THE UNIVERSITY OF MELBOURNE

COMP90024 Cluster and Cloud Computing S1 2019

Project 2 Report

Harvesting Tweets using UniMelb Research Cloud to analyse exercising trends in Urban Australia

Team 46

TEAM DETAILS

Team Members:

Student	**ID**
Advait Mangesh Deshpande	1005024
Ansh Juneja	1027339
Saransh Srivastava	1031073
Siyu Biyan	984002
Waqar Ul Islam	1065823

Cities Analysed:

Adelaide	Brisbane	Canberra
Geelong	Gold Coast	Melbourne
Newcastle	Perth	Sydney
Townsville		

Table of Contents

1. ABSTRACT

Social media and specifically Twitter, has been the centre of human sentiment analysis since the inception of micro-blogging. A lot of research have undertaken in previous years around core human emotions but far less work in the more darker side of emotions. In this analysis we found sentiments of people who micro-blog about physical fitness and exercise. We selected top 13 cities within Australia who tweet about their exercise activities and analysed them against different datasets available through AURIN and freely available Australian government statistics.

2. INTRODUCTION

The purpose of this project is analysing the information about the Seven Deadly Sins through the social media - Twitter, such as the emotion (happy or sad) of people reflected by their tweets, and show the result with the proportion in the map of Australia.

Sloth is one of the Seven Deadly Sins. It is easy for people to think that more obese people may be slothful than others. Because of this, sloth is associated with exercise, which may be a more positive aspect than sloth to proceed this project. Therefore, “exercise” is chosen as the main topic, and the searching topic of this system is expanded to a set of synonyms of “exercise”, such as “workout”, “cycling”, “yoga” and “fitness”.

Besides, there are many tools and functionalities used to implement this system. For example, Ansible, CouchDB and Map-Reduce are applied in different layers of this system.

3. SYSTEM DESIGN AND ARCHITECTURE

The system is designed based on the basics of cloud computing architecture shown in Figure 1.

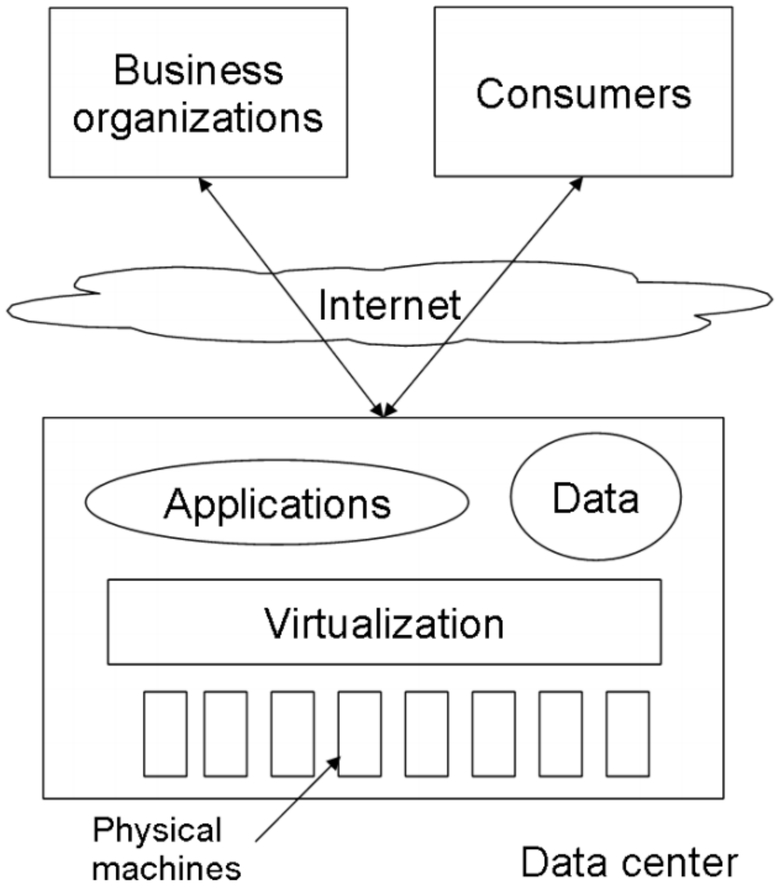


Figure 3.1: Basic Cloud Computing Architecture

According to Figure 1 and the purpose of this project, the architecture of this system is shown in Figure 2. There are 4 layers of the whole system, which will be introduced in detail, respectively.

