Course Evaluations Analysis

Ananya Iyer (ai6792)

2025-01-28

```
# Load libraries
library(ggplot2)
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(kableExtra)
```

```
##
## Attaching package: 'kableExtra'

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

Problem 1: Beauty, or not, in the classroom

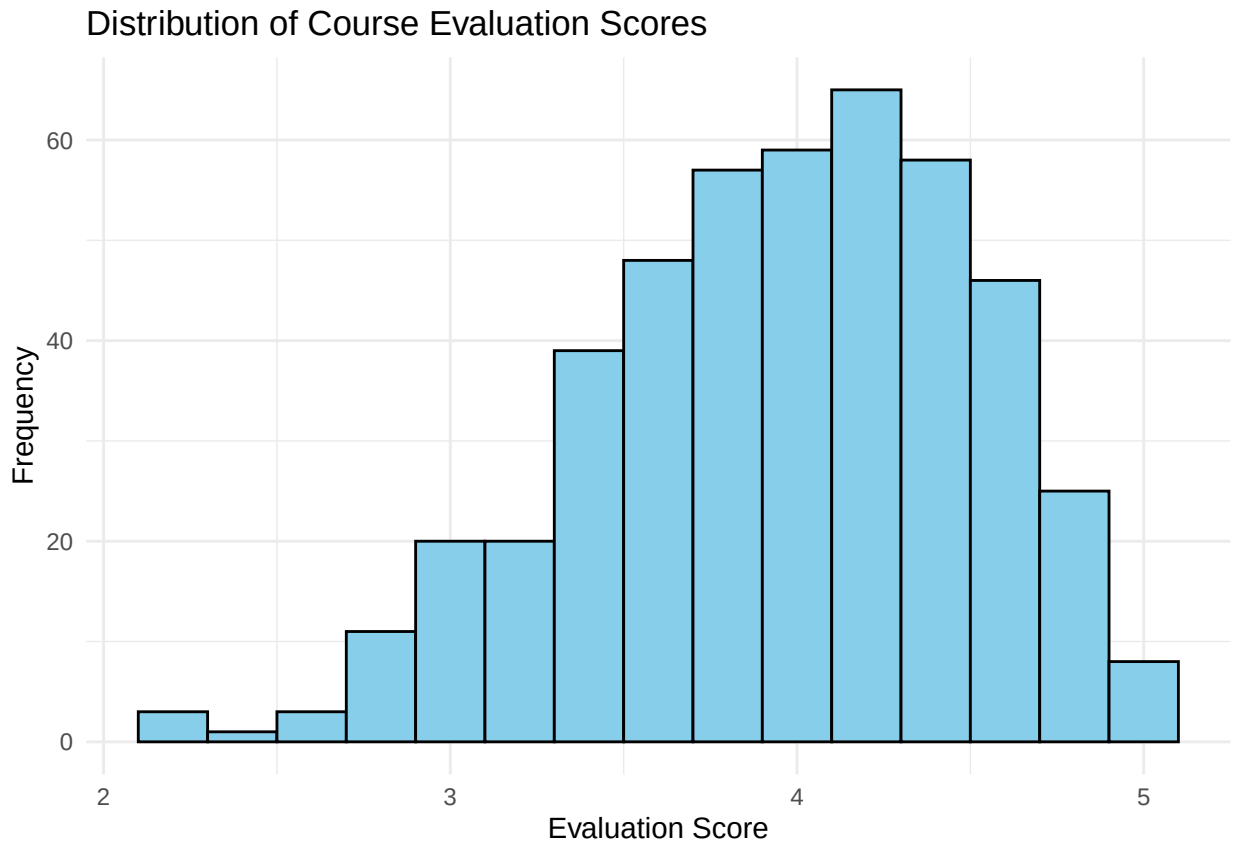
```
#Read data
profs <- read.csv("profs.csv")

#look at structure
str(profs)
```

```
## 'data.frame':    463 obs. of  12 variables:
##  $ minority   : chr  "yes" "no" "no" "no" ...
##  $ age        : int  36 59 51 40 31 62 33 51 33 47 ...
##  $ gender     : chr  "female" "male" "male" "female" ...
##  $ credits    : chr  "more" "more" "more" "more" ...
```

