COMP90024 CLUSTER AND CLOUD COMPUTING

Assignment 2 - #TheDeadlySins #2019

MAY 15, 2019

TEAM 19

Niloy Sarkar, Sania Khan, Sai Deepthi, Sunanda, Athulya JU

TABLE OF CONTENTS

Introduction

Color in certain places has the great value of making the outlines and structural planes seem more energetic. One such colourful and energetic city is Melbourne. That is the very reason why it attracts people from all across the world to live here, making it the second populous city of Victoria. The main aim of this report is to discuss and describe how a deadly sin, in this happening city, can be portrayed with the help of some technological amuses like Twitter and some authenticated dataset from Aurin.

The seven deadly sins also known as cardinal sins, is a grouping and classification of vices within the Christian teachings. Classification of habits or behaviours is done if they lead to immoralities. According to the standard list, they are pride, greed, lust, envy, gluttony, wrath and sloth. These sins are often thought to be abuses or excessive versions of one's natural faculties or passions.

In this era where social media rules the world and people spend time more on these platforms rather than with each other, it is becoming quite easier to get hold of a person's mindset and behaviour if his/her profile in social media is tracked.

Twitter, one of the most universally wide used application with about 330 million users across the world is brimming with millions of tweets, photos and videos on a daily basis. It also provides public API which allows developers work on its vast network of contents with the help of live/stored tweets anytime, anywhere and by anyone to build other related applications.

Getting hold of a bunch of data from Twitter and with the help of some analytical tools like Nectar, CouchDB, Python and other libraries we have created a software for sentiment analysis based on one of the sin, Gluttony. In other words, Greed is a urge for excessive acquisition or possessing more than what is actually required. Narrowing it down to one particular channel, we have chosen Workaholism. Greed in our society is manifested as workaholism. Note that, we're not talking about the actual medical condition where someone loves working or is addicted to work. We're emphasizing on: the addiction to earning. There are people who does not love their jobs and do the bare minimum for their hourly pay, but will do extra hours, extra work and take on secondary jobs to get more money in the bank. They will then use that money to buy material goods to hoard or boast of, or events and experiences to boast of. All that matters is accumulating enough wealth. And most people indulge in greed to a degree.

Yet thanks to a combination of greed, pride and envy, we don’t realize how greedy or jealous we actually are, as we will spend right the way to the edge of every pay check trying to keep up with the Joneses. Plenty of people survive on 4-10k/year in Western countries. Yet a workaholic with a 200k pay check and 100k in debt feels like they are living a bare basics lifestyle when, in fact, they are consumed by greed and envy.

Thus summarizing, we’re focusing on the concept of work people do and how it changes based on the wage per hour, inclining to the greed factor rather being in the necessity or satisfaction scale. We have made use of datasets from AURIN to compare our gathered data from Twitter and thereby added visual analysis of the same.

System Architecture

The system architecture of this project develop a Cloud based solution that utilizes a multitude of virtual machines across the UniMelb NeCTAR Research Cloud for harvesting the tweets through the Twitter APIs. The system of our project uses four instances running on ubuntu 18.04. In these instances, one is acting as a web server, one of these hosted a CouchDB node for storing the tweets that are harvested by using the Twitter APIs. Out of the four instances one is used for harvesting the tweets using the Stream API and one is used for harvesting the tweets using the Search APIs.

