Cap Metro Storytelling

2024-08-12

Visual story telling part 2: Capital Metro data

```
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(ggplot2)
```

```
## Warning: package 'ggplot2' was built under R version 4.3.2
```

```
library(tidyverse)
```

```
## Warning: package 'tidyr' was built under R version 4.3.2
```

```
## ── Attaching core tidyverse packages ──────────────────────── tidyverse 2.0.0 ──
## ✔ forcats   1.0.0     ✔ stringr   1.5.1
## ✔ lubridate 1.9.3     ✔ tibble    3.2.1
## ✔ purrr     1.0.2     ✔ tidyr     1.3.1
## ✔ readr     2.1.5
```

```
## ── Conflicts ────────────────────────────────────────── tidyverse_conflicts() ──
## ✖ dplyr::filter() masks stats::filter()
## ✖ dplyr::lag()    masks stats::lag()
## ℹ Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts
to become errors
```

```
library(tidyr)
library(corrplot)
```

```
## corrplot 0.92 loaded
```

```
library(dbscan)
```

```
## Warning: package 'dbscan' was built under R version 4.3.3
```

```
##
## Attaching package: 'dbscan'
##
## The following object is masked from 'package:stats':
##
##     as.dendrogram
```

The file capmetro_UT.csv contains data from Austin's own Capital Metro bus network, including shuttles to, from, and around the UT campus. These data track ridership on buses in the UT area. Ridership is measured by an optical scanner that counts how many people embark and alight the bus at each stop. Each row in the data set corresponds to a 15-minute period between the hours of 6 AM and 10 PM, each and every day, from September through November 2018. The variables are:

timestamp: the beginning of the 15-minute window for that row of data *boarding*: how many people got on board any Capital Metro bus on the UT campus in the specific 15 minute window *alighting*: how many people got off ("alit") any Capital Metro bus on the UT campus in the specific 15 minute window *day_of_week and weekend*: Monday, Tuesday, etc, as well as an indicator for whether it's a weekend. *temperature*: temperature at that time in degrees F *hour_of_day*: on 24-hour time, so 6 for 6 AM, 13 for 1 PM, 14 for 2 PM, etc. *month*: July through December

Objective) 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.

Note: You have broad freedom to look at any variables you'd like here – try to find that sweet spot where you're showing genuinely interesting relationships among more than just two variables, but where the resulting figure or set of figures doesn't become overwhelming/confusing. (Faceting/panel plots might be especially useful here.)

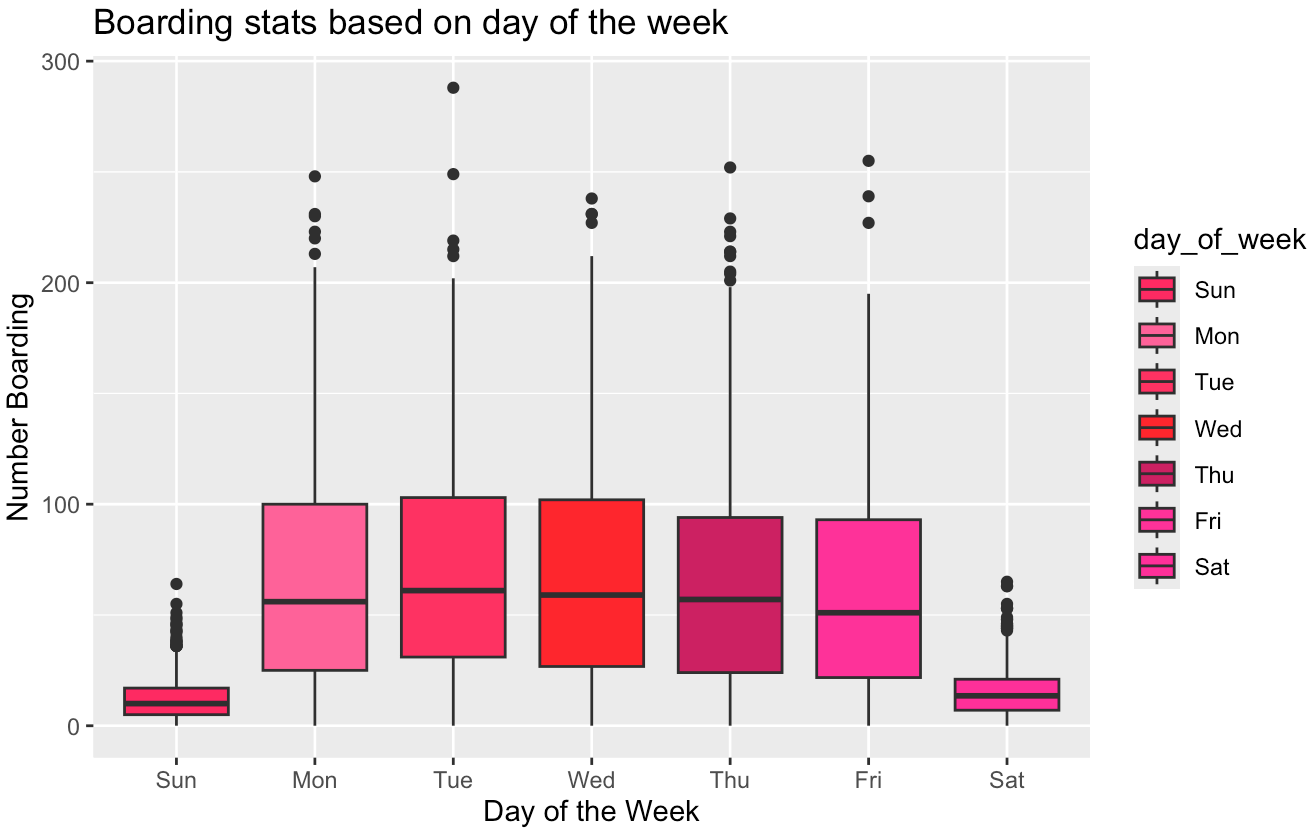
```
capmetro <- read.csv("capmetro_UT.csv")
head(capmetro, n=5)
```

	timestamp <chr>	boarding <int>	alighting <int>	day_of_week <chr>	temperature <dbl>	hour_of_day <int>	m… <chr>	we <ch
1	2018-09-01 06:00:00	0	1	Sat	74.82	6	Sep	wee
2	2018-09-01 06:15:00	2	1	Sat	74.82	6	Sep	wee
3	2018-09-01 06:30:00	3	4	Sat	74.82	6	Sep	wee
4	2018-09-01 06:45:00	3	4	Sat	74.82	6	Sep	wee

	timestamp	boarding	alighting	day_of_week	temperature	hour_of_day	m…	we
	<chr>	<int>	<int>	<chr>	<dbl>	<int>	<chr>	<ch
5	2018-09-01 07:00:00	2	4	Sat	74.39	7	Sep	wee

