This project involves analysis of Yelp data on AWS to solve two questions. First: Are users more likely to follow elite users, as compared to non-elite users? Second: Predict customer volumes (check-in) for the businesses.

Question 1: Are users more likely to follow elite users, as compared to non-elite users?

outline

1. Using 2017 elites and Poisson Regression (# of followees 2017 ~ is_elite in 2017).
2. Whether or not the user is elite in 2016.
3. How many times that a user was awarded as elite in history.
4. The number of reviews they made.
5. The number of photos they post.
6. The number of tips they give.

Coding Notebook:

```
library(DBI)

library(RMySQL)

library(dplyr)

# The “dbConnect” command is to connect R studio with AWS Server, so as to retrive the database from AWS Server. The parameters, user name, password, dbname, host address and port are set from AWS.

mydb = dbConnect(MySQL(), host="34.216.20.221", dbname="yelp_db", user="user", password="msba_2018")

# After connecting R with the database "yelp_db" on AWS, checking the tables inside with "dbListTables"

dbListTables(mydb)

# Outcome: [1] "attribute"    "business"    "category"    "checkin"    "elite_years"    "friend"
"hours"    "photo"    "review"    "tip"    "user"

# Extracting elites’ users in 2017 (corresponding to Hint 1).

res = dbSendQuery(mydb, "select distinct user_id from elite_years where year = '2017' ")

elite_2017 = fetch(res, n = -1)

# Outcome: 34928 observations, 1 variable.
```

```
dbClearResult(res)

# Extracting elites' users in 2016 (corresponding to Hint 2).

res = dbSendQuery(mydb, "select distinct user_id from elite_years where year = '2016'")

elite_2016 = fetch(res, n = -1)

# Outcome: 30856 observations, 1 variable.

dbClearResult(res)

# Calculating the times that a user was awarded as elite in history (corresponding to Hint 3).

res = dbSendQuery(mydb, "select user_id, year from elite_years")

elite_history = fetch(res, n=-1)

elite_history$year = 1

elite_count<-elite_history %>% group_by(user_id) %>% summarise(elite_times = sum(year))

# Outcome: 60818 observations, 2 variables (user_id, elite_times).

dbClearResult(res)

# Calcuting the number of friends each user has.

res = dbSendQuery(mydb, "select user_id, count(1) from friend group by user_id")

num_friend = fetch(res, -1)

names(num_friend)[2] = "cnt_friends"

# Outcome: 760008 observations, 2 variables (user_id, cnt_friends).

dbClearResult(res)


# Selecting the number of reviews users made, average star-ratings and the number of useful comments
(corresponding to Hint 4).

res = dbSendQuery(mydb, "select id, review_count, average_stars, useful from user group by id")

user = fetch(res, n=-1)

names(user)[1] = 'user_id'
```

Outcome: 1326101 observations, 4 variables (user_id, review_count, average_stars, useful).

dbClearResult(res)

Calculating the number of photos users post (corresponding to Hint 5).

res = dbSendQuery(mydb, "select id, label from photo group by id")

photo_cnt = fetch(res, n=-1)

photo_cnt$label = 1

photo_count<-photo_cnt %>% group_by(id) %>% summarise(count = sum(label))

From the result of photo_count, we find out that each distinct user posts only 1 photo through the Yelp. There is no any data distribution regarding the number of photos users post. Therefore, we decided not to apply variable photo_cnt into our prediction model.

dbClearResult(res)

Calculating the number of tips users give (corresponding to Hint 6).

res = dbSendQuery(mydb, "select user_id, likes from tip group by id")

tip_cnt = fetch(res, n=-1)

tip_cnt$likes = 1

tip_count<-tip_cnt %>% group_by(user_id) %>% summarise(tip_count = sum(likes))

Outcome: 271680 observations, 2 variables (user_id, tip_count).

dbClearResult(res)

Merging all required variables (is_elite 2017, is_elite_before 2016, elite_count, user, tip_count) into one data frame, we called it "df".

df = num_friend

df$is_elite = 0

df$is_elite[df$user_id %in% elite_2017$user_id] = 1

df$is_elite_before = 0

df$is_elite_before[df$user_id %in% elite_2016$user_id] = 1 df<- merge(df,elite_count,by='user_id',all.x=TRUE)

df<- merge(df,user,by='user_id',all.x=TRUE)

df<- merge(df,tip_count,by='user_id',all.x=TRUE)

df[is.na(df)] <- 0

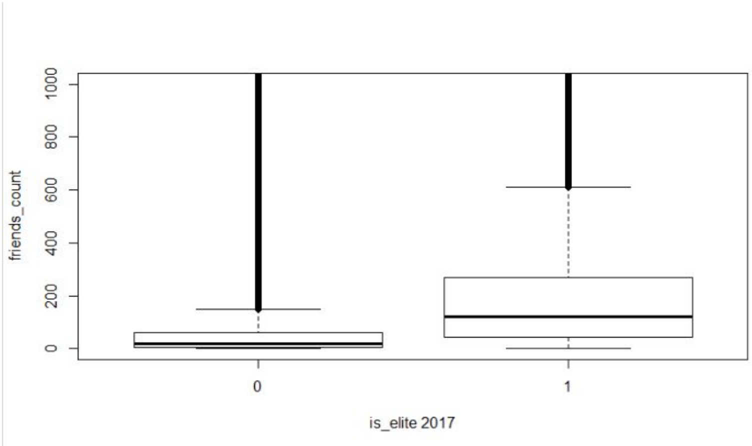
Outcome: 760008 observations, 9 variables (user_id, cnt_friends, is_elite, is_elite_before, elite_times, review_count, average_stars, useful, tip_count).

