THE UNIVERSITY OF MELBOURNE

COMP90024

Cloud-based Geospatial Analysis of Twitter in Victoria

May 26, 2020

Team 1

Che-Hao Chang (1040445)
Yu Ting Lin (762826)
Abdul Rehman Mohammad (569618)
Isaac Pedroza Aguirre (1004268)
Maxim Priymak (216213)

Abstract

We design and implement a cloud-based solution to collect, process, analyze, and visualize integrated multi-source data. The solution architecture successfully implements several leading heterogeneous technologies and is purposefully designed to be automated and scalable. The resultant system is leveraged to explore several scenarios regarding Australian society and lifestyle through the integration and geospatial analytics of publicly-available social-media data and official statistics. Sentiment analysis is conducted on harvested tweets originating in Victorian Local Government Areas (LGA) to identify whether there exists a meaningful correlation between: 1) socio-economic well-being of an area and the number of favourable tweets regarding welfare, 2) relative affluence of a region and user preference regarding mobile device brands, and 3) overall negativity of tweets and the mental health of a region.

Contents

1 Introduction

With the continuous development of the internet, social media begins to play a prominent role in our everyday lives. Twitter is one of the leading micro-blogging sites that boasts of over 340 million users internationally [9], and allows people to post short text messages expressing their thoughts and emotions. Twitter has about 5.3 million unique monthly users in Australia [11], who constantly share documents and opinions. Twitter readily provides a public API (with some limitations for the free-of-charge version) that allows developers to incorporate its vast network of content into 3rd party applications. This has allowed Twitter to become the focus of sentiment analysis by academic researchers and businesses. Analysis of Twitter data holds great potential benefit – it can inform researchers and policy makers about social and economic issues prevalent within Australian society.

In this work, we consider how sentiment analysis of Twitter texts could be utilized to construct models that reflect and predict various aspects of Australian life. In particular, we examine whether correlations exist between:

1. the welfare-related Twitter sentiment and the socio-economic well-being of a region,
2. preference for Android and the wealth of a region,
3. negativity of tweets and the prevalence of regional mental-health-related complications.

To this end, we design and build a cloud-based data-analytics application that runs on virtual machines across the University of Melbourne Research Cloud [5]. The application harvests and categorizes Twitter data based on its geometric location and sentiment, and correlates this with relevant information extracted from Australian Urban Research Infrastructure Network (AURIN) [1]. The tweets are harvested via both Stream and Search API interfaces, and subsequently processed and stored into a distributed CouchDB database. In this work, due to the limitations of our sentiment analyzer, we only consider English language tweets. Basic sentiment analysis is performed on the collected tweet texts, by assigning each tweet a sentiment polarity from $-\infty$ to $+\infty$, expressing an abstracted aggregation of positive and negative emotions contained within the text. For each analytical scenario, we map the AURIN data onto a choropleth map for easy visualization and construct linear regression models to examine whether correlations between social-media data and AURIN statistics exist.

Module	Description	Contributing Member
Cluster Configuration	Ansible automation	Abdul Rehman Mohammad Isaac Pedroza Aguirre
Tweet harvesting	Search/stream/filter Twitter input	Che-Hao Chang
Scenarios	AURIN data extraction	Yu Ting Lin Maxim Priymak
Data analysis	Data processing pipeline	Maxim Priymak Che-Hao Chang
CouchDB cluster	Configuration of database	Abdul Rehman Mohammad
Docker swarm	Configuration of Docker swarm	Isaac Pedroza Aguirre
Web-server	Configuration of Web server	Che-Hao Chang Yu Ting Lin
Documentation	Collation of the report	Maxim Priymak

Table 1: Outline of the contribution of each team member to this project.

The project GitHub repository is located at: `https://github.com/isaacpedroza/city-analytics`

1.1 Contributions

All members of the group have worked tirelessly to implement the system discussed in this report. Every group member was tasked with several non-overlapping responsibilities in order to avoid redundancy. Ultimately, however, due to the large scope of the project, every member of the group cooperated with and aided each and every other member of the group, and therefore contributed to each other's work. Table 1 roughly outlines the primary tasks that each group member was occupied with during the implementation of the system.

2 Analytical Scenarios

In this project, we explore how Twitter data can be augmented by data available within the AURIN platform to improve our knowledge of life in Victoria, Australia. Throughout this work, we assume positive/negative sentiment tweets to have polarity > 0 and < 0 respectively. Also, to avoid division by 0, all ratios are computed as $f(x_1, x_2) = x_1/(x_2 + 1)$. A variety of social-media data analytics scenarios are examined:

1. We attempt to correlate the data from AURIN pertaining to the social well-being of a region with the sentiment polarity of Twit-

ter posts regarding 'Centrelink', 'JobSeeker', 'Welfare', 'JobKeeper', 'AUSTUDY', and other keywords related to welfare. In particular, for each geographical location, we extract the ratio of positive (sentiment > 0) to negative (sentiment < 0) tweets that contain any Centrelink-related keywords and plot this against the IRSD metric of a Victorian Local Government Area.

The Index of Relative Socio-economic Disadvantage (IRSD) is a general metric that summarises a range of information about the economic and social conditions of people and households within an area [2]. This index includes measures of relative disadvantage. A high/low score indicates lower/greater disadvantage respectively, and a value of 1000 represents the national average. For example, an area could have a low score if there are:

- many households with low income,
- many people with no qualifications,
- many residents for whom English is a second language,
- many people in low skill occupations.

This composite index is recommended in situations where the user:

- wants to look at relative disadvantage of geographical regions,
- requires a broad measure of disadvantage, rather than a specific measure (such as low income).

We posit that there should be a significant inverse correlation between sentiment ratio and the IRSD metric. We hypothesize that this is due to the fact that socially-disadvantaged individuals are more likely to be supportive of welfare payments due to their difficult financial and personal circumstances, while the wealthy individuals are less likely to be supportive of welfare. We anticipate a lower support for welfare among the socially-advantaged regions due to the extant system of progressive taxation in Australia – it is plausible that many wealthy individuals would prefer lower taxation to higher welfare expenditure. For example, 34% of low income respondents stated that current level of welfare is 'too little' whereas only 21% of high income respondents agree. Conversely, 42% of high income earners responded that there is 'too much welfare', whereas only 28% of low income earners agreed [16]. Additionally, 68% of welfare recipients have a favourable view

regarding welfare programs, compared to only 51% of non-recipients [13].

2. For each geographical region, we examine the correlation between the wealth of a region and the user preference regarding iPhone vs. Android usage. User predilection towards specific mobile handheld platforms is expressed as the ratio of tweets that originated from an Android versus an iPhone platforms, where the specifics of user platform is extracted from tweet metadata. The wealth metric of a region is defined as $(U - L)(U + L)$, where U and L represent the proportion of regional households within the upper/lower income brackets respectively.

   According to OECD specifications, the poverty line is located at half of the total weekly median household income [17]. Australian Bureau of Statistics specifies the Australian gross median weekly household income to be 1700 AUD [3]. We therefore set the poverty line threshold at 800 AUD weekly gross household income for each LGA. All households that fall below this bracket are deemed to be 'low-income'. Likewise, the 'wealthy' households are deemed to have household incomes above 1.5 Australian medians (i.e. above 2500 AUD). Therefore, L and U are calculated as $N_{\text{poor}}/N_{\text{total}}$ and $N_{\text{wealthy}}/N_{\text{total}}$ respectively, where N_{total} represents the total number of households in an LGA. Gross household income was chosen for our custom metric due to the well-defined poverty threshold, and its close correlation to disposable household income (disposable income is often spent on consumer goods/electronics). The wealth metric is negative/positive for wealthy/poor regions, and approaches 0 for regions with a more sizeable middle class.