```
##  $ beauty     : num  0.29 -0.738 -0.572 -0.678 1.51 ...
##  $ eval       : num  4.3 4.5 3.7 4.3 4.4 4.2 4 3.4 4.5 3.9 ...
##  $ division   : chr  "upper" "upper" "upper" "upper" ...
##  $ native     : chr  "yes" "yes" "yes" "yes" ...
##  $ tenure     : chr  "yes" "yes" "yes" "yes" ...
##  $ students   : int  24 17 55 40 42 182 33 25 48 16 ...
##  $ allstudents: int  43 20 55 46 48 282 41 41 60 19 ...
##  $ prof       : int  1 2 3 4 5 6 7 8 9 10 ...
```

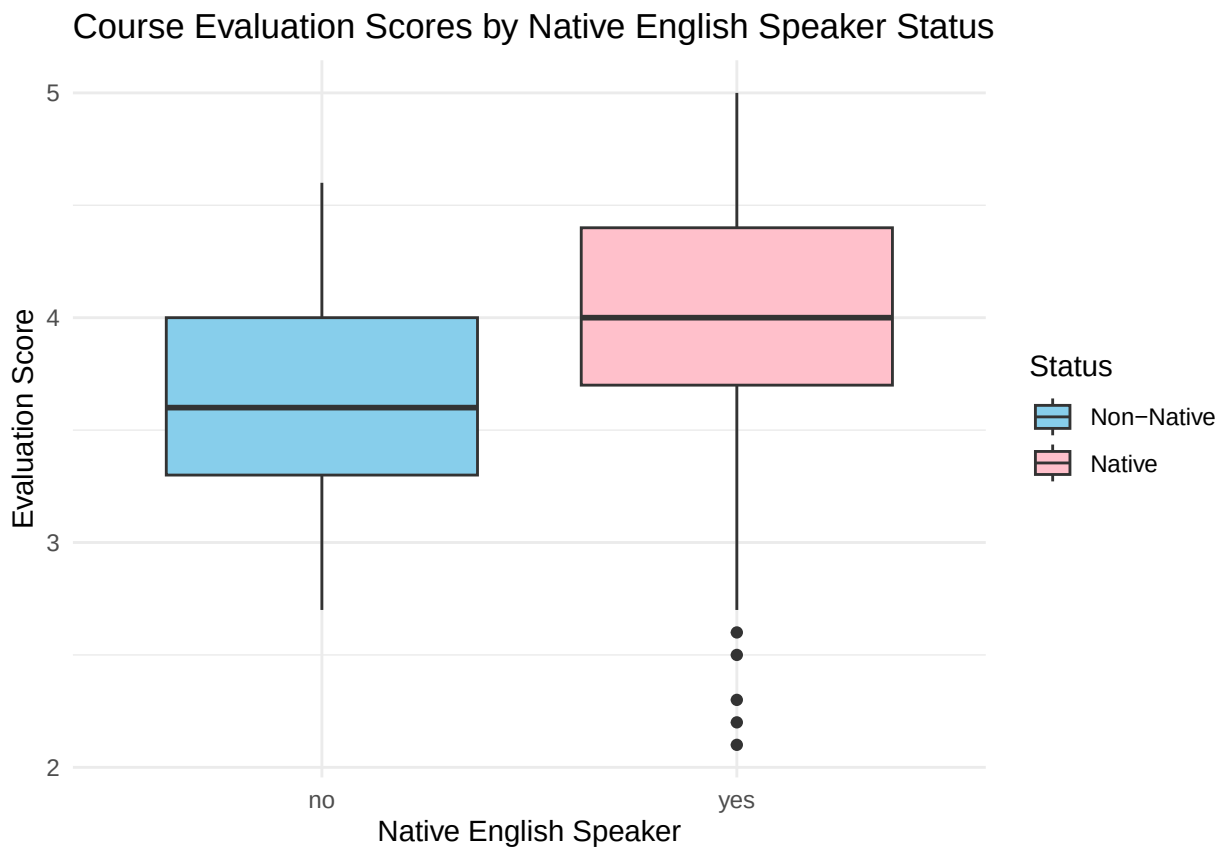
Part A

```
#Histogram
ggplot(profs, aes(x = eval)) +
  geom_histogram(binwidth = 0.2, fill = "skyblue", color = "black") +
  labs(title = "Distribution of Course Evaluation Scores",
       x = "Evaluation Score",
       y = "Frequency") +
  theme_minimal()
```



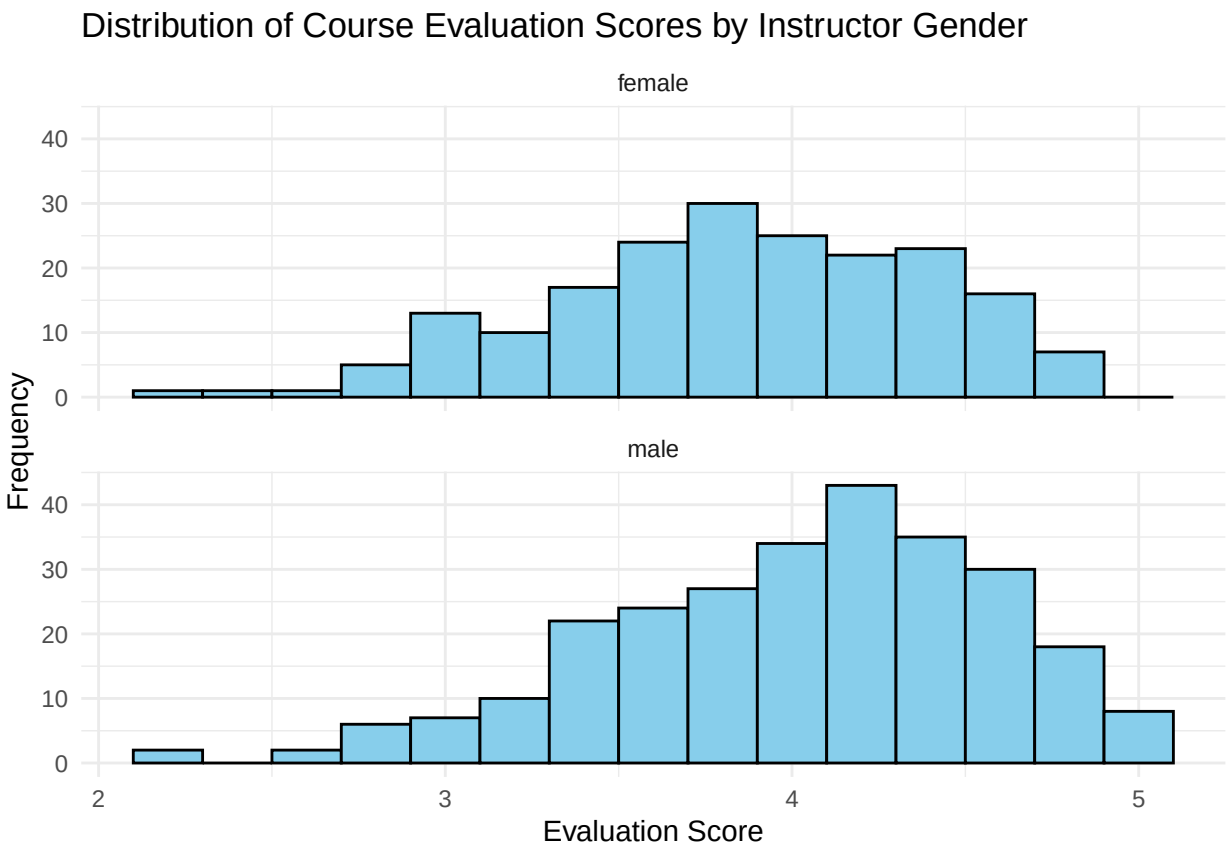
Part B

