Super User Distinguish and Recommendation System based on Massive YELP data

Canyang Liu [*] Jia Liu [†] Wenjia Xie[‡]

April 28, 2018

Abstract

This article aimed to recommend superusers (influential users) dig from the YELP dataset to the relative restaurants and provided suggestions to users who want to become superusers. Data mining method and sentiment analysis were applied to find the superusers, and multiple cluster analysis were used to accurately recommend the superusers to the restaurants. Multiple regression models were fitted to analyze the relationship between influential strength and users behavior. Using the methods above, this article provided each restaurant with its unique top ten superusers. Also, this article found the most relevant users behavior to strengthen their influence.

[*]Department of Statistics. Student ID: 9079192556 Email: cliu523@wisc.edu
[†]Department of Statistics. Student ID: 9077012988 Email: jliu647@wisc.edu
[‡]Department of Statistics. Student ID: 9079192507 Email: wxie42@wisc.edu

1 Introduction

Nowadays, together with the explosively increasing number of YELP's users, more and more people tend to search for restaurants on the advertising platform like YELP. Thus, it's reasonable to consider YELP as a bridge between the merchants and the customers. This article had two main purposes.

The first purpose was recommending the superusers to restaurants. The review for restaurants is a very important factor to attract customers. It can provide evaluation from multiple angles. An excellent review may attract hundreds of followers to the restaurants, therefore, it is valuable to find the superusers who were influential enough to promote the performance of a business.

这个空行是不是不该有?

The second purpose was giving users suggestions to become a superuser.

The data was from YELP Dataset. It included 6 datasets, "business", "review", "user", "checkin", "tip" and "photos". Here we mainly used "business", "review" and "user".

2 Recommend Superuser

To recommend the superusers to restaurants, the first step was finding superusers. After that, we need to classify the restaurants. Recommending one superuser to similar restaurants based on the type of restaurants he or she had visited.

2.1 Finding Superusers

the, a 感觉可以去掉

To find the superusers, the idea was finding the influential users first. How to define an influential user? If a user-A commented on a restaurant, his or her friends also commented on this restaurant after A's comment's date, then we defined A to be an influential user, and his or her friends to be followers. The influential users who had more than a certain number of followers were superusers.

if从句应该用现在时

if后面不需要then

直接写100吧

2.1.1 Finding Influential user

This article studied 1,326,101 users and 32,235 American restaurants out of 174,567 businesses in total.

all the datasets were joint first. 祈使句不能用过去时

这两个the都不需要我觉得

Joined all the dataset first. Dataset "user" and dataset "review" can be joined by the user id, dataset "business" and dataset "review" can be joined by the business id. Ordered the dataset "review by the business id, a list of reviews could be obtained. In each list, checked whether two reviewers are friends, and treated the early reviews owner to be the influential user who had influence on the later reviews owner. Finally, there were 70,861 influential users who influenced at least one of his or her friends.

少了引号

By ordering

加主语

2.1.2 Finding Superusers

If an influential user had more than or equal to 100 followers, $\overset{\text{typo}}{\text{than he}}$ or she would be defined as a superuser. By this way, there were 2,418 superusers selected from all 79,861 influential users in total.

~~In case of a special situation that~~ if a person left a negative comment on a restaurant, his or her friends still came to this restaurant, $\overset{\text{it is}}{\text{thats}}$ irrational to say those "followers 少引号 少引号 came after this negative comment, more likely, a real celebrity left a positive comment before that negative comment, and by coincidence the owners of these two opposite reviews shared same followers.

Therefore, sentiment analysis was applied to weed out the followers affected by the negative reviews.

In the former analysis, 2418 superusers were selected, then screened out 211533 reviews from the superusers, calculated the number of followers influenced by each review respectively. After those reviews being separated into word pieces, where each word piece implied a mix of different dimensions of sentiment, the strength of 17 dimensions of sentiment from all the 211533 reviews was obtained. This sentiment was used to predict the number of followers that should be influenced by each review. In fact, if a review contained several negative sentiments, it might be less likely to attract followers. Two regressions were applied on the 17 dimensions of sentiment.

For each review, comparing with the real number of followers being attracted and the fitted values in both model, if the fitted value was significantly smaller than the true value, treat this review as a "fake influential review, 少引号 thus the number of followers being attracted by this review could be reduced. By this criterion, 2408 out of 2418 superusers were selected as the result.

2.2 Classify Restaurants

To find appropriate super users for a specific restaurant, doing cluster analysis on restaurants.

There were about 600 restaurant categories. Considering categories which contained more than 30 restaurants, it gave about 152 categories. Trying PCA to reduce dimension, however, the result was not good, since the first principle component can only interpret 2.2% of the data's volatility. The other method was using k mode cluster, it was applicative and classify the restaurant into 14 clusters.

Next, embed the category cluster result into the original data set exclude all 152 category variables. Then conducted k prototype cluster to all restaurants' features expect longitude and latitude, got the result in 10 attribute classes.

Finally, clustered the restaurants again only based on location variables and divided them into 7 area groups.