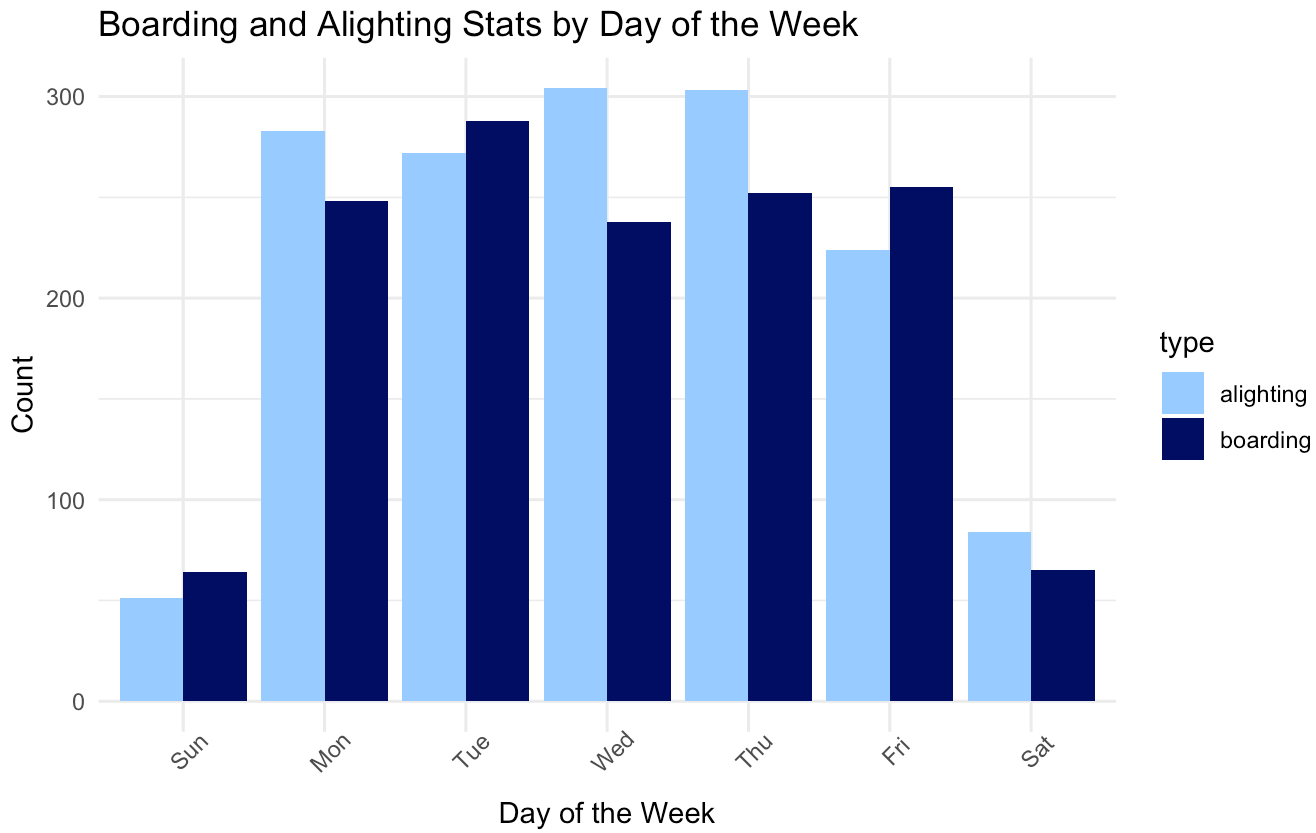
5 rows

```
#is.factor(capmetro$day_of_week)
# day_order <- c("Sunday", "Monday", "Tuesday", "Wednesday","Thursday", "Friday", "Saturday")
# capmetro$day_of_week <- factor(capmetro$day_of_week, levels = day_order)

#is.factor(capmetro$day_of_week)
#levels(capmetro$day_of_week)
capmetro$day_of_week <- factor(capmetro$day_of_week, levels = c("Sun", "Mon", "Tue", "Wed", "Thu", "Fri", "Sat"))
# levels(week_order)

ggplot(capmetro, aes(x = day_of_week, y = boarding, fill = day_of_week, order=day_of_week)) +
  geom_boxplot() +
  scale_fill_manual(values = c("#FF0066", "#FF6699", "#FF3366", "#FF0033", "#CC0066", "#FF3399", "#FF0099")) +  # Set custom colors
  labs(
    x = "Day of the Week",
    y = "Number Boarding",
    title = "Boarding stats based on day of the week",
    caption = "The boxplots for the weekdays appear to be more similar in their larger structure
    and spread than the boxpots for weekends. This could be because students are more likey to
    have classes during the weekdays and as such will need to use the UT bus system more often
    on those days. On the weekends they might not have much use for the UT bus."
    )
```



The boxplots for the weekdays appear to be more similar in their larger structure and spread than the boxpots for weekends. This could be because students are more likley to have classes during the weekdays and as such will need to use the UT bus system more often on those days. On the weekends they might not have much use for the UT bus.

```
capmetro_long <- capmetro %>%
  pivot_longer(cols = c(boarding, alighting), names_to = "type", values_to = "count")


ggplot(capmetro_long, aes(x = day_of_week, y = count, fill = type)) +
  geom_bar(stat = "identity", position = "dodge") +
  scale_fill_manual(values = c("boarding" = "#000066", "alighting" = "#99CCFF")) +  # Set custom colors
  labs(
    x = "Day of the Week",
    y = "Count",
    title = "Boarding and Alighting Stats by Day of the Week",
    caption = "Like the boxplots, there are more values for the weekdays. There is no constant or
    clear pattern between boarding and alighting; however, this could be due to further dependencies
    between temperature or hour of the day. If it is not as hot or if it is busy, students may potentiialy
    be walking instead of using the bus."
    ) +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45))  # Rotate x-axis labels for better readability
```



