STA380_James_Problems

Grayson Merritt

2024-07-31

Problem 1

We are looking for P(Yes|TC) P(Yes|RC) = .5 P(No|RC) = .5 P(Yes) = .65 P(No) = .35 P(RC) = .3

The total law of probability states that P(A) = the summation of P(A|B) * P(B) The P(Yes) comes from only two conditional probabilities: P(Yes|RC) and P(Yes|TC) So P(Yes)=P(Yes RC)P(RC)+P(Yes TC)P(TC) Using some algebra I can arrange this to (P(Yes) - P(Yes|RC)P(RC)) / P(TC) = P(Yes TC) Thus, P(Yes|TC) is **71.43%**

Part B

Sensitivity = P(P|D) = .993 Specificity = P(N|ND) = .9999 Disease = P(D) = .000025 The question we are solving is: What is the P(D|P)? Bayes Theorem is P(D|P) = (P(P|D)*P(D)) / P(P) We have P(P|D) and P(D), so we need to find P(P). This will require using the rule of total probability So P(P) = P(P|D) P(D) + P(P|ND) * P(ND) So we need P(ND) and P(P|ND) No disease = P(ND) = 1 - P(D) = .999975 P(P|ND) = 1 - P(N|ND) = .0001 (This is the False Positive case) P(P) = .00012 After calculating all of my needed info and applying Bayes Theorem, I get that the Probability of a person having the disease given a positive test is *19.88%*

```
p_yes_rc = .5
p_no_rc = .5
p_yes = .65
p_no = .35
p_rc = .3
p_tc = .7
p_yes_tc = (p_yes-p_yes_rc*p_rc) / p_tc
print(p_yes_tc)
```

```
## [1] 0.7142857
```

```
# Part B
sensitivity = .993
specificity = .9999
disease = .000025
no_disease = 1- disease
no_disease
```

```
## [1] 0.999975
```

```r
false_postive = 1 - specificity
false_postive
```

```
## [1] 1e-04
```

```r
positive = sensitivity * disease + false_postive * no_disease
positive
```

```
## [1] 0.0001248225
```

```r
disease_given_positive = (sensitivity * disease)/ positive
print(disease_given_positive)
```

```
## [1] 0.1988824
```

Question 2

Part A

This table shows the top ten most popular songs since 1958 based on how long they were on the billboard 100. Most of these songs were produced in the last 21 years. I find it interesting that there are no repeats of performers on this top ten list.

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.4     v readr     2.1.5
## v forcats   1.0.0     v stringr   1.5.1
## v ggplot2   3.5.1     v tibble    3.2.1
## v lubridate 1.9.3     v tidyr     1.3.1
## v purrr     1.0.2
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to become errors
## New names:
## Rows: 327895 Columns: 13
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr (5): url, week_id, song, performer, song_id
## dbl (8): ...1, week_position, instance, previous_week_position, peak_positio...
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
## `summarise()` has grouped output by 'performer'. You can override using the `.groups` argument.
```

```
## # A tibble: 29,389 x 3
## # Groups:   performer [10,061]
##    performer                                song                          count
##    <chr>                                    <chr>                         <int>
##  1 Imagine Dragons                          Radioactive                      87
```

```
##  2 AWOLNATION                                  Sail                           79
##  3 Jason Mraz                                  I'm Yours                      76
##  4 The Weeknd                                  Blinding Lights                76
##  5 LeAnn Rimes                                 How Do I Live                  69
##  6 LMFAO Featuring Lauren Bennett & GoonRock   Party Rock Anthem              68
##  7 OneRepublic                                 Counting Stars                 68
##  8 Adele                                       Rolling In The Deep            65
##  9 Jewel                                       Foolish Games/You Were Meant~  65
## 10 Carrie Underwood                            Before He Cheats               64
## # i 29,379 more rows
```

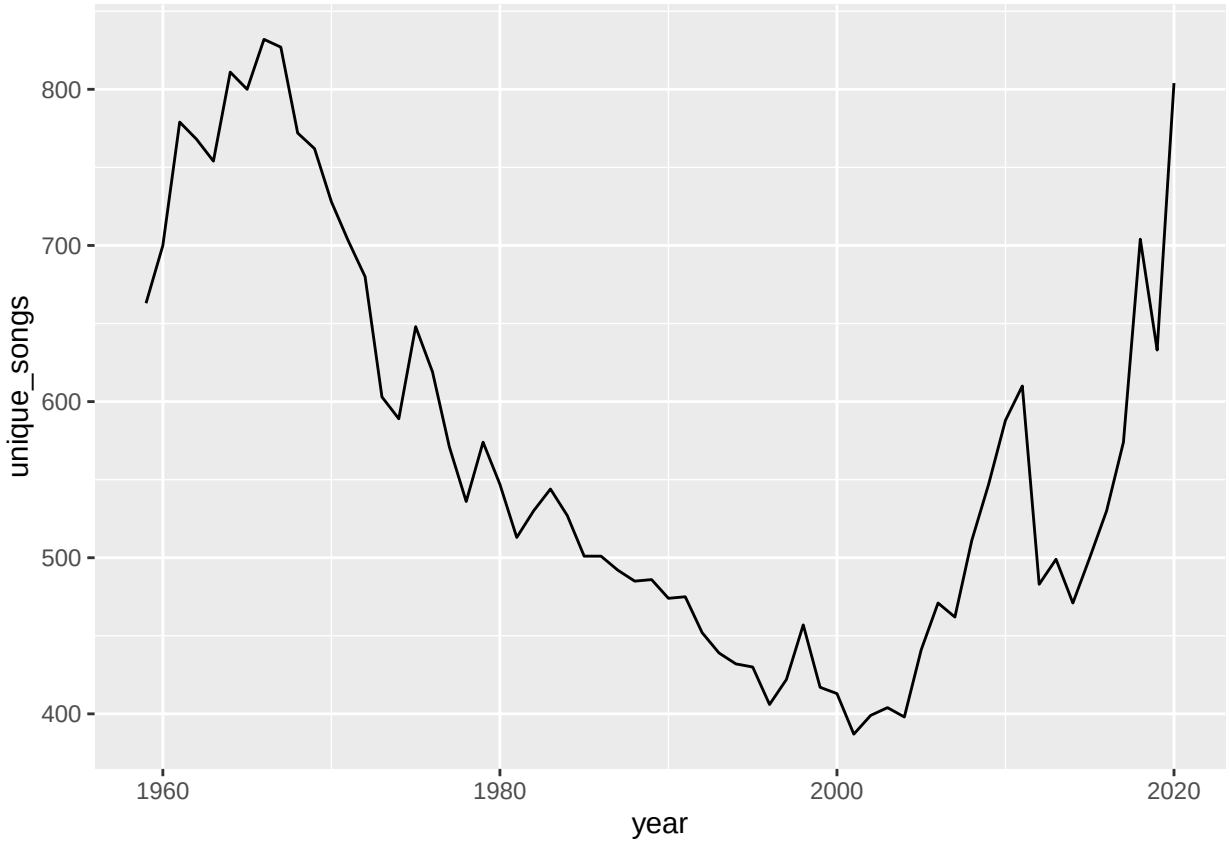
Part B

This plot shows the total number of unique songs that charted the Billboard 100 per year. I guess my parents were correct when they said music was better in the 80's! The number of unique songs that chart peaks at around 1967 and then rapidly declines until around 2002, where more unique songs started to chart. This could potentially be due to the rise if iTunes. There was a decline around 2011 followed by rapid unique song growth.