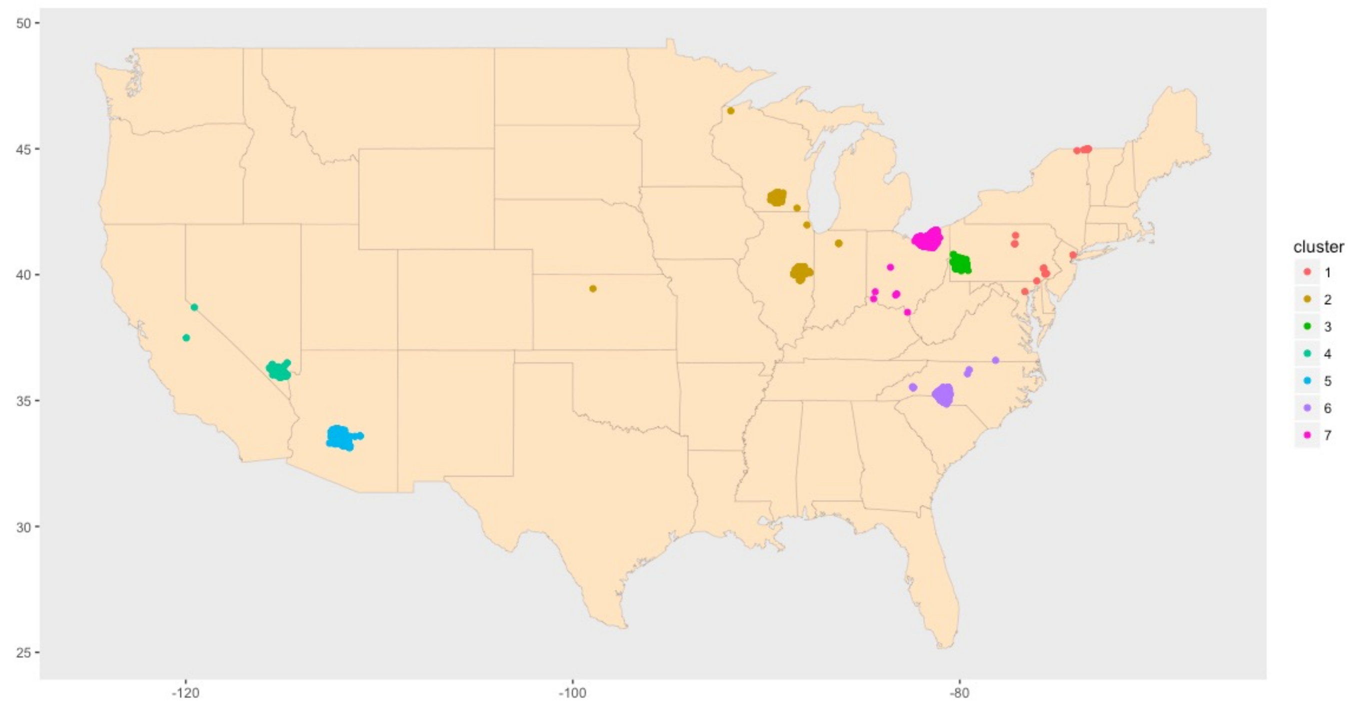


Figure 1: Restaurants Clusters in 7 area groups

Considering those restaurants can firstly be divided into 10 groups, and then the location divided each group into 7 groups. The cluster process gave us 70 clusters in total.

