Diploma Project

SENTIMENT AND TOPIC ANALYSIS FROM TWITTER BASED ON GEOLOCATION

Christodoulos Loukaides

UNIVERSITY OF CYPRUS

DEPARTMENT OF COMPUTER SCIENCE

May 2020

UNIVERSITY OF CYPRUS

DEPARTMENT OF COMPUTER SCIENCE

Sentiment and Topic Analysis from Twitter based on Geolocation

Christodoulos Loukaides

Supervising Professor

Marios Dikaiakos

The Diploma Thesis was submitted for partial fulfillment of the requirements for obtaining a degree in Computer Science from the Department of Computer Science of the University Of Cyprus

May 2020

Acknowledgements

I have to start by thanking my supervising professor, Dr. Marios Dikaiakos. We had an excellent cooperation; throughout the year he gladly offered me his knowledge, time, advice and useful material which all contributed to the outcome of this project.

I would also like to thank Demetris Paschalides and Dimosthenis Stefanidis for offering me their assistance is some practical aspects of the project.

I have to also thank all my professors for helping me develop and reinforce my Computer Science related skills, and especially Dr. George Pallis, whose course “Data Mining on the Web” offered crucial help for the development of many of the features of this Diploma project.

Finally, I would like to thank my family for supporting me throughout my university years and providing everything for me, so all I had to do was focus on my studies.

Summary

It is widely accepted worldwide, that technology consists of a huge influence on the way we live our lives. People spend multiple hours each day on Social Networking platforms. As a result, many terabytes of data are produced daily through those platforms. This data is of big significance, as all types of information can be retrieved, such as how people from around the globe live their lives, what they are concerned or interested about, how they feel, which people they interact with etc.

Twitter has a user base of about 330 million users. Additionally, it has the most extensive set of API tools in comparison to all other famous Social Networking platforms, with rather lenient limitations on data requests, allowing hundreds of tweets to be fetched per minute. All these reasons make Twitter is an ideal tool for the purposes of this Diploma project.

This project's main purpose is to extract and analyze data, in order to extract useful information about people depending on their geographic location. More specifically, the project aims to produce information about the trending topics and sentiment for each country. The recent Covid-19 pandemic has drastically changed our lives for the past couple of months. This situation is reflected in the content that people post on social networks. This provides a great opportunity for testing the tools that were developed on the project in order to provide information about how people in different regions around the world react emotionally to this situation.

The project is split into multiple components. There are components for fetching and storing tweets, components that predict the user's location in case that it is not provided, components that perform sentiment analysis and components that perform topic analysis, all of which will be thoroughly described in this paper.

Table of Contents

Chapter 1 - Introduction

In this chapter, the basic goals, incentives and outline of the project are described. The fundamental concepts are introduced and the main idea behind the project is explained.

1.1 Incentive of the project

Data Science has bloomed over the past years and the reason for this, is the fact that millions of people produce data on Social Networks. This data is then processed with the purpose to produce information that could be useful in a variety of ways. Nowadays, it is possible to extract personality features, interests, relationships, political views and many other useful elements of a single person, or even groups of people.

This information allows analysts to form conclusions and produce statistics about certain situations and people, and helps businesses boost their productivity and effectiveness over their customers, by examining their behaviors, thoughts and feelings on their social networking profiles.

The main focus of this project is to define characteristics of people depending on where they live. This way, at any given time, analytics can be produced regarding the topics that interest people and how they feel in each country of the world individually, or globally.

The significance of this information extraction has led to the development of powerful tools for harnessing it. Such tools include APIs (Application Programming Interfaces), NLP (Natural Language Processing) tools, subcategory of which are the sentiment analysis tools, and machine learning modules for pattern recognition and feature categorization. These tools aid the development of projects such as this one, by offering automated toolkits, saving thousands of lines of code and allowing advanced functions to be operated without requiring a high skill and knowledge level.

Twitter fairly complements the incentive, by providing powerful API tools for harnessing massive volume of data, from a huge user base which spans all around the world and consists of public figures and businesses besides normal people. The type of content on the platform also greatly benefits the incentive, as it consists of text limited to 280 characters with hashtags that define the focus of the content.

1.2 Goals of the project

This project's main purpose is to extract the trending topics and sentiment of people depending on their geographic location, draw conclusions and gather statistics based on this information. This is a very broad goal, consisting of the following sub-goals:

1. Fetching of tweets, either by random sampling or using specific filters
2. Storing fetched content in a database system
3. Predict the location of the users for tweets that didn't provide geolocation information
4. Perform sentiment analysis on the stored data
5. Perform trending topics analysis on the stored data
6. Visually represent the results that were generated

1.3 Outline of the project

In this chapter, the incentive, goals and outline of the project are described.

In the second chapter, the theoretical background will be presented, describing all technologies, platforms and tools that were used to achieve the goals of the project.

The third chapter contains a detailed and procedural description of how the developed algorithms operate and how data is handled at all phases.

In the fourth chapter, the implementation details are described. The development techniques used, and the code structure for all the scripts that the algorithms use will be explained.

The fifth chapter consists of the assessment and evaluation of the project, as to the quality and accuracy of the results, and the completion level of the goals that were set for the project.

In the sixth chapter, the final and holistic conclusions regarding the project will be drawn and explained.

Chapter 2 – Theoretical Background

In this chapter, the Twitter platform will be described, as to the way its content is offered, and the Streaming mechanism provided to the developers.

This will be followed by a description of all other tools and technologies utilized, including the means of storing data and the programming language chosen, as well as the packages that were used.

This chapter is of great importance, as the implementation of the project heavily relies on the synergy of all these tools and technologies in order to produce the desired results.

2.1 Problem Definition

The main goal of this project is the extraction and visual representation of trending topics and sentiment of people based on their geolocation. In order for this goal to be achieved, a pipeline of algorithms is executed. The different parts of it will be highlighted below.

The first part consists of fetching tweets through the Twitter Streaming API, extracting the data we want to keep from each, and pushing them to a Kafka Producer.

The second part consists of receiving the tweets from a Kafka Consumer and storing them in a database with their respective fields. Before storing the data, the text pre-processing occurs, as well as the prediction of the user's location, if it is not provided. It should be noted that the lack of geo-location information from most tweets consisted of the greatest challenge in the project, as only about 0.85% of tweets provide geo-location data [1]. The sentiment analysis also happens at this point. All these data are stored in the database, alongside the tweets.

The third part is the sentiment analysis per country. The sentiment information is extracted from the stored tweets and a sentiment report is created for all countries worldwide, accommodated by a visual representation of the sentiment.

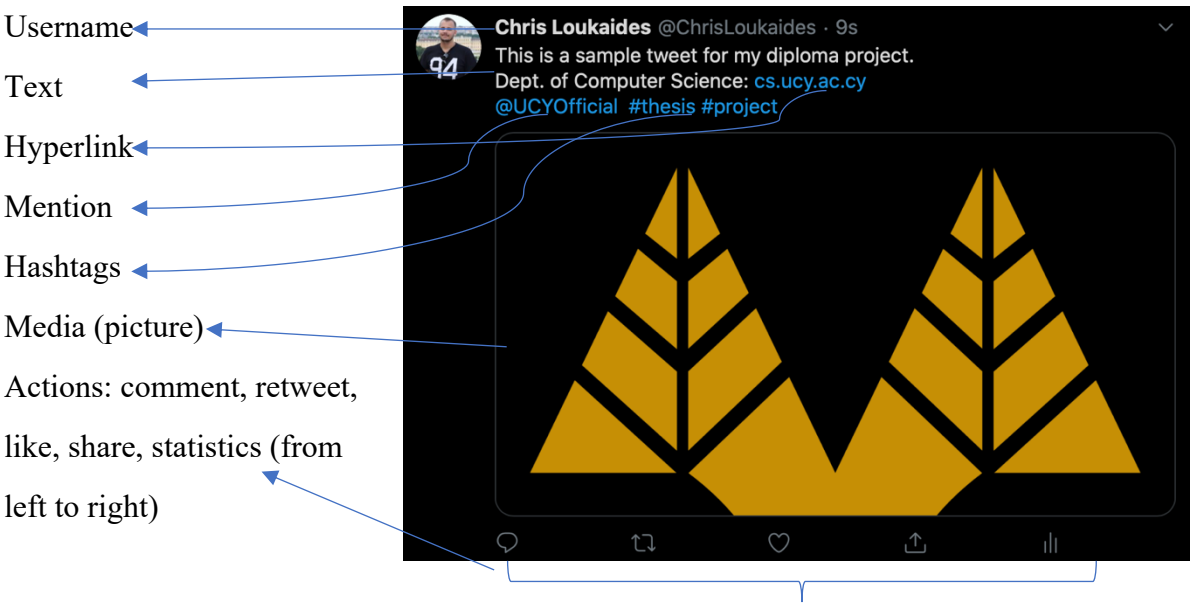
The fourth part is the topics analysis per country. Individual trending topics reports are generated, as well as visual representations for each country, after analysis in performed on the stored tweets.

2.2 Twitter

Twitter is an American social networking platform that was founded in 2006. It has about 330 million users, 150 million of which are active daily. The content is offered in the form of "tweets". About 500 million tweets are sent per day [2]. Twitter user profiles and tweets can be accessed without an account, unless a user has chosen to switch to a private account, in which case other users must request permission to view their content.

