Yelp Dataset Review

John Antony

March 4, 2017

DOES LOCATION ATTRACT HIGHER RATING

```
#read the data and load the libraries
library(readr)
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(stringr)
library(jsonlite)

business_file_path = "/Users/JohnAntony/Desktop/Main/Applications/R/MachineLearning/Yelp/yelp_dataset_challenge_r
ound9/yelp_academic_dataset_business.json"

business_lines <- read_lines(business_file_path, progress = FALSE)

business_combined <- str_c("[", str_c(business_lines, collapse = ", "), "]")

business <- fromJSON(business_combined) %>%
  flatten() %>%
  tbl_df()
```

For the decision of a business at a location, lets keep the area to the Las Vegas city.

```
#filter out NV from the data
business_NV <- subset(business, business$state == 'NV')
city_NV = data.frame(unique(business_NV$city))#find all unique cities in NV to see different variations of Las Ve
gas

#convert all occurences of varying Las Vegas to a standard
business_NV <- lapply(business_NV, function(x) {gsub("South Las Vegas", "Las Vegas", x)})
business_NV <- lapply(business_NV, function(x) {gsub("North Las Vegas", "Las Vegas", x)})
business_NV <- lapply(business_NV, function(x) {gsub("Las Vegas East", "Las Vegas", x)})
business_NV <- lapply(business_NV, function(x) {gsub("N. Las Vegas", "Las Vegas", x)})
business_NV <- lapply(business_NV, function(x) {gsub("West Las Vegas", "Las Vegas", x)})
business_NV <- lapply(business_NV, function(x) {gsub("N W Las Vegas", "Las Vegas", x)})
business_NV <- lapply(business_NV, function(x) {gsub("N E Las Vegas", "Las Vegas", x)})
business_NV <- lapply(business_NV, function(x) {gsub("las vegas", "Las Vegas", x)})
business_NV <- lapply(business_NV, function(x) {gsub("Las  Vegas", "Las Vegas", x)})
business_NV <- lapply(business_NV, function(x) {gsub("Las Vegas NV", "Las Vegas", x)})
business_NV <- lapply(business_NV, function(x) {gsub("Las vegas", "Las Vegas", x)})
business_NV <- lapply(business_NV, function(x) {gsub("Las Vegass", "Las Vegas", x)})
business_NV <- lapply(business_NV, function(x) {gsub("LasVegas", "Las Vegas", x)})
business_NV <- lapply(business_NV, function(x) {gsub("LasVegas", "Las Vegas", x)})
business_NV <- lapply(business_NV, function(x) {gsub("LasVegas", "Las Vegas", x)})

business_LV <- subset(business, business$city == 'Las Vegas')
```

A general notion is, for better ratings, it is preferred to open a business a a popular location. So we find the most popular location in the most popular neghborhood.

```
# count the frequency for each neighborhood to find the most popular neighbourhood
temp <- as.vector(business_LV['neighborhood']])
n_freq = as.data.frame(table(unlist(temp)))

#count the frequncy of coordinate to find a single popular coordinate. This coordinate will be used as the center
 and find distance from this point

library(dplyr)
coord_westside <- subset(business_LV, business$neighborhood == 'Westside')
coord_westside <- dplyr::select(coord_westside, business_id,longitude, latitude)
coord_westside <- coord_westside%>%tidyr::unite(coordinates, longitude, latitude, sep = ",")
temp_coord <- as.vector(coord_westside['coordinates']])
coord_freq = as.data.frame(table(unlist(temp_coord))) #-115.2659777,36.1988542(long,lati): 4 entries at this coor
dinate
```

Now, we calculate the distance of every business from this point.

```
#finding distance
library(geosphere)
```

```
## Loading required package: sp
```

```
lv_dist_matrix <- dplyr::select(business_LV, business_id,name, neighborhood, stars, review_count, longitude, lati
tude, categories)

distance <- function(longitude, latitude)
{
  distance= distm (c(-115.1963, 36.17226), c(longitude, latitude), fun = distHaversine)
  return(distance)
}

lv_dist_matrix['distance']=0
for(i in 1:nrow(lv_dist_matrix)){
  lv_dist_matrix$distance[i] = distance(lv_dist_matrix$longitude[i],lv_dist_matrix$latitude[i])
}
```

```
lv_dist_r10 <- subset(lv_dist_matrix, lv_dist_matrix$distance < 15000)

library(ggmap)
```

