



THE UNIVERSITY OF
MELBOURNE

COMP90024 Cluster and Cloud Computing
Twitter Data Harvesting and Analysis
Assignment 2 Report: Team 48

By

Arnav Garg (Student ID: 1248298)

Piyush Bhandula (Student ID: 1163716)

Gurkirat Singh Chohan (Student ID: 1226595)

Vishnu Priya G (Student ID: 1230719)

Jay Dave (Student ID: 1175625)

The University of Melbourne

May 25, 2021

ABSTRACT

This report discusses the application architecture and the overall system which was developed for the COMP90024 Cluster and Cloud Computing course at the University of Melbourne (Semester 1, 2021). For this assignment, the team decided to focus on three scenarios and for each of these respective scenarios there will be different types of graphs and analysis associated with it. This report incorporates a detailed explanation of each component of the system developed. The source code for this assignment, which is publicly available, can be found under the following.

Git Repository URL:

<https://github.com/arnavgarg123/COMP90024-ClusterAndCloudComputing-Assignment2>

YouTube Links:

1. Deployment: https://youtu.be/KpFQ2QEE_Hw
2. Webapp demo: https://youtu.be/_WJC8DAGJRA

Table of Contents

| | |
|---|----|
| 1 Introduction | 4 |
| 2 Scenario Overview: Three Scenarios | 5 |
| 2.1 Scenario One | 5 |
| 2.2 Scenario Two | 6 |
| 2.3 Scenario Three | 7 |
| 3 System Architecture | 8 |
| 3.1 Overview of the System Design | 8 |
| 3.2 Deployment of the System | 9 |
| 3.3 Web Server and WSGI layer | 10 |
| 4 Cloud Infrastructure | 10 |
| 4.1 Unimelb Research Cloud Infrastructure | 10 |
| 4.2 Ansible Cloud | 11 |
| 4.3 Docker Container | 13 |
| 5 Data Gathering | 14 |
| 5.1 Twitter Harvester | 14 |
| 5.2 Aurin Data | 15 |
| 5.3 GeoJSON | 15 |
| 6 Data Storage | 15 |
| 6.1 CouchDB Cluster Formation | 16 |
| 6.2 Database | 17 |
| 7 Fault Tolerance and Error Handling | 17 |
| 8 Data Visualization | 18 |
| 8.1 Maps | 18 |
| 8.2 Graphs | 19 |
| 9 User Guide | 26 |
| 10 Team Collaboration | 27 |
| 11 Conclusion | 27 |
| 12 Future Work | 28 |
| 13 References | 28 |

1 Introduction

Over the years, social media and its use have evolved in a way that it has become an integral part of a lot of people's lives. Individuals share their feelings and highlights of their lives to express themselves. Many firms worldwide use this data, the content shared by the users to the public, to put themselves in a profitable position. For example, a firm may target a specific group of people living in a particular region to market their product. This is because the firm believes that this product can cater for the wants and needs of the public living in this particular region and therefore the product has a great chance of performing well in terms of sales and generating greater revenue for the firm. In this project, we are using Twitter which is one of the biggest social media platforms. The team decided to harvest tweets from across the cities of Australia on the UniMelbResearch Cloud and undertake a variety of social media data analytics scenarios. There will be a total of three scenarios which the team will be focusing on. In each of these different scenarios, there will be different types of graphs and charts associated with it to better analyze the data gathered. For data collection, the Twitter Harvester will be used along with the Australian Urban Research Infrastructure Network (AURIN). For each of the scenarios, different types of analysis are performed. The main objective is to see whether in these scenarios there any interesting correlation(s) between the variables are and if it can help improve our knowledge of life in Australia.

The design of the overall system and the components needed to deploy the system on the University of Melbourne Research Cloud are described in section 3. This system is designed to operate without any issues in a cloud environment. We use Ansible to automate creation of instances and deployment of the application. Melbourne Research Cloud uses Openstack as the cloud infrastructure. In section 4, the entire cloud infrastructure regarding this project is discussed in further detail. For this task, a large Twitter dataset consisting of tweets from across Australia (Sydney, Melbourne, Adelaide, Perth and several other cities) was collected. The dataset was extracted from Twitter using specific keywords. Data collection is further discussed in section 5 and what resources were used to gather the data. The system employs a CouchDB database for the storage of the data. The way we have utilized CouchDB for this system is discussed in further

detail in Section 6. The system is robust and fault tolerant due to our effective handling of errors, should they arise. The approaches we have taken for error handling are further elaborated upon in section 7. Users of this system can inspect various analytics through our data visualisation tool which is in the form of a web-based application. The data visualization tool comprises Backend and Frontend components. The Backend is used for fetching data and processing it from the CouchDB database and the Frontend demonstrates the results. This visualization tool was created using the Python Flask framework. In section 8, the data visualization tool, including the results we obtained, are discussed in further detail.

2 Scenario Overview: Three Scenarios

Scenario 1 provides a general overview of life in Australia, covering a wide range of topics from political party preference to the most frequently used hashtags in tweets. In scenario 2, analysis was performed on tweets relating to Covid-19 to see how people are feeling with regards to the ongoing pandemic. Finally, in scenario 3, we delve into unemployment in Australia and its possible correlation with education levels. The analysis for this scenario was performed on a subset of tweets related to work and education.

2.1 Scenario One

Scenario 1 provides an overview of life in Australia. In this scenario, the various types of graphs and what they represent are outlined below:

1. Combination bar and line chart depicting the relationship between population counts (bars) and total number of tweets (line) for certain Australian cities. We are attempting to determine if there is any correlation between the two variables, i.e., if a city has a higher population, does it mean there are more tweets from that city.
2. Combination bar and line chart depicting the relationship between average income (bars) and calculated sentiment scores of tweets (line) for certain Australian cities. We are attempting to determine if there is any correlation between the two variables, i.e., if people

in a particular city have a higher average income, does it mean that the people living in that city express more positive sentiment as observed through their tweets.

3. Pie chart showing which major Australian political party (out of top two) is favored based on sentiment score of related tweets. Here, we are attempting to extract valuable insight of political party preference based on the sentiment expressed by Twitter users in their tweets related to these political parties.
4. Word cloud showing the most frequently used hashtags by Twitter users from all over Australia. The word cloud enables us to gain insight into which topics interest people in Australia the most.

2.2 Scenario Two

Scenario 2 seeks to gain insight into how people in Australia are feeling with regards to the ongoing Covid-19 pandemic. We will be using the terms polarity score and subjectivity score rather extensively in our description and analysis and hence, we shall first define these terms. Polarity score refers to a number lying in the range $[-1,1]$ where 1 means positive statement and -1 means a negative statement. The subjectivity score refers to a number within the range $[0.0, 1.0]$ where 0.0 is very objective and 1.0 is very subjective.