	user_id	cnt_friends	is_elite	is_elite_before	elite_times	review_count	average_stars	useful	tip_count
1	___fEWIObjtPaZ-pK0eq9g	1	0	0	0	7	5.00	0	0
2	___I9ZYdYGkZ6dMYxwJEIQ	161	0	0	0	168	3.98	140	3
3	___MTsBIoH4jvybJ5DrTYw	6	0	0	0	10	4.90	3	0
4	__-FcRiBzu8tqWYGofto1Q	57	1	1	2	102	4.39	7	0
5	__-jTbcqIU4pwDy4BZ9JIQ	11	0	0	1	55	3.73	7	0
6	__-Kt26YrtJxGdWs8FqKCg	106	0	0	0	7	4.43	0	6
7	__-Xtxtb7z5YEag85AaHWw	2	0	0	0	9	3.89	0	0
8	__02xIXHMOZda_nPoBTnoQ	2	0	0	0	1	5.00	0	3
9	__05rytNjsye9MBhqB0DMA	149	1	1	9	1037	3.64	7	0
10	__0cgHc1KI1O7WhfIPTZFA	37	0	0	0	40	3.93	1	0
11	__0D94KGQI7dBCcA2MmH0w	9	0	0	0	37	3.54	6	0
12	__0h-8X-zMnS_ghVoxprUg	28	0	0	0	3	5.00	0	0
13	__0NoInkjvjBExSstL7_ww	75	0	0	0	71	4.11	31	1
14	__1cb6cwI3uAbMTK3xaGbg	3	0	0	0	4	4.25	0	0
15	__2dq1OFY1onl-e60macuw	10	0	0	0	6	2.50	2	5
16	__2Xu2F0Z1gAodYpIdOsCQ	459	0	0	5	300	3.80	56	0
17	__3DNxSoaA-wE5t_hbLN-Q	90	0	0	0	4	3.25	0	0
18	__3gdV_ALx9QQzdXWHHUew	4	0	0	0	40	4.05	7	2
19	__3Lm1VioOK5WHL2tt476w	68	0	0	1	75	3.51	17	0

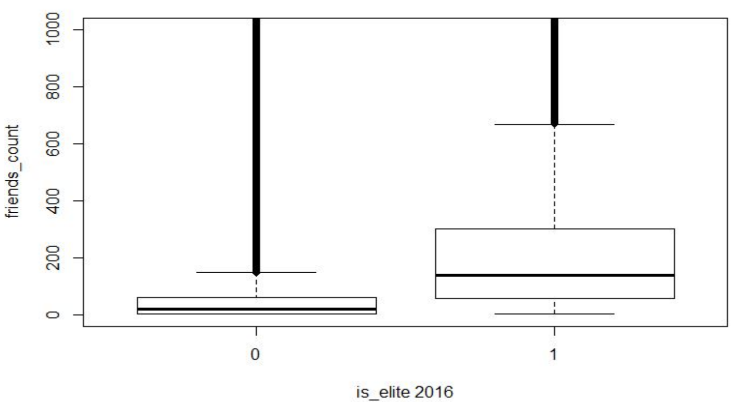
Showing 1 to 19 of 760,008 entries

Plot Visualization:

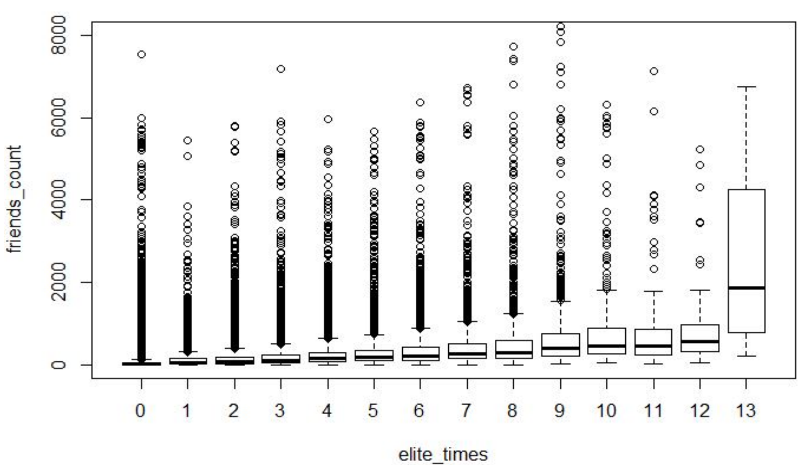
boxplot(df$cnt_friends ~ df$is_elite, df, ylim = c(0, 1000), ylab="friends_count", xlab="is_elite 2017")



boxplot(df$cnt_friends ~ df$is_elite_before, df, ylim = c(0, 1000),ylab="friends_count", xlab="is_elite 2016")

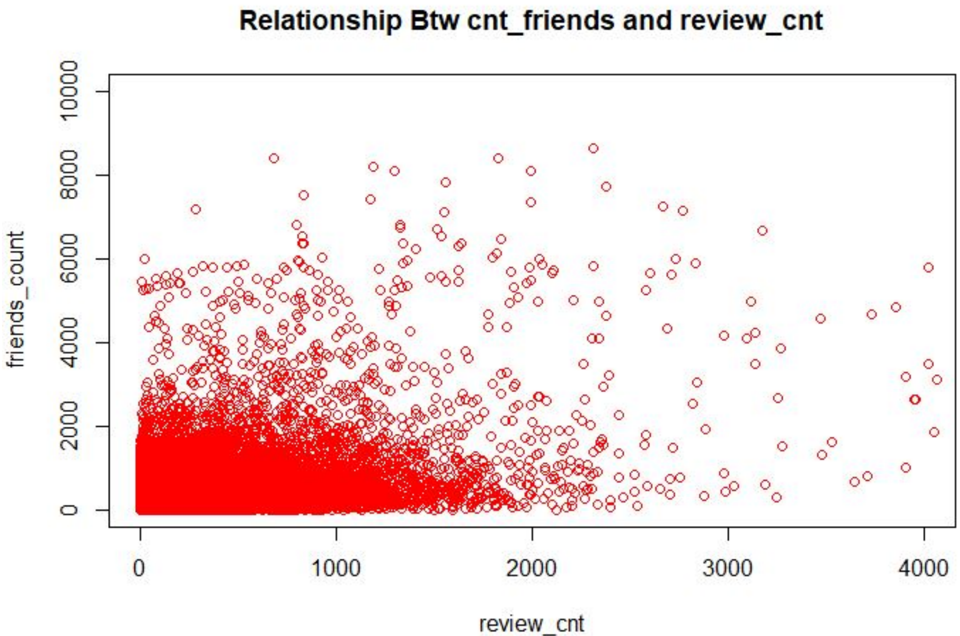


boxplot(df$cnt_friends ~ df$elite_times, df, ylim = c(0, 8000),ylab="friends_count", xlab="elite_times")



