Analytics & Business Intelligence Excercise

Benjamin Riesser

December 1, 2018

```
##load packages
library(tidyverse)
```

```
## -- Attaching packages --------------------------------------- tidyverse 1.2.1 --

## v ggplot2 3.0.0     v purrr   0.2.5
## v tibble  1.4.2     v dplyr   0.7.6
## v tidyr   0.8.1     v stringr 1.3.1
## v readr   1.1.1     v forcats 0.3.0

## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
```

```
library(ggplot2)
library(kableExtra)
library(pander)
library(ggthemes)
library(gridExtra)
```

```
##
## Attaching package: 'gridExtra'

## The following object is masked from 'package:dplyr':
##
##     combine
```

```
library(corrr)


##set wd
setwd("C:/Users/Ben Riesser/Desktop/GC_assignment")

##load data set
reviews <- read_csv("C:/Users/Ben Riesser/Desktop/GC_assignment/beer_reviews.tar/beer_reviews/beer_revi
```

```
## Parsed with column specification:
## cols(
##   brewery_id = col_integer(),
##   brewery_name = col_character(),
##   review_time = col_integer(),
##   review_overall = col_double(),
##   review_aroma = col_double(),
##   review_appearance = col_double(),
##   review_profilename = col_character(),
##   beer_style = col_character(),
##   review_palate = col_double(),
##   review_taste = col_double(),
##   beer_name = col_character(),
##   beer_abv = col_double(),
##   beer_beerid = col_integer()
## )
```

```
###add a function
nonNAs <-  function(x) {
  as.vector(apply(x,2,function(x) length(which(!is.na(x)))))
}


#missing data table
missing_data <- data.frame(
  names <- as.vector(names(reviews)),
  reviews_nonNA = length(reviews$review_overall)-nonNAs(reviews)
)
missing_data_names <- c("Variable Name", "Number of NAs")
names(missing_data) <- missing_data_names

missing_abv <- missing_data %>%
  filter(`Variable Name` == "beer_abv") %>%
  select(`Number of NAs`)
missing_abv <- missing_abv[1,1]
percent_of_missing_abv <- round((missing_abv/length(reviews$beer_abv)*100),2)




##exploratory data mining
##the following table helped me to understand the variables and how the data are set up.
unique_beer_ids <- length(unique(reviews$beer_beerid))
unique_beer_names <- length(unique(reviews$beer_name))
unique_brewery_ids <- length(unique(reviews$brewery_id))
unique_brewery_names <- length(unique(reviews$brewery_name))
unique_brewery_ids_noformat <- length(unique(reviews$brewery_id))

diff_beer_id_names <- unique_beer_ids - unique_beer_names
diff_brewery_id_names <- unique_brewery_ids - unique_brewery_names


unique_beer_ids <- formatC(length(unique(reviews$beer_beerid)), format="d", big.mark=",")
unique_beer_names <- formatC(length(unique(reviews$beer_name)), format="d", big.mark=",")
unique_brewery_ids <- formatC(length(unique(reviews$brewery_id)), format="d", big.mark=",")
unique_brewery_names <- formatC(length(unique(reviews$brewery_name)), format="d", big.mark=",")
diff_beer_id_names <- formatC(diff_beer_id_names, format="d", big.mark=",")
diff_brewery_id_names <- formatC(diff_brewery_id_names, format="d", big.mark=",")

find_beers_with_extra_ids <- reviews %>%
  distinct(beer_beerid, .keep_all = TRUE) %>%
  group_by(brewery_name, beer_name) %>%
  select(brewery_name, beer_name, beer_beerid) %>%
  summarise(n = n()) %>%
  arrange(desc(n)) %>%
  filter(n == 5)

find_brewerys_with_extra_ids <- reviews %>%
  distinct(brewery_id, .keep_all = TRUE) %>%
  group_by(brewery_name) %>%
  select(brewery_name, brewery_id) %>%
  summarise(n = n()) %>%
  filter(n > 1) %>%
  arrange(desc(n)) %>%
  filter(n >= 7)



Hops_Grillhouse <- reviews %>%
  distinct(beer_beerid, .keep_all = TRUE) %>%
  group_by(brewery_name, beer_name) %>%
  select(brewery_name, brewery_id, beer_name, beer_beerid) %>%
  filter(brewery_name == "Hops Grillhouse & Brewery") %>%
  filter(beer_name == "Alligator Ale")
```

I am very pleased to present my analysis of this dataset. I have included two documents. First, a finished PDF, created by using the Rmarkdown language, out of R studio. Second, the same document is included a second time, only including printout of the R code that produced the document. Please note that the code includes many comments throughout. My intend is that the comments with help to illustrate my approach with the R language.