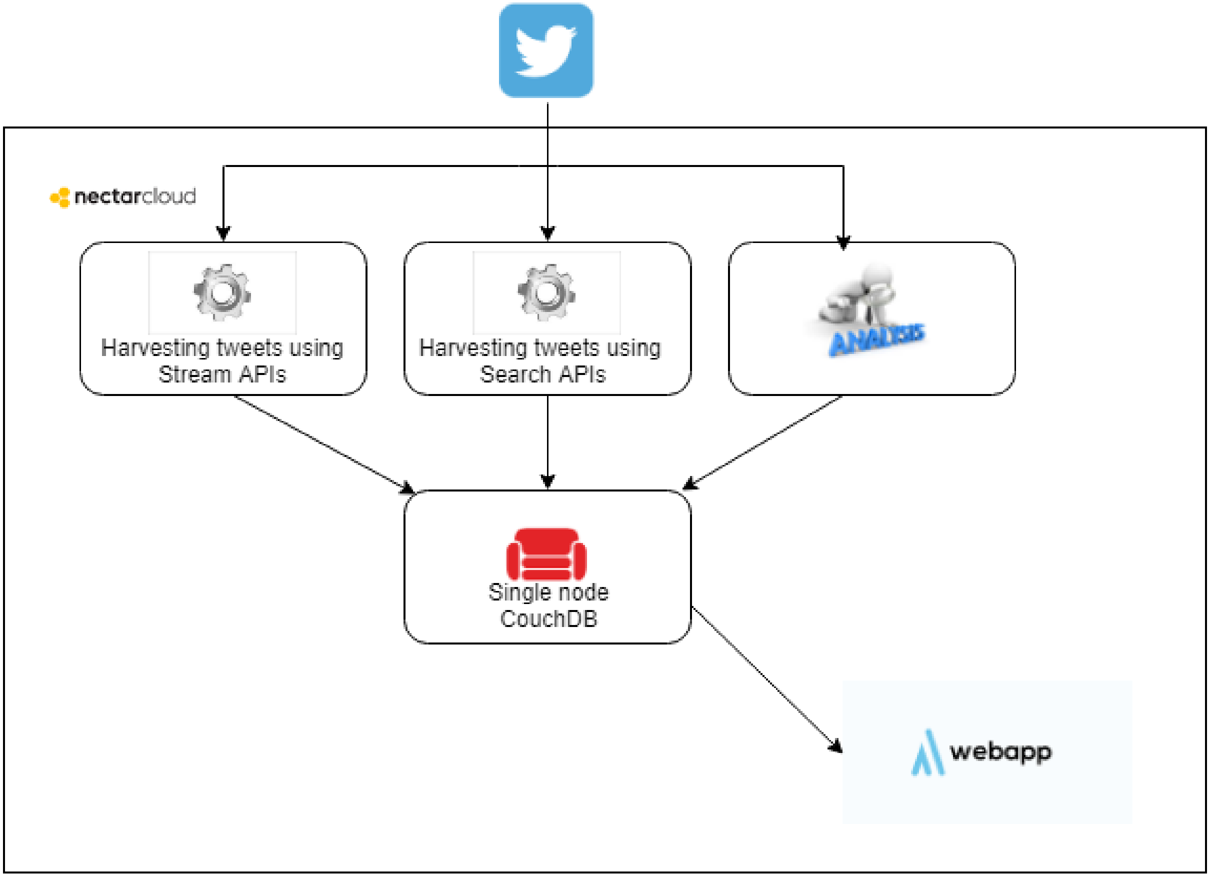

nectarcloud
Harvesting tweets using Stream APIs
Harvesting tweets using Search APIs
ANALYSIS
Single node CouchDB
webapp

Figure 1: System architecture

The volume that is assigned to the CouchDB in this project is 50 GB. More storage can be available depending on the amount of data that is stored in CouchDB.

System Design

The system is basically designed to develop a Cloud-based solution that utilizes a multitude of the virtual machine across the Unimelb Research Cloud for harvesting the tweets from the twitter using APIs. After harvesting the tweets, the next part is to store the tweets in a database. The CouchDB database is used for storing the tweets that are harvested by the harvester. After storing them in the CouchDB views are created and the API of the views is used to link the data from the database to the web application,

The CouchDB is a document-oriented database management system that stores data as structured document, like the XML or in the form of JSON object. The tweets that are coming from the harvester are stored in the CouchDB in the form of JSON object. Also the data we get from the AURIN is stored in the CouchDB. A CouchDB instance can have many databases, each database can have its own shards, where adding and deleting is done through HTTP calls. This database also provides the functionality of MapReduce Algorithm in which the map function is used to distribute the data over the machines while the reduce function hierarchically summarizes them until the result is obtained. The main advantage of the MapReduce algorithm is moving the process to where data are, and this way it reduces the network traffic. In this project we use a single node CouchDB.