In this scenario, the various types of graphs and what they represent are outlined below:

1. Histogram of polarity score of Covid-19 related tweets. We are trying to determine if Covid-19 related tweets from Australia are mostly positive, negative or neutral statements in order to gain insight into people's general feelings towards the pandemic.
2. Heat map depicting the relationship between polarity score and subjectivity score of Covid-19 related tweets from Australia. We are attempting to determine in which range of score values the densities are highest and any correlation between both types of scores.

3. Combination bar and line chart depicting the relationship between total number of tweets (bars) and number of Covid-19 related tweets (line) for certain locations across Australia. We are attempting to determine if there is any correlation between the two variables, i.e., if a location with higher total number of tweets, has high Covid-19 related tweets from that very location.
4. Word cloud showing the most frequently used keywords in Covid-19 related tweets from all over Australia. The word cloud enables us to gain insight into which aspect related to Covid-19 people in Australia are talking about the most.

2.3 Scenario Three

Scenario 3 focuses on unemployment in Australia and seeks to determine if there is any relationship between unemployment rate and highest level of education attained. The unemployment rate here refers to the number of unemployed people expressed as a proportion of the total labour force (employed and unemployed).

In this scenario, the various types of graphs and what they represent are outlined below:

1. Combination bar and line chart depicting the relationship between polarity score of work-related tweets (bars) and unemployment rate (line) for 5 major Australian cities. We are attempting to determine if there is any correlation between the two variables, i.e., if a city with higher polarity score for work related tweets (implying more positive statements), has a lower unemployment rate as well.
2. Word cloud showing the most frequently used keywords in work and education related tweets from all over Australia. The word cloud enables us to gain insight into which aspect related to work and education people in Australia are talking about the most.
3. Combination bar and line chart depicting the relationship between subjectivity score of work-related tweets (bars) and unemployment rate (line) for 5 major Australian cities. We

are attempting to determine if there is any correlation between the two variables, i.e. a city having a higher subjectivity score of work-related tweets (implying more emotion-based rather than fact-based tweets), has a higher unemployment rate too.

4. Line graph showing relationship between unemployment rate and highest level of education attained by people across the whole of Australia. Here, we are attempting to determine if a higher level of education means low levels of unemployment and vice versa.

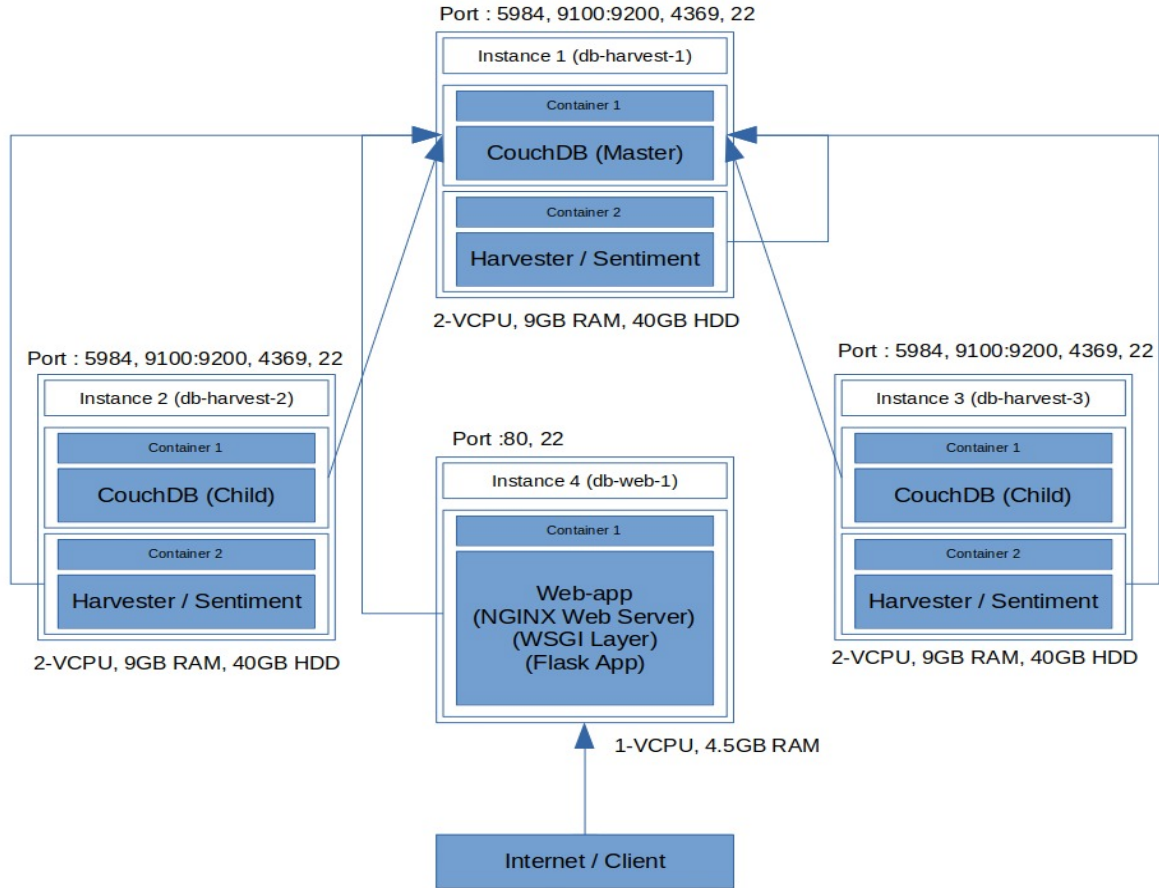
3 System Architecture

The architecture of the system developed for this project has been designed in a way where scalability and reliability are the primary focus and therefore the system is made up of multi-layer architecture where each layer has certain responsibilities:

- Data Gathering
- Data Storage
- Data Analytics
- Data Visualization

3.1 Overview of the System Design

We have to essentially build a pipeline of data which can then handle the amount of data which is being streamed into the system in real-time. The tweets are gathered through twitter harvester and sentiment analysis is performed at the same time. Then this data is pushed to the database in place known as *tweet*.



3.2 Deployment of the System

The whole system is deployed on a total of four computing instances running on the University of Melbourne Research Cloud. An overview of the computing nodes as well as their respective services is given in the above architecture diagram. Three of the four computing instances that were deployed on the cloud (server-01, server-02 and server-03) have been designated as “Data Nodes”. Each Data Node runs an instance of the Twitter Harvester and the CouchDB database in a Docker container environment, thereby providing the opportunity for future scaling in case it is required. Server-04 has been designated as the “Processing Node” which takes care of processing and analyzing the data. It also hosts the components that are required for running the web-based visualization frontend. All components again are being run on a Docker container environment.

3.3 Web Server and WSGI layer

The (Web Server Gateway Interface) WSGI layer of flask is not made for production. Therefore, we have used Gunicorn as a wsgi layer. Gunicorn also offers some additional features such as load balancing and fault tolerance. In the Gunicorn layer we can mention how many nodes of our webapp we want. Gunicorn automatically handles the requests coming in and distributes it accordingly. For our web server we used nginx. Nginx redirects the request coming in to the wsgi layer. Nginx also has a load balancer and can also be used for scaling. We can easily scale this up or down.