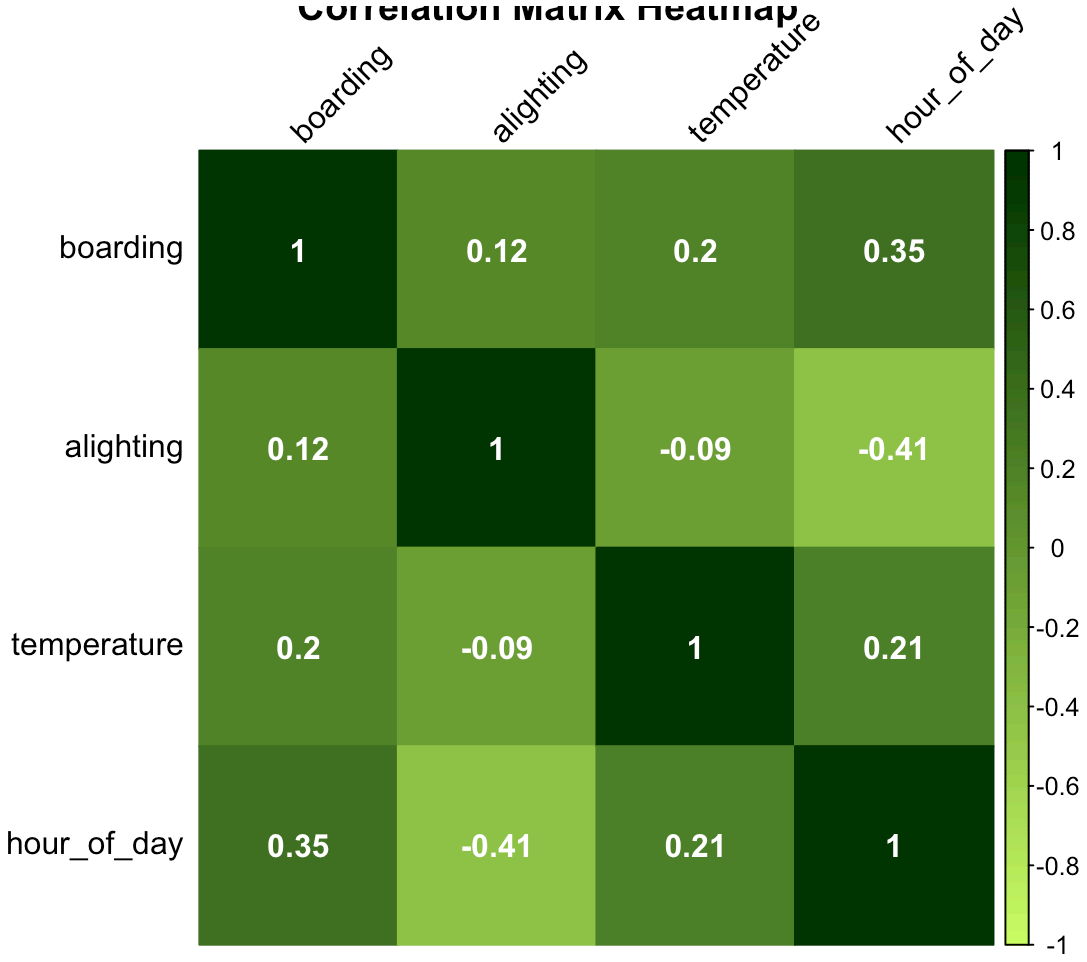
Like the boxplots, there are more values for the weekdays. There is no constant or clear pattern between boarding and alighting; however, this could be due to further dependencies between temperature or hour of the day. If it is not as hot or if it is busy, students may potentiialy be walking instead of using the bus.

```
numeric_cols <- capmetro[sapply(capmetro, is.numeric)]
corr_matrix <- cor(numeric_cols, use = "complete.obs")

# Plot the correlation matrix
corrplot(corr_matrix,
         method = "color",
         col = colorRampPalette(c("#CCFF66","#669933", "#003300"))(200),
         addCoef.col = "white",
         tl.col = "black",
         tl.srt = 45,
         title = "Correlation Matrix Heatmap",
        caption = "The correlation heatmap can help us to see if there are any
        strong relationships between variables. For example we can see if there
        is a correlation between boarding and temperature. Some observations
        from the heatmap that we can see is how boarding has the most and the
        strongest correlation with the other variables than the other variables
        do with each other.")
```

```
## Warning in text.default(pos.xlabel[, 1], pos.xlabel[, 2], newcolnames, srt =
## tl.srt, : "caption" is not a graphical parameter
```

