agreement between author and company covers only the required knowledge to implement the selected subset of features.

Twitter is the only social media that was considered as the data source. Moreover, between different techniques to get data from Twitter, only get trend/place and get tweet/search was considered as the way to get the data from Twitter. Sentiment analysis also at this step was done in only English language and it doesn't cover the Finnish language. Additionally, between different libraries for analysing the sentiment of the texts, only Sentiment library has been used. A web application with minimum features was implemented which displays the result of sentiment analysis in form of pie chart.

2 PRODUCT FEATURES

2.1 Sentiment analyser (single or multiple)

As it was mentioned earlier sentiment analysis or opinion mining is a field of science which aims to understand user's opinion about a special topic in the range of positive, negative or neutral. Sentiment analysis is one of the text mining applications.

Text mining refers to the process of extract interesting patterns or knowledge from unstructured text documents. Text mining has many different applications which include security applications, biomedical applications, business applications, marketing applications and social media applications. In social media applications, text mining is looking for extract opinions, emotions and sentiments. Sentiment analysis by itself has many issues which should be addressed, however as the final purpose of this study is making a functional application we don't want to focus on sentiment analysis techniques and sentiment analysis patterns. Instead of those, we will check libraries or API or third-party services which provide the result of sentiment analysis from the texts.

In future, we need to provide our own sentiment analysis library in order to analyse the Finnish language texts. In this chapter, we will try to focus on one of the application's features which are related to increasing accuracy of sentiment analysis for the current version of the application. We just use one library, however in the final version of the application, in order to have the more accurate result, we can use more than one sentiment analyser option. As we mentioned earlier, this option could be selected from the open sources available library or our own developed Library or using third party APIs.

We are not going to introduce all available options related to do sentiment analysis but by considering that getting accurate results is one of the most important features of the application, especially for the premium version, we should check some of those options. First, we are going to check Node.JS libraries and then we will check two other options which are open source and non-open source APIs.

2.1.1 Sentiment

Since our application is a Node.JS application, the libraries which we can use to get sentiment results are limited to available libraries in NPM (Node Package Manager). There are many different libraries available for other frameworks as well, which is out of the scope of this study.

Sentiment has been selected for development of this project. This library is open source and it is available on NPM. The Sentiment is a Node.js module that uses the AFINN-165 wordlist and Emoji Sentiment Ranking to perform sentiment analysis on arbitrary blocks of input text. Sentiment also has the ability to append and overwrite word/value pairs from the AFINN word list. AFINN is a list of English words rated for valence with an integer between minus five (negative) and plus five (positive). The words have been manually labelled by Finn Årup Nielsen in 2009-2011.

There are some features which make Sentiment package a good option. First, being open-source could be argued as a positive or negative point of one product, but as sentiment analysis is a growing field and needs to be updated, being open-source provide this opportunity to benefit from the strong and good community. Also, as it is mentioned in its documents, based on benchmarks this service is about twice as fast in comparison with alternative solutions.

2.1.2 Text processing

The text-processing.com API is a simple JSON over HTTP web service for text mining and natural language processing. It is currently free and open for public use without authentication. To analyse the sentiment of some text, should do an HTTP POST to http://text-processing.com/api/sentiment/ with form encoded data contain the text you want to analyse. A JSON object response with two attributes as the response will be returned. Label and probability.

The label is either **pos** if the text is determined to be positive, **neg** if the text is negative, or neutral if the text is neither pos nor neg. The probability is an object that contains the probability for each label. The neg and pos will add up to 1, while neutral is standalone.

If the neutral value is greater than 0.5 then the label will be neutral. Otherwise, the label will be pos or neg, whichever has the greater probability.

As we can see in picture 2, we have three different type of texts. For the first text 'great', if we check the probability value, we can see the value for a positive probability is more than others and it's around 0.7 and it is labelled as positive. For the second text 'terrible', as we can see the negative value is around 0.7 and it is labelled as negative. Finally, in the third example for text 'hi friend', the value for neutral is more than others and it is more than 0.7 and it is labelled as neutral.
A good point of using this service is that it will return us a value where we can use in our final application to show the degree of being positive or negative to our users.

```
$ curl -d "text=great" http://text-processing.com/api/sentiment/
{
        "probability": {
                "neg": 0.39680315784838732,
                "neutral": 0.28207586364297021,
                "pos": 0.60319684215161262
        },
        "label": "pos"
}

$ curl -d "text=terrible" http://text-processing.com/api/sentiment/
{
        "probability": {
                "neg": 0.68846305481785608,
                "neutral": 0.38637609994709854,
                "pos": 0.31153694518214375
        },
        "label": "neg"
}

$ curl -d "text=hi friend" http://text-processing.com/api/sentiment/
{
        "probability": {
                "neg": 0.59797768649386562,
                "neutral": 0.74939503025120124,
                "pos": 0.40202231350613421
        },
        "label": "neutral"
}
```

PICTURE 2. An example of using text-processing service for sentiment analysis

2.1.3 Bitext and sentiment analysis domain levels

In order to understand the main reason for selecting Bitext as an option, first, we need to know about sentiment analysis domain levels.

Sentiment analysis can be considered as a classification process. There are three main classification levels in sentiment analysis. Document-level, sentence-level and aspect-level. Document-level aims to classify an opinion document as expressing a positive or

negative sentiment. It considers the whole document a basic information unit. Sentence-level sentiment analysis aims to classify sentiment expressed in each sentence. The first step is to identify whether the sentence is subjective or objective. If the sentence is subjective, sentence-level will determine whether the sentence expresses positive or negative opinions. However, there is no fundamental difference between document and sentence level classifications because sentences are just short documents. Classifying text at the document level or at the sentence level does not provide the necessary detail needed opinions on all aspects of the entity which is needed in many applications. To obtain these details, we need to go to the aspect level. Aspect-level sentiment analysis aims to classify the sentiment with respect to the specific aspects of entities. The first step is to identify the entities and their aspects. The opinion holders can give different opinions on different aspects of the same entity like this sentence ‘‘This Android I've got is awful, my iPhone was so sleek, I loved it!’’

One thing about this classification is that we can't completely separate sentiment classification levels from each other if we are looking for the best result, one reason is that when we are analysing text data to understand attitudes behind that, we are dealing with so many different situations which are unique and it needs to be handled in its own way. As an example when we are going to analyse a document as a whole, we have to take this fact into account that a document consists of several opinions which might be close or not.

Bitext, as one of solution provider in the market for sentiment analysis of data, divide a text from document level to aspect level analysis. As an example, we want to analyse the mentioned document. “This Android I've got is awful, my iPhone was so sleek, I loved it!”. This text includes 3 opinions, which has different sentiment or polarity and strength, one solution for this type of sentences is that after separate text based on the opinion, we have to measure a score for each sentence and then we can return an overall score. If we check our example again, after separate that to the different opinion we can have something like the picture 3. As we can see this text include three sentences. In the first sentence, 'awful' is the opinion word with negative sentiment. Bitext for first sentence return -4.0. Then for the second sentence, it returns 3.0 since 'sleek' is the opinion word with positive polarity. For the last sentence also, Bitext returns 3.0 with opinion word loved and positive polarity. Finally, it sums up all values with their polarities and returns value 2.0 as overall value for the text.

```
"text":"this Android I've got is awful, my iPhone was so sleek,
I loved it",
"global_value":2.000000,
"details": [
{
"valuables": "Android",
"valuers": "awful",
"value":-4.000000
},
{
"valuables": "iPhone",
"valuers": "sleek",
"value":3.000000
},
{
"valuables": "iPhone",
"valuers": "loved",
"value":3.000000
}
```


#opinion 1
#sentiment topic 1
#sentiment text 1

#opinion 2
#sentiment topic 2
#sentiment text 2

#opinion 3
#sentiment topic 3
#sentiment text 3

PICTURE 3. The result of sentiment analysis returned by Bitext API

It is good to mention that although Bitext returns these values based on its algorithms and techniques, other sentiment analysers could return different values, especially this could be important when we just want to show the overall value for a text or in our case, a Tweet. Thus, it would be possible to have different values and even different polarity depends on the sentiment analyser and this could show us the value of using multiple sentiment analysers as we return the average of more than one analysis.

2.2 Input resources

Selection of data source to conduct the sentiment analysis plays a significant role. Social media platforms as the data sources are broadly categorised into three general categories. Blogs, micro-blogging sites, and review site. Among all categories, a micro-blogging site such as Twitter has gained higher popularity due to its limited strength of

the content and publically availability of data. From the following statistics of the Twitter growth rate, it's evident to use Twitter as the data source for sentiment analysis. Approximately 6,000 tweets are tweeted on Twitter on per second basis. It resembles 350,000 tweets sent per minute and 500 million tweets per day. That makes it around 200 billion tweets per year. In Twitter's history, the number of Tweets increased from 5,000 tweets per day in 2007 to 500,000,000 tweets per day in 2013, that is approximately a six orders of magnitude. At the intermediate stages, it has the statistics of 300,000 tweets per day in 2008, 2.5 million tweets per day in 2009, 35 million tweets per day in 2010, 200 million tweets per day in 2011. And 340 million tweets per day six years after the emergence of Twitter i.e. on March 21, 2012. These statistics conclude the use of Twitter for our research.

As per the recent work, the studies carry out on Twitter data are in the field of healthcare, marketing, politics, advertising market, athletics etc. Analysis techniques used in these studies include qualitative content analysis, network or graph analysis, linguistic or psycholinguistic analysis, word clouds and histograms. In addition, Twitter has been voted as the most promising source for the studies such as community or influence detection, topic discovery, market and business predictions, recommendation systems and tweet classification.

The message posted on Twitter is called Tweet, which is limited to 140 characters. Tweets are generally composed of one of the followings. Text, links, emoticons, and images. A six seconds video is even added as a Tweet component in 2012. Based on these components the mining is applied to classify text, links, images, emoji or emoticons and even videos. The Tweets contains three notations including hashtags (#), retweets (RT) and account Id (@).

2.2.1 Twitter and Tweet

Based on what is mentioned in Wikipedia, Twitter is an online news and social networking service where users post and interact with messages, "Tweets", restricted to 140 characters. Registered users can post Tweets, but those who are unregistered can only read them. Users access Twitter through its website interface, SMS or a mobile device app.

2.2.2 Hashtags

A hashtag is a type of label or metadata tag used on social network and microblogging services which make it easier for users to find messages with a specific theme or content. Users create and use hashtags by placing the hash character # (also known as the number sign or pound sign) in front of a word or unspaced phrase, either in the main text of a message or at the end. Searching for that hashtag will yield each message that has been tagged with it. A hashtag archive is consequently collected into a single stream under the same hashtag. For example, on the photo-sharing service Instagram, the hashtag #bluesky allows users to find all the posts that have been tagged using that hashtag. Because of its widespread use, the hashtag was added to the Oxford English Dictionary in June 2014. The term hashtag can also refer to the hash symbol itself when used in the context of a hashtag.