```
#b
billboard_cutoff = billboard %>% filter(year != 1958 & year != 2021)
table_with_counts = billboard_cutoff %>% group_by(performer,song,year) %>%
  summarize(total_count = n())
```

```
## `summarise()` has grouped output by 'performer', 'song'. You can override using
## the `.groups` argument.
```

```
unique_song_count = table_with_counts %>% group_by(year) %>%
  summarize(unique_songs = n())
ggplot(unique_song_count) + geom_line(aes(x=year,y=unique_songs))
```

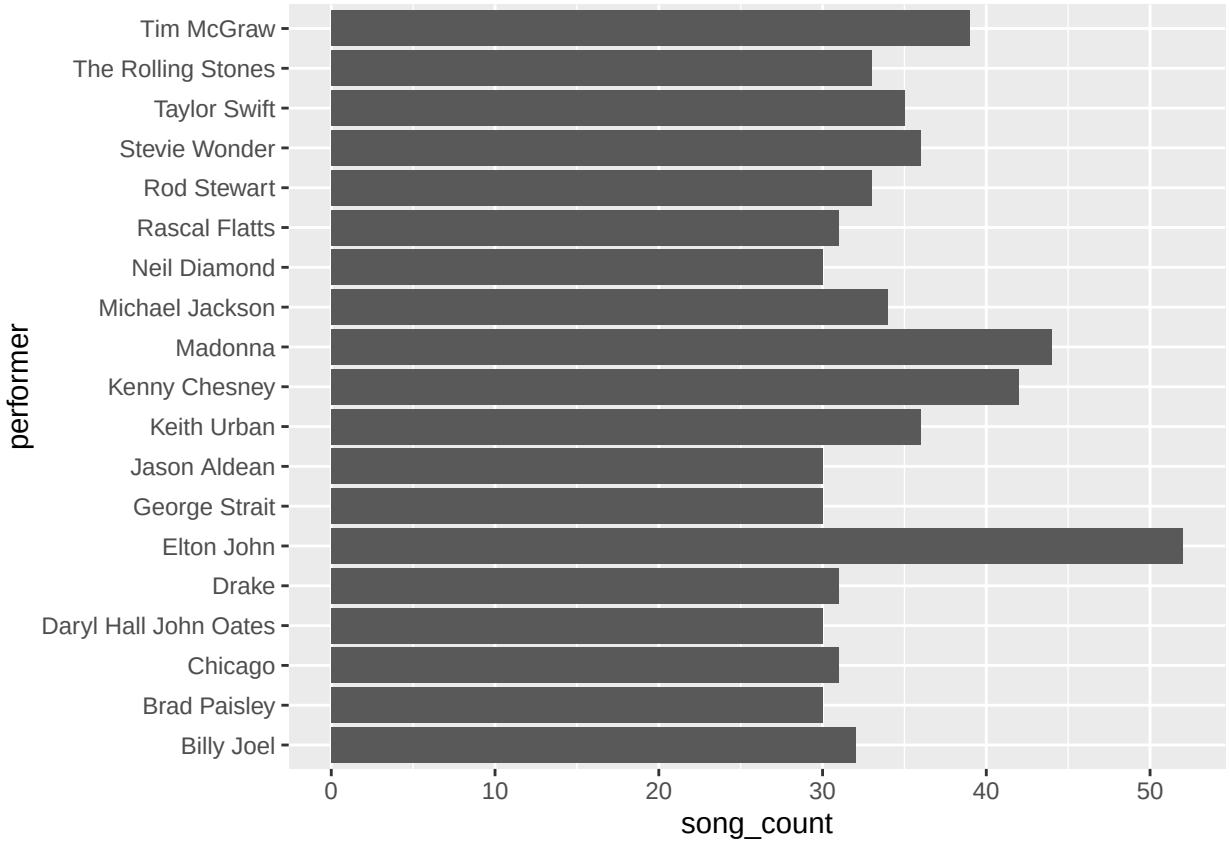


Part C This plot shows artists who have had 30 songs chart for at least ten weeks. Elton John has the highest number of songs with 52 songs. I find it interesting that there are a good amount of country artists filled. I would have thought this list would have been mainly filled with pop and rock artists