```r
#Boxplots by speaker status
ggplot(profs, aes(x = as.factor(native), y = eval, fill = as.factor(native))) +
  geom_boxplot() +
  scale_fill_manual(values = c("skyblue", "pink"), labels = c("Non-Native", "Native")) +
  labs(title = "Course Evaluation Scores by Native English Speaker Status",
       x = "Native English Speaker",
       y = "Evaluation Score",
       fill = "Status") +
  theme_minimal()
```



Part C

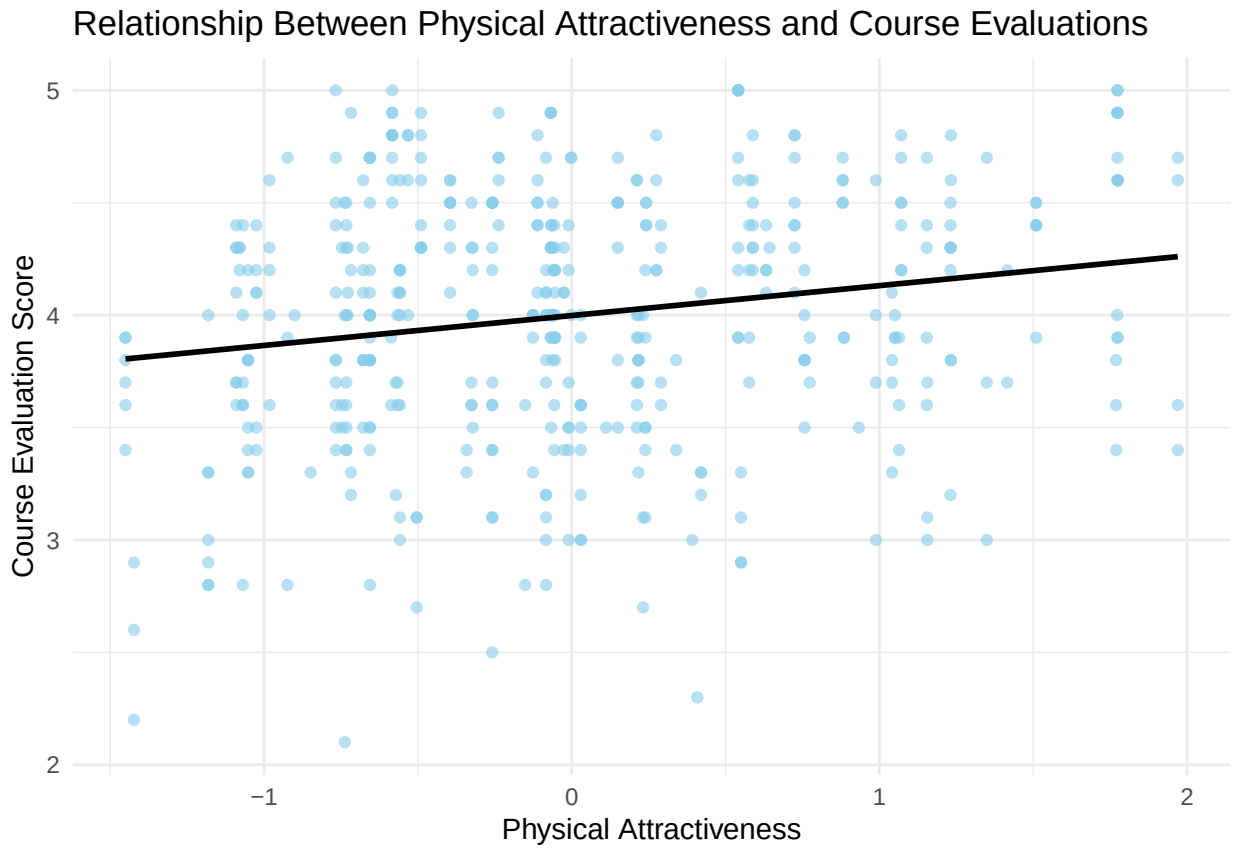
```r
#Faceted histograms by gender
ggplot(profs, aes(x = eval)) +
  geom_histogram(binwidth = 0.2, fill = "skyblue", color = "black") +
  facet_wrap(~ gender, nrow = 2) +
  labs(title = "Distribution of Course Evaluation Scores by Instructor Gender",
       x = "Evaluation Score",
       y = "Frequency") +
  theme_minimal()
```



Part D