Signing up for an account allows a user to interact with other people's tweets. The available actions will be mentioned below. Having an account also allows the user to follow other users and have their tweets appear on his homepage. This applies the other way around as well, as other users can follow a user to have his content displayed on their homepage. The platform offers instant messaging capabilities between users.

Tweets consist of text limited to 280 characters and can also have media like pictures and videos attached to them as well. The text may contain special elements, such as mentions ('@'), hashtags ('#') and hyperlinks. Mentions give the capability of mentioning other users in a tweet. Hashtags allow users to define the topic of the content they are posting. Four different actions can be performed on a tweet. A user can like a tweet, comment on a tweet, share it via twitter messages or by copying the link, and retweet it. Retweeting a tweet allows a user to share another user's tweet on their profile and comment on it, if they so desire. Users can also see statistics for their own tweets.



2.2.1 Twitter Streaming API

APIs (Application Programming Interfaces) are interfaces that allow developers to perform requests to a service and get responses with the desire content. They simplify the procedure of interacting with a software service by abstracting the underlying implementation and only exposing objects the developer needs.

Twitter offers very powerful API tools for developers to be able to push to and pull data from the platform. Any action that can be performed on the GUI (Graphical User Interface) of the platform through the web or the mobile apps can be performed through API calls. Additionally, API calls allow for requests of big batches of data, for example: all the tweets of a user.

The Streaming API is one of the multiple API tools provided by Twitter and it is the one crucial to the implementation of this project. It allows for infinite streaming (fetching) of tweets. There are two mode of operation. The first mode streams a random 1% of the total tweets published at the time. The second mode of operation is streaming based of filters. These filters consist of tweets from specific users, tweets containing specific keywords or hashtags and tweets from specific locations. The responses returned are in JSON (JavaScript Object Notation) format, which has been the standard for web API responses for many years and provides great ease and flexibility with its easily readable format, object-oriented approach and light bandwidth requirements [3].

A JSON response from the Streaming API is presented below:


```
{
  "created_at" : "Thu Apr 06 15:24:15 +0000 2017" ,
  "id_str" : "850006245121695744" ,
  "text" : "1\/ Today we\u2019re sharing our vision for the future of the Twitter API platform!\nhttps:\/\/t.co\/XweGngmxlP" ,
  "user" : {
    "id" : 2244994945 ,
    "name" : "Twitter Dev" ,
    "screen_name" : "TwitterDev" ,
    "location" : "Internet" ,
    "url" : "https:\/\/dev.twitter.com\/" ,
    "description" : "Your official source for Twitter Platform news, updates & events. Need technical help? Visit https:\/\/twittercommunity.com\/ \u2328\ufe0f #TapIntoTwitter"
  } ,
  "place" : {
  } ,
  "entities" : {
    "hashtags" : [
    ] ,
    "urls" : [
      {
        "url" : "https:\/\/t.co\/XweGngmxlP" ,
        "unwound" : {
          "url" : "https:\/\/cards.twitter.com\/cards\/18ce53wgo4h\/3xo1c" ,
          "title" : "Building the Future of the Twitter API Platform"
        }
      }
    ] ,
    "user_mentions" : [
    ]
  }
}
```


The purple text is the attributes of the tweet, like the date it was posted or its text. The green text is the content of each attribute [4].

2.3 MongoDB

MongoDB was released in February 2009. It is a cross-platform document-oriented NoSQL database software. It is one of the most widely used database tools in Big Data and Real-Time Web applications due to the many advantages it has over other popular database software.

Being a NoSQL database system, it inherits all the advantages that this database structure offers. Consistency of the data structure is sacrificed in favor of availability, performance and partition tolerance [5]. In the case of this project, the consistency of the data entries not only is not needed, but it could be deemed as undesired. The reason for this is the fact that not all tweets that are fetched have the same attributes. In fact, most tweets have at least one attribute missing (such as geo-location, or URLs). A rigid data structure, such as the one required in traditional SQL systems would introduce many null entries. NoSQL allows data entries without limitation on their fields. A data entry missing some fields would require less storage space in a NoSQL database, since it would not have fields with 'null' value. Instead, that field would not exist at all. And since Big Data applications, such as this project have thousands of data entries, significant amounts of storage space are saved and the performance is much greater by using a NoSQL database system, instead of an SQL one.

In addition to the advantages offered by the NoSQL principles, MongoDB offers many unique advantages of its own. First of all, it offers powerful query commands and tools, similar in syntax and functionality to common SQL query commands, offering easy and fast database manipulation. Data is stored in the form of JSON documents, instead of the traditional row/column model [6]. This advantage is of high importance, since data is stored in an object-oriented format that supports arrays and nested objects as values. The JSON format is also easily readable and lightweight in terms of both performance and data size.

2.4 Apache Kafka

Apache Kafka is an open-source distributed stream-processing software platform developed by LinkedIn and donated to the Apache Software Foundation in January 2011. The platform is extensively used in Big Data applications built around real-time data feeds.

Handling real-time streams of large volumes of data was a challenge for a long time in this field of Computer Science. The goal was to be able to process that data in a performance-optimized and fault-tolerant way. This is exactly what Kafka offers. It can process streams of data by allocating workers to divide and push the incoming data to the processing and storage units of a system according the capability and availability of its resources (CPUs, HDDs). This model of data processing prevents the system resources from being flooded with data and offers fault tolerance [7].

The Kafka platform offers 5 core APIs. For this project, the Producer API and the Consumer API were utilized. These APIs work on the principle of topics. The Kafka cluster stores the data streams in categories, called topics. Multiple streams can be processed simultaneously in different topics. The Producer API pushes data from a stream to a topic on the Kafka cluster. The data remain available within that topic until they get consumed or discarded. This is where the Consumer API takes action, consuming the data from a topic. The cluster works on the principles of partitioning. Data within a topic is published by the producer in various partitions. The consumer pulls data from these partitions. The write/read actions on the partitions from the producers and the consumers are synchronized by the cluster, so optimal performance can be achieved. The Producer-Consumer combo provide a complete streaming data processing model.

2.5 Python and packages

Python is a high-level general-purpose programming language released in 1991. It is an open-source and cross-platform piece of software. It has a unique and easily readable syntax which is visually uncluttered and uses English words for most statements and expressions instead of punctuation unlike most other languages. The functionality of curly braces and semicolons is replaced by indentation [8].

Python is the most popular programming language (on the time period this report is written at) with a popularity of 31.17% (based on amount of google searches occurred about language tutorials) [9]. The reason for the huge increase in popularity in the last years is the Python Package Index (PyPI). It is an online repository of Python code which consists of fully implemented functions and applications. This way, programmers don't have to "reinvent the wheel" every time they write a new piece of code and can instead import packages from PyPI that get the job done. This service is also open-source, and anyone can contribute their code, if they so desire.

As mentioned earlier, Data Science and Big Data Analysis is trending nowadays. The fact that very powerful Python packages for Data Analytics exist also greatly contributes to the popularity of the language. Many of them have been used for this project and are crucial for its operation and will be presented below.

2.5.1 NLTK and Textblob

NLTK (Natural Language Toolkit) is the most popular package for Natural Language processing. NLP is an integral aspect of Data Analysis, as it is the core of techniques that allow us to be able to efficiently extract information from natural language (text data) with as little noise as possible. Such techniques are stemming, lemmatization and part-of-speech (POS) tagging. The techniques utilized in this project will be explained in detail in the next chapter, where the project's way of operation will be discussed.

Textblob is a package built on top of NLTK, utilizing many of its features and capabilities and offering some additional ones. NLTK is an extensive package in terms of functionality, so it's syntax and utilization require a learning curve before a programmer can get familiarized with it. Textblob offers simplified syntax for the most popular NLTK features. In addition, it integrates other features, such as sentiment analysis, text translation through the Google Translate API and spelling/grammar correction. These features are of utmost importance in order for the goals of the project to be achieved, as tweets consist of text written in many different languages and text that often has spelling mistakes. The fact that these tools work in full integration with the NLTK tools make the text processing aspect of this project a lot easier and straight-forward.

2.5.2 Tweepy

Tweepy is a package with the purpose of giving the user an easy, straightforward way of accessing the Twitter API. All possible API calls offered by the Twitter API can be executed through tweepy commands. The package manages the encoding and decoding of the JSON files that are exchanged between the client and the API and allows simplified fetching of specific elements from the JSON responses without the user having to interact at all with the JSON file.

2.5.2 Geopy

Geopy is a geo-location focused package. It has two modes of operation. The first one returns the coordinates (latitude and longitude) when given an address as input. The second mode offers the reverse functionality, returning an address when given a set of coordinates. In order to implement those features, it utilizes 3 different APIs: Google Maps, Bing Maps and Nominatim. The user can select which API he wants to be used. One of the main focuses of this project is geo-location, which is why this package is crucial for the operation of the algorithms.

2.5.3 Googletrans

The googletrans package is a wrapper around the Google Translate API. It gives the programmer an easy way to send requests and get responses from the Google Translate API. It supports three features: text translation, language detection and simultaneous translation of bulks (big batches) of data. Text from tweets comes in many different languages, so the translation feature provided by this package really assists the processing of the text of tweets.