```
#C
billboard_ten_week = billboard %>% group_by(performer,song) %>%
  summarize(count = n()) %>%
  filter(count >=10)
```

```
## `summarise()` has grouped output by 'performer'. You can override using the
## `.groups` argument.
```

```
billboard_19_artists = billboard_ten_week %>% group_by(performer) %>%
  summarize(song_count =n()) %>% filter(song_count >=30)
ggplot(billboard_19_artists) + geom_col(aes(x=performer, y=song_count)) +
  coord_flip()
```



Problem 3 In order to agree with the stats guru's conclusion that building a green building makes sense, we first have to do our own analysis of the data and see what findings we can come up with. We first took a look at the median rent of green buildings vs non green buildings and confirmed that on average green buildings earn about $2.6 more per sq ft than non green buildings. Our general strategy is to see what factors could affect rent and see if these factors are over or under represented in the green buildings. We then looked to see if green buildings tended to be more of Class A. We found that around 80% of green rated buildings are class A, versus only 36% of non green buildings. This may be a potential confounder, as class A buildings will command more rent. We plotted Rent vs age and found that an older building commands less rent. However, the non green buildings are almost 50 years old on average vs 24 years for green buildings! This could certainly be a confounder. Rent is LOWER for renovated buildings. This is not what we expected. Renovated buildings command less rent by $3.79! Most green buildings (around 79%), however, are NOT renovated! This means that they are drawing higher prices due to not being renovated.

So far we have found 3 plausible explanations for why green buildings demand higher rent. The class A variable is likely one of the biggest confounders, so we want to see the median rent within non class A buildings faceted by green rating. We found that after adjusting for class A buildings the premium is only $2.12 now.

We think this trend will continue for most of these confounders. We think the best way to adjust for these confounders is to use a technique called matching. Matching relies on a simple principle: compare like with like. In this example, that means if we have a 25-year-old, Class A building that is renovated with a green rating, we try to find another 25-year old, Class A renovated building without a green rating to compare it to. Matching constructs a balanced data set from an unbalanced one. This matched data can them be compared by their rents to see if green buildings truly cause a higher premium.

We think the stats guru did not take into account any confounders in his model. While green builfsing may deman higher rent, we think there any more variables at play here.

```r
library(mosaic)
```

```
## Registered S3 method overwritten by 'mosaic':
##   method                           from
##   fortify.SpatialPolygonsDataFrame ggplot2

##
## The 'mosaic' package masks several functions from core packages in order to add
## additional features.  The original behavior of these functions should not be affected by this.

##
## Attaching package: 'mosaic'

## The following object is masked from 'package:Matrix':
##
##     mean

## The following objects are masked from 'package:dplyr':
##
##     count, do, tally

## The following object is masked from 'package:purrr':
##
##     cross

## The following object is masked from 'package:ggplot2':
##
##     stat

## The following objects are masked from 'package:stats':
##
##     binom.test, cor, cor.test, cov, fivenum, IQR, median, prop.test,
##     quantile, sd, t.test, var

## The following objects are masked from 'package:base':
##
##     max, mean, min, prod, range, sample, sum
```

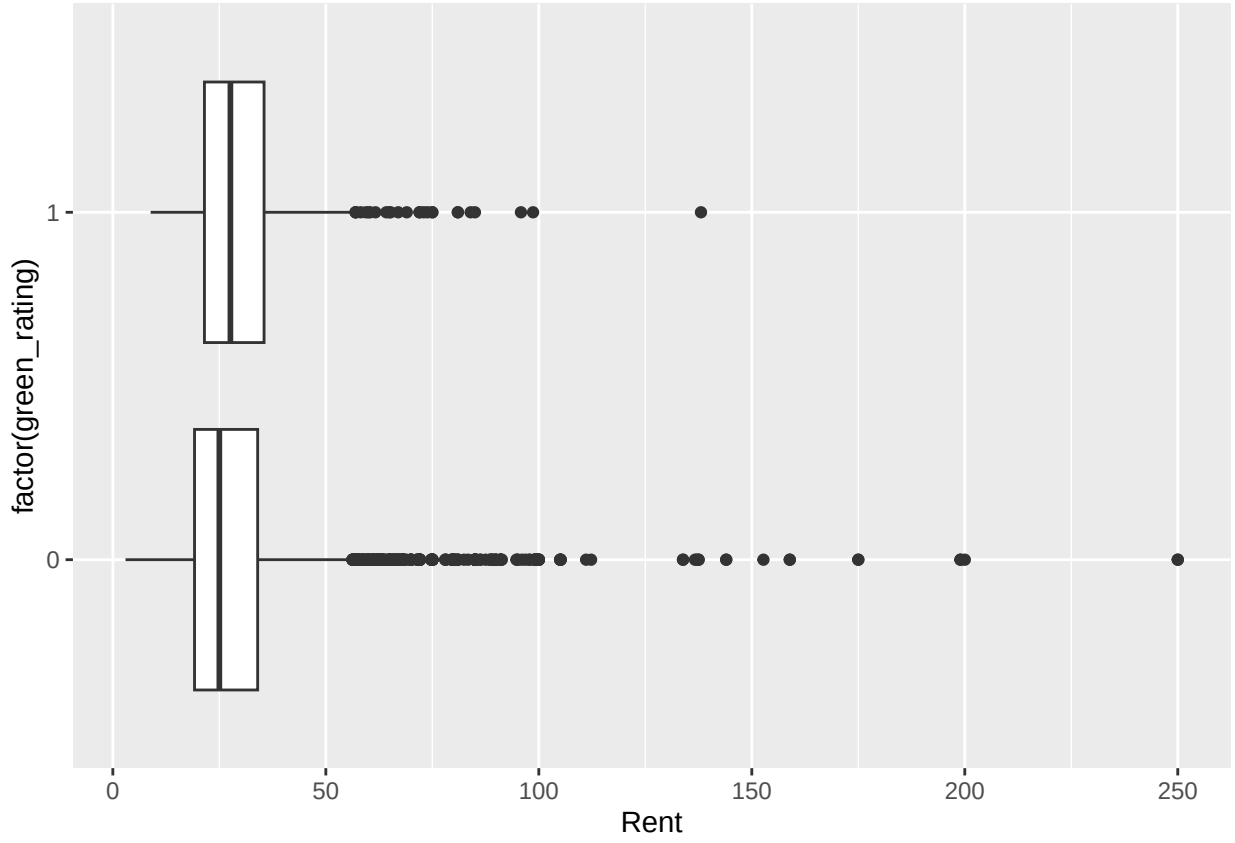
```r
greenbuildings = read_csv("greenbuildings.csv")
```

