Problem Set

2024-08-14

Group Members: Quinlan O'Connell, Neal Makwana, Maru Lasala, Ashley Hattendorf

Wrangling Billboard Top 100

Part A: Make a table of the top 10 most popular songs since 1958, as measured by the *total number of weeks that a song spent on the Billboard Top 100.* Note that these data end in week 22 of 2021, so the most popular songs of 2021 will not have up-to-the-minute data; please send our apologies to The Weeknd.

Your table should have **10 rows** and **3 columns**: `performer`, `song`, and `count`, where `count` represents the number of weeks that song appeared in the Billboard Top 100. Make sure the entries are sorted in descending order of the `count` variable, so that the more popular songs appear at the top of the table. Give your table a short caption describing what is shown in the table.

```r
top_10 <- billboard %>%
  select(performer, song) %>%
  group_by(performer, song) %>%
  summarize(count = n(), .groups = 'drop') %>% #use summarize to count entries per song, .groups = 'dro
  arrange(desc(count)) %>%
  head(10)
top_10
```

```
## # A tibble: 10 x 3
##    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
```

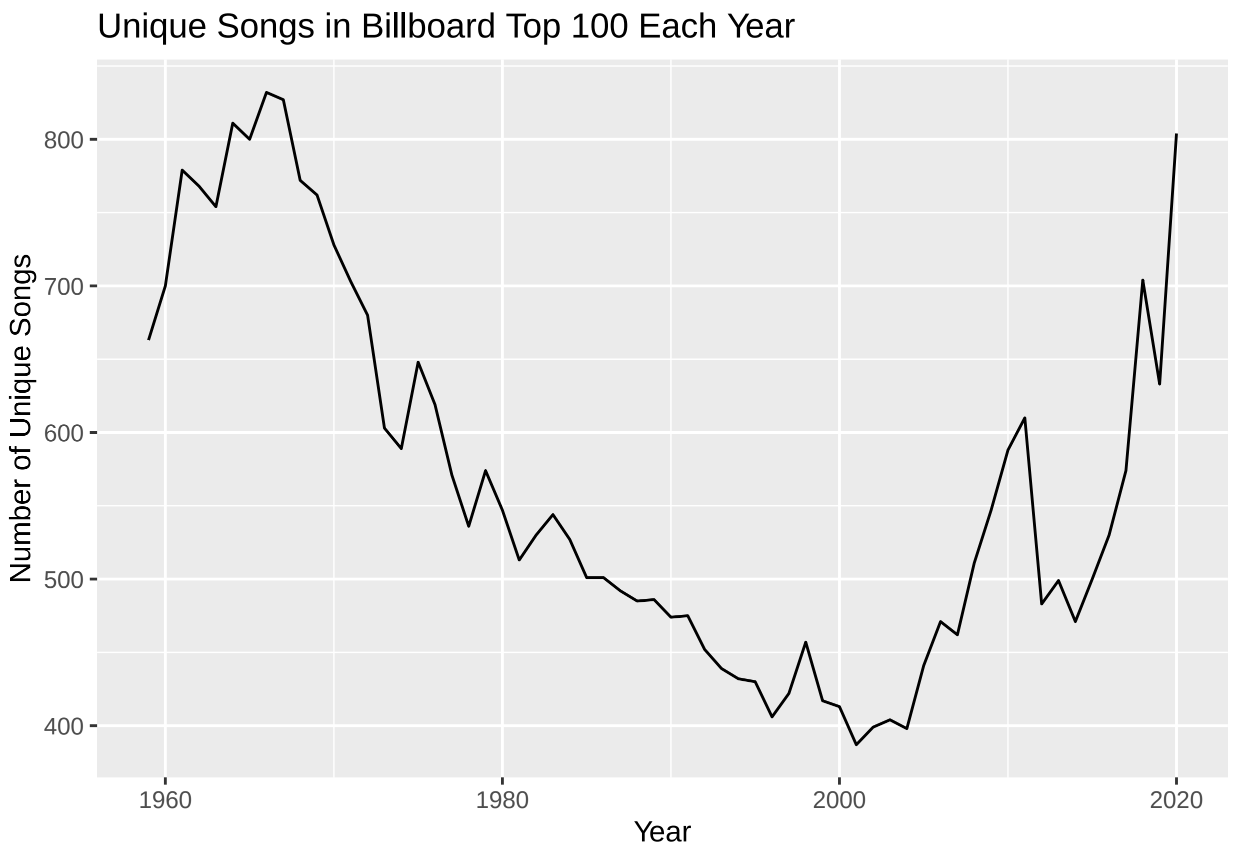
The table above shows the top 10 most popular billboard songs. "Radioactive" has the highest count, with a count of 87. Other popular songs like Sail, I'm Yours, and Blinding Lights follow, with a count in the 76-79 range.

Part B: Is the "musical diversity" of the Billboard Top 100 changing over time? Let's find out. We'll measure the musical diversity of given year as *the number of unique songs that appeared in the Billboard Top 100 that year.* Make a line graph that plots this measure of musical diversity over the years. The x axis should show the year, while the y axis should show the number of unique songs appearing at any position on the Billboard Top 100 chart in any week that year. For this part, please filter the data set so that it excludes the years 1958 and 2021, since we do not have complete data on either of those years. Give the figure an informative caption in which you explain what is shown in the figure and comment on any interesting trends you see.

```r
unique_billboard <- billboard %>%
  filter(year != 1958 & year != 2021) %>%
  group_by(year, performer, song) %>%
  summarize(count = n()) %>%
  group_by(year) %>%
  summarize(new_count = n())
unique_billboard
```

```
## # A tibble: 62 x 2
##     year new_count
##    <int>     <int>
##  1  1959       663
##  2  1960       700
##  3  1961       779
##  4  1962       768
##  5  1963       754
##  6  1964       811
##  7  1965       800
##  8  1966       832
##  9  1967       827
## 10  1968       772
## # i 52 more rows
```