Exploration and Cleaning of the dataset

Origins of the dataset
I would like to include a summary of my general approach to this type of analysis. First, when looking at a dataset which I have never explored before, I attempt to understand how the dataset was created. A quick internet search brings us to the Beer Advocate website. Looking over the review process helped me to understand how the data were collected and the meaning of some of the variables.
Cleaning and reformatting
Second, I proceeded with the task of data cleaning. The initial task is to determine the magnitude of the missing data, and how that may affect conclusions of the analysis. In the case of the Beer Advocate dataset, there are only a few variables where there are missing values. The table below summarizes this.

```
missing_data %>%
pander()
```

Variable Name	Number of NAs
brewery_id	0
brewery_name	15
review_time	0
review_overall	0
review_aroma	0
review_appearance	0
review_profilename	348
beer_style	0
review_palate	0
review_taste	0
beer_name	0
beer_abv	67785
beer_beerid	0

Other than a small number of missing data, the dataset is very clean and ready for analysis. We can note that there are only a few missing brewery names and profile names. However, 4.27% of reviews are missing

ABV%s. This missing data could affect the conclusions we arrive at for the first question in the exercise.
The next task in the data cleaning process is to determine the "unit of analysis". I noticed that there are 66,055 unique beer ids, and 56,857 beer names. There are 9,198 more ids than beer names. Likewise, there are 5,840 Brewery ids and 5,743 Brewery names, a difference of 97. The difference in beers can be easily explained. Many different breweries may name a beer with the same name as other breweries. For example, many breweries have an "Oktoberfest" seasonal beer.
However, before we can answer the exercise questions, we must make a choice about the brewery names that have more than one id. There are two explanations. First, there are two or more breweries with the same name. Second, that there is some sort of error in the id assignment process, and a single brewery is getting more than one id.
After running further exploratory tables on the dataset, I have decided to use the beer_id variable as the unit of analysis. The table below, showing information for the brewery "Hops Grillhouse & Brewery", helps to explain this decision. The table shows that there must have been five different locations of the same brewery, each brewing their own version of the "Alligator Ale", which each receives its own beer id.

```
Hops_Grillhouse %>%
  pander()
```

brewery_name	brewery_id	beer_name	beer_beerid
Hops Grillhouse & Brewery	6565	Alligator Ale	32704
Hops Grillhouse & Brewery	1095	Alligator Ale	47118
Hops Grillhouse & Brewery	4214	Alligator Ale	30720
Hops Grillhouse & Brewery	12807	Alligator Ale	40788
Hops Grillhouse & Brewery	1932	Alligator Ale	25412

