---
title: "EC349 Individual Assignment"
author: "Ng Yi Fan, 2100907"
date: "`r Sys.Date()`"
output: html_document

[Link to GitHub](https://github.com/angelineyf/EC349-assignment.git)

```{r setup, include=FALSE}
knitr::opts_chunk$set(echo = TRUE)

#Load packages
library(dplyr)
library(tidyverse)
library(readr)
library(jsonlite)
library(caret)
library(stringr)
library(glmnet)

#Clear
cat("\014")
rm(list=ls())

#Set Directory as appropriate
setwd("C:/Users/angel/OneDrive - University of Warwick/Year 3/EC349 Data Science/R
projects/EC349-assignment/Yelp-datasets")

#Load .json Data
business_data <- stream_in(file("yelp_academic_dataset_business.json"))
checkin_data  <- stream_in(file("yelp_academic_dataset_checkin.json"))
tip_data  <- stream_in(file("yelp_academic_dataset_tip.json"))

#Load small data
review_data  <- load(file='yelp_review_small.Rda')
user_data  <- load(file='yelp_user_small.Rda')

```

1   Problem Definition

Can information on users and businesses help predict how many stars a user *X* rates a business *Y*? To answer this question, we will look into data from Yelp – an online platform known for its crowd-sourced business reviews and ratings. Using data on users, businesses, reviews, tips and check-ins, I will construct a predictive model for users star rating.

2   Data Understanding and Organisation

2.1   Data requirements

Firstly, I identified the factors that could most likely predict users star rating:

- Business’s star rating
- Business’s review count
- Power of the business’s star rating based on its review count
- Number of tips for each business
- Business check-ins
- Business category
- Opens over the weekend?
- Total weekly open hours
- Location
- Total number of reviews that a user has given

- User’s average star rating by business category
- Is the user an “elite”?
- Difference between the average rating by an “elite” compared to the “non-elite”, by each business
- How long has the user been using Yelp?

The list of factors has gone through several rounds of iterations by repetitive understanding of the data.

2.2   Data preparation

Some variables require more processing than others while some may be more self-explanatory. Therefore, I will only elaborate on a selected few that are worth noting in this report:

```{r, include=FALSE}
### 1. BUSINESS DATA
#Keep relevant business info
business_data <- business_data %>%
  select(business_id, name, city, state, postal_code, stars, review_count, categories, hours)

#Extract hours dataframe
hours <- flatten(business_data$hours)

#Is the business open on weekends (both days)?
hours <- hours %>%
  mutate(wkend_open = ifelse(!is.na(Saturday) & Saturday != "0:0-0:0" & !is.na(Sunday) & Sunday != "0:0-0:0", 1, 0))

#Keep only total_hours and wkend_open
hours <- hours %>%
  select(wkend_open)

#Combine hours with business data
business_data <- cbind(business_data, hours)

```

1. **The “power” of the business’s star rating**. It is commonly understood that high business ratings are not necessarily credible if the review count is low. Therefore, I constructed a `power` matrix where $rating\_power = ln(stars \cdot review\_count)$. The natural logarithm is taken to reduce the value range and hence the variance.

```{r}
business_data <- business_data %>%
  mutate(rating_power = log(review_count*stars, base = exp(1)))
```

2. **Business Category**. The high-level business category consists of 22 categories listed on Yelp’s website. I transformed the string variables of `categories` keywords into this better-structured business category variable.

```{r}
# Define the list of business categories (source: https://blog.yelp.com/businesses/yelp_category_list/)
business_category <- c("Active Life", "Arts & Entertainment", "Automotive", "Beauty & Spas", "Education", "Event Planning & Services", "Financial Services", "Food", "Health & Medical", "Home Services", "Hotels & Travel", "Local Flavor", "Local Services", "Mass Media", "Nightlife", "Pets", "Professional Services", "Public Services & Government", "Real Estate", "Religious Organizations", "Restaurants", "Shopping")
```

# Create a new variable that maps the business category
business_data <- business_data %>%
  mutate(category = map_chr(categories, ~str_extract(., paste(business_category, collapse = "|"))))

```

3. **Business check-ins**. The dates of business check-ins are a series of time stamps when a particular business establishment receives a review. Since check-ins are usually motivated by offers initiated by the businesses, businesses with more frequent check-ins are more likely to receive high reviews as these reviews were incentivised by promotional deals.

