MSDS 6306 Doing Data Science

Case Study 1 with R and RStudio

Group Members

Member Name	GitHub Username	Project Duties
Peter Flaming	PeterFlaming	Answers to Questions 1:7/README
Brock Friedrich	la-mar	R Code/R Markdown
Samuel K		
Quinton Nixon	qatsmu	Conclusion/Presentation
Matthew Trevathan	mrtrevathan0	Introduction/Purpose of Study

Purose of Case Study 1

The client wishes to invest in the beer brewing industry, and doesn't know the demand of today's consumer, or what type of beer would be high in demand. We were hired to find out what kind of beer is in high demand by the consumer today, and what makes it so favorable to the consumer in a highly competive market. We gathered data from the DDS Case Study 1 database on the nation's breweries and beers focusing our study on the listed levels of alcohol by volume (ABV) and the international bitterness unit (IBU). From these data we found the most desired beer in the U.S. by carefully measuring the correlation of alcohol by volume (ABV) with international bitterness unit (IBU) from the most frequently brewed beer styles across each state in the nation. The resulting list of beers includes the best styles of beers to invest in for our client.

Updates for Case Study 1

A number of changes have been made to this case study for the **Doing Data Science Course** with plans of **Reproducibility within R and RStudio**. Most importantly the case study structure has been planned to take full advantage of relative file paths to reproduce this research.

Current Version

For the current version of this **Case Study** see here (https://github.com/la-mar/DDS_Case_Study_1/blob/master/README.md).

Reproduce the Case Study

Use the following directions to reproduce the data gathering, analysis, and presentation documents.

First download this repository onto your computer.

```
# Set the working directory to this repository as needed for your system
setwd("https://github.com/la-mar/DDS_Case_Study_1")
```

Load and cite R packages

```
# Create list of packages
PackagesUsed <- c("dplyr","tidyr", "knitr", "ggplot2", "maptools", "RColorBrewer", "m
agrittr", "repmis")

# Load PackagesUsed and create .bib BibTeX file
# Note must have repmis package installed.
repmis::LoadandCite(PackagesUsed, file = "Packages.bib", install = TRUE)

# Create package BibTeX file
knitr::write_bib(PackagesUsed, file = "Packages.bib")
```

Data Gathering

Use the following code to reproduce the data gathering:

Import Breweries Data

In this section we load and begin cleaning the data in order to aid our exploratory analysis. Column names are set to lowercase for ease of reading and we begin to summarize the data.

```
#import breweries data
breweries_data <- read.csv("../data/Breweries.csv", header=TRUE)

colnames(breweries_data) %<>% tolower #lower case colnames

breweries_data %<>% rename(brewery_id = brew_id) #rename

#summary of breweries raw data
brewery_summary_raw <- select(breweries_data, state, brewery_id) %>% #select columns
                   dplyr::group_by(state) %>% #group by
                   dplyr::summarize_all(funs(n_distinct(.)))

kable(brewery_summary_raw, digits = 0)

# #TODO: Fix chart
# ggplot(brewery_summary_raw, aes(x=state, color=brewery_id)) +
#     geom_histogram(aes(y=..density..,fill = brewery_id, alpha=0.5), binwidth = 5, s
tat="count") +
#     #stat_count(aes("identity")) +
#     geom_density(alpha=.2, fill="black", color="black") +
#     #facet_grid(. ~ gender) +
#     theme_bw() +
#     theme(legend.position="bottom")
#     #theme(legend.position="none")
#     #theme(text = element_text(size=10), axis.text.x = element_text(angle=90, hjust
=1))
```

Note that you will need to have GNU Make set up on your computer.

Clean Breweries Data

Before we can confidently proceed with our analysis it's important to ensure we have scrubbed the data, removed duplicates, and decide how we will deal with errors and missing values.

We start this process by removing punctuation and whitespace from columns. Humans are fallible and typos are easy to make. Without knowing the origin of the data in the files provided, its prudent to assume that mistakes have been made and take measures to correct them.

