Assignment 2 Report

Social Media Sentiment analysis on Victorians

By Xiangwen Deng, Yuqi Ma, Jiaying Wang, Meiya Lian, Zixin Wang

Abstract

Our project implements a system that provides insights into citizen's emotions across Victoria using the Twitter data and datasets provided by Aurin, to discover correlations between people's mood and age, as well as the area, population density, income level, and periods during a day. This document provides a detailed introduction and user guide of the system, from system architecture design to the system's deployment and its usage.

Table of contents

1. Data Collection

1.1 Aurin Data Collection

The data of Victoria population density, number of residents, and income levels in each Victoria Local Government Area are collected from the AURIN platform and used in this project. Local Government Area is used to categorize different areas in Victoria for analysis in this project as using suburb division is too detailed and using greater capital city division is too broad. These data are used to compare with Twitter sentiment analysis. All of the data are downloaded in JSON format as it is convenient for data extraction through programming.

1.2 Twitter Data Collection

Two libraries are used for Twitter data collection in this project, details and reasons are introduced in the two subsections below.

1.2.1 Tweepy

The Tweepy Python library is used to access the official Twitter API to retrieve tweet data. Both the standard search and the streaming API are used in the project, as parts of the Tweet collector component in the system (refer to section 2.1). The standard search API is retrospective which collects matched tweets from the past given criteria. However, it can only search against a sampling of recent Tweets published in the past 7 days. Also, there is a rate limit of 120 requests per minute, for an average of 2 queries per second (QPS). Compared to the standard search API, the streaming API allows the collector to harvest real-time data with low latency and it does not have a rate limit. However, using this API alone is not efficient enough for data retrieval, as it only returns a small fraction of all the tweets tweeted at a time due to the in-built threshold.

1.2.2 Twint

As stated above, there are various limitations in using the official Twitter API. In order to maximise the amount of data collected, a tool named Twint is used in the system. It is an advanced Twitter scraping tool written in Python that doesn't use Twitter's API, which helps the collector to bypass the limitations stated in section 1.2.1.

2. System Architecture and Design

2.1 System Architecture Diagram

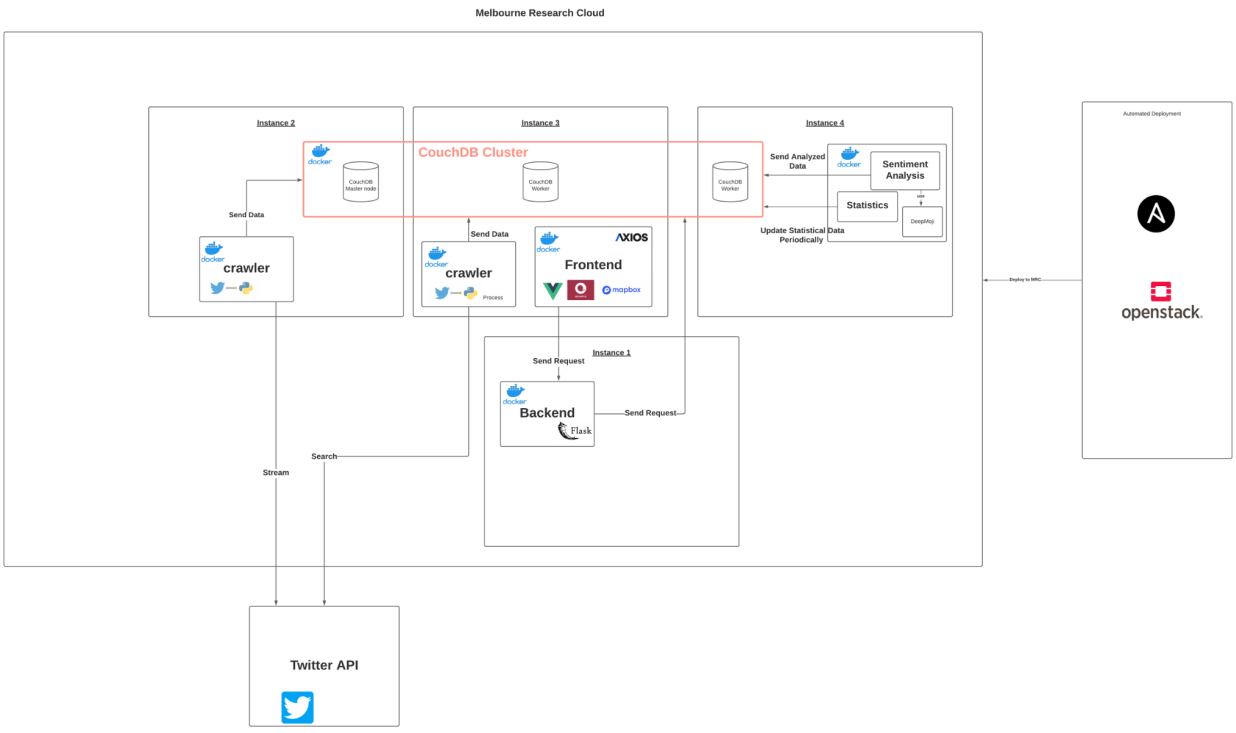


Figure 1. System Architecture Diagram

2.2 Tweet Collector

The crawler is a python program that takes command-line arguments, to either search or stream the tweet data around an area into the database. As displayed in the architecture, the crawler is able to deploy and run in multiple instances at the same time. In the current system, the crawler is deployed and running on two instances at the same time, one is using the search method to collect data and the other one is using the stream method. What enables this is that before storing tweet data, the collector always checks if the tweet already existed in the database by *tweet_id*, which is a unique id allocated to each tweet by Twitter. In this way, the crawler can be scaled up/down based on available resources without getting document insertion conflicts/redundancy issues.

2.2.1 Preprocessing

Besides checking if the tweet collected is already stored, there are also some steps to process the raw data before storing it in the database, which not only reduces the data size but also makes the later analysis steps easier.

Firstly, the *tweet_id* is used as the _id value of each tweet document to replace the default UUID value to assist the redundancy check. Also, retweets are removed, by checking if the collected Twitter Status objects contain the *retweet_status* attribute (normal tweets do not have this attribute in their Status objects). Furthermore, tweets with languages other than English are filtered, through checking the *lang* attribute of the Status object. Lastly, each tweet will be categorized into one Victoria Local Government Area. This is done by checking the location labelled in each tweet, either a coordinate or a bounding box. The Victoria Local Government Area data taken from Aurin provides each area's location represented by a multipolygon list. Using the python shapely library (a package for manipulation and analysis of planar geometric objects), the tweet can be categorized into different areas provided the above information.

Eventually, each tweet will be stored in a document with the following format:

```
tweet_obj = {
            "type": "tweet",
            "Text": "text" , # the content of the tweet
            "hashtags": [], # this attribute is not used in analysis, hence, ignore
            "time": "%d/%m/%Y %H:%M:%S", # the creation time of the tweet
            "location": location, # tweet location
            "ts": "%d/%m/%Y %H:%M:%S" #creation time of the doc
            }
```

2.2.1 Twint assistance with data collection

In the real-world scenario, we found out that by going through the above selection of the tweets, the collection rate is extremely low. To be more specific, only tweets that are not retweets, have geo-tagged which are labeled within Victoria and written in English, can be stored in the database. Along with the Twitter API limitation, the system cannot collect enough data for analysis before the deadline. Therefore, another script is written to assist the data collection. The script called “GetOldTweets.py” uses the Twint library introduced in section 1.2.2 to collect data in the past years for analyzing scenarios. The script is not deployed in the system as it is an unofficial approach to harvest tweets and hence it is not a stable and guaranteed component for the tweet collector.