   Our hypothesis is that there will be significant inverse correlation between the wealth of a region and the Android-to-iPhone ratio (i.e. platform ratio) due to the fact that for many individuals, the possession of the latest iPhone model represents a wealth-related status symbol, whereas many prefer Android based on the higher cost-efficiency that it offers.

3. We examine the potential for correlation between the ratio of the number of positive-to-negative sentiment tweets from a region with a mental-health metric of said region. The positive- and negative-sentiment tweets are defined as those tweets with +ve and -ve polarity scores that lie outside of one standard deviation within the sentiment

distribution. The filtering of tweets by polarity aims to consider only the messages that display strong emotion and satisfaction/dissatisfaction. We expect that the propensity for displays of extreme negative emotions in tweets is more reflective of the potential for underlying mental health pathology.

The mental-health metric for a region is inferred via the aggregation of statistics pertaining to the number of hospital admissions. In particular, we sum the number of yearly admissions (per 100k residents) to all hospitals due to complications related to mental-health problems or mood-affective disorders. Since the national averages regarding these statistics are difficult to extract from existing publications [6], we instead standardize the total number of admissions across all of the 79 Local Government Areas in Victoria.

We postulate that the mental health metric may be correlated with the ratio of positive to negative tweets due to the fact that mental health complications commonly reveal themselves through informal social interactions between individuals. Some of the most common mood-affective psychological disorders are major depression, bipolar disorder, and post-traumatic stress disorder. The World Health Organization has projected that by 2020, depression will become the second leading cause of disability worldwide. On average, the estimated lifetime prevalence of depression is as high as 15% in high-income countries [15]. This is consistent with the statistics regarding incidence of depression in Australia [7]. Additionally, depression is one of the major causes of mental-health-related hospital admissions in Australia [6].

We therefore anticipate that regions with a frequent occurrence of hospital admissions due to mood-affective or psychological disorders will generate a higher proportion of tweets with a strong negative sentiment. Hence, we expect the mental health metric to be inversely correlated with the sentiment ratio.

3 System Requirements

For this project, we require a system that is able to collect twitter data, process said data, store it and analyze it together with AURIN data, and display the data to help visualize it:

3.1 Functional requirements

1. Harvester script with sentiment analysis capabilities to collect relevant Twitter data.
2. Database to store the collected Twitter data.
3. Application to extract data from the database, process it together with the locally-stored AURIN data, and send the result to the front-end.
4. Web server to receive the data from application, and serve users by returning requested HTML pages.

3.2 Non-Functional requirements

1. Scalability:
   (a) Database, application, and web server setup must be automated for expedient deployment.
   (b) Must use a distributed database architecture such that more database nodes may be added at will in order to expand the storage capacity, parallel accessibility, and failure tolerance of the database cluster.
   (c) All the components of the system must be able to scale in order to handle the 4 V's of big data (Velocity, Volume, Variety, Veracity) in response to changing requirements.
   (d) The number of nodes can vary without affecting the stability of the system or the performance dropping below pre-defined thresholds.
   (e) Nodes can be added and removed without excessive human intervention.
2. Fault tolerance:
   (a) Database data must be replicated (i.e. sharding of database and certain level of redundancy) so that a single node failure does not result in loss of data or failure of the entire system.
   (b) When a single service fails, the system operation should not be affected besides a temporary loss of functionality.
   (c) Any failed service must be automatically resumed as soon as possible.

3. Openness and extensibility:

    (a) Open standards should be utilized to construct the system. Usage of simple and well-documented interfaces can make the systems adaptable and extensible.

4. Portability:

    (a) Our software must be self-contained and easily ported to most cloud services that offer Infrastructure as a Service (IaaS).

5. Concurrency:

    (a) The system must allow multiple users to concurrently request resources from the application without deterioration of Quality of Service (QoS).

6. Heterogeneity:

    (a) System should be able to run on a wide range of hardware and software platforms. This is accomplished by utilizing Virtual Machines (VM) and Docker containers.

7. Transparency:

    (a) Location transparency - the user is unaware where in the distributed system the resources are located. This transparency is achieved by the abstraction layer that the Web Server introduces.

    (b) Replication transparency - the exact location of replicated data does not affect the operation parameters of the system – no explicit user intervention is required to ensure that the data is replicated.

    (c) Concurrency transparency - multiple processes have to be able to use the same resources without affecting the operation parameters of the system. That is, the system should operate exactly the same for multiple users as it does for a single user.

    (d) Scalability transparency - the number of nodes should be able to change without affecting system operations.

    (e) Failure transparency - some components of the distributed system should be able to fail without affecting the distributed systems operation.

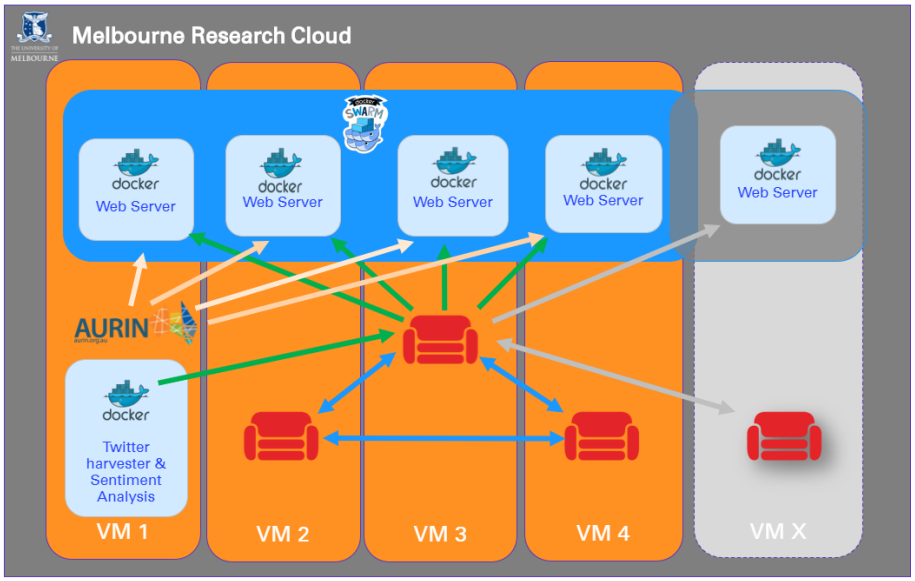


Figure 1: Schematic of system architecture.

Usage of reliable and flexible 3rd party software frameworks with well-documented interfaces, as well as reliance on RESTful architectural style [14] allows us to ensure that the final system is designed in accordance with the above-mentioned design principles.