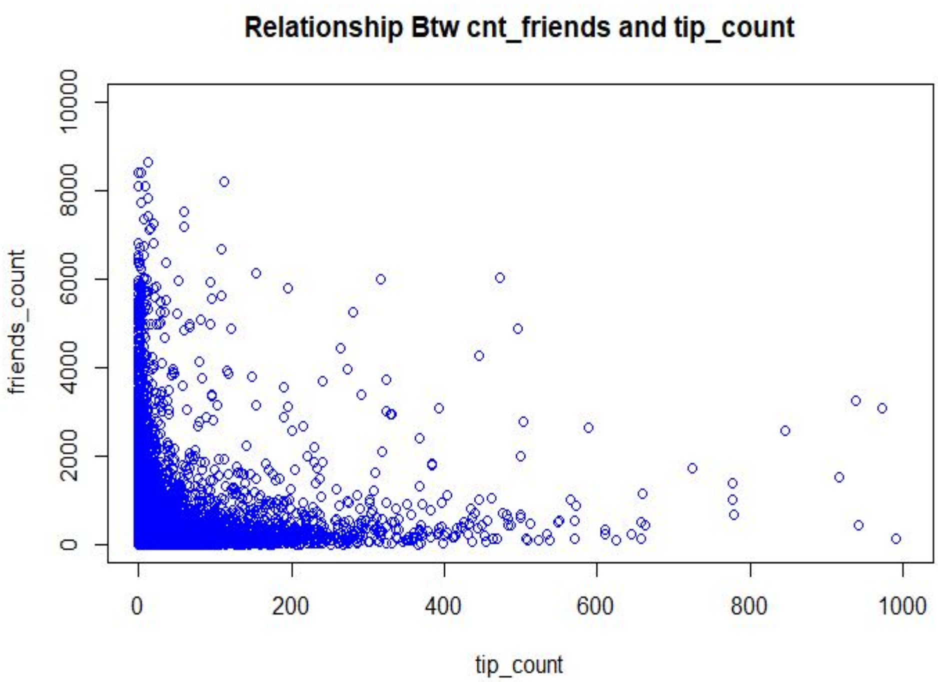
plot(y=df$cnt_friends, x=df$review_count, col="red", xlim=c(0,4000), ylim=c(0, 10000),

main="Relationship Btw cnt_friends and review_cnt",   ylab="friends_count", xlab="review_cnt")



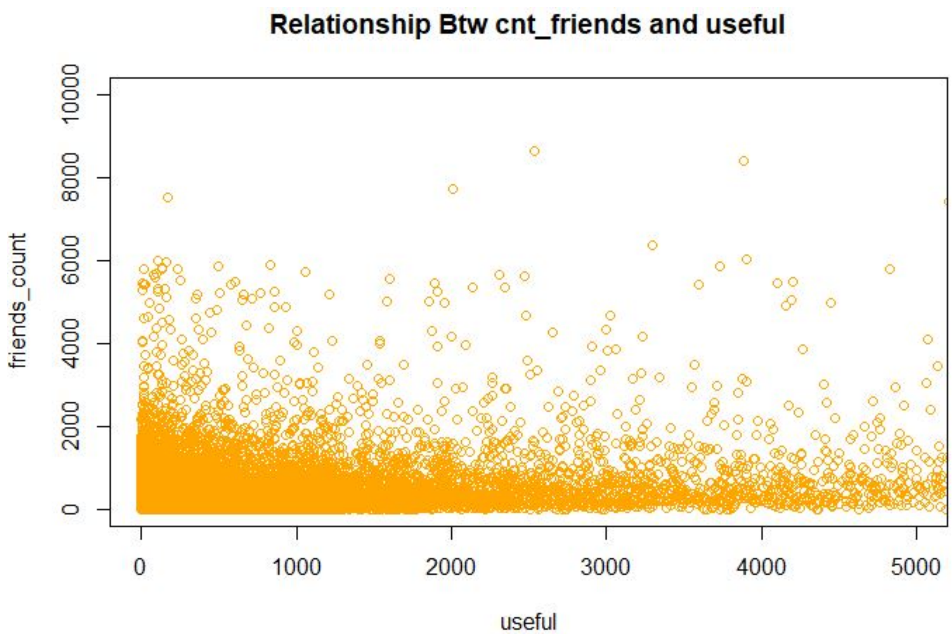
plot(y=df$cnt_friends, x=df$tip_count, col="blue", xlim=c(0,1000), ylim=c(0, 10000),

main="Relationship Btw cnt_friends and tip_count", ylab="friends_count", xlab="tip_count")



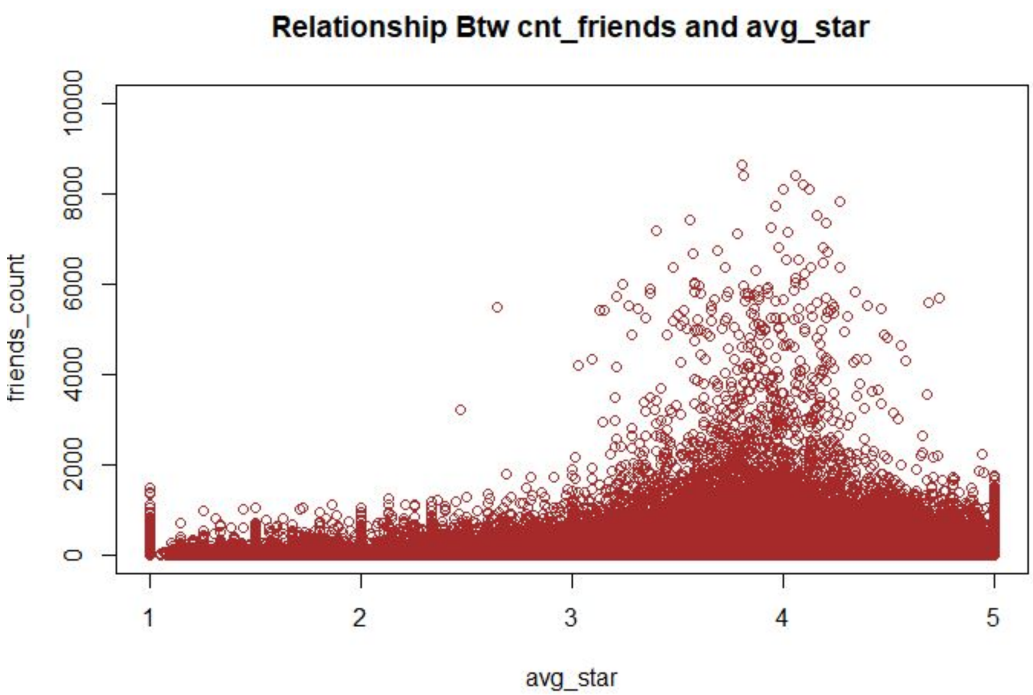
plot(y=df$cnt_friends, x=df$useful, col="orange", xlim=c(0,5000), ylim=c(0, 10000),

main="Relationship Btw cnt_friends and useful", ylab="friends_count", xlab="useful")




```
plot(y=df$cnt_friends, df$average_star, col="brown", ylim=c(0, 10000),

main="Relationship Btw cnt_friends and avg_star", ylab="friends_count", xlab="avg_star")
```