2.3 CouchDB Database

2.3.1 Advantage of CouchDB

Big data is quite challenging due to the volume, velocity, variety and veracity. Since the relational Database Management System(DBMS) like MySQL cannot handle well in variety and veracity for big data, we decided to use NoSQL DBMSs.

CouchDB is one of the popular NoSQL DBMS. It is a document-oriented DBMS and stores the data as structured documents. CouchDB is open-source and has MapReduce queries. The HTTP API makes it easy to use and interact. It is also easy to set up a cluster. By setting the couchdb cluster, the availability of the dataset will be improved.

2.3.2 MapReduce Functions

The design documents are defined and uploaded to the databases once they are created (using the prepdb/prep_views.py). In this way, the B-Tree for each view will be generated before the data grows, which guarantees the views can be returned within a short time even for extremely large datasets and therefore avoids the server timeout error for processing the views.

2.3.3 Data Storage

There are two databases in the system, one is called **vic_tweets**, for storing the collected tweets and the information of each area retrieved from the Aurin platform. The other one is called **analysis**, for storing the analysis of each tweet as well as two statistics documents which will be explained in detail in the next section.

2.4 Data Analyzer/ Statistics

2.4.1 Overall Architecture/Workflow

The data analyzer consistently listens to the changes made to *vic_tweets*. By using the filter function defined in the design document of the database, it only focuses on the new tweet insertion change. Then, it will pull the new data from the database, do analysis on them and create corresponding analysis documents in the *analysis*. Meanwhile, the statistics program is an individual component, doing statistical calculations using MapReduce functions and updates the statistics documents periodically (default set to every 500 seconds).

In this way, the whole system is loosely coupled. The crawler, analyzer, and statistics update programs are completely isolated from each other, which makes the whole system more flexible, as well as increases the reliability of the system, as the failure of one component, does not affect the others.

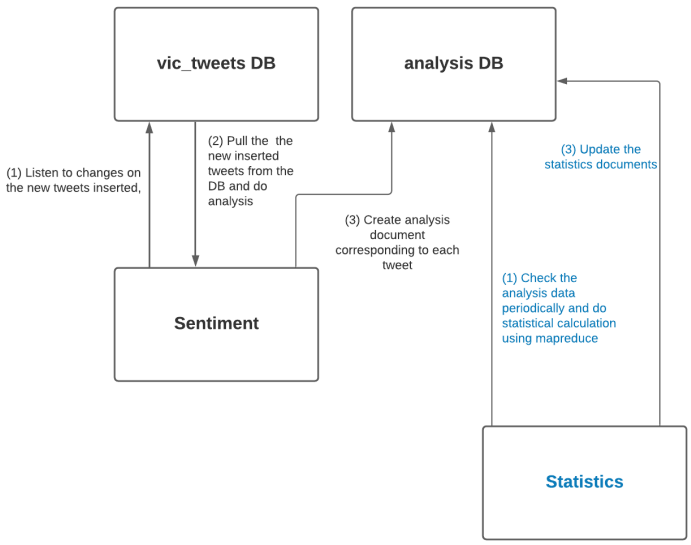


Figure 2. The workflow of the data analyzer(sentiment) and statistics

2.4.2 Statistics Documents

There are two statistics documents (the targets for the statistics.py to update), named **area_stats** and **hour_stats.**

The **area_stats** includes the positive and negative tweets ratio and the number of tweets corresponding to each emoji(detail explained in section 3.2.4) for each area, and the **hour_stats** includes the positive and negative tweets ratio and the number of tweets corresponding to each emoji for each time slot (0-23).

In this way, this backend server introduced in the next section simply needs to query to the database and to view these two stats files instead of spending time to do these stats every time the frontend sends requests to the server, which as a result, speeds up the server's speed for processing requests. One might argue that the stats data is not real-time. Nevertheless, this can be enhanced by increasing the update frequency of the statistics program. Furthermore, in the context of big data, a small amount of data updates does not affect the overall results. Hence, 500-seconds is an acceptable delay for the statistical results.

2.5 Analysis Data Server (Backend Application)

Considering that our backend server is not complex (as we shifted most of the statistical calculation logic to the statistics component introduced in section 2.4), Flask-RESTful is chosen to build the backend. It is a micro web framework for python that supports one to easily build RESTful APIs. Compared to other python frameworks such as Django, Flask is lightweight and requires minimal setup, and also easy to learn, which is perfect for this project.

The usage of the server is stated in section 4.7 of this document.

2.6 Visualization (Frontend Application)

Front end is implemented by applying Vue.js, mapbox-gl, and echarts to display the data analysis results.

2.6.1 Vue.js

The system uses Vue.js to implement the front-end website that shows the analysis results with diagrams. Vue is a progressive framework for building user interfaces. Unlike other monolithic frameworks, Vue is designed from the ground up to be incrementally adoptable. The core library is focused on the view layer only, and is easy to pick up and integrate with other libraries or existing projects. [Introduction — Vue.js, 2021)]

The Model-View-modelview (MVVM) is the one biggest feature of Vue, which separates the graphical user interface from the development of business logic or back-end logic. [Model–view–viewmodel - Wikipedia, 2021] Based on this feature, the ViewModel will monitor the change and notify the View to make the corresponding changes when the Model's data is modified. Otherwise, if the View is modified, the Model's data will be notified to change. This feature achieves the mutual decoupling of the View and the Model, so that it reduces the complexity of the business and the development time. The componentized development is another feature of Vue, which allows users to decompose the various modules in a single-page application into an independent and reusable small component, making it more flexible for subsequent updates or fixes. Also, Vue currently supports many free libraries. In this project, there are two libraries we have used: element-ui and axios. Element-ui is a set of component libraries implemented by using Vue 2.0 as the basic framework, which can help developers quickly build a website and reduce the development cost. Axios is a tool library for sending HTTP requests which is based on Promise. These are the main reasons why Vue.js was chosen.

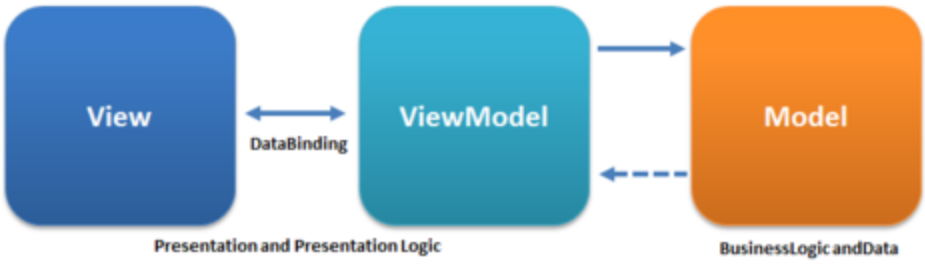


Figure 3. Model–view–viewmodel[Model–view–viewmodel - Wikipedia, 2021]

2.6.2 Mapbox-gl

Mapbox GL JS is a JavaScript library that uses WebGL to render interactive maps from vector tiles and Mapbox styles.[API Reference | Mapbox GL JS, 2021] Mapbox-gl can render a large number of map elements and has smooth interaction and animation effects. The first reason we choose Mapbox-gl to visualize data on maps is that we can use MapBox to create and customize personalized maps for free. The second reason is that users don't need to spend much time studying how to use it. The reason is that mapbox describes the entire map through a style document and the style document is a JSON object with specific root level and nested properties. It is such a powerful tool that allows us to visualize analysis results—Sentiment Percentage, Age level, and Top 5 Emoji—for each local government area of Victoria on the map by simply adding a popup.