```{r, include=FALSE}
#Drop categories & hours
business_data <- business_data %>%
  select(-categories, -hours)

### 2. TIP DATA
# Count the number of tips received by each business
tip_count <- table(tip_data$business_id)

# Convert the table to a data frame
tip_count_df <- as.data.frame(tip_count)
colnames(tip_count_df) <- c("business_id", "tip_count")

# Merge the tip_count with the business_data
business_data <- full_join(business_data, tip_count_df, by = "business_id")

# If a business_id does not have a tip review, replace NA with 0
business_data$tip_count[is.na(business_data$tip_count)] <- 0

```

```{r}
# number of check-ins recorded for each business
checkin_data$checkin_freq <- sapply(checkin_data$date, length)
```

```{r, include=FALSE}
#Drop checkin_data$date
checkin_data <- checkin_data %>%
  select(-date)

#Merge with business_data
business_data <- full_join(business_data, checkin_data, by = "business_id")
business_data$checkin_freq[is.na(business_data$checkin_freq)] <- 0

###REVIEW AND USER DATA
#Keep only y variable (stars rating)
review_data_small <- review_data_small %>%
  select(review_id, user_id, business_id, stars, date)

#Keep relevant user info
user_data_small <- user_data_small %>%
  select(user_id, review_count, yelping_since, elite)

##Transform date variables to dttm format
review_data_small$date <- parse_datetime(review_data_small$date, format = "%Y-%m-%d %H:%M:%S", na = c("", "NA"), locale = default_locale(), trim_ws = TRUE)
user_data_small$yelping_since <- parse_datetime(user_data_small$yelping_since, format = "%Y-%m-%d %H:%M:%S", na = c("", "NA"), locale = default_locale(), trim_ws = TRUE)

## merge review data with user data

```
review_data <- left_join(review_data_small, user_data_small, by = "user_id")
```

4. **Years** ***yelping***. Difference between review date and `yelping_since`.

```{r}
# Create new variable to indicate number of years a user has been yelping
review_data$years_yelping <- as.numeric(difftime(review_data$date,
review_data$yelping_since, units = "weeks")) / 52.25

```

5. **Yelp “Elite”**. According to Yelp, a user is considered an “elite” if they give well-written reviews and high-quality tips, among other criteria. Based on the list of years a user is an “elite”, I constructed a new dummy variable `is_elite` to indicate whether the user is an “elite” at the time of posting the review. This would serve an important indicator that allows the model to weight heavier on an elite’s star rating.

```{r}
# Convert review date to year format
review_data$review_year <- as.numeric(format(as.Date(review_data$date), "%Y"))

# Create a function to check if review year is in elite years
is_elite <- function(elite, review_year) {
  # Check if elite_years is missing
  if (is.na(elite)) {
    return(0)
  }

  elite_years_vector <- as.numeric(str_split(elite, ",")[[1]])
  return(as.integer(review_year %in% elite_years_vector))
}

# Apply the function to each row
review_data$is_elite <- mapply(is_elite, review_data$elite, review_data$review_year)
```

5. **User’s average star ratings by business category**. Although the user’s average star rating is already available on Yelp users' data, I decided to construct this new variable by segmenting the ratings for each business category. This would greatly increase the power of the model.

```{r, include=FALSE}

review_data <- review_data %>%
  select(-elite, -review_year)

# Calculate the average stars for each business_id for elite and non-elite users
avg_stars <- review_data %>%
  group_by(business_id, is_elite) %>%
  summarise(avg_stars = mean(stars, na.rm = TRUE)) %>%
  spread(is_elite, avg_stars)

# Calculate the difference in averages
avg_stars$diff <- avg_stars$"1" - avg_stars$"0"

# Join the difference back to the original data frame
review_data <- review_data %>%
  left_join(avg_stars %>% select(business_id, diff), by = "business_id")

### COMBINE ALL DATA
master_data <- inner_join(review_data, business_data, by = "business_id")

```

````
```{r message=FALSE, warning=TRUE}

# User's average star rating given, by business category
cat_stars <- master_data %>%
  group_by(user_id, category) %>%
  summarise(cat_stars = mean(stars.x, na.rm = TRUE)) %>%
  pivot_wider(names_from = category, values_from = cat_stars)

```
````

Other than these, some variables could have been good predictors but were left out of the dataset due to difficult or overly tedious programming for the purpose of this assignment.

1. **Total weekly opening hours**. I split the daily opening hours into opening and closing times but could not unnest the character matrices that store these variables. Further inspection on the data entry reveals highly irregular time format which makes parsing difficult.

2. **Mapping of the user’s friends**. This could potentially be a strong predictor because the star rating given by a user could be highly influenced by the ratings given by her friends, assuming that friends have similar preferences. However, it is exceptionally programming-heavy to model the social network of each user hence I decided to forgo this variable.

Since I am using the "yelp_user_small.Rda" data file, some user IDs might be missing when mapping the user profile to the review dataset. Therefore, after completing the construction of all relevant variables and merging the datasets, I dropped all observations in the review dataset that has either missing user or business information. The final `master_data` ready for analysis and modelling consists of 173227 observations (down from 1.4 million observations) and 22 variables.