4 Cloud Infrastructure

The architecture of the system developed employed the University of Melbourne Research Cloud infrastructure. In addition to this, we have also utilized several cloud managing tools such as Ansible and Docker. We used Ansible to automate creation of instance and deployment of application and used Docker for containerizing our application. We have four instances and three of them have 2vcpu 9Gb RAM and 40 GB storage each. Webapp's instance has 1vcpu and 4.5 GB RAM.

4.1 Unimelb Research Cloud Infrastructure

The University of Melbourne has its own cloud and computing infrastructure in place and that comprises about 20,000 virtual cores. Additionally, this cloud infrastructure is built using OpenStack. This allows the existing libraries to access the OpenStack Application Programming Interface (API). Throughout this project there has been on-going interaction with the University of Melbourne Research Cloud and as a result we were able to identify advantages and disadvantages of the cloud infrastructure in place. These advantages and disadvantages are:

Advantages

- The University of Melbourne Research Cloud provides a very practical graphical user interface and an Application Programming Interface that provides a programmatic provisioning for additional resources.

- The University of Melbourne Research Cloud provides Cross-cloud portability. The scripts and the source code that is being deployed are interoperable for any provider that can run on OpenStack and therefore it is putting a limit on the lock-in effects and this allows for any type of migration that may be needed in the future.

Disadvantages

- There are some limitations in terms of service that is provided. For example, in the market there are other cloud services like Microsoft Azure, Amazon's EC2, and many others that provide services which can actually help increase the rate at which the application is being developed.
- Another disadvantage of the Melbourne Research Cloud is that Proxy has to be set up at multiple locations such as at docker level and then again at server level.

4.2 Ansible Cloud

As a team, we wanted to make sure the computing resources are dynamically scaled within the cloud environment and therefore we decided to employ Ansible. Ansible is an information technology automation engine which automates everything in a cloud environment. With help of Ansible, we were also able to manage deployments be it simple or complex. For this assignment, an Ansible Playbook was created which allowed the deployment and set up for the entire system. The Ansible playbook tasks are shown below:

```
▼ AnsiblePlaybook
  > inventory
  ▼ roles
    > common
    > couch-cluster
    > couch-volumes
    > docker
    > docker-compose
    > git-clone
    > openstack-common
    > openstack-instance / tasks
    > openstack-security-group
    > openstack-volume
    > web-app-docker-compose
  ▼ variable
    ! couchdb_var.yaml
    ! var.yaml
    ! mrc.yaml
    ⓘ README.md
```

Roles:

1. Openstack-common -> This runs on localhost. It installs python-pip, openstack. These are needed for the ansible script to run.
2. Openstack-volumes -> It creates volumes to attach to CouchDB.
3. Openstack-security-group -> It creates security groups to open up required ports (ports opened-> 22(ssh) 5984(CouchDB access) 80(web app) 9100-9200(CouchDB cluster comm) 4369(CouchDB cluster comm)).
4. Openstack-instance -> This creates 3 harvester/CouchDB instances and 1 web app instance.
5. Common -> This is used for setting up unimelb proxy on each instance.
6. Couch-volumes -> It installs a few dependencies to make a file system. Also, this attaches 40 gb storage to each of the harvester instance.
7. Docker -> This installs docker and its dependencies on each instance, adds unimelb proxy to docker and restarts daemon, docker service.
8. Git-clone -> It clones our GitHub repository and creates a text file with the ip address of the master.
9. Docker-compose -> It spawns docker container for couch db and harvester on 3 instances.
10. Couch-cluster -> It is used for setting up a cluster of couch db.

Infrastructure as Code or IaC is the paradigm that Ansible follows. This paradigm essentially manages and deploys the infrastructure that is being computed. This is done through the machine-readable files. By following this approach, it ensures that deployments are repeatable and also predictable. At the same time, it also reduces the need for tedious configuration steps.

4.3 Docker Container

The system has employed Microservice based architecture which uses the Docker container to deploy the application. Each container has a service that runs in it and only particular functionalities are exposed to the application which is interacting with it. Every time a particular or new service is deployed, the Ansible task is executed and as a result it pushes any changes to the Git Repository that contains the source code for this project. There is a docker file corresponding to each service. Over time the Ansible task is used to build the Docker container depending upon the instructions given by the docker file. This type of approach has allowed the team to deploy services one after another as the features are being implemented. Lastly, the fact that each component which is needed for the entire system to be deployed is able to stay isolated in different docker containers assists in local development.

While using the Docker container technology, the team did run into several challenges, but we were able to overcome those obstacles. Some of the challenges we faced and how we were able to resolve those issues are listed below:

- Even after setting proxy for the instance, we were unable to download docker images. Therefore, to resolve this issue we had to set up a proxy for docker separately and had to restart daemon and docker services.
- After downloading the images required to build our images, we were unable to update the different container images and were unable to install dependencies inside the container. To resolve this conflict, for each container we had to set up a proxy separately. We did this by including environment variables in our docker file.
- For the Web application, we needed inter container communications so that nginx could access our apps wsgi layer and serve it to clients. We overcame this by adding both containers in the same docker compose file. It automatically created an internal network and gave nginx access to 5000 port of our app.

5 Data Gathering

This section describes the process that is involved in data gathering. Section 5.1 describes the setup that was used to harvest data from Twitter, while section 5.2 describes the various ways in which data from AURIN was mined. Section 5.3 introduces the GeoJSON standard that was leveraged to display information on maps.

5.1 Twitter Harvester

We have three instances which have harvesters running on them. At the start, these three instances read api keys from the database and set the flags to 1 for whichever api they were able to read. Once the api was read, the model we created for sentiment analysis was loaded along with the tokenizer. Each harvester queried CouchDB to get a region code. Once the region code was read, the flag was set to 1 and that meant that no other harvester could read the same region till it had been released. Now, the harvester used its twitter api key and with the help of tweepy library made a connection with twitter. If this was the first time the harvester was running, then it queried the max amount of data for that region that it could get. But if this was not the first time then, the last tweet id was queried from our region database and was passed as an additional parameter while querying twitter. This ensures that we do not get the same tweet again and again. Once it received the data, this data was passed through our nlp model and sentiment score was extracted. This was attached to our data and the entire thing was pushed to "tweet" db. Also, along with this the last tweet id was pushed to our region database. Now the lock from the region was released. The above process is run over and over again with a certain amount of sleep time between each iteration.

There are certain limitations imposed by twitter on the free api key. These limit the amount of data that can be pulled from twitter. Currently twitter gives only 1% of the data via their free keys and a total number of tweets per api call is also limited. We overcame this by using multiple api keys over different instances. Also, when using tweepy we set `wait_on_rate_limit` as `True` so if any of the api exceed their daily limit our function sleeps for some time and retries.