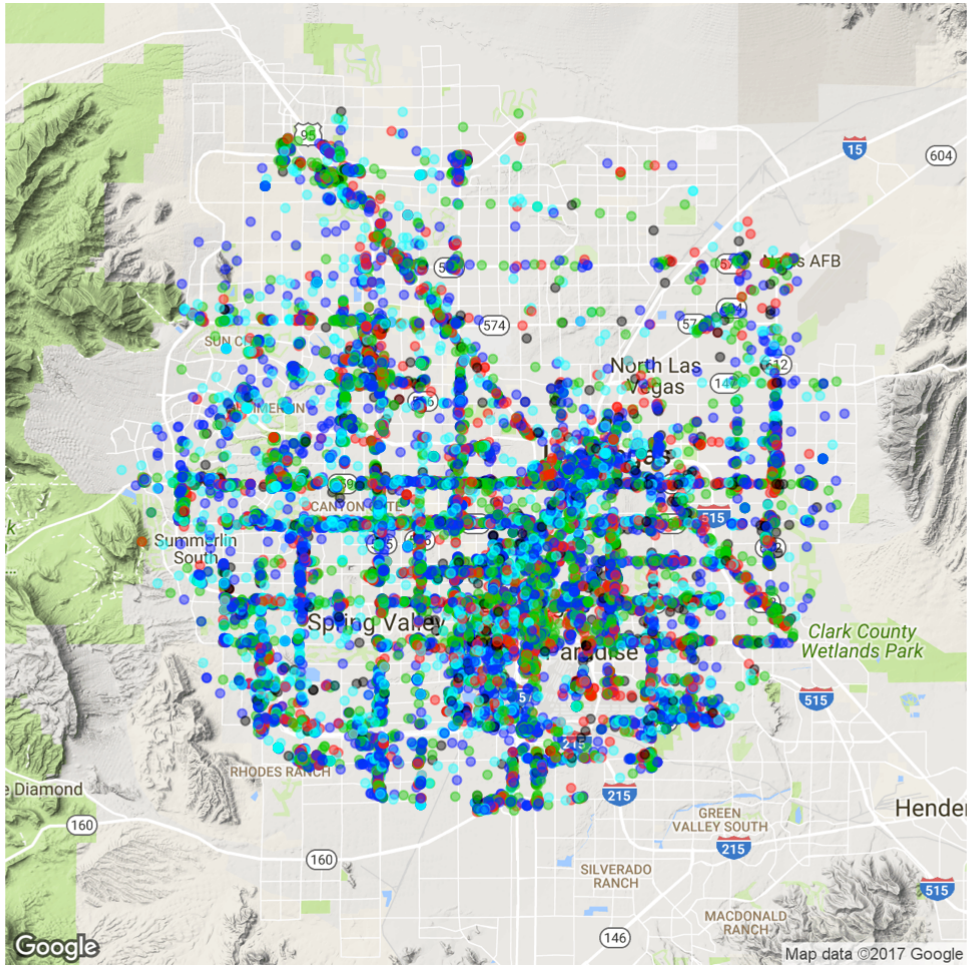
```
## Loading required package: ggplot2
```

```
#map_center <- get_map(location = c(lon = -115.2659777, lat = 36.1988542), zoom = 13, scale = "auto")
map_plot <- qmap(location = c(lon = -115.1949777, lat = 36.1588542), zoom = 11, scale = "auto", legend = "right")
```

```
## Map from URL : http://maps.googleapis.com/maps/api/staticmap?center=36.158854,-115.194978&zoom=11&size=640x640
&scale=2&maptype=terrain&language=en-EN&sensor=false
```

```
## Warning: `panel.margin` is deprecated. Please use `panel.spacing` property
## instead
```

```
map_plot+
  geom_point(aes(x = lv_dist_r10$longitude, y = lv_dist_r10$latitude, size=lv_dist_r10$stars), data =
lv_dist_r10, , color= lv_dist_r10$stars, alpha = .5, size = 2, pch = 20)+
  scale_color_manual(lv_dist_r10$stars)
```



Once we have all the distances and the star rating of all the businesses, we try to corelate them. Following are the results.

```
#is there any corelation between a central location and the ratings
cor(lv_dist_r10$distance, lv_dist_r10$stars)
```

```
## [1] 0.007405653
```

```
#NOT CORELATED

cor(lv_dist_r10$stars, lv_dist_r10$review_count)
```

```
## [1] 0.01409579
```

