DSS Capstone - Connecting Users to Build a Social Network In Yelp!

Usamah Khan

November 20, 2015

Introduction

As part of completing the Coursera Data Science Specialization, we were given a Yelp! dataset and tasked with exploring the data, determining a question and building a model or product associated with it. This report outlines the process undertaken and the methods applied.

The Dataset was supplied by Yelp from this address:

https://d396qusza40orc.cloudfront.net/dsscapstone/dataset/yelp_dataset_challenge_academic_dataset.zip

The folder contains 5 datasets of type .JSON and a PDF detailing terms of use. The datasets comprise the following information (Note - the names of the datasets as they were read in to R follow the names):

Business –> *business.data*
Check-in –> *checkin.data*
Review –> *review.data*
Tip –> *tip.data*
User –> *user.data*

After some exploratory analysis, performed in part for the Capstone quiz and for determining interesting observations in the data, the following question was isolated with the rationale below:

Can Yelp be used to create a social network by recommending friends to you based on location, check-in, ratings and other factors?

People are inclined to trust friends for recommendations and like to meet new people to go and eat with. This can be done by following those who you believe have good reviews but if there were a passive way to find recommendations like on Instagram's main feed users may find it interesting. The feedback that was recieved was positive but with one reviewer of five stating that they might not have a need for it. The question was undertaken and was found to be possible with certain limitations.

Methods and Data

Part I - Manipulating and Extracting the Data

Since the question is primarily about geographic data, the first thing to do was to determine how the data was too be isolated to a specific zone. The process for doing required isolating business ids in each of the 10 cities and link them to user data through reviews. i.e

Business Data –> Review Data –> User Data –> Isolated Users

The data for businesses was dirty and hence had to be cleaned since there were more cities and states than were actually possible. A method of K-means clustering was employed to accurately group businesses. Initial guesses were made based on longitudes and latitudes from the *geocode* package. With those extracted,

NOTE - All code chunks presented only contains relevent snippets. No cleaning and intermediate steps are shown but can all be found for further inspection over at the project repository on GitHub

```
coordinates <- as.data.table(cbind(business.data$longitude, business.data$latitude))

cities <- c('Edinburgh, UK', 'Karlsruhe, Germany', 'Montreal, Canada', 'Waterloo, Canada',
            'Pittsburgh, PA', 'Charlotte, NC', 'Urbana-Champaign, IL', 'Phoenix, AZ',
            'Las Vegas, NV', 'Madison, WI')

city.center <- geocode(cities) # Get initial guesses

city.clusters <-kmeans(coordinates, city.center) # Perform K-means

coordinates.clusters <- cbind(city.clusters$cluster, coordinates) # Get indexed cluster
                                                                  # info
```

Exploring the data, the city centers were found to be:

```
##                             lon      lat
## Edinburgh, UK          -3.193272 55.94834
## Karlsruhe, Germany      8.385830 49.00472
## Montreal, Canada      -73.587810 45.50884
## Waterloo, Canada      -80.516390 43.46680
## Pittsburgh, PA        -79.950968 40.47441
## Charlotte, NC         -80.804151 35.26002
## Urbana-Champaign, IL  -88.207270 40.11059
## Phoenix, AZ          -112.074040 33.44838
## Las Vegas, NV        -115.206969 36.30107
## Madison, WI           -89.423861 43.06956
```

Now all that was left to do was bind the clusters to the business data:

```
business.data <- cbind(coordinates.clusters$City,business.data) # Bind with business data
```

Now that all the data was seperated it was determined that the scope of the problem would be related to a single city. The city of Montreal was isolated for this analysis. Hence Montreal Business Data was subset:

```
business.data.mtl <- subset(business.data, business.data$City == "Montreal, Canada")
business.data.mtl <- business.data.mtl[,c("City", "business_id", "longitude", "latitude")]
```

With this step completed it became relatively straightforward to get an infered dataset of users based in Montreal. The code and the steps are outlined below:

```
# Subset from reviews, all businesses reviewed in Montreal

review.data.mtl <- review.data[which(review.data$business_id %in% business.data.mtl$
                                       business_id),]

# Subset from review.data.mtl, all users - Hence all users based out of Montreal

user.data.mtl <- user.data[which(user.data$user_id %in% review.data.mtl$user_id),]
```

This reduced the size of the *user.data* set down to 13861 from 366715. A more manageable size.