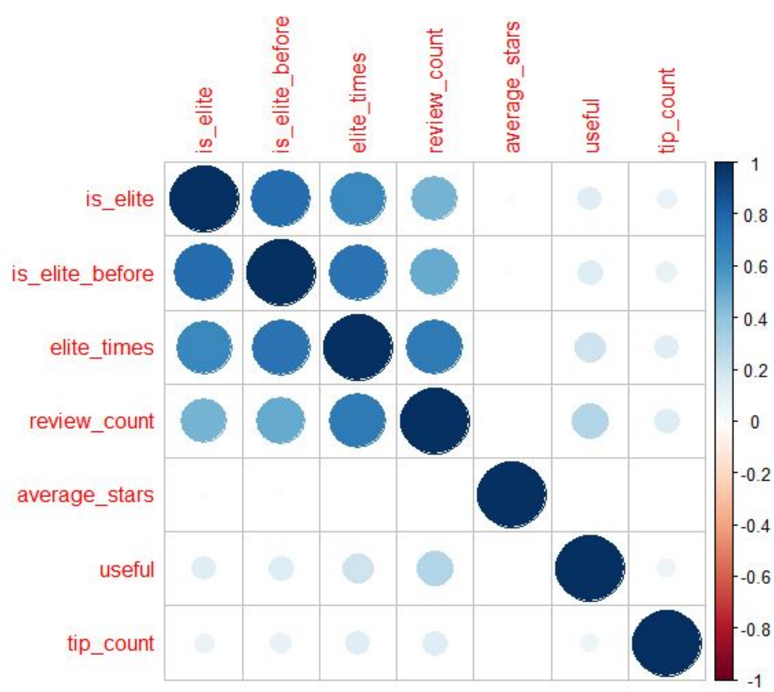
Poisson Regression Model:

Correlation Analysis: for the purpose of building better regression model to predict whether users are more likely to follow elite users, as compared to non-elite users, we should not use independent variables which are relatively highly correlated (P value is > 0.5).

```
correlation_data<-subset(df, select=c(is_elite, is_elite_before, elite_times,
                                      review_count, average_stars, useful, tip_count))
library('corrplot')
correlation_matrix<-cor(correlation_data)
```

	is_elite	is_elite_before	elite_times	review_count	average_stars	useful	tip_count
is_elite	1.00000000	0.76159318	0.646262966	0.463463289	0.021384350	0.126132365	0.087520863
is_elite_before	0.76159318	1.00000000	0.736690410	0.501889638	0.014261907	0.138274793	0.094186868
elite_times	0.64626297	0.73669041	1.000000000	0.701191953	0.006283994	0.201959679	0.120769562
review_count	0.46346329	0.50188964	0.701191953	1.000000000	-0.005605705	0.291651169	0.134460390
average_stars	0.02138435	0.01426191	0.006283994	-0.005605705	1.000000000	0.001528892	0.004733455
useful	0.12613236	0.13827479	0.201959679	0.291651169	0.001528892	1.000000000	0.077402241
tip_count	0.08752086	0.09418687	0.120769562	0.134460390	0.004733455	0.077402241	1.000000000

```
corrplot(correlation_matrix, method = "circle")
```



Based on the correlation matrix and corrplot above, we find that the following variables are relatively highly correlated (P value is > 0.5).

is_elite and is_elite_before: P = 0.76;

is_elite_before and elite_times: P = 0.74;

elite_times and review_count: P = 0.70;

elite_times and is_elite: P = 0.65;

After analyzing the matrix above, we decide to use the following independent variables for the regression model:

is_elite, average_stars, useful, tip_count

fit1 = glm(cnt_friends ~ is_elite + average_stars + useful + tip_count, data = df, family = poisson)

summary(fit1)

```
Call:
glm(formula = cnt_friends ~ is_elite + average_stars + useful +
    tip_count, family = poisson, data = df)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-335.36    -9.16    -5.82     0.81   357.93

Coefficients:
               Estimate Std. Error z value Pr(>|z|)
(Intercept)   3.687e+00  6.544e-04  5635.0   <2e-16 ***
is_elite      1.423e+00  3.797e-04  3748.4   <2e-16 ***
average_stars 8.918e-02  1.640e-04   543.6   <2e-16 ***
useful        2.541e-05  1.260e-08  2016.7   <2e-16 ***
tip_count     2.711e-03  2.530e-06  1071.7   <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

(Dispersion parameter for poisson family taken to be 1)

    Null deviance: 106227778  on 760007  degrees of freedom
Residual deviance:  92625817  on 760003  degrees of freedom
AIC: 96229068

Number of Fisher Scoring iterations: 9
```

exp(fit2$coefficients) – 1

Outcome:

(Intercept)	is_elite	average_stars	useful	tip_count
3.894190e+01	3.150237e+00	9.327360e-02	2.541176e-05	2.715148e-03

Interpretation:

After running the Poisson Regression model, we found that P values for is_elite, average_stars, useful and tip_count are far less than 0.05, which indicate that these 4 variables are significant factors regarding the prediction "whether users are more likely to follow elite users, as compared to non-elite users.

Among these 4 variables, is_elite has the largest coefficient value (1.423) and the smallest P value ($p < 2.2e-16$). Therefore, we could say that is_elite is the most significant variable for the prediction model. From the Yelp case, we could draw the conclusion that users are more likely to follow elite users, as compared to non-elite users.

Then, we keep running Exp function on the model, we could make the following conclusions:

is_elite, average_stars, useful and tip_count would impose positive impacts on the "cnt_friends".

Since we have already known that is_elite is the most significant predictors for the outcome, we could say:

For Per units increases in elite users, the odds of converting users to be followers increases by approximately 315% (Holding everything else constant).

Question 2: Predict customer volumes (check-in) for the businesses with selected variables.

outline:

1. Using checkin_cnt from table “checkin” and Poisson Regression.
2. Neighborhood of the business.
3. The state or city of the business.
4. The category.
5. The attribute.
6. Volume and valence of the ratings.