4 System Design and Architecture

4.1 Basic System Overview

There are 3 main components in our system. First is the twitter harvester which extracts the data from Twitter using Twitter Search and Stream API and conducts basic preprocessing of Twitter text and Sentiment Analysis. The harvester continuously writes to the CouchDB database, which contains the predefined views and MapReduce functionality. Finally, the web server extracts the data from the CouchDB endpoint, and displays the data once it receives a HTTP request. A high-level schematic of our system is shown in Fig. 1

4.2 System Architecture

System architecture defines the structure of software components, the relationship between components, and is a coherent design package with certain properties that allows reuse [12]. Two major types of distributed system architectures are: Service-Oriented Architecture (SOA), and Resource-Oriented Architecture (ROA). A SOA service is a discrete unit of functionality that is independent of vendors, products and technologies that can be accessed remotely and acted upon and updated independently. SOA integrates distributed, separately maintained and deployed software components, and dictates technologies and standards that facilitate components' communication and cooperation over a network infrastructure. On the other hand, ROA is a specific set of guidelines of an implementation of the RESTful architecture supporting the internetworking of resources, where a resource in this context is any entity that can be identified and assigned a Uniform Resource Identifier (URI). Within the ROA concept, resources include not only IT infrastructure elements such as servers, computers and other devices, but also Web pages, scripts, etc. Four essential concepts comprise ROA [14]:

- resources (i.e. articles on Wikipedia),
- their names (URIs – where the URI is the name and address of a resource),
- their representations (a resource is a source of representations),
- the links between them (normally, a hypermedia representation of a resource contains links to other resources)

and four principles

- addressability (addressable applications expose a URI for every piece of information they might conceivably serve),
- statelessness (every HTTP requires happens in complete isolation – the server never relies on information from previous requests),
- connectedness (a web service is connected to the extent that you can put the service in different states just by following links),
- uniform interface (in ROA, HTTP is the uniform interface, where GET methods retrieves a representation of a resource, PUT/POST method

deposits a resource to an existing/new URI, and DELETE method to remove an existing resource).

In our system, the main relationship b/w software components is the retrieval of information. Thus, ROA is the most suitable solution. Representational State Transfer (ReST) defines a uniform connecter interface that identifies and navigates resources [14]. ROA architecture allows our application to turn into a ReSTful web service.

4.3 Detailed System Architecture

We are building our system on top of the Melbourne Research Cloud (MRC) that utilizes cloud orchestration technology OpenStack, and provides Infrastructure As A Service (IaaS). Openstack manages the creation of a virtual machine and decides on which hypervisor (i.e. physical host) to start it, enables VM migration features between hosts, allocates storage volumes and attaches them to VMs, meters usage information for billing and more. Openstack software platform consists of interrelated component that control diverse, multi-vendor hardware pools of processing, storage, and networking resources throughout a data centre – this allows us to manage our deployment on MRC via a web-based dashboard, command-line tools or through RESTful services. IaaS provides us the storage, network, server and virtualization infrastructure, while allowing us to explicitly configure every other component of the system.

Four instances and 250GB of disk space are allocated to us on MRS for this project. We utilize all four instances in order to adhere to the non-functional requirements specified in Sec. 3.

4.3.1 Data Collector and Processor

Data collection is implemented via calls to Twitter API, which provides both standard Search API and Stream API for harvesting tweets in the past 7 days. However, standard Search API keeps a 15 minute access time limit and Stream API is also restricted to one connection with one developer access token. Hence, we created a hybrid crawler that leverages both Search and Stream APIs for tweet harvesting.

Firstly, we create a geo-location filter box (by longitude and latitude) around Australia, and collect all of the English language tweets that originate from this bounding box. Some tweets filtered from Stream API may not contains precise coordination as point longitude/latitude but only contain the area name and bounding box, the size of the bounding box varies

from a single confined point to a polygon that covers the entire city – in our work, we assign the location of the tweet as the centre of the bounding box.

Additionally, we implemented a suspend/wake mechanism for search API to cease querying after hitting rate limit and restart after a given time. In the twitter harvester, we also implemented an embedded sentiment analyzer to assign a polarity to the tweet prior to exporting the tweet to the CouchDB database.

We will pre-process the twitter data before storing it into the database. In order to deal with recurring tweets, we set the 'id' of a tweet, which is generated by Twitter, as the '_id' parameter when storing into the database. Since the CouchDB uses the '_id' as the only identification for each item, this will avoid the problem of storing duplicate tweets.

Additionally, since Twitter allows 140 - 280 characters per tweet, we stored the 'full_text' value from the original tweet object. The sentiment score and subjectivity score are analysed by using the Afinn python module on twitter text that has had stop-words removed via NLTK, and corrected for spelling errors via textblob. The sentiment score ranges from $-\infty$ to $+\infty$, with the negative score referring to a negative sentiment and vice versa.

4.3.2 CouchDB

One advantage of CouchDB is that it supports cluster operation and automatic replicator. In our system, the cluster is using default setting: q=8 and n=3, which means that each database (and secondary index) is split into 8 shards, with 3 replicas per shard. Therefore, there are 24 shard replica files distributed in 3 nodes. This redundancy means that the database can tolerate a failure of 2 out of the 3 nodes and retain data integrity. Moreover, due to the eventual consistency property of NoSQL databases, the active node will disseminate database updates to the failed nodes that rejoin the cluster.

Map/Reduce views For each of the scenarios, the Map/Reduce functionality of CouchDB was utilized to develop the Views inside the Design documents. Since a wide variety keywords could relate scenario one and two, the queries were written in such a way that more keywords (which may arise on the topic) could be added to the query of the Map function, without the need to edit the logic of the program much. An example is shown in Fig 2:

As can be seen in Fig. 2, the keywords array can be appended with additional keywords that may arise according to a theme. E.g. '#JobMaker'

```
function (doc) {
  var keywords = ['jobseeker','unemployment','centrelink','jobkeeper','welfare','dole','jobmaker','austudy','newstart'];
  var words = doc.text.toLowerCase().split(/\W+/);
  var words2 = doc.extended_tweet.full_text.toLowerCase().split(/\W+/);
  if ((words.filter(value => keywords.includes(value))).length > 0 && (doc.sentiment <= 0)){
      emit([doc.place.bounding_box.coordinates], doc.sentiment);
  }
  else if ((words2.filter(value => keywords.includes(value))).length > 0 && (doc.sentiment <= 0)){
    emit([doc.place.bounding_box.coordinates], doc.sentiment);
  }
}
```

Figure 2: MapReduce for JobSeeker.

```
function (doc) {
    if (doc.sentiment < 0) {
      emit([doc.place.bounding_box.coordinates], doc.sentiment);
    }
}
```

Figure 3: MapReduce for depression.

started trending on the 26th May 2020, and a slight edit to the query enabled tweets containing that keyword to be added to the view. The value emitted by the query is the sentiment of the tweet, consequently allowing for the selection of tweets that demonstrate extreme emotion when the view is called. For scenario 1, the Design Document consisted of two views, one that would retrieve all tweets with a sentiment score greater than 0 (positive sentiment), and the other that would retrieve all tweets with a sentiment less than 0 (negative sentiment). An example is shown in Fig 3.

The reduce queries used were just the '_stats' default, and no custom reduce queries were designed for the scenario. A further improvement/iteration would be to develop the reduce function further for each scenario to further drill down on other aspects of the tweet

Accessing CouchDB views (via REST) The views develop for each scenario were accessible to the front end via the HTTP, using the GET operation. The format of the data returned from a call was JSON, which was further processed by the webserver.

The general structure of the HTTP request to retrieve a view was,

```
http://${username}:${password}@${url}:5984/${dbname}/_design/
${design_document}/_view/${view-name}?reduce=false
```

Where the variables were defined as followed:

- ${username} – username of the couchdb cluster
- ${password} – password of the couchdb cluster
- ${url} – url of the couchdb cluster
- ${dbname} – database name of the couchdb cluster
- ${design_document} – design document of the couchdb cluster
- ${view name} – specific name of the view required

This rest point was accessible using the cloudant package in python, by the webserver.