Remvoing punctuation allows us to mitigate the possibility of commas being erroneously typed as periods. "Detroit, MI", for example, would be identified as a different city than "Detroit. MI" Removing punctuation resolves this issue.
Both city/state combinations simply become "Detroit MI."

Likewise, it's helpful to remove whitespace. Although whitespace can appear "invisible" to the human eye, computers can "see" this space as if it were a number or a letter.

We use the apply function to make these changes to every row in the dataframe.

Removing duplicates is more of a challenge. Before we can remove duplicates we need to confirm whether or not two rows are the same. We identify duplicates by creating a unique key for each brewery that's a combination of the brewery ID, city, and state.

De-duplicating in this case is a multi-step process. We start by identifying brewery ids that show up more than once which indicate possible duplicates.
Further investigation determines whether or not they are actually duplicates.

In addition to removing identifying and removing duplicates programatically, we also need to correct a few entries manually. There are some entries that are clearly mis-spelled and need to be addressed.

Once potential duplicates are identified and assigned temporary keys, they are evaluated apart from the main dataset and returned to the main dataset once duplicates have been removed.

```
#TODO: Breakup chunk

# remove punctionation from all columns and trim whitespace
breweries_data <- as.data.frame(
                      apply(breweries_data #data set
                            , 2 #apply function column-wise
                            , function(x) trimws(gsub('[[:punct:] ]+',' ',x))) #anony
mous function to remove punctuation and trim whitespace
                            , stringsAsFactors = FALSE)  #do not implicitly convert s
trings to factors

breweries_data$name <- as.factor(breweries_data$name) # convert Name column to factor
breweries_data$brewery_id <- as.integer(breweries_data$brewery_id) # convert Brew_ID
to integer

# confirm Brew_ID + City + State is a unique key
breweries_summary <-
  select(breweries_data, brewery_id, city, state, name) %>%
  group_by(name) %>%
  summarize_all(funs(
    count = n_distinct(brewery_id, city, state))) %>%
  select(name, brewery_id_count) %>% # select only Name and Brew_ID_count columns
  arrange(desc(brewery_id_count)) # sort by Brew_ID_count desc

# capture potential duplicates
breweries_dups <- filter(breweries_summary, brewery_id_count > 1) # if Brew_ID_count
> 1 then there is a potential duplicate on that Brew_ID

# rejoin potential dups to original dataset
breweries_dups <- select(breweries_dups %>% inner_join(breweries_data, by="name"), -e
nds_with("_count"))

# Fix Errors #

# Fix Brew_ID=378, change City(Menominee -> Menominie)
breweries_dups <- breweries_dups %>%
     mutate(City=replace(city, brewery_id==378, "Menominie")) %>%
     as.data.frame()

#TODO: Fix SKs

# Fix Brew_ID=96, change State(MA -> MI)
breweries_dups <- breweries_dups %>%
     mutate(State=replace(state, brewery_id==96, "MI")) %>%
     as.data.frame()

#capture known duplicates
breweries_dups <- breweries_dups %>%
                  group_by(name, city, state) %>%
                  filter(n()>1)

#create surrogate key for duplicates
breweries_sk <- breweries_dups %>%
                    group_by(name, city, state) %>%
                    summarize_all(funs(
                        brew_sk = (sum(brewery_id)*sum(brewery_id)),
                        count = n()
                        )) %>% #end summarize_all
                    ungroup() %>%
                    right_join(breweries_dups, by = c("name", "city", "state")) %>% #
rejoin to dupes by name, city, state
                    select(brewery_id, brewery_id_brew_sk)

breweries_data$brewery_id[(breweries_data$brewery_id %in% breweries_sk$brewery_id)] <
- breweries_sk$brewery_id # update Brew_ID in original dataset

breweries_clean <- distinct(breweries_data, brewery_id, .keep_all = TRUE) %>% rename(
brewery_name = name) # select distinct breweries according to the unique composite ke
y and rename
```

Import Breers Data

```
#import beers data
beer_data <- read.csv("../data/Beers.csv", header=TRUE)

#check the import of beers data
head(beer_data)
```