Coding Notebook:

```
# Calculating the number of check-ins from table “checkin” (corresponding to Hint 1).

res = dbSendQuery(mydb, "select business_id, count from checkin")

checkin = fetch(res, n=-1)

# Outcome: 3911218 observations, 2 variables (business_id, count).

checkin_count<-checkin %>% group_by(business_id) %>% summarise(checkin_count = sum(count))

# Outcome: 146350 observations, 2 variables (business_id, checkin_count).

dbClearResult(res)

# Selecting neighborhood, state, review_count and stars of the business (corresponding to Hint 2, 3, 6).

res = dbSendQuery(mydb, "select id, state, neighborhood, review_count, stars from business")

business = fetch(res,n=-1)

names(business)[1] = 'business_id'

dbClearResult(res)

# Classifying neighborhood into “downtown” (1) or not (0), “nb” represents “neighborhood”.

business$nb_downtown = 0

business$nb_downtown[business$neighborhood =='Downtown' | business$neighborhood =="Downtown Core"] =1
```

```
business$neighborhood<-NULL

# Selecting category of the business (corresponding to Hint 4).

res = dbSendQuery(mydb, "select business_id, category from category")

category = fetch(res,n=-1)

# Classifying category into "restaurant" (1) or not (0).

category$restaurant = 0

category$restaurant [category$category=="Restaurants"] = 1

category$category<-NULL

category<-category %>% group_by(business_id) %>% summarise(restaurant=sum(restaurant))

# Outcome: 174067 observations, 2 variables (business_id, restaurant).

dbClearResult(res)

# Selecting attribute (ByAppointmetOnly & BusinessAcceptsCreditCards) of the business (corresponding to Hint 5).

res = dbSendQuery(mydb, "select business_id from attribute

                    where attribute.name = 'ByAppointmentOnly' and attribute.value = 1 ")

AppointmentOnly = fetch(res,n=-1)

dbClearResult(res)



res = dbSendQuery(mydb, "select business_id from attribute

                    where attribute.name = 'BusinessAcceptsCreditCards' and attribute.value = 1")

CreditCard = fetch(res,n=-1)

dbClearResult(res)
```

```
# Classifying attribute “ByAppointmetOnly” into “yes” (1) or no (0).

# Classifying attribute “BusinessAcceptsCreditCards” into “yes” (1) or no (0).

AppointmentOnly$ByAppointmentOnly = 0

AppointmentOnly$ByAppointmentOnly [checkin_count$business_id %in% AppointmentOnly$business_id] = 1

CreditCard$AcceptsCreditCard = 0

CreditCard$AcceptsCreditCard [checkin_count$business_id %in% CreditCard$business_id] = 1

# Merging all required variables (review_count, stars, nb_downtown, restaurant, ByAppointmentOnly,
AcceptsCreditCard) into one data frame, we called it “checkin_cnt”.

checkin_cnt<-merge(checkin_count, business, by="business_id")

checkin_cnt<-merge(checkin_cnt, category, by="business_id", all.x=TRUE)

checkin_cnt<-merge(checkin_cnt, AppointmentOnly, by="business_id", all.x=TRUE)

checkin_cnt<-merge(checkin_cnt, CreditCard, by="business_id", all.x=TRUE)

# Converting variable “state” (categorical data) into binary values. Our logic is to count the number of distinct
states, and find out which state has the largest number. For example, if state “CA” has the largest count, we will
create a new variable called “state_CA”. Then, classifying all states into “state_CA” (1) or not (0).

checkin_cnt$state_count = 1

state_order<- checkin_cnt %>% group_by(state) %>% summarise(count = sum(state_count))
```

	state	count
9	AZ	42051
40	NV	28569
44	ON	26085
35	NC	11062
43	OH	10760
45	PA	8637
46	QC	7549
57	WI	3940
19	EDH	3026
11	BW	2012

```
checkin_cnt$state_AZ = 0

checkin_cnt$state_AZ [checkin_cnt$state == "AZ"] = 1

checkin_cnt$state<-NULL

checkin_cnt$state_count<-NULL

view(checkin_cnt)
```

Outcome: 146350 observations, 9 variables (business_id, checkin_count, review_count, stars, nb_downtown, restaurant, ByAppointmentOnly, AcceptsCredictCard, state_AZ).

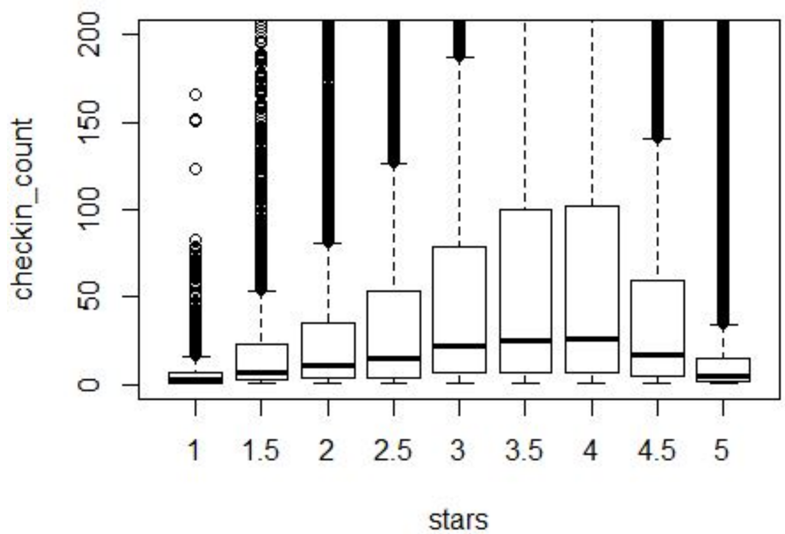
	business_id	checkin_count	review_count	stars	nb_downtown	restaurant	ByAppointmentOnly	AcceptsCredictCard	state_AZ
1	__1uG7MLxWGFIv2fCGPiQQ	33	26	5.0	0	0	0	1	1
2	__3I-DDkqM9XjLH1cJI3VA	8	13	5.0	0	0	0	0	0
3	__3qOwWFBUE8mdOToI7YrQ	1	12	1.0	0	0	0	1	0
4	__6jYJ6Hm-Qq8XQEGDrOGQ	2	4	4.0	0	0	1	0	1
5	__8j8yhsmE98wNWHJNyAgw	69	73	3.0	0	1	0	1	0
6	__aKnGBedQ51_hEc3D9ARw	662	75	3.0	0	1	0	1	1
7	__bqGGnOjtY9eEhrZAUsgA	19	20	3.0	0	1	0	1	0
8	__CQ2SE4NXFFjYfrB_TJ6w	5	6	3.0	0	0	0	0	0
9	__D6AVR_hLpW_bott0-upA	12	11	5.0	0	0	1	1	1
10	__eb2f_wEBrEI0xCyLqDeQ	2	7	3.0	0	1	0	1	0
11	__FFoyg0XmJIuBBNE0QP0w	64	22	5.0	0	0	1	1	1
12	__fMLrmv9M1_W4kBvR2VnQ	82	15	3.5	0	1	0	1	0
13	__fyRzU8kL6HkVV3wgxfmQ	13	3	3.5	0	1	0	1	0
14	__G0Ug3CK2yCDdQLYpd0ww	1	20	3.0	0	0	0	0	0
15	__H_61gpm7eViPMbWxPZSg	26	5	4.5	0	1	0	0	1
16	__hvr-Q534NEWQL1D4T5qg	28	7	3.0	0	0	0	1	1
17	__IFWGnWgMJV-55JIQfzjw	337	81	2.5	0	1	0	1	1
18	__iqJ91sPngnwEa3nIQP8Q	11	10	4.5	0	1	0	1	0
19	__IsqCZAF9YTcvKPKj2dZg	74	25	2.5	0	0	0	1	0