2.5.4 Matplotlib

The matplotlib package provides powerful visualization tools. It can produce different plots, like bar charts, scatterplots, histograms and 3D-plots automatically with data provided by the user. It is fully compatible with the pandas and NumPy packages, which are popular packages in the field of Data Science due to their capabilities of storing and managing objects that hold and can perform mathematical operations on massive data frames. This project harnesses the functionality of the package with the purpose of visually presenting the data produced by the sentiment and topic analysis.

2.5.5 Scikit-Learn and LDA

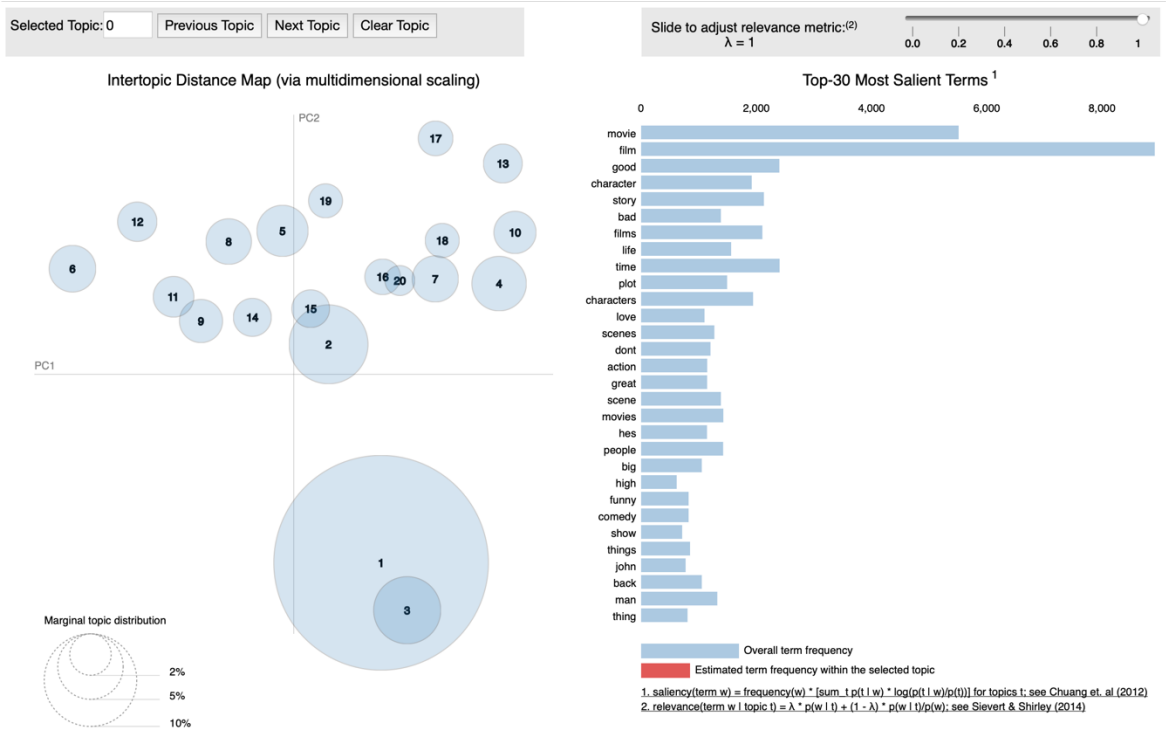
Scikit-learn is the most widely used Machine Learning package for Python. It provides implementations of the most useful machine learning modules for classification, regression, clustering and dimensionality reduction. It provides tools for automatic preprocessing of the data and machine learning model configuration and training, relieving the programmer from having to implement these very advanced features that would otherwise require thousands of lines of code.

In this project, we specifically use the Latent Dirichlet Allocation (LDA) model. It is a statistical model that follows the principles of k-means clustering to produce its results.

There are two areas in which this algorithm is the most favorable. The first one is Biology, where LDA is used to detect the presence of structured genetic variation in a group of individuals. The second area, which is the one that concerns us, is Natural Language Processing, and more specifically Individual Topic Detection from a given text input.

2.5.6 pyLDAvis

pyLDAvis is a package that provides visualization for the topics produced by an LDA model. The package creates an html file with a dynamic representation of the topics discovered by an LDA model. The representation displays all the topics on a 2-axis plot, displaying the most frequent words for each topic, alongside their frequency. The package has built-in integration with the LDA model of the scikit-learn package, and it can automatically produce the visualization once the model has finished training and has returned its results. Below, please find a visualization example, provided by the developers of the package [10].



Chapter 3 – Description and Analysis of the System

In this chapter, the algorithms that constitute the implementation of the project will be discussed. The pipeline of tasks will be thoroughly described and analyzed. The algorithms are presented in the same order they are actually executed chronologically in the project.

The full journey of the data will be explained, from the moment they are fetched by the application, until the final phase where they have evolved into visualized information.

The essence of this chapter lies in the way data is being manipulated, how they pass from one phase to the next one and the processing that occurs to them in each phase.

3.1 Data Fetch from Twitter and Push to Kafka

The first task that has to be done is to fetch tweets. For fetching, the Twitter Streaming API is utilized, providing continuous and uninterrupted access to tweets that are posted in real time.

The tweepy package is used to make the API requests, as it automatically handles the configuration and authentication process required to make the requests and allows for straightforward access and manipulation of the different elements of the JSON responses from the API.

Before the streaming process is initiated, a tweepy Stream Listener instance is created with the API keys provided through the Twitter Developer account. Additionally, the Kafka Producer instance is created and is configured.

A filter has to be selected for streaming to begin pulling data. There are 3 filter types available:

1. Sample Stream: returns a random sample of real-time tweets without any filter. The volume of tweets that can be requested in this way is capped to a maximum of 1% of the total tweets posted at that moment in time.
2. Track (Keyword) Filter: returns tweets that contain specific keywords in their text section or in their hashtags
3. Location Filter: returns tweets posted from a specific location. Location is specified by two sets of coordinates (latitude-longitude pairs). Twitter uses the bounding box model to specify locations [11]. A bounding box forms a rectangle over the world map with the lower left corner being the first coordinate set and the upper right corner being the second coordinate set. The only restriction is that each side does not cover more than 25 miles of distance.

coordinate sets that form a bounding box

After creating the Stream Listener, the Kafka Producer and setting the preferred filter, the streaming procedure is initiated.

For each twitter arriving through the stream, the desired elements are stored to variables. The elements of interest are:

- User ID
- Date and Time of posting
- Tweet Text
- User Location (from user profile)
- Geo-Location (coordinates from which the tweet was posted)
- Retweet Status

The collected elements are then parsed to a single string variable and are pushed to the Kafka cluster through the Kafka Producer.

The stream runs indefinitely (unless an error occurs). A time-out mechanism was added to the script, so that the user can force the stream to close, after running for a certain amount of time.

3.2 Data Pull from Kafka and Storage to MongoDB

In this step of the project, the following actions occur:

1. Tweets retrieval from the Kafka cluster
2. Text Preprocessing and Sentiment Analysis
3. User Location prediction (if it is not provided)
4. Storage of the tweets on MongoDB collection

Although the text preprocessing, sentiment analysis and user location prediction take place in this part of the project, they will be individually analyzed in the next sections of this chapter, as they consist of big and very significant parts of the project that are simply embedded in this one.

As mentioned above, the first action is the retrieval of the tweets from the Kafka cluster. A Kafka Consumer group is created and configured. Before starting to consume the tweets, the MongoDB client is configured, as well as the geopy Nominatim geolocator, which are both essential components for the process that takes place once the consumption has begun.

Once the consumption has begun, the first step for each message is to decompose it to its different elements (user ID, text, location etc.). The preprocessing of the text and the sentiment analysis on it takes place afterwards. If the geo-location is provided, the geopy Nominatim geolocator is used to convert the coordinates to the user location's name. Otherwise, the location prediction algorithm is executed for the tweet.

Once all the tweet elements are on their final form and all the desired processing has occurred, the tweet is inserted into the database.

This process is repeated for all messages pulled from the Kafka Consumer. As long as the stream is actively running and fetching tweets, the consumer is also running and is consuming them. The consumer group is set to automatically shut itself down once it hasn't received any messages from the Producer for 30 consecutive seconds.

Due to limits set by Twitter regarding the volume of information that can be requested from a user's profile which will be explained later, the location prediction algorithm has to pause for some minutes within every 15-minute window. While it cannot operate, the process continues processing tweets that provide geo-location information, until the location prediction is back up.

Two variations of this algorithm are available. A serial version, which processes one tweet at a time, and a multi-threading enabled variation which can process multiple tweets at once and produce results much faster, given the necessary resources.

3.3 Text Pre-Processing

The results of the project and the conclusions that are drawn all depend on the text of the tweets. It is the heart of the data, so it is crucial that we make the most out of it and use it to extract as much information as possible.

Tweets are not written in a formal style. Users want to express their opinions or feelings towards certain subjects they are interested or concerned about, with their only purpose being to get their point through, not caring about how their language and writing is. As a result, it is very common to come across misspelled words or words with random upper-case or lower-case letters, or sometimes words in full upper-case. In addition to this, elements like hyperlinks, emojis and random symbols thrown in the text only introduce noise to the data without adding real value to the tweet. On top of all that, we very frequently come across tweets written in foreign languages. These tweets introduce yet another major issue, since the state-of-the-art NLP tools are developed to work strictly with the English language.