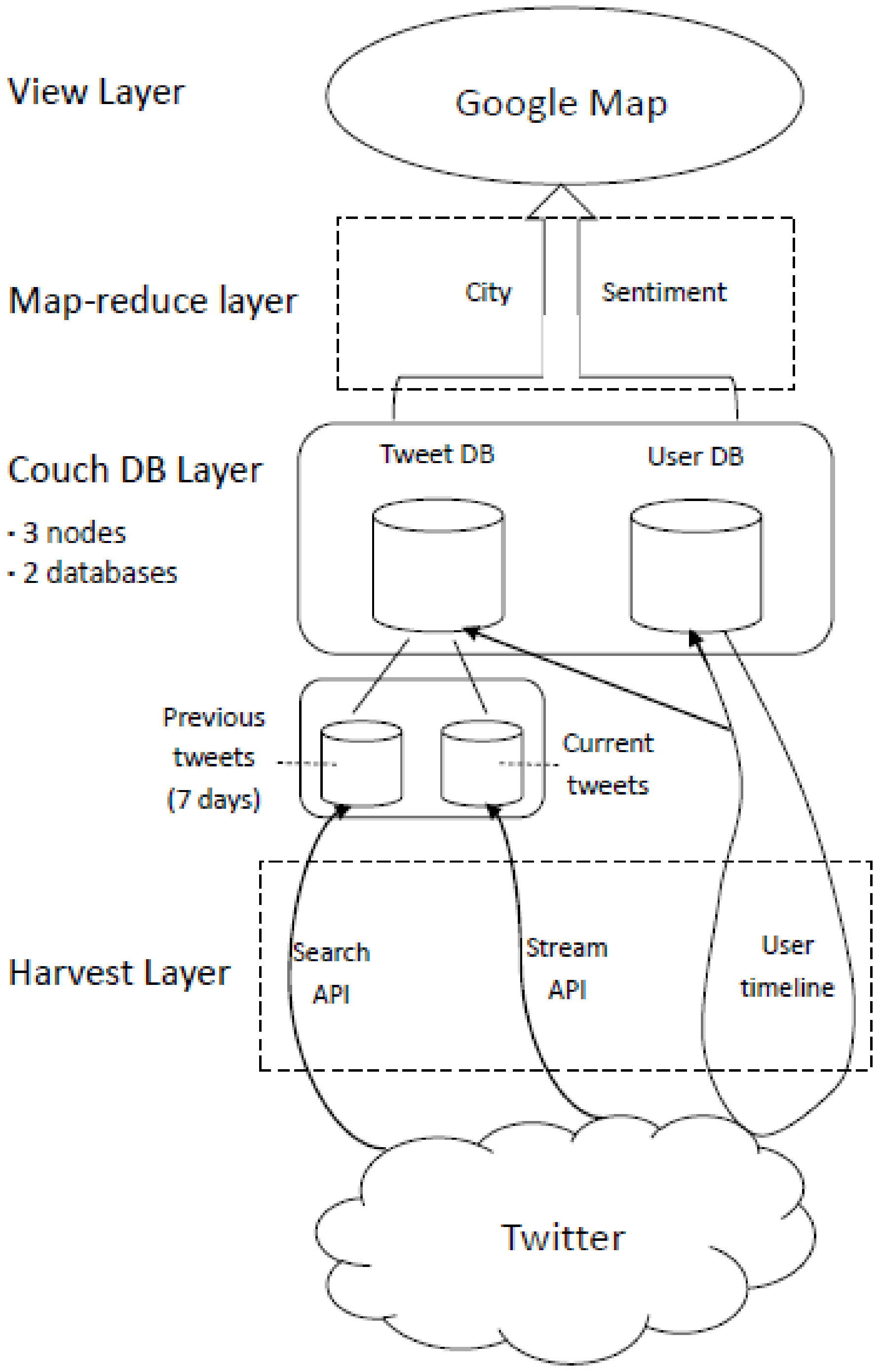


Figure 3.2: System Architecture

3.1 Implementation stages

To achieve this system, there are four main stages which are listed as follows.

3.1.1 Setup CouchDB

The CouchDB database is set up to store different kinds of data in different instances, separately.

3.1.2 Harvest tweets

There are a large number of tweets of Twitter, while only a small part of them is useful in this system. Because of this, harvesting the data followed by analysing is helpful to reduce the size of the database in CouchDB. Additionally, collected tweets are stored with different status to show the positive or negative emotion about exercise of authors.

To expand out search query words, the nltk wordnet package is used to obtain multiple synsets of the primary topic.

3.1.3 Display the statistic results with the map

3.1.4 Combine the data in CouchDB with AURIN

3.2 CouchDB

All of the data is stored in CouchDB. There are 4 nodes storing different kinds of data.

(1) Node 1, 3 and 4 are used for harvesting.

- Node 1 uses tweepy cursor to harvest and apscheduler to fetch every 10 minutes.
- Node 2 also hosts the website.
- Node 4 fetches all tweets of a particular user from their timeline. These users include all the tweet authors captured above and their user mentions.

(2) Node 3 uses twitter streaming fetching most recent tweets for approximate 8 minutes followed by 2 minutes testing. After this, CouchDB disconnects with Twitter, and then starts the process again.

There are also 2 databases showed in the Figure 2. Except the tweets (stored in the Tweet DB), all users, as well as user mentions which may or may not be in Australia, are also stored in another database (User DB).

3.3 Harvest data

Only the relative tweets, which contain at least one of the searching topics from different cities in Australia on the UniMelb Research Cloud, are harvested by this system. There are two main methods are listed as follows.

- Using the search API interface to search the words as same as the search topic in the text, followed by harvesting the relative tweets into CouchDB.
- Using the stream API interface to compare the coordinates of each tweet with the location of different cities in Australia.

We filter out any tweet which is outside Australia or does not related to our topic, which results in that our database contains only about 40K genuine tweets, with useful information. The total number of relative tweets can also be shown in each city (Adelaide, Brisbane, Gold Coast, Melbourne, Perth and Sydney), separately.

3.4 UI

The map combined with the database shows the statistic result of the harvested tweets in Australia. Using the drop-down list, each city can be manually selected with showing the statistic graph as well as the percentage of positive and the percentage of negative tweets below the map.

4. PROS AND CONS OF THE UNIMELB RESEARCH CLOUD

The UniMelb Research Cloud is a private cloud. It has many essential characteristics, and its own pros and cons.

4.1 Characteristics

- As a cloud, it is on-demand and easy for customers to use, and has the resource pooling to store a huge amount of data.
- The characteristics of scalability and rapid elasticity allow us using many services (e.g. 100 services) to collect and analyze data, instead of manually dividing all of the

data into different servers.

- It provides the measured service. Customers can define the input, output and CPU to satisfy their own needs.

4.2 Pros

- In terms of the security, it is only for people who are from the University of Melbourne. People from other places cannot login, and the university firewall can stop any other unallowed connection.
- In terms of the control and consolidation, customers can get much more control about their own resources.
- In terms of more trust, customers gain much more trust in the thing that cooperating with locally. For example, research can be done in the university, managing in the way which we know.

4.3 Cons

- In terms of the relevance to core business, it not easy to move a huge amount of data from a private cloud to a public cloud, if the database is need to be shared with other companies or organisations.
- In terms of the staff or management overheads, as same as other private clouds, the unimelb cloud costs much money, such as keeping the cloud running and paying for the staffs who manage the cloud.
- In terms of the hardware obsolescence, whether these equipments should be thrown away, or be carried to another place waiting for new one coming in, because these machines can still be used.
- In terms of over or under utilization challenges, for example, there are not enough people to use it. Sometimes, too much money are spent on it, but the number of users who use it is not as much as expected. It is very difficult and complicated to estimate how much equipment and resource are needed.