5.2 Aurin Data

Alongside the Twitter data, we used two datasets from Aurin which enabled us to perform comparisons required for in-depth analysis. The description and role of each dataset are outlined below:

1. Population Counts by city - used in Scenario 1, Graph 1
 - a. Dataset title: ABS - Data by Region - Population & People (LGA) 2011-2019
 - b. Provider Organisation: AU_Govt_ABS
2. Unemployment Rate by city - used in Scenario 3, Graphs 1 and 3
 - a. Dataset title: DESE - Labour Market - Summary Data (SA4) December 2020
 - b. Provider organization: AU_Govt_DESE

5.3 GeoJSON

GeoJSON is a Geographic Information System (GIS) standard, based on JSON, used for representing geographical features in the form of vectors in web maps. It supports multiple geometry types such as points, polygons along with additional properties in a single 'Feature' object. A set of 'Feature' objects are used to depict the whole geometry of the geographical entity. We are using the GeoJSON files to represent the coordinates of the tweets which we harvested from twitter and showing the aggregate number of tweets in a particular location.

6 Data Storage

The data storage for the entire system was created by employing CouchDB. In total there were three nodes (server-01, server-02 and server-03) that were chosen for the formation of this CouchDB database in a clustered form. In section 6.1, it is discussed in more detail. There were numerous components of the system that heavily depended on the CouchDB storage. In section 6.2, it is discussed in further detail.

6.1 CouchDB Cluster Formation

CouchDB is a modern NoSQL database that can be configured for operation in a clustered environment. In a clustered environment it would have higher capacity and availability than it would with single-node configuration. This is exactly how the formation is for this assignment. We have a clustered environment using three nodes. Regarding this assignment, our team used the images provided by docker hub. The steps for CouchDB deployment are:

1. couch-volumes -> Installs a few dependencies to make a file system. Attaches 40 gb storage to each of the harvester instance.
2. docker -> Installs docker and its dependencies on each instance, adds unimelb proxy to docker and restarts daemon, docker service.
3. git-clone -> Clones our GitHub repository, creates a txt file with ip address of the master (used by harvester to access CouchDB)
4. docker-compose -> Spawns docker container for couch db and harvester on 3 instances.
5. couch-cluster -> Sets up cluster of couch db.
6. web-app-docker -> Runs docker compose up on a web-app instance and spawns nginx container (web server) and python container.

When employing the CouchDB in a clustered formation, there were some challenges which we faced. However, with persistence we were able to overcome these challenges. Some of the challenges faced by the team and how we overcame them when dealing with CouchDB are listed below:

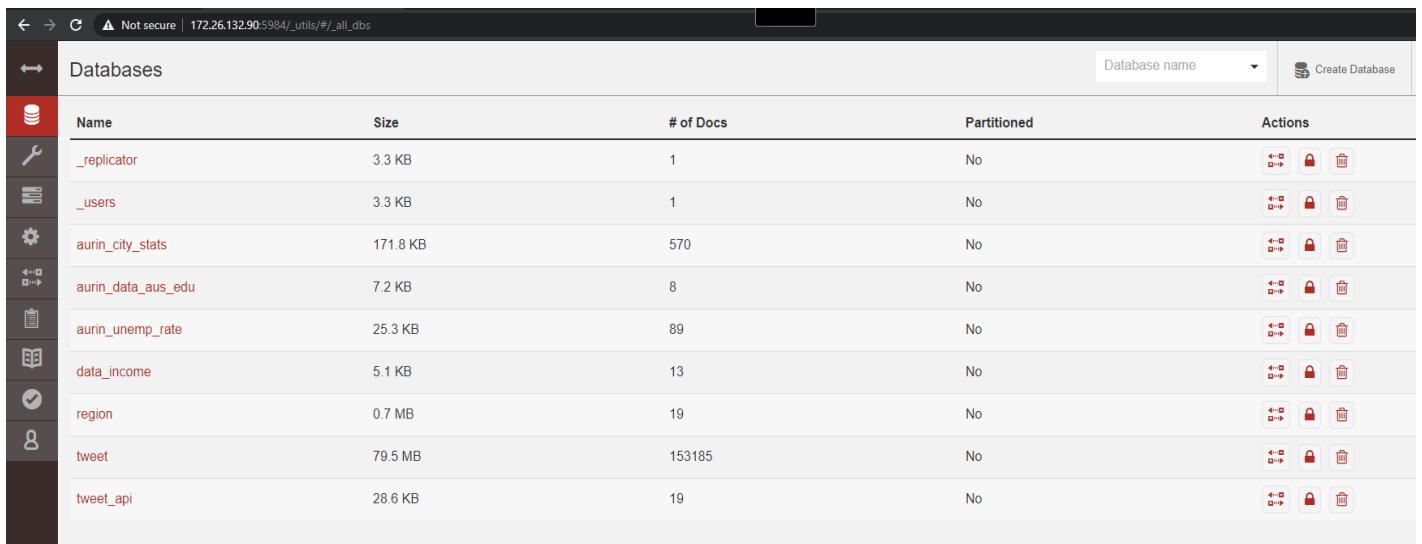
- Problem accessing CouchDB instances from other instances. We were able to overcome this by using erlflag and giving each instance its name "couchdb@its_ip_addr"
- Setting these names dynamically in each container over different instances. We solved this issue by creating a docker-compose file using ansible's docker_container function. Hence, we were able to attach a variable which represented the instance the script was running on.

- If we tried to run the same ansible script after the deployment and cluster formation was done, we were getting an error code relating to the cluster already created. We overcame this by adding this error code to acceptable code.

6.2 Database

The CouchDB database plays a major role in numerous components of the entire system. Some of the key responsibilities it has is:

1. Storing the raw and processed twitter data(tweets)
2. Storing the processed data from AURIN for comparison in the three different scenarios for this assignment
3. Storing the configuration parameters (API Credentials, PS coordinates, etc.) which are used for Twitter Harvester.



| Name | Size | # of Docs | Partitioned | Actions |
|-------------------------------------|----------|-----------|-------------|---------|
| _replicator | 3.3 KB | 1 | No | |
| _users | 3.3 KB | 1 | No | |
| aurin_city_stats | 171.8 KB | 570 | No | |
| aurin_data_au_s_edu | 7.2 KB | 8 | No | |
| aurin_unemp_rate | 25.3 KB | 89 | No | |
| data_income | 5.1 KB | 13 | No | |
| region | 0.7 MB | 19 | No | |
| tweet | 79.5 MB | 153185 | No | |
| tweet_api | 28.6 KB | 19 | No | |