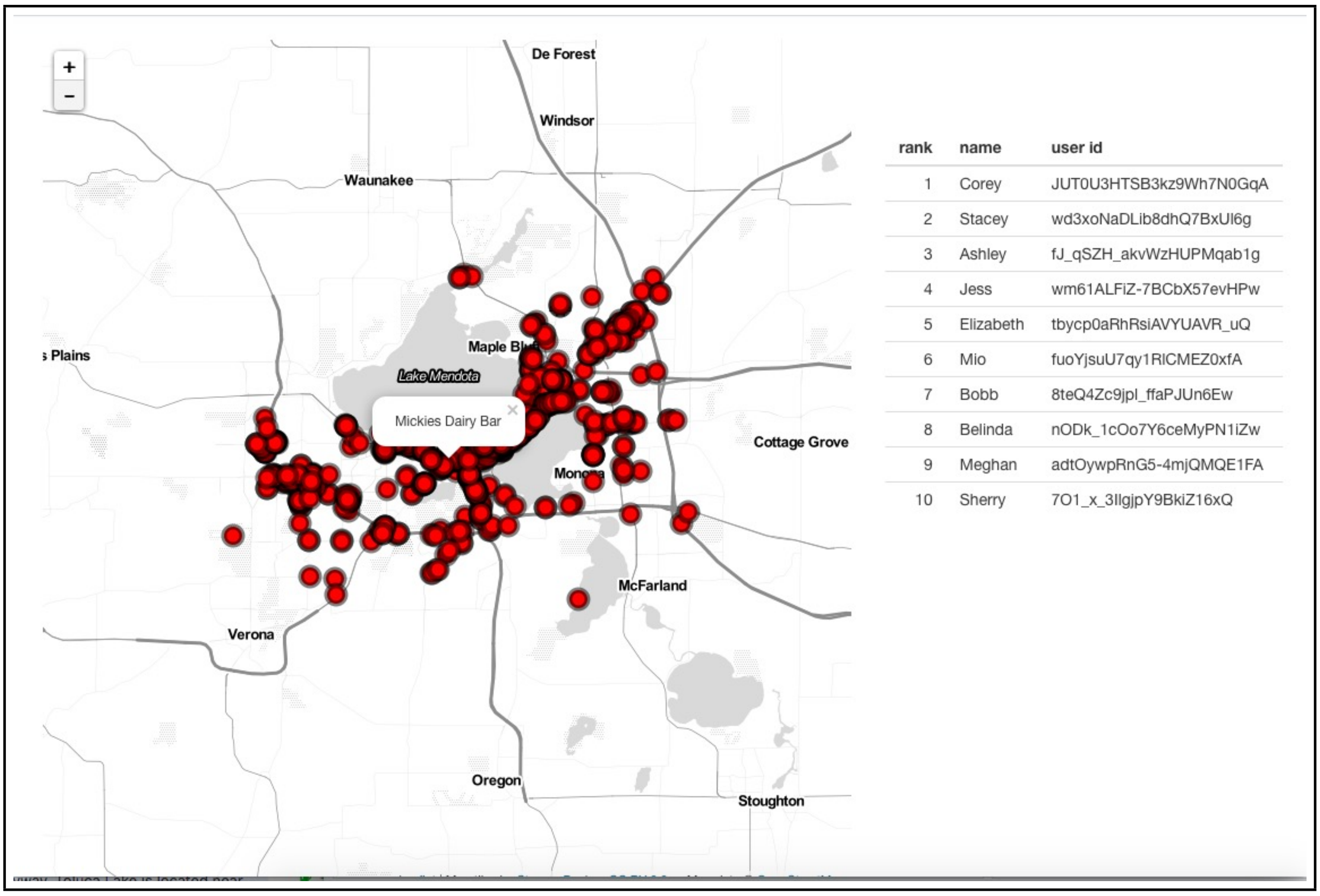
2.3 Recommend Superuser to Restaurants

First, it was essential to find the restaurants which were commented by each superuser. Then, in each of the 70 clusters, there should be a ranking of superusers by the numbers of restaurants which were commented by each superuser. Recommended the top 10 superuser to this cluster of restaurants.

Also, it was effective if we chose an exact place like a city or a county. Under this circumstance, there was no need to consider the location, divided the city or county into 10 groups was enough.

For example, focused on restaurants in Madison. There were 1,026 restaurants in Madison, after the classify, it became 10 groups each contains 217, 16, 114, 108, 101, 21, 195, 196, 58 restaurants. After finding the top 10 superuser for each group, the recommendation system was shape up.

Here is the screenshot of shiny app. If one restaurant is selected, the point will show its name and on the right side, there shall be the recommend superuser for this restaurant.



rank	name	user id
1	Corey	JUT0U3HTSB3kz9Wh7N0GqA
2	Stacey	wd3xoNaDLib8dhQ7BxUl6g
3	Ashley	fJ_qSZH_akvWzHUPMqab1g
4	Jess	wm61ALFiZ-7BCbX57evHPw
5	Elizabeth	tbycp0aRhRsiAVYUAVR_uQ
6	Mio	fuoYjsuU7qy1RICMEZ0xfA
7	Bobb	8teQ4Zc9jpl_ffaPJUn6Ew
8	Belinda	nODk_1cOo7Y6ceMyPN1iZw
9	Meghan	adtOywpRnG5-4mjQMQE1FA
10	Sherry	7O1_x_3IlgjpY9BkiZ16xQ

Figure 2: Recommendation for Madison Restaurants

3 How to Become a Superuser

To offer advice to users who are willing to become a superuser, three regression models were fitted. These models can find out the main factors that may enhance users' performance. Due to the model assumptions not being held, a logistic regression or negative binomial regression may be better. In addition, an ordinary regression applied with BOX-COX transformation was provided as a contrast. By analyzing dataset "user", there were 17 numerical variables to be selected initially.

3.1 The first method: Logistic Regression Model

Under this model, the dependent variable (y) was an indicator function of whether a user is a superuser. If he or she is a superuser, set y to be 1, otherwise set y to be 0. After stepwise variable selection, 7 variables were selected. Here is the result of the model. (The detail is in Appendix Output 1)

$$\begin{aligned}10^3 \times \mathbf{1}_{Is\ Superuser} =& -6.3 \times 10^3 - 1.2\ fans + 2.2\ compliment\ writer \\ & - 7.8\ compliment\ profile + 1.4\ compliment\ funny \\ & - 1.9\ compliment\ plain + 1.5\ friend\ number + 3.1\ review\ count\end{aligned} \tag{1}$$

