Carlos Sathler | cssathler@gmail.com
INFO-I 535 – MGMT ACCESS USE BIG DATA (Fall 2016)

PROJECT REPORT

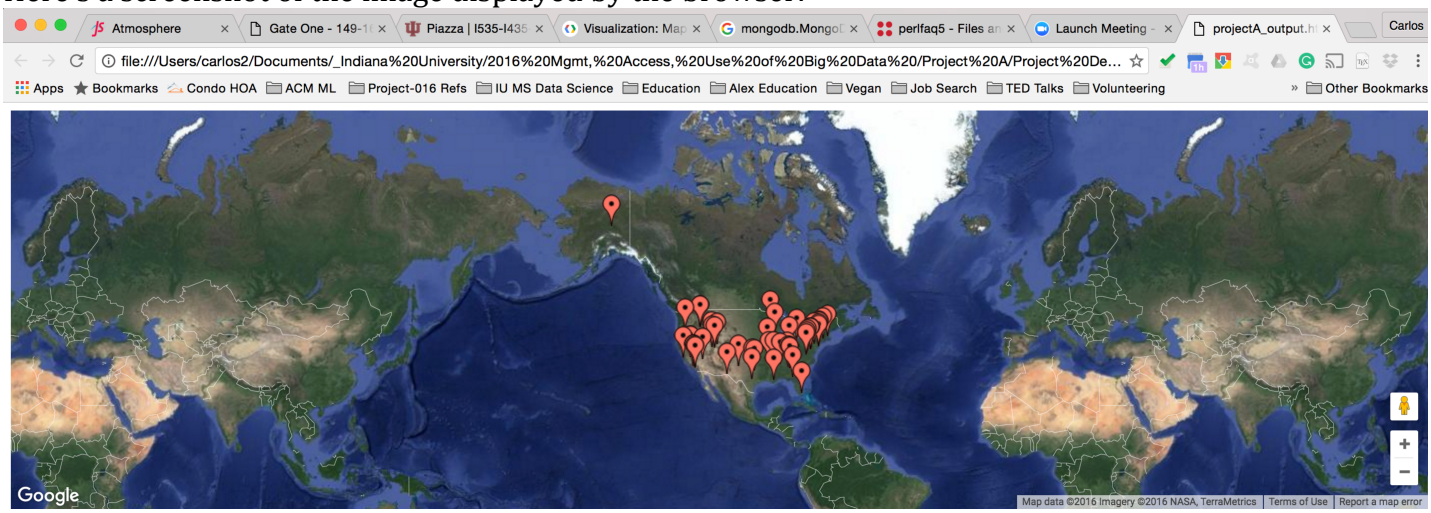
1. The exported portion of your MongoDB dataset in tab separated value form. The dataset will include only those profiles that you chose to plot.

Please see file "projectA_output.tsv" uploaded to Canvas.
Please see shell script "export.sh" in Appendix A. The script produced the csv file used to create tsv .
Shell "export.sh" uses file "fieldsFile.txt" also uploaded to Canvas.
Tsv file was manually created from csv.

2. The html file that underlies the Google map picture of your selected region.

Please see file "projectA_output.html" uploaded to Canvas.
The contents of the file are also included in Appendix B of this project report.

Here's a screenshot of the image displayed by the browser:



3.a - Lists all sources of help that you consulted

Mongo DB manual pages
http://docs.MongoDB.org/manual/reference
https://docs.mongodb.com/manual/reference/program/mongoexport/
https://docs.mongodb.com/v3.2/reference/sql-comparison/
https://docs.mongodb.com/v3.2/reference/operator/aggregation/or/

Python manual pages
https://docs.python.org/2/
https://docs.python.org/2/library/string.html
https://docs.python.org/2/tutorial/inputoutput.html

korn shell book (even though I used the bash shell for the project scripts)
Olczak, Anatole. *Korn Shell Quick Reference Guide*. ASP, Incorporated, 1998.

Carlos Sathler | cssathler@gmail.com
INFO-I 535 – MGMT ACCESS USE BIG DATA (Fall 2016)

Sed and Awk book
Dougherty, Dale, and Arnold Robbins. *Sed & awk*. " O'Reilly Media, Inc.", 1997.

Google documentation pages
https://developers.google.com/chart/interactive/docs/gallery/map
https://developers.google.com/chart/interactive/docs/gallery/map
https://developers.google.com/chart/interactive/docs/reference

3.b.i Question 1 - How many locations were you able to validate (i.e., geolocate)?