For collecting the tweets, three instances is used. One instance is used to harvest the tweets using the stream APIs. Another instance is used to collect the tweet from the twitter by using the search APIs. One instance is used for parsing the tweets and then one instance is made to store all the tweets that are coming from the harvester and on this instance CouchDB is hosted.

The UniMelb NeCTAR Research cloud is used for creating the virtual machine and the CouchDB is used for storing the data and for the automation part ansible scripts are being used in this project. The user interface is linked with the views of the CouchDB.

Deployment

The main goal of our project is to automate the system. For that we use the ansible script. Ansible is open source used for configuring and managing the computers. The ansible script runs on one host machine which will create all the instances on the server and also automatically deploy the CouchDB and the harvester that is used for harvesting the tweets from the user.

This works in the order which is called orchestration, that is the automated configuration and coordination of the computer system. First of all, we connect to the UniMelb Research Cloud and then check our host machine with the necessary software. After that we list all the images, create the volumes and the security groups with security rules. Then instance is then launched and finally attach the volume and the security groups to the instances and then create snapshots of the volumes.

In this project we created four instances one for the Database Server on which CouchDB is installed. All the tweets that are harvested by the harvester using the search APIs and the stream APIs are stored on this CouchDB instance.

NeCTAR Research Cloud

The Nectar Research cloud is National eResearch Collaboration Tools and Resources lead by University of Melbourne. This research cloud has four key aspects. One is Research cloud program that provides open stack infrastructure as a service. Another one it provides virtual laboratories programs. It also provides the Research data services.

The research cloud is an open source cloud technology and it provides many services like image service, network service, orchestration service, metering service compute service, security management, object storage and block storage services.

In our project first we made up the instances and used the orchestration service for configuring and coordinating the system. Then provide the volume like we provide 50 GB volume to the CouchDB instance. Then attaching the security groups for providing the security management.

The main advantages of the NeCTAR research cloud are as follows:

- Access to Remote virtual machines

  As it provides access to remote virtual machines that help in the analysis of the data that would be heavy on our local systems. Remote access to the virtual machine provides us the benefit to work from anywhere.

- Access to Multiple machines

  The analysis and the retrieval of the data is split up on different machines that provide us a greater number of resources and saves a lot of time.

- Library of Images

  It provides a vast library of images that helps us to choose any operating system we are comfortable in.

- Security management services

  The research cloud provides us different security management services for safekeeping our research. It provides the ability to build our own security measures to overcome oversight.

- Facility of setting up the security groups

  It provides the facility of setting up the security groups to our instances. Security groups control the opening and the closing port inside the system. It provides the security protocols.

The disadvantages of the NeCTAR Research Cloud are:

- Network issues can cause work slow down. Since we are dealing with a large amount of data at a time, the system slows down causing a temporary pause or delay in work.
- Network only available through connecting the university
- Steep learning curve
- Working with ubuntu is fine as it is free but not user friendly

- Faced issues with keypairs while assigning them to instances
- Security has always been a concern when it comes to cloud. The possibility of data being lost or leaked always persists.
- One other challenge encountered with Nectar is the movement of large files. It was both time consuming and tedious, given the json formats of the files.

Challenges

There are a few challenges our team faced while working on this project. Some of which are listed here. The first and foremost problem faced was with the json types. The harvesters produced a different json type(different arrangement of the attributes) and the ones in Unimelb research cloud is different. Thus, we had to validate those and edit it accordingly. Once that was done, we had to work on Aurin data. Again, the json formatting with Aurin was complex and entirely different which lead us to sort that based on acceptable json format. So mapping each json to the proper needed attributes was time consuming.

The next issue was with populating the database Due to Twitter's rate limit, the amount of data that can be obtained in one go is fairly limited. Which is why

(a) the scripts needed to run for a long time to have data adequate enough for analysis

(b) find other ways(json from unimelb research cloud) to help with the analysis.

The other challenges faced were while doing the Ansible part. Initially we wanted to complete the script with few files but after that due to the issues with orchestration and knowledge constraint on ansible it was difficult. So we used roles to create the script.