3.2 The second method: Negative-Binomial Regression Model

Since a majority of users had zero followers, the influence number (y) had an approximate geometric distribution, which is a special case of negative binomial distribution. Therefore, a negative binomial regression model was fitted to overcome the unsatisfied model assumption. Same stepwise variable selection was used, 13 variables were selected. Here is the result of the model. (The detail is in Appendix Output 2)

$$\begin{aligned}10^3 \times Influence\ number =& -3.2 \times 10^3 + 0.4\ compliment\ photos + 0.6\ compliment\ funny \\ & - 0.4\ compliment\ hot - 2.7\ compliment\ writer \\ & - 0.8\ compliment\ plain - 17.1\ compliment\ list \\ & + 16.9\ compliment\ more + 5.3 \times 10^{-2}\ useful - 7.0 \times 10^{-2}\ cool \\ & - 3.5\ fans + 1.2 \times 10^2\ average\ stars + 2.6\ friend\ number \\ & + 6.7\ review\ count\end{aligned} \tag{2}$$

3.3 The third method:OLS Model using Box-Cox Transformation

By contrast, an ordinary least square regression model was fitted applied with BOX-COX transformation to overcome the non-normal distribution of the dependent variable existed in the variables. Doing log transformation on the dependent variable and using the same stepwise variable selection, 12 variables were selected. Finally drew the following result. (The detail is in Appendix Output 3)

$$\begin{aligned}10^4 \times log(Influence\ number + 1) =& 9.4 - 8.8\ compliment\ list + 7.6\ compliment\ funny \\ & - 6.5\ compliment\ plain + 9.9\ compliment\ more \\ & - 1.5\ compliment\ hot - 6.9\ compliment\ profile \\ & + 17.3\ review\ count + 7.8\ friendnumber + 6.2\ fans \\ & + 0.3\ funny - 0.2\ cool + 66.1\ average\ stars\end{aligned} \tag{3}$$

3.4 Model Comparison and Conclusions

Though the variables selected from those three models were partly different, the variables appeared in all models were the significant factors that influenced the users behavior. Users should use more funny words in their review, instead of plain words. More reviews should be written to enlarge the possibility of being read by others. More new friends should be made to strengthen the association in the network.

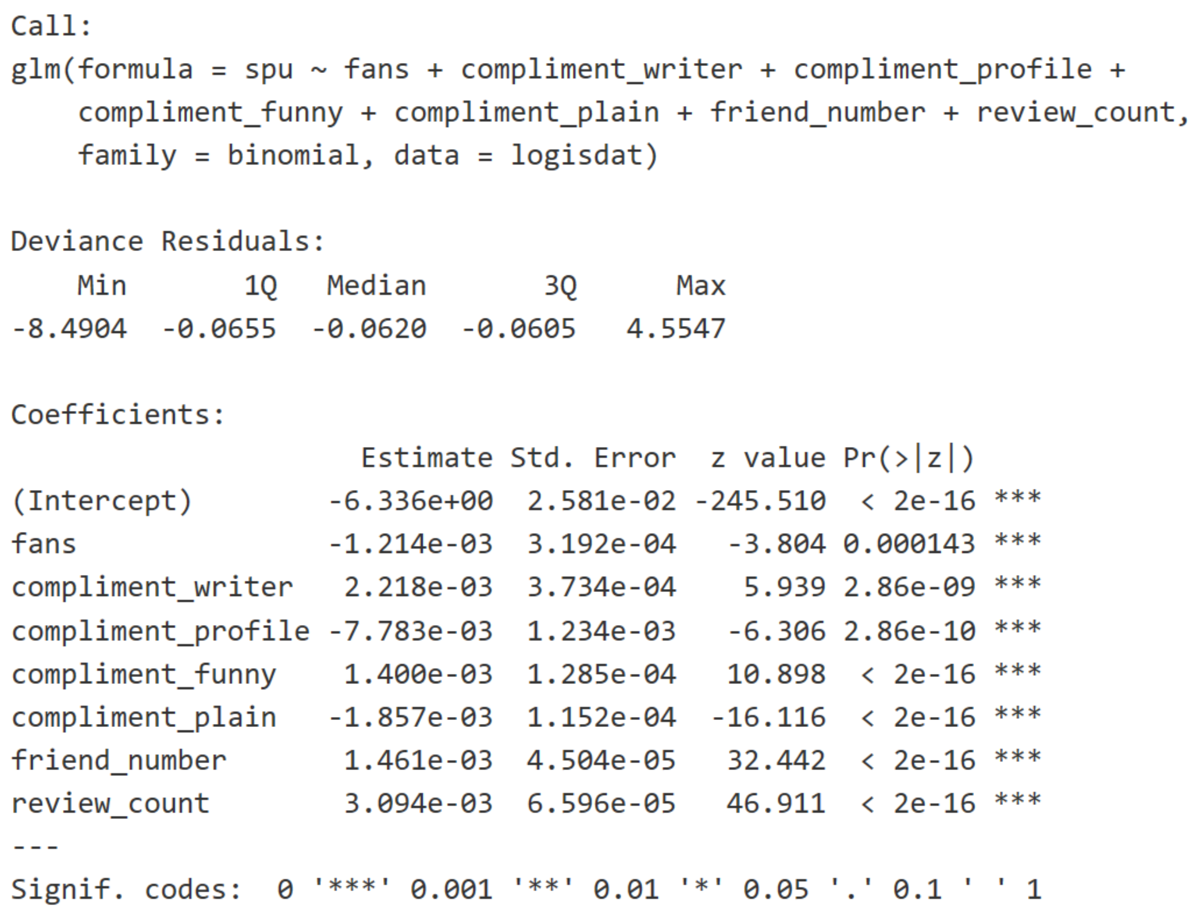
4 Conclusions

This article reached two goals, recommending superusers to restaurants and helping users to become superusers for the propose of promoting the performance of restaurants and users.