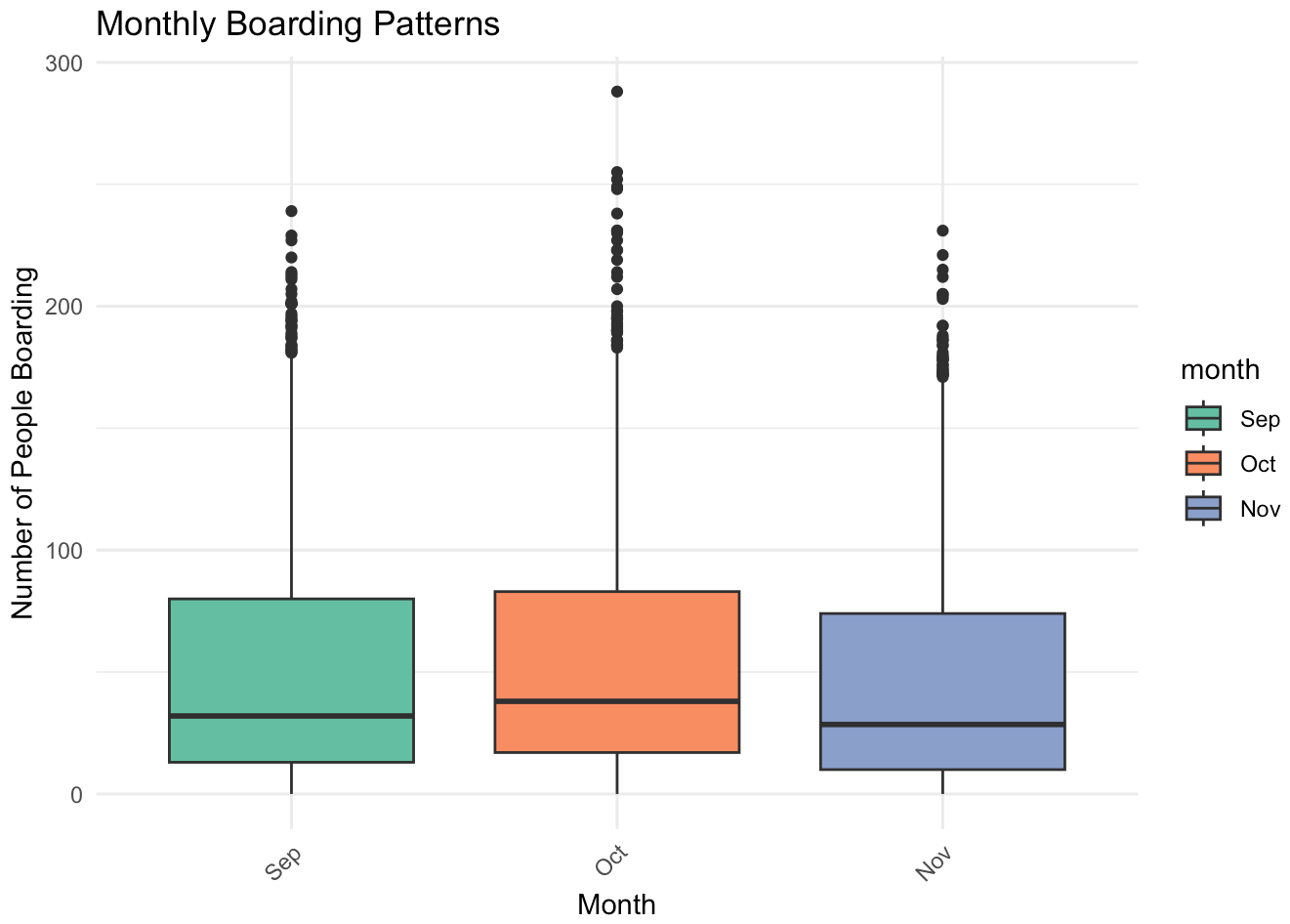
```
## Warning in text.default(pos.ylabel[, 1], pos.ylabel[, 2], newrownames, col =
## tl.col, : "caption" is not a graphical parameter
```

```
## Warning in title(title, ...): "caption" is not a graphical parameter
```