Clean Beers Data

A similar process is used to remove duplicates from the Beers dataset.

```
colnames(beer_data) %<>% tolower #lower case colnames

beer_data$brewery_id[(beer_data$brewery_id %in% breweries_sk$brewery_id)]  <- breweri
es_sk$brewery_id_brew_sk # update brewery_ids from brewery_sk data

beer_clean <- distinct(beer_data)#%>% rename(Brew_ID = Brewery_id, Beer_Name = Name)
#

# kable(as.data.frame(summarytools::descr(beer_clean)),digits = 2)
```

Analysis & Presentation Documents

Use the following code to reproduce the analysis and presentation documents:

```
# Set the working directory to this repository as needed for your system
setwd("https://github.com/la-mar/DDS_Case_Study_1/r_code")
```

Loading required libraries

The following code loads useful libraries that aren't included in base R. The of these libraries come from the "tidyverse" including dplyr for manipulating dataframes, tidyr for making data tidy, knitr for creating reproducible documents ggplot2 for plots, maps for help with geographic plots, RColorBrewer…

```
# keep echo=FALSE, tidy=TRUE

require(dplyr)
require(tidyr)
require(knitr)
require(ggplot2)
require(maps)
require(RColorBrewer)
require(summarytools)
require(magrittr)
#automatically set working directory to the directory containing this R script

opts_chunk$set(results='asis') #table format option
```

Render the Analysis Rmarkdown for reference

```
# Website.html
rmarkdown::render("Analysis_Final.Rmd")
```

CodeBook of Required Libraries

```
library(dplyr)
library(tidyr)
library(knitr)
library(ggplot2)
library(maps)
library(RColorBrewer)
library(summarytools)
library(magrittr)
```