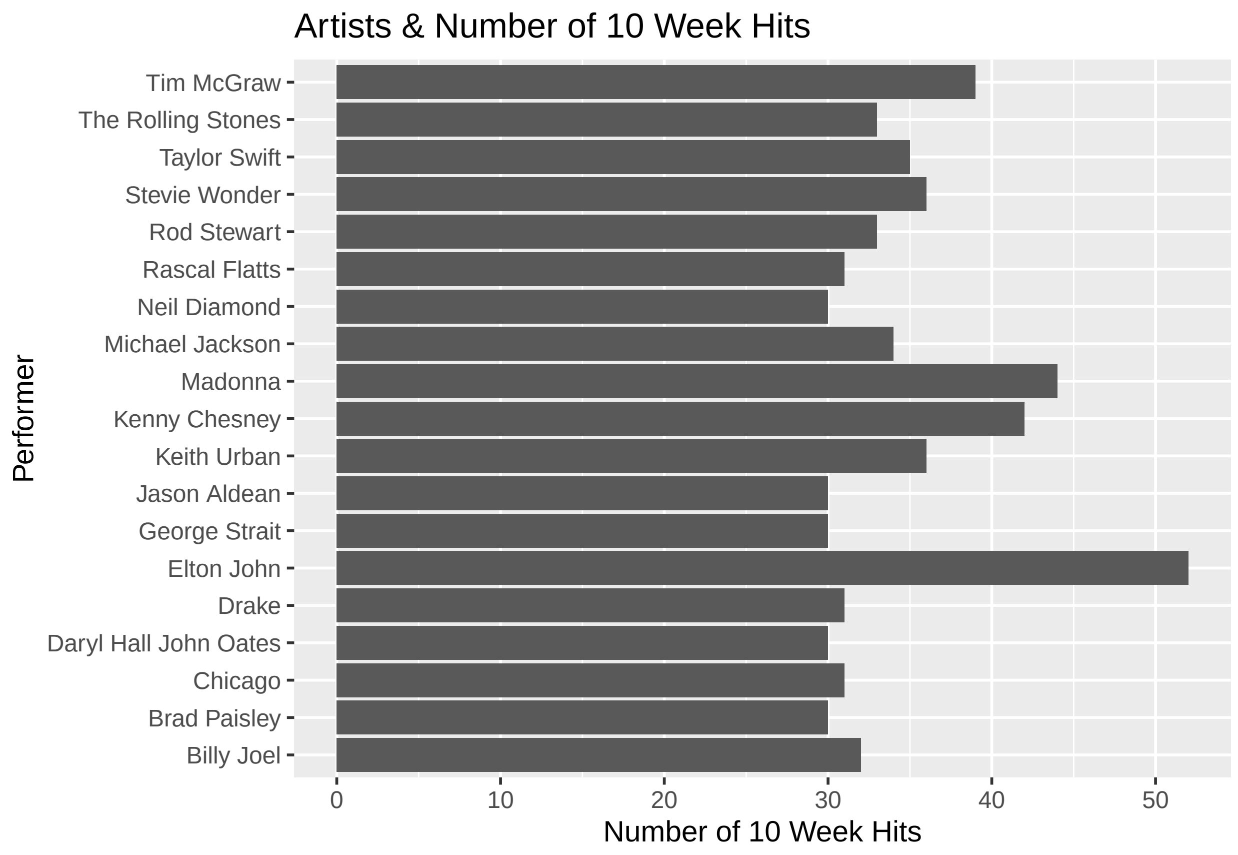
```r
ggplot(unique_billboard) +
  geom_line(aes(x= year, y= new_count)) +
  labs(x= "Year",
       y="Number of Unique Songs",
       title= "Unique Songs in Billboard Top 100 Each Year")
```



The line graph above shows the number of unique songs that appeared in the Billboard Top 100, based on the year. We can see that the year with the most unique songs to make the list was in about 1967, where the data peaks at about 850 unique songs making the list that year. The data then trends downward until it reached a low in around 2001, at about 380 unique songs on the list. From here, the data trends upward until 2011, where it hits a short dip and then continues back upward until 2020.

Part C: Let's define a "ten-week hit" as a single song that appeared on the Billboard Top 100 for at least ten weeks. There are 19 artists in U.S. musical history since 1958 who have had *at least 30 songs* that were "ten-week hits." Make a bar plot for these 19 artists, showing how many ten-week hits each one had in their musical career. Give the plot an informative caption in which you explain what is shown.

```r
ten_weeks_billboard <- billboard %>%
  group_by(performer, song) %>%
  summarize(n = n()) %>%
  filter(n >= 10) %>%
  group_by(performer) %>%
  summarize(n = n()) %>%
  arrange(desc(n)) %>%
  filter(n >= 30)
ggplot(ten_weeks_billboard) +
  geom_col(aes(x = performer, y = n)) +
  labs(x="Performer",
       y="Number of 10 Week Hits",
       title= "Artists & Number of 10 Week Hits") +
  coord_flip()
```



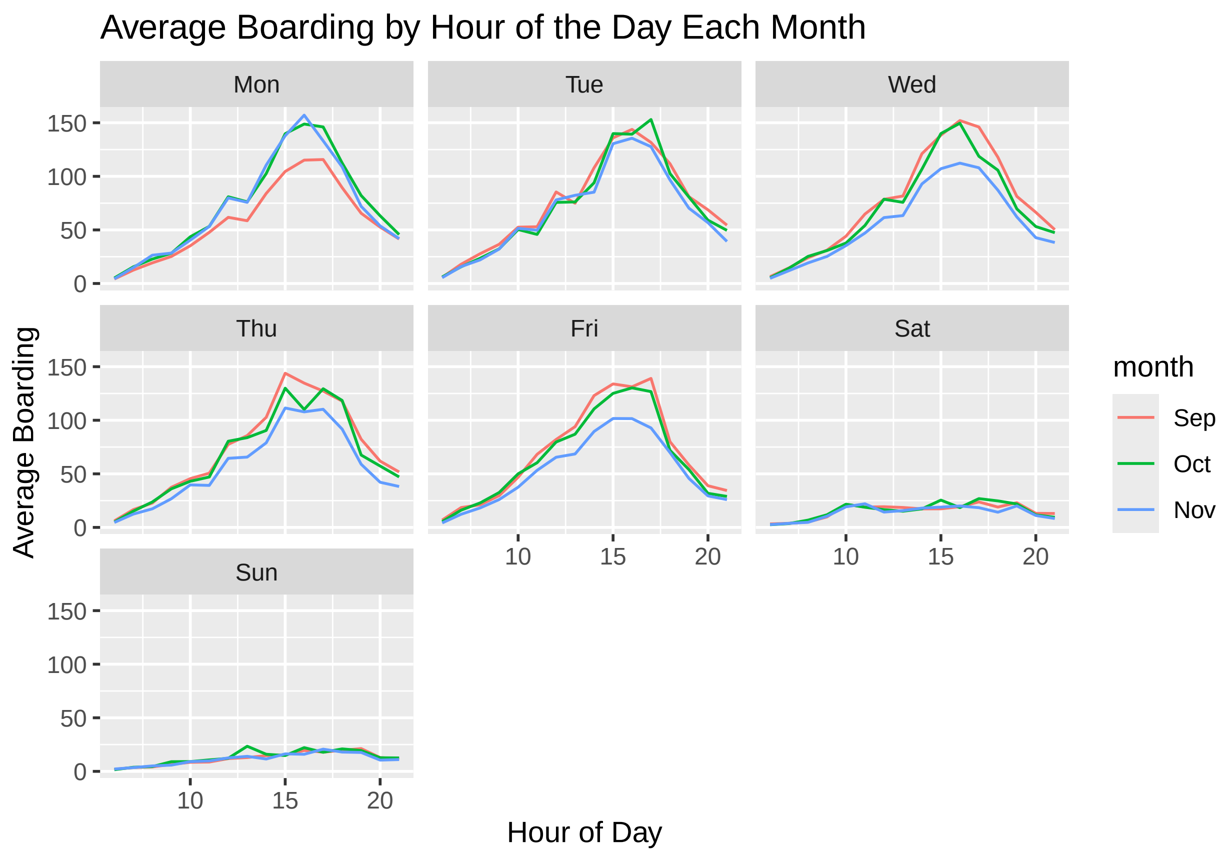
The bar plot above shows artists who have had at least 30 songs that were "ten-week hits" (songs that appeared on the Billboard Top 100 for at least 10 weeks). 19 artists in history have had at least 30 songs that were "ten-week hits", and we can see above the number of these (x-axis) based on the artist (y-axis). As seen from the data, the artist with the most "ten-week hits" is Elton John, as he has 52 songs that spent at least 10 weeks on the Billboard Top 100. He is followed by Madonna and Kenny Chesney, who have 44 and 42 ten week hits, respectively.