```
## Rows: 7894 Columns: 23

## -- Column specification --------------------------------------------------------
## Delimiter: ","
## dbl (23): CS_PropertyID, cluster, size, empl_gr, Rent, leasing_rate, stories...
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
# also remove the occupancy rate of less than 10%
greenbuildings = greenbuildings %>% filter('leasing_rate' >= 10)

median(Rent ~ green_rating, data=greenbuildings)
```

```
##    0    1
## 25.0 27.6
```

```
ggplot(greenbuildings) +
  geom_boxplot(aes(x=factor(green_rating), y=Rent)) +
  coord_flip()
```




```
# Look at which buildings are "more desirable" (Class A)
xtabs(~ class_a + green_rating, data=greenbuildings) %>%
  prop.table(margin=2)
```

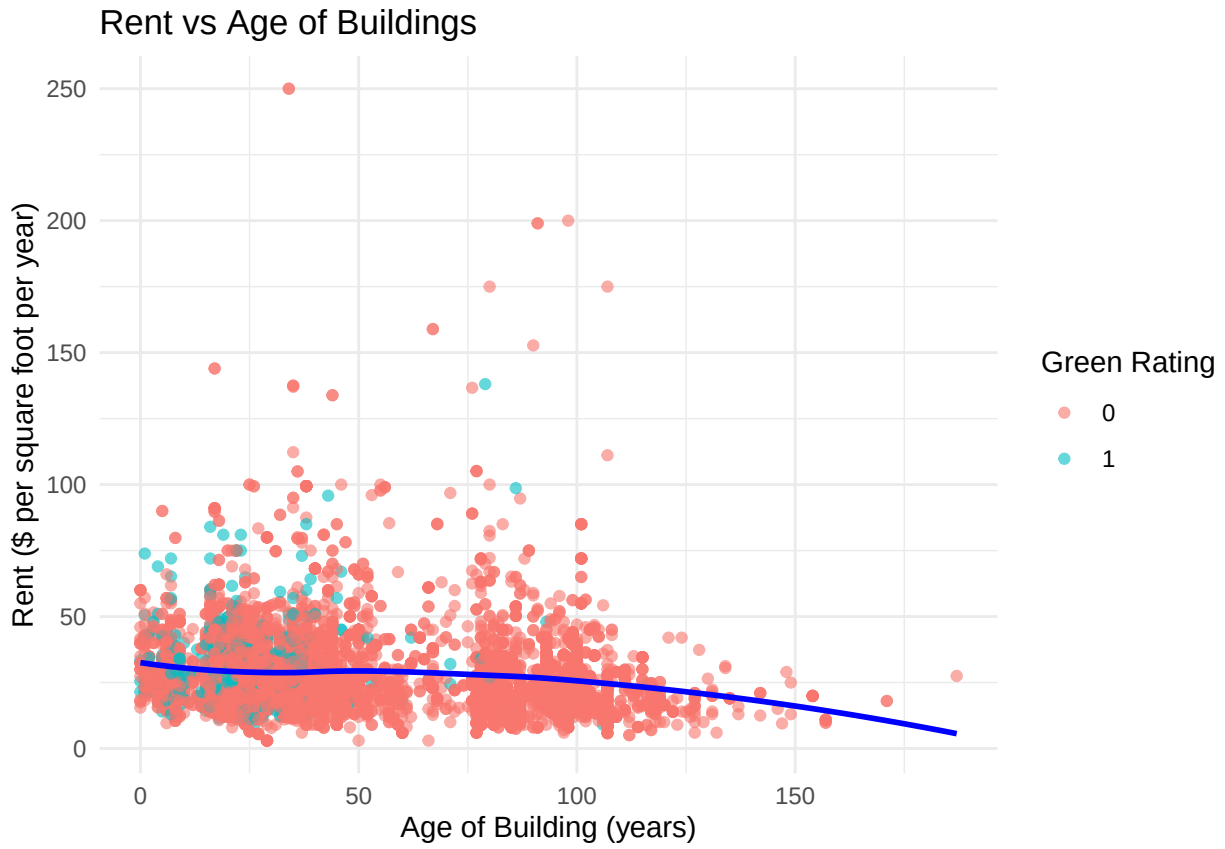
```
##         green_rating
## class_a         0         1
##       0 0.6378138 0.2029197
##       1 0.3621862 0.7970803
```