```
#Scatterplot for beauty vs. eval
ggplot(profs, aes(x = beauty, y = eval)) +
  geom_point(color = "skyblue", alpha = 0.6) +
  geom_smooth(method = "lm", color = "black", se = FALSE) +
  labs(title = "Relationship Between Physical Attractiveness and Course Evaluations",
       x = "Physical Attractiveness",
       y = "Course Evaluation Score") +
  theme_minimal()
```

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



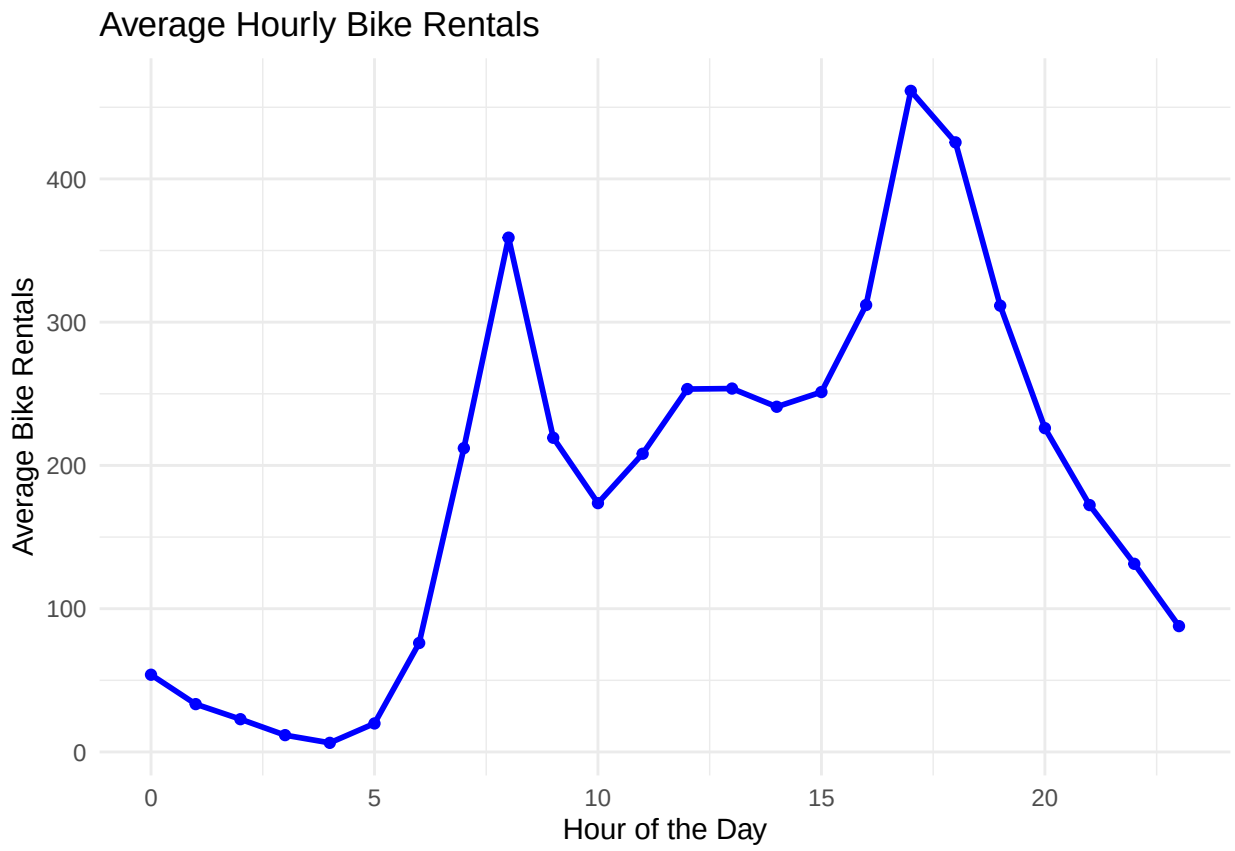
Problem 2: bike sharing

```
#Read data
bikeshare <- read.csv("bikeshare.csv")
```

Plot A

```
#Calculate average rentals by hour
plot_a_data <- bikeshare %>%
  group_by(hr) %>%
  summarize(avg_rentals = mean(total))