We also faced network issues due to the usage of wrong providers in network. We faced challenges when creating the couchdb and while giving permissions to the other instances to the couchdb, which was resolved with references

Problems were faced while mounting the volumes to each instances, which was solved with practice and reference. Ubuntu, not being so user-friendly gave us enough trouble while we had to find our way through it.

Tweet Scenario Analysis

The main idea of this project is to filter the tweets based on the requirement. Given the concept of Greed, we could not just take all the tweets based on work, pay or wages or number of hours being worked for instance. There had to be a differentiation between the tweets that conveyed love or passion towards the work and that of work done under pressure.

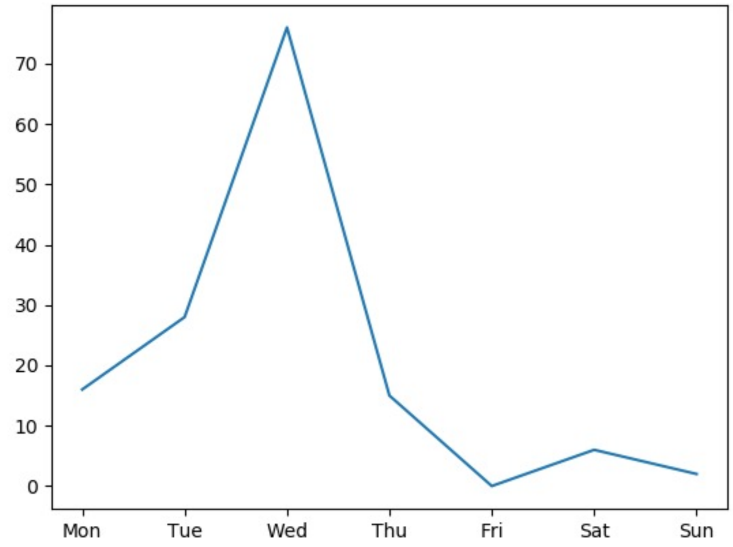
The dataset from AURIN was taken to perform a comparison with the tweets harvested from Twitter and provide a detailed analysis of our concept. We used the dataset Sa3-w17a occupation by total personal income(weekly) by hours worked from AURIN for our comparison purpose.

Twitter provides a free API for the developer usage which can be used to stream tweets. By using Tweepy, a python library which helps in streaming the tweets live, we got hold of the tweets by customizing the parameters such that it gave us the data we required. We gave a few keywords related to workaholic, hoarder, greed and money. We also compared the data with the AURIN dataset which proves that people are working double the time given a high pay.

Twitter Official API has limitation of time constraints, which is that tweets older than a week cannot be fetched. Thus, we got hold of tweets from the Unimelb research cloud using the curl commands to search tweets according to our needs. The other way to go about this was by using the python library GetOldTweets. This library has no restriction based on the number of tweets, instead it fetches tweets based on the user. Basically, when something is entered on the twitter page a scroll loader starts, when it is scrolled down we can find more and more tweets, all through calls to a JSON provider. Using that same technique, in browser the older tweets were obtained.

All the tweets that were fetched were stored in our database on CouchDB in Nectar. The twitter API is such that, for each tweet, it gives a JSON that has information about the tweet like its creator, the time of posting and all the contents involved in it. After acquiring our tweets, we started storing it in CouchDB with the help of map reduce.

Visualisation



x-axis : Days of week

y-axis : number of hours of work

The main idea of our analysis revolves around the sin “Greed”. In generic greed is classified when people comes up with ideas like “I want” or “I desire” and so on. Trying to cut the cliché here, we went on with choosing work as a path of greed. Working or being workaholic is also a form of gluttony. It is driven by a strong urge or desire to earn more money or stay ahead of your peers in the workplace.