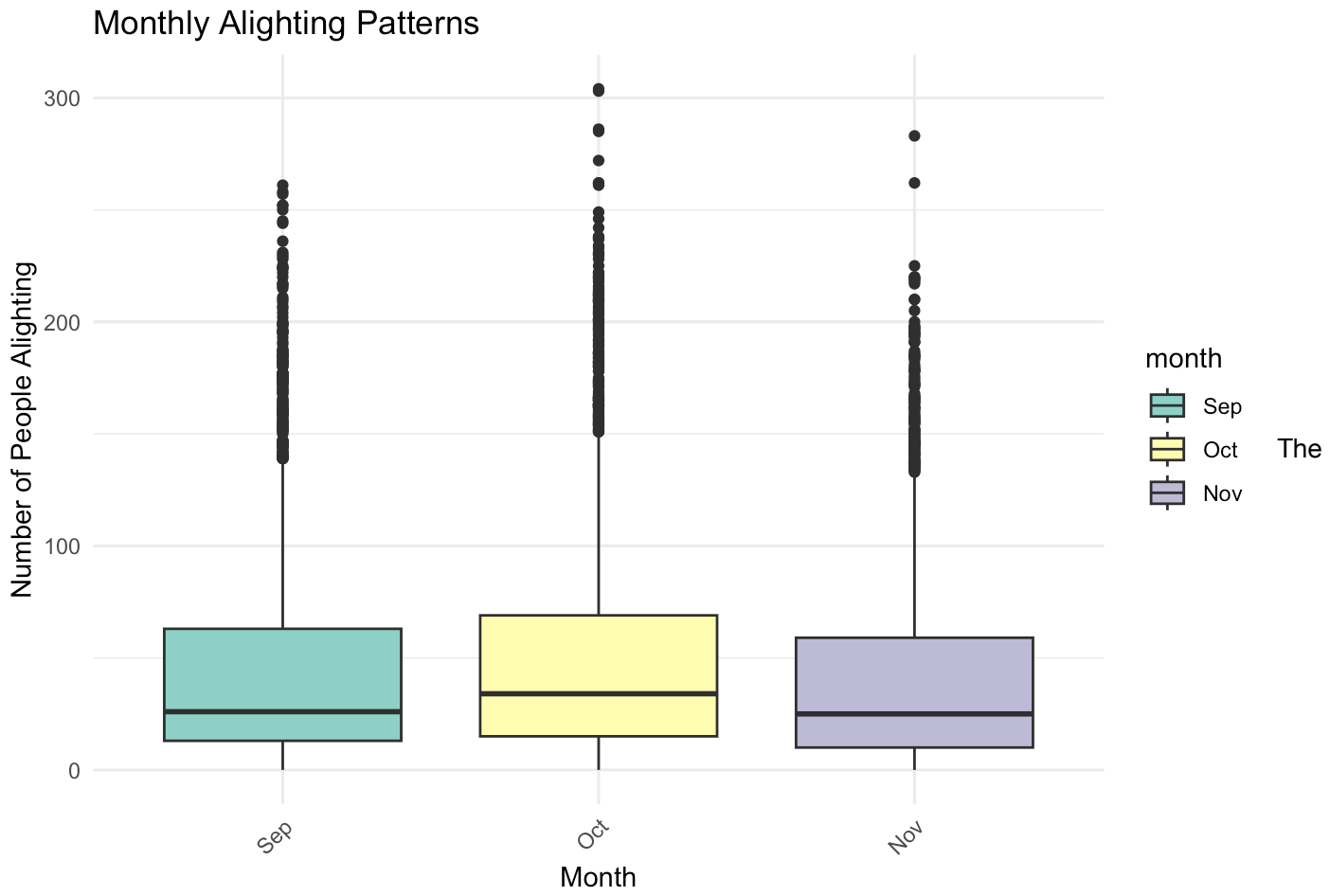


The correlation heatmap can help us to see if there are anystrong relationships between variables. For example we can see if there is a correlation between boarding and temperature. Some observations from the heatmap that we can see is how boarding has the most and the strongest correlation with the other variables than the other variables do with each other.

```
capmetro$month = factor(month(capmetro$timestamp, label = TRUE),
                        levels = c("Sep", "Oct", "Nov"))
par(mfrow = c(1,2))
ggplot(capmetro, aes(x = month, y = boarding, fill = month)) +
  geom_boxplot() +
  labs(title = "Monthly Boarding Patterns",
       x = "Month",
       y = "Number of People Boarding") +
  scale_fill_brewer(palette = "Set2") +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```

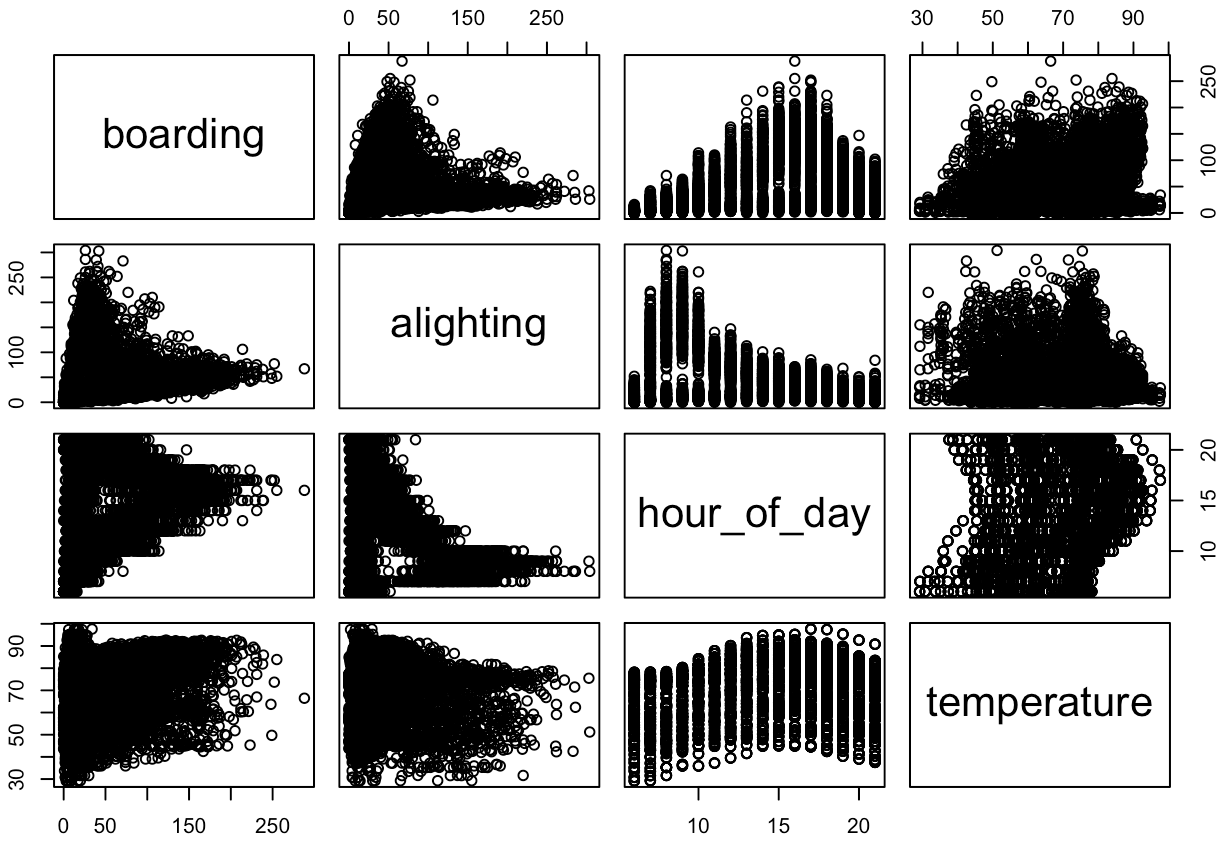



```
ggplot(capmetro, aes(x = month, y = alighting, fill = month)) +
  geom_boxplot() +
  labs(title = "Monthly Alighting Patterns",
       x = "Month",
       y = "Number of People Alighting") +
  scale_fill_brewer(palette = "Set3") +
  theme_minimal() +
  theme(axis.text.x = element_text(angle = 45, hjust = 1))
```



boxplot for boarding has a larger spread/range than the boxplot for the alighting. Both boxplots also have quite a few outliers with alighting having more than boarding.

```
capmetro <- read.csv("capmetro_UT.csv")
temperature <- capmetro$temperature
boarding <- capmetro$boarding
alighting <- capmetro$alighting
hour_of_day <- capmetro$hour_of_day
pairs(~ boarding + alighting + hour_of_day + temperature, data = capmetro)
```



This graph was not a big part of our answer for this question but we did it to explore the relationships between all the variables in our dataset in the form of scatterplots. We can see how some have greater variance and how others dont have the variance and are instead spread in vertical lines.