#Line plot
ggplot(plot_a_data, aes(x = hr, y = avg_rentals)) +
  geom_line(color = "blue", size = 1) +
  geom_point(color = "blue") +
  labs(
    title = "Average Hourly Bike Rentals",
    x = "Hour of the Day",
    y = "Average Bike Rentals"
  ) +
  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.
```



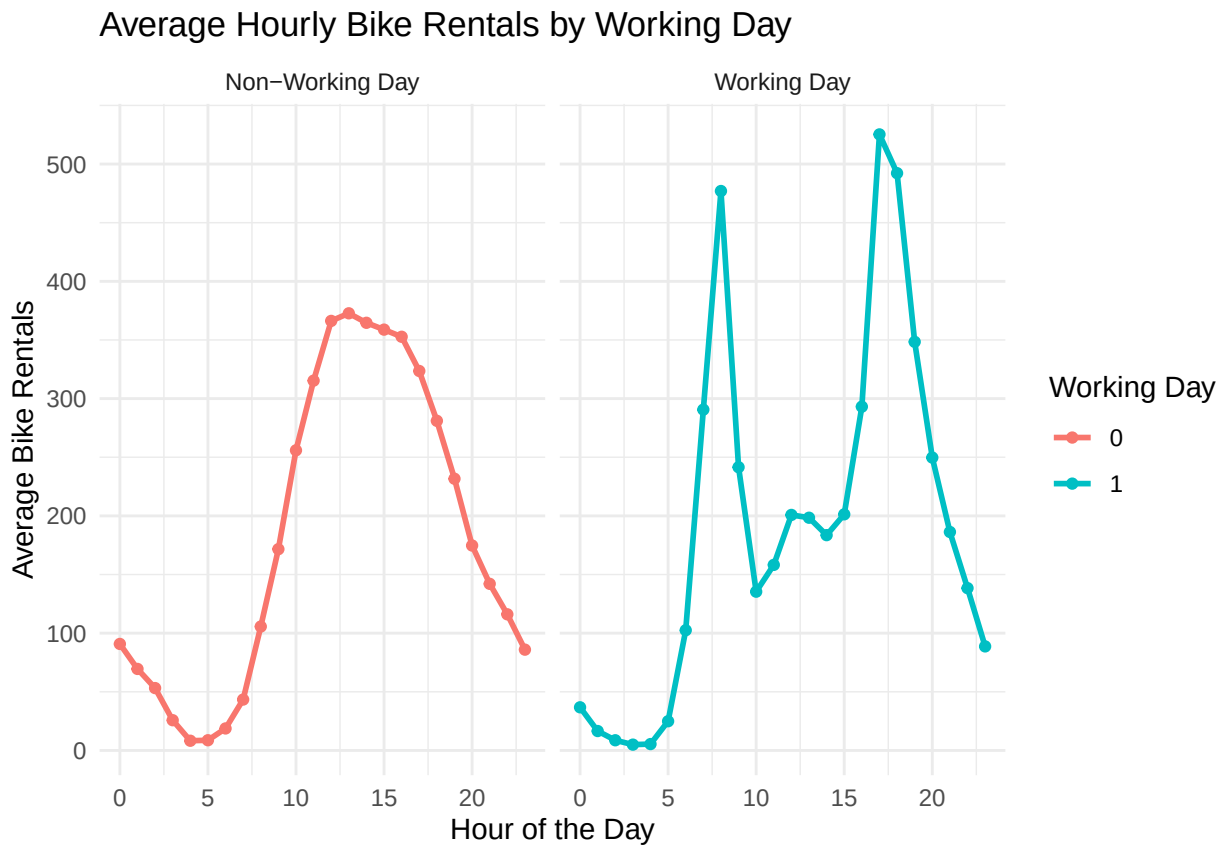
Plot B

```
# Calculate average rentals by hour and working day
plot_b_data <- bikeshare %>%
  group_by(hr, workingday) %>%
  summarize(avg_rentals = mean(total))
```

```
## ‘summarise()‘ has grouped output by ’hr’. You can override using the ‘.groups‘
## argument.
```

```
#Faceted line plot
ggplot(plot_b_data, aes(x = hr, y = avg_rentals, color = factor(workingday))) +
  geom_line(size = 1) +
  geom_point() +
```

```
  facet_wrap(~workingday, labeller = as_labeller(c(`0` = "Non-Working Day", `1` = "Working Day"))) +
  labs(
    title = "Average Hourly Bike Rentals by Working Day",
    x = "Hour of the Day",
    y = "Average Bike Rentals",
    color = "Working Day"
  ) +
  theme_minimal()
```



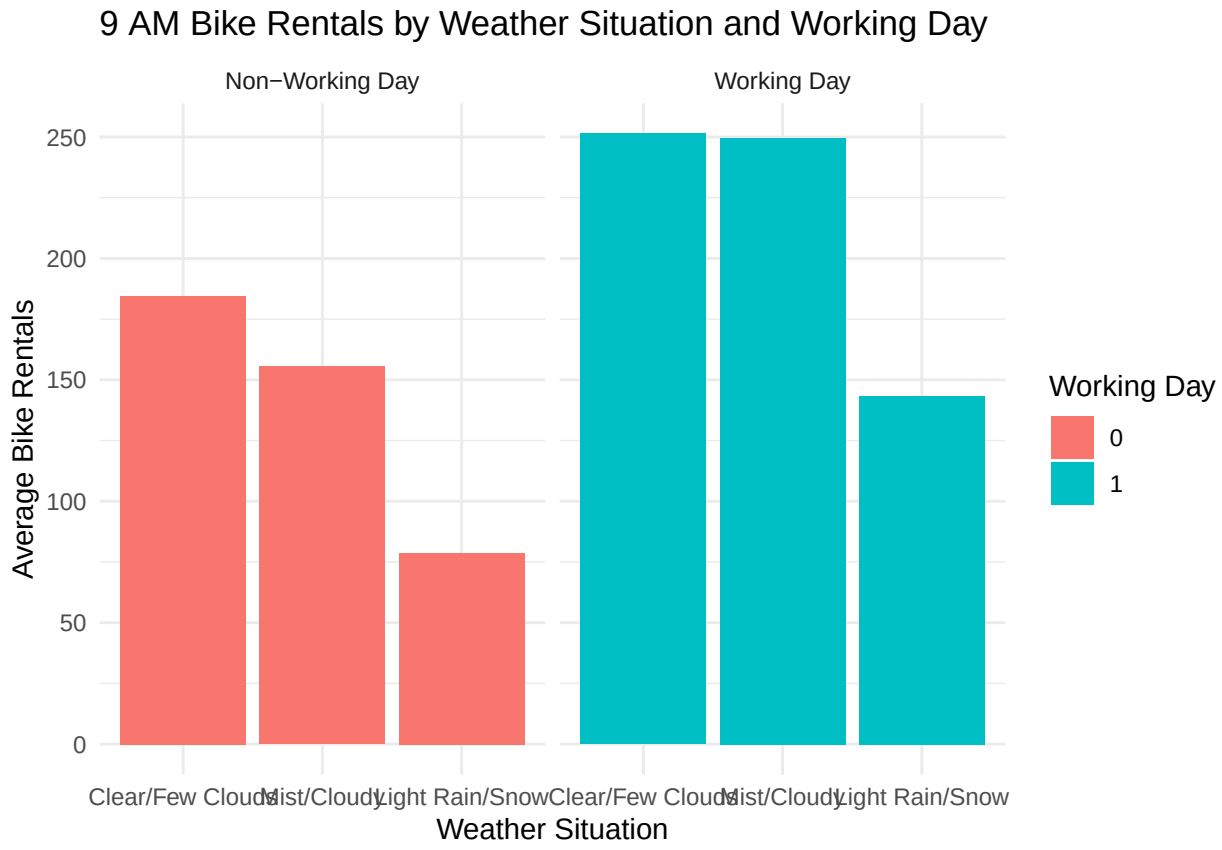
Plot C