5. SYSTEM FUNCTIONALITIES

5.1 Using Python to harvest tweets

5.1.1 Parameters

API_KEY	Twitter user unique key
API_SECRET_KEY	Twitter user unique secret key
ACCESS_TOKEN	Twitter user unique token access key
ACCESS_TOKEN_SECRET	Twitter user unique access token secret key
SEARCH_TOPIC	Search topic from the tweets extracted
POS_SENTIMENT	Positive sentiment value = 1
NEG_SENTIMENT	Negative sentiment value = -1
SERVER_1	Node 1 IP address
SERVER_2	Node 2 IP address
SERVER_3	Node 3 IP address
SERVER_4	Node 4 IP address
UserDb_Design	User database design document
UserDb_View	User database design view identifier
db	Database python object
dbUser	User database python object
AUS_LAT_MIN	Australia minimum latitude (as per AURIN)
AUS_LON_MIN	Australia minimum longitude (as per AURIN)
AUS_LAT_MAX	Australia maximum latitude (as per AURIN)
AUS_LON_MAX	Australia maximum longitude (as per AURIN)

5.1.2 Methods

getServer	Get the IP address of the local server
storeInUserDb	Stores user information in User database
storeInDb	Stores indivizual twitter information in twitter database. The uniqueness of tweet is ensured via tweet id as key. The validity of location of the tweet is compared before storing in the database
isValidLoc	Checks if the latitude and longitude are within Australia
getSynonyms	Augments the search query using synsets from nltk wordnet
getSentiment	Augments tweet with sentiment polarity. 0 signifies neutral, 1 signifies positive and -1 negative sentiment
removeSChar	Removes special character and link from the tweet text
isEligible	Checks eligibility of the tweet by comparing the text of the tweet with our augmented search topic
getPlace	Returns the place of origin of the tweet. This is obtained using place key from raw tweet. If unavailable the author location is used
putJsonInDb	Json data object converted to dictionary. This method checks the eligibility of the text before sending it to save in database
harvestTwBySearch	Main harvestor method running on Node 1 consuming tweets from twitter using tweepy Cursor API. It looks for maximum 2000 tweets in one request
harvestTwByStream	Main harvestor method running on Node 3 consuming tweets from twitter using tweepy stream API in async mode. The system sleeps for 500 seconds before disconnecting the stream
harvestTwByUser	Main harvestor method running on Node 4 consuming tweets from twitter using tweepy Cursor API. The tweet are all the previous tweets w.r.t. to a particular user.
harvestor	Main harvestor method calling harvestor based on which node it is deployed on

5.2 Information stored in Databases

5.2.1 Twitter Database

_id	Id assigned by CouchDb
_rev	Unique revision number
type	Type of data. Presently suppoorts two types: twitter & user
value	Value obtained from the particular row
username	Twitter username
longitude	Longitude of the place from where tweet was made
sentiment	Sentiment of the tweet text
text	Original text of the tweet
created_at	Human readable format time of the tweet created
userId	User Id of the user who has tweeted
coordinates	Raw coordinates, including latitude and longitude of the tweet
place	Name of the city or region augmented
geoRaw	Raw geocode from twitter raw data
latitude	Latitude of the place from where tweet originated
placeRaw	Raw place from twitter
full_name	Full name of the user who originated the tweet
url	URL of the tweet
country	Country of origin of the tweet
place_type	Type of place - city, POI etc
bounding_box	Raw coordinate box of the tweet
country_code	Country code of the origin of the tweet
id	Unique ID of the tweet. This is used to make each tweet in the database unique
Name	Place name

5.2.2 User Database

_id	Unique ID of the document stored in User database
_rev	Revision number of the user information
Username	Username of the person
Status	Status = 0, 1
Type	Type of data - User data

6. TOOLS AND PROCESSES FOR VM DEPLOYMENT AND COUCHDB INSTALLATION

As discussed in the previous sections, we have created our cluster on the UniMelb cloud research platform, NeCTAR.

We used openstack (using **openstacksdk**) to communicate with the NeCTAR system and used **Ansible scripts** to launch instances on the cloud.

To communicate with the NeCTAR system, we created a key pair using the inbuilt functionality (under the *Compute* à *Key Pairs* tab) and updated that in the local machine. On the cloud system, we first created an instance manually using the “Launch Instance” functionality of NeCTAR. This allowed the team to explore and play around with the system while the ansible script was being written. The ansible script itself was written on WSL with Ubuntu.

The ansible script has 3 major parts.

6.1 Roles

6.1.1 Volume

Creates 4 volumes, 1 for each Virtual Machine (VM). They are only accessible in the “melbourne-qh2-uom” availability zone (i.e., the user must be on the UniMelb network). All of them have a size of 50 GB each.

- We decided to give all of the VMs volumes of equal sizes as we were unsure of how much data each of them will need.
- In hindsight, we could have used a lot less volume as well.

6.1.2 Security Group

Creates a security group that is to be used on all VMs. The security group allows the VMs to egress to all protocols, on all ranges and on all IPs. Similarly, it allows ingress from all 3 IP protocols (TCP, ICMP and UDP) on all port ranges. On all 3 of these protocols, it allows ingress from any IP address and from any VM that has the same security group as itself.

- We allowed ingress from all VMs that have the same security group to allow the VMs to interact with each other.
- We allowed ingress from all IP addresses to allow the VMs to get data from Aurin and twitter.

We can also restrict the ingress to the ports being used (22 (SSH), 5984 and 5986 (CouchDB) and 5000 on VM#2 (for the UI))

6.1.3 Instances

The script was used to create 4 instances of the VMs (named instance1/2/3 and 4). Each of them has “Ubuntu 18.04 LTS (bionic) amd64 [v14]” and have 2 cores, and 8GB RAM. As with the volumes, they are only available in the “melbourne-qh2-uom” availability zone. They use the key that was created on the NeCTAR system and have the security group mentioned above.

6.1.4 Setting up the instances

This role installs all the required packages using apt and pip, adds the key pairs (so that the instances can communicate with each other) and updates the /etc/environment variable.

6.1.5 Setting up and configuring CouchDB

This role installs and configures the CouchDB. To install CouchDB, it gets the repository from git and installs it using apt. Then it configures the CouchDB using the templates and .ini files, as explained in the following parts.