CapMetro Storytelling

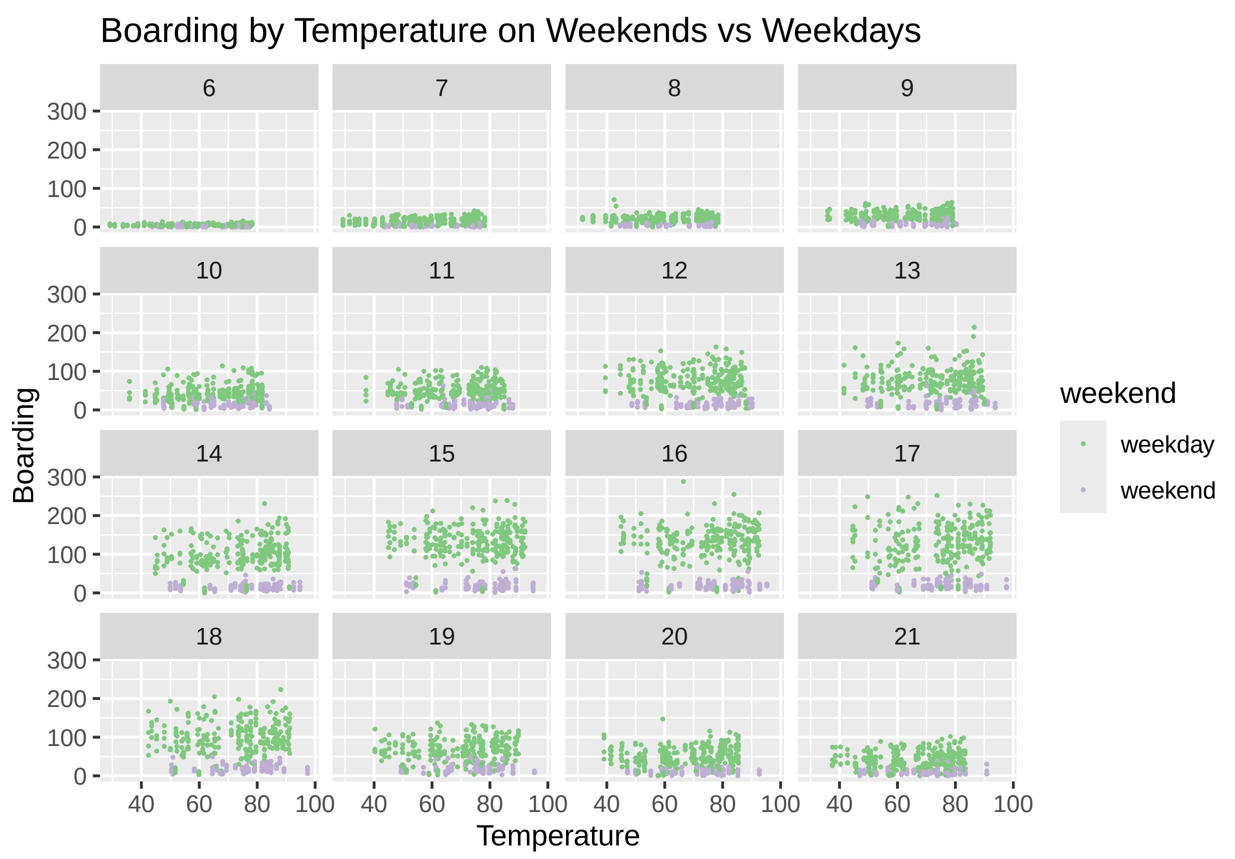
Your task is to create a figure, or set of related figures, that tell an interesting story about Capital Metro ridership patterns around the UT-Austin campus during the semester in question. Provide a clear annotation/caption for each figure, but the figure(s) should be more or less stand-alone, in that you shouldn't need many, many paragraphs to convey its meaning. Rather, the figure together with a concise caption should speak for itself as far as possible.

```r
# named it capmetro_UT_data
capmetro_UT = read.csv("capmetro_UT_data.csv")
boarding_data = mutate(capmetro_UT,
                       day_of_week = factor(day_of_week,
                                            levels=c("Mon", "Tue", "Wed","Thu", "Fri", "Sat", "Sun")),
                       month = factor(month,
                                      levels=c("Sep", "Oct","Nov")))
boarding_data = boarding_data %>%
  group_by(hour_of_day, day_of_week, month) %>%
  summarize(avg_board = mean(boarding))
```