4.3.3 Front End

Web Server The Nginx container image was used as the base for the webserver, as it is a well - defined open source reverse proxy server for HTTP, HTTPS, SMTP, POP3, and IMAP protocols, as well as a load balancer, HTTP cache, and a web server (origin server). This image was modified to create the custom webserver image, with the necessary dependencies for accessing the CouchDB API endpoint, as shown in the Dockerfile shown in Fig.4.

Data Processing There are two datasets required to make the analysis for our scenarios. The first one comes from the collected twitter data, which contains the coordinates from which the tweet was sent from. In order to obtain the said dataset, we are using python cloudant library as an interface to our database. Multiple endpoint addresses are stored in the program to access to various views on CouchDB. To adapt to any possible change of the CouchDB address, the IP address is parsed in as an argument of the python program. Since the twitter data was coming from Twitter Stream API, as explained earlier, we are taking only the centre point of the bounding box.

Th AURIN data, on the other hand required a bit more processing as it did not just need individual coordinates, but the geographical boundaries marking the statistical areas. After choosing the data to use, we used the data spatialising tool in AURIN to generate a geojson file containing the geographical boundaries. This file was however too large as it could go up to 20mb for Victoria or more than a 100mb for the whole of Australia and

```
FROM nginx:latest

ENV WELCOME_STRING "nginx in Docker"

WORKDIR /usr/share/nginx/html

COPY /city-analytics/nginx-webserver/front-end /usr/share/nginx/html

EXPOSE 80

ARG ip_address

RUN echo ${ip_address} && \
    cp index.html index_backup.html; \
    apt-get update && apt-get install -qy vim; \
    apt-get install -y python3 && \
    apt-get install -y python3-pip && \
    pip3 install folium && \
    pip3 install pandas && \
    pip3 install cloudant && \
    chmod +x htmlupdated.sh && \
    pip3 install shapely && \
    python3 map.py ${ip_address}

CMD nginx -g 'daemon off;'
```

Figure 4: Dockerfile for setting the environment of the webserver.

cause loading of the page to be too slow. To deal with this issue we used a vector simplification tool called MapShaper which uses a line simplification algorithm called the Visvalingam's algorithm to remove up to 90% of coordinates while keeping the topology of the map. After that we then processed the resulting geojson using geojson-precision to limit the number to limit the number of decimal points to 4. This greatly compressed the data to 10mb or less without losing too much accuracy from the map. Considering the AURIN data is completely static, it is made directly accessible by storing them in the container.

After reading the AURIN data and receiving the tweet coordinates from the views, they are then processed into proper formats for the use in later stages. The next step is aggregating the coordinates on areas defined by the AURIN data. This is made possible with the python library Shapely, which can help decide whether a polygon contains a certain point on a 2-dimensional plane. There is one caveat to this process, the data compression from the previous section is causing boundary polygons to be self-intersecting and ultimately causing the program to fail. Luckily, this issue can be solved by buffering the polygons as a mean of anti-aliasing.

Map Rendering To visualise the map, we decided to use Folium, a Python library which allows the user to create different types of Leaflet maps. Leaflet is a JavaScript library for embedding interactive maps on the front-end. By using it, we avoided the low-level detail of front-end language such as HTML, CSS and JavaScript so we can focus on the presentation and the analytics. Also, Folium can process geojson data and comes with a diverse range of mapping features which are useful to our project. The layer structure of our Leaflet map is four-fold. The base of the map is a choropleth, which takes the attributes of the AURIN dataset and calculates a scale based on the factors mentioned in the scenarios. This layer represents the ground truth demographic distribution of the topic of interest. Each area is covered with an assigned depth of colour, where a larger number indicates a darker colour. For example, the unemployment rate attribute is retrieved from the Aurin dataset and colours are assigned to areas on the map with darker colours being a higher unemployment rate and vice versa. Due to the inherited inaccuracy of tweet coordinates, heatmap would be a better visual representation of the location of tweets. Then on the top of all layers is an invisible layer that served as a popup layer. The tweet aggregation statistic for that area will pop up upon clicked. Last but not least, the correlation analysis graph is always floating in the bottom left corner

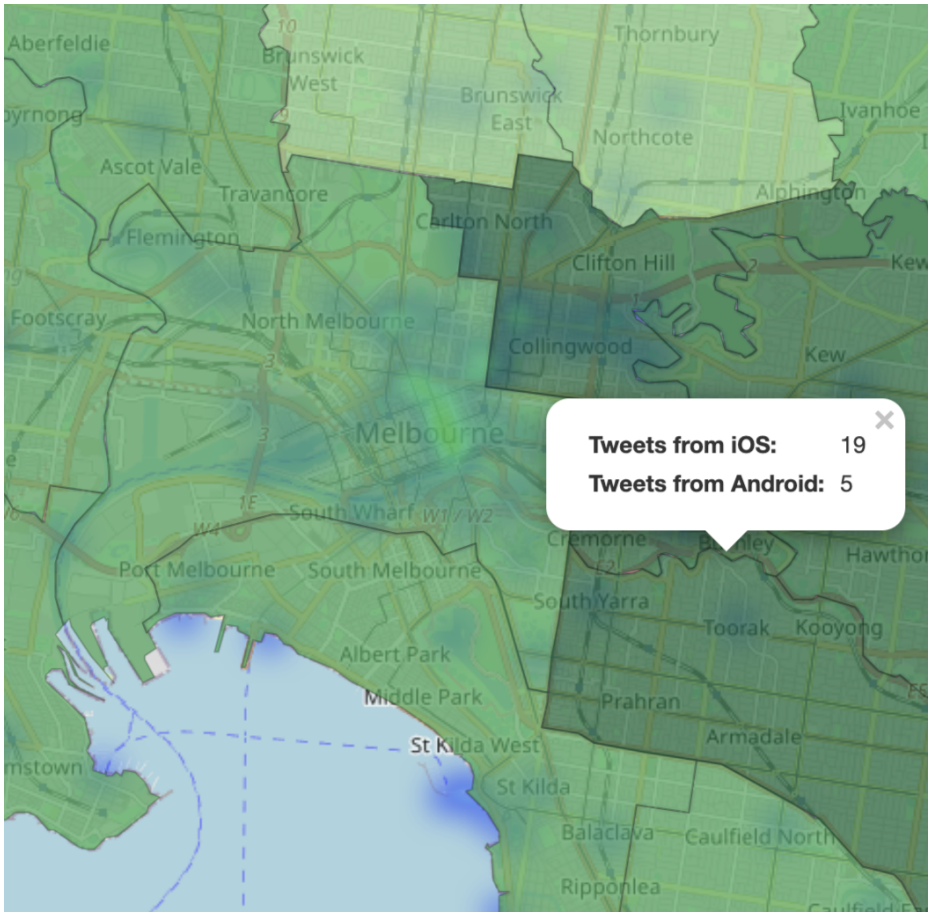


Figure 5: Screenshot of the front end of our application.

for user's reference. Fig. 5 shows a picture of our front end design.

5 System Deployment

5.1 General Deployment

The system was deployed using Ansible, a software provisioning tool which allows for ad-hoc task execution and configuration management. A long sequence of instructions was developed in an Ansible 'playbook', which is written in YAML. This structure of the playbook is shown in Fig. 6.