```
yearly_usage_stats = capmetro %>% group_by(month = month(timestamp)) %>% summarise(avgBo
arding = mean(boarding), avgAlighting = mean(alighting))


capmetro$month = month(capmetro$timestamp, label = TRUE)


mean_temperature = aggregate(temperature ~ month, data = capmetro, FUN = mean)

captionText = "Temperature plays an interesting factor in this dataset. Even though temp
eratures
start to decreases towrds November, the range/spread of students using the UT bus system
stays
about constant and does not increase. It does slightly increase between September to Oct
ober but
then falls back to regulalr values in November. This could be due to students traveling
or having
holidays and not having a greater need for the bus."
wrappedCaption = str_wrap(captionText, width = 80)


boardingPlot = ggplot(capmetro, aes(x = month, y = boarding, fill=month)) +
  geom_boxplot( )+
  scale_fill_brewer(palette = "Pastel1")
  theme_minimal()
```

```
## List of 136
##  $ line                            :List of 6
##   ..$ colour       : chr "black"
##   ..$ linewidth    : num 0.5
##   ..$ linetype     : num 1
##   ..$ lineend      : chr "butt"
##   ..$ arrow        : logi FALSE
##   ..$ inherit.blank: logi TRUE
##   ..- attr(*, "class")= chr [1:2] "element_line" "element"
##  $ rect                            :List of 5
##   ..$ fill         : chr "white"
##   ..$ colour       : chr "black"
##   ..$ linewidth    : num 0.5
##   ..$ linetype     : num 1
##   ..$ inherit.blank: logi TRUE
##   ..- attr(*, "class")= chr [1:2] "element_rect" "element"
##  $ text                            :List of 11
##   ..$ family       : chr ""
##   ..$ face         : chr "plain"
##   ..$ colour       : chr "black"
##   ..$ size         : num 11
##   ..$ hjust        : num 0.5
##   ..$ vjust        : num 0.5
##   ..$ angle        : num 0
##   ..$ lineheight   : num 0.9
##   ..$ margin       : 'margin' num [1:4] 0points 0points 0points 0points
##   .. ..- attr(*, "unit")= int 8
##   ..$ debug        : logi FALSE
##   ..$ inherit.blank: logi TRUE
##   ..- attr(*, "class")= chr [1:2] "element_text" "element"
##  $ title                           : NULL
##  $ aspect.ratio                    : NULL
##  $ axis.title                      : NULL
##  $ axis.title.x                    :List of 11
##   ..$ family       : NULL
##   ..$ face         : NULL
##   ..$ colour       : NULL
##   ..$ size         : NULL
##   ..$ hjust        : NULL
##   ..$ vjust        : num 1
##   ..$ angle        : NULL
##   ..$ lineheight   : NULL
##   ..$ margin       : 'margin' num [1:4] 2.75points 0points 0points 0points
##   .. ..- attr(*, "unit")= int 8
##   ..$ debug        : NULL
##   ..$ inherit.blank: logi TRUE
##   ..- attr(*, "class")= chr [1:2] "element_text" "element"
##  $ axis.title.x.top                :List of 11
##   ..$ family       : NULL
##   ..$ face         : NULL
##   ..$ colour       : NULL
##   ..$ size         : NULL
```

```
##   ..$ hjust        : NULL
##   ..$ vjust        : num 0
##   ..$ angle        : NULL
##   ..$ lineheight   : NULL
##   ..$ margin       : 'margin' num [1:4] 0points 0points 2.75points 0points
##   .. ..- attr(*, "unit")= int 8
##   ..$ debug        : NULL
##   ..$ inherit.blank: logi TRUE
##   ..- attr(*, "class")= chr [1:2] "element_text" "element"
##  $ axis.title.x.bottom             : NULL
##  $ axis.title.y                    :List of 11
##   ..$ family       : NULL
##   ..$ face         : NULL
##   ..$ colour       : NULL
##   ..$ size         : NULL
##   ..$ hjust        : NULL
##   ..$ vjust        : num 1
##   ..$ angle        : num 90
##   ..$ lineheight   : NULL
##   ..$ margin       : 'margin' num [1:4] 0points 2.75points 0points 0points
##   .. ..- attr(*, "unit")= int 8
##   ..$ debug        : NULL
##   ..$ inherit.blank: logi TRUE
##   ..- attr(*, "class")= chr [1:2] "element_text" "element"
##  $ axis.title.y.left               : NULL
##  $ axis.title.y.right              :List of 11
##   ..$ family       : NULL
##   ..$ face         : NULL
##   ..$ colour       : NULL
##   ..$ size         : NULL
##   ..$ hjust        : NULL
##   ..$ vjust        : num 1
##   ..$ angle        : num -90
##   ..$ lineheight   : NULL
##   ..$ margin       : 'margin' num [1:4] 0points 0points 0points 2.75points
##   .. ..- attr(*, "unit")= int 8
##   ..$ debug        : NULL
##   ..$ inherit.blank: logi TRUE
##   ..- attr(*, "class")= chr [1:2] "element_text" "element"
##  $ axis.text                       :List of 11
##   ..$ family       : NULL
##   ..$ face         : NULL
##   ..$ colour       : chr "grey30"
##   ..$ size         : 'rel' num 0.8
##   ..$ hjust        : NULL
##   ..$ vjust        : NULL
##   ..$ angle        : NULL
##   ..$ lineheight   : NULL
##   ..$ margin       : NULL
##   ..$ debug        : NULL
##   ..$ inherit.blank: logi TRUE
##   ..- attr(*, "class")= chr [1:2] "element_text" "element"
```