The importance of the data cleaning, data exploration, and the re-formatting task cannot be overstated. In the present exercise, the data exploration is important because we need to know the unit to use for question #1. If we sorted our data by brewery_name, as opposed to brewery_id, we may not determine the most accurate answer for question #1.

```
####before we can answer question 1, we need to look at how the missing data will affect the conclusion
missing_abv <- reviews %>%
  filter(is.na(beer_abv)) %>%
  distinct(beer_beerid, .keep_all = TRUE) %>%
  group_by(brewery_id) %>%
  summarise(missing_data= n())
count_abv <- reviews %>%
  group_by(brewery_id) %>%
  distinct(beer_beerid) %>%
  summarise(number_of_beers = n())
missing_abv <- count_abv %>%
  left_join(missing_abv, by = "brewery_id") %>%
  mutate(missing_data = replace_na(missing_data, 0)) %>%
  mutate(percent_na = missing_data/(number_of_beers)) %>%
  mutate(percent_na = round(percent_na, 3)*100)
summary_percent_na <- summary(missing_abv$percent_na)

number_of_bwry_no_abv_reported <- missing_abv %>%
  filter(percent_na == 100)

number_of_bwry_no_abv_reported <- length(number_of_bwry_no_abv_reported$percent_na)
percent_of_bwry_no_abv_reported <- round((number_of_bwry_no_abv_reported/unique_brewery_ids_noformat),3
```

```r
plot_histo_missing_abv <- ggplot(missing_abv, aes(percent_na)) +
  geom_histogram() +
  theme_hc() +
  scale_fill_manual(values = c("#f6ce00","#103b84")) +
  xlab("Percent of missing ABV% by brewery")

#we find that 10.4% of the dataset (brewries) is un-useable for this question.

#create look-up table to help complete a final table table.
brewery_lookup <- reviews %>%
  distinct(brewery_id, .keep_all = TRUE) %>%
  select(brewery_id, brewery_name)



####create table to answer question 1####
strongest_abv <- reviews %>%
  group_by(brewery_id) %>%
  filter(!is.na(beer_abv)) %>%
  distinct(beer_beerid, .keep_all = TRUE) %>%
  summarise(average_abv = mean(beer_abv), number_of_beer_ids = n(), sd = sd(beer_abv)) %>%
  arrange(desc(average_abv)) %>%
  left_join(brewery_lookup, by = "brewery_id")

#create binary variable to show strongest brewey for graphing.
max_abv_bwry <- max(strongest_abv$average_abv)

strongest_abv <- strongest_abv %>%
  mutate(strongest_beer = if_else(average_abv == max_abv_bwry,
                                  "Strongest Brewery",
                                  "Other Breweries"))

head_strongest_abv <- head(strongest_abv)

#check to see if there are duplicate brewery names.
unique_brewerys_strongest_abv_list <- length(strongest_abv$brewery_id) -   length(unique(strongest_abv$
#no, no overlap here, we would have gotten the same results from using "brewery_name"

#plot solution to question 1
plot_point_ABV <- ggplot(strongest_abv, aes(average_abv, number_of_beer_ids, colour = strongest_beer))
 geom_point(size = 5) +
  theme_hc(base_size = 10) +
  scale_color_manual(values = c("#f6ce00","#103b84")) +
  xlab("Average ABV% of beers for each brewery") +
  ylab("Number of beers Reviewed")+
   theme(legend.title=element_blank())

plot_histo_ABV <- ggplot(strongest_abv, aes(average_abv, fill = strongest_beer)) +
 geom_histogram() +
  theme_hc(base_size = 10) +
  scale_fill_manual(values = c("#f6ce00","#103b84")) +
  theme(legend.title=element_blank()) +
  xlab("Average ABV% by brewery")
```

```r
#create values to answer question 1.
strongest_bwry_abv_table <- strongest_abv %>%
  filter(average_abv == max_abv_bwry) %>%
  select(`Name of Brewery` = brewery_name, `Average ABV%` = average_abv, sd) %>%
  t()
strongest_bwry_abv_name <-  as.vector(  strongest_bwry_abv_table[1,1] )

#t-test for the data
strongest_abv_ttest <- reviews %>%
  group_by(brewery_id) %>%
  filter(!is.na(beer_abv)) %>%
  distinct(beer_beerid, .keep_all = TRUE) %>%
  select(brewery_id, brewery_name, beer_abv, beer_beerid, beer_name) %>%
  mutate(strongest_bwry_id = if_else(brewery_id == 6513, 1,0)) %>%
  mutate(strongest_bwry_id = as.character(strongest_bwry_id))

dif_means_abv <- t.test(beer_abv ~ strongest_bwry_id, data = strongest_abv_ttest)

#Extract normal t-test stuff
abv.p <- dif_means_abv$p.value

#round p
abv.p <- round(abv.p,4)

t <- dif_means_abv$statistic
df <- dif_means_abv$parameter

#round t
t <- round(t,3)

#put some text into objects
abv.stars <- c("**")

#Put df and t into a vector together using paste()
df.t <- paste(t,df, sep = ", ")

#The names
my.rownames <- c("P value",
              "P value summary",
              "Are means significantly different (P<0.05)",
              "t and df")

#the info
statoutput <- c(abv.p,abv.stars,"Yes",df.t)

#put into dataframe
table30.2 <- data.frame(my.rownames,statoutput)


#change names of table
names(table30.2) <- c("UNPAIRED t TEST","")
```