The deployment of the system was separated into four stages:

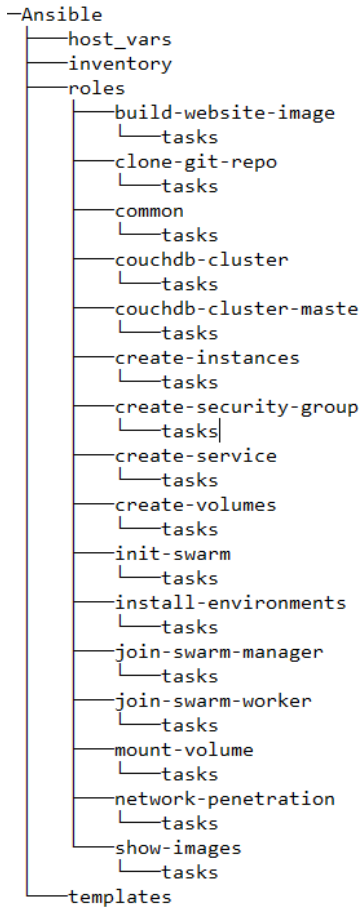


Figure 6: Ansible playbook for our system.

- Deployment of virtual machines on the Melbourne Research Cloud (MRC)
- Installation of key environment dependencies
- Install and deployment of the CouchDB database cluster and twitter harvester
- Installation and deployment of the Docker Swarm and the website

The execution details of each stage are further described below, as well as the intended outcomes of each step.

5.1.1 Deployment of virtual machines

The first stage of the deployment was of the virtual machines (instances) on the MRC, coupled with the creation of the volumes associated with each instance. This can be executed with the following command:

```
./launch-nectar.sh
```

The command launches the playbook called launch-nectar.yaml, which creates the necessary security groups, creates the volumes and creates four instances. The instances are given the appropriate labels which will then be utilized later in the deployment sequence. These labels are whether the instance is a dbMaster or dbSlave (for CouchDB), a leader or worker (for Docker Swarm).

5.1.2 Installation of key environment dependencies

The next stage of the deployment is the installation of the environments dependencies required for both the database and webserver applications, which was Docker. The execution of this step is performed with the following script,

```
./install-environments.sh
```

This shell script runs the playbook called install_environments.yaml. Moreover, this playbook also performs key tasks to provide networking access to the instances, as well as the Docker containers and Docker Swarm launched on them. This was done by adding a series of proxy addresses to the environment variables.

5.1.3 Install and deployment of the CouchDB database cluster

This stage of the deployment creates the CouchDB containers of three of the instances and sets up the CouchDB cluster. The execution is performed by executing the following script,

```
./couch-setup.sh
```

Once the containers are set up, the database is created, and the design documents are imported into the CouchDB database – where the views can be accessed.

5.1.4 Installation and deployment of the Docker Swarm (website and twitter harvester)

The final step of the deployment is launching of the initialization of the Docker Swarm, and consequently the webserver – where the front end is hosted. The following command is executed to enable this,

```
./deploy-swarm.sh
```

This shell script runs the playbook called deploy_swarm.yaml, which runs a series of commands to build to initialize the swarm, build the website image, build the twitter harvester image and create two docker services across all the nodes. After this step, the system infrastructure is set up.

5.2 Scaling of system

A playbook was also developed with allowed for scaling of the system, to increase the nodes available for the CouchDB clusters, as well as adding another worker to the Docker Swarm (another webserver). This creates the ability for the system to handle greater load on the webserver, in the scenario that there may be an increased number of visitors accessing the website, as well as increased capacity of storage and querying on the CouchDB cluster.

A separate playbook was created for this feature, with the structure shown in Fig.7.

To execute the scaling, the following shell script must be executed with three arguments. One argument variable specifying the number of extra instances to be added, a second one which specifies the IP address of the CouchDB masternode and the third one which specifies the IP address of the Docker Swarm leader.

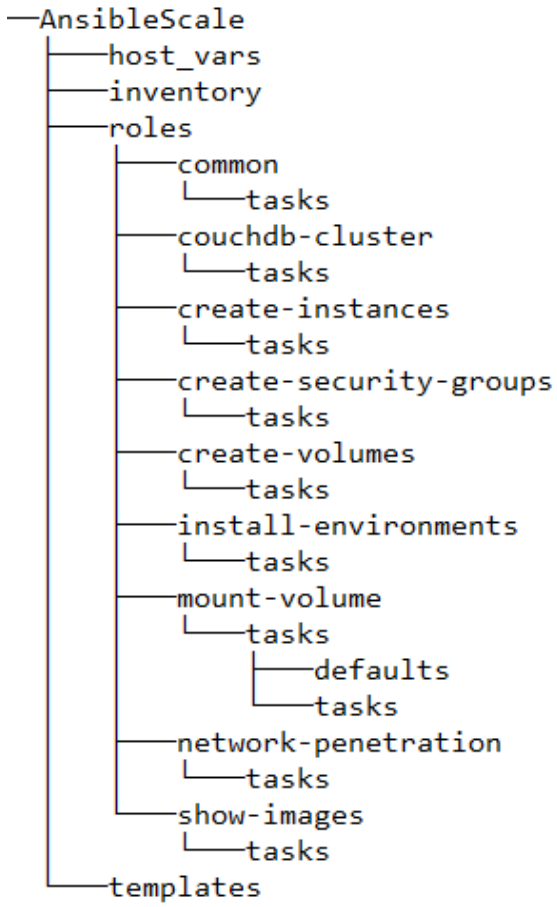


Figure 7: Ansible playbook for scaling.

```
#!/bin/bash
# set n to 1
n=1
now=$(date +'%d%m%Y%H%M%S')
# continue until $n equals 5
while [ $n -le $1 ]
do
    echo "Welcome scaling cluster $n"
    sed -i -e 's/addition_instance=[0-9].*/addition_instance='"${now}"'/g' ./inventory/hosts.ini
    ./launch-nectar-servers.sh
    ./install-environments.sh
    n=$(( n+1 ))     # increments $n
done
```

Figure 8: Bash script.

`./scalecouch.sh [number]`

This shell script executes the playbook to launch a single instance, with a CouchDB container which then connects to the existing CouchDB cluster and joins the existing Docker Swarm as a worker. The shell script is shown in Fig. 8.

5.3 Security

The security of our system relies primarily on the inbuilt security features of the MRC service. One can only access our instances under the network environment of UoM LAN. Besides, a unique private key is required for connecting to each instance. Moreover, a security group is attached to each instance with only the ports 5984, 4963, 9100-9200 being open – most of these are restricted to intra-cluster communications. These CouchDB ports are not open to the public and only port 5984 can be used to access the CouchDB instance. Additionally, CouchDB is secured by a username and password. Although we typically set these to 'admin:admin' throughout the building and testing stage, these are replaced with more complex ones for the final system deployment.

5.4 Fault Handling

5.4.1 Twitter Harvesting

The twitter streaming API (tweepy) was running continuously, and it was observed that it had the tendency to throw errors such as incomplete reading of data – caused by interruptions in the network connection with Twitter. As the collection of data was required to be continuous, as interruptions would not only lead to increased monitoring require of the program, but

```
1
2     while True:
3         try:
4             listener = Listener()
5             auth = OAuthHandler(CONSUMER_KEY, CONSUMER_SECRET)
6             auth.set_access_token(ACCESS_TOKEN, ACCESS_TOKEN_SECRET)
7             stream = Stream(auth, listener)
8             stream.filter(locations = [113.338953078, -43.6345972634, 153.569469029, -10.6681857235])
9         except Exception as e:
10            with open("error_log.txt", 'a') as file2:
11                file2.write(str(e) + '\n')
12
```

Figure 9: Python code to avoid harvester interruptions.

also potential irregularities in the data. If there was an instance where there was an interruption of twitter data collection during an important period, correlations of the scenarios would be significantly impacted.