2.2.3 Trending topic

A word, phrase or topic that is mentioned at a greater rate than others is said to be a "trending topic". Trending topics become popular either through a concerted effort by users or because of an event that prompts people to talk about a specific topic. These topics help Twitter and their users to understand what is happening in the world and what people's opinions are about it.

The Twitter web interface displays a list of trending topics on a sidebar on the home page, along with sponsored content. Other ways to access to trending topic is using Twitter API which returns the trending topics.

2.3 Result format

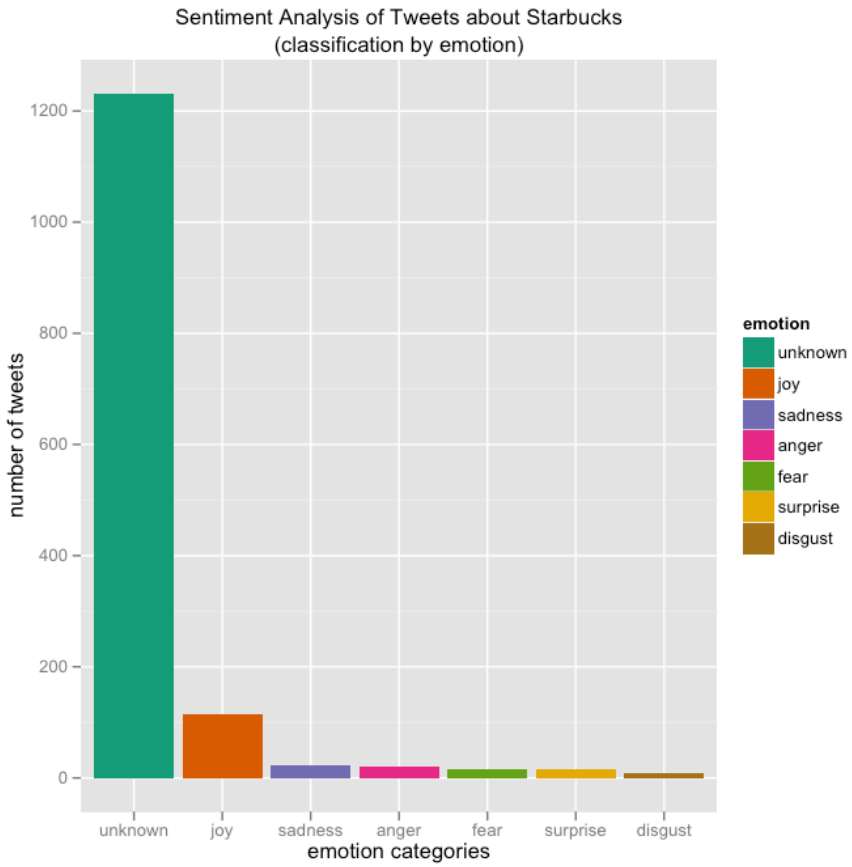
One of the most important features of the application is about how to present the data to the users. After we decide which version of application we are going to support, the

next step will be how to present the data to users. The result of sentiment analysis could be mostly published in two forms of diagrams and tables. However, these two suggested forms doesn't mean the results will be limited to only these two types, but at the time of writing the thesis, these two ways were the most concrete possible way. In order to show the result as a diagram, we first have to check which ways are possible to show the results.

2.3.1 Context classification

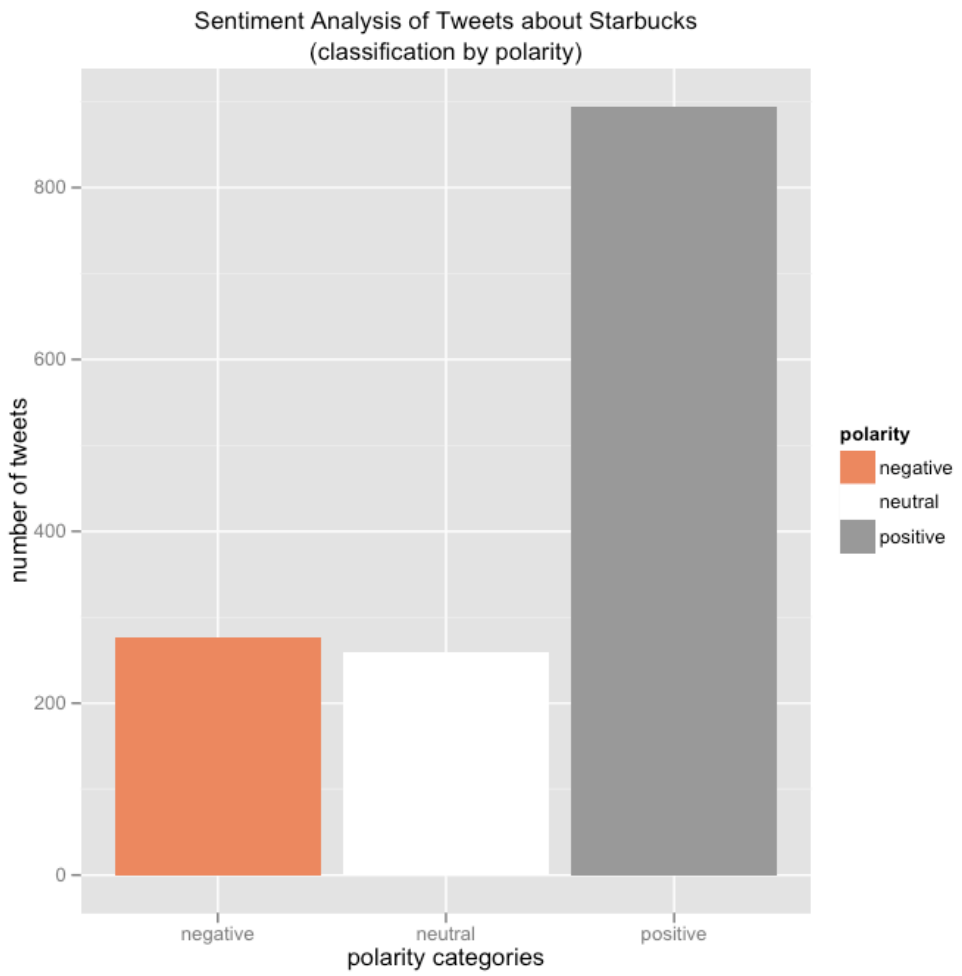
It is important to understand which context we want to display, the way which we select to show data, affect the type of data we are analysing and affect on our technology selection. We can classify the output in general into two groups. Classify emotion and classify polarity.

By classifying emotion, we can classify the text in a different type of emotions such as anger, disgust, fear, joy, sadness and surprise. It is important to mention that this classification could be done through different algorithms and techniques. As an example, picture 4 shows the result of sentiment analysis of Tweets about Starbuck by users in form of classifying emotion.



Picture 4. Show the result of sentiment analysis based on emotional classification.

Classification based on polarity let us categorise text based on positive or negative sentiment. This classification also could be done through any kind of sentiment analysis techniques. Picture 5 show the result of sentiment analysis of Tweets about Starbucks in form of classifying polarity.



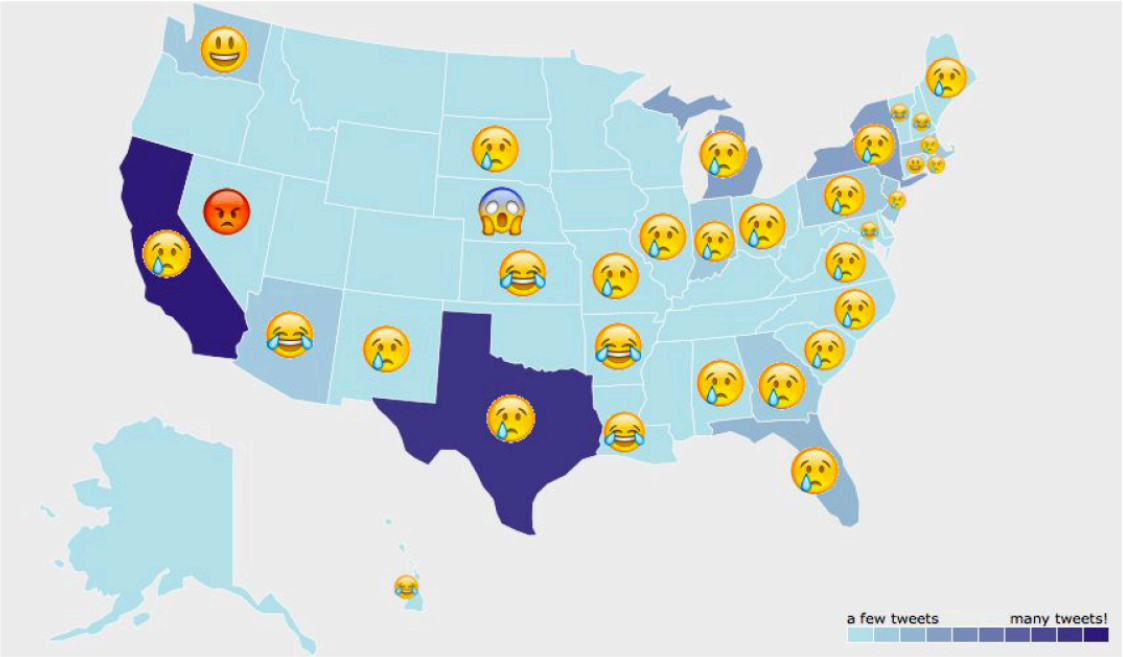
Picture 5.Show the result of sentiment analysis based on polarity classification.

2.3.2 Integrity

Another thing which in presenting data is important, finding a way which user can get a date in most effective and efficient way. We aim to analyse some of the possible ways to display data. As it was mentioned, it is really important to have a good analyse which is dependent on the way we use for calculate the sentiment of text and also process of preparing our date set, however in order to get the most benefit of this result we have to gather all result in one place. As an example, we check one of the most popular applications for Twitter sentiment analysis which is political issues. In the following section, we will check different idea to present sentiment analysis resulted about political topics which could be used for other applications as well.