Question (1) - Which brewery produces the strongest beers by ABV%?
If we look back at the missing data table, earlier in the document, we notice that the ABV% variable has the highest number of missing data points. In fact, 4.27% of reviews are for beer which no ABV% is reported. We need to consider that some breweries do not report this. I found that 608 breweries had no reported ABV% at all. So, 10.4% of the breweries are not included in the analysis for question 1. Therefore, when we come to a final conclusion, we must modify the question to: "Of the breweries that reported ABV%, which brewery produces the strongest beers by ABV%?"
Likewise, we should consider that some breweries only reported ABV% for a percentage of their beers. The table below shows the quartile summary for this variable. Also, a histogram of this data is shown. While the rest of the dataset is very clean, the amount of missing data should lead us to make an qualified conclusion to question 1. We can see the vast majority of breweries have ABV% reported, but the number of missing data should be noted.

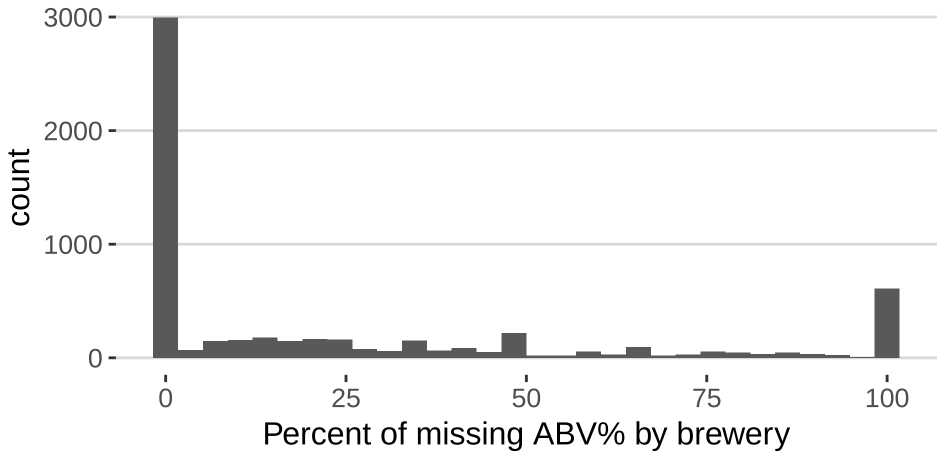
```
summary_percent_na %>% pander()
```

Min.	1st Qu.	Median	Mean	3rd Qu.	Max.
0	0	0	24.18	40	100

Histogram Showing the distribution of percent missing data

```
plot_histo_missing_abv
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



After filtering the data to a format of a single brewery id with the number of different beers, and the average ABV% of their beers, we find that a very distinct leader emerges. Schorschbräu has the beers with the highest ABV%.

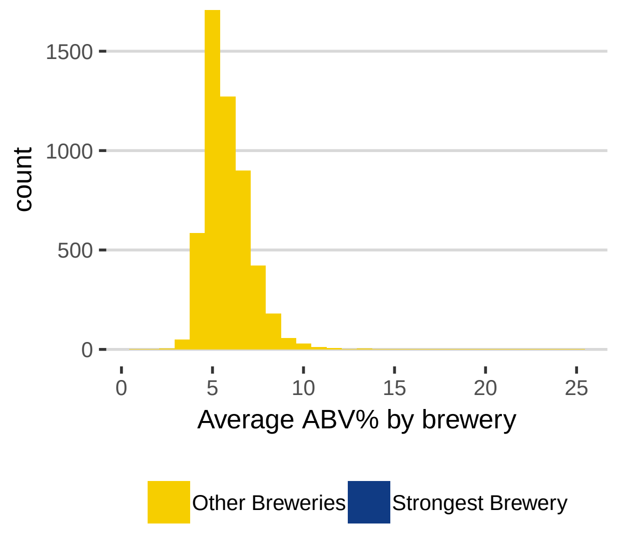
```
strongest_bwry_abv_table %>%
  pander()
```

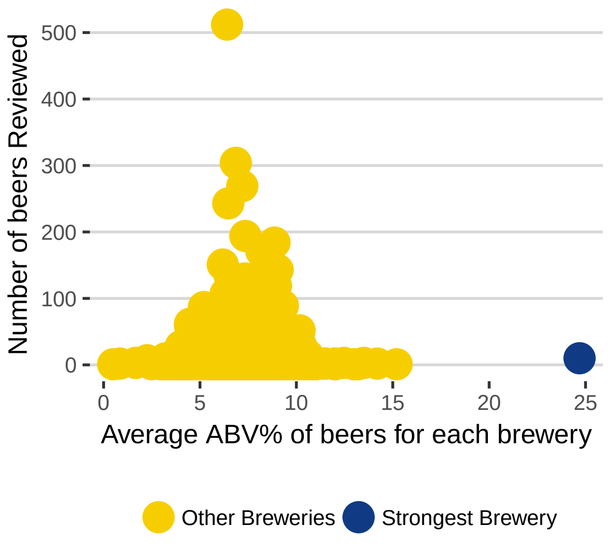
Name of Brewery	SchorschbrÃ¤u
Average ABV%	24.69
sd	17.10488

The two plots below visualize the strong difference in average ABV% for Schorschbräu. The left is a histogram. It is less affective in communicating this fact, as the Schorschbräu brewery is all the way to the right tail by its self. The right plot adds the "Number of Beers" variable on the y-axsis. This improves the plot's readability. And, moreover, could lead us to addition questions about the dataset. For example: What brewery is producing 500 different beers to be review?