Table below describes some of the important variables in the dataset:

Column 1	Column 2
Variable Name	**Description**
stars.x	The star rating given by a user
review_count.x	The total number of reviews given by a user
years_yelping	The number of years a user has been on Yelp at the time of posting the review
is_elite	Whether the user is an "elite" at the time of posting the review
diff	The difference between the average star rating given by an "elite" and a "non-elite" to a business
stars.y	The business's star rating
review_count.y	The total number of reviews received by a business
wkend_open	Whether the business opens over the weekend
rating_power	The power of a business's star rating based on the total number of reviews received
category	High-level category of the business
tip_count	The total number of tips review received by a business
checkin_freq	The total number of check-ins at the business establishment
cat_stars	The average star rating by category

````
```{r include=FALSE}

# Convert cat_stars to long format
cat_stars_long <- cat_stars %>%
  pivot_longer(cols = -user_id, names_to = "category", values_to = "cat_stars")

# Merge master_data with cat_stars_long
master_data <- master_data %>%
  left_join(cat_stars_long, by = c("user_id", "category"))
````
````

```
#remove rows of missing x variables
master_data <- master_data[complete.cases(master_data), ]

```

Figure below shows the correlation of each of the (numerical) variable to the review star ratings:

```{r, echo=FALSE}
# Potential factors
factors <- master_data[, c("stars.x", "review_count.x", "years_yelping", "is_elite",
"diff", "stars.y", "review_count.y", "wkend_open", "rating_power", "tip_count",
"checkin_freq", "cat_stars")]

correlations <- cor(factors, use = "pairwise.complete.obs")
vars <- rownames(correlations)
cors <- correlations[vars, "stars.x"]
correlation_data <- data.frame(Variable = vars, Correlation = cors)

# Plot the correlations
ggplot(correlation_data, aes(x = reorder(Variable, Correlation), y = Correlation)) +
  geom_bar(stat = "identity") +
  coord_flip() +
  labs(x = "Variable", y = "Correlation with stars.x", title = "Correlations with user's
star ratings")
```

3   Validation and Deployment

3.1   Modelling

I first ran a couple of preliminary linear regressions to identify the best model specification. After multiple iterations, the one main alteration to the model is the removal of the `city` and `postal_code` variables due to insufficient match between training and test data.

However, I remain that the location of a business is an important predictor for user ratings as some places would perform better than others due to the degree of market competitiveness in the area. To circumvent this, I performed a CLRM to identify the significant regions in determining star ratings at the 10% level. The model identified 212 (out of 1304) postcodes and 208 (out of 641) cities that are of statistical significance. I then mapped these lists of significant cities and postcodes to the review data and created two new dummy variables to indicate whether the business is located in these regions.

The final model specification is expressed below:

$$
\begin{align*}
stars.x = & \ \beta_0 + \beta_1 \cdot review\_count.x + \beta_2 \cdot years\_yelping + \beta_3 \cdot is\_elite + \beta_4 \cdot (diff \cdot is\_elite) \\
& + \beta_5 \cdot state + \beta_6 \cdot significant\_city + \beta_7 \cdot significant\_postcode \\
& + \beta_8 \cdot stars.y + \beta_9 \cdot review\_count.y + \beta_{10} \cdot wkend\_open \\
& + \beta_{11} \cdot rating\_power + \beta_{12} \cdot tip\_count + \beta_{13} \cdot checkin\_freq \\
& + \beta_{14} \cdot category + \beta_{15} \cdot cat\_stars + \epsilon
\end{align*}
$$

where ϵ is the error term.

3.2 Evaluation

To recall, the model aims to predict the number of star rating a user would give to a business, based on a number of attributes regarding the business and the user herself. Since the outcome variable is a set of discrete numbers ranging from 1 to 5, a regression model would suffice to make this prediction. I used the `caret` package in R to construct 3 different regression models, namely linear regression, Ridge regression and LASSO regression.

The Ridge and LASSO regression models introduce a penalisation parameter, λ. Taking an empirical approach, I performed k-fold cross-validation using R-squared metric to identify λ with the best performance in the test set. This process is streamlined using the `caret` package.

The table below summarises the best parameters, R-squared, RMSE and normalised RMSE for each of the 3 models:

Model	Ridge	LASSO	Linear
λ	0.1	0.1	
R^2	0.80243	0.79597	0.80309
RMSE	0.59443	0.60774	0.58815
Normalised RMSE	0.14861	0.15193	0.14704
RMSE with rounded predictions	0.61147	0.62105	0.60490
Normalised RMSE with rounded predictions	0.15287	0.15526	0.15122

The predicted star ratings are then rounded to the nearest integer. Since the range of the outcome variable is relatively small, a normalised RMSE where `RMSE / [max(stars.x) - min(stars.x)]` would be a better indicator of the performance of the model. Based on the results, I believe that the model specification is well constructed and that the linear model is marginally the most efficient in predicting users star ratings among the 3 models.

Final Notes

This project employs John Rollin's General Data Science Methodology due to its intuitive workflow. The methodology is closely followed throughout the entire project and is apparent in the structure of this report. The biggest challenge I encountered when carrying out this project was how computationally heavy and time-consuming it is to process data of such dimensions. The amount of time it takes to perform each alteration and iteration has largely exceeded my initial expectations. The time constraint factor has also led to many compromises to the data processing and model specification.

References

Yelp for Business (2023) *The complete Yelp business category list*. Yelp Blog. Available at: <https://blog.yelp.com/businesses/yelp_category_list/>

Data Career (2019) *Ridge and Lasso in R*. Data Career. Available at: <https://www.datacareer.ch/blog/ridge-and-lasso-in-r/>