2.6.3 ECharts

ECharts is an open-sourced JavaScript visualization tool and it depends on ZRender, a graphic rendering engine, to create intuitive, interactive, and highly-customizable charts.[Features - Apache ECharts, 2021] The reason that we choose it to visualize our analysis data since it can fully satisfy our needs. First, ECharts supports the basic chart types, including line chart, bar chart, and pie chart, and it's easy to create a combination of them with ECharts. In this project, we combine different charts for intuitive comparison and analysis data. For example, we try to analyze the relation between income and positive/negative sentiment percentage by putting the data of average income and the data of positive/negative sentiment percentage in one graph and confirming whether these two data have a proportional relationship. The second reason is that users don't need to spend much time studying how to use it. Like the style document of Mapbox-gl, ECharts uses 'option' to visualize data and the 'option' is a JSON object with specific root level and nested properties. Finally, ECharts provides many features that really help us to mine information from data. The feature of Interactive Data Exploration In-Depth is really helpful, as interaction is an important means of mining information from data. In this project, there are several scenarios that need to display above 80 local government area's data in one graph and this makes it difficult to see the detailed information. To fix this problem, we overview data first, then we use the 'dataZoom' component provided by ECharts to zoom a specific area to view details as needed.

2.7 Docker

Docker provides containerization for deploying the application in a uniform computing environment. All the environment settings, dependencies installation, required libraries can be bundled in one package. With containerization, all the differences in the computing environment can be solved. Docker containers allow scheduling multiple tasks which can greatly save time and workload.

Docker-compose enables quick start and stop applications with a simple ‘docker-compose up’ or ‘docker-compose down’ command. Without using docker compose, the container ID needs to be specified which is not convenient. Docker can quickly restart a service if the service hasn’t changed a lot, docker will use the caching ability and reuse the containers, which means the deployment will be fast.

The rapid deployment of Docker is one of the features that creates the container but the process, but not boot the Operating system. Each container has its own system and process, thus, they are isolated on docker which can decrease the system overhead.

2.8 UniMelb Research Cloud

The uniMelb Research Cloud is built on openstack and it has many associated services such as image service,network services,etc. It can offer great performance for the researchers to access efficient and scalable computing power and do the complex res

Advantages

1. There is no extra hardware cost for researchers to run their applications. Hardware may result in high cost due to the required great computing power. The researchers don’t need to pay any extra hardware infrastructure maintenance costs as well. Researchers can also access the instances at any time with great flexibility.
2. The high level security is achieved in uniMelb Research Cloud. Login to the instances requires the encrypted key pair in order to protect data and instances. The security rule can be customized by users and added into the instance's security group. These rules can control the incoming and outgoing requests.

3. uniMelb Research Cloud uses a private cloud model which is easier to control resource allocation and all the data. The administrator can easily allocate resources and control who can have access to the resources.
4. Researchers can manage the instance environment with high dynamics and scalability. The ansible scripts can be used for automated deployment of instances and setting up the environments.

Disadvantages

1. The resource is limited for each project, and the research needs to make a request for more resources.
2. Researchers can only have access to uniMelb Research Cloud by using uniMelb internet network or VPN, which is hard for some potential researchers.

3. System Capabilities

3.1 Motivation

The motivation for the system is to detect the emotion and discover correlations between emotion and people's age, as well as the area, population density, income level, and periods during a day. Since the sentiment analysis for positive and negative is quite limited, we also detect the emotion by deep learning approach, DeepMoji [Felbo et al., 2017].

3.2 Sentiment Analysis

NLP stands for Natural Language Processing, is a technique that is designed to analyse and represent naturally occurring text at one or more levels of linguistic analysis for achieving human-like language processing in a series of tasks or applications [Liddy, 2001]. Sentiment analysis is one of the NLP tasks that detect polarity and emotions. In the project, we use both rule-based and machine learning approaches to detect the sentiment.

3.2.1 Text Preprocessing

In the real world, tweets contain a lot of information and some of them might be redundant and not useful for our sentiment analysis models. Therefore, text preprocessing is quite

important for our Twitter data. In this project, we lowercase the world in each tweet and remove the username by regular expression. For example, the word after '@' symbol will be removed. We also remove the url by matching "http" prefix. Moreover, the emoji in the tweet will also be removed by using emoji[1] package in python.

3.2.2 Rule-based Method

Rule-based method is one of the base approaches to detect the polarity (Positive or Negative). In the project, we use the similar approach in assignment one. The polarity is detected by exact matching words or phrases in the pre-defined word dictionary. We compute the dictionary by "AFINN.txt" which has a set of words and phrases associated with positive or negative sentiment scores. For example, the word 'Wonderful' will have a positive score '4' and the word hate will have a negative score '-3'. If the word or phrase in the AFINN.txt has been exactly matched in the pre-processed text, the sentiment score will be calculated. As a result, each tweet will be classified as either positive or negative by the sentiment score if the score is not zero.

3.2.3 Machine Learning

Machine learning is important and has been found useful in Natural Language Processing. The input data can be trained in the machine learning model to predict the result. For example, given a set of positive and negative tweets as input, a binary classification model can be trained by a different machine learning method. The model will learn the pattern and representation of the positive and negative tweets and used to predict the unknown tweet.

As a subset of machine learning, deep learning is quite popular recently. Many deep learning methods have good performance in NLP tasks. Recurrent neural network(RNN) is the type of feedforward neural network that is extended to include feedback connections and it is quite useful for sequence modelling [Goodfellow et al., 2016]. For example, the word relation can be captured sequentially in the model. Long short-term memory recurrent neural networks(LSTM) follow the same basic structure as recurrent neural networks and solve vanishing gradient problems [Hochreiter and Schmidhuber, 1997]. It

[1] https://pypi.org/project/emoji

generally performs better than RNN. A Bidirectional LSTM(bi-LSTM) consists of two LSTM, it can train the text in both forward and backward direction [Huang et al., 2015]. Therefore, the word relation and semantic in both precious and back word can be captured.

3.2.4 DeepMoji

DeepMoji is a machine learning model that can be used to detect emotions. The DeepMoji uses two bidirectional LSTM with an attention layer and is trained on the 1.2 billion relevant tweets [Felbo et al., 2017]. It achieves state-of-the-art performance within sentiment, emotion and sarcasm detection. The model will detect the emotion of the tweets by the probability of the emojis. For example, the top five probability of the emoji for each tweet has shown in Figure 4.