In order to convert the text to its most useful and clean form, the following Natural Language Processing techniques are used:

- Spelling and Grammar correction: misspelled words are corrected, using the Textblob package. As mentioned in the previous chapter, textblob utilizes the NLTK package to perform its functions. For spelling correction, it matches all words found in the text to dictionary entries and replaces misspelled words with the best approximation from the dictionary.
- Text Translation: tweets written in foreign languages are translated using the Textblob package, which utilizes the Google Translate API to provide text translation.
- Text Normalization: all words are converted to lowercase. This way, we prevent the issue of the algorithms used later for data analysis from considering a lowercase and an uppercase version of a word to be two individual words. Hyperlinks, emojis, unnecessary indentation and symbols are also removed from the text.
- Stop-Words Removal: Stop words, such as “I, we, an, was” are useless in terms of context. They introduce noise to the data, rather than giving useful information to work

with. The NLTK package has a solid dictionary of stop words which is used to remove any and all stop words from the tweets.

- Lemmatization: the technique of converting all words to their base dictionary form, otherwise known as their 'lemma' [12]. For example, the word 'better' will be converted to 'good'. This technique helps increase the uniformity of data, which results in less noise and a 'cleaner' information to work with. The NLTK package is used for this technique as well, as it provides one of the most trusted and popular lemmatizers in the field.

3.4 Sentiment Analysis

Sentiment Analysis is also performed right before storing the tweets on the MongoDB collection. The sentiment data is produced and stored in the database alongside the rest of the elements of each tweet.

Textblob is used for sentiment analysis. It provides tools that automatically perform the analysis and return the results. The package provides two scores, given a piece of text. The first score is the polarity score, ranging from -1.0 (very negative sentiment) to 1.0 (very positive sentiment). A polarity score of 0 indicates a sentiment-neutral piece of text. The second score is the subjectivity score, ranging from 0.0 (completely objective) to 1.0 (completely subjective).

Textblob uses NLTK dictionaries that provide sentiment scores for each word. It does, however go deeper than simply calculating the score of a sentence based on the scores of the individual words it contains. It can analyze the sentence structure to marginally increase its accuracy. An example follows:

- The sentence: "This phone is nice." gets a polarity score of **0.6**.
- If we change the sentence to: "This phone is very nice.", textblob can recognize that the word 'very' is a 'polarity booster' for the adjective that follows, and it gives the sentence a polarity score of **0.78**.

- If we change the sentence to “This phone is not nice”, textblob recognizes that the word ‘not’ is a ‘polarity negator’ for the adjective that follows and gives the sentence a polarity score of **-0.3**.

It should be noted that for all three examples above, the subjectivity score returned is **1.0**, since the sentence describes something completely subjective.

3.5 User Location Prediction

As mentioned in the problem definition section of Chapter 2, only 0.85% of the tweets provide geo-location information. The goal of this project is to extract topic and sentiment information per country, a task that requires thousands of tweets for each country in order to return accurate and representative data. This makes the existence of this algorithm a necessity for the project.

The main idea behind the algorithm is that the location of a user can be predicted based on information found on his profile. Four types of information are collected for the prediction to happen:

- the user’s Location profile attribute (if he has provided it)
- the text from the 300 most recent tweets of the user
- the profile location of the 50 most recent followers of the user
- the profile location of the 50 most recent friends (people the user follows) of the user

The number of tweets and friends/followers fetched was chosen in such a way to strike an optimal balance between prediction accuracy and time required for the prediction to be made. The accuracy and performance of the algorithm will be discussed in Chapter 5.

After the recent tweets and the locations on the friends/followers are collected, they are scanned for country and region mentions. To do this, the geograpy package is used, which is essentially a dictionary of countries, regions and places. In addition, custom dictionaries were manually created. These custom dictionaries contain the names of all countries in their native languages

and dialects. This allows for scanning location mentions in foreign languages as well. It should be noted here, that before these dictionaries were created, the googletrans package was used to translate the text in order for location mentions could be traced. This process was very time-consuming, and it would cause the system to reach an API Rate Limit very fast, rendering the process useless. This issue is what led to the creation of the custom dictionaries, which made the whole process much faster.

After gathering all the location mentions, the occurrences of each location are counted. This produces two different location predictions, one from the locations of the friends/followers and one from the text of the recent tweets. We also have the location prediction based on the location the user has indicated on his profile, which is checked as to whether it refers to a real country, because the location field doesn't require the user to enter a real location, it can be whatever piece of text the user desires.

With these three results, the prediction decision is made and is returned by the function, so the location can be added to the database, alongside the rest of the tweet elements. The algorithm behaves as described:

- if the profile location is a valid country name:
    - if the friends/followers location and tweets location return the same location prediction and that prediction is different from the profile location:
        - return the location predicted by the friends/followers and tweets
    - otherwise:
        - return the profile location
- if the profile location is not a valid country name:
    - return the location predicted by the friends/followers

The algorithm behaves this way due to observations made by metrics taken while testing the algorithm. The reasons behind those decisions, which are based on the prediction accuracy of each individual prediction will be discussed in Chapter 5 alongside the prediction accuracy and all metrics produced.

3.6 Sentiment Analytics

This part of the algorithm focuses on producing the sentiment analytics. It chronically takes places after the tweets were stored in the database. As mentioned above, the sentiment analysis data is produced at the same time the tweets are fetched from the Kafka Consumer. In this section, the data is aggregated in order to produce holistic information about each country.

More specifically, an average polarity and subjectivity is calculated for each country. A global average is also calculated.

The matplotlib package is used to visually represent the data. The representation of choice is a scatterplot, where every country is indicated by a point on the plot. The two axes are polarity and subjectivity. The scatterplot was chosen because it allows for all information to be represented in a single plane, it is easily readable and understandable, and it allows for conclusions to be drawn just by looking at it for mere seconds. Scatterplot examples from real data will be presented along with the rest of the results from the project on Chapter 5.

3.7 Topic Analysis

This part of the project focuses on detecting and visually representing the trending topics per country, as well as globally. Just like the sentiment analytics, it takes place after the tweets were stored in the database.

The first step of this algorithm is to split the tweets to different groups, based on the country they come from. Since topic analysis focuses on word appearances, before the machine learning part of this algorithm, all indentation is removed, including spaces, newlines (\n) and tabs.

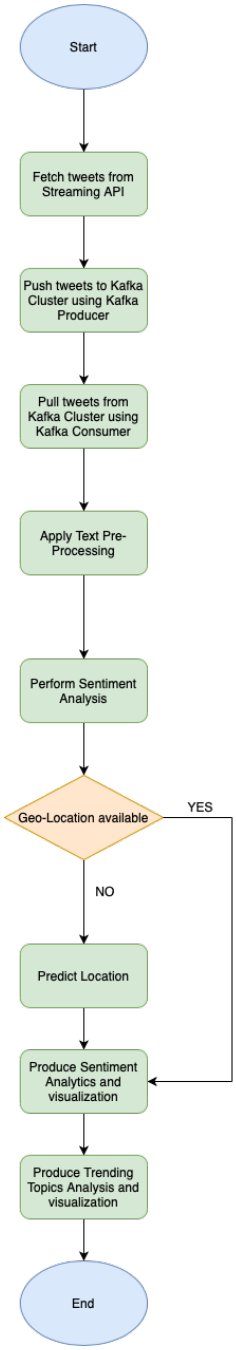
After this grouping and additional text pre-processing, the data is ready to be fitted to a machine learning model in order for the trending topics to be produced. The Latent Dirichlet Allocation model provided by the scikit-learn package is the model of choice, as described in Chapter 2.

The data is vectorized and is then fed into the LDA model, which detects the topics. Trending topics are generated both individually for each country, and globally.

After the model training is complete and the results are ready, the final step is to visually represent the data. This is implemented with the help of the pyLDAvis package which was mentioned in Chapter 2. For each country, a visual representation is generated in an HTML file. The visual representation is an inter-topic distance map that displays all topics on a plot, and for each topic it presents the most frequent words. More details about the representation will be provided in Chapter 5, in which results from topics produced from collected tweets will also be presented.

3.8 Flowchart of System Operation

This flowchart provides a visualization of the pipeline of actions implemented on the project.



Chapter 4 – Implementation Details

In this chapter, details regarding the implementation of several parts of the code will be discussed.

While the previous chapter primarily focused on explaining what each part of the algorithm does, this one will focus on how it does it from a programming perspective, as well as provide descriptions for parts of code that are transparent to the main workflow of the algorithm but play a vital role in it.

4.1 Data Fetch from Twitter and Push to Kafka

Tweets are fetched using the tweepy package as mentioned. While the responses are fetched in JSON format, because we only wish to store certain elements from those responses, the response is decomposed and only the desired elements are stored in variables. The string with all the desired elements is then constructed. The Kafka Producer is configured to encode the generated messages in JSON format before pushing them to the Kafka Cluster. The various advantages of the JSON format were previously mentioned, and they are the reason we choose this format for parsing the data. So, the general scheme used is the following:

- fetch JSON response
- decode JSON and extract desired elements
- construct new message with only the required elements in it
- re-encode in JSON format and push data to the next phase

Twitter would only allow a maximum of 140 characters per tweet. This restriction was modified to allow up to 280 characters in 2017. By default, the API fetches 140 characters per tweet, in order to maintain compatibility with already existing applications that used the API. In this project, we would like to get the entire tweets, if they are longer than 140 characters. The tweepy streaming client is thus configured to work in "extended mode", which includes a "full text" element with up to 280 characters of text, in addition to the default "text" element that only displays the first 140 characters [13]. Another implementation detail about this procedure is the fact that if the tweet does not exceed 140 characters, the "full text" element doesn't exist and returns an "Attribute Error" if requested. This led to the addition of try-except clauses in the code in order to fetch the element only if it exists.