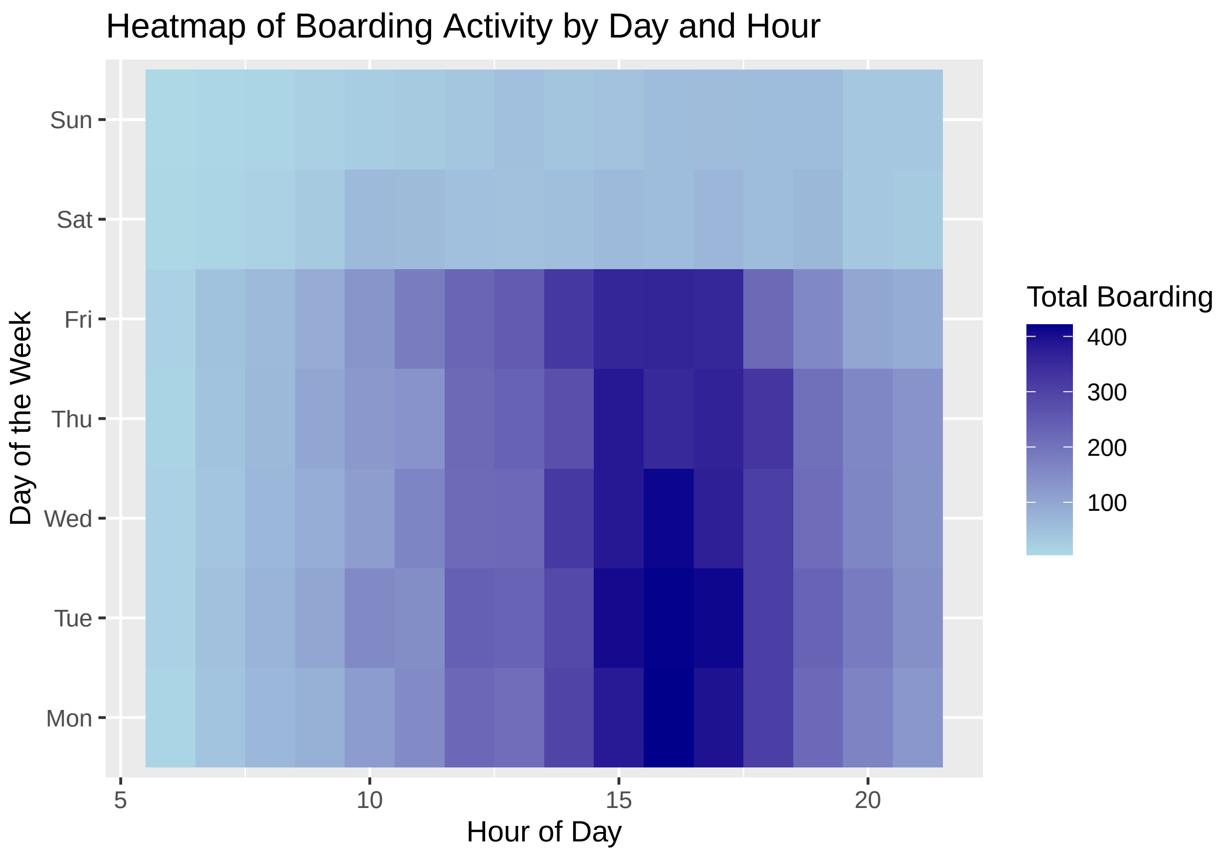
```r
ggplot(boarding_data) +
  geom_line(aes(x=hour_of_day, y=avg_board, color = month)) +
  facet_wrap(~day_of_week) +
  labs(x="Hour of Day",
       y="Average Boarding",
       title="Average Boarding by Hour of the Day Each Month")
```



```r
temperature_capmetro = capmetro_UT %>%
  group_by(temperature, boarding, hour_of_day, weekend)
ggplot(temperature_capmetro) +
  geom_point(aes(x=temperature, y=boarding, color=weekend), size=0.2) +
  facet_wrap(~hour_of_day) +
  scale_color_brewer(type="qual") +
  labs(x="Temperature",
       y="Boarding",
       title= "Boarding by Temperature on Weekends vs Weekdays")
```



```r
boarding_heatmap <- boarding_data %>%
  group_by(day_of_week, hour_of_day) %>%
  summarize(total_boarding = sum(avg_board))
ggplot(boarding_heatmap, aes(x=hour_of_day, y=day_of_week, fill=total_boarding)) +
  geom_tile() +
  scale_fill_gradient(low="lightblue", high="darkblue") +
  labs(x="Hour of Day",
       y="Day of the Week",
       fill="Total Boarding",
       title="Heatmap of Boarding Activity by Day and Hour")
```



In the top figure, the average boardings on the UT Metro, organized by the day of the week, hour of the day, and month of the year. Three different lines on each graph represent different months of the year, and the X-Axis shows hour of the day while the Y-Axis shows the number of average boardings. According to the data above, the hour of peak boarding is very similar across all three months, but only on the days Monday-Friday. The number of average boardings tends to peak around hour 15 (3PM) during the week, which is likely explained by this time being a heavy travel time for students going to and from class.

In the middle figure, the scatter plot shows the number of UT Metro boardings throughout the hours of the day, based on the temperature and type of day it is (weekday or weekend). It seems like the peak demand for the UT metro is in hours 15-17 (3-5PM) during the week, and 17-18 (5-6PM) on the weekend. Boarding is much less abundant in the early hours (hour 6-9), which is unsurprising considering many college students would be asleep then. According to the data, it seems like when we hold hour of day and weekend status constant, temperature does not seem to have a noticeable effect on the number of UT students riding the bus. When analyzing the data, there is always a heavy distribution of boardings across many different temperatures.

In the bottom figure, the heatmap helps to visualize activity by the hour of the day. We can see a hotter portion on the heatmap in the later afternoon day, which likely indicates the end of the school/workday. We can also see a clear distinction between the weekdays and weekend, which may indicate a decline in ridership when people are not working/in school/commuting to campus.

Rule Mining Association

```r
# Generating association rules
rules <- apriori(transactions, parameter = list(supp = 0.001, conf = 0.5))
```

```
## Apriori
##
## Parameter specification:
##  confidence minval smax arem  aval originalSupport maxtime support minlen
##         0.5    0.1    1 none FALSE            TRUE       5   0.001      1
##  maxlen target  ext
##      10  rules TRUE
##
## Algorithmic control:
##  filter tree heap memopt load sort verbose
##     0.1 TRUE TRUE  FALSE TRUE    2    TRUE
##
## Absolute minimum support count: 9
##
## set item appearances ...[0 item(s)] done [0.00s].
## set transactions ...[169 item(s), 9835 transaction(s)] done [0.01s].
## sorting and recoding items ... [157 item(s)] done [0.00s].
## creating transaction tree ... done [0.00s].
## checking subsets of size 1 2 3 4 5 6 done [0.02s].
## writing ... [5668 rule(s)] done [0.00s].
## creating S4 object  ... done [0.00s].
```

```
# chose base items

# number of items in rules
length(rules)
```