For the twitter streaming/harvesting program to continuously scrape data without any interruption, a try/catch mechanism was implemented inside a while loop (see Fig. 9).

The loop shown in Fig. 9 would enable the continuous operation of the program to scrape tweets from the Twitter API endpoint, without interruptions caused by errors. Errors are saved in an error log (error_log.txt) which can be analyzed for debugging, as well as further improvements of the program to evade errors which may arise.

5.4.2 Docker Swarm – Container Reloading

Docker swarm provides the healing capability that allows the system to recover in case of container failure. If a container stops, the swarm will create another container to replace it and maintain the same number replicas.

As the system counts with 3 managers, it can handle one manager failure. If more nodes join the swarm, the system can handle their failure. This means, that the Application will still be available despite the failure. If more than one manager fails, the Application will continue running, but a new cluster needs to be created to recover.

5.4.3 Docker Swarm – Updates

If a website update is needed, we can roll updates across the nodes by cloning the git repository with the updated website, build the images, tag and push it to the Docker registry, and run the docker swarm update command.

Command for image building:

```
sudo docker build --build-arg ip\_address=\${masternode} -t
website -f city-analytics/nginx-webserver/Dockerfile .
```

Figure 10: Execution of Docker Swarm update.

Command for swarm update:

```
sudo docker tag website isaacpedroza/city-analytics:website
```

The execution of the swarm update is shown in Fig. 10.

6 Difficulties and Limitations

- An instance cannot be immediately accessed after its creation. Our Ansible script is set to ascertain that the instance has been successfully spun up prior to applying further configurations.
- The sentiment analysis tools used in this system have a limitation in judging the sentiment of a tweet. Afinn library assigns a sentiment score to 3000+ words, and the aggregated sentiment score is obtained by summing the individual scores of every word. However, this is not capable of discerning a myriad of potential emotions within the text, as well as correctly gauging sarcasm. A possible improvement would be to implement RNN or BERT models (pre-trained on social media text), but given the amount of tweets available, such an approach may be computationally expensive.
- Twitter rate limit restrictions result in 'GET statuses/user_timeline' having a rate limit of 900 requests per 15 minute window. In order to avoid potential disconnections situation affected by rate limit error, we set our Twitter harvester to sleep for 180 seconds when a rate limit error occurs.
- Although Twitter has a large amount of data readily available for analysis, only a small fraction of this data contains location information (either via a bounding box or location co-ordinates). This makes it difficult to obtain statistically significant amounts of data to perform spatial analysis on a fine-grained scale. This can be potentially ameliorated by using machine learning and network analysis in order to predict user location based on tweet content and social interactions.

- Twitter has a restriction that only the latest 7 days of historical tweets can be collected via the standard search API. Hence, a large amount of data is difficult to collect in practice, affecting the quality of the final analysis.
- In order to deal with the potential problem of conflicts b/w duplicate tweets, we decide upon using an integer representation of the unique identifier that is provided by Twitter API as the '_id' parameter when exporting into the database (instead of using the automatically generated '_id' by CouchDB). Since '_id' and 'rev' represents CouchDB's inbuilt version-control system for a particular instance of a document, the issue of duplicate tweets is therefore avoided.
- In our project, the AURIN datasets were manually constructed, downloaded and processed. This increases the maintenance cost of the application in the long run. It would be ideal to have data retrieved from AURIN automatically through programmatic API calls – this would allow new data to be easily added and updated on a regular basis.

7 Results

In this section we discuss the salient results of each Scenario presented in Section 2. All features are standardized to a mean of 0 and a standard deviation of 1 in order to scale them to similar magnitudes (all features are measured at different scales), while preserving the statistics of the distribution. For each scenario, we fit a line of best fit via Linear Regression, and deduce the coefficient of determination (R^2) of the fit. R^2 is given by:

$$R^2 = 1 - \frac{\sum_i (y_i - f_i)^2}{\sum_i (y_i - \bar{y})^2}, \tag{1}$$

where y_i are the empirical measurements, $\bar{y}$ is their mean, and f_i are the model predictions. This metric provides a measure of how well observed outcomes are replicated by our regression model, based on the proportion of total variation of outcomes explained by the model. Effectively, R^2 measures the goodness-of-fit of regression and normally ranges from 0 to 1 (although negative values are possible in cases of extremely poor fit).

Additionally, to measure the linear correlation of the data, we employ the Pearson Correlation Coefficient (P). Pearson correlation coefficient is given by:

$$P = \frac{\sum_i (x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_i (x_i - \bar{x})^2} \sqrt{\sum_i (y_i - \bar{y})^2}}, \tag{2}$$

and is a measure of the linear correlation between two variables x and y. It has a value between +1 and -1, where 0 represents no linear correlation and +1 and -1 represent positive and negative correlations respectively. It is important to note that P is invariant to linear scale transformations, and only reflects the strength and direction of a linear relationship – it is incapable of capturing non-linear aspects as well as the exact magnitude of a linear relationship. Alternatively, Mutual Information with equal-frequency or equal-range binning can be utilized to ascertain non-linear correlations – however, we do not examine Mutual Information in this work.

We also calculate Spearman's rank correlation coefficient (S) for each scenario. Spearman's coefficient is given by:

$$S = \frac{\text{cov}(R_x, R_y)}{\sigma_{R_x}\sigma_{R_y}}, \tag{3}$$

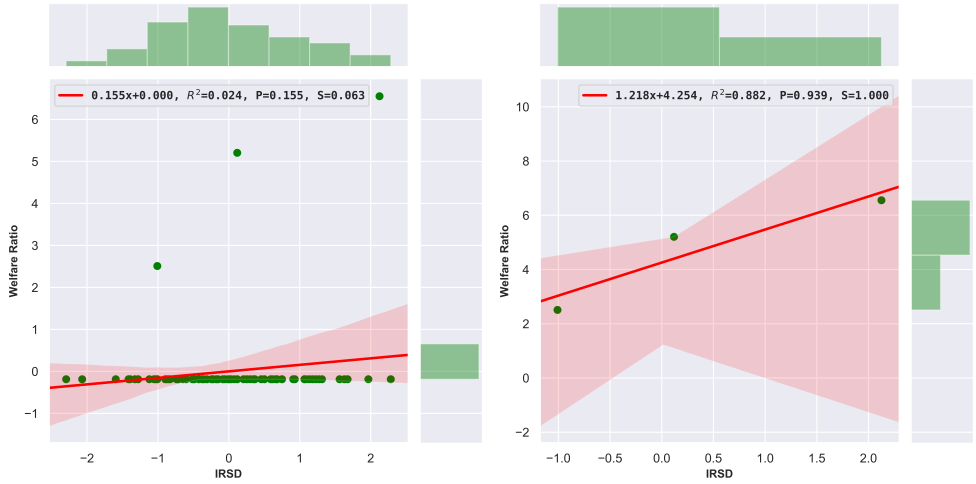
where $\text{cov}(R_x, R_y)$ is the covariance of the rank variables[1] and $\sigma_{R_x}, \sigma_{R_y}$ are the standard deviations of rank variables. S provides a non-parametric measure of statistical dependence between the rankings of two variables (i.e. it assesses how well the relationship between the two variables can be described by a monotonic function). Spearman coefficient ranges from +1 to -1, is appropriate for both continuous and discrete ordinal variables, and is capable of gauging a non-linear monotonic relationship.