Many tweets are retweets. In this case, because the "text" attribute would be empty or would simply contain a comment from the user on the item he re-tweeted, additional code was added to fetch the text from the original tweet that was retweeted, so we have the full information about the tweet.

4.2 Data Pull from Kafka and Storage to MongoDB

As mentioned previously, the messages are encoded to JSON format before being pushed to the Kafka cluster by the Producer. Thus, it can be easily implied that the Kafka Consumer is responsible for decoding the JSON message.

In order to make good use of the advantages of the NoSQL characteristics of MongoDB, elements that are null (for example if a user hasn't added a location profile) are not included on the record that will be pushed to the database.

4.3 User Location Prediction

The 'extended mode' is used here again, in order to get up to the maximum of 280 characters per tweet.

After collecting the most recent tweets, the next step requests the list of followers and friends of the user, in order to collect their location. The Twitter API has various rate limits on the number of specific types of request that can be made. The rate limits are reset every 15 minutes. An enterprise developer account doesn't have these restrictions, but since this project uses a normal developer account, the requests have to be made in such a manner that doesn't use up more requests than they absolute necessary.

A rate limit is in place, restricting how much friends/followers can be fetched from a user profile with the corresponding requests. The algorithm was designed to omit the prediction for users with over 5000 followers or friends, as fetching such volumes of data causes the rate limit to be reached very fast, which stalls the whole system from running for 15 minutes.

While the algorithm searches for country name appearances, after several testing sessions of the algorithm, a big error was noticed for the US, where people would use their specific state as their location, rather than using the country name. UK was a similar case, where people would type the specific region (Scotland, Wales etc.) rather than 'United Kingdom'. In order to counter this issue, text files with US states and UK regions records were created and are

used by the algorithm to correspond the states/regions to their respective country, thus improving the accuracy for those two countries.

As mentioned in the description of this algorithm, dictionaries with all countries in their native languages are used to relieve the system from having to translate every single tweet and every single user location written in a foreign language. These dictionaries were created with data scraped from the Wikipedia site containing all countries and their capitals written in the native languages and dialects [14]. Tweets are cross-checked with the dictionaries, and in case a country name in a foreign language is detected, that name is converted to its English alternative. This way, the country name can be detected by the geograpy package and be added to the list keeping track of the occurrences of each country name which will then be the criterion for making the prediction.

The screenshot below shows the structure of the dictionaries:

```
# dictionary for all countries in native language
native = {"Afghanistan": ["افغانستان"], "Albania": ["Shqipëria"], "Algeria": ["ⵍⵣⵣⴰⵢⵔ", "الجزائر"],
          "American Samoa": ["Amerika Sāmoa"], "Armenia": ["Հայաստան"], "Austria": ["Österreich"],
          "Azerbaijan": ["Azərbaycan"], "Bahrain": ["البحرين"], "Bangladesh": ["বাংলাদেশ"], "Belarus": ["Беларусь"],
          "Benin": ["Bénin"], "Bhutan": ["འབྲུག་ཡུལ"], "Bolivia": ["Buliwya", "Wuliwya", "Volívia"],
          "Bosnia and Herzegovina": ["Босна и Херцеговина"], "Brunei": ["بروني"], "Bulgaria": ["България"],
          "Burundi": ["Uburundi"], "Cambodia": ["កម្ពុជា"], "Cameroon": ["Cameroun"], "Cape Verde": ["Cabo Verde"],
          "Central African Republic": ["Ködörösêse tî Bêafrîka"], "Chad": ["تشاد"], "China": ["中国", "中华人民共和国"],
          "Comoros": ["Komori", "جزر القمر", "Comores"], "Congo": ["République du Congo", "Repubilíki ya Kongó"],
          "Cook Islands": ["Kuki Airani"], "Croatia": ["Hrvatska"], "Curaçao": ["Kòrsou"], "Cyprus": ["Κύπρος"],
          "Czech Republic": ["Česká republika", "Česko"], "Denmark": ["Danmark"], "Djibouti": ["جيبوتي"],
          "Dominican Republic": ["República Dominicana"], "Egypt": ["مصر"],
          "Equatorial Guinea": ["Guinea Ecuatorial",
                                "Guinée équatoriale", "Guiné Equatorial"], "Eritrea": ["ኤርትራ", "إرتريا"],
          "Estonia": ["Eesti"],
          "Ethiopia": ["ኢትዮጵያ"], "Faroe Islands": ["Føroyar", "Færgerne"], "Fiji": ["Viti", "फिजी"],
          "Finland": ["Suomi"], "French Guiana": ["Guyane"], "French Polynesia": ["Polynésie française"],
          "Gabon": ["République gabonaise"], "Georgia": ["საქართველო"], "Germany": ["Deutschland"],
          "Ghana": ["Gaana", "Gana"], "Greece": ["Ελλάδα"], "Greenland": ["Kalaallit Nunaat", "Grønland"],
          "Guam": ["Guåhån"], "Guinea": ["Guinée", "Gine"], "Guinea-Bissau": ["Guiné-Bissau"],
          "Haiti": ["Haïti", "Ayiti"], "Hong Kong": ["香港"], "Hungary": ["Magyarország"], "Iceland": ["Ísland"],
          "India": ["ভারত", "ভারত", "ભારત", "भारत", "ಭಾರತ", "भारत", "ഭാരതം", "ଭାରତ", "ਭਾਰਤ", "भारतम्", "இந்தியா",
                    "భారత దేశం"], "Iran": ["ایران"], "Iraq": ["العراق"], "Ireland": ["Éire"],
          "Isle of Man": ["Ellan Vannin"], "Israel": ["ישראל", "إسرائيل"], "Italy": ["Italia"], "Japan": ["日本"],
          "Jersey": ["Jèrri"], "Jordan": ["الأردن"], "Kazakhstan": ["قازاقستان", "Казахстан"],
          "North Korea": ["조선 / 朝鮮", "북조선"], "South Korea": ["한국 / 韓國", "남한"], "Kuwait": ["دولة الكويت", "الكويت"],
```

A python dictionary entry has the following structure: Key: Value. In this case, the key is the country name in English and the value is a list with the country name in all native languages. This part of the algorithm finds and replaces values (country names in foreign languages) in the text, with their respective keys (country names in English).

After collecting and counting the occurrences of each country name, the most occurred country name from the user's tweets and from his followers/friends, as well as the profile location (if it is provided) are individually validated by the ISO3166 file of standards, which contains the names, official names and country codes for all entities registered as countries worldwide [15]. After successful validation, the prediction result is returned.

4.4 Sentiment Analytics

This part of the system also relies heavily on python dictionaries for its operations. The dictionary entries have the following structure: Key (country name): [Polarity score, Subjectivity score, Tweets count].

For each tweet in the database, its polarity and subjectivity score are summed up with the current scores for its country of origin, and the tweets count is incremented by 1 for the dictionary entry of the specific country.

It should also be noted here, that tweets with a subjectivity score ranging from 0.0 to 0.1 (almost or completely subjective) are not taken into account when calculating the national and global average polarity and subjectivity scores. The reasons for this choice will be explained in the next chapter where the results and the observations about them will be presented.

4.5 Topic Analysis

Just like Sentiment Analytics, the python dictionaries are the dominant data types used. A dictionary entry is generated for each country, containing all tweets posted by users of the respective country.

Before fitting the data to the LDA model, the words are scanned through, and appearances of the name of the country are removed. The same process occurs for collection words. The

collection words are the ones used to filter the Tweets Stream. More details regarding this choice will be discussed in the next chapter.

The machine learning models work strictly with numeric data, so raw text had to be converted to representative numeric values. The automated vectorization tool provided by the scikit-learn package was used. This vectorizer applies the ‘Bag of Words’ technique. This technique creates a dictionary with all the words present in the training data. Then, every data entry is converted to an integer list. Every index in the list has a ‘0’ or a ‘1’ value in it. The value is ‘1’ if the word mapping to the corresponding list item exists in the data entry, or ‘0’ if it doesn’t. The ‘1’ value is incremented for every additional appearance of the respective word.

Chapter 5 – Results Presentation and System/Algorithms Evaluation

In this chapter, all algorithms and components of the system will be evaluated, and accuracy metrics will be presented.

Choices made in order to improve accuracy and bypass some imposed difficulties will be discussed.

Metrics regarding the performance and the resources required for the system to operate will also be described, as well as the resources allocation demands.

5.1 Hardware Performance and Time/Storage Space Requirements

When dealing with massive batches of data, measuring and optimizing the processing and storage requirements is of crucial importance.

Find below the important hardware specifications of the devices used for running and testing the project:

MacBook Pro 15-inch 2018:

- 2.6GHz 6-core Intel Core i7, Turbo Boost up to 4.3GHz, with 9MB shared L3 cache
- 16GB RAM memory (DDR4 2400MHz)
- 512GB SSD (3.2GB/s sequential read/write speed)
- AMD Radeon Pro 560X GPU (4GB GDDR5 memory)

Internet Connection specifications:

- 120Mbps download / 12Mbps upload
- ~9ms latency (ping)
- IEEE 802.11 ac Wi-Fi (1300Mbps theoretical maximum transfer speed)

Twitter Streaming API:

An average of 6000 tweets are posted every second [16]. As mentioned, the Streaming API fetches a maximum of 1% of the total tweets posted at the time, which translates to roughly 60 tweets fetched per second.