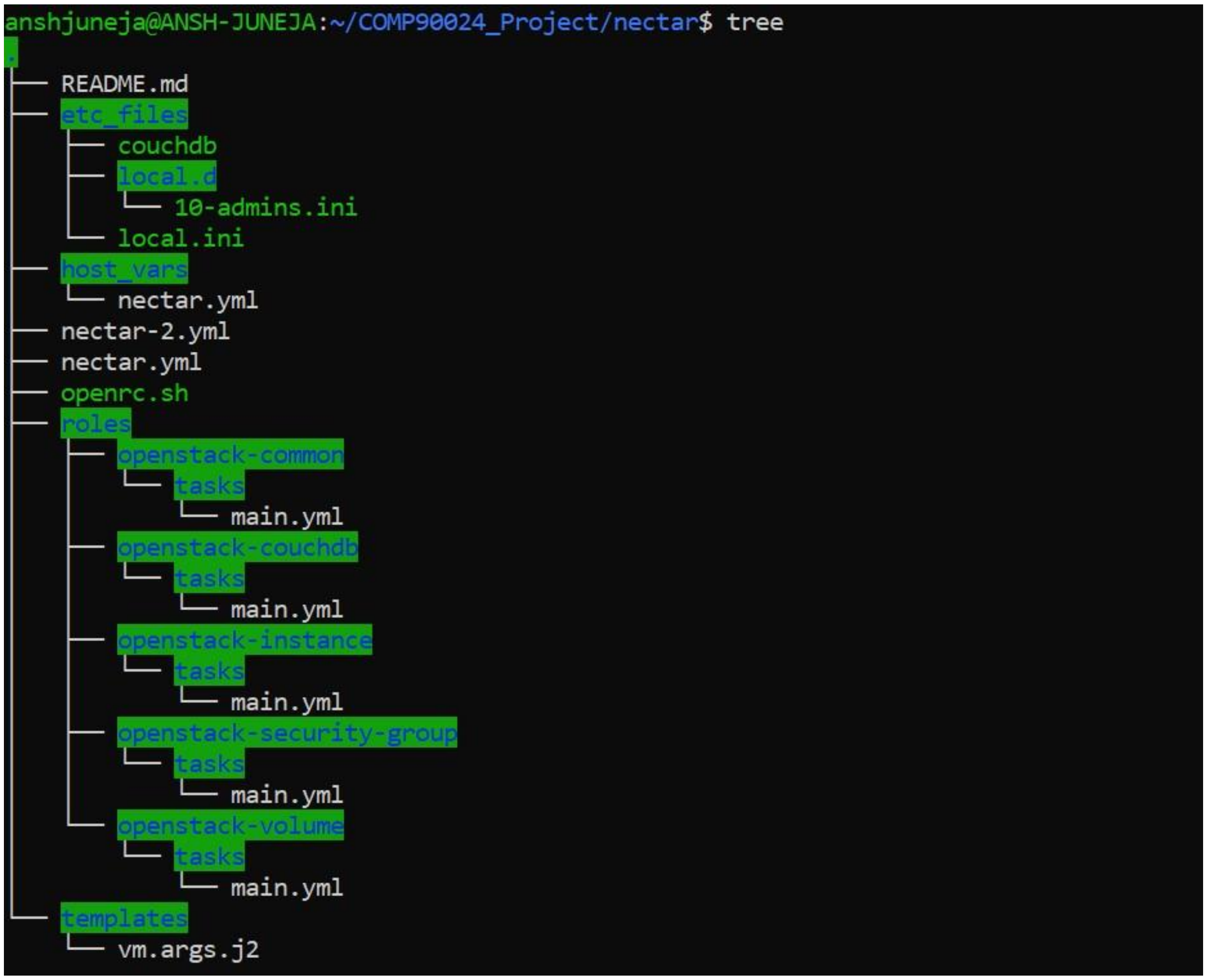
6.2 Variables

The ansible script uses global (as a .yaml file in the host_vars folder) and playbook level variables (by defining them under 'vars'). It also uses a template for the vm.args file for CouchDB. The template is defined in Jinja2

6.3 .ini files

The script also uses .ini files (default.ini and local.d/10-admins.ini) that were taken from a test instance of the CouchDB instance.

As said above, we installed a test instance of CouchDB on the 4 instances using Fauxton and then used the vm.args and the .ini from this test instance. CouchDB has been installed in the cluster mode and it has instance 1, 3 and 4 as it's nodes. By doing this, we were able to skip over a lot of code in the ansible script to configure CouchDB. Also, this way, we didn't have to encrypt the password while storing it in the .ini files (as it was already encrypted).



```
anshjuneja@ANSH-JUNEJA:~/COMP90024_Project/nectar$ tree
.
├── README.md
├── etc_files
│   ├── couchdb
│   ├── local.d
│   │   └── 10-admins.ini
│   └── local.ini
├── host_vars
│   └── nectar.yml
├── nectar-2.yml
├── nectar.yml
├── openrc.sh
├── roles
│   ├── openstack-common
│   │   └── tasks
│   │       └── main.yml
│   ├── openstack-couchdb
│   │   └── tasks
│   │       └── main.yml
│   ├── openstack-instance
│   │   └── tasks
│   │       └── main.yml
│   ├── openstack-security-group
│   │   └── tasks
│   │       └── main.yml
│   └── openstack-volume
│       └── tasks
│           └── main.yml
└── templates
    └── vm.args.j2
```

Figure6.1: A screenshot of the directory structure that were used in the ansible process

7. UI

We have used React as a front end technology to develop the web application. React JS was developed by Facebook in 2011 and open sourced in 2013. After that React Native(Mobile Development) was developed in the 2013 and outsourced in 2015.

React is component based front end framework which is very similar to the Angular framework. It can be used only to develop the front end and can be integrated with various backend technologies like Node.js or Java or Python etc.

In our project, we have used React version "15.6.1". We have used various React libraries to develop the application. For example, JQuery to make API calls to the CouchDB, Chart.js to integrate the Charts on the UI, react-google-maps package to include Google Maps.

The whole UI is based on the conditional rendering of the components and in accordance with the industry standards.