```
# Look at how age affects rent
mean(age ~ green_rating, data=greenbuildings)
```

```
##        0        1
## 49.46733 23.84526
```

```
ggplot(greenbuildings, aes(x = age, y = Rent)) +
  geom_point(aes(color = as.factor(green_rating)), alpha = 0.6) +
  geom_smooth(method = "loess", se = FALSE, color = "blue") +
  labs(title = "Rent vs Age of Buildings",
       x = "Age of Building (years)",
       y = "Rent ($ per square foot per year)",
       color = "Green Rating") +
  theme_minimal()
```

```
## `geom_smooth()` using formula = 'y ~ x'
```




```
# look to see if renovated affects rent
mean(Rent ~ renovated, data = greenbuildings)
```

```
##        0        1
## 29.85776 26.06570
```

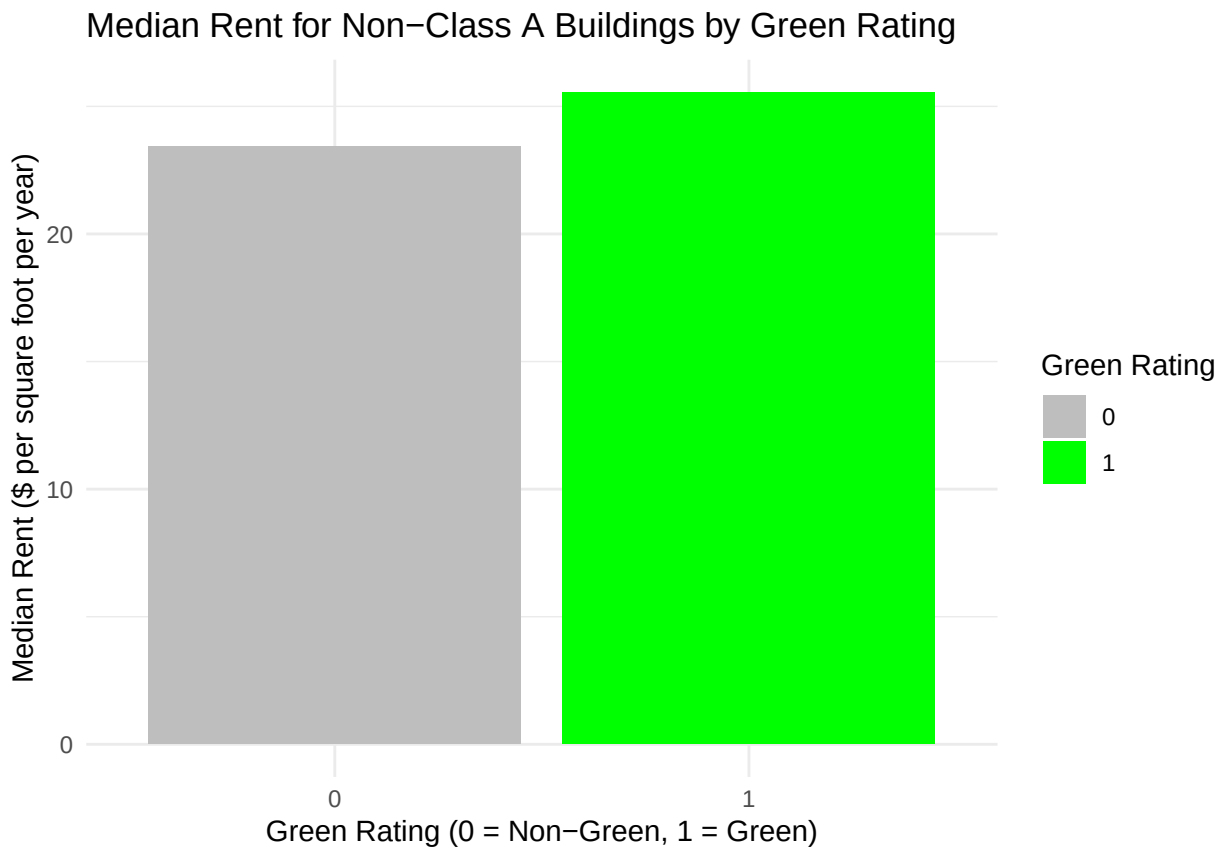
```
xtabs(~ renovated + green_rating, data=greenbuildings) %>%
  prop.table(margin=2)
```

```
##          green_rating
## renovated         0         1
##         0 0.6046608 0.7868613
##         1 0.3953392 0.2131387
```

```r
# Look at non class A buildings
median_rent_non_class_a <- greenbuildings %>%
  filter(class_a == 0) %>%  # Exclude Class A buildings
  group_by(green_rating) %>%
  summarise(median_rent = median(Rent))