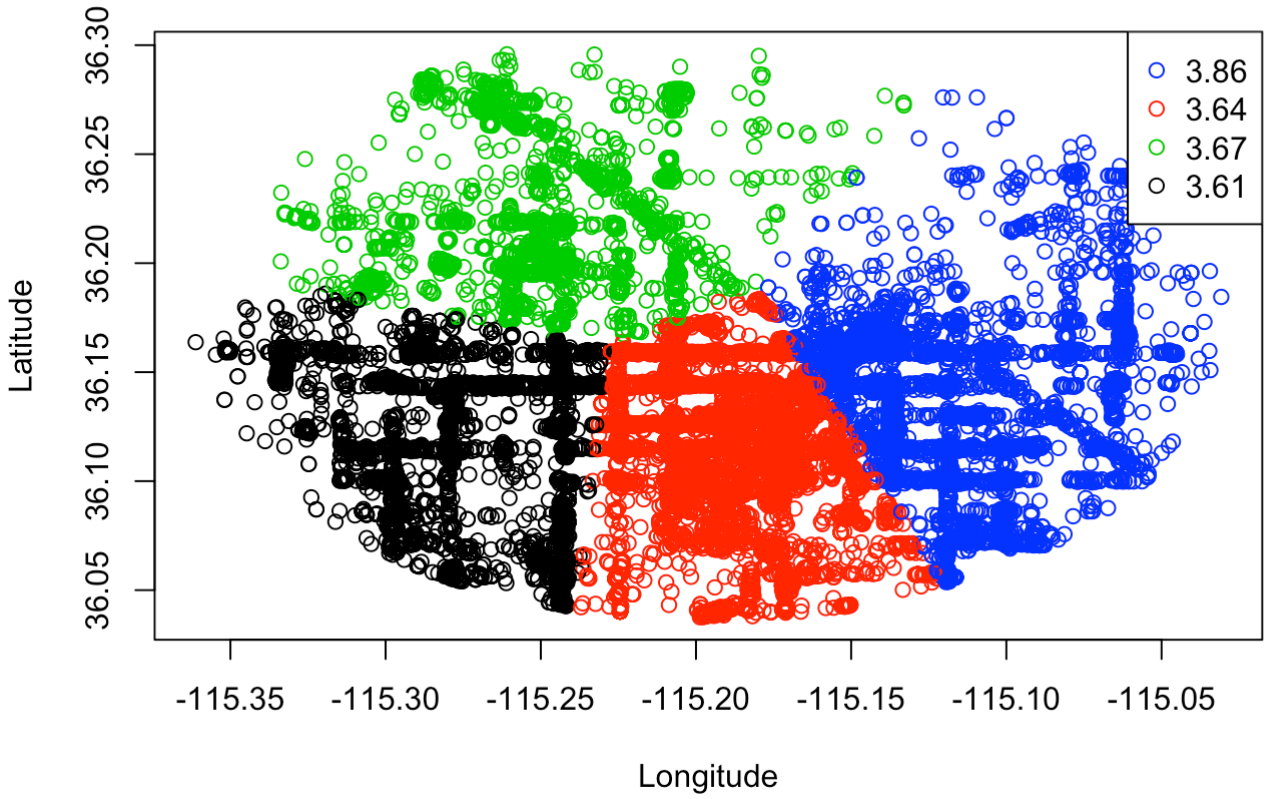
```
#NOT CORELATED
```

Once we figure out that location is irrelevant to the star rating of a business, we may try to find the competition among various parts of the city. So we divide the city in clusters and take a look at the average rating of each cluster. The cluster plot gives us a basic idea of competition in each cluster.

```
#CLUSTERING LOCATIONS
km <- kmeans(cbind(lv_dist_r10$latitude, lv_dist_r10$longitude), centers = 4)

#attach cluster number to each point to calculate cluster mean rating
lv_dist_r10 <- cbind(lv_dist_r10, clust_num = km$cluster)
#calculate mean by cluster
clust_means = aggregate( lv_dist_r10$stars~lv_dist_r10$clust_num, lv_dist_r10, mean )
clust_means$`lv_dist_r10$stars` = round(clust_means$`lv_dist_r10$stars`,digits=2)
clust_means = cbind(clust_means, centers = km$centers)

plot(lv_dist_r10$longitude, lv_dist_r10$latitude, col = km$cluster, xlab = "Longitude", ylab = "Latitude")
plot_legend = clust_means$`lv_dist_r10$stars`
legend("topright", legend = plot_legend, col = km$cluster, pch = 1)
```


Latitude
36.30
36.25
36.20
36.15
36.10
36.05
-115.35
-115.30
-115.25
-115.20
-115.15
-115.10
-115.05
Longitude
3.86
3.64
3.67
3.61