Do Friends on Yelp Give Similar Overall Ratings?

A Zabrodski

November 18, 2015

Introduction

Yelp is a social network for reviewing businesses. Users access the website or use an app on their mobile phone to post reviews about their experience at a business. Other users can read these reviews which will help guide their choice about where to go for tacos or which dentist is the best in their local area. More information is available on the Yelp (www.yelp.com) website. As of the second quarter of 2015, Yelp had a monthly average of 83 million unique visitors who visited Yelp via their mobile device and written more than 83 million reviews.

The purpose of this analysis is to look for predictive factors to estimate the average business rating of a users friends on Yelp. That is to say, do users who give high overall ratings also have a group of friends who give higher ratings. To be far more cynical, the question can be phrased as “Are jerks also friends with other jerks?”

Methods

The question in it’s most basic form asks if there is a relationship between a users average star rating and there friends. In order to keep the results interpretable we will start with a simple linear regression and analysis between a users rating and that of their friends. Secondly, we will expand the analysis to other measurable factors in the data such as if the user has given many compliments, or has received many votes that they had a “funny” or “cool” review.

Using linear regression allows for very interpretable coefficients, as each coefficient quantifies the level of influence from each factor and it can be tested for significance.

We will start by loading the data:

```
url <- "https://d396qusza40orc.cloudfront.net/dsscapstone/dataset/yelp_dataset_cha
llenge_academic_dataset.zip"
download.file(url, "yelp_data.zip", method = "curl")
unzip("yelp_data.zip")
setwd("/Users/azabrodski/Documents/Coursera DS/yelp_dataset_challenge_academic_dat
aset") # set to your working directory
user_data <- stream_in(file("yelp_academic_dataset_user.json"))

saveRDS(user_data, "user_data.RDS") # save data as single R object for future
```

Let’s explore the data set!

```
user <- readRDS("user_data.RDS")
dim(user)
```

```
## [1] 366715     11
```

```
names(user)
```

```
##  [1] "yelping_since" "votes"         "review_count"  "name"
##  [5] "user_id"       "friends"       "fans"          "average_stars"
##  [9] "type"          "compliments"   "elite"
```

There are over 35,000 rows (wow) and 11 columns. We have the user's ID, and a list of the IDs of their friends under the friends column. The votes and compliments columns are actually nested data frames and can be expanded for future analysis.

In order to examine the ratings of a users friend group, we need to ensure that they actually have friends. To do this we will add a column to the dataframe with the number of friends each user has.

```
for (i in 1:nrow(user)){
    index <- user$friends[i] # List of friends is nested within a list
    user$num_friends[i] <- length(index[[1]]) # assign number of friends
```

In order to achieve accurate results, we must limit the users to those that have enough friends, but not too many. They should have at least 30 in order to get a true average, and fewer than 250 since users with more than 250 friends are extreme outliers that will bias the results. We also need to clean the data to remove rows that contain any NAs since they cannot be used in regression.

```
# subset into 30 - 250 friends and remove NAs
user_sub <- subset(user, num_friends >= 30 & num_friends <= 250)
user_complete <- user_sub[complete.cases(user_sub$votes) ,]
user_complete <- user_sub[complete.cases(user_sub$compliments) ,]
```

```
nrow(user_complete)
```

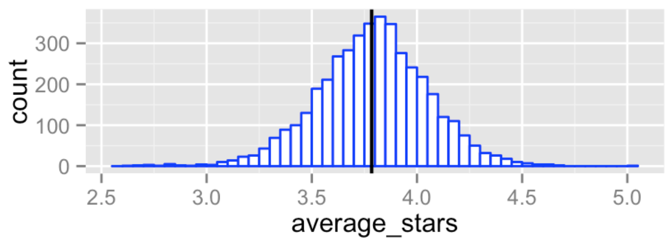
```
## [1] 4226
```

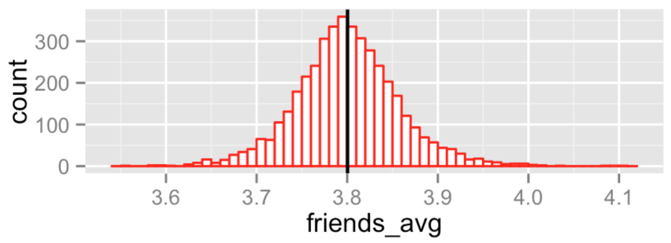
This leaves us with over 4,000 complete rows in which to work with. We want to know the average rating of a users friends, but only a list of each users friends IDs are given. We will need write a script that calculates the average rating from the list of friends.

```
for (items in 1:nrow(user_complete)){
    avg = vector(mode = "numeric")
    for (i in 1:length(user_complete$friends[[items]])){
        index = match(user_complete$friends[[items]][i], user$user_id)
        avg[i] = user$average_stars[index]
    }
    user_complete$friends_avg[items] <- mean(avg)
    user_complete$friends_median[items] <- median(avg)
    user_complete$std[items] <- sd(avg)
}
```