The first part of the article dig out the influential users from all 1.3 million users by the number of friends who came to the same restaurants after these influential users review. An influential user who had more than 100 followers was temporarily defined as a “superuser. Then, a sentiment analysis was applied to double check the accuracy of superusers. Based on the influence number and the cluster of the restaurants, we recommended the superusers to the restaurants.

The next part of this article focused on the relationship between the followers' number and users behavior, which gave suggestions to become a superuser. The content of reviews as well as the activeness in the network were both important factors that could help attract more followers.

Appendix: Outputs

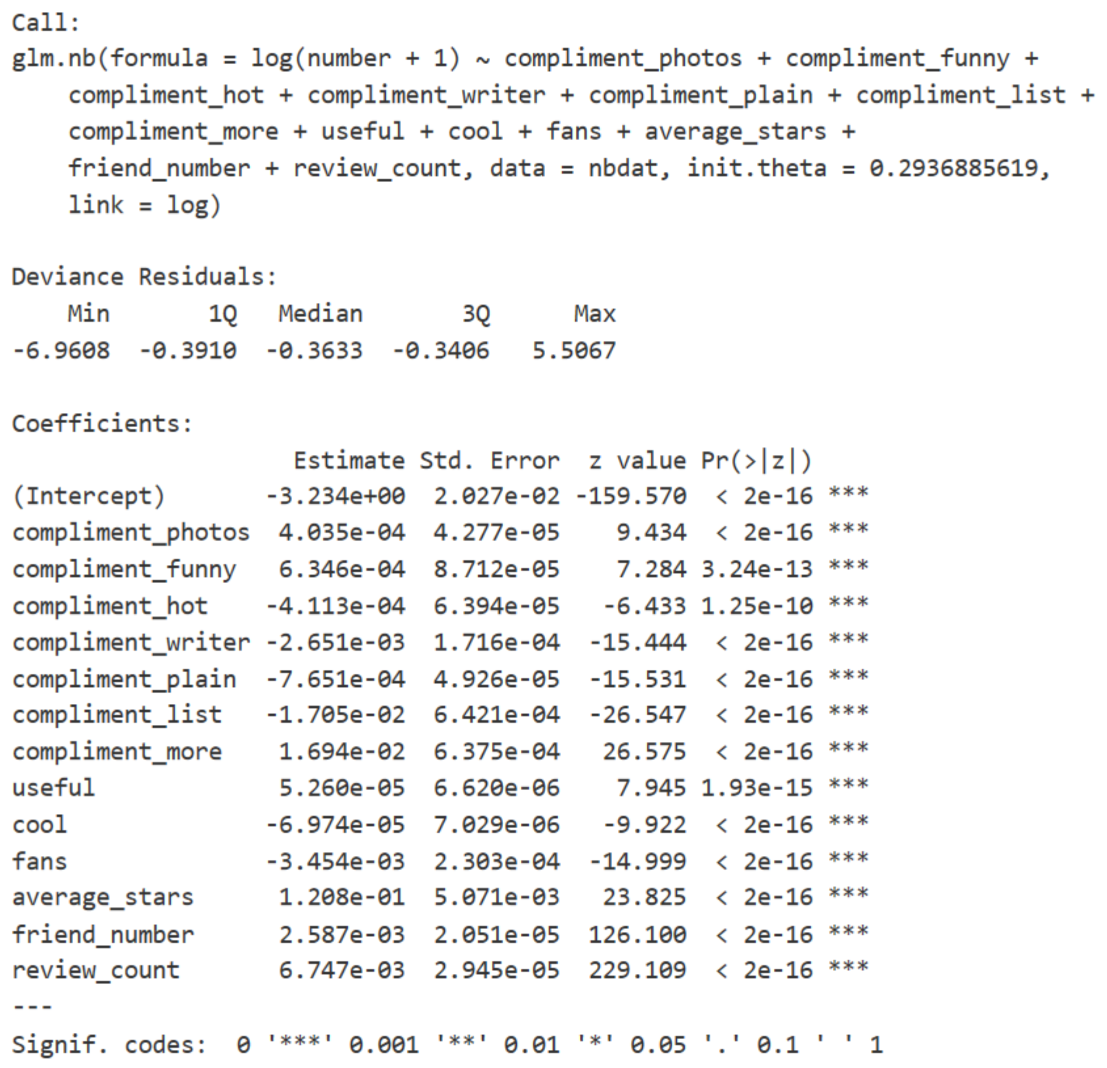


```
Call:
glm(formula = spu ~ fans + compliment_writer + compliment_profile +