7.1 Scenario 1

This scenario examines the possibility of correlation between the IRSD composite metric of LGA (i.e. its metric of social disadvantage) and the ratio of tweets containing favourable and unfavourable sentiments regarding welfare. The experimental finding are shown in Fig.11. It is difficult to ascertain the presence of a trend in the data due to the statistically insignificant sample size – we managed to extract approximately 150 tweets regarding welfare in Australia. We find this fact surprising, considering how prominent the topic of welfare has been in recent months due to the international COVID-19 pandemic crisis, the consequent economic shutdown in Australia, and the unprecedented Federal Government stimulus package for those affected by the crisis.

Many data points have a Welfare Ratio close to 0 since most Victorian regions recorded no welfare-related tweets whatsoever. The resultant linear regression shows slight positive correlation, but is not a meaningful model

[1] i.e. raw scores x_i, y_i converted to ranks.,



(a) (b)

Figure 11: IRSD metric versus the ratio of positive and negative sentiment tweets (related to welfare) where **a)** all Victorian LGAs are shown, and **b)** Victorian LGAs with no positive-sentiment tweets are excluded.

due to R^2 being close to 0. P and S show that no significant linear or monotonic relationship exists in the data. If we remove the data points for which no statistics have been collected, the positive correlation becomes more apparent. R^2, P, and S near 1, signify a good fit by the linear model, as well as meaningful and monotonic correlation between the variables. However, due to insufficient statistics for this scenario, we caution against a premature interpretation of this result.

If, upon the collection of more data, it turns out that a positive correlation does, indeed, exist between the variables, this result would run contrary to our original hypothesis. Contradicting previous research, it would mean that regions with more advantageous socio-economic environment have a generally more favourable opinion of welfare.

7.2 Scenario 2

In this scenario we examined whether there is a statistically-meaningful relationship between the wealth of a region and the propensity of the residents living therein to choose one brand of mobile platform versus another (i.e. iPhone vs Android). For the whole of Australia, we manage to extract 103k and 65k tweets that are marked as originating from an iPhone and Android

devices respectively. Of these, 45k have been posted in Victoria. The ratio of android to iPhone tweets versus the wealth metric of a Victorian region in shown in Fig. 12.

Interestingly, we find that the number of tweets that originate from an iPhone dominates the number of those that originate from an Android platform 2:1 (30k vs 15k). Considering the fact that Android devices own 86.2% of the market share in comparison to 13.8% of iPhone [10], we are at a loss to explain such discrepancy between expectations and observations. We posit that such bias towards iPhones could be due to a combination of several factors – for example:

1. iPhone may be the preferred platform of choice for the younger age bracket that are more active on social media platforms like Twitter (i.e. 10-30 y.o.) [8],

2. iPhone platform is somehow more prone to encoding the origin of the tweet within the tweet metadata than Android.

The linear model shows to be a poor fit to the data ($R^2 \approx 0$). However, some linearity and monotonicity can be seen in the data ($P \approx 0.3$ and $S \approx 0.3$). The scatter plot seems to suggest that there is some kind of positive correlation between the wealth of the region and a higher preference towards the Android platform.

This is in contradiction to our hypothesis in Section 2. and is a surprising result considering that previous studies have demonstrated the higher average earning potential of iPhone users in comparison to Android users [4]. Although the exact relationship between variables is unclear at this stage, we are at a loss to explain the positive correlation suggested by our data.

7.3 Scenario 3

In this scenario we examined the relationship between the ratio of the number of tweets with strong positive and strong negative sentiment with the mental health metric of a region. For the whole of Australia, we extracted 62k tweets (positive and negative), of which 25k are in Victoria (16k/9k positive/negative). The remaining tweets in the database have a 'neutral' sentiment polarity and are not considered. The results of our experiment is shown in Fig. 13.

Due to the insignificant R^2, the linear model is a very poor fit to the data. Additionally, $P < 0$ and $S < 0$ suggest that a negative correlation, if any, may exist in the data. We expect a negative correlation between the

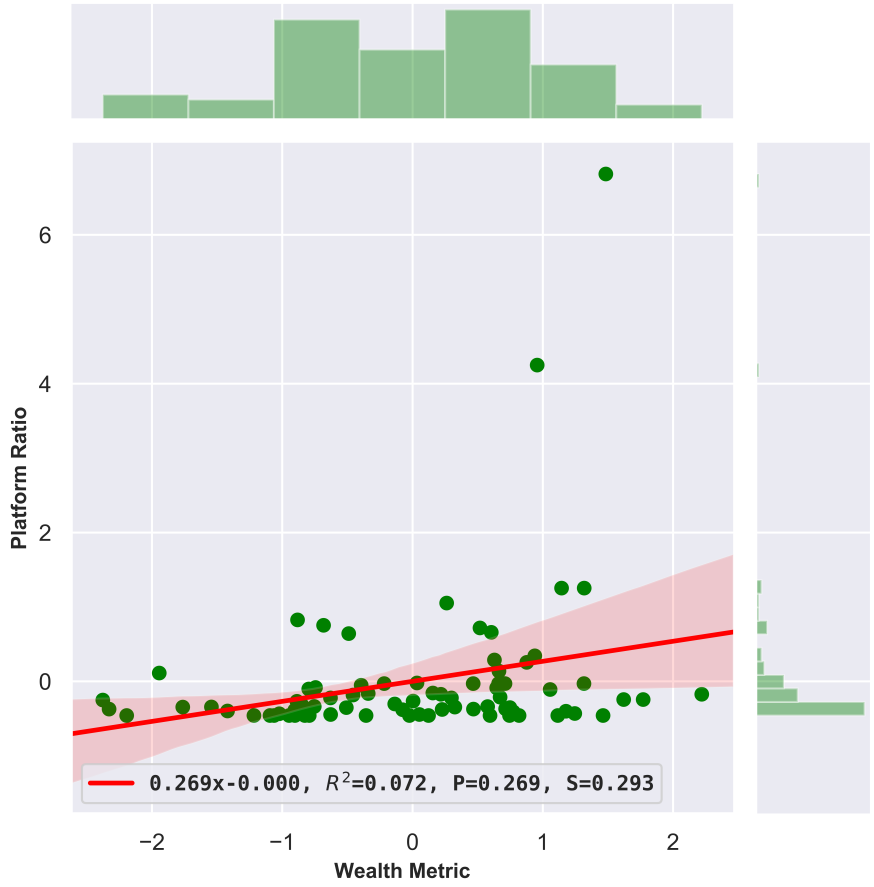


Figure 12: Wealth metric versus the ratio of Android-to-iPhone usage.

sentiment ratio and mental health metric, since a region with higher number of hospital admissions (per 100k residents) due to psychological concerns (i.e. higher Mental Health Metric) is likely to contain a higher proportion of residents with mental illnesses (such as depression). Such residents are much more likely to disseminate tweets with strongly negative sentiment. Although there are some outliers, a possible negative correlation may exist and would be in accordance with our hypothesis.