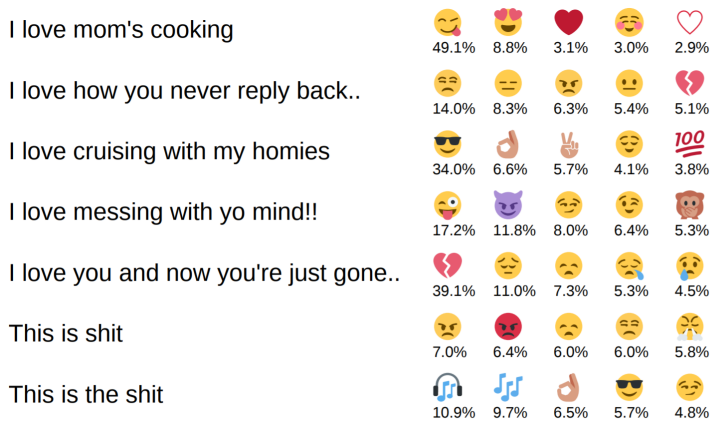


Figure 4. [Felbo et al., 2017].

We use their pre-trained model with 64 emojis from torchMoji[2]. The model will detect the emotion of the tweets by these 64 pre-defined emojis. For each tweet, only the highest probability of the emoji will be selected.

3.3 Supported Scenarios and Results

In this system, we are focused on Twitter sentiment analysis. The following are the different scenarios we have chosen:

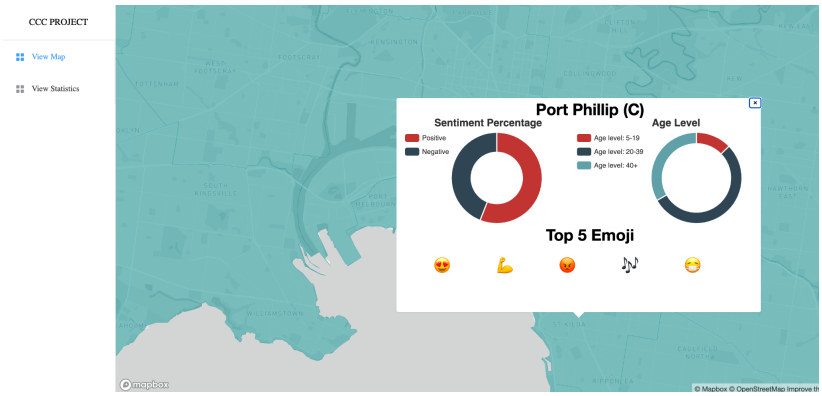
[2] https://github.com/huggingface/torchMoji

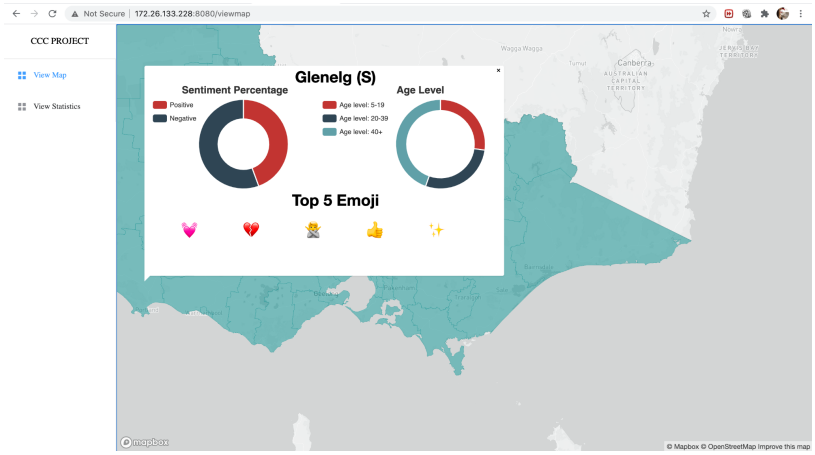
3.3.1 Twitter sentiment analysis for each area in Victoria

Description:

This scenario focuses on analyzing the sentiment percentage, the percentage of different age levels, and the top 5 Emoji representations for each area.

Result and analysis:





As shown in the figures above, this is the result of the area Glenelg and Port Phillip. In the sentiment percentage pie chart, positive percentage and negative percentage are shown. The other pie chart shows the age level percentage.

We can observes from the results that people from Glenelg are more emotional compared to people in Port Phillip, as the top1 emoji representation of the tweets is 💗 and the second one is 💔. In Port Phillip, the top emoji representation is 😍, which also reflects in the positive/negative tweets ratio pie chart, that in Port Phillip, the amount of positive tweets are greater than negative tweets. The results do make some sense in a way that we can connect it to reality. Port Phillip is a tourism place, where people tend to have a good mood when they tweet.

3.3.2 The relation between posted time and sentiment

Description

This scenario focuses on analyzing the relation between the Sentiment trend and the daily time period in Victoria.

Result and analysis

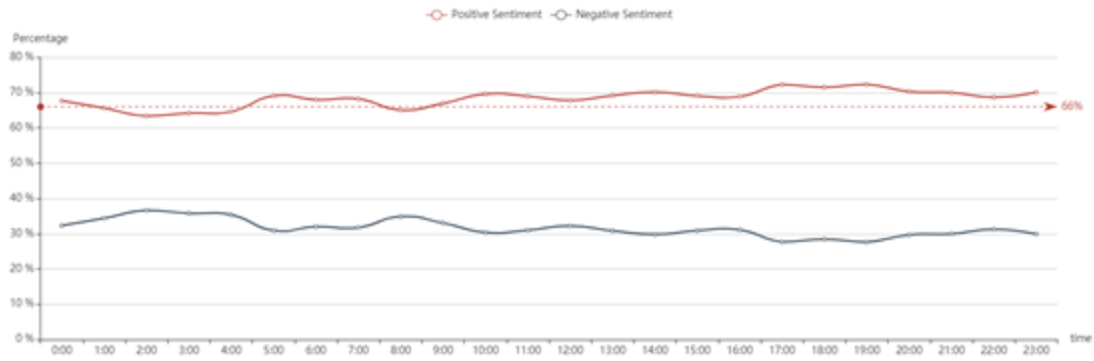


Figure 5. Positive Sentiment percentage and Negative Sentiment percentage in each time period