print(median_rent_non_class_a)
```

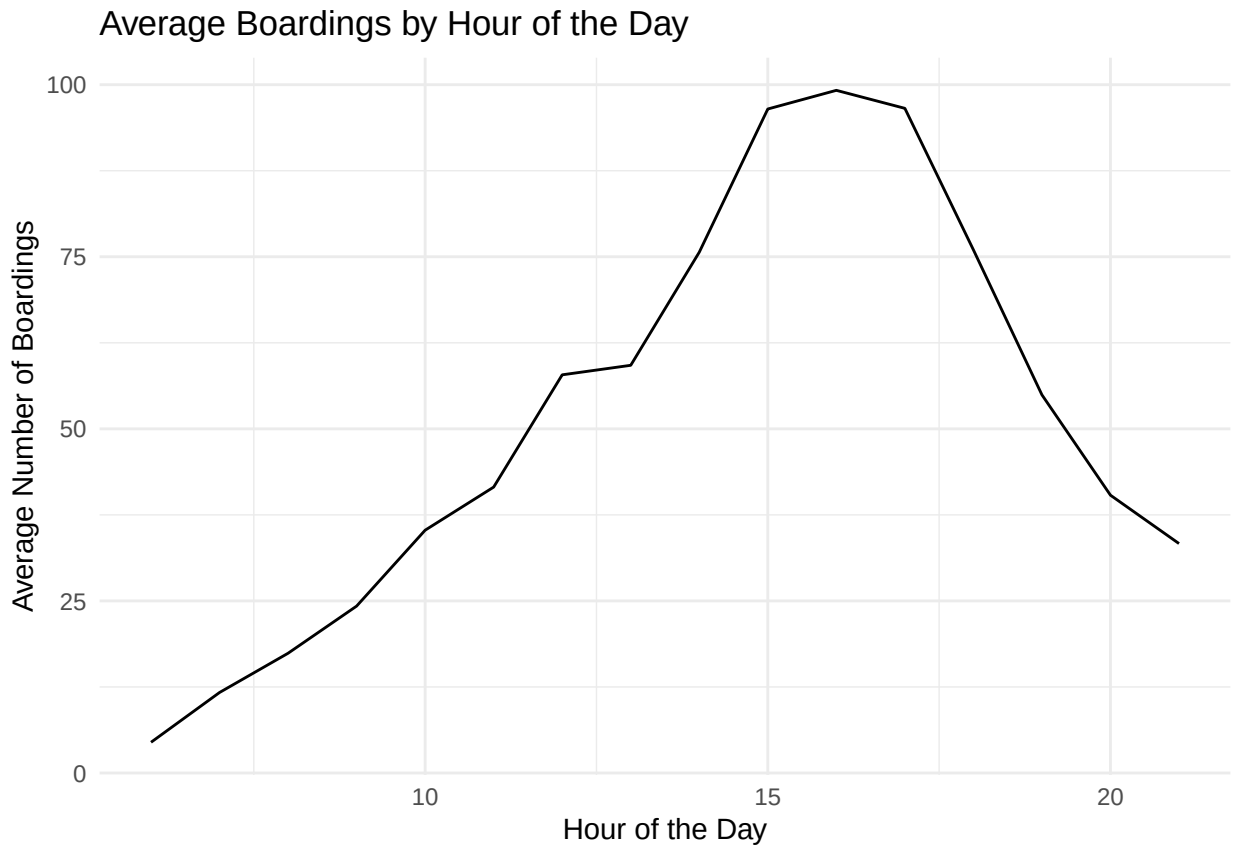
```
## # A tibble: 2 x 2
##   green_rating median_rent
##          <dbl>       <dbl>
## 1            0        23.4
## 2            1        25.6
```

```r
ggplot(median_rent_non_class_a, aes(x = factor(green_rating), y = median_rent, fill = factor(green_rati
  geom_bar(stat = "identity") +
  labs(title = "Median Rent for Non-Class A Buildings by Green Rating",
       x = "Green Rating (0 = Non-Green, 1 = Green)",
       y = "Median Rent ($ per square foot per year)",
       fill = "Green Rating") +
  scale_fill_manual(values = c("0" = "grey", "1" = "green")) +
  theme_minimal()
```

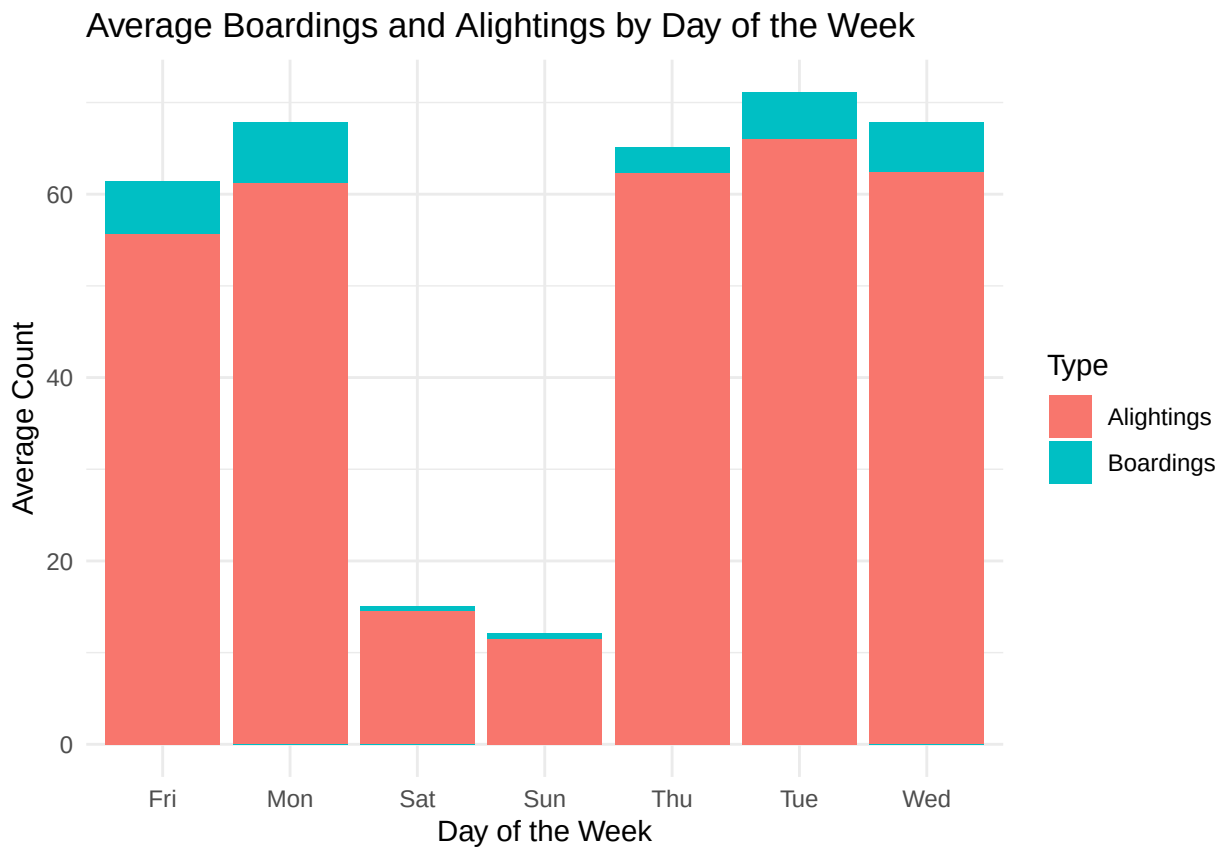


Problem 4