```
#Filter and calculate average rentals by weather and working day
plot_c_data <- bikeshare %>%
  filter(hr == 9) %>%
  group_by(weathersit, workingday) %>%
  summarize(avg_rentals = mean(total))
```

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

```
#Faceted bar plot
ggplot(plot_c_data, aes(x = factor(weathersit), y = avg_rentals, fill = factor(workingday))) +
```

```
  geom_bar(stat = "identity", position = "dodge") +
  facet_wrap(~workingday, labeller = as_labeller(c(`0` = "Non-Working Day", `1` = "Working Day"))) +
  labs(
    title = "9 AM Bike Rentals by Weather Situation and Working Day",
    x = "Weather Situation",
    y = "Average Bike Rentals",
    fill = "Working Day"
  ) +
  scale_x_discrete(labels = c("1" = "Clear/Few Clouds", "2" = "Mist/Cloudy",
                              "3" = "Light Rain/Snow", "4" = "Heavy Rain/Fog")) +
  theme_minimal()
```



**Problem 3: Capital Metro UT Ridership*

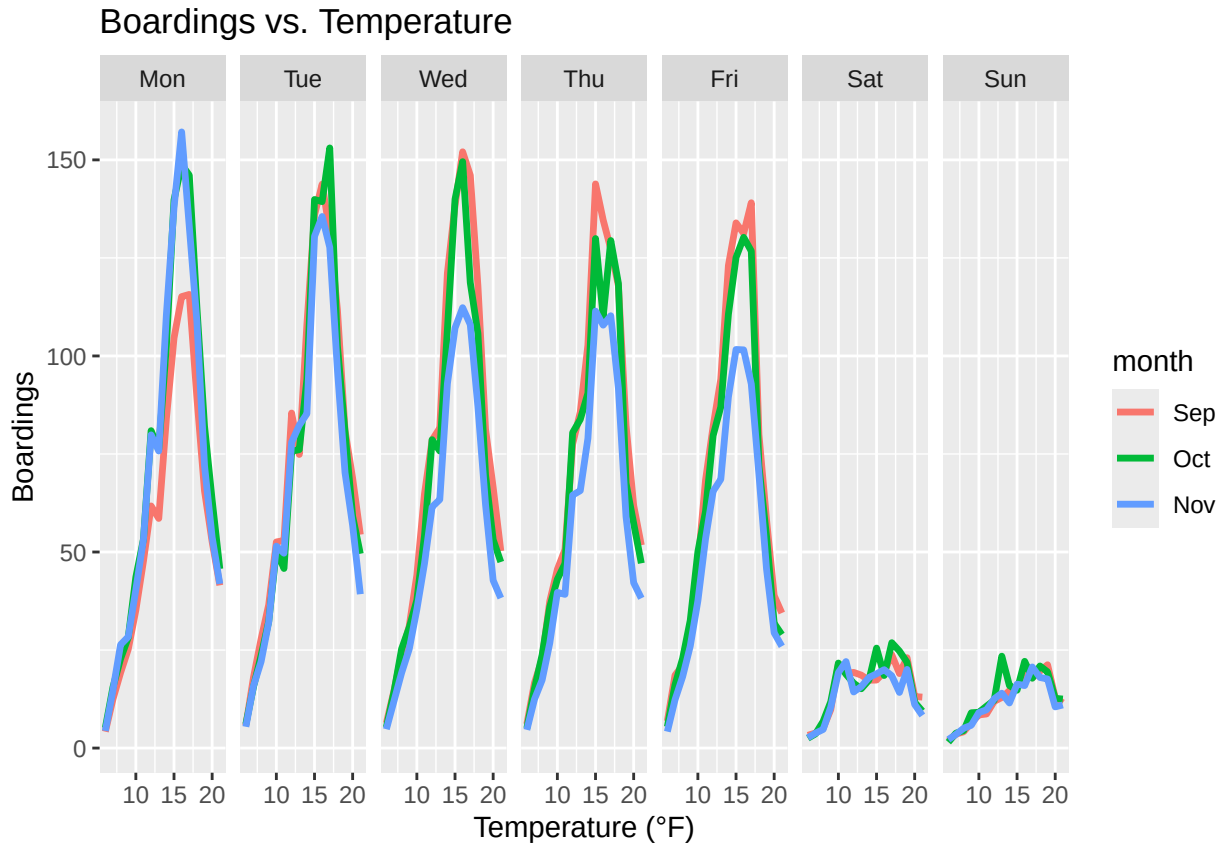
```
#Read data
capmetro_UT <- read.csv("capmetro_UT.csv")

#Reorder variables
capmetro_UT <- 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")))
```

Part 1

