Analysis of My Personal Netflix Viewing History

Ellie Burton

MATH 452 Final Project

I. Introduction

For this project, I analyzed my own Netflix viewing history to understand my consumption habits over the last year. I used R for data cleaning and visualization. I wanted to investigate a few curiosities, such as:

- How much time have I actually spent watching Netflix since I created my account?
- What is my most watched series?
- What is the ratio of TV shows to movies in my diet?

Statistically, I was also interested in seeing if there were patterns in *when* I watched Netflix. I wanted to assess if my "binge-watching" habits followed patterns such as being more likely on weekends or in certain months of the year. The primary objective of this project is to determine if there is a statistically significant relationship between the day of the week or month of the year and the intensity of Netflix consumption. Specifically, the research question is:

> "On days when I actively watch Netflix, does the intensity of my viewing (total minutes) vary significantly by the day of the week (or month of the year)?"

It is important to note the distinction here between frequency and intensity. An analysis of "average daily viewing" would include days with zero activity, conflating the decision to watch at all with the decision of how long to watch. By filtering for only "active" days (with > 0 minutes watched), we are isolating days I already chose to watch Netflix and only assessing the length of watch time. I chose to do this in part because there are periods of time in which I had canceled Netflix, was watching more content on other streaming platforms, or other external circumstances that affected frequency of viewing.

II. Data

A. Data Source

The data for this analysis was obtained directly from Netflix via their "Download Your Information" feature. The dataset, ViewingActivity.csv, contains a comprehensive log of every piece of content viewed on the account profile, including timestamps, duration, and device information. The specific columns used in my analysis were:

- **Profile:** Profile the content was watched on.
- **Start Time:** The UTC timestamp of when content was started.
- **Duration:** How long it was watched.
- **Title:** The name of the show or movie.

Table 1 shows a sample of the data used in the analysis.

Table 1

Sample of Viewing Data

Profile	*Start.Time*	*Duration*	*Title*
Ellie	2025-11-14 21:34:06	01:27:51	Hot Frosty
Ellie	2025-11-12 04:00:13	00:00:03	Grey's Anatomy: Season 4: Piece of My Heart (Episode 13)
Ellie	2025-11-12 03:15:1	00:41:50	Grey's Anatomy: Season 4: Where the Wild Things Are (Episode 12)

B. Data Cleaning

Using R, I performed preprocessing steps to prepare the data for analysis. I filtered out non-informative content such as trailers and promotional clips, as these do not represent true viewing intent. I performed time standardization by converting timestamps from UTC strings to local date-time objects so viewing sessions were assigned to the correct calendar day. I aggregated the data into a new feature, Total_Minutes, which is a sum of all viewing durations for a single calendar date.

III. Preliminary Analysis

A. Date Analysis

In my preliminary analysis, I noticed that there were many months with 0 viewing sessions or very sparse data, as seen in Figure 1 and Table 2 below.