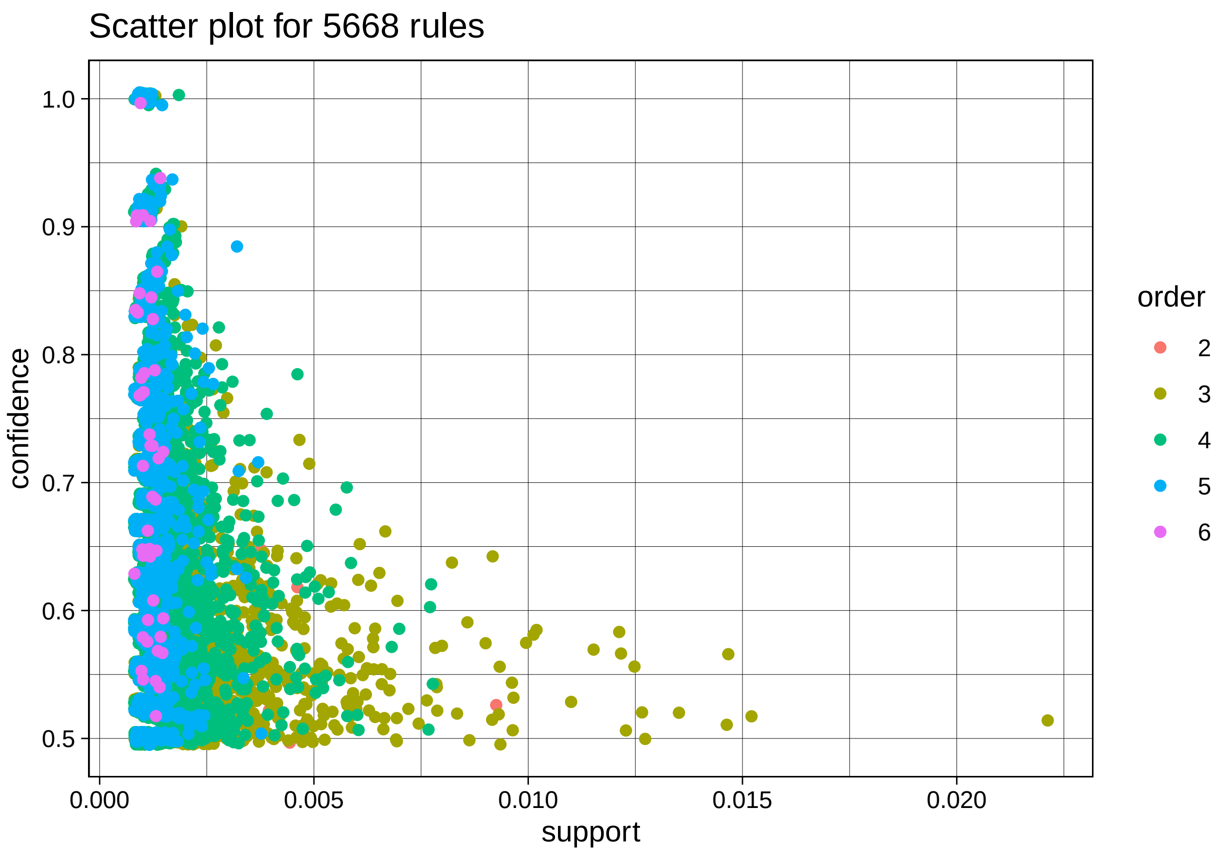
```
## [1] 5668
```

```
# HIGH Confidence Rules!
#inspect(sort(rules, by = "confidence"))
#arules::inspect(sort(rules, by = "confidence"))

# transactions with whole milk / other vegetables seem to have very high confidence values


# Inspect the top rules by lift
# Sort the rules by lift to find the most interesting ones
sorted_rules <- sort(rules, by = "lift")
# show rules
arules::inspect(sorted_rules[1:30])
```

```
##      lhs                            rhs                        support confidence    coverage      lift c
## [1]  {Instant food products,
##       soda}                      => {hamburger meat}     0.001220132  0.6315789 0.001931876 18.995654
## [2]  {popcorn,
##       soda}                      => {salty snack}        0.001220132  0.6315789 0.001931876 16.697793
## [3]  {baking powder,
##       flour}                     => {sugar}              0.001016777  0.5555556 0.001830198 16.408075
## [4]  {ham,
##       processed cheese}          => {white bread}        0.001931876  0.6333333 0.003050330 15.045491
## [5]  {Instant food products,
##       whole milk}                => {hamburger meat}     0.001525165  0.5000000 0.003050330 15.038226
## [6]  {curd,
##       other vegetables,
##       whipped/sour cream,
##       yogurt}                 => {cream cheese}       0.001016777  0.5882353 0.001728521 14.834087
## [7]  {domestic eggs,
##       processed cheese}       => {white bread}        0.001118454  0.5238095 0.002135231 12.443639
## [8]  {other vegetables,
##       tropical fruit,
##       white bread,
##       yogurt}                 => {butter}             0.001016777  0.6666667 0.001525165 12.030581
## [9]  {hamburger meat,
##       whipped/sour cream,
##       yogurt}                 => {butter}             0.001016777  0.6250000 0.001626843 11.278670
## [10] {domestic eggs,
##       other vegetables,
##       tropical fruit,
##       whole milk,
##       yogurt}                 => {butter}             0.001016777  0.6250000 0.001626843 11.278670
## [11] {liquor,
##       red/blush wine}         => {bottled beer}       0.001931876  0.9047619 0.002135231 11.235269
## [12] {cream cheese,
##       other vegetables,
##       whipped/sour cream,
##       yogurt}                 => {curd}               0.001016777  0.5882353 0.001728521 11.040638
## [13] {hard cheese,
##       whipped/sour cream,
##       yogurt}                 => {butter}             0.001016777  0.5882353 0.001728521 10.615219
## [14] {other vegetables,
##       rolls/buns,
##       root vegetables,
##       tropical fruit,
##       whole milk}             => {beef}               0.001118454  0.5500000 0.002033554 10.483043
## [15] {sliced cheese,
##       tropical fruit,
##       whole milk,
##       yogurt}                 => {butter}             0.001016777  0.5555556 0.001830198 10.025484
## [16] {butter,
##       other vegetables,
##       sugar}                  => {whipped/sour cream} 0.001016777  0.7142857 0.001423488  9.964539
## [17] {hard cheese,
##       whipped/sour cream,
##       whole milk}             => {butter}             0.001423488  0.5384615 0.002643620  9.717008
## [18] {domestic eggs,
##       hard cheese,
##       other vegetables}       => {butter}             0.001016777  0.5263158 0.001931876  9.497827
## [19] {fruit/vegetable juice,
##       other vegetables,
##       tropical fruit,
##       whipped/sour cream}     => {butter}             0.001016777  0.5263158 0.001931876  9.497827
## [20] {onions,
##       tropical fruit,
##       yogurt}                 => {butter}             0.001118454  0.5238095 0.002135231  9.452599
## [21] {domestic eggs,
##       other vegetables,
##       tropical fruit,
##       yogurt}                 => {butter}            0.001118454  0.5238095 0.002135231  9.452599
## [22] {butter,
##       pastry,
##       yogurt}                 => {curd}              0.001016777  0.5000000 0.002033554  9.384542
## [23] {butter,
##       hard cheese,
##       whole milk}             => {whipped/sour cream} 0.001423488  0.6666667 0.002135231  9.300236
## [24] {butter,
##       fruit/vegetable juice,
##       other vegetables,
##       tropical fruit}         => {whipped/sour cream} 0.001016777  0.6666667 0.001525165  9.300236
## [25] {citrus fruit,
##       cream cheese,
##       other vegetables,
##       whole milk}             => {domestic eggs}     0.001118454  0.5789474 0.001931876  9.124916
## [26] {cream cheese,
##       curd,
##       whole milk,
##       yogurt}                 => {whipped/sour cream} 0.001118454  0.6470588 0.001728521  9.026700
## [27] {hard cheese,
##       other vegetables,
##       tropical fruit}         => {butter}            0.001016777  0.5000000 0.002033554  9.022936
## [28] {napkins,
##       other vegetables,
##       whipped/sour cream,
##       whole milk}             => {butter}            0.001016777  0.5000000 0.002033554  9.022936
## [29] {citrus fruit,
##       cream cheese,
##       whole milk}             => {domestic eggs}     0.001626843  0.5714286 0.002846975  9.006410
## [30] {frozen fish,
##       other vegetables,
##       tropical fruit}         => {pip fruit}         0.001016777  0.6666667 0.001525165  8.812724
```