```
file_path <- "C:/Users/grays/OneDrive/Documents/STA 380/capmetro_UT.csv"
capmetro_UT <- read.csv(file_path)

# Convert timestamp to datetime
capmetro_UT$timestamp <- ymd_hms(capmetro_UT$timestamp)

# Plot average boardings by hour of the day
hour_summary <- capmetro_UT %>%
  group_by(hour_of_day) %>%
  summarize(mean_boardings = mean(boarding))

ggplot(hour_summary) +
  geom_line(aes(x = hour_of_day, y = mean_boardings)) +
  labs(title = "Average Boardings by Hour of the Day",
       x = "Hour of the Day",
       y = "Average Number of Boardings") +
  theme_minimal()
```

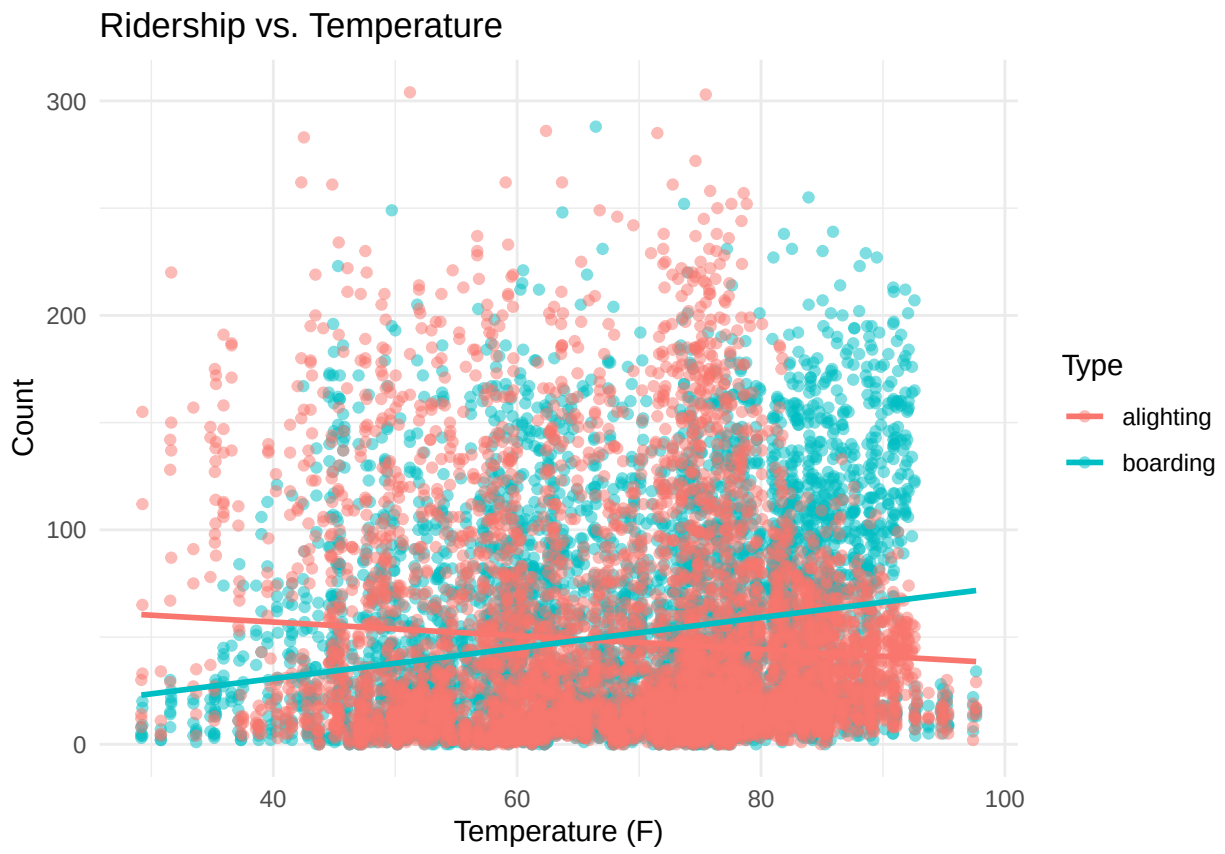