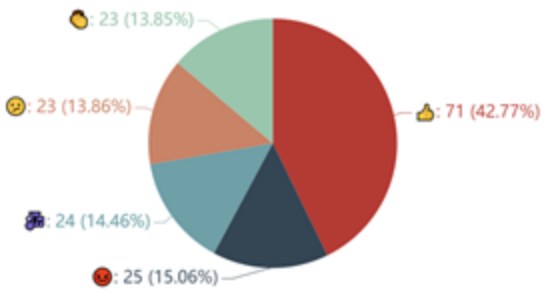


Figure 6. Top 5 emoji in 1:00 am

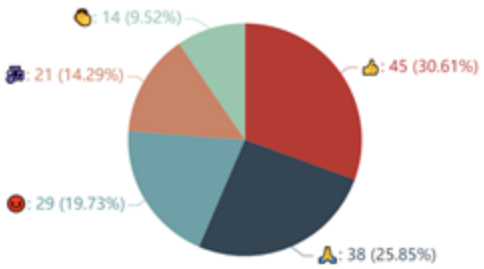


Figure 7. Top 5 emoji in 2:00 am

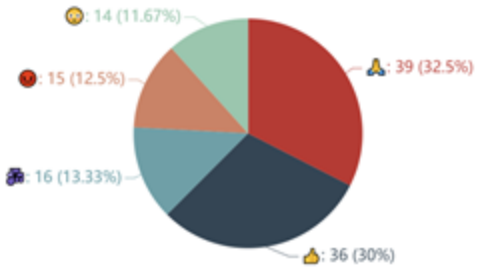


Figure 8. Top 5 emoji in 3:00 am

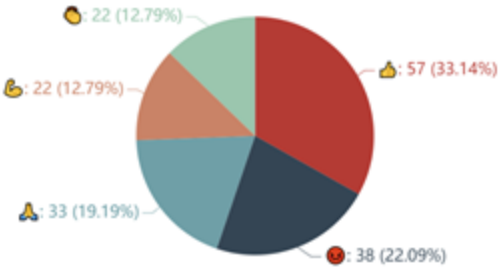


Figure 9. Top 5 emoji in 4:00 am

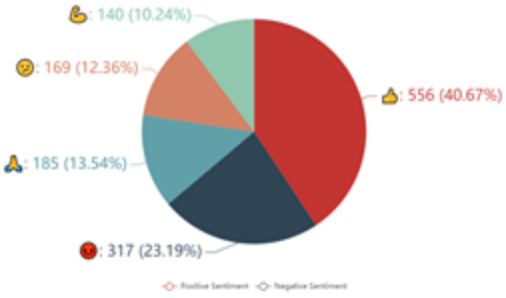


Figure 10. Top 5 emoji in 8:00 am

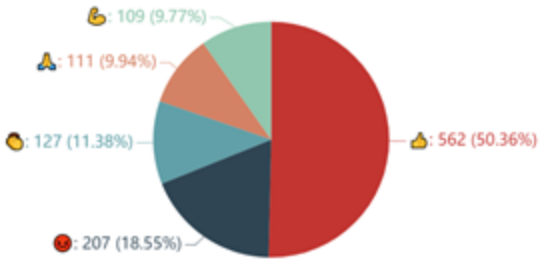
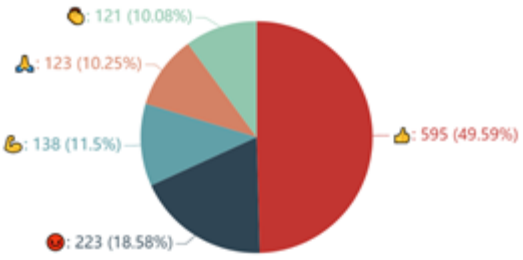


Figure 11. Top 5 emoji in 17:00 am



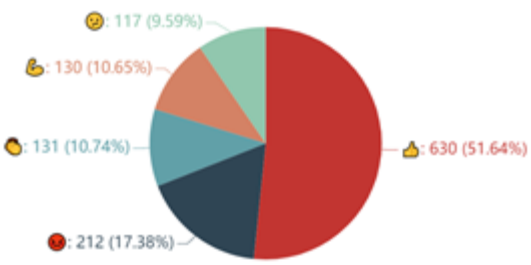


Figure 12. Top 5 emoji in 18:00 am Figure 13. Top 5 emoji in 19:00 am

The figure 5 has shown the trend of positive/negative sentiment percentage in each time period. By comparing the sentiment percentage in each time period of Victoria, we can identify the time period with high and low sentiment. From 1:00 am to 4:00 am and 8:00 am, the positive sentiment percentage of these time periods are the lowest four positive sentiment percentages (66%), which are low sentiment. From figures 6 to 9, we can find that Twitter usage is gradually decreasing, as most people start to fall asleep before 1:00 am. Also, we find that the emoji of pray is increasing and the emoji of good is decreasing from 1:00 am to 3:00 am. Generally, people don't sleep because they have some personal reasons, such as insomnia, depression, learning and work. These personal reasons all can reduce a person's sentiment and it's consistent with the result that 1:00 am to 4:00 are low sentiment periods. After 8:am, many people wake up and the twitter usage is dramatically increasing. However, the positive sentiment percentage is decreasing and the proportion of Angry emoji is increasing. The main reason might be related to traffic, as we find that people will easily get angry in traffic.[Deffenbacher, Lynch, Oetting and Swaim, 2002] From 17:00 am to 19:00 am, the positive sentiment percentage of these time periods are the highest three positive sentiment percentages, which are high sentiment. The positive emoji is increasing, the twitter usage is dramatically increasing and the proportion of good is around 50%. The main reason is that people can get rest after work hours.

Conclusion

Overall, we can determine that the time period of 1:00 am, 2:00 am, 3:00 am, 4:00 am and 8:00 am are low sentiments and the time period of 17:00 am, 18:00 am, and 19:00 am are high sentiments.

3.3.3 The relation between age and sentiment

Description

This scenario focuses on the relationship between age and the numbers of sentiment.

Result and analysis:

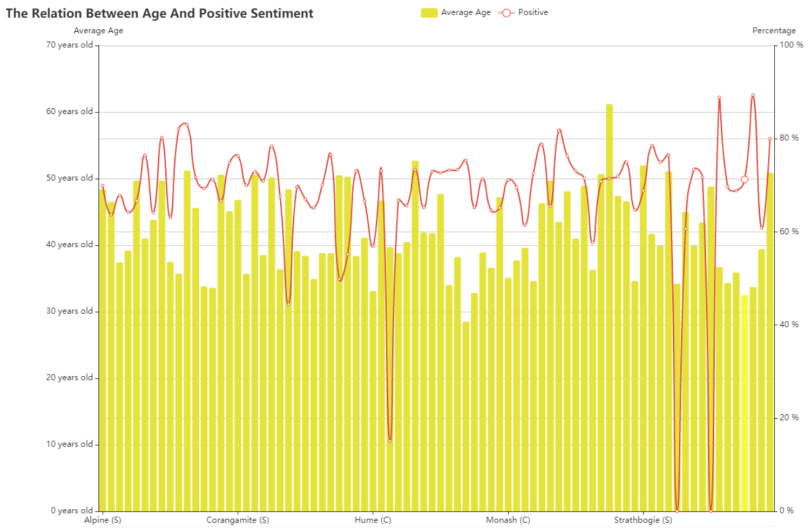


Figure 14

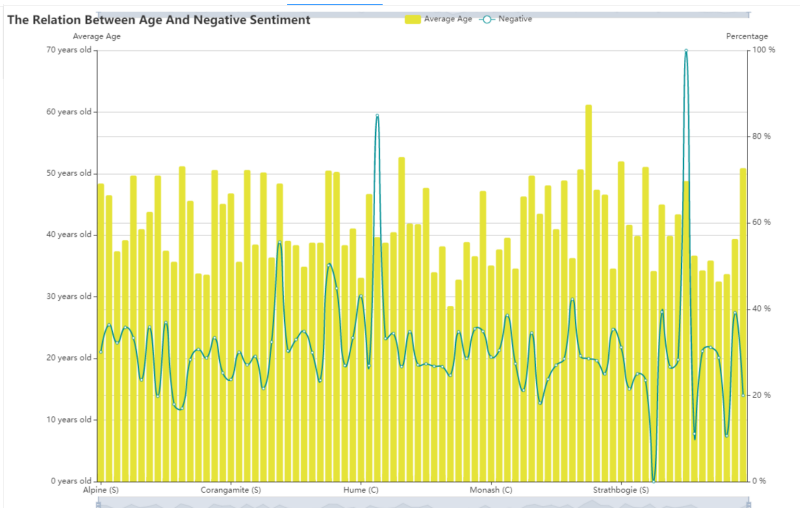


Figure 15

conclusion

From figure 14, we can see that the Yara has the most positive sentiment score where the average age is quite low. From the figure, We can inform that most of the low age demographic have a high positive sentiment score.