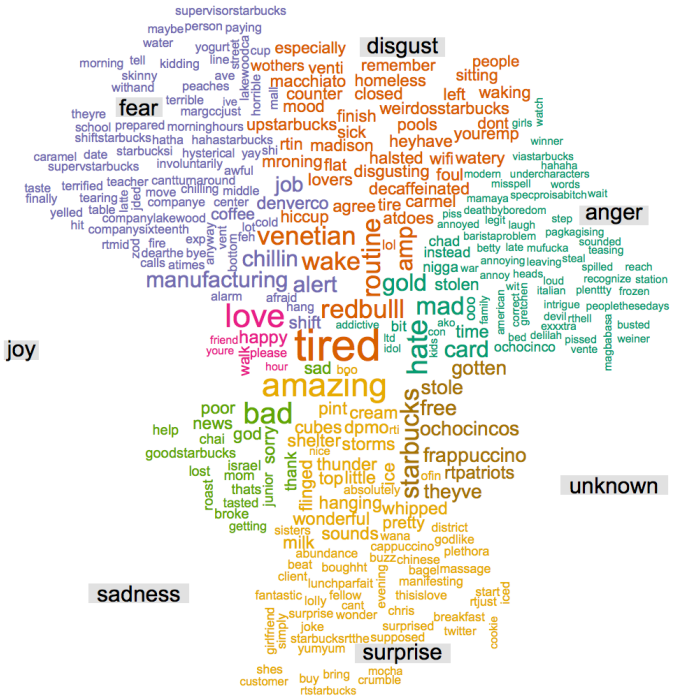
Delenn and Anna in their research which has been done about presidential candidates 2016 in the US analyse tweets includes emoji and analyse the sentiment of that, they

tried to show the result of sentiment analysis on the map. Using map is one of the most effective ways to show the result of sentiment analysis. The map could be used for either emotion classify or polarity classification. Picture 6 shows people reaction about topic 'Trump' based on user's tweets. As we can see, each state has been labelled with one emoji which shows that state attitude toward selected topic. These emojis could show feelings such as happy, angry, surprised, etc.



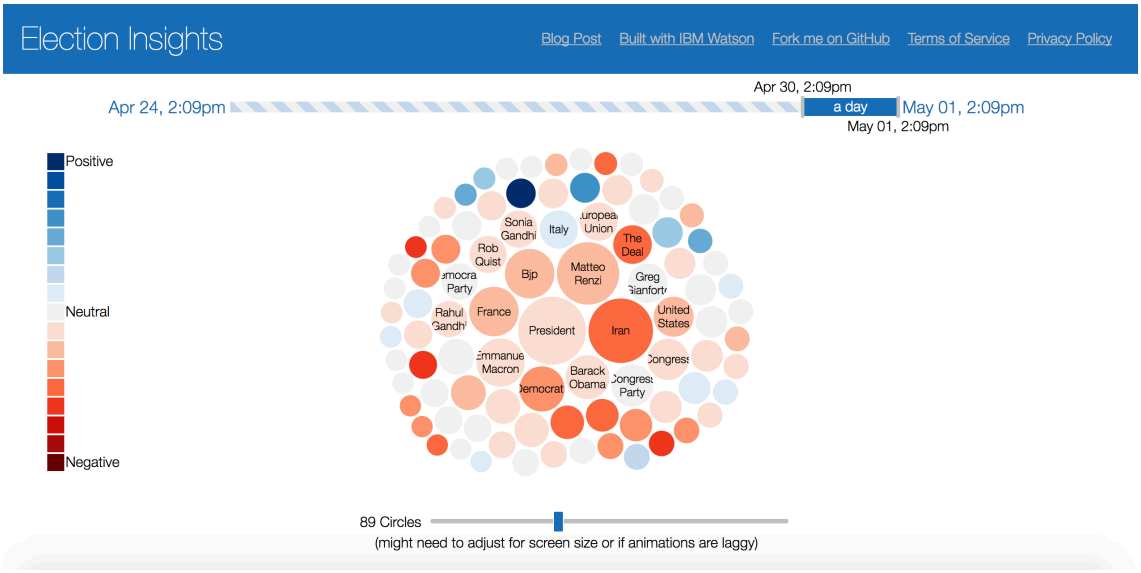
Picture 6. The result of sentiment analysis about Trump topic during the presidential election 2016 in the US.

Additionally, one of the effective and popular ways to show the result of sentiment analysis is using word cloud. The idea about word cloud is a visual representation of text data, typically used to depict keyword metadata (tags) on websites or to visualise free form text. Tags are usually single words, and the importance of each tag is shown with font size or colour. This format is useful for quickly perceiving the most prominent terms and for locating a term alphabetically to determine its relative prominence. When used as website navigation aids, the terms are hyperlinked to items associated with the tag. Picture 7 shows word cloud result for a topic. As we can see, the result of analysis has been categorised based on emotional classifications in different colours, also the size of each emotion is relevant to its score from analysing results. For example based on picture 7, the most powerful feelings of the analysis are tired which belong to disgust category.



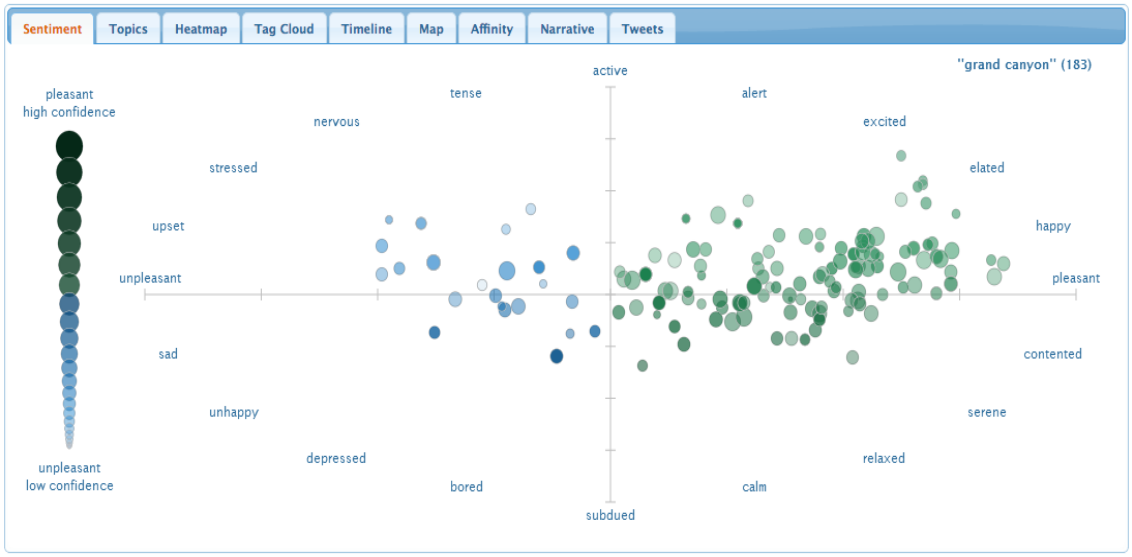
Picture 7. Word cloud for a topic to show emotions from sentiment analysis.

Election insight is an example of presenting the result in another way. This service uses IBM Watson for their analysis and presents the idea by circles. For their front-end implementation, they have used React.js. This project is open-source and available on GitHub.



Picture 8. Election insight, another example of presenting sentiment analysis result.

Finally, we can present the result of sentiment analysis, on a chart. We can define different type of chart such as pie chart and depend on our context select most suitable and populate the date to our chart. Picture 9 shows Sentiment Viz, a web application which provides sentiment analysis result in the different form of diagrams. The chart below shows classify emotion for a topic on the chart.



Picture 9. Sentiment Viz provides the result of sentiment analysis about selected topic in the different form of charts.

2.4 Application format

It is completely crucial to select a good framework for developing front-end side. There are many different criteria which effect on selection. First, we have to decide about our target, which is mobile users, desktop or anything else. Sentiment analysis is kind of services which is working underneath, it means that we will need this service as a consequence of another service. As an example, I need the result of sentiment analysis of presidential debates when I care about politics in advance. As a result making a desktop application has more priority than a mobile application. Another fact is that users normally use the mobile application in order to access to specific data rather than checking analysis. However, one of best places to put the result in mobile application will be notification centre or as a widget in the mobile application. Using the result in notification

centre has some benefits like running the app in the background make the result updated as a background task and the user always accesses to update data. Additionally, we can use this style on other devices connected to the phone such as smart watch which is increasingly using by users.

Picture 10. Show the result of sentiment analysis to users in the notification panel.

If we are planning to implement the application for mobile devices and smart tools we have to decide which frameworks we are going to implement our application. Currently, we have 3 main mobile operating system, Windows, Android and iOS. There are some facts regarding select frameworks. The first factor which is most important thing in selecting framework is popularity. As we can see from figure 2, based on published report by Gartner in the fourth quarter of 2016, 81.7% of market share is for Android, 17.9% is for iOS, 0.3% for Windows 10 mobile and 0.1 % for all other platforms. By considering this fact it is better to implementing mobile application starts with Android and then IOS. However, we have to take into account that for implementing the application for Android we have to deal with some issues like having many different types of devices which need to be planned properly.

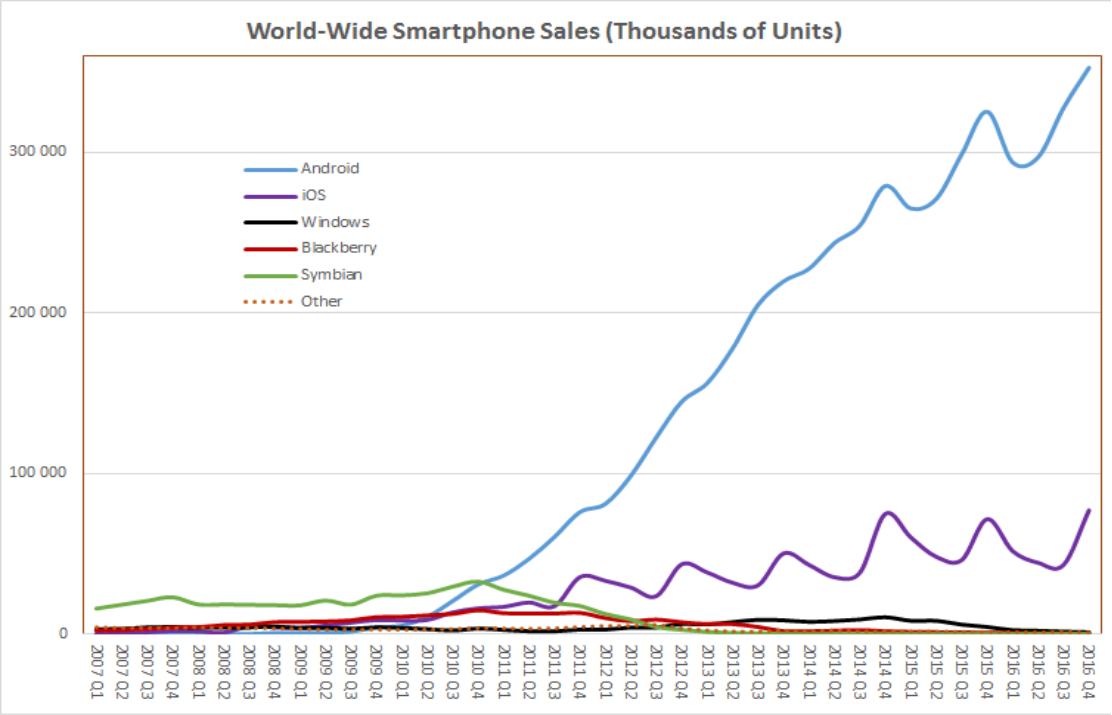


Figure 2. worldwide smartphone sale report

Although mobile device usage is growing faster and gaining more share of user access to the internet, however, the desktop application is still very popular. Based on the recently published result by W3tech, it seems that JavaScript frameworks are the most popular front-end frameworks, however between different JavaScript's frameworks, React.js and angular have the most share in comparison with others. In the next chapter, the most proper available frameworks for our implementation will be analysed.