```
##  $ axis.text.x                   :List of 11
##   ..$ family       : NULL
##   ..$ face         : NULL
##   ..$ colour       : NULL
##   ..$ size         : NULL
##   ..$ hjust        : NULL
##   ..$ vjust        : num 1
##   ..$ angle        : NULL
##   ..$ lineheight   : NULL
##   ..$ margin       : 'margin' num [1:4] 2.2points 0points 0points 0points
##   .. ..- attr(*, "unit")= int 8
##   ..$ debug        : NULL
##   ..$ inherit.blank: logi TRUE
##   ..- attr(*, "class")= chr [1:2] "element_text" "element"
##  $ axis.text.x.top               :List of 11
##   ..$ family       : NULL
##   ..$ face         : NULL
##   ..$ colour       : NULL
##   ..$ size         : NULL
##   ..$ hjust        : NULL
##   ..$ vjust        : num 0
##   ..$ angle        : NULL
##   ..$ lineheight   : NULL
##   ..$ margin       : 'margin' num [1:4] 0points 0points 2.2points 0points
##   .. ..- attr(*, "unit")= int 8
##   ..$ debug        : NULL
##   ..$ inherit.blank: logi TRUE
##   ..- attr(*, "class")= chr [1:2] "element_text" "element"
##  $ axis.text.x.bottom            : NULL
##  $ axis.text.y                   :List of 11
##   ..$ family       : NULL
##   ..$ face         : NULL
##   ..$ colour       : NULL
##   ..$ size         : NULL
##   ..$ hjust        : num 1
##   ..$ vjust        : NULL
##   ..$ angle        : NULL
##   ..$ lineheight   : NULL
##   ..$ margin       : 'margin' num [1:4] 0points 2.2points 0points 0points
##   .. ..- attr(*, "unit")= int 8
##   ..$ debug        : NULL
##   ..$ inherit.blank: logi TRUE
##   ..- attr(*, "class")= chr [1:2] "element_text" "element"
##  $ axis.text.y.left              : NULL
##  $ axis.text.y.right             :List of 11
##   ..$ family       : NULL
##   ..$ face         : NULL
##   ..$ colour       : NULL
##   ..$ size         : NULL
##   ..$ hjust        : num 0
##   ..$ vjust        : NULL
##   ..$ angle        : NULL
```

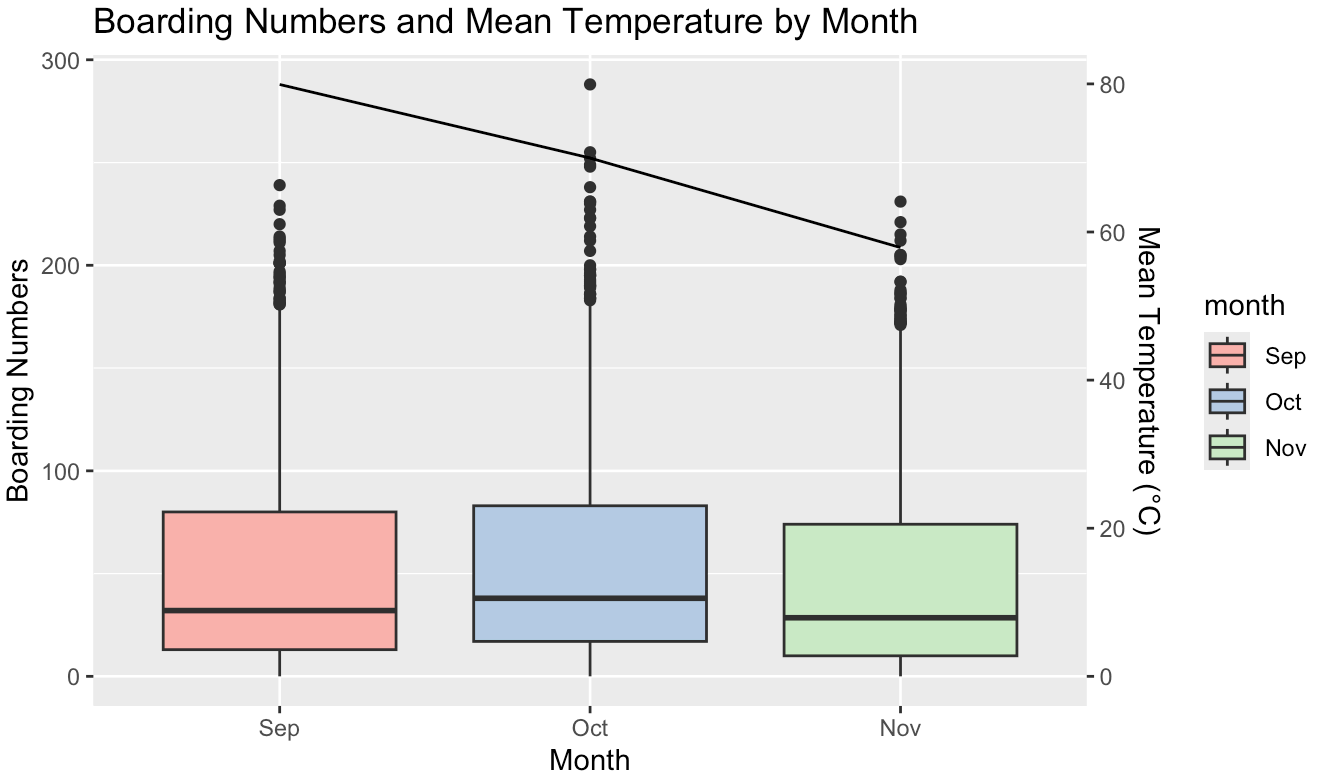
```
##   ..$ lineheight   : NULL
##   ..$ margin       : 'margin' num [1:4] 0points 0points 0points 2.2points
##   .. ..- attr(*, "unit")= int 8
##   ..$ debug        : NULL
##   ..$ inherit.blank: logi TRUE
##   ..- attr(*, "class")= chr [1:2] "element_text" "element"
##  $ axis.text.theta                : NULL
##  $ axis.text.r                    :List of 11
##   ..$ family       : NULL
##   ..$ face         : NULL
##   ..$ colour       : NULL
##   ..$ size         : NULL
##   ..$ hjust        : num 0.5
##   ..$ vjust        : NULL
##   ..$ angle        : NULL
##   ..$ lineheight   : NULL
##   ..$ margin       : 'margin' num [1:4] 0points 2.2points 0points 2.2points
##   .. ..- attr(*, "unit")= int 8
##   ..$ debug        : NULL
##   ..$ inherit.blank: logi TRUE
##   ..- attr(*, "class")= chr [1:2] "element_text" "element"
##  $ axis.ticks                     : list()
##   ..- attr(*, "class")= chr [1:2] "element_blank" "element"
##  $ axis.ticks.x                   : NULL
##  $ axis.ticks.x.top               : NULL
##  $ axis.ticks.x.bottom            : NULL
##  $ axis.ticks.y                   : NULL
##  $ axis.ticks.y.left              : NULL
##  $ axis.ticks.y.right             : NULL
##  $ axis.ticks.theta               : NULL
##  $ axis.ticks.r                   : NULL
##  $ axis.minor.ticks.x.top         : NULL
##  $ axis.minor.ticks.x.bottom      : NULL
##  $ axis.minor.ticks.y.left        : NULL
##  $ axis.minor.ticks.y.right       : NULL
##  $ axis.minor.ticks.theta         : NULL
##  $ axis.minor.ticks.r             : NULL
##  $ axis.ticks.length              : 'simpleUnit' num 2.75points
##   ..- attr(*, "unit")= int 8
##  $ axis.ticks.length.x            : NULL
##  $ axis.ticks.length.x.top        : NULL
##  $ axis.ticks.length.x.bottom     : NULL
##  $ axis.ticks.length.y            : NULL
##  $ axis.ticks.length.y.left       : NULL
##  $ axis.ticks.length.y.right      : NULL
##  $ axis.ticks.length.theta        : NULL
##  $ axis.ticks.length.r            : NULL
##  $ axis.minor.ticks.length        : 'rel' num 0.75
##  $ axis.minor.ticks.length.x      : NULL
##  $ axis.minor.ticks.length.x.top  : NULL
##  $ axis.minor.ticks.length.x.bottom: NULL
##  $ axis.minor.ticks.length.y      : NULL
```