We are using Ajax to make an asynchronous API call to the CouchDB to access the views which are used to display the markers on the map.

The default zoom of the map is set to Australia and once you start zooming in you can see the markers for the cities that are being overlapped. For example, you cannot see the marker at Geelong or Wollongong due to the default zoom but they are visible when you zoom in.

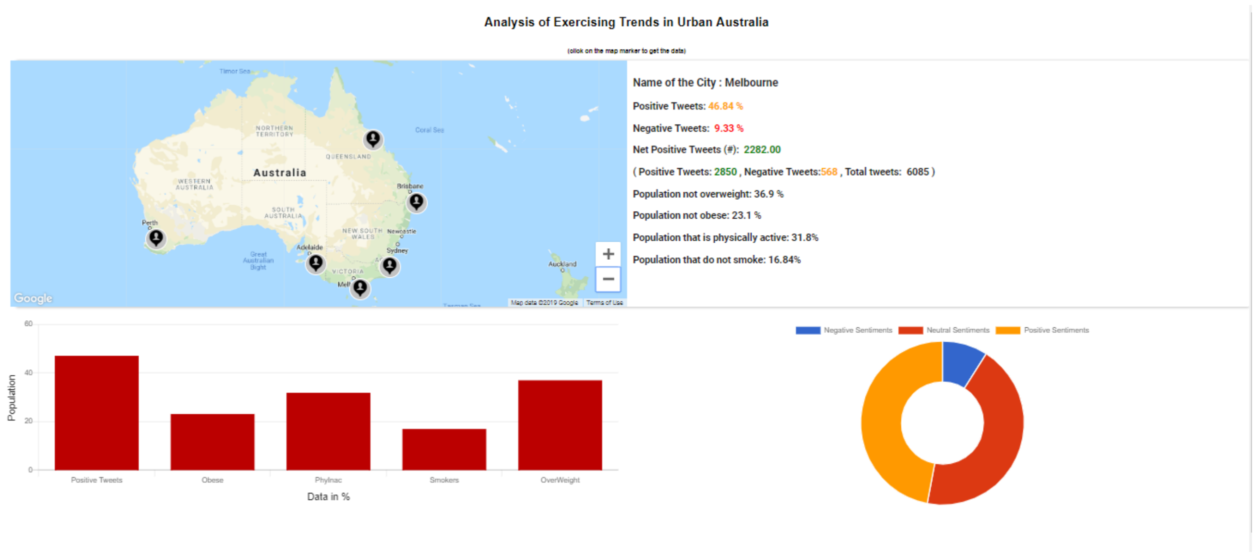
Once you click on the marker,you can see the width of the map is reduced to 50% and a textual representation of the selected city is displayed on the right side of the screen. Followed by a graphical representation of the data at the bottom of the map.

The textual data is dynamic and changes the colour depending on the scale of the data it receives.

In the graphical section, you can check the values for each of the bars in the bar graph and also you can hover over the Pie Chart and view the values.

You can restrict the data that is to be viewed on the Pie Diagram, by clicking on the legends provided on the top.

- Known issues:
  Double click on the marker due to the older version of react.


Analysis of Exercising Trends in Urban Australia

(click on the map marker to get the data)

Name of the City : Melbourne

Positive Tweets: 46.84 %

Negative Tweets: 9.33 %

Net Positive Tweets (#): 2282.00

(Positive Tweets: 2850 , Negative Tweets:568 , Total tweets: 6085)

Population not overweight: 36.9 %

Population not obese: 23.1 %

Population that is physically active: 31.8%

Population that do not smoke: 16.84%

Figure7.1: Screenshot of the working UI

8. ANALYSIS

8.1 Data Gathering from AURIN (Australian Urban Research Infrastructure Network)

For our story we had to collect the data for the number of people obese for each city that we were targeting. To select data for each individual city, we used coordinates for each city in the bounding box and selected data within the postcodes for each city. It was important including the data for each postcode to get the more accurate results.

8.2 AURIN API Framework

We needed to collect the data from AURIN using its API framework. AURIN has developed an Application Programming Interface (API) to allow users to access a whole range of open datasets, without having to log in to the portal. We registered on the portal to obtain the credentials to access the API. After writing the API access script in the python we managed to access the AURIN API framework and the datasets available on it. The first problem we encountered was the number of datasets we were receiving were not enough. The datasets on AURIN portal were 4996 but by access the datasets through API we were getting only 1200. So, to retrieve datasets which were required, we logged into the AURIN portal and individually downloaded the datasets as a csv and json file.

8.3 Information of Datasets

Using AURIN online portal we came across different data sets and the latest data set which matched our requirements was the one from the health sector. The dataset we selected was (PHIDU) Prevalence of Selected Health Risk Factors - Adults (PHA). We collected the data of obese people for each city over the age of 17 in the given post codes. The data set contains the variance, the number of people obese per 100 with their respective PHA name, and PHA code. The Population Health Areas, developed by PHIDU, are comprised of a combination of whole SA2s and multiple (aggregates of) SA2s, where the SA2 is an area in the ABS structure. The data is by Population Health Area (PHA) 2016 geographic boundaries based on the 2016 Australian Statistical Geography Standard (ASGS). Estimates for Population Health Areas (PHAs) are modelled estimates and were produced by the ABS; estimates at the LGA and PHN level were derived from the PHA estimates. AURIN has spatially enabled the original data. So, by using the analyses tools provided by AURIN we create Spatialize Aggregated Dataset to get the location for each value. This tool on AURIN creates a shape file that can used on google maps.

Additionally, we needed to create a relationship between obesity and the factors which we think plays major role in that. So, we also collected some more datasets to strengthen our story.

These data set included the data of adults as follows:

- Adults who were current smokers.
- Adults who were involved in little or no physical activity.
- Adults who were sitting for most part of the day.
- Adults who were over-weight but not obese.

All these datasets consist of variance and number of people per 100 effected by the factor. So, for each city the dataset contains values in their respective PHA code. To get the accumulative score for each city, we took average of all the values and came up with a single average value for each city. For each dataset the process was repeated to get a good average score for every factor. Then all these values written into a csv file to perform further analysis. The data was accumulated in a csv file so that further analysis can be performed on the data using R. All the data related to tweets that we gathered from the twitter and the data that we retrieved from AURIN was then imported to the R.

