**Project Report**

**Introduction**

This a project report created after the analysis of a twitter data set. The Data Set contains 10000 twitter profiles from a subset containing 3 million profiles.

**Steps before running the pipeline tools.**

1 Creation and configuration of VM

2 The VM was set up using the image and with the specified configuration given in the assignment steps.

3 RSA key setup was done and a pair of private public keys were generated

4 VM was restarted

5 Mongo DB access controls and authentication were successfully disabled by first stopping 6 Mongo DB and changing contents of mongo.conf file. Mongo Db was again restarted

7 VM was restarted

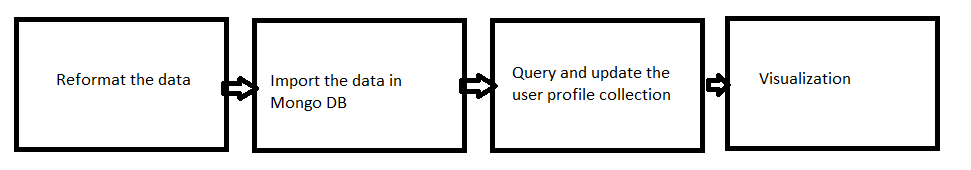
8 The projectA directory was created under the user directory and tools provided in the canvas were transferred using WinSCP

9Tools were extracted, and the properties file was modified to mention project directory

and java home path.

10 Project was successfully built, and class files successfully created.

**Q3 A**

****

**Overview**

Data arrives in text files which may or may not be in standard format. These files go through steps in which they are converted into appropriate format. Metadata is added to give structure to these files. The files are imported into the database. Tools are invoked to analyze this data analyzing each data and updating it accordingly. Rerunning of the tool ignores profiles which are already updated. The raw data is exported and transformed into data that can be visualized to make decisions.

**1 Data pipeline task 1: Reformat the data**

**Command for task**

This step involves reformatting the data from one format to another. MongoDB accepts only

UTF-8 and the given dataset is in the form of ISO-8859-1.

To proceed and be compliant with database accepted format we convert it into UTF-8 format

The file reformat.sh is used in this process

Inside the file

iconv -f ISO-8859-1 -t UTF-8 $1 > $2

Now the files are converted into TSV file with utf-8 format

We add a header line so that the database can understand and gives the data a structure so that MongoDB can understand it. The header line helps to name the data columns in the tab separated file and helps us to query the data.

**Data pipeline task 2: Import the data into MongoDB**

With the help of the following command the data is imported into MongoDB the shell script file import\_mongodb.sh is invoked and data is imported into MongoDB

**Command for task**

/home/sbedekar/projectA/I535-TwitterProjectCode/bin/import\_mongodb.sh projectA profile tsv /home/sbedekar/ProjectA/I535-TwitterProjectCode/bin/user\_10000.tsv

**Data Pipeline Task 3: Query and Update the User Profile Collection**

Twitter users can have a variety of location both real and fictional. This step in the Data pipeline uses the google geocoding API’s to validate such locations. If these locations are validated, then we get their latitude and longitude as well as formatted address and they are updated into the database.

The validation is done for each profile for which geocode does not exist in the profile collection and the validation is ignored for profiles which are already updated.

**Command for task**

/home/sbedekar/projectA/I535-TwitterProjectCode/bin/QueryAndUpdate.sh /home/sbedekar/projectA/I535-TwitterProjectCode/config/config.properties projectA profile input/query.json /home/sbedekar/projectA/I535-TwitterProjectCode/log/test44.log

The geocoding query has a limit of approx. 2500 per day but I observed once it 2946 records were processed, and 2937 records updated. This output is of MongoDB and sometimes location is null and hence the count is more.

Execution flow

This is repeated for every twitter profile in the db

Reformat the location string with ‘+’ for whitespaces if location is not empty or null.

If a location is null or empty, the geocode is null.

Next a URI is created signed and converted to ascii so that proper result can be obtained from google. This geocoding is done anonymously so we don’t have client id and API key and so the geocoding is limited to 2500

URI is converted to URL

Through a series of steps like opening a HTTPURL connection to getting a message

we get the output in JSON format from which we get the latitude and longitude.

MongoDB is updated with the formatted address, latitude and longitude within the geo code field.

**Data Pipeline Step 4:  Visualization**

Visualization is the final step in which we see where users are located on the map. In this step we transform the raw data exported from Mongo DB which cannot be analyzed by looking at it into a visual display of location of profiles by plotting of profiles on a google map.

**Steps**

Extract 50 or more subset of all records where geocode is not null and exists in a CSV file.

Format the CSV file data in a format Google chart JavaScript lib can use.

Create a html file and insert the formatted data in an array.

Open the file in a web broswer to see the visualization.

mongoexport --db projectA --collection profile --query '{ $and: [ { geocode: { $ne: null} }, {geocode: { $exists: true } }, {"user\_location" : {$regex : ".\*,.\*"}}] }' --type=csv --limit 100 --fields "user\_name,geocode.location.lat,geocode.location.lng" --out "/home/sbedekar/projectA/subset\_profiles.csv"

Notice the above query gave me profile containing **commas in their addresses. To have a better visualization I added two profiles with only country as the location**

**Q3 B**

7468 locations have the geocode locations i.e. have latitude and longitude. That is in addition to exists condition we have to specify geocode should not be null

2516 records have updated the geocode as null and the geocode exists conditions returns these records.