7 Fault Tolerance and Error Handling

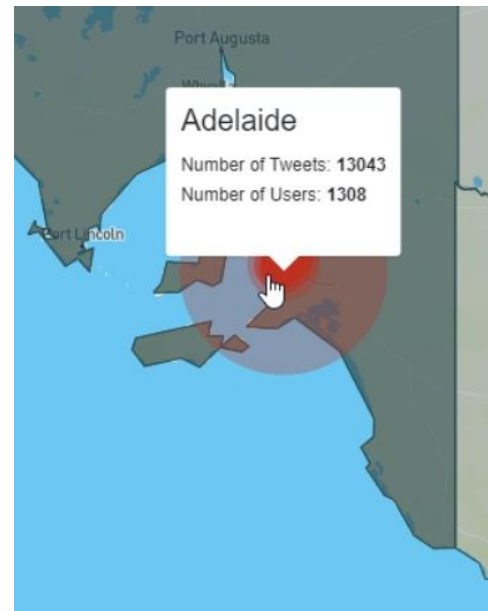
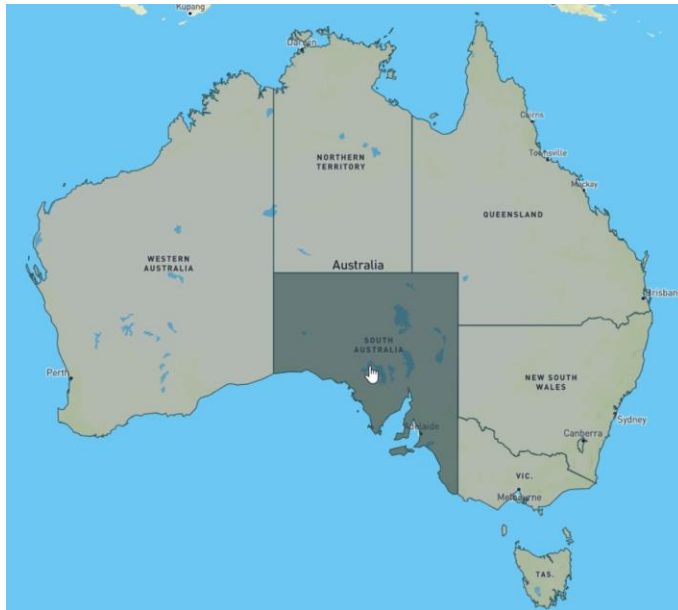
The system that we created is very robust. We handled errors in the following way:

- We have 3 nodes for CouchDB in a cluster with replication enabled. If any of the couch db instances fails, the other instances take over and the app keeps running. Once the failed CouchDB instance comes back online it rejoins the cluster.
- We have 3 harvesters running. At times due to http proxy settings, the harvester couldn't establish a connection to twitter api. We implemented exception handling around these sections. Whenever an exception occurs, we release lock from the region and try the whole function after some sleep time.
- For our webapp, we have multiple nodes running. If we have a high traffic, our load balancer distributes the traffic evenly. In case one of the nodes fails, other nodes take over and the user does not get to know about the failure.

8 Data Visualization

8.1 Maps

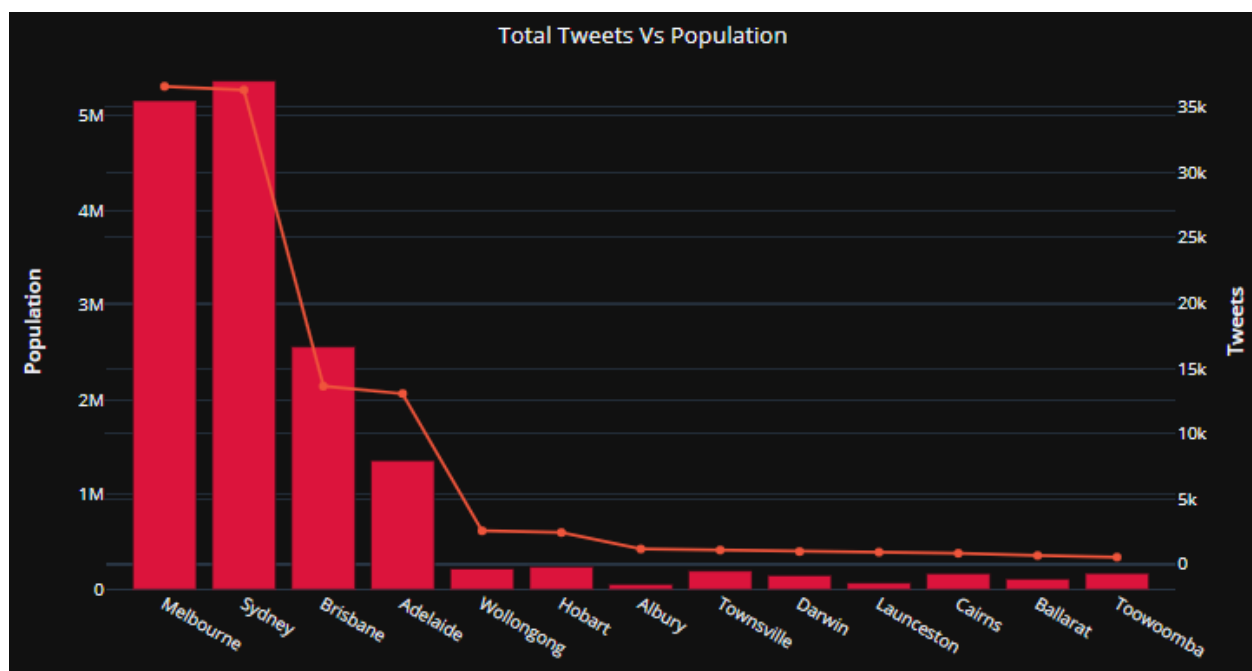
The map is used to plot the locations captured from the tweets, with the radius of the circles in proportion to the number of tweets gathered at a point.



8.2 Graphs

Scenario 1:

Graph 1:



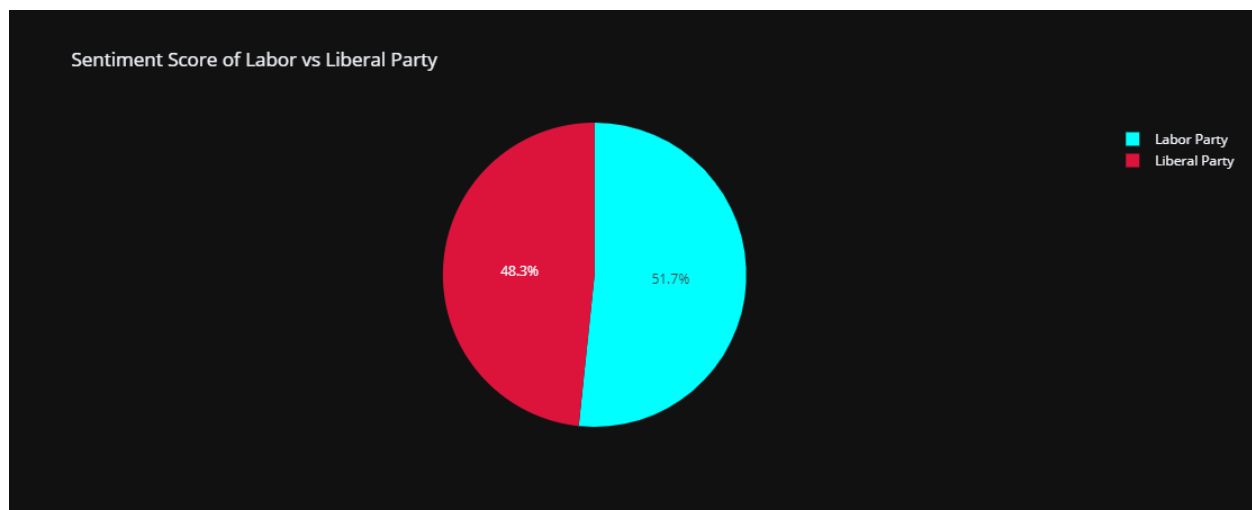
Observation: As expected, cities with higher population counts generally have higher total number of tweets. This suggests a strong correlation between population and total number of tweets.