A single tweet (as fetched by the Streaming API in JSON format) has a file size of about 4KB. An internet speed of 1.92Mbps is capable of taking full advantage of the maximum streaming traffic (assuming the 60 tweets per second and 4KB file size per tweet average values and no packet loss). This makes the internet connection used more than capable of offering maximum operating performance.

MongoDB:

The average size per database entry for each tweet with all the data produced for it is 1.1KB. A 20MB collection can hold about 18930 tweets.

Sentiment and Topic Analysis:

Sentiment Analysis produces a single image (scatterplot) in PNG format requiring about 130KB of space.

Topic Analysis produces an interactive HTML file for each country. The size of each file is 45KB on average.

Time Requirements per Project component:

- Data Fetch from Twitter and Push to Kafka: it takes an average of 17ms to fetch a single tweet through the Streaming API, process it to get the desired elements from it and push it to the Kafka Cluster.

- Data Pull from Kafka and Storage to MongoDB: the time required just for pulling the tweet from the Kafka cluster and pushing it to the database is extremely small (<1ms). It is the processing of the tweet that takes a significant amount of time to complete. The time required for each individual part of the processing is mentioned below.

- Text Pre-Processing: the entire pre-processing procedure takes 3 seconds (2962ms) on average.

- Sentiment Analysis: about 36ms required for performing sentiment analysis on a single tweet.

- User Location Prediction: the entire process takes an average of 56 seconds (56136ms) per user. It is by far the most time-consuming process of the system and it is the main reason that the tweet processing algorithms were developed to take advantage of parallel execution.

- Sentiment Analytics: an average of 2.8 seconds (2758ms) is required for the sentiment analytics to be calculated and for the visualization to be generated. The reason this process takes so little time is because the sentiment analysis is performed before storing the tweets to the database, and this part of the algorithm only has to aggregate the data, compute the averages per country and export the scatterplot.

- Topics Analysis: Fetching the content from the database and performing the required processing for grouping the data by country and parsing them so they can be fed to the LDA model takes an average of 49 seconds (4862ms). Running the machine learning procedure and exporting the HTML file with the topic visualization for each country takes an average of 6 seconds (5934ms) for each country. It should be noted, though, that the time required for this procedure varies between countries depending on how many tweets exist per country. For a dataset of 20000 tweets, the time consumed for a country varies from 311ms all the way up to 224 seconds (224315ms) to create the global trending topics. The total execution time for the mentioned data is 7 minutes and 49 seconds (468584ms).

5.2 Location Prediction Algorithm Performance Metrics and Results

The algorithm generates 3 results in order to determine to user's location, as mentioned in the previous chapters: the user's location (from his profile, if he has provided it), the prediction based on his friends/followers and the prediction based on his tweets. The accuracy metrics below were produced based on a dataset of 7000 tweets fetched through the Streaming API. For each prediction, the algorithm uses the 300 most recent tweets of the user and his 50 most recent friends and 50 most recent followers. These numbers were chosen in order for an overall accuracy of over 80% to be achieved, which is the goal accuracy threshold. It can be easily inferred that increasing the numbers results in higher accuracy but requires more time per prediction, and that decreasing the numbers results in lower accuracy but a faster calculation.

- Percentage of Users Predicted Correctly: **83.88 %**
- Percentage of Users Predicted Correctly by all 3 ways: **10.33 %**
- Percentage of Users Predicted Correctly by 2 ways: **58.26 %**
- Percentage of Users Predicted Correctly by Profile Location: **22.31 %**
- Percentage of Users Predicted Correctly by Followers/Friends Location: **69.42 %**
- Percentage of Users Predicted Correctly by Tweets Location: **40.08 %**
- Percentage of Users for which Profile location could not be predicted: **74.79 %**
- Percentage of Users for which Followers/Friends location could not be predicted: **4.55 %**
- Percentage of Users for which Tweets location could not be predicted: **25.62 %**

The overall accuracy is close to 84%, which is a very satisfactory value. Breaking down the results from the individual metrics helps us draw many interesting conclusions.

For 22% of users, their location was predicted correctly by their profile location. The profile location could not be predicted for 75% of the users, either because they didn't provide it or because they provided an invalid location. That means that out of the 25% of users that actually filled in the profile location with valid information, 22% was actually correct. That makes it by far the most trustworthy metric.

The prediction from the followers and friends also has rather high accuracy, predicting correctly 69% of the tweets and it can return a prediction on about 95% of the time.

The least reliable prediction proved to be the one from the user's tweets, achieving only about 40% accuracy. About 26% of the tweets could not get a prediction from the tweets at all, so we have 40% out of the 74% that could be predicted, which is not a disappointing fraction of correct predictions.

It should be noted that adding the dictionary with the countries in their native languages was really helpful and caused a drastic increase in overall accuracy, as text written in foreign languages was basically unusable and would result in prediction failure. For example, the prediction accuracy from the user's tweets was just about 23%.

While the prediction accuracy and the execution time are on satisfactory levels, the Rate Limits of the API cause a problem. The Twitter API has Rate Limits in place regarding how many friends/followers and how much user profile information can be fetched, which are the key elements required for the algorithm to operate. More specifically, in every 15-minute window, there can be made: 900 requests for user profile information and 15 requests for friends/followers lists, with each list containing up to no more than 5000 friends/followers. The first mentioned rate limit causes a problem because a total of 100 user profiles are requested per prediction (the profile of the user, 50 followers and 50 friends).

The latter rate limit causes a problem because it doesn't allow the algorithm to run for users with over 5000 friends/followers and because a single prediction makes 2 requests for the lists of followers and friends.

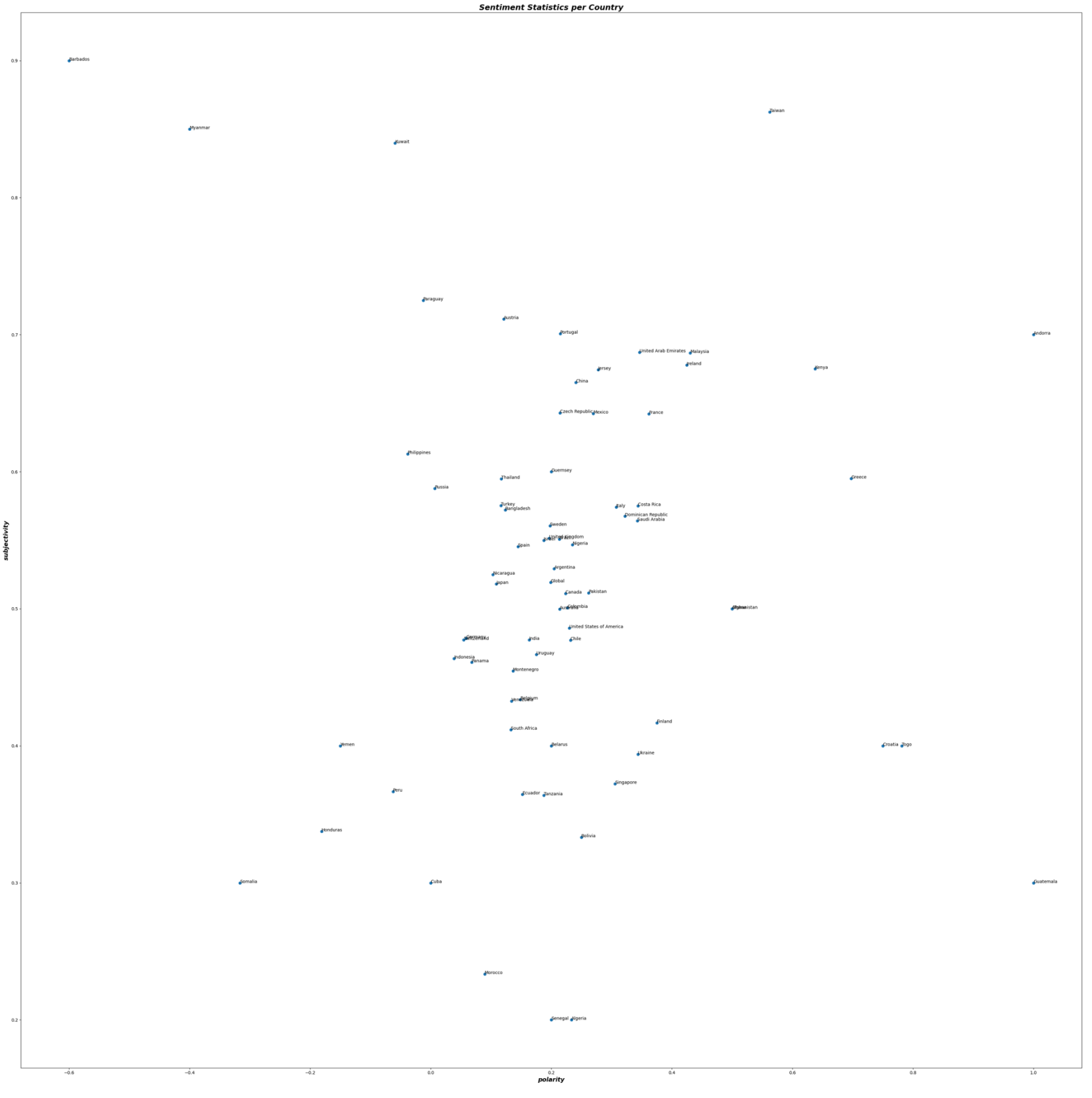
By doing the math, it can be concluded that only 8 requests can be made per 15-minute windows, maybe 9 in case where a user have less than 50 followers or friends, so less than 101 requests are required for the prediction in such cases. This consists of a performance bottleneck for the algorithm, since it is capable of making about 16 predictions every 15 minutes, but with these rate limit restrictions, only half of those can be made. An easy solution would be the acquirement of an enterprise developer account which doesn't have these rate limits, as they only apply to standard developer accounts like the one used for this project.