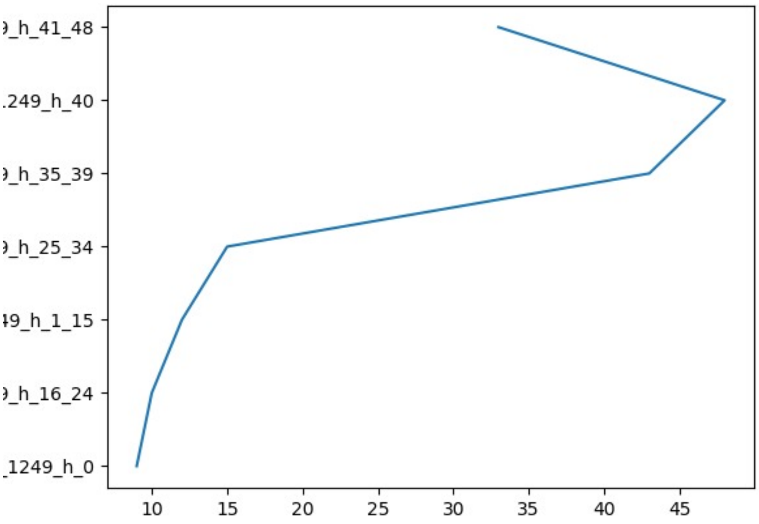
We decided to analyse this concept with the help of tweets. We made sure that the normal quotes like, work or money is not included in our list. We used a set of keywords related to greed, workaholic (extremity) and other similar words with the help of thesaurus.

In the appendix section of the assignment are certain curl commands to access the university research cloud data. We run that as a shell script that's stored in a json file. Reader.py in our project(included in github) access this json line by line using file handling, extracts the necessary information and adds them to the couchdb server to further augment our dataset.

We do this because of Twitter's rate limit the rate at which we get data is slow and to improve our final analysis when we're augmenting our data.

The above graph is a representation of our analysis which shows that the work rate is higher at the mid-week section and decreases gradually by the week-end. This is mainly due to seek attention among the peers.

Similarly, we also got the result where the graph hikes up during the weekend, which directly shows that the work done is due to the high pay during weekends. The comparison of our theory was using datasets from AURIN.



x-axis: number of managers

y-axis: number of hours

The graph above proves our analysis that the number of managers working more than 45-48 hours is higher. And the higher they work the larger the amount they get paid for it.

Error Handling

1. Retweets removal has been handled through a retweet key checker when the script accesses the documents and load the data into pandas data frame. It checks the unique Id of the tweet and drops if it is repeated i.e retweeted again.
2. Try and catch block are used to handle Twitter's rate limit. The block looks for certain error codes sent by Twitter and if the error code is in 400s there is a connection issue and the programs stops(sleeps) for a delay of 15 mins

3. Large JSON files are parsed using File handling to avoid loading large JSON files and running out of memory.
4. The Twitter app credentails i.e. consumer_key,consumer_secret, access_token and access_token_secret are stored in the database and retrieved during runtime to avoid it being leaked.
5. Couch db stays in its separate instance to constantly have its data being built and updated that keeps the data safe.
6. Twitter standard Search API makes a 100 requests and then sleeps for 15 minutes in order to again not exhaust rate limit.
7. Twitter Stream API has a constant connection but in any case it expires,it waits and then restarts the connection.

Conclusion

As quoted by Gandhi, there is enough in this world for man's need, but not quite enough to quench his greed. As energetic and colourful life is, it is highly necessary that we understand its importance and enjoy when you still can. All work and no play makes everyone a dull soul.

This research paved an interesting way for all of us to learn technologies. Beginning from choosing a sin and working its way to fetch tweets, store it in CouchDB, analysing the data, we got hands-on experience with all these new technologies.

APPENDIX

Deployment:

These are some of the commands that needs to be followed for the deployment:

Source ./run-instances.sh

The above command creates four instances.

Source ./run-CouchDB.sh

The above command will install Couch DB

Source ./run-AppServer.sh

The above command will install dependencies software and trigger Analysis.py which will run the analysis.

Source /run-ProcenServer.sh

Installs dependencies and search.py which does the tweets harvesting

Source ./run-Streamserver.sh

Installs dependencies and run stream.py

Links :

Github link : https://github.com/snap995/COMP90024-Cluster-and-Cloud-Computing-/new/master

Youtube Demo link: https://www.youtube.com/channel/UCKC4T5ToNvrtNxND--uXEuw?view_as=subscriber