I was able to properly geolocate 7,259 records.
The details are explained below:

```
There are 9,984 records with geolocation
> db.profile.find( { geocode: { $exists: true } } ).count()
9984

But 2,725 of these records have a null geolocation
> db.profile.find( { geocode: { $exists: true, $in: [null] } } ).count()
2725

For example:
> db.profile.find( { geocode: { $exists: true, $in: [null] } } ).limit(1)
{ "_id" : ObjectId("580594ccd307063e47bb6d89"), "user_id" : 100009841, "user_name" :
"ChelseaBex", "friend_count" : 152, "follower_count" : 50, "status_count" : 394,
"favorite_count" : 0, "account_age" : "28 Dec 2009 18:05:43 GMT", "user_location" : "",
"geocode" : null }

There are 7,259 Records with geolocation that is not null
> db.profile.find( { geocode: { $exists: true, $nin: [null] } } ).count()
7259

Sample record with geolocation that is not null
> db.profile.find( { geocode: { $exists: true, $nin: [null] } } ).limit(1)
{ "_id" : ObjectId("580594ccd307063e47bb6d88"), "user_id" : 100008949, "user_name" :
"esttrellitta", "friend_count" : 264, "follower_count" : 44, "status_count" : 6853,
"favorite_count" : 0, "account_age" : "28 Dec 2009 18:01:42 GMT", "user_location" : "El
Paso,Tx.", "geocode" : { "formatted_address" : "El Paso, TX, USA", "location" : { "lat" :
31.7618778, "lng" : -106.4850217 } }
```

3.b.i Question 2 - What is the remaining number?

The number of records with no geolocation or where geolocation is null is: 2,741
The explanation is below:

```
> db.profile.find( { $or: [ { geocode: { $exists: true, $in: [null] } }, { geocode: {
$exists: false } } ] } ).count()
2741
```

3.b.i Question 3 - Give suggestions for resolving those that you were not able to resolve.

The input data for the project consists of twitter user profiles. These profiles were saved with incorrect or missing location information (for example, see below that 1,390 profile records exhibit a blank location). One way to solve the issue would be to enforce twitter users to enter a valid location when

saving their profiles.

```
> db.profile.find( { user_location: { $in: [""] } } ).count()
1390
> db.profile.find( { user_location: { $in: [""] } } ).limit(1)
{ "_id" : ObjectId("580594ccd307063e47bb6d89"), "user_id" : 100009841, "user_name" :
"ChelseaBex", "friend_count" : 152, "follower_count" : 50, "status_count" : 394,
"favorite_count" : 0, "account_age" : "28 Dec 2009 18:05:43 GMT", "user_location" : "",
"geocode" : null }
```

3.b.ii - List ways in which you think this pipeline could be improved, including other tools that could be used.

The pipeline could be improved through automation of the steps we performed manually. For example, a job (shell script) could be scheduled to download (new) twitter profiles periodically, load them to mongoDB, update geocode, export query results and refresh an html file template automatically. To illustrate the point I created a shell script to extract the 50 profile records that I used on my project. The script inserts the 50 records into an html file template using Python to create the project deliverable "projectA_output.html". Please refer to appendix C, D and E for details of "sample.sh", "sample.py" and "template.html".

APPENDIX A – Shell script "export.sh"

```
#!/bin/bash

# export all records
mongoexport --db projectA --collection profile --type=csv       \
            --query '{ $and: [ { geocode: { $exists: true, $nin: [null] } }, { "geocode.formatted_address":
/USA/ } ] }'        \
            --fieldFile fieldFile.txt --out tmp.out

# select the first 50 to match what was used for the google html file
head -51 tmp.out | tail -50 > tmp2.out

# remove all special characters from file to create a true csv file
sed -e 's/"{""formatted_address"":""/"/g; s/","location"":{""lat"":/,/; s/""lng"":/; s/}}"//' tmp2.out >
projectA_output.csv

rm tmp.out tmp2.out
```