3 PLANNING AND TECHNOLOGY SELECTION

This chapter aims to explain the planning and technology selection of the project. First, we have to divide the scope of the project. The final version of the project has some features which need different requirement than the light version of the application. The technologies which have been used to implement the light version of the application will be explained in detail in next chapters. These chapters include how to communicate with Twitter APIs and how to benefit from Twitter data, then explain the backend technologies and implementation and finally, front-end technologies and implementation will be explained.

On the other hand, for the final version of the application, we have to select technologies which first could support the desired features and then be able to support future upcoming features. As we can see from figure 3, the final version of the application in very abstract level include different parts. After we decide about which social media and resources we want to support, we need to define the ways to communicate with that. For reasons which have been mentioned in different parts of the thesis, Twitter is the only target for us at this level.

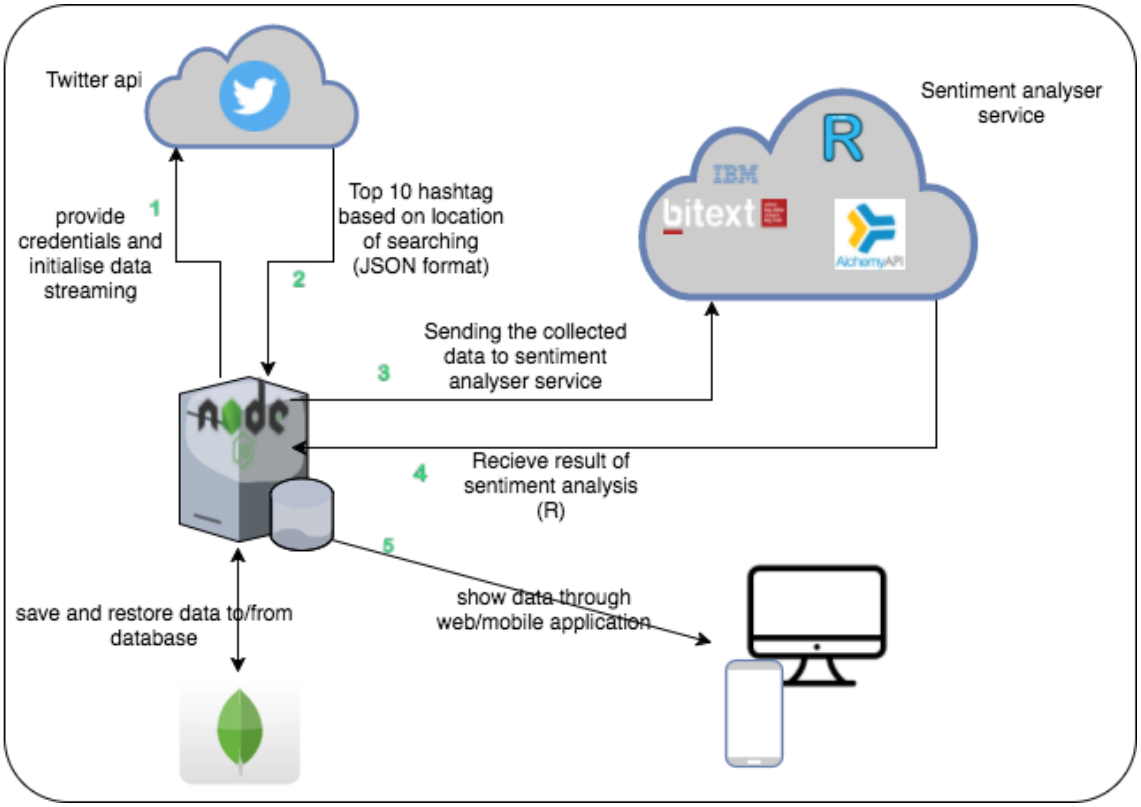


Figure 3. Solution architecture

After selecting the main source, in next step, we need to plan and select technology for back-end and data store. Node.JS for backend and MongoDB for the data store, are the options which have been selected for developing the final version of the application. The main reason for the selecting Node.JS is access to many different libraries, easy to develop, strong community and scalability. Additionally, MongoDB has a good performance of working with big data. These technologies will be explained in more detail in next chapters.

After selecting backend and data-storage technologies, we need to define the ways we are going to get sentiment analysis. This data has been explained in detail in chapter 'Working with Twitter data'.

Finally, for front-end development, we have to take into account that we need different technologies for the light version of the application in comparison with the final version of the application. Two front-end frameworks are proposed for implementing the final version of the application. React.JS and Angular.

React (sometimes styled React.js or ReactJS) is an open-source JavaScript library for

building user interfaces. It is maintained by Facebook, Instagram and a community of individual developers and corporations. According to JavaScript analytics service Libscore, React is currently being used on the websites of Netflix, Imgur, Bleacher Report, Feedly, Airbnb, SeatGeek, HelloSign, Walmart, and others.

React allows developers to create large web applications that use data which can change over time, without reloading the page. Its main goal is to be fast, simple and scalable. React processes only user interface in applications. This corresponds to View in the Model-View-Controller (MVC) template and can be used in combination with other JavaScript libraries or frameworks in MVC, such as AngularJS. It can also be used with React based on add-ons to take care of without the user interface parts of web developing.

AngularJS (commonly referred to as "Angular.js") is a JavaScript-based open-source front-end web application framework mainly maintained by Google and by a community of individuals and corporations to address many of the challenges encountered in developing single-page applications. The JavaScript components complement Apache Cordova, the framework used for developing cross-platform mobile apps. It aims to simplify both the development and the testing of such applications by providing a framework for client-side model–view–controller (MVC) and Model–View–View-Model (MVVM) architectures, along with components commonly used in rich Internet applications. In 2014, the original AngularJS team began working on Angular (Application Platform).

The AngularJS framework works by first reading the HTML page, which has embedded into it additional custom tag attributes. Angular interprets those attributes as directives to bind input or output parts of the page to a model that is represented by standard JavaScript variables. The values of those JavaScript variables can be manually set within the code, or retrieved from static or dynamic JSON resources.

As both of these frameworks, provide different features we have to select the technology based on required features. The technology selection, especially for front-end framework, also depends on the company strategies. As most of the software solutions provided by the Unseen company have been implemented by Angular Framework, probably the front-end framework will be Angular. For the light version of the

application, the D3.js has been selected. The details related to D3 will be explained in chapter ‘Front-end design and architecture’.

4 WORKING WITH TWITTER DATA

There are three possible ways to collect data from Tweets (table 1). For this study, only Twitter APIs were selected as the way to get data from Twitter.

Table 1. Three possible ways to get data from tweets (Desai & Mehta 2016, 3)

Twitter APIs	Twitter REST APIs and Twitter Stream APIs
Data repositories	UCI, Friendster, Kdnuggets, and SNAP
Automated tools	Radian6, Sysomos, Simplify360, Lithium and non-premium tools such as Keyhole, Topsy, Tagboard and SocialMention.

4.1 Twitter APIs

In general, Twitter APIs includes two main groups. REST APIs and Streaming APIs. The REST APIs provides programmatic access to read and write Twitter data such as author a new Tweet, read author profile and follower data, and more. The REST API identifies Twitter applications and users using OAuth, responses are available in JSON format.

On the other hand, the Streaming APIs continuously deliver new responses to REST API queries over a long-lived HTTP connection. Receive updates on the latest Tweets matching a search query, stay in sync with user profile updates, and such. The Streaming APIs give developers low latency access to Twitter's global stream of Tweet data.

4.1.1 REST APIs

As we have already mentioned Twitter provides access to data based on the different requirement in form of REST APIs. These APIs are defined and categorised for different purposes and could be accessed from dev.twitter.com. what we should do is to find most suitable APIs for our purpose and get the data from those APIs. For each API, there are suitable documentations available in dev.twiiter.com. These documentations

include the name of API, an explanation about that, an example of how to use the API and an example of the response of calling the API. (Picture 11)

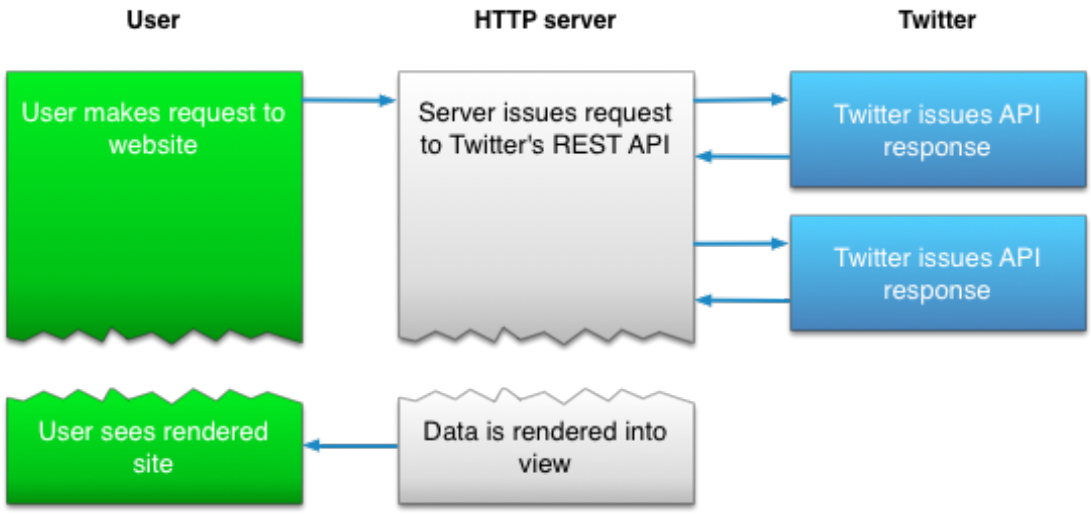
PICTURE 11. The Twitter search API Info page include description and example and related links.

4.1.2 Streaming APIs

The Streaming APIs give developers low latency access to Twitter's global stream of Tweet data. A streaming client will be pushed messages indicating Tweets and other events have occurred, without any of the overhead associated with polling a REST end-point.