```
# Group data
plot1_data <- capmetro_UT %>%
  group_by(hour_of_day, day_of_week, month) %>%
  summarize(avg_boardings = mean(boarding, na.rm = TRUE))
```

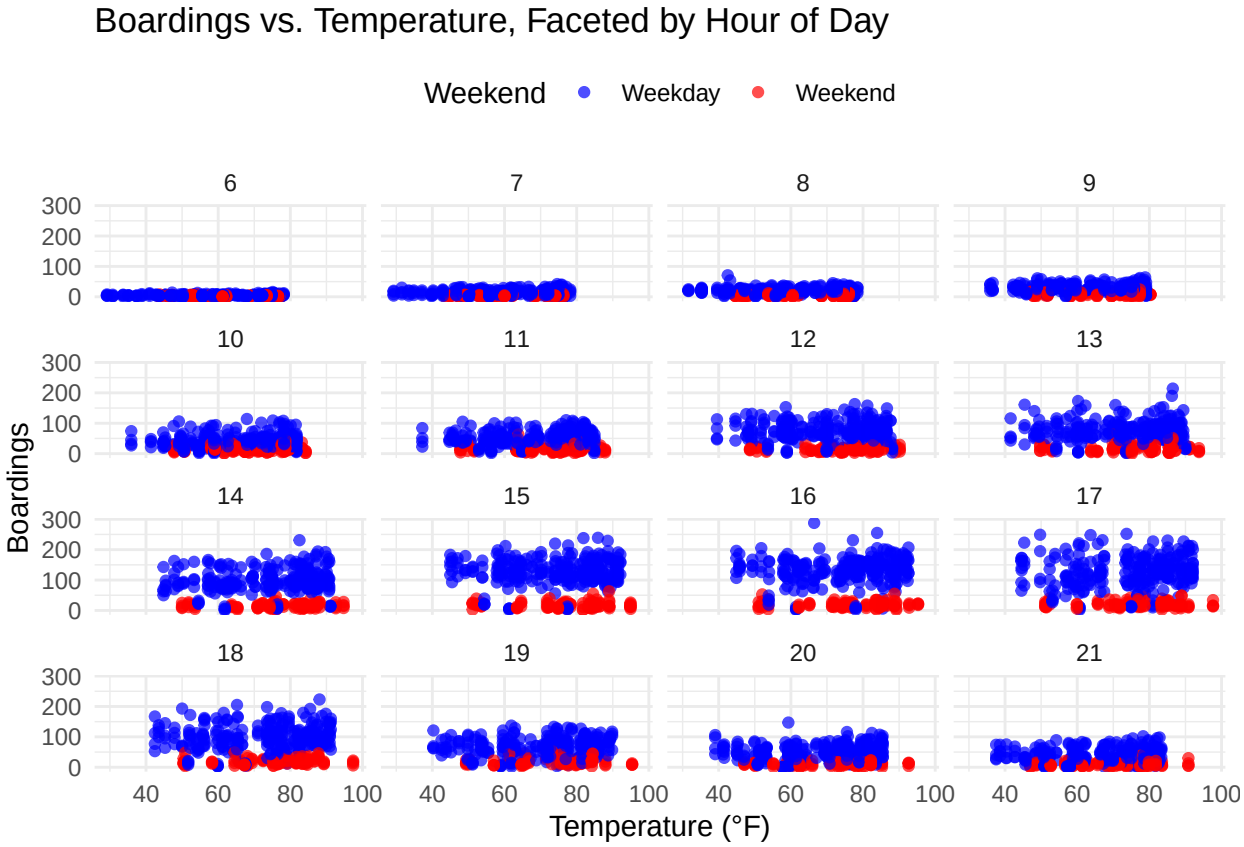
```
## `summarise()` has grouped output by 'hour_of_day', 'day_of_week'. You can
## override using the `.groups` argument.
```

```
#Faceted line graph
ggplot(plot1_data, aes(x = hour_of_day, y = avg_boardings, color = month, group = month)) + geom_line(s
  labs(
    title = "Boardings vs. Temperature",
    x = "Temperature (°F)",
    y = "Boardings"
  )+
  facet_wrap(~day_of_week, nrow = 1)    # Facet by day
```



Part 2

```r
#Faceted scatter plot
ggplot(capmetro_UT, aes(x = temperature, y = boarding, color = factor(weekend))) +
  geom_point(alpha = 0.7) +
  facet_wrap(~hour_of_day, ncol = 4) +
  labs(
    title = "Boardings vs. Temperature, Faceted by Hour of Day",
    x = "Temperature (°F)",
    y = "Boardings",
    color = "Weekend"
  ) +
  scale_color_manual(values = c("blue", "red"), labels = c("Weekday", "Weekend")) +
  theme_minimal() +
  theme(legend.position = "top")
```



**Problem 4: Wrangling the Billboard Top 100*

```r
#Read data
billboard_data <- read.csv("billboard.csv")
```

Part A

```
#Group by performer and song, and calculate total weeks on the Billboard Top 100
song_counts <- billboard_data %>%
  group_by(performer, song) %>%
  summarise(count = n(), .groups = "drop")

#Arrange in descending order and filter the top 10
top_10_songs <- song_counts %>%
  arrange(desc(count)) %>%
  slice(1:10)