    compliment_funny + compliment_plain + friend_number + review_count,
    family = binomial, data = logisdat)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-8.4904  -0.0655  -0.0620  -0.0605   4.5547

Coefficients:
                     Estimate Std. Error  z value Pr(>|z|)
(Intercept)        -6.336e+00  2.581e-02 -245.510  < 2e-16 ***
fans               -1.214e-03  3.192e-04   -3.804 0.000143 ***
compliment_writer   2.218e-03  3.734e-04    5.939 2.86e-09 ***
compliment_profile -7.783e-03  1.234e-03   -6.306 2.86e-10 ***
compliment_funny    1.400e-03  1.285e-04   10.898  < 2e-16 ***
compliment_plain   -1.857e-03  1.152e-04  -16.116  < 2e-16 ***
friend_number       1.461e-03  4.504e-05   32.442  < 2e-16 ***
review_count        3.094e-03  6.596e-05   46.911  < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Output 1 : Summary for Logistic Regression Model

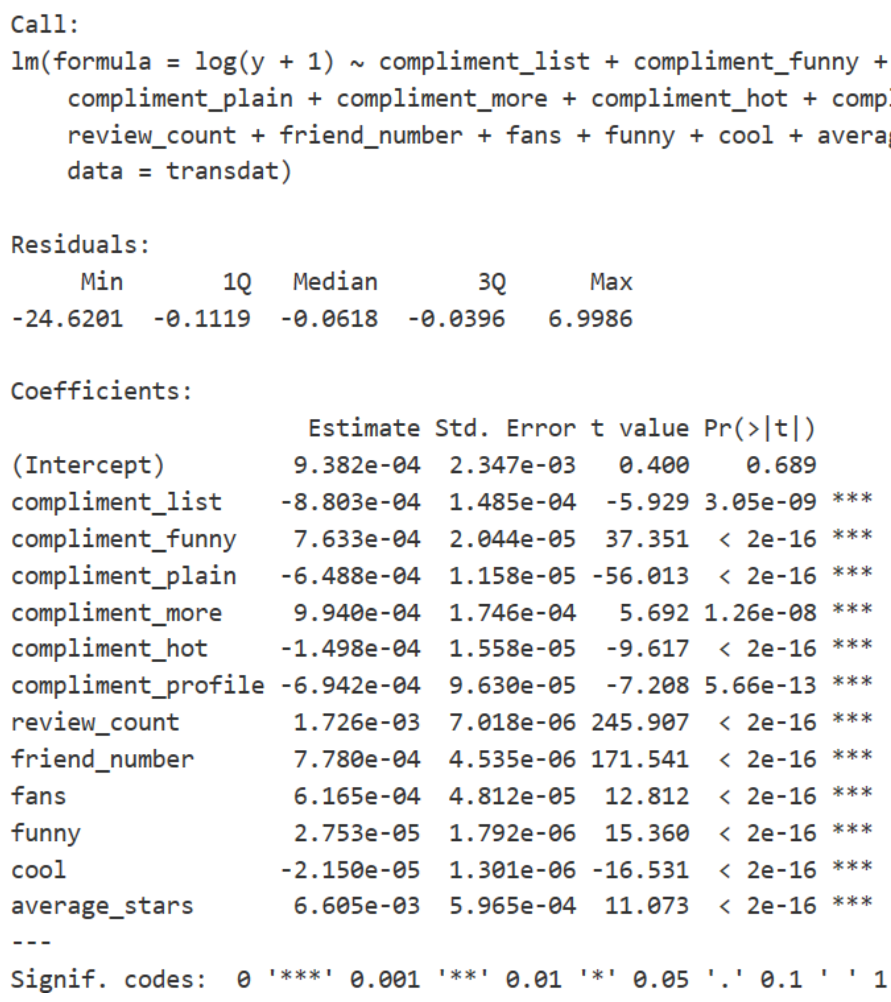