From figure 15, we can see that Kingston(C) has a quite high negative score. From the figure, It is more likely that the order people tweet less negative tweets.

3.3.4 The relation between average income and sentiment

Description

This scenario focuses on the relationship between income and sentiment.

Result and analysis:

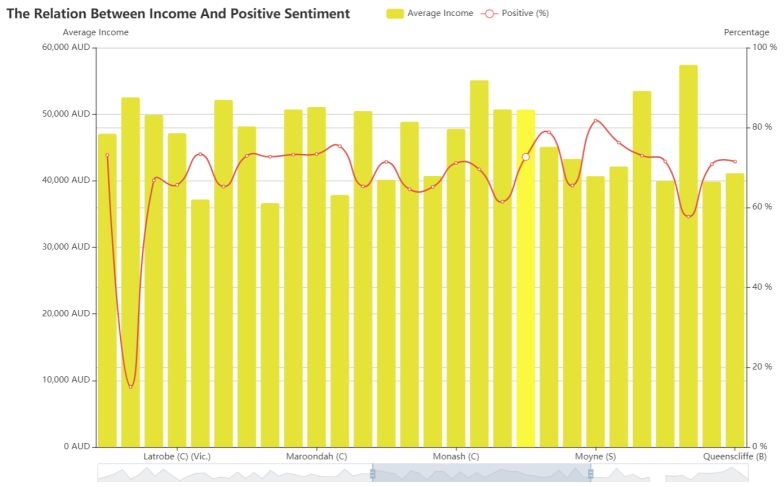


Figure 16

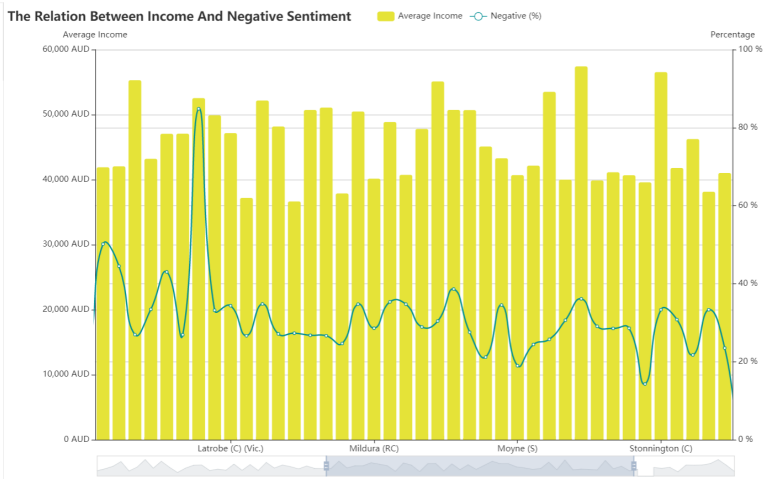


Figure 17

Conclusion

From figure 16, We can inform that most of the areas with a high income will have a more positive sentiment.

From figure 17, we can see that It is more likely that the poorer people tweet more negative tweets.

3.3.5 The relation between density and sentiment

Description

This scenario focuses on the relationship between population density and sentiment.

Result and analysis:

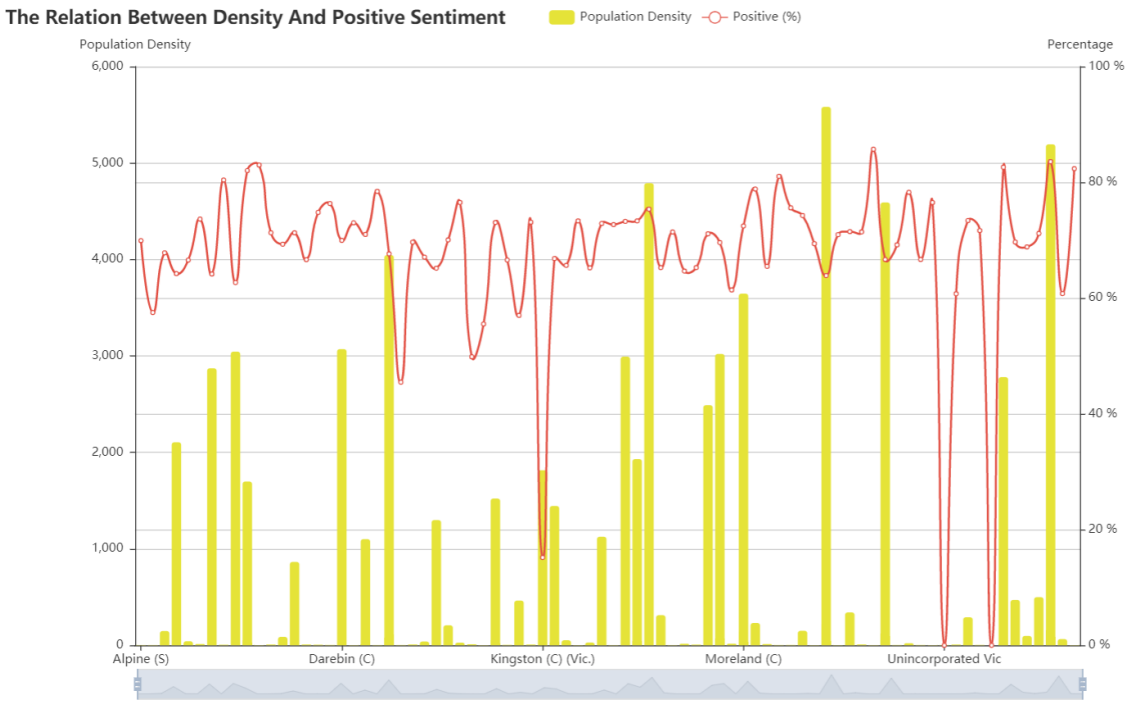


Figure 18

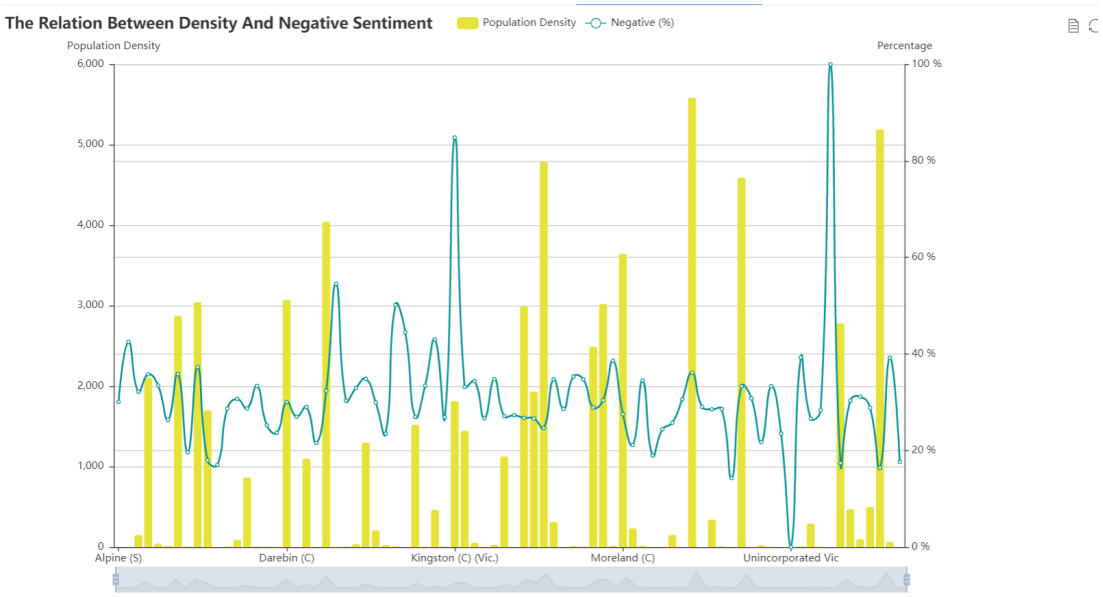


Figure 19

Conclusion

From figure 18 and figure 19, we can conclude that there is no strong correlation between population density and people's sentiment when they tweet.

3.4 Limitation

3.4.1 Data Collection

For the search API and the Twint approach to collect data, the central point of each area is collected by using Google Map API. The limitation is that we cannot guarantee that the central point is actually the best point that estimates the center of the area. The geolocation has been collected by searching the area name from the Google Map, which may lack some accuracy. Moreover, the default radius has been set to 15km for every searched area. Therefore, for quite big areas, some of the tweets may not be collected in that area.

Also, only tweets with English content will be collected and analyzed in this system. As a result, the analysis results only apply to people living in Victoria who use English to tweet.

3.4.2 Sentiment Analysis

Sentiment analysis is still quite challenging in the real world. For our rule-based method, the method might not be the best approach due to non-consideration of emoticons, modifiers etc. Some of the positive or negative tweets might not be classified due to the limitation of the sentiment word in the "AFINN.txt". In the future, we might consider using transfer learning approaches (e.g. using pre-trained BERT model) to classify the polarity.

There are also some limitations for DeepMoji. Although it has state-of-the-art results for many NLP tasks, we still cannot guarantee that every tweet has been accurately classified. For example, some tweets might have a low probability of the top two emojis and the probability are quite close. However, only the first emoji has been selected in our analysis.

4. User Guide

Link to the source code:

All the following commands are required to run inside /ansible directory. OpenStack password from Melbourne Research Cloud is required for authorization.

4.1 Instance Deployment

This section includes the automated deployment of instances (on Melbourne Research Cloud by using ansible playbook tasks. The tasks are listed below:

Tasks	
common	Common configuration for instances such as 'install pip, openstacksdk', etc
show_images	Shows all the available images
create_volume	Created for later storage and backup for each instance.

create_security_groups	Adding security groups and the security rules for managing network access control. The security rule such as the port 22 is opened for using ssh.
create_instances	Create instances on MRC A list of hosts ip addresses will be produced after this task inside /inventory/hosts.ini, this file will be useful for hosts allocation.

The file ' /host_vars/mrc.yaml' defines the variables such as chosen image, flavor, availability zone, etc. Image Ubuntu 18.04 LTS (Bionic) amd64 (with Docker) is used for deployment because Docker is already installed. The chosen flavor uom.mse.2c9g is allocated for three instances and uom.mse.1c4g is allocated for one instance due to the compute resource for this project.

To deploy the instances, run the command **./run-nectar.sh**

4.2 Instance Configuration

This section includes instance configuration, how to set up the environment and install all the required dependencies. The ansible playbook tasks are listed below step by step.

Tasks	
add_key	Add the key to the localhosts,so that the user can login to the instances via ssh.
add_proxy	Add the unimelb proxy to the /etc/env, to set up the environment
install_dependencies	sudo apt-get update; sudo apt-get install (git, python3-setuptools,etc)
configure_docker	Configure proxy directory for docker, add proxy to environment, and restart docker
clone_git	Clone the git repository to each instance under the path /home/ubuntu/COMP90024
mount_volumes	Mount the volume for each instance by using the mount point in the '/host_vars/mrc.yaml'

To configure instances, run the command **./configure.sh**

4.3 Hosts Allocation

A file called create_hosts.py in '/inventory ' can allocate jobs to different hosts inside the 'hosts.ini' file. We use this file to generate 'host_job.ini' . Host allocation includes allocate hosts for couchdb deployment, couchdb master node and workers for clustering, crawler, tweet analysis and statistics, backend and frontend deployment.

To allocate hosts, run **python inventory/create_hosts.py**

4.4 Database Deployment

This section includes how to deploy couchdb and couchdb clustering. The following tasks run in the ansible playbook. [database],[masternode] refers to 'inventory/host_job.ini'. All the configuration of couchdb such as couchdb user name and password can be found inside 'host_vars/configure.yaml'

Instances	**Tasks**	
All the instances in [database]	**create_couchdb**	Create the docker container with a couchdb for all the instances in [database]
	prepare_couchdb	Prepare the view document for couchdb
[masternode]	**create_couchdb_cluster**	Set up couchdb cluster on masternode, and the workers for masternode

To deploy couchdb, run the command **./deploy_couchdb.sh**

4.5 Crawler

4.5.1 Deployment

The crawler will be deployed on two instances in order to balance the server load. Therefore, the task is separate into streaming and searching.

Ansible playbook tasks ‘deploy_crawler_search’ and ‘deploy_crawler_stream’ using the same dockerfile entrypoint to build the container on two instances, but run with different arguments. Before running the docker, the task will stop running the same image. The Aurin data will be analyzed and stored in the couchdb as well.

To deploy the crawler, run the command **./deploy_crawler.sh**

4.5.2 Usage

The crawler can either be set to search mode or stream mode by setting different arguments to the scripts. The arguments are listed below:

-h, --help	show this help message and exit
--stream	streaming twitter
--location x1 y1 x2 y2	streaming twitter --> specify (x1, y1, x2, y2), default set to Victoria
--search	search twitter
--storeAurinData	store data from Aurin
--area AREA	search twitter --> specify search area or default set to Victoria

The search method can only define a searching area represented by a central point with radius, and the stream method can only define a streaming area represented by a bounding box. The default searching area is **"-37.03521,145.23218,400km"** and the default streaming area is **[141.03,-38.41,148.7,-33.98]** which both roughly cover the whole of Victoria.

4.6 Sentiment analysis and statistics Deployment

Both analysis and statistics programs are deployed on the same instance, and use the same docker image. Specifying ‘sentiment.py’ or ‘statistics.py’ is required.

To deploy sentiment analysis and statistics, run the command, **./deploy_analysis.sh**

4.7 Backend

4.7.1 Deployment

To deploy backend service, one ansible task is used to firstly stop the previous docker server, and then redeploy the latest backend code, and start the server by specifying port number.