Showing 1 to 19 of 146,350 entries

Plot Visualization:

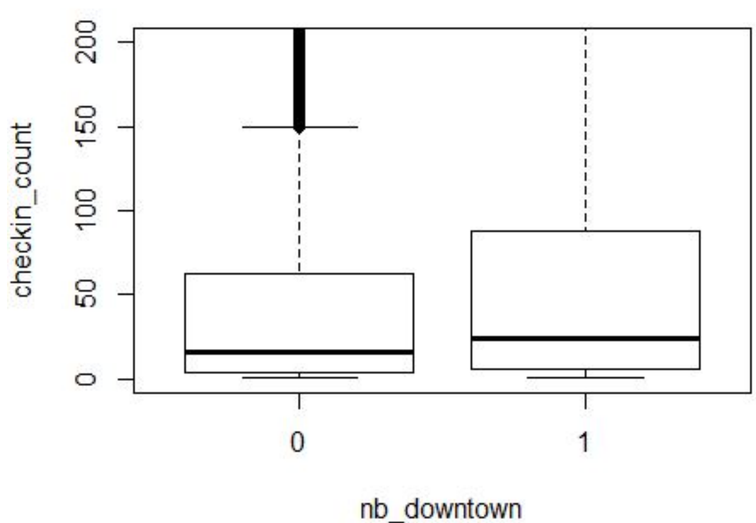
boxplot(checkin_cnt$checkin_count ~ checkin_cnt$stars, checkin_cnt,

ylim = c(0,200), ylab="checkin_count", xlab="stars")



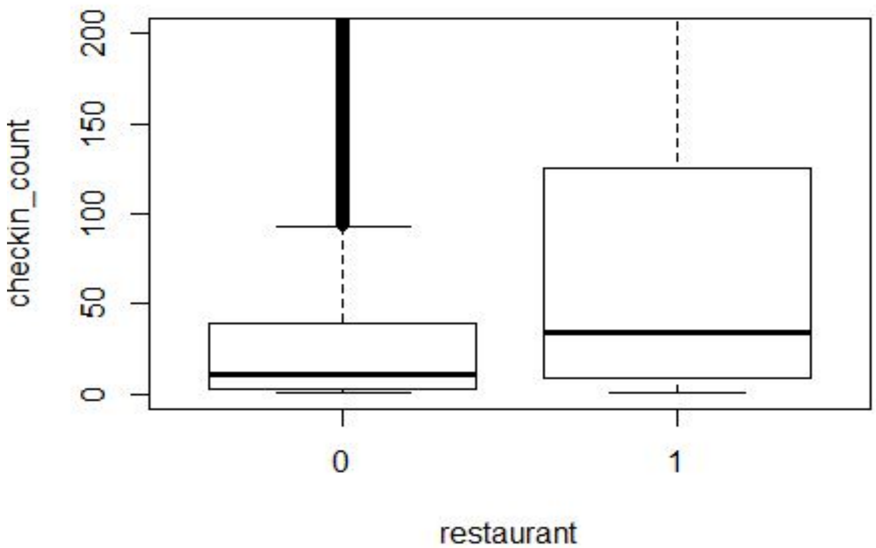
boxplot(checkin_cnt$checkin_count ~ checkin_cnt$nb_downtown, checkin_cnt,

ylim = c(0,200), ylab="checkin_count", xlab="nb_downtown")



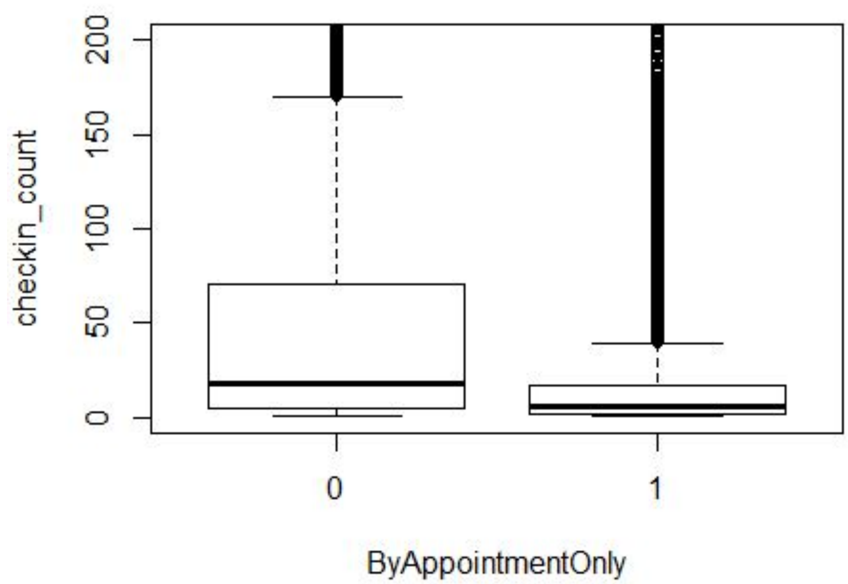
boxplot(checkin_cnt$checkin_count ~ checkin_cnt$restaurant, checkin_cnt,

ylim = c(0,200), ylab="checkin_count", xlab="restaurant")