```r
# Plot average boardings and alightings by day of the week
day_summary <- capmetro_UT %>%
  group_by(day_of_week) %>%
  summarise(mean_boardings = mean(boarding), mean_alightings = mean(alighting))

ggplot(day_summary) +
  geom_bar(aes(x = day_of_week, y = mean_boardings, fill = "Boardings"), stat = "identity", position =
  geom_bar(aes(x = day_of_week, y = mean_alightings, fill = "Alightings"), stat = "identity", position =
  labs(title = "Average Boardings and Alightings by Day of the Week",
       x = "Day of the Week",
       y = "Average Count",
       fill = "Type") +
  theme_minimal()
```




```
# Plot ridership vs temperature
temp_ridership <- capmetro_UT %>%
  gather(key = "type", value = "count", boarding, alighting)

ggplot(temp_ridership, aes(x = temperature, y = count, color = type)) +
  geom_point(alpha = 0.5) +
  geom_smooth(method = "lm", se = FALSE) +
  labs(title = "Ridership vs. Temperature",
       x = "Temperature (F)",
       y = "Count",
       color = "Type") +
  theme_minimal()
```

```
## `geom_smooth()` using formula = 'y ~ x'
```

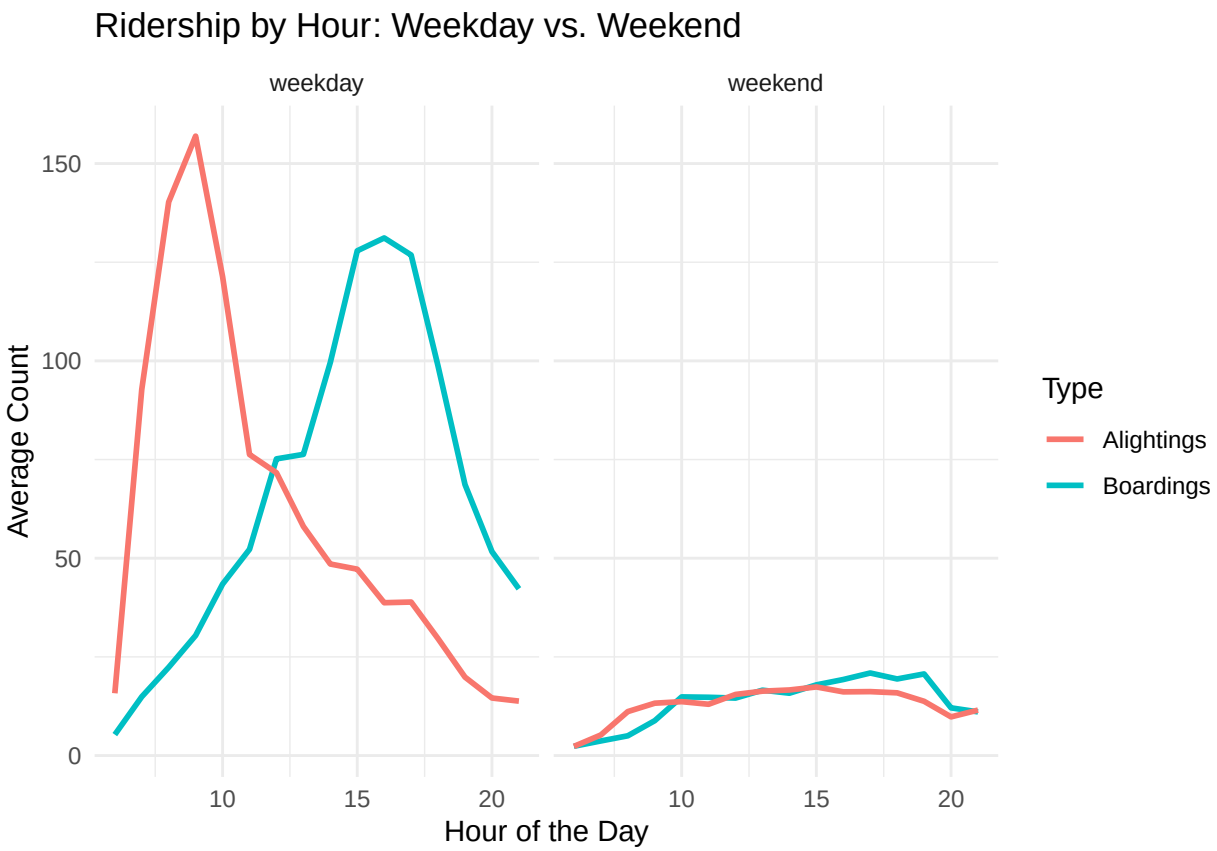



```
# Plot ridership by weekend status
weekend_summary <- capmetro_UT %>%
  group_by(weekend, hour_of_day) %>%
  summarise(mean_boardings = mean(boarding), mean_alightings = mean(alighting))
```

```
## `summarise()` has grouped output by 'weekend'. You can override using the
## `.groups` argument.
```

```
ggplot(weekend_summary) +
  geom_line(aes(x = hour_of_day, y = mean_boardings, color = "Boardings"), size = 1) +
  geom_line(aes(x = hour_of_day, y = mean_alightings, color = "Alightings"), size = 1) +
  facet_wrap(~weekend) +
  labs(title = "Ridership by Hour: Weekday vs. Weekend",
       x = "Hour of the Day",
       y = "Average Count",
       color = "Type") +
  theme_minimal()
```

```
## Warning: Using `size` aesthetic for lines was deprecated in ggplot2 3.4.0.
## i Please use `linewidth` instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```



Note that the `echo = FALSE` parameter was added to the code chunk to prevent printing of the R code that generated the plot.