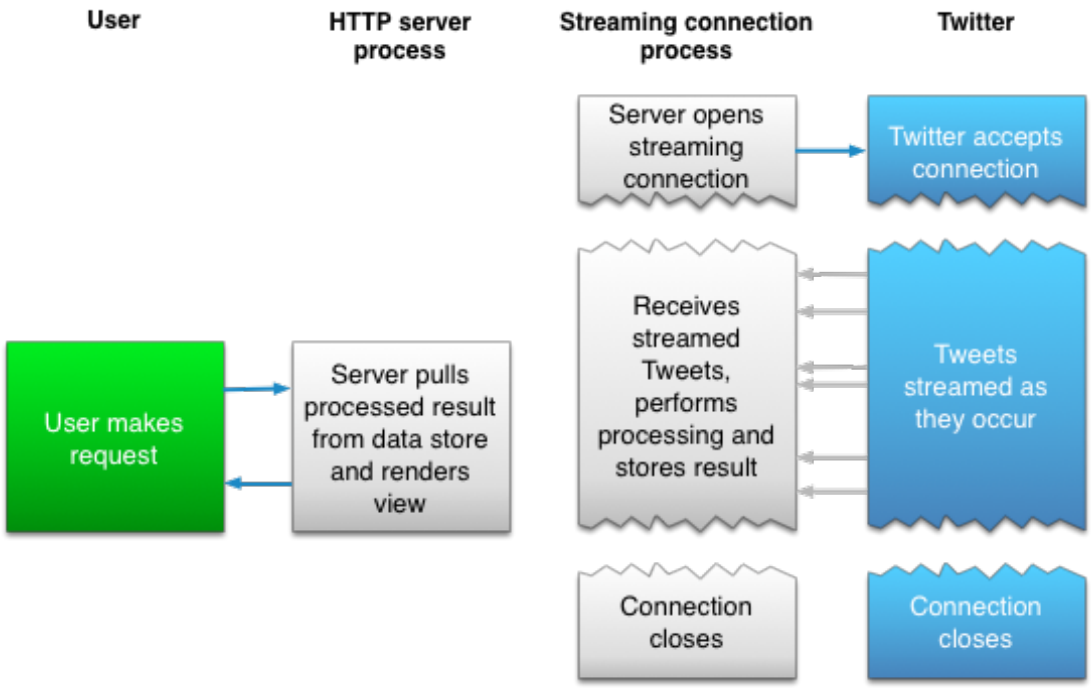
4.1.3 Difference between REST API and Streaming API

Connecting to the streaming API requires keeping a persistent HTTP connection open. In many cases, this involves thinking about the application differently than if the application were interacting with the REST API. For example, consider a web application which accepts user requests, makes one or more requests to Twitter's API, then formats and prints the result to the user, as a response to the user's initial request.



PICTURE 12. The structure how REST API works.

An app which connects to the Streaming APIs will not be able to establish a connection in response to a user request, as shown. Instead, the code for maintaining the Streaming connection is typically run in a process separate from the process which handles HTTP requests.



PICTURE 13. The structure how streaming API works

The streaming process gets the input Tweets and performs any parsing, filtering, and/or aggregation needed before storing the result to a data store. The HTTP handling process queries the data store for results in response to user requests. While this model is more complex than the REST example, the benefits from having a real-time stream of Tweet data make the integration worthwhile for many types of applications.

4.2 Getting data from Twitter API

What we need in this step is getting our required date from Twitter and then make some calculation and finally store the result in the database. In order to cover the whole of these, we divide these into different steps and then check about each in detail. In the first step, we need to read data from Twitter. This data should be 10 current hashtag trends and will be updated each 15 second.

Twitter by itself provides a wide variety of different APIs for developers. It is possible to access to all the data through dev.twitter.com. First, we need to get Trending topics based on our location. The API which we need to work with that is one of the Twitter REST APIs. GET trends/place returns the top 50 trending topics for a specific location if trending information is available for that location, otherwise, the trending data should be determined by other ways. The location data is giving in WOEID format which is an abbreviation of Where On Earth ID.

WOEID is a unique 32-bit reference identifier, originally defined by GeoPlanet and now assigned by Yahoo!, that identifies any feature on Earth. In 2009, Yahoo! released Geo-Planet's WOEID data to the public, with the last release on 1 June 2012, after which Yahoo! decided to cease making the data downloadable until they determine a better way to surface the data as a part of the service.

- Tampere, Tampereen, Pirkanmaa, Finland (573760)

PICTURE 14. WOEID example for Tampere, as we can see the WOEID value for Tampere is 573760.

The response is an array of "trend" objects that encode the name of the trending topic. This information is cached for 5 minutes. Requesting more frequently than that will not return any more data, and will count against your rate limit usage.

Table 2. Resource information for Get trend/place API

Response format	**JSON**
Requires authentication	Yes
Rate limited	Yes
Requests / 15-min window (user auth)	75
Requests/ 15min window (app auth)	75

Name	Required	Description	Default Value	Example
id	required	The Yahoo! Where On Earth ID of the location to return trending information for. Global information is available by using *1* as the *WOEID* .		*1*
exclude	optional	Setting this equal to *hashtags* will remove all hashtags from the trends list.		

PICTURE 15. Parameters for Get trend/place

As an example, if we send a request with id = 1 it returns most popular trends in the world. the request is same as this:

GET https://api.twitter.com/1.1/trends/place.json?id=1

4.2.1 Authentication

Twitter use OAuth to provide authenticated access to its APIs. There are two main reasons for using OAuth: being secure and being standard. Being secure means that because no need to share an account with third-party applications, so the account will be safe. On the other hand, as it follows the standards, there's a lot of examples and libraries to understand about this topic. In general, this authentication could be in two form. User authentication and application-only authentication.

User authentication is the most common form of resource authentication in Twitter's OAuth 1.0a implementation. A signed request identifies an application's identity in addition to the identity accompanying granted permissions of the end-user the application is making API calls on behalf of, represented by the user's access token.

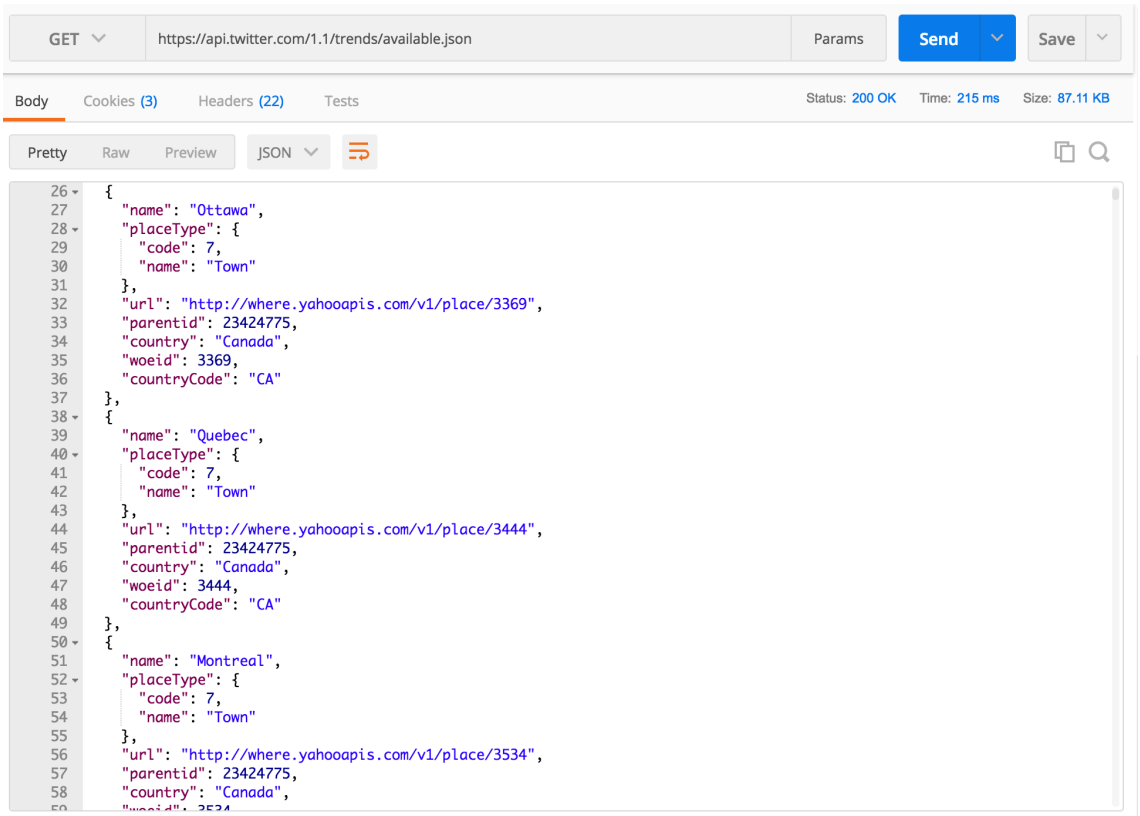
An application-only authentication is a form of authentication where an application makes API requests on its own behalf, without a user context. API calls are still rate limited per API method, but the pool each method draws from belongs to the entire application at large, rather than from a per-user limit. API methods that support this form of authentication will contain two rate limits in their documentation, one that is per user (for application user authentication) and the other is per app (for this form of application-only authentication). Not all API methods support application-only authentication because some methods require a user context (for example, a Tweet can only be created by a logged-in user, so user context is required for that operation).

To get a response from Twitter, we should prepare Consumer key (API key) and Consumer Secret (API secret). First, we have to create an application in Twitter development space and then we can use consumer key and consumer secret to getting a response. We will use Postman to test the response of API.


```
[
  {
    "trends": [
      {
        "name": "#SweetCreature",
        "url": "http://twitter.com/search?q=%23SweetCreature",
        "promoted_content": null,
        "query": "%23SweetCreature",
        "tweet_volume": 184481
      },
      {
        "name": "#HomaidanAlTurki",
        "url": "http://twitter.com/search?q=%23HomaidanAlTurki",
        "promoted_content": null,
        "query": "%23HomaidanAlTurki",
        "tweet_volume": null
      },
      {
        "name": "Bruno Mars",
        "url": "http://twitter.com/search?q=%22Bruno+Mars%22",
        "promoted_content": null,
        "query": "%22Bruno+Mars%22",
        "tweet_volume": 44794
      },
      {
        "name": "#DafBama2017_EXO",
        "url": "http://twitter.com/search?q=%23DafBama2017_EXO",
        "promoted_content": null,
        "query": "%23DafBama2017_EXO",
        "tweet_volume": 710828
      },
      {
        "name": "#2May",
```


PICTURE 16. The response of getting Trends from Twitter

Twitter has provided an API which returns locations that Trends data are available for, The response is an array of “locations” that encode the location’s WOEID and some other human-readable information such as a canonical name and country the location belongs in.