```
##  $ axis.minor.ticks.length.y.left  : NULL
##  $ axis.minor.ticks.length.y.right : NULL
##  $ axis.minor.ticks.length.theta   : NULL
##  $ axis.minor.ticks.length.r       : NULL
##  $ axis.line                       : list()
##   ..- attr(*, "class")= chr [1:2] "element_blank" "element"
##  $ axis.line.x                     : NULL
##  $ axis.line.x.top                 : NULL
##  $ axis.line.x.bottom              : NULL
##  $ axis.line.y                     : NULL
##  $ axis.line.y.left                : NULL
##  $ axis.line.y.right               : NULL
##  $ axis.line.theta                 : NULL
##  $ axis.line.r                     : NULL
##  $ legend.background               : list()
##   ..- attr(*, "class")= chr [1:2] "element_blank" "element"
##  $ legend.margin                   : 'margin' num [1:4] 5.5points 5.5points 5.5points
5.5points
##   ..- attr(*, "unit")= int 8
##  $ legend.spacing                  : 'simpleUnit' num 11points
##   ..- attr(*, "unit")= int 8
##  $ legend.spacing.x                : NULL
##  $ legend.spacing.y                : NULL
##  $ legend.key                      : list()
##   ..- attr(*, "class")= chr [1:2] "element_blank" "element"
##  $ legend.key.size                 : 'simpleUnit' num 1.2lines
##   ..- attr(*, "unit")= int 3
##  $ legend.key.height               : NULL
##  $ legend.key.width                : NULL
##  $ legend.key.spacing              : 'simpleUnit' num 5.5points
##   ..- attr(*, "unit")= int 8
##  $ legend.key.spacing.x            : NULL
##  $ legend.key.spacing.y            : NULL
##  $ legend.frame                    : NULL
##  $ legend.ticks                    : NULL
##  $ legend.ticks.length             : 'rel' num 0.2
##  $ legend.axis.line                : NULL
##  $ legend.text                     :List of 11
##   ..$ family       : NULL
##   ..$ face         : NULL
##   ..$ colour       : NULL
##   ..$ size         : 'rel' num 0.8
##   ..$ hjust        : NULL
##   ..$ vjust        : NULL
##   ..$ angle        : NULL
##   ..$ lineheight   : NULL
##   ..$ margin       : NULL
##   ..$ debug        : NULL
##   ..$ inherit.blank: logi TRUE
##   ..- attr(*, "class")= chr [1:2] "element_text" "element"
##  $ legend.text.position            : NULL
##  $ legend.title                    :List of 11
```

```
##   ..$ family       : NULL
##   ..$ face         : NULL
##   ..$ colour       : NULL
##   ..$ size         : NULL
##   ..$ hjust        : num 0
##   ..$ vjust        : NULL
##   ..$ angle        : NULL
##   ..$ lineheight   : NULL
##   ..$ margin       : NULL
##   ..$ debug        : NULL
##   ..$ inherit.blank: logi TRUE
##   ..- attr(*, "class")= chr [1:2] "element_text" "element"
##  $ legend.title.position           : NULL
##  $ legend.position                 : chr "right"
##  $ legend.position.inside          : NULL
##  $ legend.direction                : NULL
##  $ legend.byrow                    : NULL
##  $ legend.justification            : chr "center"
##  $ legend.justification.top        : NULL
##  $ legend.justification.bottom     : NULL
##  $ legend.justification.left       : NULL
##  $ legend.justification.right      : NULL
##  $ legend.justification.inside     : NULL
##  $ legend.location                 : NULL
##  $ legend.box                      : NULL
##  $ legend.box.just                 : NULL
##  $ legend.box.margin               : 'margin' num [1:4] 0cm 0cm 0cm 0cm
##   ..- attr(*, "unit")= int 1
##  $ legend.box.background           : list()
##   ..- attr(*, "class")= chr [1:2] "element_blank" "element"
##  $ legend.box.spacing              : 'simpleUnit' num 11points
##   ..- attr(*, "unit")= int 8
##   [list output truncated]
##  - attr(*, "class")= chr [1:2] "theme" "gg"
##  - attr(*, "complete")= logi TRUE
##  - attr(*, "validate")= logi TRUE
```

```
scalingFactor = max(capmetro$boarding) / max(mean_temperature$temperature)


boardingPlot +
  geom_line(data = mean_temperature, aes(x = month, y = temperature * scalingFactor, gro
up = 1), color = "black") +
  scale_y_continuous(name = "Boarding Numbers",
                     sec.axis = sec_axis(trans = ~ . / scalingFactor, name = "Mean Tempe
rature (°C)")) +
  labs(title = "Boarding Numbers and Mean Temperature by Month",
       x = "Month",
       caption = wrappedCaption) +
  theme(plot.caption = element_text(hjust = 0))
```

```
## Warning: The `trans` argument of `sec_axis()` is deprecated as of ggplot2 3.5.0.
## ℹ Please use the `transform` argument instead.
## This warning is displayed once every 8 hours.
## Call `lifecycle::last_lifecycle_warnings()` to see where this warning was
## generated.
```



Temperature plays an interesting factor in this dataset. Even though temperatures start to decreases towrds November, the range/spread of students using the UT bus system stays about constant and does not increase. It does slightly increase between September to October but then falls back to regulalr values in November. This could be due to students traveling or having holidays and not having a greater need for the bus.

These are the graphs we selected to explore and display the relationships between different variables in the capmetro dataset. We found some interesting and surprising conclusions and the relationship between the time of the year, the temperature, and etc… and how it influences boarding and alighting stats.