Determining a "home" location for users was the next step. Since no data exists for there home address and location (rightly so) an inference was made based on visited locations. The idea was to find the center of

mass of all visited restaurants in the Montreal region by a user and assume that to be where the user likely spends most of their time at businesses and restaurants. A calculation on the mean of all the longitude and latitudes of businesses visited by a user was performed.

```
for(i in i:nrow(user.data.mtl)) # Loop to isolate all businesses visited by User
  {                             # to determine center of mass
    user.id <- subset(review.user.mtl, review.user.mtl$user_id == user.data.mtl[i,7])
    mx <- mean(user.id$longitude)
    my <- mean(user.id$latitude)

    user.id.location[i,1] <- user.data.mtl[i,7]
    user.id.location[i,2] <- mx
    user.id.location[i,3] <- my
}

user.id.location <- merge(user.data.mtl[,c(6,7,5,214,215)], user.id.location, by =
                            "user_id") # Add relevent data for users
```

Part II - Building a Data product

With all the relevent data isolated, the next step was to determine how one could connect users. For this, product research was done on how, if this add-on existed, would it be used. The initial idea was to create a feed like on Instagram, however, it became clear that people prefer to have a visual resource to find friends, more like Tinder. So, a data product was built in shiny with extensive use of leaflet to give an example of how the data could be used.

The idea was plot all data for the centroids of user activity, and from there give users an option to find other users within a specific radius.

To do this a UI.R and Server.R were written to include an interactive map with reactive inputs to users and distance. When clicked, the markers display popups with the following information: Name, Average Stars, Number of Reviews and Number of Fans. Enough information to be of interest to a user wanting to connnect.

The second aspect of the methods was to allow further subsetting to the data based on checking all mutually reviewed businesses and star ratings. This was to be achieved by calling a user and isolating businesses reviewed and then referencing that with all users in the specific zone of reference. The app would then drop other users with little relevance to the user in question.

Results

The purpose of this Capstone was to build a model, statistical or predictive, yet since the final course dealt with Developing Data Products, a lot of time was taken to build a tangible, dynamic environment for exploring the data in addition to a model. As a result, the result of the analysis came in the form of an app that can be loaded here:

```
runGitHub( "DSS-Capstone", "Usamahk")
```

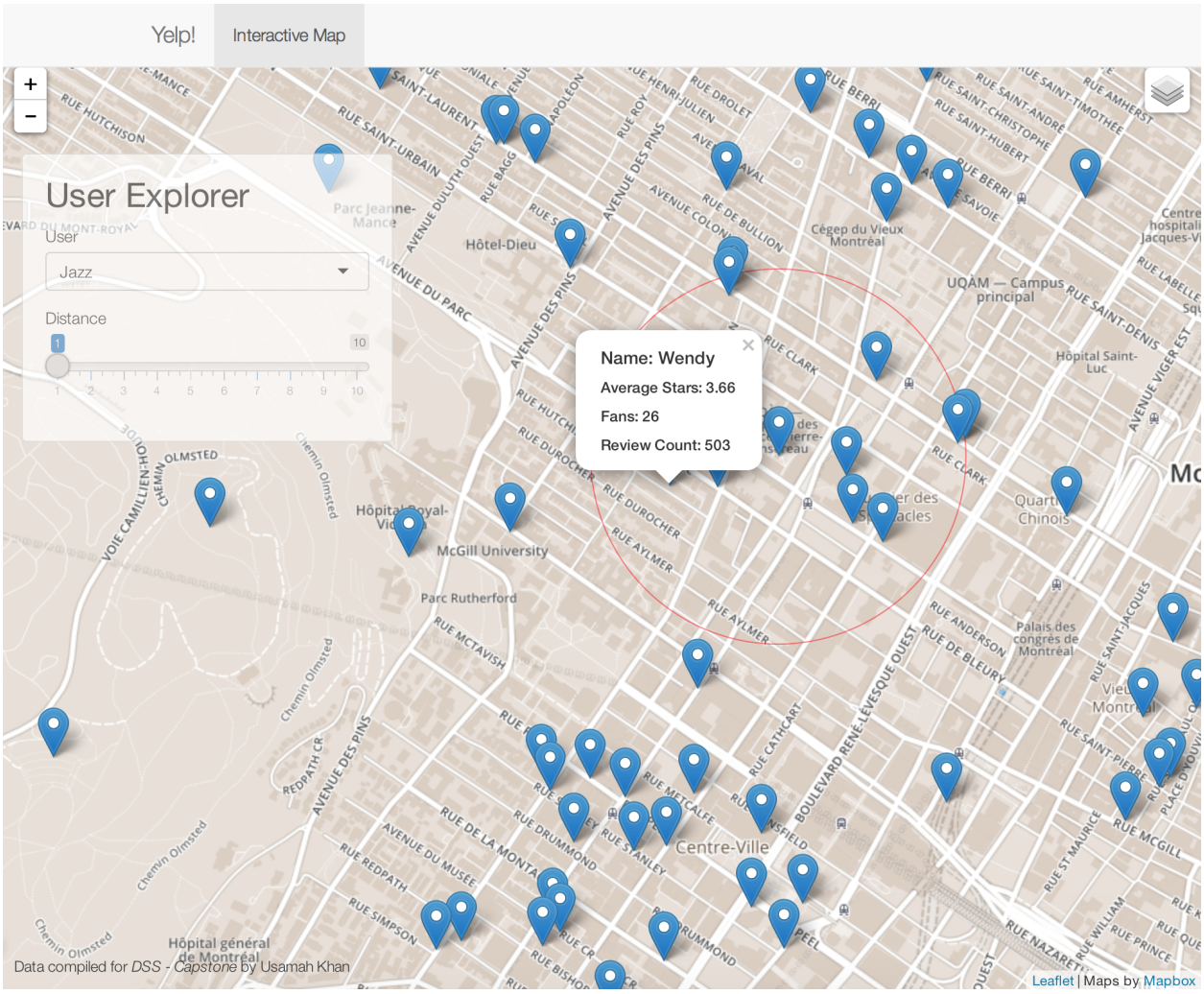
It was useful and showed a possible solution to the initial question laid out, however, with some caveats. First, to understand how the app works, the screenshot below shows an example of the usage and the zone of interest.The final data frame that was cleaned and used as input for all information on the map is in the following format:

Note - this is a test set used for display:

```
##                  user_id
## 1 __0s_Oby_akeEKnkgTrz4w
## 2 __3cGi5oLgTK4EkFwJETSg
## 3 __53csc1JFS0I0lpxZXn_g
## 4 _-pw6eL7Vgku-DssJL_g-A
## 5 _-RoUSf53-Nw9qPgeyLpng
## 6 _-wI5M_GcBLFhsExd474fw
```

```
##      name review_count fans average_stars longitude latitude
## 1  Roland           18    0          3.67 -73.56573 45.50711
## 2 Michael           34    0          3.09 -73.56316 45.50948
## 3     Ali            6    0           3.6 -73.57141 45.50842
## 4    Neil           40    2          3.65 -73.50997 45.53848
## 5  Nicole            2    0             4 -73.57439 45.52221
## 6   Aaron           10    0           4.4 -73.58080 45.52724
```

Below is an example of a screenshot of the app. A "User Explorer" overlay gives the option of choosing a user and based on that user, centering a circle of a variable radius. Once the zone is yet, the user can explore the region and other users inside by clicking on the marker which initiates a popup with relevent information pertaining to the user.



Discussion

In the end, using Shiny and Leaflet proved to be a useful resource in terms of getting an example of the potential of the data use. The question was most certainly answerable but there were limitations due to both the methods presented and assumptions that were made.

Montreal was chosen since it was the environment where I had the most knowledge of so outliers and other issues could be spotted by visual inspection. The assumption to take the location of a user as their center of mass was not necessarily correct. That was apparent by looking at the map and realising that some users were located in zones where they could not possibly live (a forest and lake to name a few) and were also present in areas that are not necessarily residential. It becomes a good inference for users who have many reviews as it points to at about the middle of where they are most likely to be, but confining the search within bounds of natural and realistic locations to live would be better. This would require a slightly more intimate knowledge of the region and data, hence it may be infeasible.

The second issue was that building a script that dropped users based on mutual reviews did not turn out as well. Because of the limitations of the software, and indeed personal knowledge of scripting, the code ran very slow and was prone to frequent crashing and freezing. This was due to trying to create a table for a single user and do this with every user whilst counting and storing shared mutual information. It became a problem when dealing with multiple users with >200 reviews. Hence it was determined that a number of factors could make the app better:

- Another scripting language could speed up the process
- Altering the method
- Limiting the scope

These were options that were discussed with other members of the course and online. With certain changes that were not feasible during the time frame of the analysis, a potential for a more accurate solution exists.