#Create table
top_10_songs %>%
  kable(
    col.names = c("Performer", "Song", "Weeks on Chart"),
    caption = "Top 10 Most Popular Songs Since 1958 (Based on Total Weeks on Billboard Top 100)"
  )
```

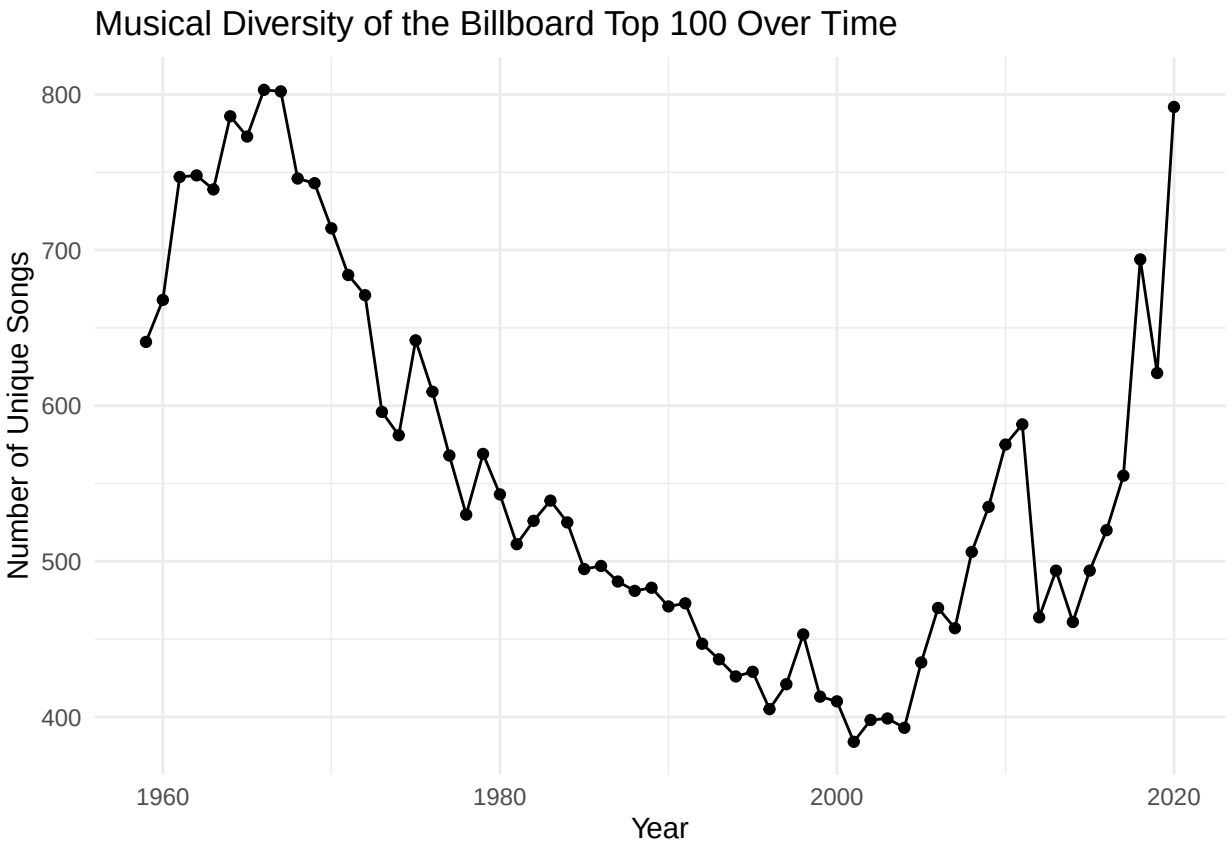
Table 1: Top 10 Most Popular Songs Since 1958 (Based on Total Weeks on Billboard Top 100)

Performer	Song	Weeks on Chart
Imagine Dragons	Radioactive	87
AWOLNATION	Sail	79
Jason Mraz	I'm Yours	76
The Weeknd	Blinding Lights	76
LeAnn Rimes	How Do I Live	69
LMFAO Featuring Lauren Bennett & GoonRock	Party Rock Anthem	68
OneRepublic	Counting Stars	68
Adele	Rolling In The Deep	65
Jewel	Foolish Games/You Were Meant For Me	65
Carrie Underwood	Before He Cheats	64

Part B

```
#Filter data
filtered_data <- billboard_data %>%
  filter(year != 1958, year != 2021)

#Count unique songs per year
unique_songs_per_year <- filtered_data %>%
  group_by(year) %>%
  summarize(unique_songs = n_distinct(song))

#Line graph of musical diversity
ggplot(unique_songs_per_year, aes(x = year, y = unique_songs)) +
  geom_line() +
```

```
  geom_point() +
  labs(title = "Musical Diversity of the Billboard Top 100 Over Time",
       x = "Year",
       y = "Number of Unique Songs"
       ) +
  theme_minimal()
```



Part C

```
#Filter songs
ten_week_hits <- billboard_data %>%
  group_by(performer, song) %>%
  summarise(weeks_on_chart = n_distinct(week)) %>%
  filter(weeks_on_chart >= 10)
```

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

```
#Count ten-week hits for each artist
performer_hits <- ten_week_hits %>%
  group_by(performer) %>%
  summarise(num_ten_week_hits = n()) %>%
```

```r
  filter(num_ten_week_hits >= 30)

#Bar plot
ggplot(performer_hits, aes(x = reorder(performer, num_ten_week_hits), y = num_ten_week_hits)) +
  geom_bar(stat = "identity", fill = "skyblue") +
  coord_flip() +  # Flip the plot to make labels readable
  labs(title = "Number of Ten-Week Hits by performer",
       x = "Artist",
       y = "Number of Ten-Week Hits") +
  theme_minimal()
```