8 Conclusion

This work has developed a cloud-based solution to geospatial analysis of integrated data extracted from social-media and government statistics. The system is designed with automation and scalability in mind. The system has to be highly available and always online. To achieve this, replication is applied at multiple places to increase availability of the system and guarantee the system has high fault tolerance. For example, instead of using a single CouchDB node, a cluster of nodes is used to facilitate these requirements.

At the centre of the system architecture is the document-oriented NoSQL database CouchDB. The database stores each tweet in a document format which is subsequently filtered and retrieved through CouchDB-native MapReduce function. Additionally, the system provides real-time tweet harvesting and sentiment analysis, allowing for a range of sentiment-oriented research to be conducted on the collected social-media data. The system supports sharding in order to provide data safety and redundancy, and is configured to operate in a Docker swarm to enhance fault tolerance and availability. Additionally, the Ansible auto-deployment script allows the system to be easily scaled up as more cloud resources become available or the load on the system increases. The resultant system is reasonably flexible and is able to support a multitude of research areas that can benefit from Big data analytics of social media.

We demonstrate the capability of our system to aid socio-economic research in Australia via an analysis of potential correlations in Victorian Local Government Areas between:

1. tweet sentiment regarding welfare and socio-economic advantage,
2. tweet platform of choice and regional affluence,
3. average tweet sentiment and the incidence of psychological disorders.

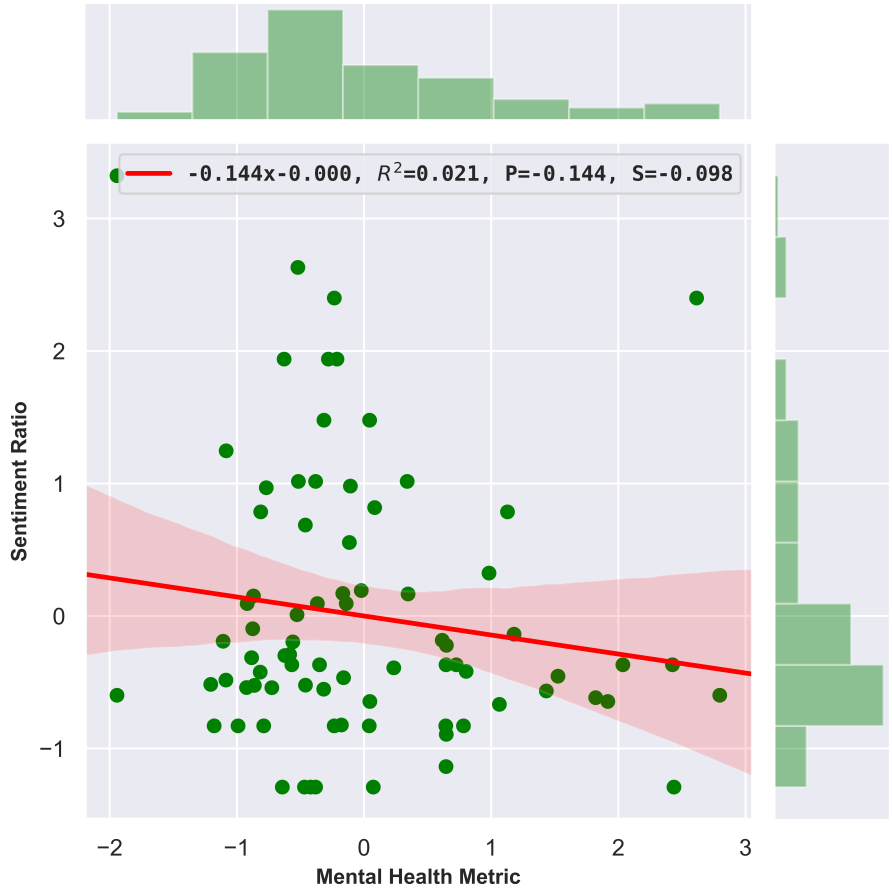


Figure 13: Mental health metric versus the ratio of tweets with strong negative and positive sentiments.

Contrary to expectations, we find a potentially positive correlation between the positive sentiment regarding welfare and socio-economic well-being of a region, although more data is required to ascertain the significance of the correlation. In addition, we find that there may be a positive correlation between preference for Android and wealth of a region. Lastly, empirical results suggest that a possible negative correlation may exist between the fraction of tweets with strong negative sentiment and the incidence of serious mental illness. However, in order to more reliably ascertain the presence of correlations, we need to ameliorate the effects of noise and outliers - this requires more empirical data to be collected.

Acknowledgments

The group would like to thank their friends and family for understanding (and tolerating) the prolonged coding/debugging sessions that each group member had to undertake for this project.

References

[1] Australian urban research infrastructure network (aurin). https://aurin.org.au/. Accessed: 2020-05-27.

[2] Census of population and housing: Socio-economic indexes for areas (seifa), australia, 2016. https://www.abs.gov.au/ausstats/abs@.nsf/Lookup/by%20Subject/2033.0.55.001~2016~Main%20Features~IRSD~19. Accessed: 2020-05-27.

[3] Household income economic wellbeing indicators(a), australia, 2007–08 to 2017–18. https://www.abs.gov.au/household-income. Accessed: 2020-05-27.

[4] iphone users spend 101 every month on tech purchases, nearly double of android users, according to a survey conducted by slickdeals. https://www.prnewswire.com/news-releases/iphone-users-spend-101-every-month-on-tech-purchases-nearly-double-of-android-us
html?c=n. Accessed: 2020-05-27.

[5] Melbourne research cloud documentation. https://docs.cloud.unimelb.edu.au/. Accessed: 2020-05-27.

[6] Mental health services in brief 2018. https://www.aihw.gov.au/getmedia/0e102c2f-694b-4949-84fb-e5db1c941a58/aihw-hse-211.pdf.aspx?inline=true. Accessed: 2020-05-27.

[7] National health survey: First results, 2017-18. https://www.abs.gov.au/ausstats/abs@.nsf/Lookup/by%20Subject/4364.0.55.001~2017-18~Main%20Features~Mental%20and%20behavioural%20conditions~70. Accessed: 2020-05-27.

[8] Over 80apple. https://amp.businessinsider.com/apple-iphone-popularity-teens-piper-jaffray-2018-4?__twitter_impression=true. Accessed: 2020-05-27.

[9] Q1 2019 earnings report. https://s22.q4cdn.com/826641620/files/doc_financials/2019/q1/Q1-2019-Slide-Presentation.pdf. Accessed: 2020-05-27.

[10] Share of global smartphone shipments by operating system from 2014 to 2023. https://www.statista.com/statistics/272307/market-share-forecast-for-smartphone-operating-systems/. Accessed: 2020-05-27.

[11] Social media statistics australia – january 2020. https://www.socialmedianews.com.au/social-media-statistics-australia-january-2020/. Accessed: 2020-05-27.

[12] L. Bass, P. Clements, and R. Kazman. *Software architecture in practice*. Addison-Wesley Professional, 2003.

[13] E. E. Ekins. What americans think about poverty, wealth, and work. *The Cato*, 2019.

[14] R. T. Fielding and R. N. Taylor. *Architectural styles and the design of network-based software architectures*, volume 7. University of California, Irvine Irvine, 2000.

[15] R. C. Kessler and E. J. Bromet. The epidemiology of depression across cultures. *Annual review of public health*, 34:119–138, 2013.

[16] F. Lazarevic. Public opinion on welfare: An analysis of survey data.

[17] U. Sila and V. Dugain. Income poverty in australia. 2019.