5.3 Sentiment Analysis Results / Evaluation

Tweets with a subjectivity score ranging from 0.0 to 0.1 were omitted when generating sentiment analytics because it was noticed that the vast majority of tweets that were so objective also had a polarity score of about 0.0, reflecting no sentiment. Most of these tweets were simply mentioning an event that occurred and would many times also include a hyperlink to a news site about that event. The big volume of such tweets would only introduce excessive noise for the analysis. When running the analytics algorithm without omitting this data, the polarity score of most countries displayed a tendency of leaning towards neutral polarity (0.0), so the results were not representative.

Performing the sentiment analytics algorithm on different datasets with several thousands of tweets, the average global polarity score is about **0.198** (a little above positive) and the average global subjectivity score is about **0.52**. These results are more realistic. The fact that the sentiment is, in general, almost neutral is because there were many slightly positive and many slightly negative tweets. More specifically, the standard deviation for polarity was **0.30**, while the standard deviation for the subjectivity was **0.21**. This pattern could possibly be linked to the coronavirus pandemic which was affecting the entire globe while the datasets were created.

It should also be noted that the country distribution of tweets is extremely uneven. The reason behind this is the fact that Twitter is very popular in some countries, and not so popular in others. For example, Russia has a population of about 146 million from which 23.5 million have Twitter accounts, while Japan which has a smaller population of about 126 million has around 48.5 million Twitter accounts, which is more than double [17]. Even in very big datasets with around 20000 tweets, some countries have less than 10 tweets. Countries with a small number of tweets are not included in the scatterplots, as they are not representative of the sentiment in the entire country when they are so few in size.

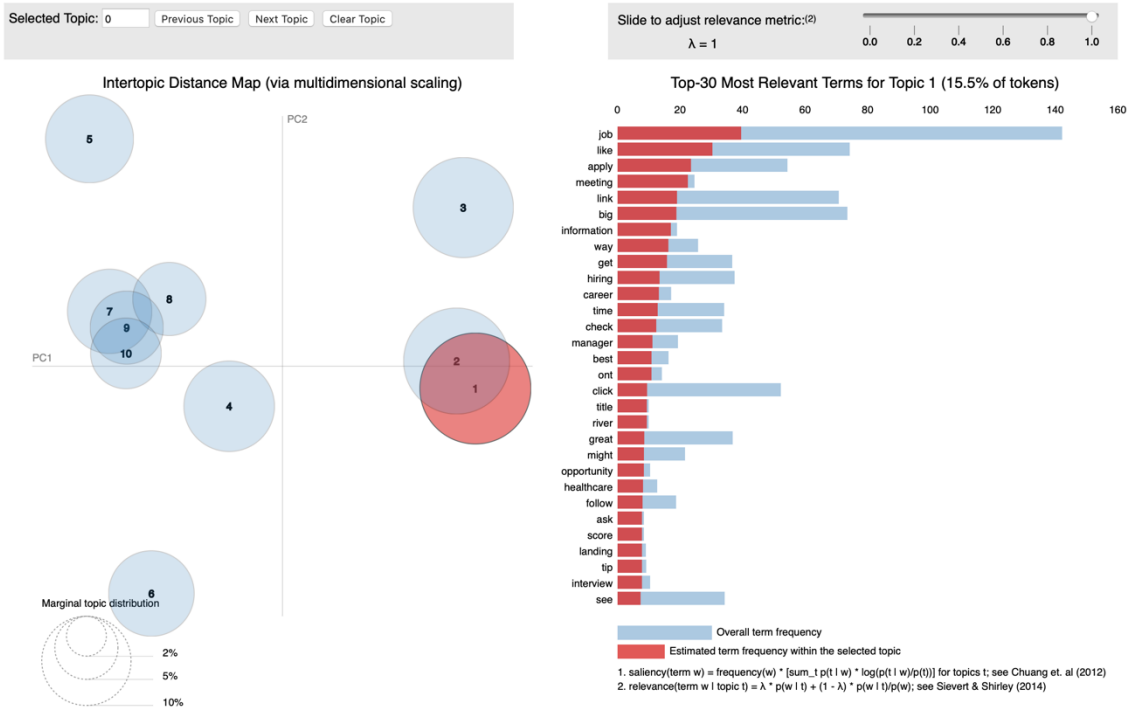


Above you can find a scatterplot produced from a 10000-tweet dataset obtained by random sampling (without any filters applied to the stream). The x-axis represents the polarity and the y-axis represents the subjectivity.

In this point, a comment should be made regarding the sentiment analysis. Even though very popular and state-of-the-art tools were used, the sentiment analysis and Natural Language Processing still hasn't reached a very high level of maturity. For instance, it is still tricky and most of the times impossible to detect humor or sarcasm in a piece of text.
Emojis most of the time express the exact sentiment we have about a post we make, but no state-of-the-art tools for Python (as of the time of writing this dissertation) are developed to examine emojis for sentiment. While there exist packages with sentiment values for the emojis, they work solely for emojis, none of them are integrated in a complete sentiment analyzer.

5.4 Topic Analysis Results / Evaluation

The algorithm is set by default to detect 10 topics for each country, but any number of topics can be chosen. It should also be noted that the algorithm supports setting a minimum tweet threshold and omits the calculation for countries whose tweets in the given database are below the threshold (default threshold value is 100 tweets). This is done for the same reason as the suchlike feature for the sentiment analysis (a small sample cannot accurately represent a country).
Find below a screenshot from a visualization, which will be later discussed in detail.



Starting with the contents of the visualization: This visualization was proposed by the Stanford University [18]. On the left side, we have an Inter-topic Distance map that uses multidimensional space to represent the distance between the topics. We have two principal components (PC1 and PC2) as the axes.
On the right side, we have the 30 words with the highest overall frequency within the topic, accommodated by the term frequency of each word within the topic which is displayed with the red line over the blue line that indicates the overall frequency. The relevance metric (λ) can be adjusted according to user preference. It represents the probability of the words being categorized within the selected topic.

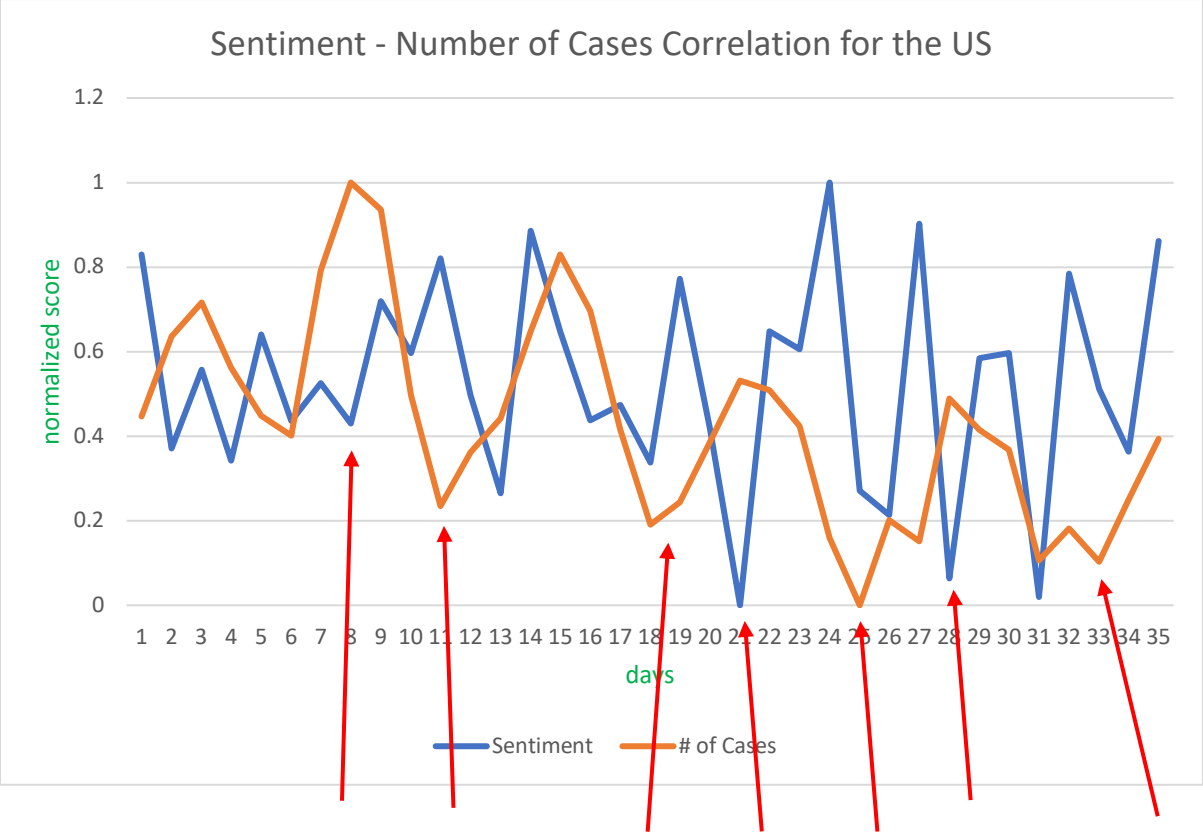
The screenshot above is from the visualization of trending topics in the United States of America (USA) from a dataset of 3000 tweets fetched in the week of 11-17th of April 2020. By taking a closer look at the frequent words, one can easily discern that the selected topic is about jobs, as we can see words like: job, apply, career, manager, opportunity, interview.