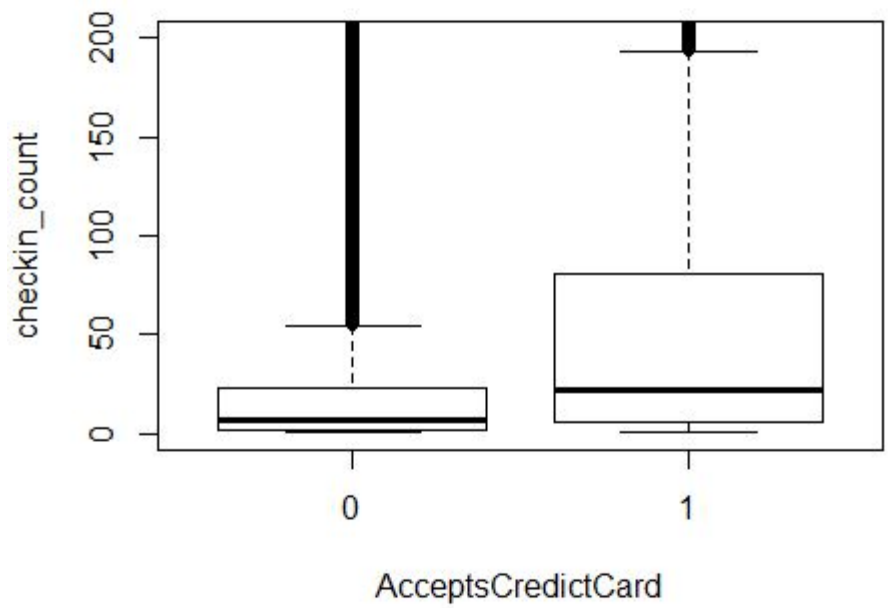
boxplot(checkin_cnt$checkin_count ~ checkin_cnt$ByAppointmentOnly, checkin_cnt,

ylim = c(0,200), ylab="checkin_count", xlab="ByAppointmentOnly")



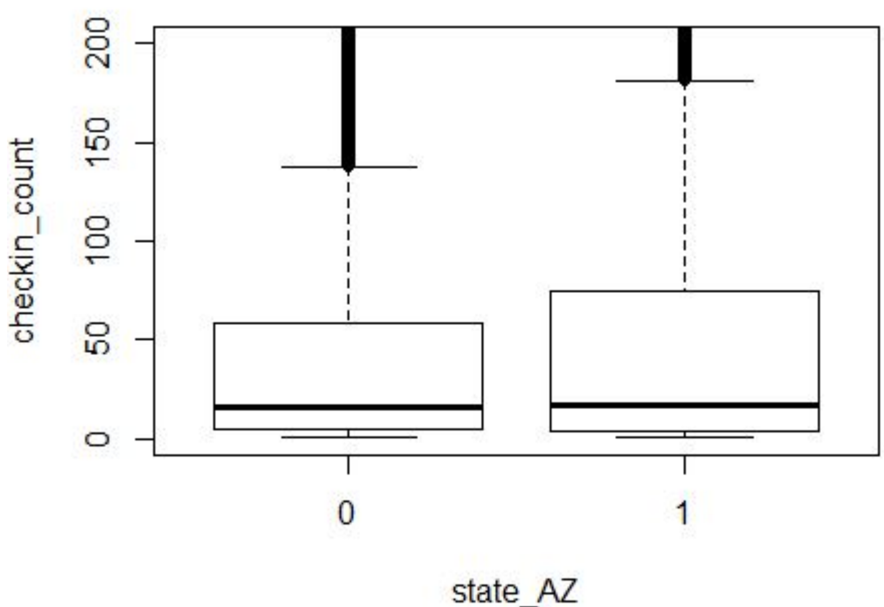
boxplot(checkin_cnt$checkin_count ~ checkin_cnt$AcceptsCreditCard, checkin_cnt,

ylim = c(0,200), ylab="checkin_count", xlab="AcceptsCreditCard")

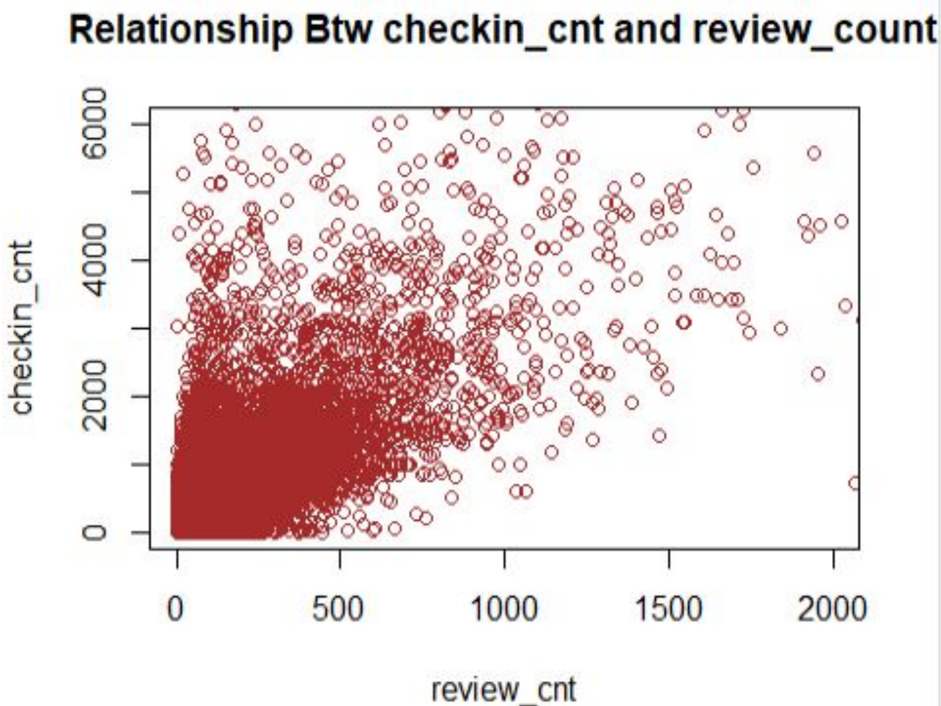


boxplot(checkin_cnt$checkin_count ~ checkin_cnt$state_AZ, checkin_cnt,

ylim = c(0,200), ylab="checkin_count", xlab="state_AZ")



```
plot(y=checkin_cnt$checkin_count, x=checkin_cnt$review_count, col="brown", xlim = c(0,2000), ylim=c(0, 6000),
main="Relationship Btw checkin_cnt and review_count", ylab="checkin_cnt", xlab="review_cnt")
```



Poisson Regression Model (corresponding to Hint 1):

Correlation Analysis: for the purpose of building better regression model to predict customer volumes (check-in) for the businesses with selected variables, we should not use independent variables which are relatively highly correlated (P value is > 0.5).