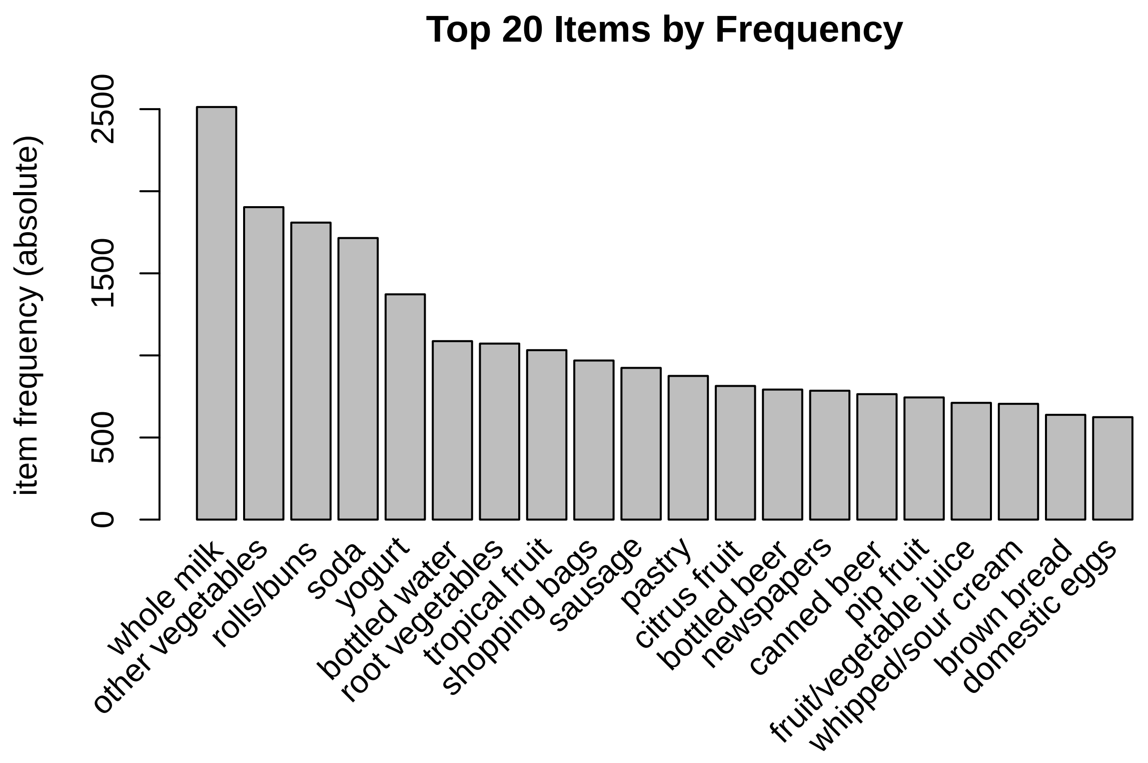
```r
plot(rules, method='two-key plot')
```

```
## To reduce overplotting, jitter is added! Use jitter = 0 to prevent jitter.
```

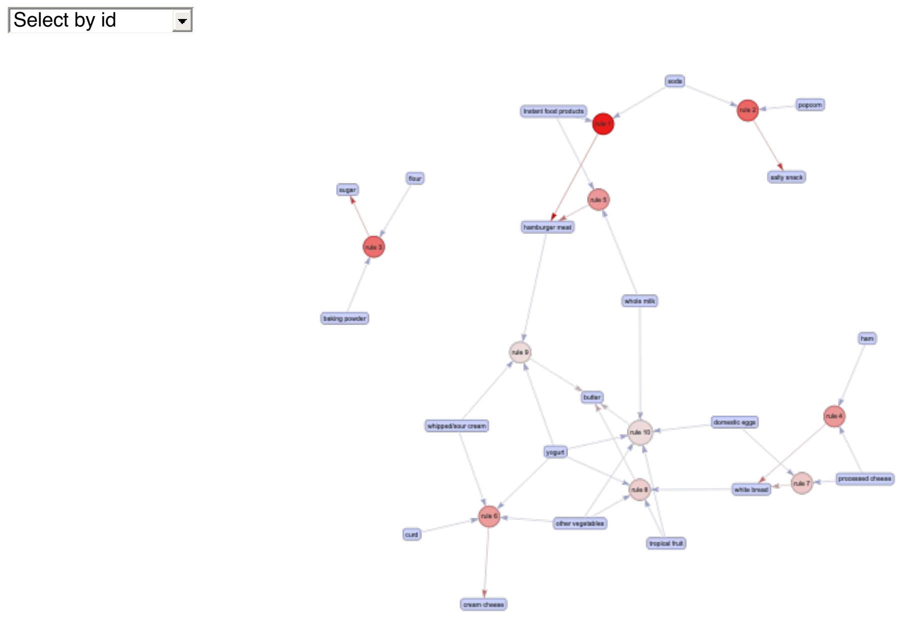


Two-Key plot shows that baskets with two rules are very rare. It shows a lot of 3, 4, and 5 order rules. 3 order rules tend to have lower values for confidence, but higher levels in support. 4 order rules tend to have a high range of confidence, but low support, indicating they don't appear very often. 5 order rules tend to also have a high range of confidence, but low support.