Graph 2:

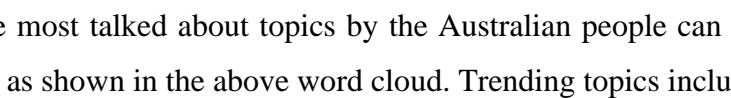


Observation: Since the average income levels across the selected cities are rather similar, we cannot observe any correlation between average income and sentiment score of tweets. This analysis can perhaps be improved by collecting more data across a larger number of cities.

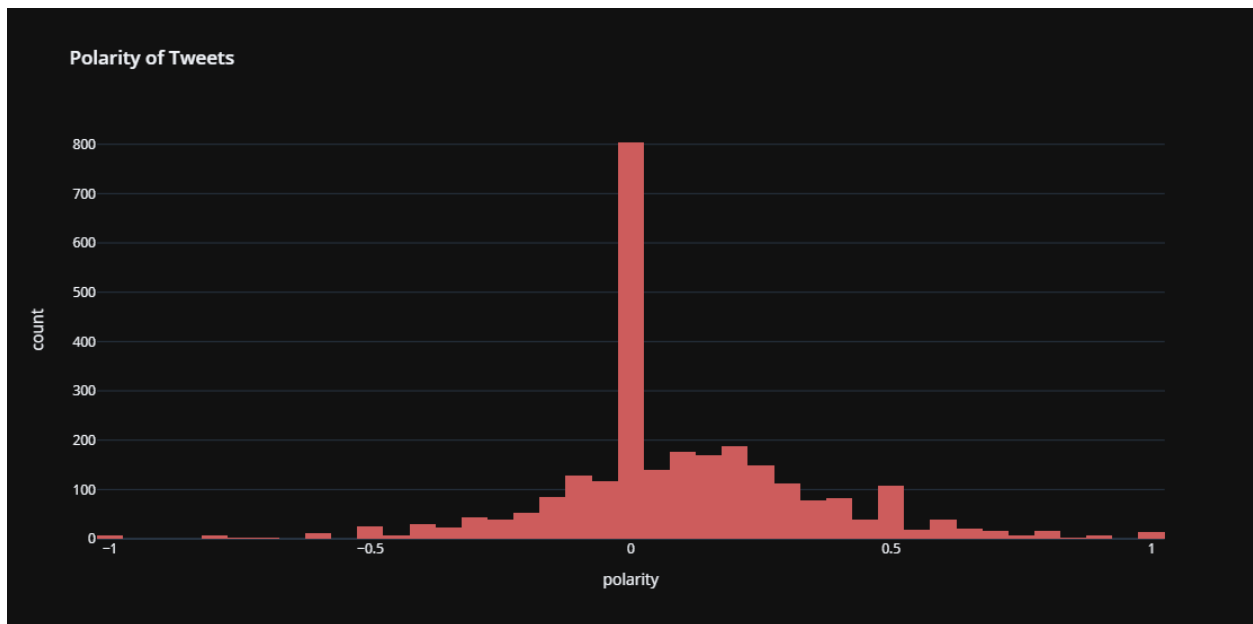
Graph 3:



Graph 4:

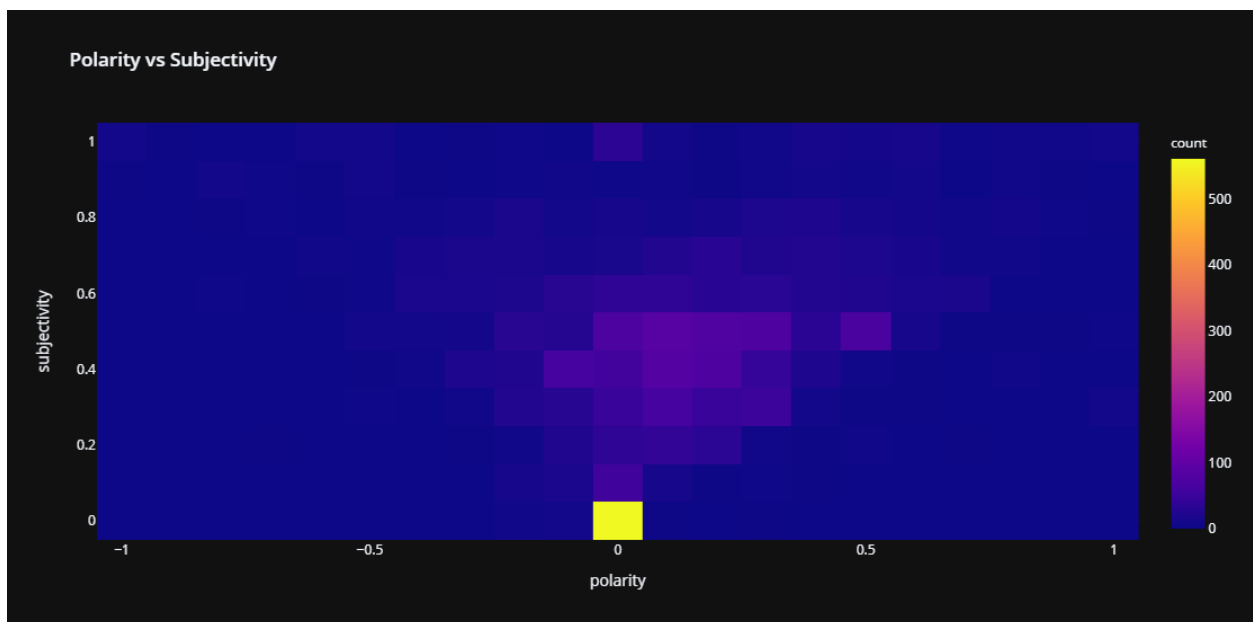


Scenario 2:



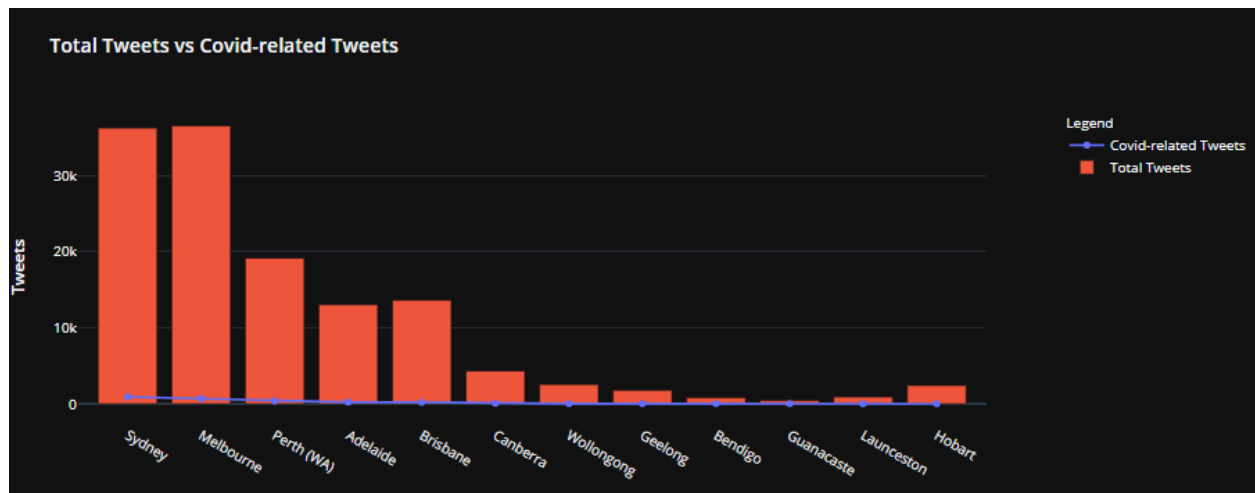
Observation: Tweets relating to Covid-19 mostly have a neutral to positive polarity score indicating that the Australian people have a positive outlook despite the ongoing pandemic.

Graph 2:



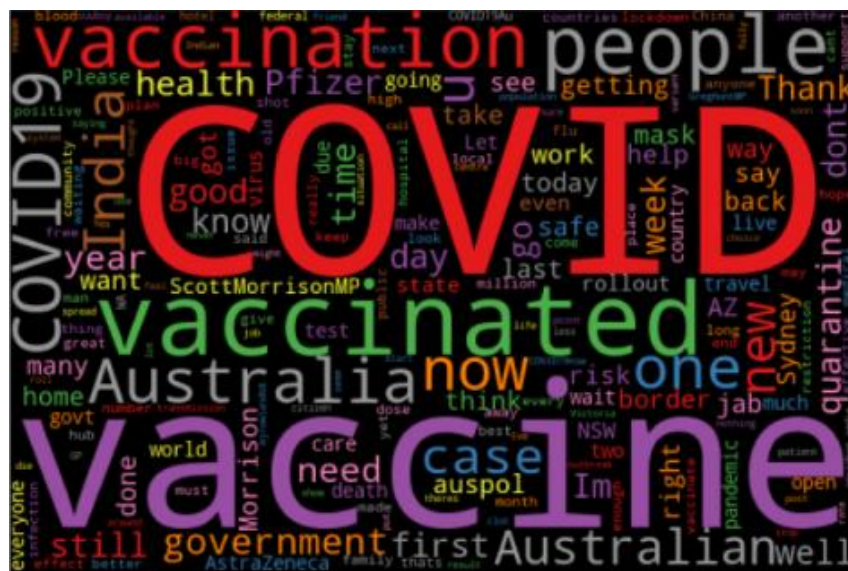
Observation: Covid-19 related tweets mostly have a neutral polarity and are generally fact based. Some of the tweets are derived from emotions and are on the positive side.

Graph 3:



Observation: As Australian cities have very low numbers of covid cases, people are tweeting very less about covid.

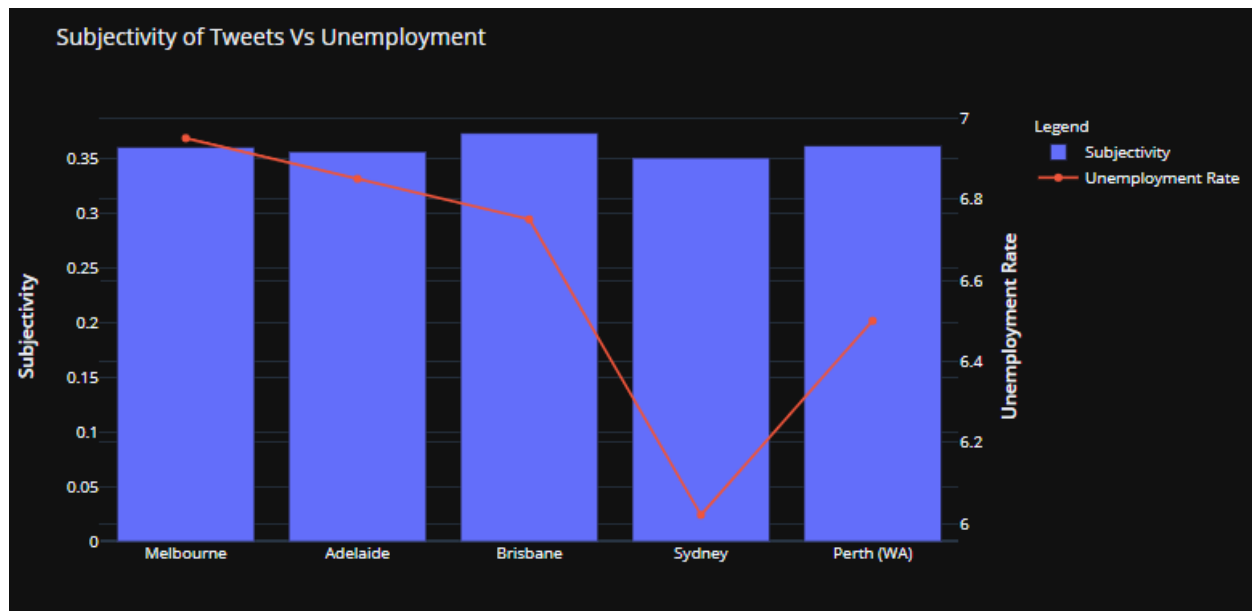
Graph 4:



Observation: We started data collection 4 weeks ago. During the initial stage of our analysis, we were getting many tweets about Covid and its impact. Now that vaccines are out, the topic of contention has shifted to vaccines.