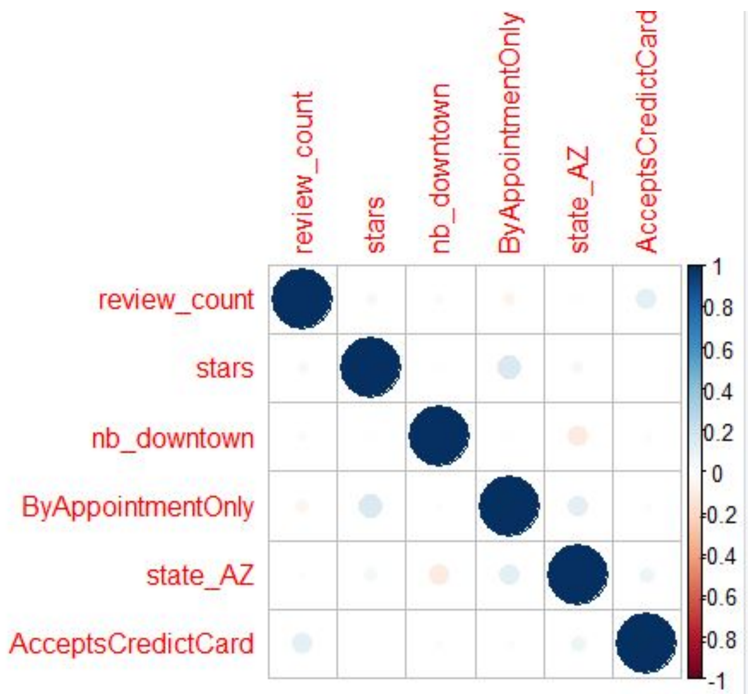
```
correlation_data2<- subset(checkin_cnt, select=c(review_count, stars, nb_downtown, ByAppointmentOnly,
                                                        AcceptsCredictCard, state_AZ))

library('corrplot')

correlation_matrix2<-cor(correlation_data2)
```

	review_count	stars	nb_downtown	ByAppointmentOnly	state_AZ	AcceptsCredictCard
review_count	1.00000000	0.039902219	0.02133049	-0.05229284	0.01326689	0.107574394
stars	0.03990222	1.000000000	-0.01052804	0.15072314	0.04300195	0.008867818
nb_downtown	0.02133049	-0.010528039	1.00000000	-0.01521566	-0.10099282	-0.015843577
ByAppointmentOnly	-0.05229284	0.150723138	-0.01521566	1.00000000	0.11471394	0.017165807
state_AZ	0.01326689	0.043001953	-0.10099282	0.11471394	1.00000000	0.063464256
AcceptsCredictCard	0.10757439	0.008867818	-0.01584358	0.01716581	0.06346426	1.000000000

```
corrplot(correlation_matrix2, method = "circle")
```



Based on the correlation matrix and corrplot above, we find that all the variables are not relatively correlated.

fit2 = glm(checkin_count ~ review_count + stars + nb_downtown + restaurant + ByAppointmentOnly

+ state_AZ + AcceptsCreditCard, data = checkin_cnt, family = poisson)

summary(fit2)

```
Call:
glm(formula = checkin_count ~ review_count + stars + nb_downtown +
    restaurant + ByAppointmentOnly + state_AZ + AcceptsCredictCard,
    family = poisson, data = checkin_cnt)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-823.38   -12.04    -8.64    -3.13   952.98

Coefficients:
                    Estimate Std. Error z value Pr(>|z|)
(Intercept)        3.465e+00  1.327e-03  2610.8   <2e-16 ***
review_count       1.079e-03  1.639e-07  6587.5   <2e-16 ***
stars              1.121e-01  2.966e-04   377.8   <2e-16 ***
nb_downtown        4.679e-01  1.365e-03   342.7   <2e-16 ***
restaurant         5.372e-01  5.298e-04  1014.1   <2e-16 ***
ByAppointmentOnly -1.463e+00  1.797e-03  -814.2   <2e-16 ***
state_AZ           4.131e-01  5.427e-04   761.1   <2e-16 ***
AcceptsCredictCard 5.449e-01  7.367e-04   739.7   <2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

(Dispersion parameter for poisson family taken to be 1)

    Null deviance: 61622871  on 146079  degrees of freedom
Residual deviance: 43976753  on 146072  degrees of freedom
  (270 observations deleted due to missingness)
AIC: 44673164

Number of Fisher Scoring iterations: 8
```

exp(fit2$coefficients) – 1

Outcome:

(Intercept)	review_count	stars	nb_downtown	restaurant	ByAppointmentOnly
30.966980564	0.001080051	0.118588752	0.596675665	0.711211432	-0.768492216
state_AZ	AcceptsCredictCard				
0.511439466	0.724463854				

Interpretation:

After running the Poisson Regression model, we found that P values for review_count, stars, nb_downtown, restaurant, ByAppointmentOnly, AcceptsCreditCard and state_AZ are far less than 0.05, which indicate that these 7 variables are significant factors regarding the prediction "customer volumes (check-in) for the businesses with selected variables.

Among these 7 variables, AcceptCreditCard, restaurant, nb_downtown and state_AZ have much larger coefficient value, compared to the other variables. Therefore, we could say that these 4 variables are the key significant factors for the prediction model. From the Yelp case, we could draw the conclusion that if you hope to increase your customer volumes (check-in) for the business, you better need to pay more attention on the 4 factors above. The optimal check-in situation is that you have restaurants close to downtown, accepting credit card payment, as well as locating in the Arizona state.

Then, we keep running Exp function on the model, we could make the following conclusions:

review_count, stars, nb_downtown, restaurant, AcceptsCreditCard and state_AZ would impose positive impacts on the "checkin_count".

ByAppointmentOnly would impose negative impacts on the "adopter".

For Per units increases in review stars, the odds of converting users to check-in increases by approximately 12% (Holding everything else constant).

For Per units increases in nb_downtown, the odds of converting users to check-in increases by approximately 60% (Holding everything else constant).

For Per units increases in restaurant category, the odds of converting users to check-in increases by approximately 71% (Holding everything else constant).

For Per units increases in ByAppointmentOnly attribute, the odds of converting users to check-in decreases by approximately 77% (Holding everything else constant).

For Per units increases in AcceptCreditCard attribute, the odds of converting users to check-in increases by approximately 72% (Holding everything else constant).

For Per units increases in Arizona state, the odds of converting users to check-in increases by approximately 51% (Holding everything else constant).