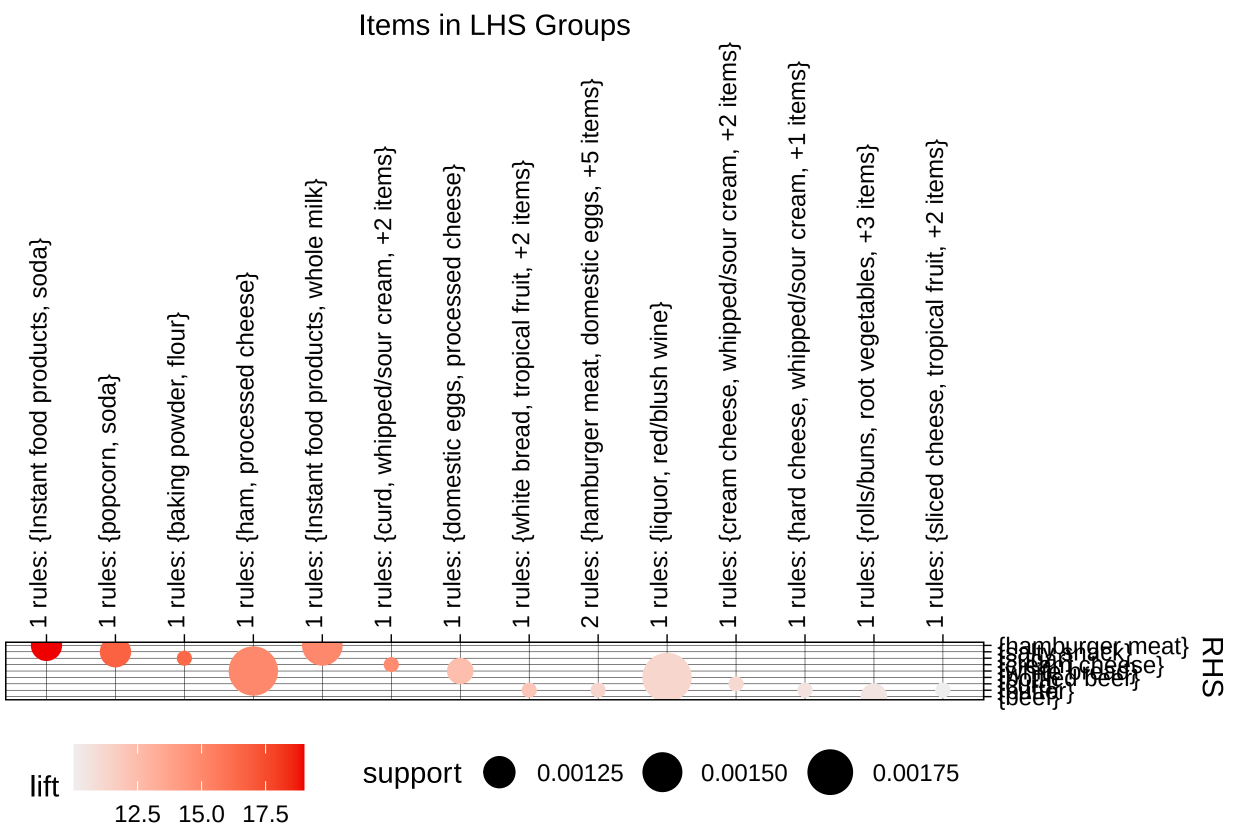
```r
# Load visualization package
library(arulesViz)

# Plot item frequencies
itemFrequencyPlot(transactions, topN = 20, type = "absolute", main = "Top 20 Items by Frequency")
```



```
# Plot rules using different methods
plot(sorted_rules[1:10], method = "graph", engine = "htmlwidget")
```



```
plot(sorted_rules[1:15], method = "grouped")
```



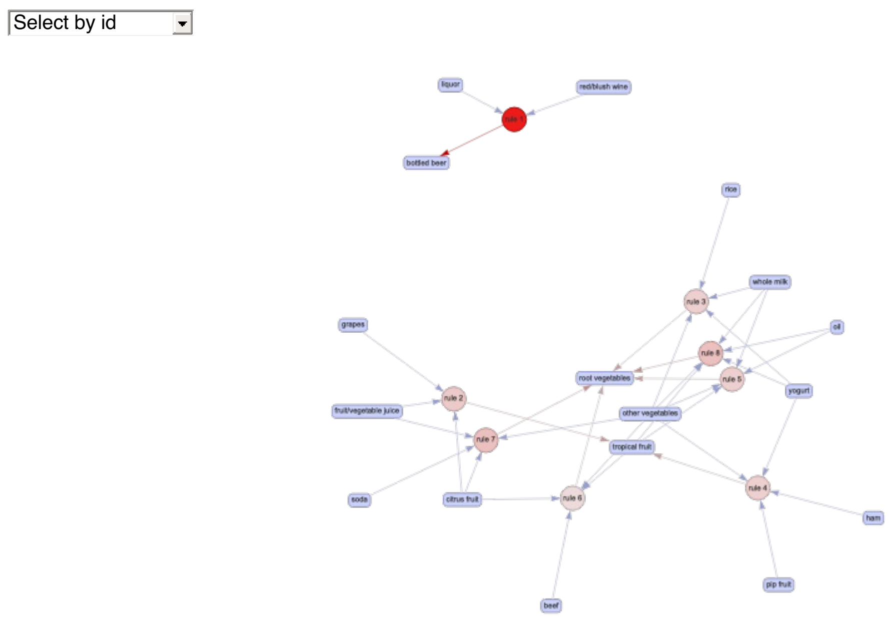
Looking at the item frequency plot, we can see the top items are whole milk, other vegetables, rolls/buns, soda, and yogurt. From here, the count of frequency tends to drop off. These items are likely in a lot of carts, and are likely to have a strong effect on the rules we see. An interesting point on the grouped dot plot shows a point with very high lift and high confidence, which correlates to the relationship between instant food products, soda, and hamburger meat. This may indicate someone who is shopping for cheaper items in the store.