8.4 Data Gathered from Twitter

We have gathered the tweets which depicts positive, negative and neutral sentiment about the physical activities in the following cities. Most of our data we gathered from Sydney and Melbourne respectively as these two cities tweeted the most about the physical activities. The sentiment was on positive side for both cities. Brisbane and Adelaide also tweeted positive along with the other cities. However the city of Canberra is only city which tweeted negative about the physical activities.

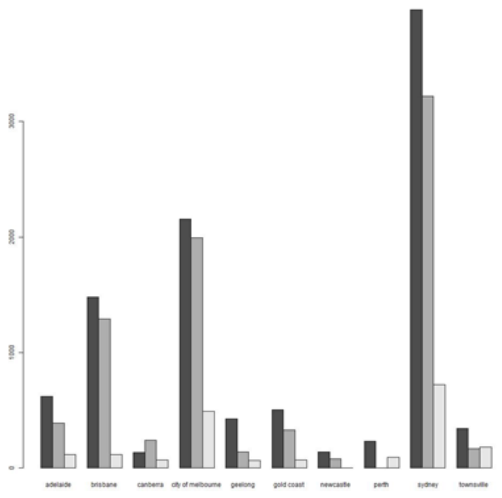


Figure 8.1: Bar chart of Twitter data

8.5 Analysis

In the analysis part we performed different types of test on our data to get the meaning out of all the numbers we have collected. For this we performed the following tests:

- Normality tests on our sample space
- Correlation between positive sentiments and the factors of our story
- Correlation matrix to get the deeper understanding of our data and how it is related to all the factors on which we are working.
- Importance Graph.

8.6 Normality in Sample Space

After collecting the data of obese population for each individual city, we first calculated the population which was not obese. The calculation made by R gave us a sample space for the people who were not obese. To verify how good our sample is we perform this normality test. Our sample for people who are not obese is following normal distribution. People who are not smokers also follows normal distribution however the data regarding people who are not physical active does not follow normal distribution.

Following is the image which represents the normality of our sample space.

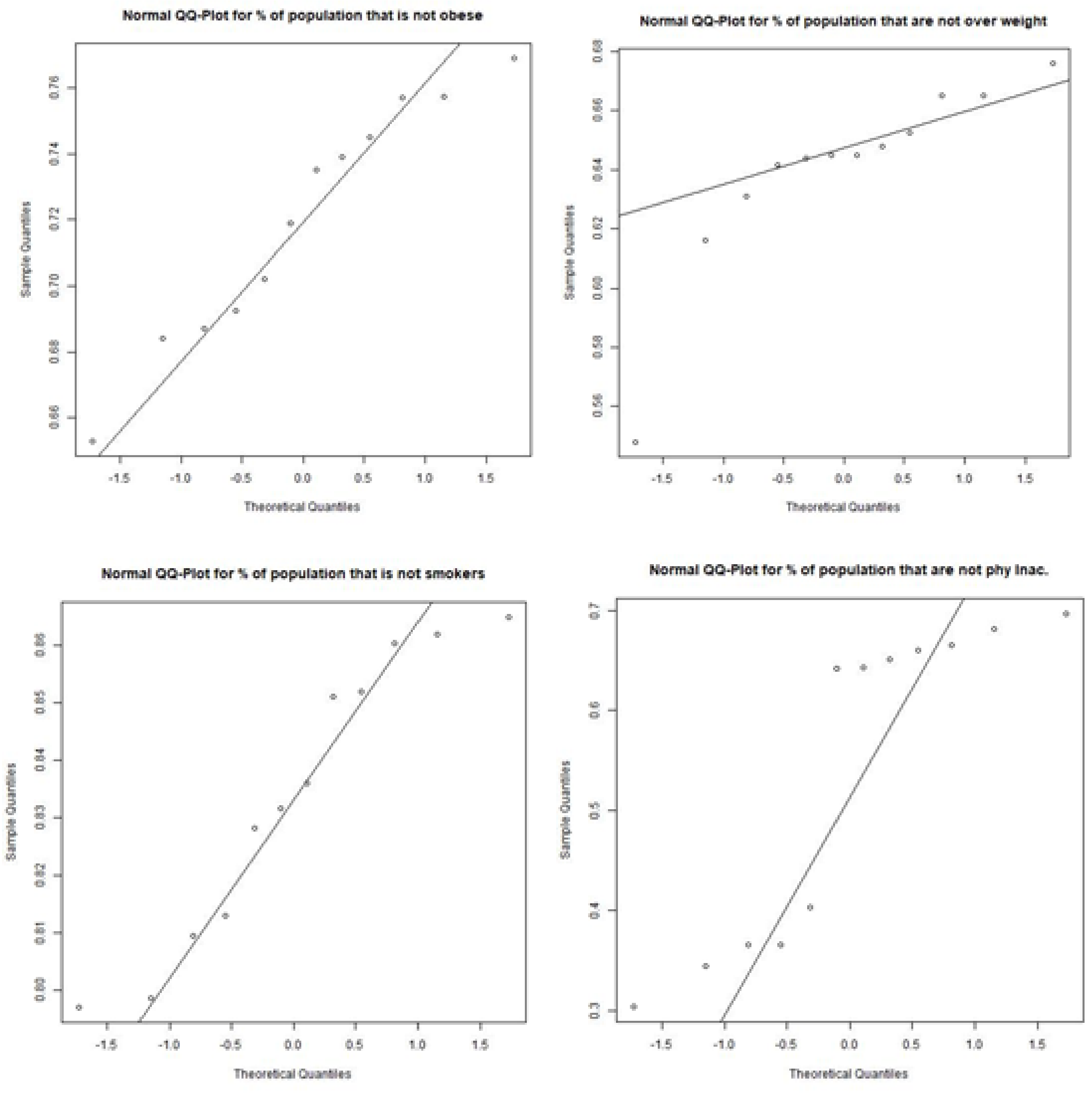


Figure 8.2: Normality tests(QQ Plots)

As shown in the figure above each of our sample follows normal distribution except for fourth graph which represent the data of people who are inactive.