16 records could not be updated.

For the past 8 days I have been running the command for geo code resolution but all 10000 records are not updated

**Description of records that were not validated.**

1 These records had the users entered any arbitrary value as the location in their twitter profile

for e.g.

La Mirada/Whittier Border

A Stadium or Arena Near You

Justin bieber heart :D

/ me, my tears, & I /3

The above locations are unusual, and they did not exist in the google database and could not be validated

2 Some had location as empty values

It is not possible to get geocodes for such locations as the tool adds geocodes for a location which is not empty or null

3 Some of them had multiple locations like

VSU/ Brooklyn/ Florida

VCU Philadelphia

These could have been able to geolocate if there had been a function to preprocess such locations

For e.g.

VSU/ Brooklyn/ Florida

This is confusing location for geocoding API

Exceptions to the unusual address

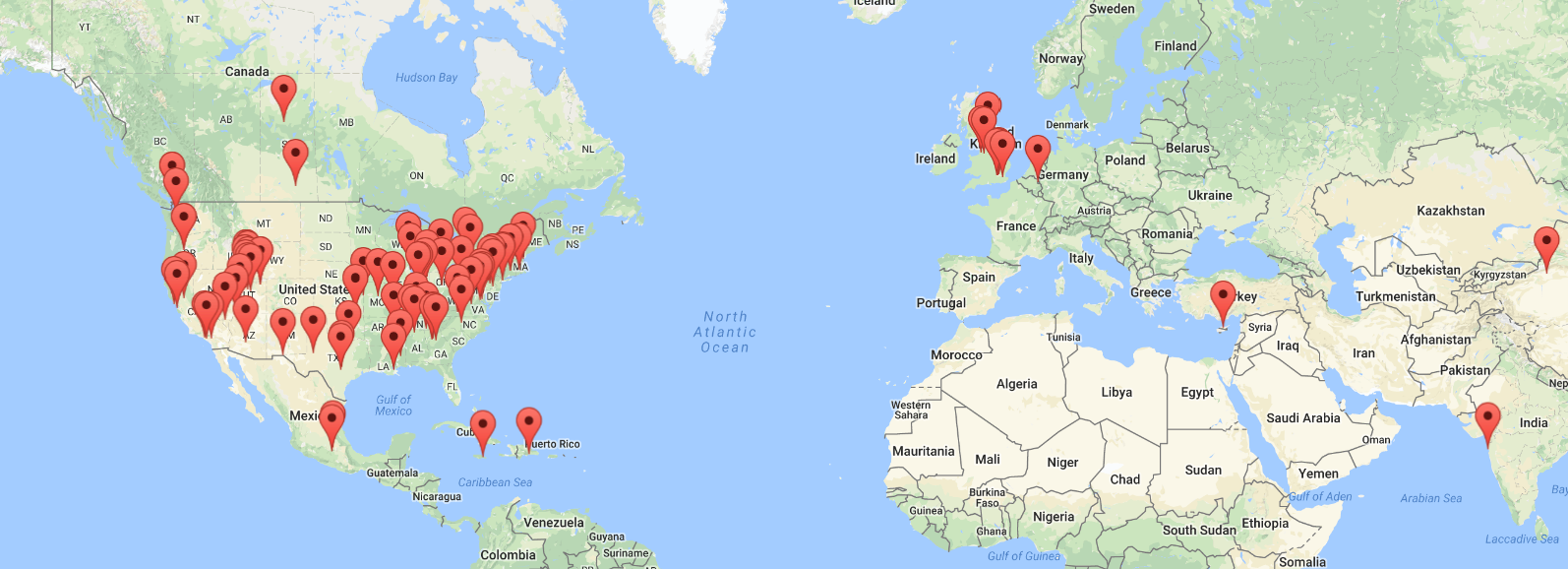
Some were validated despite unusual addresses like PLANET HIP HOP.

It turned out that this place is a gym in Irvington new jersey. But it is also a school in Missouri

It seems the geocoding was done on first come first basis Irvington came first in the list

The conclusion is what may seem unusual to us may be a place.

**Q3 C**



Most of the users of twitter profiles were in big and small towns. Assuming this as a correct result it is safe to assume that twitter users are urban and less rural. This also leads to the possibility that the users have given the big city as their location instead of the suburb.

Some profile users which appear to be rural have not given the exact address

For e.g.

amberdiamond29 has given the location as Canada. The geocoding if it detects a country name returns the center of the country in terms of latitude and longitude. Hence this user may or may not be rural. Same is the case with sunnyskystar who has given United States as the location.

The Twitter profile are mostly concentrated around the west, east coast and south of united states suggesting that these regions are more social active.