CodeBook of Object Variables

```
breweries_data <- read.csv("../data/Breweries.csv", header=TRUE)
```

```
brewery_summary_raw <- select(breweries_data, state, brewery_id) %>% #select columns
                   dplyr::group_by(state) %>% #group by
                   dplyr::summarize_all(funs(n_distinct(.)))
```

```
breweries_data <- as.data.frame(
                      apply(breweries_data #data set
                            , 2 #apply function column-wise
                            , function(x) trimws(gsub('[[:punct:] ]+',' ',x)))
                            , stringsAsFactors = FALSE)  #do not implicitly convert s
trings to factors
```

```
breweries_data$name <- as.factor(breweries_data$name)
```

```
breweries_data$brewery_id <- as.integer(breweries_data$brewery_id)
```

```
breweries_summary <-
  select(breweries_data, brewery_id, city, state, name) %>%
  group_by(name) %>%
  summarize_all(funs(
    count = n_distinct(brewery_id, city, state))) %>%
  select(name, brewery_id_count) %>% # select only Name and Brew_ID_count columns
  arrange(desc(brewery_id_count)) # sort by Brew_ID_count desc
```

```
breweries_dups <- filter(breweries_summary, brewery_id_count > 1)
```

```
breweries_dups <- select(breweries_dups %>% inner_join(breweries_data, by="name"), -e
nds_with("_count"))
```

```
breweries_dups <- breweries_dups %>%
     mutate(City=replace(city, brewery_id==378, "Menominie")) %>%
     as.data.frame()
```

```
breweries_dups <- breweries_dups %>%
     mutate(State=replace(state, brewery_id==96, "MI")) %>%
     as.data.frame()
```

```
breweries_dups <- breweries_dups %>%
                  group_by(name, city, state) %>%
                  filter(n()>1)
```

```
breweries_sk <- breweries_dups %>%
                    group_by(name, city, state) %>%
                    summarize_all(funs(
                        brew_sk = (sum(brewery_id)*sum(brewery_id)),
                        count = n()
                        )) %>% #end summarize_all
                    ungroup() %>%
                    right_join(breweries_dups, by = c("name", "city", "state")) %>%
                    select(brewery_id, brewery_id_brew_sk)
```

```
breweries_data$brewery_id[(breweries_data$brewery_id %in% breweries_sk$brewery_id)] <
- breweries_sk$brewery_id
```

```
breweries_clean <- distinct(breweries_data, brewery_id, .keep_all = TRUE) %>% rename(
brewery_name = name)
```

```
beer_data <- read.csv("../data/Beers.csv", header=TRUE)
```

```
beer_data$brewery_id[(beer_data$brewery_id %in% breweries_sk$brewery_id)]  <- breweri
es_sk$brewery_id_brew_sk
```

```
beer_clean <- distinct(beer_data) %>% rename(Brew_ID = Brewery_id, Beer_Name = Name)
```

```
state_ll <- read.csv("../data/state_coords.csv") %>%
                    mutate(State = toupper(State)) %>%
                    rename(name = State, lat_center = Latitude, lon_center = Longitud
e)
```

```r
states <- map_data("state") %>%
          mutate(region = toupper(region)) %>%
          rename(name=region) %>%
          select(long, lat, name, group)
```

```r
states <- states %>%
          left_join(
            states %>%
            group_by(name) %>%
            summarise_all(funs(n=n())) %>%
            select(name, group_n) %>%
            distinct(name, .keep_all = TRUE)
          )
```

```r
breweries_by_state <- select(breweries_clean, brewery_id, state) %>%
  group_by(state) %>%
  summarise_all(funs(brewery_count = n()))  %>%
  left_join(state_ll, by=c("state" = "Abbr"))
```

```r
breweries_geo <- breweries_by_state %>%
                  inner_join(states, by = c("name" = "name"))
```

```r
merged_data <- breweries_clean %>%
               full_join(beer_clean, by="brewery_id")
```

```r
merged_by_state <- select(merged_data, state, abv, ibu) %>%
                   group_by(state) %>%
                   summarise_all(median, na.rm = TRUE)
```

```r
merged_by_state$state <- as.factor(merged_by_state$state)
```

```r
max_abv <-  (select(merged_data, state, abv) %>%
                   group_by(state) %>%
                   #filter(ABV == max(ABV)) %>%
                   arrange(desc(abv))  %>% #sort by ABV
                   filter(row_number() == 1))[1,] #get first row
```

```r
max_ibu <-  (select(merged_data, state, ibu) %>%
                   group_by(state) %>%
                   #filter(ABV == max(ABV)) %>%
                   arrange(desc(ibu))  %>% #sort by ABV
                   filter(row_number() == 1))[1,] #get first row
```

```
abv_stats <- as.data.frame(t(summary(merged_data$abv))) %>% #summarize and transpose
             rename("abv"=Freq, Statistic=Var2) %>%
             select(Statistic, abv)
```

```
abv_stats$abv <- round(abv_stats$abv, digits = 3)
```

```
styles <- beer_clean %>%
              distinct(Beer_ID, Style, IBU, ABV, Ounces) %>%
              arrange(Style) %>%
              na.omit(IBU, ABV)
```

```
y <- quantile((styles$IBU %>% na.omit()), c(0.25, 0.75))
x <- qnorm(c(0.25, 0.75))
slope <- diff(y)/diff(x)
y_int <- y[1] - slope * x[1]

qq_ibu <- ggplot(styles, aes(sample = styles$IBU)) +
              geom_qq(shape = 16, size = 2, alpha = 0.5) +
              geom_abline(slope = slope, intercept = y_int, colour ='red', size = 1)
+
              ggtitle("QQ-plot of IBU") +
              theme_minimal()  +
              theme(plot.title = element_text(hjust = 0.5))
```

```
y <- quantile((styles$ABV %>% na.omit()), c(0.25, 0.75))
x <- qnorm(c(0.25, 0.75))
slope <- diff(y)/diff(x)
y_int <- y[1] - slope * x[1]

qq_abv <- ggplot(styles, aes(sample = styles$ABV)) +
            geom_qq(shape = 16, size = 2, alpha = 0.5) +
            geom_abline(slope = slope, intercept = y_int, colour ='red', size = 1) +
            ggtitle("QQ-plot of ABV") +
            theme_minimal() +
            theme(plot.title = element_text(hjust = 0.5))
```

```
hist_abv <- ggplot(styles) +
              geom_histogram(aes(x=ABV)) +
              theme(text = element_text(size=10),
                  axis.text.x = element_text(angle=90, hjust=1))
```