After performing the analysis on our sample space, we applied regression tests on our data for each of the factors contributing in our story.

8.7 Regression Tests

The regression test for each of the factors with the positive sentiment by our twitter data is displayed below.

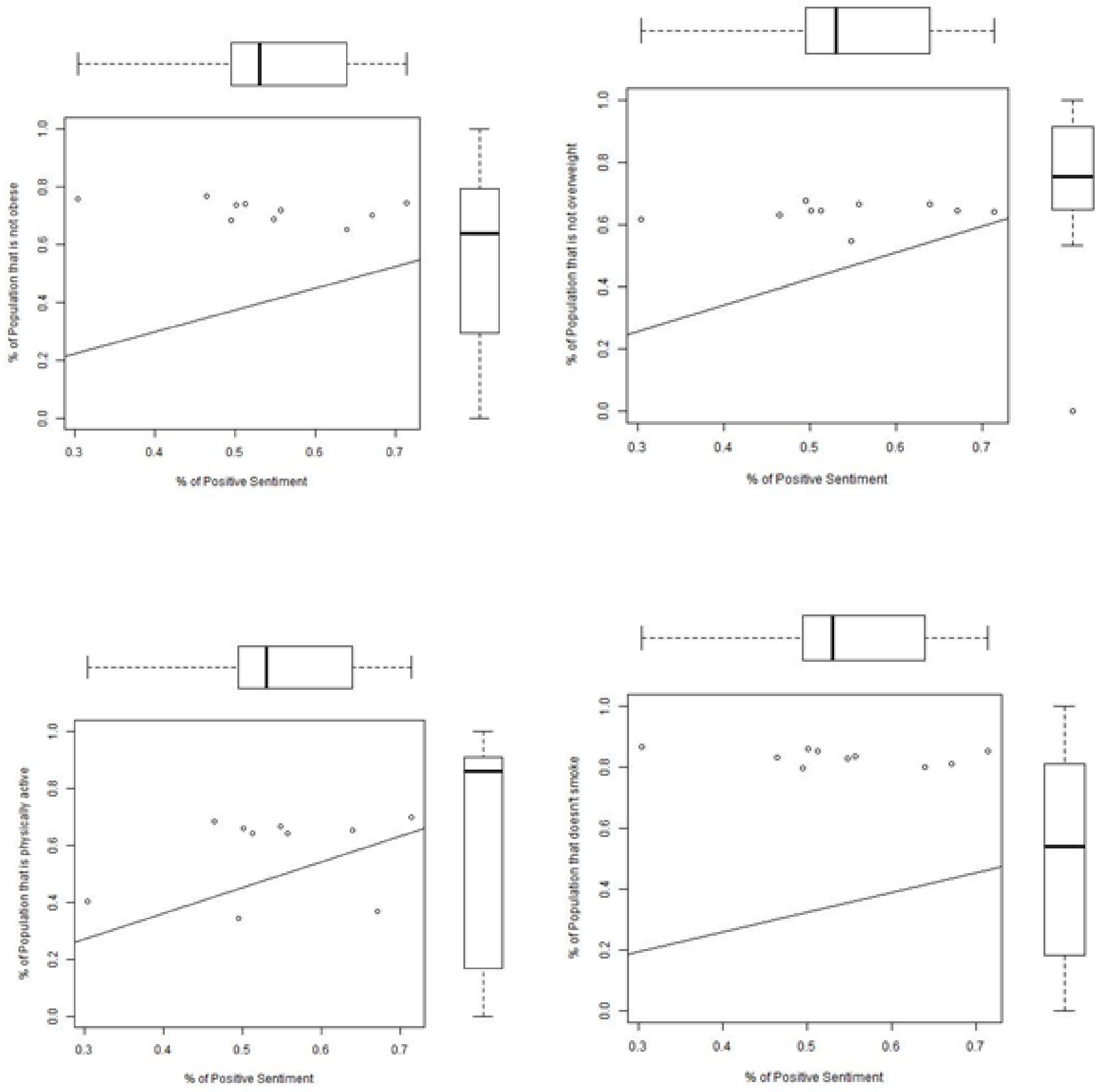


Figure 8.3: Box Plots

The regression co-efficient of people who are not obese to the positive sentiment provided by the twitter data is 0.7496. The number is on a higher side which shows the association between the two factors. Although this number increases when it comes to the regression of positive sentiment to the people who are overweight which is 0.8473. The most significant relation between the two factors was observed on the regression test of positive sentiment to the people who were physical inactive. This number shows a significant improvement reaching to highest value of 0.9023. However, this drops to 0.647 when it comes to the relationship of people who are smokers.

8.8 Correlation Matrix

We created the correlation matrix of our four factors with each other. The correlation matrix explains the relationship between each of the factors and how they are dependent on each other. Following is the correlation matrix created by using R.

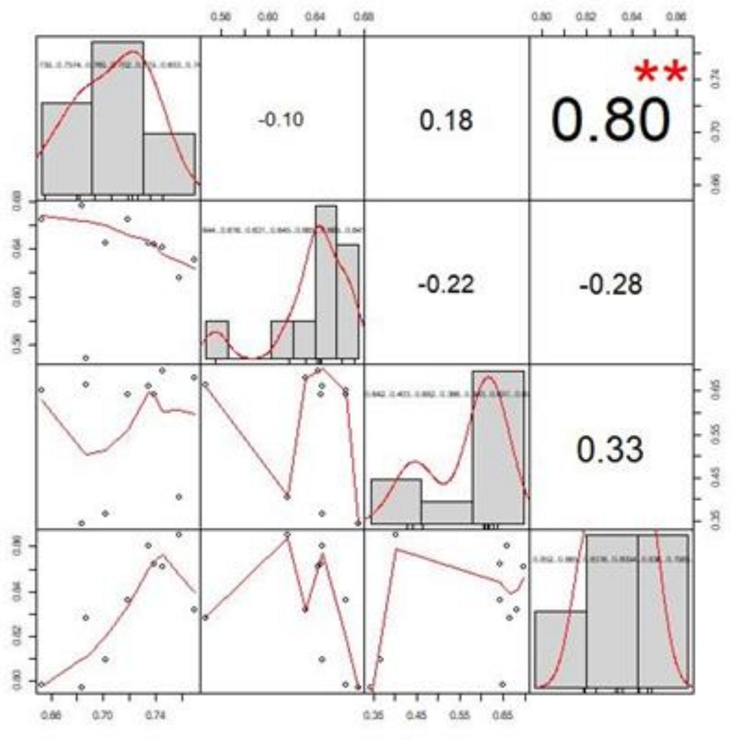


Figure 8.4: Correlation Matrix

The correlation matrix shows that how obesity is dependent on each of the other factors along with their scatter plots. The plot shows the positive correlation of obesity and physical. The correlation of obesity and smoking is the highest gives us the highest number.