5.5 Use Case: the COVID-19 Pandemic

The Covid-19 pandemic that started spreading throughout the world at the beginning of 2020 has significantly disrupted the normal way of living in most countries around the globe, causing traffic restriction measures and lockdowns.

The general sentiment around the globe has definitely shifted during this period, especially in countries with high numbers of cases and deaths. Since the necessary components have been developed, this situation gives us the opportunity to examine whether the sentiment in Twitter posts can be correlated with the rise of cases in different countries.

A big dataset of tweets was assembled in order to be able to extract accurate results. The example that will be displayed and analyzed below, is from the United States for a time period of 35 consecutive days, starting from the 14th of April 2020.



In order to represent the data in the same chart and convert them to a comparable format, they were normalized within the 0-1 range using the Rescaling (min-max normalization) technique which applies the following formula to all the data points: $x' = \frac{\text{x}-\min(\text{x})}{\max(\text{x})-\min(x)}$

From the chart, we can clearly see that in all days when there was a big increase in number of cases, there was a big decrease in sentiment polarity. The opposite is also true (big decrease in number of cases = big increase in sentiment polarity). This is pointed out on the chart, where it can be clearly seen that for every big spike on the cases line, there is a big spike towards the opposite direction for the sentiment line.

But even though the big drops in sentiment polarity do indeed indicate a big increase in cases, they cannot reflect the small increases and small drops of cases, as seen from the parts of the chart where there is no big variation in number of cases. In these cases, the sentiment polarity sometimes increases in small increase of cases or decreases in small decreases of cases.

The conclusion from the entire use case is that by detecting big increases or decreases in sentiment polarity, we can predict big increases or decreases in the number of cases. Of course, in order for as much accuracy to be achieved as possible, we need as big of a dataset as possible. This use-case clearly shows the significance of the information produced by this project, and the usefulness of the information that can by produced by applying sentiment analysis on big batches of social networking content in general.

Chapter 6 – Conclusions and Future Work

In this chapter, the conclusions from the development and the final result of the entire Diploma Project will be discussed, as well as the future work that can be put into it, to extend its capabilities and improve the already implemented features.

6.1 Conclusions

The process of studying various publications about the field of Big Data and Data Analysis, as well as the process of developing the various algorithms required for the goals of the project to be achieved has pushed me to draw many conclusions regarding both the field, the features that were implemented and the results from them.

The system that was developed has successfully fulfilled the goals of the project. It can produce accurate sentiment and trending topics analytics for every country individually. It solves the initial problem of the extremely small percentage of geo-located tweets with the prediction algorithm and it visualizes the produced information in such a way that it makes it easy for any individual to interpret it and understand the results without the need of having any computer science or statistical background.

The huge importance of time and space efficiency when having to interact with such massive amounts of data was mentioned many times throughout the dissertation. Many sections of the code and many techniques used were modified multiple times in order to achieve as good performance and as little storage space requirement as possible. With that being said, there is always room for improvement on this aspect, but the time and storage space required by the system in its final form, compared to how it was at the beginning of the final semester was a surprise, as certain procedures turned out to be up to 2 or 3 times faster.

It is important to notice that every single component was developed in such a way so that it can be executed individually, or it can be used as a standalone script in any other Python project. For example, the “text pre-processing” component can be executed with any String input and it will return the text with all the mentioned NLP techniques applied to it, alongside the subjectivity and polarity sentiment values for it. The Location Prediction script can also be executed to return the predicted location of a twitter user just by passing a string parameter of a Twitter user ID. The same standalone functionality applies to each and every component.

It is important to note that the whole platform was developed in such a way to be able to handle massive volumes of data. Components such as Kafka for the stream management and MongoDB for storing the data were chosen for this particular reason. Also, the time-consuming parts of the process, which is the text processing and the location prediction were designed to be executed in parallel, so big amounts of data can be processed simultaneously instead of serially, given the necessary workstation setup.

In this project, I got the opportunity to interact with many powerful and state-of-the-art tools and I got to combine many of them and make them work in full harmony. Having to deal with big volumes of real data and gain valuable experience in this field of Computer Science that has skyrocketed in the last couple of years has helped me develop an important skillset that will definitely be of great help in my career.

6.2 Future Work

The outcome of this project is satisfying and the goals that were set have been met. But this project can be expanded and made better in many ways.

First of all, many Artificial Intelligence tools that were used still have room for improvement. One of them is the sentiment analyzer. As mentioned before, emojis are not taken into account in the sentiment score, which is a shame considering how much they can tell us about sentiment. And even though, on most text, it does a good job of capturing the sentiment, it is not always accurate. Also, many posts contain images or GIFs, which could also help determine the sentiment of a tweet. No state-of-the-art sentiment analysis tool can take advantage of such information, as of now.

Many NLP techniques are still only available in English (like the Lemmatization). This has forced us to translate text written in foreign languages, but that is not an optimal solution, as even the best translating systems (like Google Translate) many times provide very inaccurate translations.

Last but not least, due to the Rate Limits set by the Twitter API, the volume of data that can be requested at any given time is restricted. The application has the capability to handle much more data at a single time than the one it handles in its current state, due to these limits and due to limited hardware resources. This is the reason why the Topic and Sentiment analysis are offline algorithms. If the full power could be harnessed from the Twitter API, the Topic and Sentiment analysis components could be written as online algorithms that provide live visualizations. This would possibly be implemented with the integration of the Apache Spark framework into the project.

Bibliography

[1] Luke Sloan, Jeffrey Morgan (6 Nov 2015), "Who Tweets with Their Location? Understanding the Relationship between Demographic Characteristics and the Use of Geo-services and Geotagging on Twitter", retrieved from: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4636345/

[2] Salman Aslam (10 Feb 2020), "Twitter by the Numbers: Stats, Demographics & Fun Facts", retrieved from: https://www.omnicoreagency.com/twitter-statistics/

[3] Twitter Developer API Documentation, "Filter real-time Tweets", retrieved from: https://developer.twitter.com/en/docs/tweets/filter-realtime/api-reference/post-statuses-filter

[4] Twitter Developer API Documentation, "Tweet objects", retrieved from: https://developer.twitter.com/en/docs/tweets/data-dictionary/overview/intro-to-tweet-json

[5] Katarina Grolinger, Wilson A Higashino, Abhinav Tiwari, Miriam AM Capretz, "Data management in cloud environments: NoSQL and NewSQL data stores", retrieved from: https://journalofcloudcomputing.springeropen.com/track/pdf/10.1186/2192-113X-2-22

[6] "MongoDB", retrieved from: https://www.mongodb.com

[7] "Apache Kafka", retrieved from: https://kafka.apache.org/intro

[8] "Python", retrieved from: https://www.python.org/about

[9] "PYPL Popularity of Programming Language", retrieved from: http://pypl.github.io/PYPL.html

[10] "pyLDAvis", retrieved from: https://nbviewer.jupyter.org/github/bmabey/pyLDAvis/blob/master/notebooks/pyLDAvis_overview.ipynb

[11] Twitter Developer API Documentation, "Tweet Location objects", retrieved from: https://developer.twitter.com/en/docs/tweets/data-dictionary/overview/geo-objects

[12] Stanford University (07 Apr 2009), "Stemming and lemmatization", retrieved from: https://nlp.stanford.edu/IR-book/html/htmledition/stemming-and-lemmatization-1.html

[13] "Tweepy", retrieved from: http://docs.tweepy.org/en/latest/extended_tweets.html

[14] Wikipedia, "List of countries and dependencies and their capitals in native language", retrieved from: https://en.wikipedia.org/wiki/List_of_countries_and_dependencies_and_their_capitals_in_native_languages

[15] ISO Organization, "ISO 3166", retrieved from: https://www.iso.org/obp/ui/#search/code/

[16] Internet Live Stats, "Twitter Usage Statistics", retrieved from: https://www.internetlivestats.com/twitter-statistics/

[17] J. Clement (24 Apr 2020), "Leading countries based on number of Twitter users as of April 2020", retrieved from: https://www.statista.com/statistics/242606/number-of-active-twitter-users-in-selected-countries/

[18] Carson Sievert, Kenneth E. Shirley (27 Jun 2014), "LDAvis: A method for visualizing and interpreting topics", retrieved from: https://nlp.stanford.edu/events/illvi2014/papers/sievert-illvi2014.pdf

Complete Paper Word Count: 11,858 words