```
Call:
glm.nb(formula = log(number + 1) ~ compliment_photos + compliment_funny +
    compliment_hot + compliment_writer + compliment_plain + compliment_list +
    compliment_more + useful + cool + fans + average_stars +
    friend_number + review_count, data = nbdat, init.theta = 0.2936885619,
    link = log)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-6.9608  -0.3910  -0.3633  -0.3406   5.5067

Coefficients:
                    Estimate Std. Error  z value Pr(>|z|)
(Intercept)       -3.234e+00  2.027e-02 -159.570  < 2e-16 ***
compliment_photos  4.035e-04  4.277e-05    9.434  < 2e-16 ***
compliment_funny   6.346e-04  8.712e-05    7.284 3.24e-13 ***
compliment_hot    -4.113e-04  6.394e-05   -6.433 1.25e-10 ***
compliment_writer -2.651e-03  1.716e-04  -15.444  < 2e-16 ***
compliment_plain  -7.651e-04  4.926e-05  -15.531  < 2e-16 ***
compliment_list   -1.705e-02  6.421e-04  -26.547  < 2e-16 ***
compliment_more    1.694e-02  6.375e-04   26.575  < 2e-16 ***
useful             5.260e-05  6.620e-06    7.945 1.93e-15 ***
cool              -6.974e-05  7.029e-06   -9.922  < 2e-16 ***
fans              -3.454e-03  2.303e-04  -14.999  < 2e-16 ***
average_stars      1.208e-01  5.071e-03   23.825  < 2e-16 ***
friend_number      2.587e-03  2.051e-05  126.100  < 2e-16 ***
review_count       6.747e-03  2.945e-05  229.109  < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Output 2 : Summary for N-B Regression Model



```
Call:
lm(formula = log(y + 1) ~ compliment_list + compliment_funny + 
    compliment_plain + compliment_more + compliment_hot + compliment_profile + 
    review_count + friend_number + fans + funny + cool + average_stars, 
    data = transdat)

Residuals:
     Min       1Q   Median       3Q      Max 
-24.6201  -0.1119  -0.0618  -0.0396   6.9986 

Coefficients:
                     Estimate Std. Error t value Pr(>|t|)    
(Intercept)         9.382e-04  2.347e-03   0.400    0.689    
compliment_list    -8.803e-04  1.485e-04  -5.929 3.05e-09 ***
compliment_funny    7.633e-04  2.044e-05  37.351  < 2e-16 ***
compliment_plain   -6.488e-04  1.158e-05 -56.013  < 2e-16 ***
compliment_more     9.940e-04  1.746e-04   5.692 1.26e-08 ***
compliment_hot     -1.498e-04  1.558e-05  -9.617  < 2e-16 ***
compliment_profile -6.942e-04  9.630e-05  -7.208 5.66e-13 ***
review_count        1.726e-03  7.018e-06 245.907  < 2e-16 ***
friend_number       7.780e-04  4.535e-06 171.541  < 2e-16 ***
fans                6.165e-04  4.812e-05  12.812  < 2e-16 ***
funny               2.753e-05  1.792e-06  15.360  < 2e-16 ***
cool               -2.150e-05  1.301e-06 -16.531  < 2e-16 ***
average_stars       6.605e-03  5.965e-04  11.073  < 2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Output 3 : Summary for Transform OLS Model

Appendix: R code

```
################ Finding Influential Users

matchstr=function(x){
a=length(x$user_id)
b=0
supusr=c()
usrfri=c()
for(i in 1:(a-1)){
g=which(user.friendonly$user_id==x$user_id[i])
if(length(g)){
friends=user.friendonly[g]$friends
for(j in (i+1):a){
if(is.na(str_match(friends,x$user_id[j]))){
b=b
}else{
b=b+1
supusr=c(supusr,x$user_id[i])
usrfri=c(usrfri,x$user_id[j])
}
}
}
}
return(list(number=b,superuser=supusr,follower=usrfri))
}




################ Finding Superusers and Followers

b=0
superuser=c()
follower=c()
temp=c()
for(i in 1:(length(dup)-1)){
a1=review.res_america[dup[i]:(dup[i+1]-1),3]
if(matchstr(a1)$number!=0){
temp=c(temp,i)
}
b=b+matchstr(a1)$number
superuser=c(superuser,matchstr(a1)$superuser)
follower=c(follower,matchstr(a1)$follower)
print(i)
}

################ Sentiment Analysis
text_df<-data_frame(index=1:nrow(data),text=data$review %>% as.character())
tt  = text_df %>% unnest_tokens(word, text)
```

```
dt = cast_sparse(tt, index, word)
dictionary = data.frame(word = colnames(dt)) %>% as.tbl
sdic = sentiments$word %>% table %>% sort %>% names %>% rev
sdesc = sentiments$sentiment %>% table %>% sort %>% names %>% rev
sdesc = c(NA,sdesc)
tmp = sparseMatrix(i = match(sentiments$word, sdic),
j=match(sentiments$sentiment, sdesc))
sdesc[1] = "NA"
sentMat = as.matrix(tmp); colnames(sentMat) = sdesc
sentFrame = data.frame(word = sdic, sentMat) %>% as.tbl %>%
left_join(get_sentiments("afinn")) %>% left_join(get_sentiments("bing"))
sentFrame[is.na(sentFrame)] = 0
sentFrame$sentiment[sentFrame$sentiment=="positive"] = 1
sentFrame$sentiment[sentFrame$sentiment=="negative"] = -1
sentFrame$sentiment = as.numeric(sentFrame$sentiment)
feels = left_join(dictionary, sentFrame)
goodWords = which(rowSums(is.na(feels))==0)
x = dt[,goodWords]
feels = feels[goodWords,]
sx = x%*%as.matrix(feels[,-1])
sx=as.matrix(sx)

mod1=glm(review.influence~sx,family = poisson(link="log"))
summary(mod1)

mod2=lm(review.influence~sx)
summary(mod2)






################ Classify Restaurants

# ****************** cluster based on category variables ***********
### Try PCA
pca = PCA((sapply(cateVars, function(x){as.numeric(x)})))
#pca$eig ### very bad result

### Try MCA
mca = MCA(cateVars, ncp = 50, graph = TRUE)
#mca$eig ### still bad

### Try k-modes ckuster
save(cateVars, file = 'cateVars.Rdata')

# ******* find optimal k for k-modes cluster of category variables ********
### run this on linux
load('cateVars.Rdata')
seedList <- rnorm(1000, 2, 1000)
wssDf = data.frame(matrix(rep(0, 25000), nrow = 1000, ncol = 25))
```