To deploy backend service, run the command **./deploy_backend.sh**

4.7.2 Usage

Five routes are provided when the server activates :

1. **http://172.26.130.110:5000/totalcollected**

This route returns the total collected tweets and total analysed tweets so far, noted that this is real-time data.


```
{
   "ok": true,
   "data": {
       "total_collected": 588828,
       "total_analysis": 512580
   }
}
```


2. **http://172.26.130.110:5000/getTopFiveEmoji**

This route returns the top five emoji counts across Victoria.


```
{
   "ok": true,
   "data": [
       {
           "emojicode": "\ud83d\udc4d",
           "number": 53428
       } ... ]
}
```


3. **http://172.26.130.110:5000/incomesentiment**

This route returns the positive/negative tweets ratio along with the average income data for each area.


```
{
   "ok": true,
   "data": [
       {
           "area": "Alpine (S)",
           "positive": 69.93006993006993,
           "negative": 30.069930069930066,
           "income_average": 38000
       }... ]
}
```

4. http://172.26.130.110:5000/time

This route returns the positive/negative tweets ratio along with the top five emoji counts for each time slot (0-23).

```
{
   "ok": true,
   "data": [
       {
           "time": 0,
           "positive": 67.54839946329308,
           "negative": 32.45160053670692,
           "top_five_emojis": [
               {
                   "emojicode": "\ud83d\udc4d",
                   "number": 872
               }...]

       }...]
}
```

5. http://172.26.130.110:5000/allstatistics

This route returns the positive/negative tweets ratio along with all the data (age, population density, income average) collected from Aurin for each area.

```
  { "ok": true,
   "data": [
       {
           "area": "Alpine (S)",
           "positive": 69.93006993006993,
           "negative": 30.069930069930066,
           "age_distribution": {
               "median_age": 48.4,
               "teenagers_and_young_adults_ratio": 17.2,
               "adults_ratio": 17.799999999999997,
               "middle_age_and_above_ratio": 28.1
           },
           "population_density": 2.6761999130249,
           "top_five_emojis": [
               {
                   "emojicode": "\ud83d\udc4d",
                   "number": 1825
               }...]
       }...]}
```

Error Handling

If the server does not handle the request successfully, it will return :

```
{
               "ok": False,
               "msg" : "Something went wrong: "}
```

4.8 Frontend

The URL to access the front-end application: http://172.26.133.228:8080/

4.8.1 Deployment

To deploy the frontend application, one ansible task is needed to run and it is going to build the docker compose file by specifying the image and port number.

To run the application, run the command **./deploy_frontend.sh**

4.8.2 Usage

When entering the system, the first page list the information about our group members, it looks like this:

Figure 20 .cover page

After clicking the “start” button, you will be navigated to the “View Map” page, in this page, the areas we used to study are layered with color. The map allows you to zoom in and out. When clicking one of the layered areas, the area’s information like the area name, sentiment situation, age level distribution, and the top five emoji will be displayed in a pop-up window.

Figure 21.view map page, before clicking an area

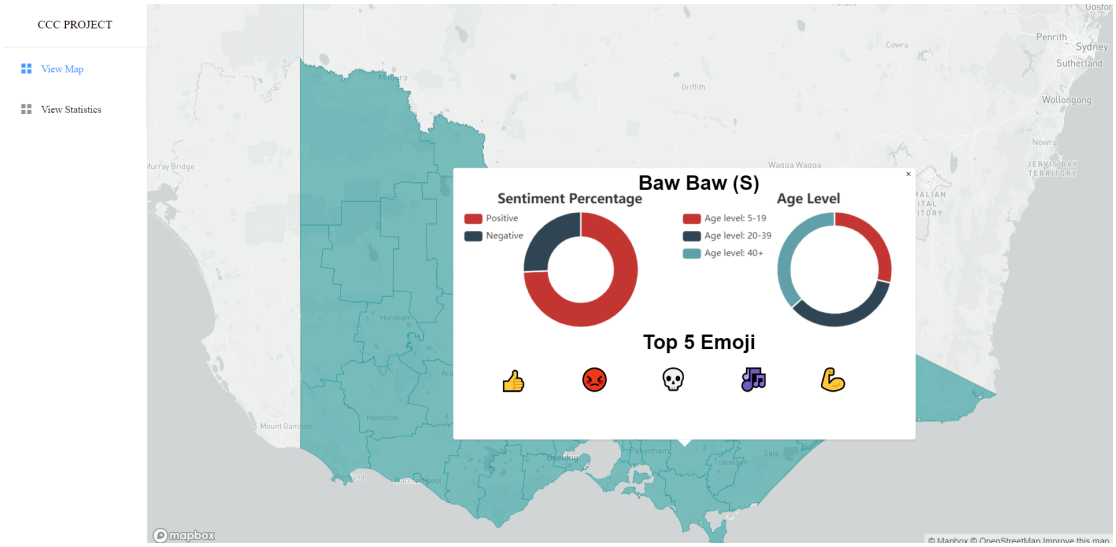


Figure 22.view map page, after clicking an area

You can access the scenarios we used by clicking the “View Statistics” button in the left-hand side menu. The first page will be the number of tweets we collected and the number of tweets we used to analyze. All the scenarios we used can be found in the top menu.

Figure 23. pages in “View Statistics”

5.0 Youtube Video Links

5.1 Deployment

https://youtu.be/InAKOWJ84f8

5.2 Demo

https://youtu.be/3H9HBDOzhN4

6.0 Github Repo

https://github.com/Catherine959/comp90024.git

Reference:

Deffenbacher, J., Lynch, R., Oetting, E. and Swaim, R., 2002. The Driving Anger Expression Inventory: a measure of how people express their anger on the road. Behaviour Research and Therapy, 40(6), pp.717-737.

Echarts.apache.org. 2021. Features - Apache ECharts. [online] Available at: <https://echarts.apache.org/en/feature.html> [Accessed 26 May 2021].

En.wikipedia.org. 2021. Model–view–viewmodel - Wikipedia. [online] Available at: <https://en.wikipedia.org/wiki/Model%E2%80%93view%E2%80%93viewmodel> [Accessed 26 May 2021].

Felbo, B., Mislove, A., Søgaard, A., Rahwan, I., & Lehmann, S. (2017). Using millions of emoji occurrences to learn any-domain representations for detecting sentiment, emotion and sarcasm. arXiv preprint arXiv:1708.00524.

Goodfellow, I.; Bengio, Y.; and Courville, A., 2016. Deep Learning. MIT Press. http://www.deeplearningbook.org.

Hochreiter, S. and Schmidhuber, J., 1997. Long short-term memory. Neural computation, 9, 8 (1997), 1735–1780.

Huang, Z., Xu, W. and Yu, K., 2015. Bidirectional LSTM-CRF models for sequence tagging. arXiv preprint arXiv:1508.01991.

Liddy, F. D., 2001. Natural Language Processing. NY. Marcel Decker, Inc., 2;2nd; edn.

Mapbox. 2021. API Reference | Mapbox GL JS. [online] Available at: <https://docs.mapbox.com/mapbox-gl-js/api/> [Accessed 26 May 2021].

Vuejs.org. 2021. Introduction — Vue.js. [online] Available at: <https://vuejs.org/v2/guide/index.html> [Accessed 26 May 2021].