```
plot_abv <- grid.arrange(plot_histo_ABV, plot_point_ABV, nrow = 1)
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```





Not only does this brewery have the strongest beers, when we run a difference or means t.test we find that the difference in means is statistically significant. It is not by chance that this brewery produces that highest ABV% beer.

Results of T.Test

```r
table30.2 %>%
  pander(justify ="left")
```

UNPAIRED t TEST	
P value	0.0078
P value summary	**
Are means significantly different (P<0.05)	Yes
t and df	-3.406, 9.00005127774281

```r
#create look up table that will be used later to add brewery name and id to summarised table
beerid_to_brewey_id_lookup <- reviews %>%
  select(beer_beerid, brewery_id, brewery_name, beer_name) %>%
  distinct(beer_beerid, .keep_all = TRUE)

#created combined score variable and filter for top three
top_three <- reviews %>%
  rowwise() %>%
  mutate(combined_score = sum(review_overall, review_aroma, review_taste, review_appearance, review_pala
  group_by(beer_beerid) %>%
  summarise(avg_combinded_score = mean(combined_score))  %>%
  top_n(3,  avg_combinded_score) %>%
  left_join(beerid_to_brewey_id_lookup, by = "beer_beerid")

#created list of breweries with a beer that has a perfect score.
bwry_with_perfect_beers <- top_three %>%
  distinct(brewery_id)

#create combined average for breweries that have a beer with a perfect score.
```

```r
additional_factor <- reviews %>%
  filter(brewery_id %in% bwry_with_perfect_beers$brewery_id) %>%
  rowwise() %>%
  mutate(combined_score = sum(review_overall, review_aroma, review_taste, review_appearance, review_pala
  group_by(brewery_id) %>%
  summarise(avg_combinded_bwry_score = mean(combined_score))

#create top that account for the score of the brewery as well as the score of the beer.
top_three_refined <- top_three  %>%
  left_join(additional_factor, by = "brewery_id") %>%
  top_n(3, avg_combinded_bwry_score) %>%
  select(beer_name, `Average combinded score for top beer` = avg_combinded_score, brewery_name, `Averag

#find number of files that has four five reviews and an overall of less than 5
four_fives<- reviews %>%
  filter(review_aroma == 5 &
           review_appearance == 5 &
           review_palate == 5 &
           review_taste == 5)  %>%
  filter(review_overall < 5) %>%
  count()
```