```
i = 1
for (seed in seedList){
set.seed(seed); k.max <- 25
wss <- sapply(2:k.max, function(k){sum(kmodes(cateVars, k)$withindiff)})
wssDf[i,] <- wss; print(i)
}
write.table(wssDf, "wssDf.txt", row.names = FALSE, col.names = TRUE, quote = FALSE)
####
## use similiar way to find optimal seed
wssSeed <- fread("wssSeed.txt", colClasses=c(seed="float",wss ="float"))
set.seed(which[wssSeed$wss = min(wssSeed$wss), 1]); kmodesRe = kmodes(cateVars, 14)

# *********** cluster based on attribute variables **********************
library("factoextra")
mainVar$review_count <- (mainVar$review_count -min(mainVar$review_count ))/(max(mainVar$review_count
seed = 10.38; set.seed(seed)
clusterRe = kproto(mainVar,10)

# ************ cluster based on location variables **********************
locationVar <- data.frame(latitude = business$latitude, longtitude = business$longitude)
locationVar <- locationVar %>% filter(longtitude < -50) # some wrong record
### tried 1000 times respectively to found the best k and seed
seed = -69.12361; set.seed(seed)
kmeansRe <- kmeans(locationVar, 7, algorithm = "MacQueen", iter.max = 5000)
locationVar$cluster = factor(kmeansRe$cluster)
```

```
################ Shiny App

server <- function(input, output) {
# prepare dataset
data=data.frame(x=as.numeric(as.character(cityMadison[,2])),
y = as.numeric(as.character(cityMadison[,1])),
id=as.character(cityMadison$name),
category = as.numeric(as.character(cityMadison[,6])))
subdata = super[,c(5,3,1,4)]
names(subdata) = c("rank", "name", "user id", 'cluster')

# create a reactive value that will store the click position
data_of_click <- reactiveValues(clickedMarker=NULL)

# Leaflet map with 2 markers
output$map <- renderLeaflet({
leaflet() %>%
addProviderTiles(providers$Stamen.TonerLite,
options = providerTileOptions(noWrap = TRUE)) %>%
addCircleMarkers(data=data, ~x , ~y, layerId=~id,
popup=~id, radius=8 , color="black",
fillColor="red", stroke = TRUE, fillOpacity = 0.99)
})

# store the click
observeEvent(input$map_marker_click,{
p <- input$map_marker_click
})

datasetInput <- eventReactive(input$map_marker_click, {
curr_id = input$map_marker_click$id
curr_cate = data[which(data$id == curr_id),4]
temp <- subdata[which(subdata$cluster == curr_cate),]
temp <- temp[,c(1,2,3)]
temp = temp[order(temp$rank),]
names(temp) <- c("rank", "name", "user id")
temp
}, ignoreNULL = FALSE)

# Filter data based on selections
output$table <- renderTable({
datasetInput()
})
}

ui <- fluidPage(
br(),
column(8,leafletOutput("map", height="750px")),
column(4,br(),br(),br(),br(),tableOutput("table")),
br()
)

shinyApp(ui = ui, server = server)




################ Models

### 1. lOGISTIC REGRESSION

###TOP 5% NUMBERS TO BE 1, OTHERS TO BE 0......(SUPERUSER IS 1, NONE-superuser is 0)
spu=rep(0,760008)
spu[(regressdat$number>=100) %>% which]=1
logisdat=cbind(regressdat,spu)
logisdat=logisdat[,-12]
logisdat$spu=as.factor(logisdat$spu)
logisdat=logisdat %>% select(spu,starts_with("compliment"),
review_count,friend_number,fans,
useful,funny,cool,average_stars)
mod.logis=glm(formula = spu~.,family = binomial, data=logisdat )
```

```
summary(mod.logis)
model.bic=step(mod.logis,k=log(760008))
mod.logis.bic=glm(formula = spu~fans+compliment_writer+
compliment_profile+compliment_funny+
compliment_plain+friend_number+review_count,
family = binomial,data=logisdat)
summary(mod.logis.bic)

### 2. BOXCOX-TRANSFORMATION: Y->LOG(Y+1)
y=regressdat$number
y=log(y+1)
transdat=cbind(y,regressdat)
transdat=transdat %>% select(y,starts_with("compliment"),
review_count,friend_number,fans,
useful,funny,cool,average_stars)
mod.trans=lm(y~.,data=transdat)
summary(mod.trans)
mod.trans.bic=step(mod.trans,k=log(760008))
###Result: bic removes compliment_more&note&list&cute, and useful
mod.trans.bic=lm(y~compliment_list+compliment_funny+compliment_plain+compliment_more+
compliment_hot+compliment_profile+review_count+friend_number+fans+
funny+cool+average_stars,data=transdat)
summary(mod.trans.bic)

### 3. NB regression
nbdat=regressdat[,-12]
mod.nb=glm.nb(log(number+1)~.,data=nbdat)
summary(mod.nb)

step(mod.nb,k=log(760008))
mod.nb.bic=glm.nb(log(number+1)~compliment_photos+compliment_funny+compliment_hot+
compliment_writer+compliment_plain+compliment_list+compliment_more+
useful+cool+fans+average_stars+friend_number+review_count,data=nbdat)
summary(mod.nb.bic)
```