PICTURE 17. An example of a response for Trends/available API. As we can see for available places the API returns its WOEID

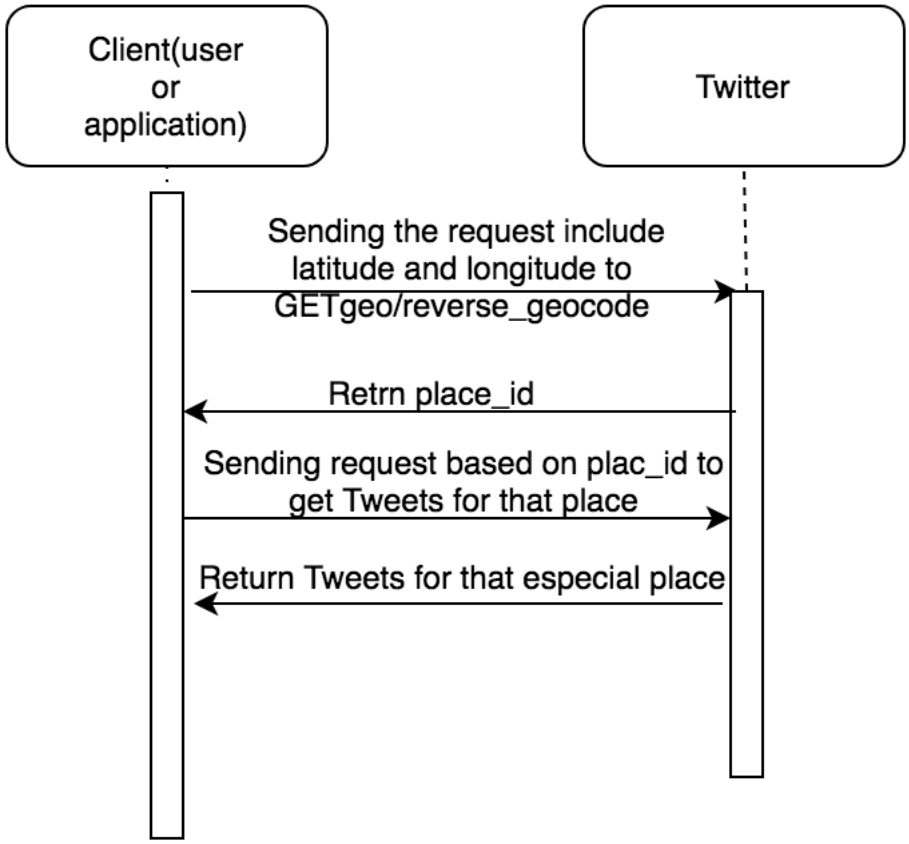
4.2.2 Data types issues and alternatives

One of the problems which are currently we are facing with is that Finland is not in response list and it means that we can't get the trends for Finland through this application. And for this matter, we need to find another way. Because of this issue, we continue with countries which Trend data is available for, however, after doing some research, this possible way is proposed for further steps. Twitter search API and Tweets by Place.

Twitter Search API is part of Twitter's REST API. It allows queries against the indices of recent or popular Tweets and behaves similarly to, but not exactly like the Search feature available in Twitter mobile or web clients, such as Twitter.com search. The Twitter Search API searches against a sampling of recent Tweets published in the past 7 days.

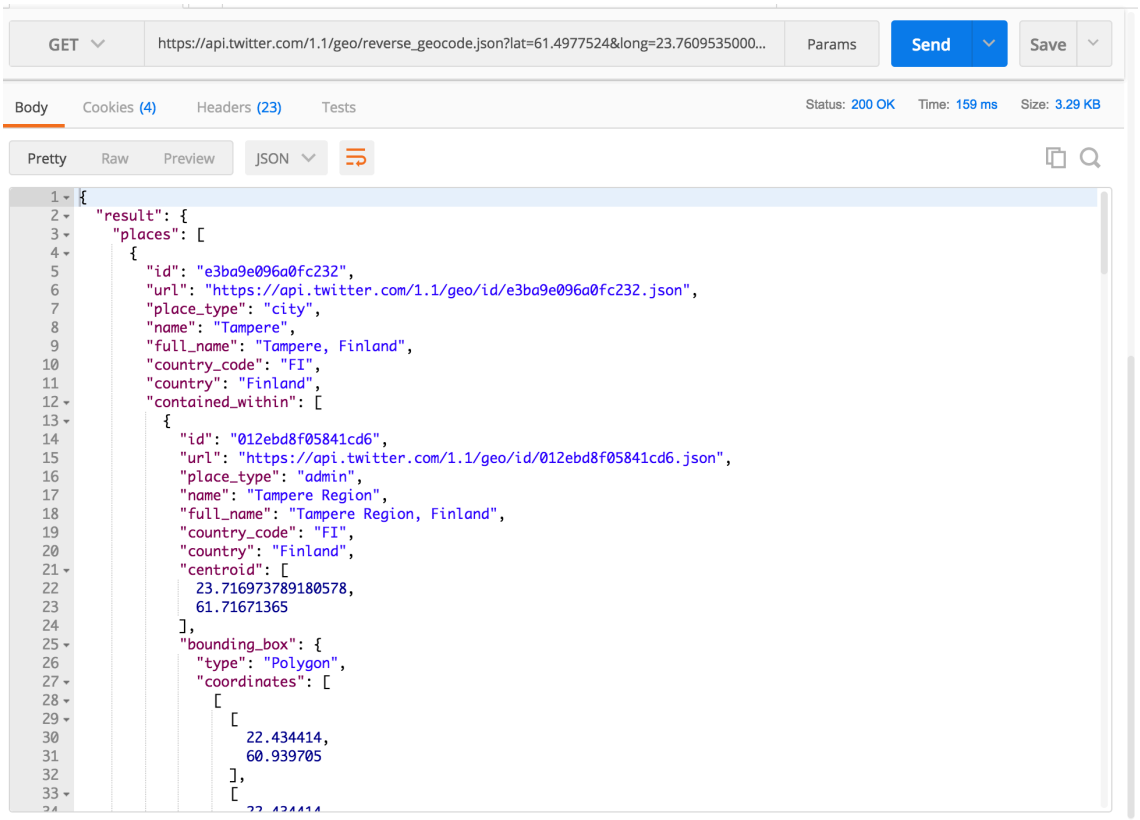
It is possible to search for Tweets about places using the place operator of the Search API. The place operator supports Twitter Place IDs. As this API needs Twitter place id,

we need to provide this ID for that, and in this order, we can use GET geo/id/:place_id API, which returns all the information about a known place. This API, get place_id as input parameter and return all data related to place. In order to get place-id, we have to use another API named GET geo/reverse_geocode. This API get latitude and longitude and return all possible place_ids for that location.



PICTURE 18. Sequence diagram for getting tweets about a particular place.

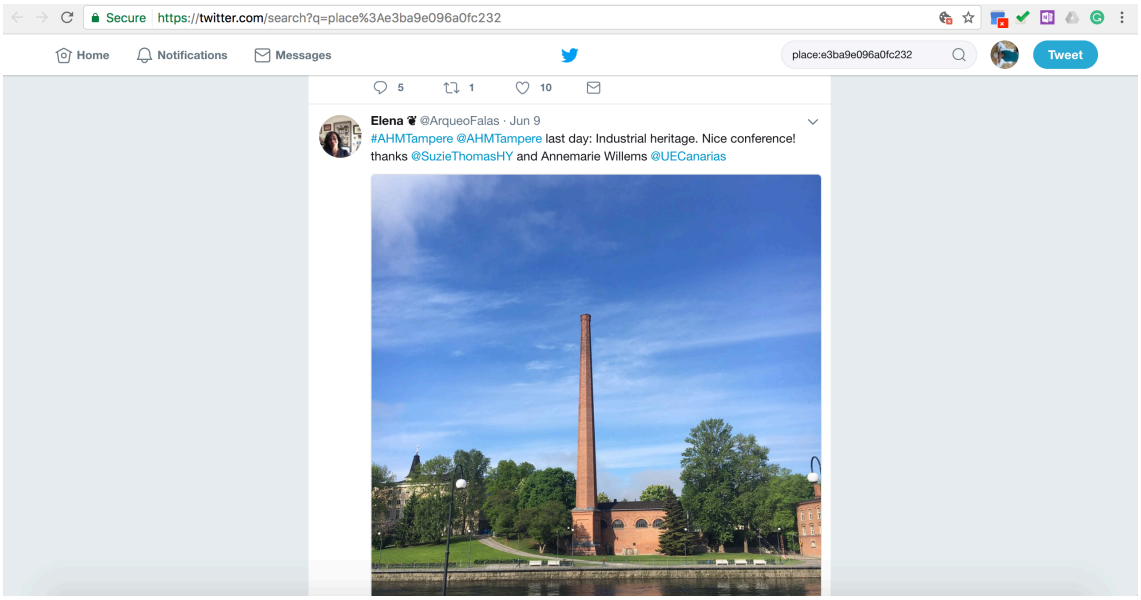
Following we can see how this sequence diagram will works. First, we need to get Place-id for current location through GET geo/reverse_geocode API. GET geo/reverse_geocode API searches for up to 20 places that can be used as a place_id when updating a status. Based on current location, the request will be as below.
https://api.twitter.com/1.1/geo/reverse_geocode.json?lat=61.4977524&long=23.760953500000028



PICTURE 19. Getting response from GET geo/reverse_geocode API related to current location.

As we can see in picture 19 there are many data related to this position which is returned by Twitter API, however, what we require at this stage is the value for place_id and it is e3ba9e096a0fc232. In next step, we will use this place_id search at Tweet by place.

If we put our place_id into https://twitter.com/search?q=place%3Ae3ba9e096a0fc232 and check it in browser, it will return a page include tweets related to that place id, or we can have josn data with this address : https://api.twitter.com/1.1/search/tweets.json?q=place%3Ae3ba9e096a0fc232



PICTURE 20. of search based on location in Browser

5 BACKEND DESIGN AND ARCHITECTURE

Backend side of the application includes running the server on a local machine and then deploy the server on cloud service. For development step we need to run the application on local machine. For deploy the project on cloud we need to evaluate possible option and after selecting an option deploy the result on that. Deployment of project on cloud server is not in scope of the thesis. However, some of possible options will be evaluated. We are going to develop the backend based on Node.JS framework.

Node.JS is an open-source, cross-platform JavaScript runtime environment for developing a diverse variety of server tools and applications. Although Node.js is not a JavaScript framework, many of its basic modules are written in JavaScript, and developers can write new modules in JavaScript. The runtime environment interprets JavaScript using Google's V8 JavaScript engine.

Node.js has an event-driven architecture capable of asynchronous I/O. These design choices aim to optimise throughput and scalability in Web applications with many input/output operations, as well as for real-time Web applications (e.g., real-time communication programs and browser games).

NPM (Node Package Manager) is automatically included when Node.js is installed. NPM consists of a command line client that interacts with a remote registry. It allows users to consume and distribute JavaScript modules that are available on the registry. Packages on the registry are in a CommonJS format and include a metadata file in JSON format. Over 280,000 packages are available on the main NPM registry. The registry has no vetting process for submission, which means that packages found there can be low quality, insecure, or malicious. However, the NPM server administrators are fully capable of deleting malicious packages or banning malicious users. NPM exposes usage statistics and number of depending packages to assist developers in picking libraries.

Packages on NPM are registered on a first come, first-served basis and do not distinguish among authors, meaning that the unpublishing of a package can not only break the projects that depend on it but also pose a security risk. For example, a package named

"left-pad" that only padded the left side of strings inadvertently caused many builds to fail when it was removed.