Question (2) - If you had to pick 3 beers to recommend using only this data, which would you pick?
The dataset has provided us with information to make an educated recommendation of three beers. The dataset includes 5 metrics that rate each beer and, many reviews for each beer in many cases. Each metric is rated from 0 to 5. As we will see in answering Question (3), all of the review variables are highly correlated. However, the Overall review variable is not a function of the other four. In fact, strangely enough, there are 1,137 cases in the data where the aroma, appearance, palate, and taste reviews are all rated at a 5 and the overall is rate less than a 5. There is no reason not to simply take a sum of the five-independent metrics to make our choice. I have no present information that would make me think I needed to weigh the variables differently. A 25 would be a perfect score.
This approach leads us to a case where there are 23 beers in the dataset that have a combined score of 25. As a result, an additional factor will be needed to make a determination. Here, I have included the average combined beer score of each of the 23 breweries whom have a beer with a perfect score. The highest ranking breweries with a perfect scoring beer made the top thrree list. The result is shown in the table below.

```r
top_three_refined %>%
  pander(split.table = Inf)
```

beer_name	Average combinded score for top beer	brewery_name	Average combinded score all beers at brewery
Edsten Triple-Wit	25	Edsten Brewing Company	24.5
Old Incinerator Brandy Barrel Barleywine	25	Rascal Creek Brewing Co.	25
KoutskÃ¡ DesÃtka	25	Pivovar Kout Na Å umave	23

This answer requires further discussion. While I have chosen to complete Question (2) in the above manner, there are certainly more ways to do this, resulting in different conclusions. The most concerning flaw in the method I have chosen is that my analysis neglects to take the number of “reviews per beer” into account. This is essentially ignoring the standard deviation of the averages. Certainly, it would be harder to get a perfect average when there are 1000 reviews, as opposed to only a few. I have chosen this simplification of

analysis due to the fact that I do not want to account for popularity of a brewery in the choice. For example, if a brewery on the list is located in a large metropolitan area, there may be more reviews for each beer. This would not necessarily mean that brewery should be more likely to end up with a beer on the final list of three. The analysis I have offered accounts only for the ratings of the beers, then, falls back on the rating of the brewery of which the beer was brewed.
Another approach could have been to give profile accounts more weight if the reviewer entered more reviews. But, again, with the available data, we have no way of knowing if we should be more confident in the validity of the reviewers with more reviews, or if they simply have a drinking problem. Again, my analysis, I think, accounts only for the available data, relaying heavily on the rating system.

```
###which of the factors are most important in determining the overall quality of a beer?
cor_set <- reviews %>%
  select(review_overall, review_taste, review_aroma, review_appearance, review_palate)
cor1 <- round(cor(cor_set),3)


networkplot <- cor_set %>%
  correlate() %>%
  network_plot(min_cor = 0.3, colours = c( "#f6ce00","#103b84"))
```

```
##
## Correlation method: 'pearson'
## Missing treated using: 'pairwise.complete.obs'
```

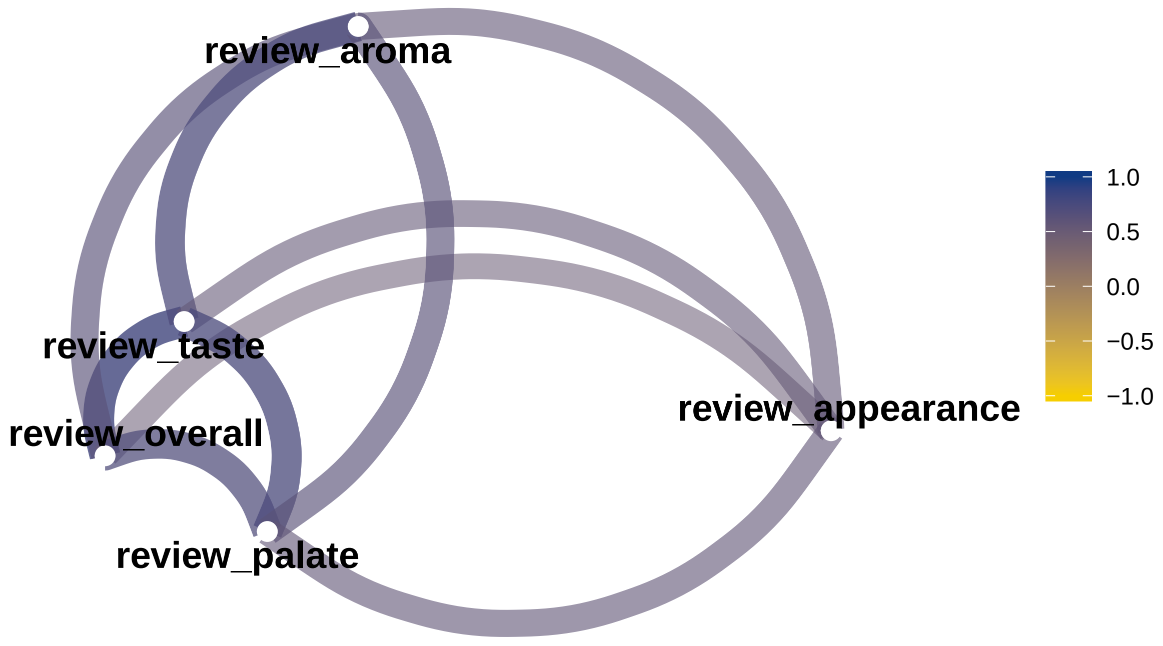
Question (3) - Which of the factors (aroma, taste, appearance, palette) are most important in determining the overall quality of a beer?
This question can be answered with a simple correlation matrix. Also, with this problem we can use all observations (reviews), so there is no need to reformate the data. Below is a summary of the resulting correlation matrix. Clearly, the review metric of taste is the most influential to reviewer's scoring of the Overall metric, with a correlation coefficient of .70. What is unexpected is that mouthfeel seems to be highly corelated with the overall score as well. Who would have thought?

```
cor1 %>%
  pander(split.table = Inf)
```