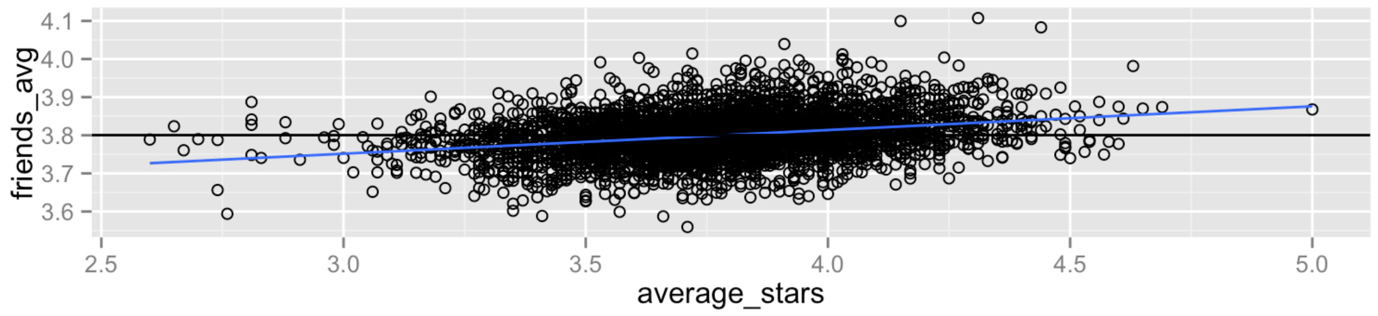
The script loops through each user, then goes through their list of friends, looks up each friends ID in the original user table, and appends their average rating to a vector. Once complete we add columns to contain the mean, median, and standard deviation of each users list of friends ratings.

Now that we have our desired outcome, we can plot the data to look for any trends and do some initial exploring before we model it.





The users average rating (blue), and the friends average rating (red) both appear to be normally distributed. The goal is to see if the tales of the distributions somewhat algin with each other. To visually explore that we will plot the users average rating vs friends average rating.



There does appear to be a linear relationship between a users average rating and that of their friends. However the line is quite flat suggesting that it is not an overly strong relationship. The outliers do seem to correlate well.

Before we model, let's take a look at the correlation between a user's average rating and of their friends average rating.

```
cor(user_complete$average_stars, user_complete$friends_avg)
```

```
## [1] 0.2855452
```

A correlation of 0.28 implies a weak or non-existent relationship. Next we will use linear regression to test this. First single variable, then multi-variate to see if any other factors are more influential.

Results

First we create the models:

```
fit1 <- lm(friends_avg ~ average_stars, data = user_complete) #single variable
#multi-variate model
fit2 <- step(lm(friends_avg ~ average_stars + votes$funny + votes$useful + votes$c
ool + compliments$profile + compliments$cute + compliments$plain + compliments$wri
ter + compliments$note + compliments$photos +compliments$hot + compliments$cool +
compliments$more + compliments$list,  data = user_complete), direction = "both", t
race = 0)
```

The step function let’s R go through the variables and optimize the best combination of inputs to explain the dependent variable.

Looking at the single variable regression

```
##
## Call:
## lm(formula = friends_avg ~ average_stars, data = user_complete)
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.236212 -0.033057 -0.001965  0.031849  0.276466
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept)   3.565160   0.012175  292.82   <2e-16 ***
## average_stars 0.062147   0.003209   19.36   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.05473 on 4224 degrees of freedom
## Multiple R-squared:  0.08154,    Adjusted R-squared:  0.08132
## F-statistic:   375 on 1 and 4224 DF,  p-value: < 2.2e-16
```

And the multi-variate self optimizing model

```
## 
## Call:
## lm(formula = friends_avg ~ average_stars + votes$useful + votes$cool + 
##     compliments$profile + compliments$cute + compliments$plain + 
##     compliments$writer + compliments$photos + compliments$cool + 
##     compliments$list, data = user_complete)
## 
## Residuals:
##       Min        1Q    Median        3Q       Max 
## -0.233904 -0.032769 -0.001654  0.032262  0.280556 
## 
## Coefficients:
##                       Estimate Std. Error t value Pr(>|t|)    
## (Intercept)          3.590e+00  1.256e-02 285.898  < 2e-16 ***
## average_stars        5.685e-02  3.263e-03  17.421  < 2e-16 ***
## votes$useful        -1.703e-05  2.982e-06  -5.713 1.19e-08 ***
## votes$cool           1.926e-05  3.668e-06   5.251 1.59e-07 ***
## compliments$profile  1.038e-04  2.780e-05   3.735 0.000191 ***
## compliments$cute    -8.880e-05  3.147e-05  -2.822 0.004798 ** 
## compliments$plain    1.167e-05  2.732e-06   4.271 1.99e-05 ***
## compliments$writer  -7.225e-05  1.905e-05  -3.793 0.000151 ***
## compliments$photos   1.848e-05  5.235e-06   3.530 0.000419 ***
## compliments$cool     1.231e-05  6.564e-06   1.875 0.060889 .  
## compliments$list    -1.209e-04  5.669e-05  -2.132 0.033029 *  
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## 
## Residual standard error: 0.05423 on 4215 degrees of freedom
## Multiple R-squared:  0.1001, Adjusted R-squared:  0.09792 
## F-statistic: 46.86 on 10 and 4215 DF,  p-value: < 2.2e-16
```

The single variable model achieved significance, but was only able to explain 8% of the variation. With the multi-variate model and with the extra inputs available, it was only able to explain 9% of the variation. Such low R squared values indicate that a user's average rating has at best case a slight influence on their friends average rating, but more than likely there is no relationship.

Discussion

The models indicated that for each unit increase in a users star rating, their friends average rating would increase by 0.05 - 0.06. That's a tiny change for such a large change in the input. So to answer the question, it seems that overall people have a diverse array of friends who rate businesses mostly independently. There is no observed relationship between a users average rating and the average rating of their friends. A potential issues with this analysis is the selection bias inherent with only using ~1/6 of the dataset which had values for all categories. The analysis was run with only these users, who were the most engaged with the Yelp network and therefore not necessarily representative of most users. A more interesting question to answer in the future would be, is there a relationship between the reviews of friends who both rated the same business.