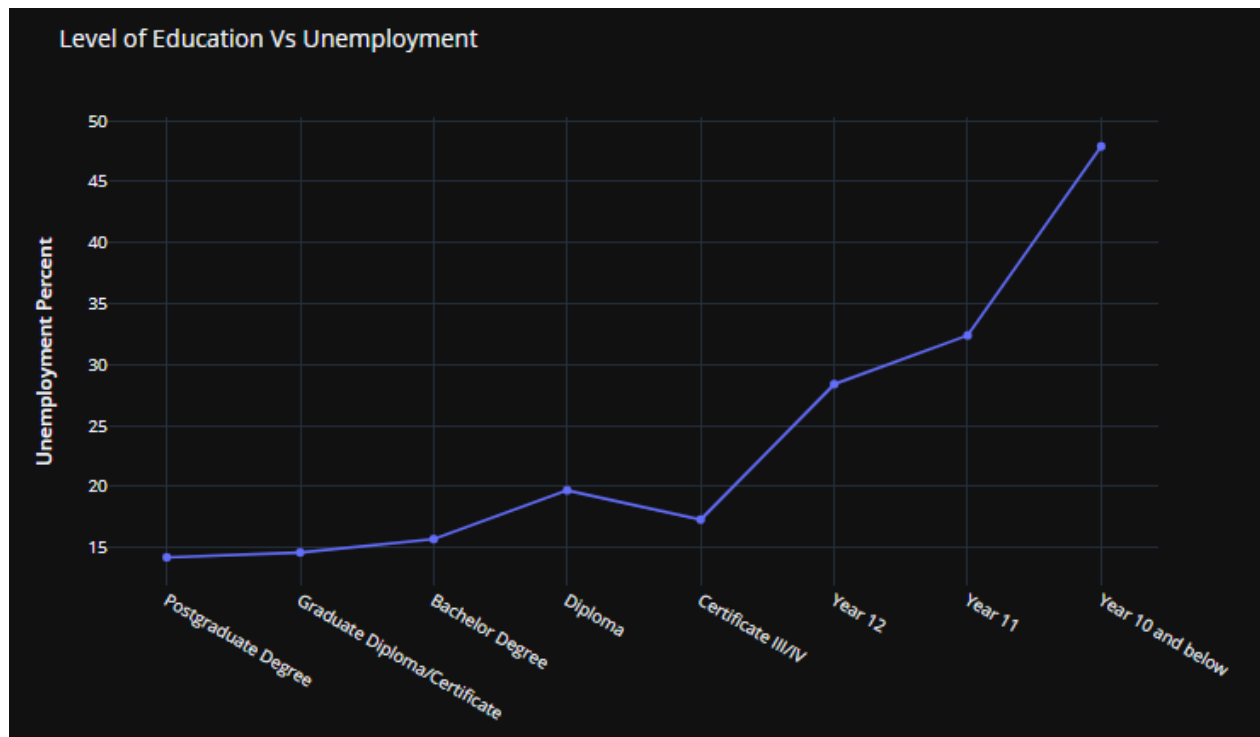
Scenario 3:

Graph 1:



Observation: Since all the cities have similar subjectivity scores and unemployment rates, no correlation can be observed.

Graph 4:



Observation: As expected, as the highest level of education attained increases, the employment rate increases. Hence, the two variables in consideration have a strong correlation.

9 User Guide

Each and every section of the code is in containers, so the deployment becomes easy. The ansible scripts starts by installing prerequisites on the local system and then moves to resource creation on cloud. Once this is finished, ansible deploys CouchDB in a cluster form on 3 instances along with harvesters. Finally, it moves to the last instance and deploys the web app. Web app is in the form of 2 containers - nginx and flask app.

Setup:

1. Python-3 needs to be installed locally.
2. You need to have Ansible installed locally.
3. You will have to get your own Openstack RC file along with a Openstack API password

Steps for Deployment:

1. Create your own key pair and upload it to MRC.
2. In the inventory.ini file, replace the value of “ansible_ssh_private_key_file” to the path of the key on your system.
3. Also in var.yaml (found in the variable directory) replace the value of “instance_key_name” to the name that you gave your key.
4. Clone GitHub directory. We need the AnsiblePlaybook directory locally.
 - a. Git clone “<https://github.com/arnavgarg123/COMP90024-ClusterAndCloudComputing-Assignment2.git>”
5. Make sure you are connected to VPN if you are accessing MRC from outside the university.
6. Now go into Ansible directory and run:
 - a. `./unimelb-comp90024-2021-grp-48-openrc.sh; ansible-playbook -i inventory/inventory.ini mrc.yaml`
 - b. Enter your password and wait for it to execute. (Takes around 15 min)

Once the system has been deployed you can access the webapp using the url (IP) of the webapp instance and navigation and actions such as hover and redirects have been provided in the front end.

10 Team Collaboration

At the beginning of the assignment, we had brainstorming sessions to discuss the approach and implementation of the assignment. We made use of Zoom to have virtual meetings as all of the team members were located in different places around the world. The backend programming language used for this assignment was Python as all of our group members were familiar with it. For the front end, we used the Bootstrap 4 framework, to help us design the HTML pages. We divided the project into the following sections:

1. CouchDB and Twitter Harvester implementation: Arnav Garg and Piyush Bhandula
2. NLP Model for Sentiment Analysis of Tweets: Arnav Garg
3. Ansible Automation: Arnav Garg and Jay Dave
4. Flask Web App Development: Vishnu Priya G and Jay Dave
5. Design and Implementation of Graphs: Gurkirat Singh Chohan and Vishnu Priya G
6. HTML and Bootstrap Layout: Piyush Bhandula and Gurkirat Singh Chohan

The team members were assigned tasks and we held regular meetings to go over the progress and discuss any issues and close them.

11 Conclusion

In this assignment, our team made use of Twitter and used the geolocation of the published tweets to demonstrate any type of correlation between different factors in different types of scenarios. We used Docker container technology for the completion of this assignment, and we did face challenges during the development of the system. However, over time we were able to overcome those obstacles. Additionally, we used CouchDB as our database. In total we had three scenarios and all of these tweets were from individuals living in Australia. The results we obtained from performing different types of analysis and other computing methods were rather interesting. From the analysis of the above scenarios, we observe that a lot of meaningful data can be extracted from twitter or social media in general. With the help of external data sources, we saw how twitter data correlates to different scenarios. This showed us that, for many use cases, we can use smart analytics on social media data to understand different aspects of life in different cities.

12 Future Work

- Data from other social media sites can be gathered and used along with current data to increase precision of our analysis.
- Data from paid twitter accounts can be taken to understand the life of people in more detail.
- Data from paid twitter accounts can also be used to track them and create a blueprint of their daily movement pattern.
- More nodes can be added to enable scaling of the system and also to collect data from more cities. This system can be applied to analyze any country.

13 References

- [1] https://docs.ansible.com/ansible/latest/user_guide/playbooks.html
- [2] <https://docs.couchdb.org/en/stable/>
- [3] <https://docs.tweepy.org/en/latest/index.html>
- [4] <https://docs.python.org/3/>
- [5] <https://docs.docker.com/get-started/overview/>
- [6] <https://docs.gunicorn.org/en/stable/configure.html>
- [7] <https://nginx.org/en/docs/>
- [8] <https://plotly.com/python/>
- [9] <https://datatracker.ietf.org/doc/html/rfc7946>
- [10] <https://getbootstrap.com/docs/4.6/getting-started/introduction/>
- [11] <https://docs.mapbox.com/>