	review_overall	review_taste	review_aroma	review_appearance	review_palate
review_overall	1	0.79	0.616	0.502	0.702
review_taste	0.79	1	0.717	0.547	0.734
review_aroma	0.616	0.717	1	0.561	0.617
review_appearance	0.502	0.547	0.561	1	0.567
review_palate	0.702	0.734	0.617	0.567	1

Normally, with a smaller dataset, I would plot a correlation with a matrix of scatter plots. Scatter plots, I think, show relationships nicely. However, with this large dataset, a scatter plot would not make much sense. And, moreover, would be too much of a drain on the system's processor.
I have plotted the correlation with a network plot instead. This plot uses color and proximity of the variable names to show the strength of the correlation. You can see that the taste and palate variables are in close proximity to the Overall variable. Appearance is the least correlated. Also notice that these variables, unsurprisingly, are positively correlated. The obvious conclusion is that is a reviewer who likes a beer will score all of the metrics higher.

```
networkplot
```



```
#created combined score variable and filter for the aroma and appearance variables
aroma_appearance_combinded <- reviews %>%
  rowwise() %>%
  mutate(combined_score = sum(review_overall, review_aroma,  review_appearance))  %>%
  group_by(beer_style) %>%
  summarise(avg_combinded_score = mean(combined_score), n = n()) %>%
  top_n(1,  avg_combinded_score) %>%
  rename(`Beer Style` = beer_style, `Average Combinded Score` = avg_combinded_score )
max_aroma_appearance <- max(aroma_appearance_combinded$avg_combinded_score)


#find the beer that the interviewer may not enjoy.
aroma_appearance_combinded_bottom <- reviews %>%
  rowwise() %>%
  mutate(combined_score = sum(review_overall, review_aroma,  review_appearance))  %>%
  group_by(beer_style) %>%
  summarise(avg_combinded_score = mean(combined_score), n = n()) %>%
  top_n(-1,  avg_combinded_score)

#create plots
aroma_appearance_plot_table <- reviews %>%
  group_by(beer_style) %>%
  summarise(avg_aroma = mean(review_aroma), avg_appearance = mean(review_appearance)) %>%
  mutate(high_combinded_score = if_else(beer_style == "American Double / Imperial Stout",
        "American Double / Imperial Stout",
        "Other Styles of beer"))

aroma_appearance_plot <- ggplot(aroma_appearance_plot_table, aes(avg_aroma, avg_appearance, colour = hi
  geom_point(size = 3) +
  theme_hc(base_size = 10) +
  scale_color_manual(values = c("#103b84", "#f6ce00")) +
  xlab("Average of Aroma") +
  ylab("Average of Appearance")+
   theme(legend.title=element_blank())

aroma_appearance_plot2 <- ggplot(aroma_appearance_plot_table, aes(avg_aroma, avg_appearance, colour = hi
  geom_point(size = 3) +
  theme_hc(base_size = 10) +
  xlim(3.75,4.2) +
  ylim(3.75,4.2) +
  geom_text(aes(label=ifelse(avg_appearance > 4.03 & avg_aroma > 4 ,as.character(beer_style),'')), hjus
  scale_color_manual(values = c("#103b84", "#f6ce00")) +
  xlab("Average of Aroma") +
  ylab("Average of Appearance") +
   theme(legend.title=element_blank())
```