**Q3 D**

**Geocode Resolution under 4 days**

First, we need to modify the query that checks the records for geocode with the condition

{ $and: [ { user\_location: { $ne: null} }, {geocode: { $exists: false} }] }

Geocode resolution can be done under 4 days if we create an array of latitude, longitude of already found locations in a vector.

Every time we get a location we would first do a lookup of the vector. The advantage of Vector is that its size can be changed.

If the next twitter profile has the same location latitude and longitude can be fetched from this vector. Of course we have to do a substring match because sandiego ,san diego SANDIEGO refer to the same place and have the same latitude and longitude

We first need to reformat the location by applying steps like

2)make all characters small

3)remove special characters like from location.

4)Do a substring match like taking three characters at a time and checking if the location in the vector matches with it. We can use a counter of how many sequences were matched. If 3 or more sequences match for long strings and 2 or more sequences for short strings, we can take the location’s latitude and longitude

When a new location arrives search the Vector for the location.

If it does not exists add the location with latitude and longitude to the vector.

If it exists do not add the location with latitude and longitude to the array and instead take its latitude and longitude information and update the Mongo DB.

For the next location if it has a location like the stored locations by substring matching in the vector then we save 1 geocoding count for the day.

For e.g. repeated locations (London, San Diego etc.) latitude and longitude can be used in subsequent geocoding instead of using google geocode API.

**Improvement for confusing locations**

VSU/ Brooklyn/ Florida

This is confusing location for geocoding API. Notice the word Florida is a state while Brooklyn is a city. The only word that is confusing is VSU

It is difficult to create a code that will remove such words from the location, but it is not impossible. What we can do is possibly do break the string into in array of words and allow these to be geocoded and then comparing latitude and longitudes of them

The below example is rare case when all individual words returned latitude and longitude and led further to the confusion and the method of breaking strings cannot be used as it is

But for the places like the method works

Dongdong City, Zombie Land

Dongdong returns latitude and longitude

Zombie Land does not return latitude and longitude

So, if this method can validate at least 20 percent non-validated records then it’s worth implementing.

But for the below location I see no hope in geocode resolution unless we can somehow get the location where the user is most logged in

Lost in my thoughts...

outchaaa

Bx but my Heart is in Harlem.

**Data pipeline task 1 Improvement 1**

iconv -f ISO-8859-1 -t UTF-8 $1 > $2

But this is hardcoded and works only for ISO-8859-1 file formats

what if our files come from different format? The above command won’t work if files come in different format.

Additionally, the command works for only a single file. Sometimes data comes in multiple files.

Solution

Take a input directory instead of a single file.

Determine the file format, encoding using Unix command.

Output the path and the format and encoding in a text file

Read from the text files line by line. This can be done by using a custom java class file that iterates over the file paths and calls a Unix command iconv -f with the file format and encode read from the file by using process builder and generates output files in utf-8 format.

**Data pipeline task 1 Improvement 2**

If all the data files have the same columns, then the process of adding the headers can be automated and will save time of manually adding the headers. The file which was provided was small and opened easily but actual data files can be large and take time to open.

A custom java class can be created which reads the files and inserts a header line.

Limitation

If the data files have different types of columns, then we must add header manually.

**Data pipeline task 2 improvement**

./bin/import\_mongodb.sh projectA profile tsv user\_10000.tsv

Again, the filename is hardcoded, and the command will not work when the file name changes.

For multiple files we have to run the command multiple times

Instead give a file path or a directory to import that will search for tsv(UTF-8) converted files and import all these files in the Mongo DB provided they use the same collection and database.

If different collections exists then we can create subdirectories within the import directory and name the folders with a collection name. Within the folder lie all the files that go into the collection.

**Q 3 E List of all the sources used in working on this project**

<https://developers.google.com/maps/documentation/directions/get-api-key>

<https://developers.google.com/maps/documentation/geocoding/start>

<https://www.latlong.net/>

[https://developers.google.com/maps/documentation/geocoding/usage-limits](https://developers.google.com/maps/documentation/geocoding/usage-limits%20)

Tutorials in the path FA17-BL-INFO-I535-33630->Files->Tutorials->Jetstream

<https://docs.mongodb.com/manual/tutorial/query-documents/>

<https://docs.mongodb.com/manual/tutorial/query-embedded-documents/>

Transferring\_files\_To\_Jetstream.pdf

<https://docs.mongodb.com/manual/reference/program/mongoexport/>

<https://developers.google.com/chart/interactive/docs/gallery/map>

<https://developers.google.com/maps/documentation/javascript/tutorial>

<https://developers.google.com/chart/interactive/docs/reference>

<http://thoughtfaucet.com/search-twitter-by-location/>

<https://developers.google.com/maps/documentation/geocoding/start>

<https://gist.github.com/grandmanitou/8863248>

Unix commands given in the assignment

**Novel Way to select Tweets**

**{geocode: { $exists: true } }, {"user\_location" : {$regex : ".\*,.\*"}}] }'**

Observe that we have used the regular expression to find comma in a address. It is my belief that locations containing commas to separate words have more detail than without it. It will file out profiles containing only one word like name of a country. What is the use of a geo plotted profile with only country when we want to analyze something like location per city, rural vs urban social networking.