Express.js, or simply Express, is a web application framework for Node.JS, released as free and open-source software under the MIT License. It is designed for building web applications and APIs. It is the de facto standard server framework for Node.js.

5.1 Running Node sever

One of the most important reasons makes Node.JS a popular platform is simplicity to run and work with Node.JS as backend server. We can start Node.JS server by running **npm start** command and then by answering some questions we can run the very basic server. As it was mentioned earlier, NPM (Node package manager) is responsible for adding all required libraries to our application. We can add all dependencies through **npm install** command. After we installed a dependency on our application, in order to use that we have to call it in our application. All of installed packages could be found in Package.json file (Picture 21). Package.json is a file that includes all dependency and basic setting for a Node.JS application.

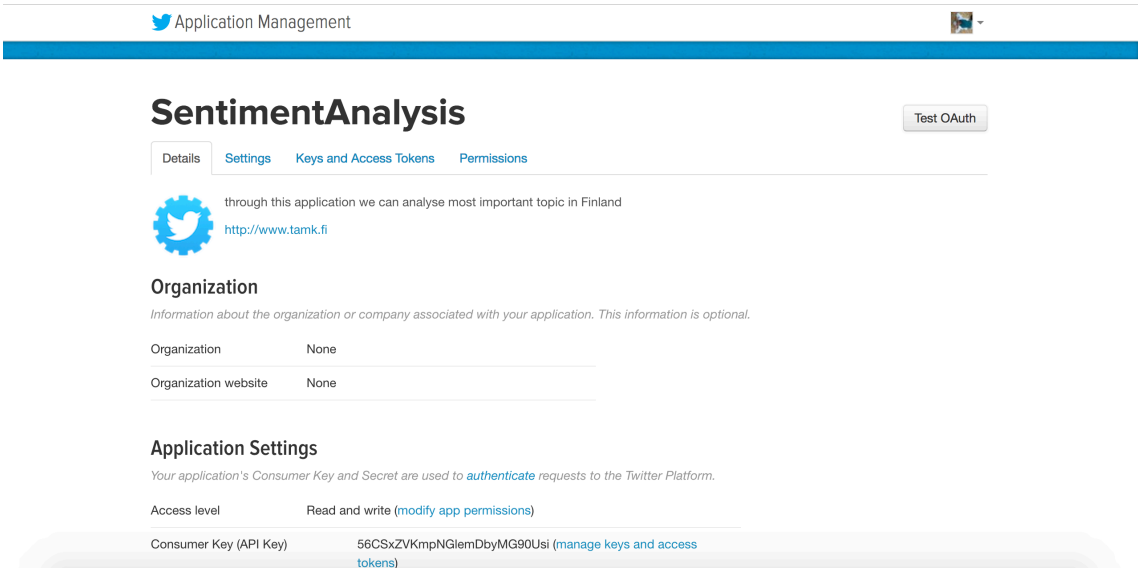
Picture 21. Package.json file for our project

For the thesis we tried to make a light version of application using Node.JS libraries. As we can see in picture 22 we have used Express which is responsible for making web application. The Twitter library is responsible for connecting to Twitter APIs and also get and post data to/ from Twitter. Finally the Sentiment library as it was mentioned earlier is responsible for analyse the text and calculate score of sentiment for the text. we can call a library through **require** command.

```
var express = require("express");
var app = express();
var Twitter = require('twitter');
var sentiment = require('sentiment');
```

Picture 22. Adding all required dependencies into project.

First we need to make a Twitter object to make connection with Twitter APIs. For this purpose, we have to first make a Twitter application through dev.twitter.com. After we made an application we can get the authentication attributes the for the connection object. Picture 23 shows a Twitter application. We can manage the settings and permissions for our application through application console. Any application which wants to work with Twitter API should be defined through dev.twitter.com.



Picture 23. Twitter application defined for this project

After defining the Twitter application, we have to define authentication object for connecting Node.JS application to Twitter APIs. As we can see from pictures 24, an object includes **consumer_key**, **consumer_secret**, **access_token_key** and **access_token_secret**.

```
var client = new Twitter({
    consumer_key: '56CSxZVKmpNGlemDbyMG90Usi',
    consumer_secret: '0jsGmVYXFztmAlGw5aAKcy7O2H1wGJUImF9kuvOErokV3TXuY6',
    access_token_key: '6100912-CRMSQN9B8p5gYerbkZAJtpLjFHsmUCV4cfqdpIkdKl',
    access_token_secret: '67EzQcO4uTxGBWtFQGgL1jKJZAgvIi6j3iLEgUdVxFDmI'
});
```

Picture 24. Authentication object defined for connecting to Twitter API

The next step is reading the data from **get Trend/place** API. We need to pass the place-id as a parameter to the function. As it was mentioned in the last chapter this API get a place-id in form of WOEID and return trend hashtags for that place.

After the function returns the response, we have to order the result based on the **tweet_volume** attribute. The tweet_volume is a number which shows how many times a hashtag has been used. Since the API return top 50 hashtag trends for a special location, after we order trends based on tweet_volume we can separate first 10 popular hashtags.(Picture 25)

```
client.get('trends/place', params, function (error, tweets, response) {
    if (error) throw  error;
    // in code below the returned hashtags will be ordered by their tweet-volume values.
    var mostPopularHashtags = tweets[0].trends.sort(function (a, b) {
        return b.tweet_volume - a.tweet_volume;
    }).splice(0, 10);
```

Picture 25. Get the top 5o popular hashtag and select most 10 popular.

The next step is to analyse sentiment and calculate the score for each of selected hashtags. For this purpose first, we have to get Tweets related to the hashtag. We can get Tweets from **Get search/tweets** API. The API returns an array of Tweets related to selected hashtag. Then we have to calculate sentiment for each Tweet. Sentiment library gets a text as input and returns a score as a result of sentiment analysis. In order to get the average score of each hashtag, we have to calculate sentiment for all tweets and finally calculate the average. .(picture 26)

```
mostPopularHashtags.forEach(function (item, index) {
    client.get('search/tweets', {q: item.name.slice(1)}, function (error, tweets, response) {
        var totalScore = 0;
        var totalComparative = 0;
        var avgScore = 0;
        var avgComparative = 0;

        var tweetslength = tweets.statuses.length;

        for (var i = 0; i < tweets.statuses.length; i++) {
            var sentimentResult = sentiment(tweets.statuses[i].text);
            totalScore += sentimentResult.score;
            totalComparative += sentimentResult.comparative;
        }
        avgScore = totalScore / tweetslength;
        avgComparative = totalComparative / tweetslength;
        sentimentResponse.push({id: index , order: 1, comparative: avgComparative ,
            label: item.name, score: avgScore});
    });
});
```

Picture 26. Get Tweets related to each hashtags and calculate sentiment for each of them

The last step the of the process in the backend is making a response in JSON format and send that to front-end side. Most of this part will be done through Express package. In each cycle front-end send a get request to backend through browser and backend after all of these processes will return the JSON data to front-end.

6 FRONT-END DESIGN AND ARCHITECTURE

For the light version of the application, we are going to implement a simple demonstration of the result of sentiment analysis for hashtag trends. First, we have to get the result of sentiment analysis from the backend. This backend result includes 10 First popular hashtag names and its sentiments which could be positive, negative or neutral. Additionally, for each sentiment, we have a score which shows how strong is each opinion.
As we mentioned earlier in case we are going to implement a web application for our final product we can go through with one of the JavaScript frameworks which mentioned in earlier chapters, but for this step, we only need a simple web page. One of the best options for this purpose is the D3 library.

6.1 D3.JS

D3 is a JavaScript library for producing dynamic, interactive data visualisations in web browsers. It uses the widely implemented SVG, HTML5, and CSS standards. In contrast to many other libraries, D3.JS allows great control over the final visual result. D3.JS is used on hundreds of thousands of websites. Some popular applications include creating interactive graphics for online news websites, information dashboards for viewing data, and producing maps from GIS map making data. In addition, the exportable nature of SVG enables graphics created by D3 to be used in print publications.

6.2 Implement the front-end with D3

First, we need to get the data from the backend. D3.JS has a function for this purpose. As we can see in picture 27, with function 'json', we can read the data from the backend.

```
d3.json("http://localhost:3000/getSentiment", function(error, data) {
    data.forEach(function(d) {
        d.id     =  d.id;
        d.order  = +d.order;
        d.color  = diagram_color(d.score);
        d.comparative = +d.comparative;
        d.realScore = d.score;
        d.score  = +Math.abs(d.score)+ 0.1;
        d.width  = +1;
        d.label  =  d.label;


    });
```

Picture 27. D3 has different methods include json to read source data.

We have to provide our API URL and we will get the result in the call-back function. After getting data, for each of objects we have to define what we need for drawing our pie chart. The label attribute will hold the hashtag name, for the score, we added 0.1 to the score value because we need to see all hashtag labels on the diagram by hovering on that. If we don't add 0.1 to the scores, then we can't see the label for neutral values, as it will be only a line.

We want to have colour for each hashtag and this colour should be green for positive, red for negative and white for neutral. Also, we would like to have the darker colour for bigger values. The code snippet below shows the calculation for the colour of each hashtag.

As we can see from picture 28, function **diagram_color** get a score as input and then will return a colour related to values of the score. First, based on the value of score we decide that the colour should be green, red or white, then using the function 'Interpolate' we define how dark colour will have.

```
function diagram_color(val) {
    red = new Color(232, 9, 26),
        white = new Color(255, 255, 255),
        green = new Color(6, 170, 60),
        start = white,
        end = green;
    if (val < 0) {
        start = white,
            end = red;
        val = Math.abs(val);
    }
    var startColors = start.getColors(),
        endColors = end.getColors();
    var r = Interpolate(startColors.r, endColors.r, 10, val);
    var g = Interpolate(startColors.g, endColors.g, 10, val);
    var b = Interpolate(startColors.b, endColors.b, 10, val);
    var colors = d3.rgb(r,g,b);
    return colors;
}
```

Picture 28.Code snippet for function diagram_color

```
function Interpolate(start, end, steps, count) {
    var s = start,
        e = end,
        final = s + (((e - s) / steps) * count);
    return Math.floor(final);
}
```

Picture 29. Function Interpolate calculates the code of colour for different scores. The Higher score should have a darker colour.

The last part of our simple front-end implementation includes draw the pie chart and publish the result for that. In this regard, we need to follow the way which is common in D3 diagrams. It starts with append an SVG to the page and set its attributes such as width and height. Following the code, we can see the way we defined the pie chart on our SVG and publish the result for that.

```
var width = 500,
    height = 500,
    radius = Math.min(width, height) / 2,
    innerRadius = 0.3 * radius;

var pie = d3.layout.pie()
    .sort(null)
    .value(function(d) { return d.width; });

var tip = d3.tip()
    .attr('class', 'd3-tip')
    .offset([0, 0])
    .html(function(d) {
        return d.data.label + ": <span style='color:orangered'>" + d.data.realScore + "</span>";
    });

var arc = d3.svg.arc()
    .innerRadius(innerRadius)
    .outerRadius(function (d) {
        return (radius - innerRadius) * (d.data.score / 5.0) + innerRadius;
    });

var outlineArc = d3.svg.arc()
    .innerRadius(innerRadius)
    .outerRadius(radius);

var svg = d3.select("body").append("svg")
    .attr("width", width)
    .attr("height", height)
    .append("g")
    .attr("transform", "translate(" + width / 2 + "," + height / 2 + ")");

svg.call(tip);
```

Picture 30. Code snippet for drawing diagram by D3. After calculating all required data we can draw diagram with d3 functions.