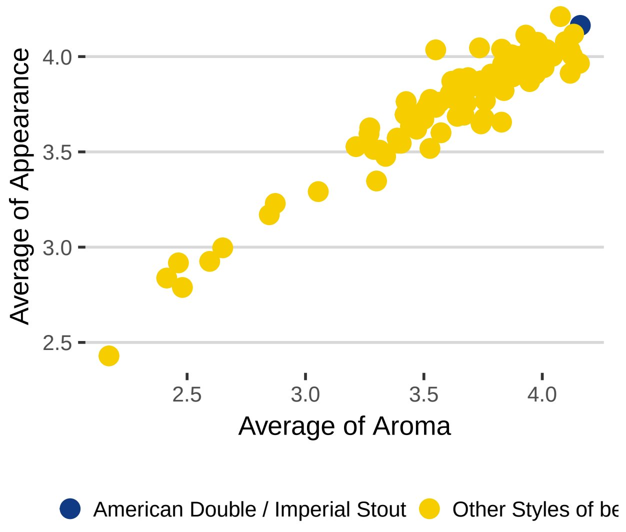
Question (4) - Lastly, if I typically enjoy a beer due to its aroma and appearance, which beer style should I try? With Question (4) I took a similar approach as with Question (2). The analysis is dependent on the validity of the review scores. First, I created a combined score of aroma and appearance. Next, I found the average for each beer type. There are only 104 styles of beers in the dataset. The beer you should enjoy is American Double / Imperial Stout.

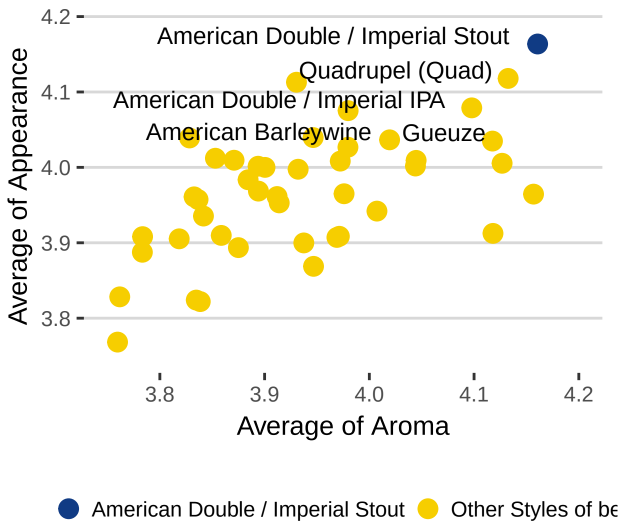
```
aroma_appearance_combinded %>%
  pander()
```

Beer Style	Average Combinded Score	n
American Double / Imperial Stout	12.35	50705

We can also plot this relationship. If we take the 104 beer styles, and find the average ratings for aroma and appearance. We can make a scatter plot with Aroma along the x-axsis, and Appearance along the y-axsis. The plots below are the result. On the left plot shows the entire scale of the metrics; from 1 to 5. The blue point is the suggested beer, American Double / Imperial Stout. The dot on the furthest bottom left is "Light Lager." This is the beer you should stay away from according to the ratings.
The plot on the right is a scatter plot that is zoomed in to the top right of the right hand plot. This shows other beer styles that may be appealing as well: Quadrupel, American Double/Imperial IPA, Gueuze, and the American Braleywine.

```
plot_aroma_appearance_3 <- grid.arrange(aroma_appearance_plot, aroma_appearance_plot2, nrow = 1)
```





Final notes I hope that my analysis of this dataset has been helpful. Every time I think I have a robust knowledge of the R programming language and its implementation in Data Science, I learn something new that changes that way I approach problems and how I use R to find solutions. Currently I am spending a large amount of time moving my use of "base" functions to the "tidyverse" framework of programing in R. I also am working on more intensive graphing options, GIS implementation of R, and use of shiny to create web-applications.