The importance graph of the four factors shows that how each factor plays its role in our story where this data is compared to the sentiment analysis of the data we have gathered from the twitter. Looking at the importance graph we can assume that the factor which plays the most important role in our hypothesis is physical inactivity. It has the influence of 38.2 percent. The data related to people who are not smokers shows it has the least impact of all the four factors. People who are overweight and people who are obese plays an important role because their number is significantly high. The numbers are 84.7 percent and 79.1% respectively.

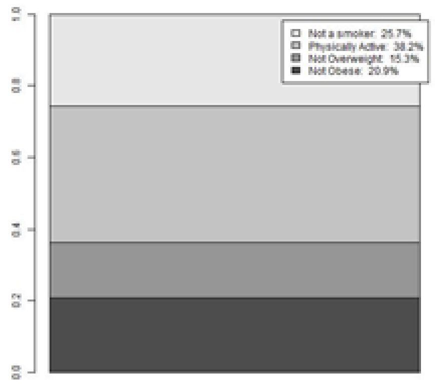


Figure 8.5: Importance

8.9 Summary

The data above shows that the two dominant factors which support our story are the number of people who are obese and the number of people who are overweight. The cities who tweets positive about the physical activity contains less population of people who are obese and overweight.

9. USER GUIDE

To set up the cluster and install CouchDB, do the following:

- run: . ./openrc.sh; ansible-playbook --ask-become-pass nectar.yml
- manually log into the 4 instances using the IP addresses (ubuntu@xxx.xxx.xxx.xxx) that are the output of the above code.
- On instance2, copy the directories "public", "src" and the file "package.json" to

/home/ubuntu, from the git repository "https://github.com/srivastava-sar/COMP90024_Project/UI_prod"

- run: . ./openrc.sh; ansible-playbook --ask-become-pass nectar-2.yml

References:

- Most of the ansible code (used in nectar.yml) has been sourced from the commands that were shown in lecture 5
- Code to install and configure for CouchDB (used in the last part of nectar-2.yml) has been sourced from https://github.com/glynnbird/ansible-install-couchdb and https://github.com/kafecho/ansible-couchdb2
- $ python twitter_harvest_prod.py

10. CHALLENGES OF HARVESTOR

10.1 Data Parsing

One of the major differences of our system from other systems was that we were extracting and saving only relevant tweets rather than all the tweets from Twitter. We realised that Australia is a fitness conscious country and hopefully micro-bloggers will tweet enough about their likes and dislikes for our system to analyse their sentiments. We decided to preprocess and remove any special characters and links and user mentions in the tweet in order to analyse their text sentiments.

10.2 Location Information

(1) Because the tweets were coming from different sources of Twitter, it was an initial challenge to assign a specific location to the origin of the tweet. Each tweet has a different format of location as well. For eg: some tweets had "Lygon st, Melbourne" whereas some had "Melbourne, Victoria" and even more, some had "Brunswick, Australia". Google Maps API being a paid service after a certain number of calls did not seem fit for an academic project and thus we decided to use open source Python package and augmented coordinates based on location. This solved our problem of treating for eg: "Melbourne" and "Melbourne, Victoria" as different groups.

(2) We realised during harvesting that historical tweets fetched from user timeline does not usually have location information. To overcome this challenge we decided to use user

account location information as the tweet location. This reduced our null location error from ~40% to <5%.

10.3 Storing user in a different database

We realised that in order to extract all the tweets from users we need to store all tweet authors and their user mentions in another database with a status code denoting if all previous tweets of that particular user has been analysed yet or not.

11. POSSIBLE FUTURE IMPROVEMENTS TO THE SYSTEM

Being a very open ended project, we realised that much can be achieved in analysing the tweets. However, being an academic semester course project, we narrowed down our problem statement and implementation.

As part of our future implementation, we intend to analyse tweet using emoticons and other special characters to assign a sentiment to the text. We also wish to use light weight machine learning algorithms to make a classifier around our present system.

- Data Collection:
  Include User followers and following to extract tweets
  Automate data collection from Aurin using the Aurin API.

- Data Analysis:
  Weigh the data by population
  Use other population statistics like gender and age, to find variance in the data.

12. CONCLUSION

Twitter contains much useful data which can be used to do some research, such as this project. How to harvest specific data into database and combined with visual output, such as map used in this system, is meaningful.

In this report, the system design and architecture are shown, which lead us to build this system. Some functionalities and tools are used in different stages, which are introduced in detail, separately. Finally, the map is successfully displayed with statistic graph and proportion.

REFERENCES

[1] Abhirup Ghosh. Cloud Computing.
https://www.cse.iitb.ac.in/~abhirup09/Docs/cloud_computing_final_report.pdf

[2] Richard O. Sinnott, Farzad Khodadadi, and Yao Pan. Cluster and Cloud Computing – Lecture 5: Cloud Computing & Getting to Grips with NeCTAR/UniMelb Cloud & Scripting & Git.

[3] Ansible playbook to install CouchDB 2.0 on Raspberry Pi.
https://github.com/glynnbird/ansible-install-couchdb

[4] React App and Google Maps.
https://github.com/Tim152/clustering-google-map-react for cluster maps integration.
https://reactjs.org/docs/getting-started.html for React JS.

[5] Asynchronous API calls.
http://api.jquery.com/jquery.ajax/ for AJAX API calls.

GIT LINK AND YOUTUBE LINK

https://github.com/srivastava-sar/COMP90024_Project

https://youtu.be/oAbC0Xkhoyo