Picture 31 shows how the result of sentiment analysis will look like. As we can see from the left side of the picture, the pie chart shows the result of 10 most popular trends and for each trend, we can check the topic by hovering on it. As an example, Bill Cosby is a negative topic with the score 1.86 is among the results.

Picture 31. Final result of application. Right side of picture shows console log for Twitter hashtags.

The codes for the light version of application is available on GitHub through this link. https://github.com/mojisilver/sentiment-Analyser.

7 CHALLENGES AND OPPURTUNITIES

Thinking about making an application without considering its issues is a big mistake. When we decide to make an application we have to know the scope of our project, who are our stakeholders, which technologies we need, how we have to make and follow our plans, how to evaluate options and many other tasks like these. Challenges are a big part of all steps in software development cycle which need to be planned and deal with those in right place. These challenges could be in any steps of software development cycle. In order to manage the project's challenges, we divide those into two main categories. Technical and business challenge.

7.1 Technical challenges

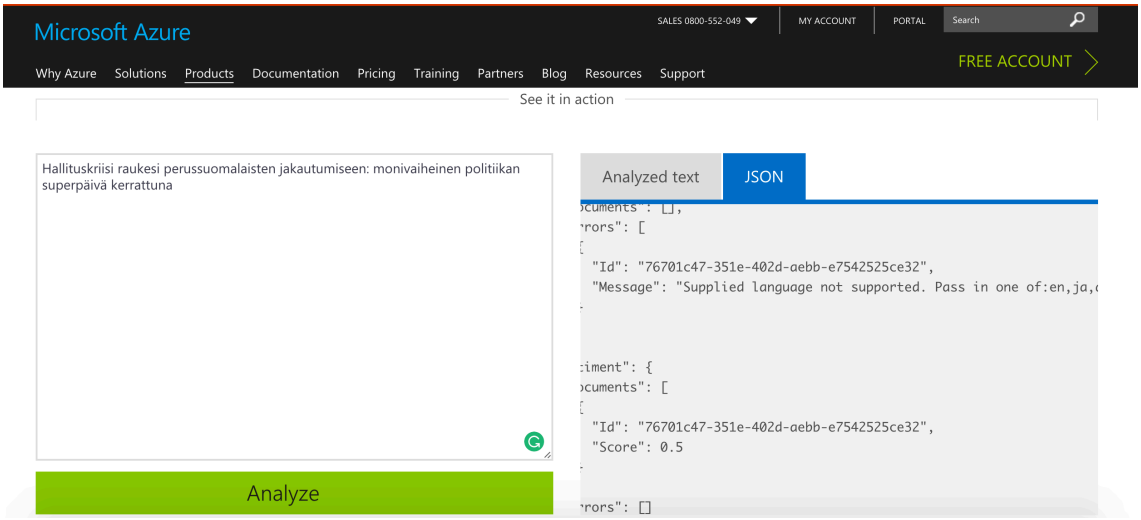
Technical challenges refer to issues related to technology selection or lack of technology in some part of software development cycle. As it was mentioned earlier the issues such as limitations related to Twitter API to provide hashtag trends for Finland through 'Get Trends/place' API or unavailability of sentiment analyser in the Finnish language is considered as the technical issues.

For the light version of the project, we deal with those issues by changing the scope to something else. As an example we change the location to the United States and English language instead of the Finnish language. However, for further development, we have to plan for solving these issues.

7.2 Business challenges

Since the scope of this project doesn't involve marketing side and business planning, we have not focused on business challenges in our study. However, it would be beneficial to mention some business challenges as well as opportunities. Sentiment analysis is very new and fast-growing knowledge and every day new applications are being defined from product analysis to politics.

Depend on which area we are going to continue our business there are some competitors which should be analysed their marketing plan. As an example, if we are planning to provide API services, we have to check available services such as Bitext and Text-processing. Moreover, most of the big companies are providing sentiment analysis service for themselves or provide this services as API. As an example, Microsoft has Text Analytics API which by getting texts returns sentiment of text, key-phrases, topics and even language of a text. Microsoft service is not available for the Finnish language.



Picture 32. Microsoft Azure service for sentiment analysis.

Google also has its own service for sentiment analysis. Natural language API is Google service for sentiment analysis and is part of Google could platform services. The code snippet below shows how we can use Google sentiment analysis service in a Node.JS server implementation. First, we need to install the client library by running **npm install --save @google-cloud/language**. After that, we should import the library to the application and make a client instance. We also need to make a Google Cloud Platform project and then use its ID to instantiate the object. Rest of the task include getting the text and use the library for analysing the text is like our current implementation with Sentiment library. It returns text and its score which shows polarity and strength of the sentiment. (Picture 33)

```
// Imports the Google Cloud client library
const Language = require('@google-cloud/language');

// Your Google Cloud Platform project ID
const projectId = 'YOUR_PROJECT_ID';

// Instantiates a client
const language = Language({
  projectId: projectId
});

// The text to analyze
const text = 'Hello, world!';

// Detects the sentiment of the text
language.detectSentiment(text)
  .then((results) => {
    const sentiment = results[0];

    console.log(`Text: ${text}`);
    console.log(`Sentiment score: ${sentiment.score}`);
    console.log(`Sentiment magnitude: ${sentiment.magnitude}`);
  })
  .catch((err) => {
    console.error('ERROR:', err);
  });
```

Picture 33. Using Google platform in Node.JS application

7.3 Opportunities

Most of the application's features which are mentioned in the first chapter could be considered as competitive advantages. Additionally, unavailability of sentiment analysis in the Finnish language makes the application unique and there are lots of possible business opportunity for that. This services could be same as available service in English such as analyse product's reviews or analysing people attitudes in social media. Also based on location, sentiment analysis could be used for different things. As an example, since weather plays a significant role in Finland people's opinion about that could be a valuable source for some business like travel agencies.

8 CONCLUSION

The objective of the study was to prepare the proof of concept (POC) about doing sentiment analysis of social media data related to Finland. A light version of the application with minimum features was developed and feasibility to create the final version of the application based on current knowledge were evaluated.

There were some issues which should be addressed in order to make the final version of the application. The first issue and the most important is about the accuracy of sentiment analysis based on current knowledge. In fact, since this field of science is very new, there are lots of weakness in sentiment analysis and its results and it required many improvements to provide the most correct response.

The second issue which also is one of the main problems is about access to the social media data which uploaded by users from Finland. Currently Twitter provides the trends related to some limited location through its own APIs and unfortunately, Finland is not among those locations. The closest location on that list is Stockholm and as author contacted with Twitter developers team, they responded that they are not planning to add new location soon. Thus, we need to figure out a way for the final version of the application to access the data precisely. Some alternative options were studied and the quality of results was evaluated. It is necessary to be mentioned that current version of application evaluates the data related to the United States and the main reason for the select United States were first because all Tweets were in English and secondly because the volume of data is much higher in compare with other places. Moreover, the data and trends in the United States are changing faster than other location which could show the result of our study better.

The third issue that should be addressed in the final version is that currently, sentiment analysis is not available for the Finnish language. In fact, most of the research and applications have been done in English language and some limited research and application have been done in other languages. There was just some research related to NLP (natural language processing) field which already has been done about Finnish language and it could be considered as a starting point for developing sentiment analyser for the Finnish language.

The result of the study which was getting the basic knowledge related to make a sentiment analysis application and develop a light version of the application was successful. However, it is necessary to take into account that for the final version of application some of the services should be improved and some should be replaced and also, we need to do planning in order to implement those missed features.

from the technical point of view also in order to deploy the real version of application we need to consider issues related to deployment such as security issues and storage issues which they had not been considered as a part of this study.

REFERENCES

Strapparave, C & Valitutti, A & Stock, O. 2006. The Affective of Lexicon. ITC-irst, 38050, Povo, Trento, Italy.

Wilson, T & Wiebe, J & Hoffmann, P. 2005. Recognizing Contextual Polarity in Phrase Level Sentiment Analysis. Proc. Of HLT-EMNLP-2005

Clifton, C. 2009. Encyclopedia Britannica: definition of data mining.

Tan, A. Text mining: the state of art and the challenges. Kent Ridge Digital Lab.

Hearst, M. 1999. Untangling Text data mining. School of information management and systems, University of California, Berkley.

Fayyad, U & Piatetsky, G & Smyth, P. 1995. From data mining to Knowledge discovery in Databases.

Ahonen, H & Heinonen, O & Klemettinen, M & Verkamo, A. Applying Data mining techniques in Text analysis. University of Helsinki, Department of Computer science.

Chakraborty, G & Pagolu, M & Garla, S. 2013. Text mining and analysis: Practical Methods, Examples and Case studies using SAS. Chapter1: 5-7

Treloar, N. 2002. Text mining: Tool, Techniques and applications, AquaQuest Inc.

PubGene Product, Coremine medical. www.pubgene.com

Wagner, J. 2013. Expert System Release Cogito Intelligence API. www.programmableweb.com.

Mikalai, T & Themis, P. 2012. Survey on mining subjective data on the web.

Nasukawa, T & Yi, J. 2003. Sentiment analysis: Capturing favorability using natural language processing. In Proc. of the Second International Conferences on Knowledge Capture.

Dave, K & Lawrence, S & Pennok, D. 2003. Mining the peanut gallery: Opinion Extraction and semantic classification of Product reviews.

Liu, B. 2012. Sentiment analysis and opinion mining. P:7.

Medhat, W & Hassan, A & Korashy, H. 2014. Sentiment Analysis algorithms and applications: A survey. School of electronic engineering, Canadian international college, Cario campus of CBU, Egypt.

Razzaz, M. Aspect level for sentiment classification for Arabic Language.

Casey, W & Navendu, G & Shlomo, A. 2005. Using appraisal groups for sentiment analysis.

Desai, M & Mehta, M. 2016, Techniques for sentiment analysis of Twitter data: A comprehensive survey.

Factiva, D. J. 2009. Direct correlation established between social media engagement and strong financial performance. PR News, 65(29), p. 3.

Chin, D & Zappone, A & Zhao, J. 2016. Analysing Twitter sentiment of the 2016 presidential candidates.

http://www.gartner.com/newsroom/id/3609817

https://w3techs.com/technologies/overview/javascript_library/all

[https://github.com/thisandagain/sentiment

http://text-processing.com/docs/sentiment