APPENDIX B – HTML file "projectA_output.html"

```
<!DOCTYPE html>
<!--
   Source code from:
   https://developers.google.com/chart/interactive/docs/gallery/map
   Location data portion of the code obtained from class project database
-->
<html>
  <head>
  <script type="text/javascript" src="https://www.gstatic.com/charts/loader.js"></script>
  <script>
    google.charts.load('upcoming', { 'packages': ['map'] });
    google.charts.setOnLoadCallback(drawMap);

    function drawMap() {
      var data = google.visualization.arrayToDataTable([
        ['Lat',     'Long',    'Name'],
[31.7618778, -106.4850217, 'esttrellitta'],
```

```
[35.20105, -91.8318334, 'GeenaJohnson'],
[36.5481036, -121.919645, 'HovMinajJackson'],
[39.3995008, -84.5613355, 'KatieStepek'],
[42.9968085, -89.56921369999999, 'TrizZyLaCreator'],
[40.164519, -74.20764199999999, 'lustalloverme'],
[38.99066570000001, -77.026088, 'TheCameronApts'],
[37.4315734, -78.6568942, 'AmeliaSparksx3'],
[40.7127837, -74.0059413, 'PheniceArielle'],
[30.267153, -97.7430608, 'Mitchamoreagent'],
[34.1444673, -118.3905628, 'kooleyCobain'],
[34.098164, -118.2152969, 'lkshw06'],
[64.7511111, -147.3494444, 'fanaaron'],
[34.8051964, -87.67206569999999, 'raeshernell'],
[39.5562446, -111.863849, 'mattn9'],
[44.8923555, -116.0934513, 'Jmcrowe13'],
[38.0293059, -78.47667810000002, 'HonorableWorm'],
[30.8327022, -83.2784851, 'LeslieJae'],
[39.3257001, -110.964579, 'ASHNYX'],
[38.9071923, -77.0368707, 'DCCelebrity'],
[40.6781784, -73.9441579, 'PoeticEvolution'],
[46.100674, -91.1476069, 'Luckyhdyt'],
[36.778261, -119.4179324, 'TwentyFourGirl'],
[29.95106579999999, -90.0715323, 'christineanne03'],
[33.609725, -84.2873662, 'iBeRawinHoes'],
[33.8352932, -117.9145036, 'Jawaadee'],
[39.825278, -74.160556, 'JamesRamirez'],
[40.4539828, -75.81797639999999, 'gavinvz'],
[40.7714263, -111.8995768, 'hereisBAR'],
[38.9071923, -77.0368707, 'ANACOSTIASERIES'],
[44.0581728, -121.3153096, '21AtSunrise'],
[40.3001519, -109.9918536, 'ScreamAimAsh'],
[38.550153, -78.1038712, 'estVXXXMCMXCI'],
[36.1699412, -115.1398296, 'jzellis'],
[44.8923555, -116.0934513, 'audiomatt'],
[32.7766642, -96.79698789999999, 'ChicModern'],
[35.0456297, -85.3096801, 'FSRChattanooga'],
[38.9517053, -92.3340724, 'TheoKeith'],
[42.4136428, -70.9915997, 'gomezXrat3d'],
[25.9428707, -80.1233802, 'TressObsessed'],
[33.8543685, -84.9435835, 'MicahMan77'],
[33.7489954, -84.3879824, 'CarterBarbie'],
[33.7174708, -117.8311428, 'jasonrhodesfor'],
[41.76371109999999, -72.6850932, 'AyanaHarry'],
[36.778261, -119.4179324, 'predatorhunting'],
[34.2331373, -102.4107493, 'SassNSauce'],
[41.375049, -81.9081937, 'JoshDieleman'],
[33.0042512, -97.486457, 'AMAZONN'],
[35.1495343, -90.0489801, 'MizzShay76'],
[40.7942419, -73.92097799999999, 'VertigoMonster'],
 ]);

    var options = {
      showTooltip: true,
      showInfoWindow: true
    };

    var map = new google.visualization.Map(document.getElementById('chart_div'));

    map.draw(data, options);
  };
  </script>
  </head>
  <body>
    <div id="chart_div"></div>
  </body>
</html>
```

APPENDIX C – Shell script "sample.sh"

```
#!/bin/bash

echo "Loging to mongoDB to select 50 USA profiles"
mongo > sample.out <<!
```

```
DBQuery.shellBatchSize = 300
use projectA
show collections
db.profile.find( { \$and: [ { geocode: { \$exists: true, \$nin: [null] } }, { "geocode.formatted_address":
/USA/ } ] }, { "geocode.location": 1, user_name: 1, _id: 0 }).limit(50)
!

echo "Cleaning data generated by mongoDB query"
# remove rows from mongodb session that do not contain data and select 50 rows from output
grep "^{" sample.out | awk '{print "[", $13, $16, ",", $4, "]"}' > tmp.out

# format data for insertion in html file
sed -e 's/"/'\''/g; s/, ]/],/; s/ , /, /; s/\[ /\[/' tmp.out > sample.out

echo "Creating file 'projectA_output.html' with geolocations for viewing in web browser"
# insert rows into the html template for viewing in web browser
python sample.py > projectA_output.html

rm tmp.out
```

APPENDIX D – Python script "sample.py"

```
# This sample python script automates creation of html file for Project A
# It provides an example of how the pipeline for Project A could be improved

# read html template
html_template = open('template.html', 'r').read()

# geolocation data
geodata = open('sample.out', 'r').read()

print html_template.replace('TABLE_DATA_GOES_HERE', geodata)
```

APPENDIX E – HTML Template "template.html"

```
<!DOCTYPE html>
<!--
   Source code from:
   https://developers.google.com/chart/interactive/docs/gallery/map
   Location data portion of the code obtained from class project database
-->
<html>
  <head>
  <script type="text/javascript" src="https://www.gstatic.com/charts/loader.js"></script>
  <script>
    google.charts.load('upcoming', { 'packages': ['map'] });
    google.charts.setOnLoadCallback(drawMap);

    function drawMap() {
      var data = google.visualization.arrayToDataTable([
   ['Lat',     'Long',    'Name'],
TABLE_DATA_GOES_HERE ]);

    var options = {
      showTooltip: true,
      showInfoWindow: true
    };

    var map = new google.visualization.Map(document.getElementById('chart_div'));

    map.draw(data, options);
  };
  </script>
  </head>
  <body>
    <div id="chart_div"></div>
  </body>
</html>
```