Focusing on High Confidence, High Lift Combination

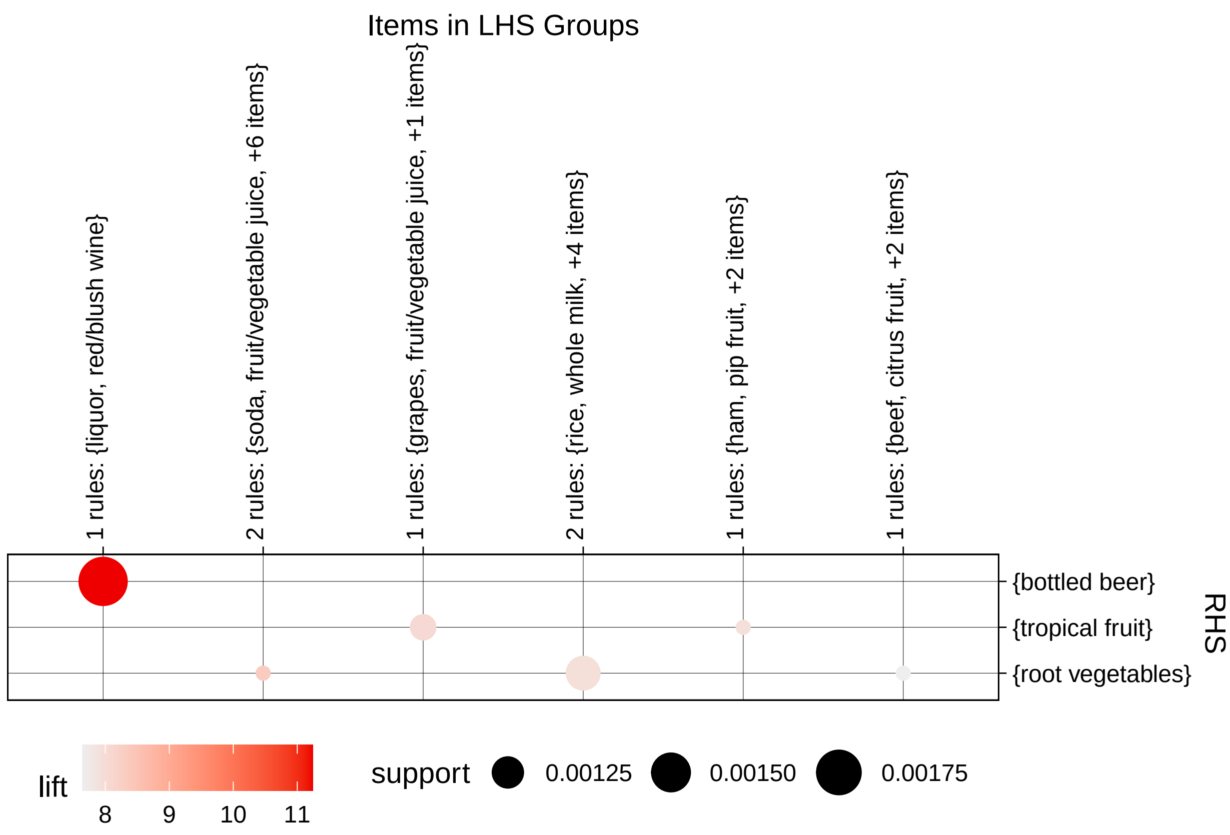
```
# Find the rules with high confidence and high lift
high_confidence_high_lift <- subset(rules, confidence > 0.8 & lift > 7.5)
arules::inspect(high_confidence_high_lift)
```

```
##     lhs                            rhs                  support confidence    coverage      lift count
## [1] {liquor,
##      red/blush wine}            => {bottled beer}    0.001931876  0.9047619 0.002135231 11.235269    19
## [2] {citrus fruit,
##      fruit/vegetable juice,
##      grapes}                    => {tropical fruit}  0.001118454  0.8461538 0.001321810  8.063879    11
## [3] {other vegetables,
##      rice,
##      whole milk,
##      yogurt}                    => {root vegetables} 0.001321810  0.8666667 0.001525165  7.951182    13
## [4] {ham,
##      other vegetables,
##      pip fruit,
##      yogurt}                  => {tropical fruit}  0.001016777  0.8333333 0.001220132  7.941699    10
## [5] {oil,
##      other vegetables,
##      tropical fruit,
##      whole milk}              => {root vegetables} 0.001321810  0.8666667 0.001525165  7.951182    13
## [6] {beef,
##      citrus fruit,
##      other vegetables,
##      tropical fruit}          => {root vegetables} 0.001016777  0.8333333 0.001220132  7.645367    10
## [7] {citrus fruit,
##      fruit/vegetable juice,
##      other vegetables,
##      soda}                    => {root vegetables} 0.001016777  0.9090909 0.001118454  8.340400    10
## [8] {oil,
##      other vegetables,
##      tropical fruit,
##      whole milk,
##      yogurt}                  => {root vegetables} 0.001016777  0.9090909 0.001118454  8.340400    10
```

```r
# visualizing these rules
plot(high_confidence_high_lift, method = "graph", engine = "htmlwidget")
```



```r
plot(high_confidence_high_lift[1:8], method = "grouped")
```



When we look at the rules with high confidence (>0.8) and high lift (>7.5) only 8 rules show up. In this case, we opted to choose high values of both to see if we could understand any deep patterns between items. When we evaluate the plot above, it seems like the high confidence, high lift, dots tend to correlate to a “general” shop, where customers seem to be buying the basic items (fruit, vegetables, meat, etc).

Interesting Rules to Note:

1. Alcohol Restock : (Highest Confidence and Lift!) - Liquor, Red/blush wine -> Bottled beer - This rule has a high confidence of 0.9 and a lift of 11.4. This means that if a customer buys liquor and red/blush wine, they are 11.4 times more likely to buy bottled beer as well. Likely customers restocking on alcohol or throwing a party!

2. Fruit Salad: Citrus fruit, fruit/vegetable juice, grapes -> tropical fruit - This rule has a high confidence of 0.85 and a lift of 8.5. This means that if a customer buys citrus fruit, fruit/vegetable juice, and grapes, they are 8.5 times more likely to buy tropical fruit as well. Likely customers buying a variety of fruits for a fruit salad or smoothie.

3. Casual Shop: Oil, other vegetables, tropical fruit, whole milk, yogurt -> root vegetables - This rule has a high confidence of 0.91 and a lift of 8.34. This grouping is likely customers shopping casually and buying the essentials for the week.

Question: Why did you choose the thresholds you did?

Thresholds Chosen:

- Support: 0.001 –> Originally tried a support level of 0.01, but this was too high and resulted in only 150 rules being generated. This support level of 0.001 generated 5000 rules when the only limiters were a support of 0.001 and confidence of 0.5.

- Lift: 7.5 –> Originally tried a lift of 1, but rules that seemed to be more “important” had very high lift values, anywhere from 8-22.

- Confidence: 0.5 –> Started with a general confidence choice of 0.5, but then used a subset of a confidence level of 0.9 to find the rules with the highest confidence and highest lift.

Question: Do your discovered item sets make sense?

Yes, the item sets that had the highest confidence and lift made sense in terms of the items grouped together. The item sets tend to me items with "interaction" such as if someone wants to make a sandwich, they are very likely to buy bread, cheese, and meat all together. Or if someone is buying a lot of fruits, they are likely to buy a variety of fruits to make a fruit salad or smoothie. If someone is hosting a party, they would tend to buy a variety of alcohol. These item sets make sense in terms of the context of the items grouped together.