Figure 1

Cumulative Daily Watch Time from 4/3/2025 to 11/14/2025

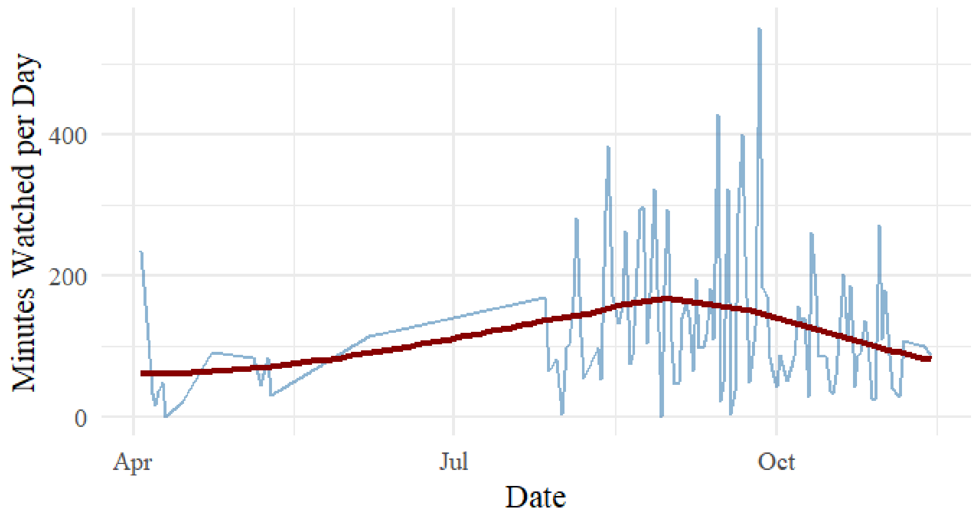


Table 2

Data Entry Count by Month

Month	04	05	06	07	08	09	10	11
Count	23	17	1	14	150	156	109	20

This is due to the aforementioned occurrences of canceling Netflix or watching more content on other streaming platforms during certain time periods. My Netflix data also only begins in April, 2025 when I started the account. Because of these limitations, I decided to focus only on the day of the week research question.

B. “Netflix Wrapped”

I also investigated some of my own curiosities about my viewing habits in a Spotify Wrapped-style analysis. I found that I watched 230.05 hours of Netflix in this time period, which is 9.59 total days of life. I also found my top 5 shows of the year by total watch time in Table 3.

Table 3

Top 5 Series by Total Watch Time

	Show_Name	Hours_Watched
1	Gilmore Girls	94.1
2	Grey's Anatomy	54.4
3	Stranger Things	16.0
4	The Hunting Wives	9.80
5	Temptation Island	9.29

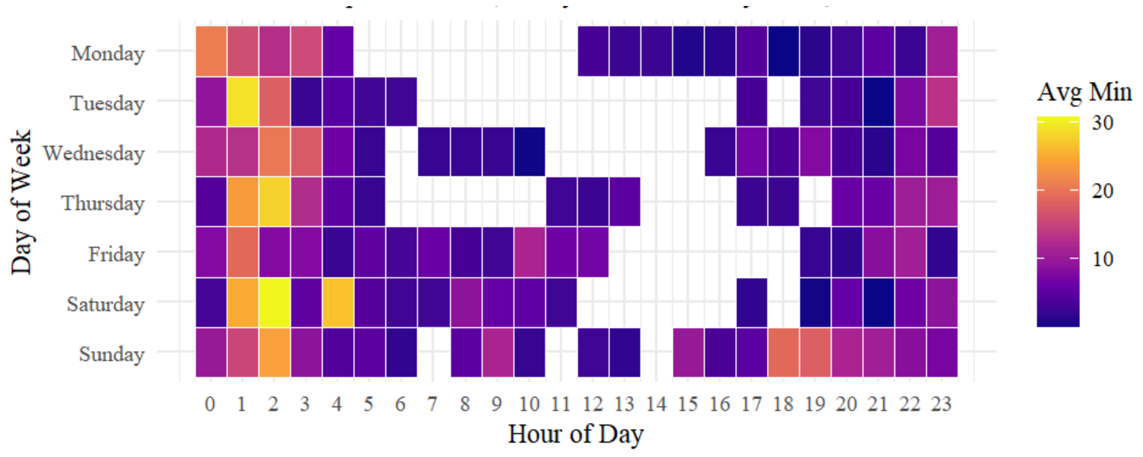
I found that I barely watched any movies (6.3%) compared to TV shows (93.7%). Finally, my highest day of watch time was Friday, September 26th with 9.18 hours! These were interesting insights to see about my own personal data.

C. Visualizations

To visualize patterns in my average watch time, I created a heatmap plotting "Hour of Day" against "Day of Week." The color intensity represents the average minutes watched in that specific slot. One important choice to note is that the data is averaged across active days for each weekday. Thus, it answers, *"On a Monday where I actually watch Netflix, when do I watch it?"*

Figure 2

Heatmap of average watch time per hour by day of week

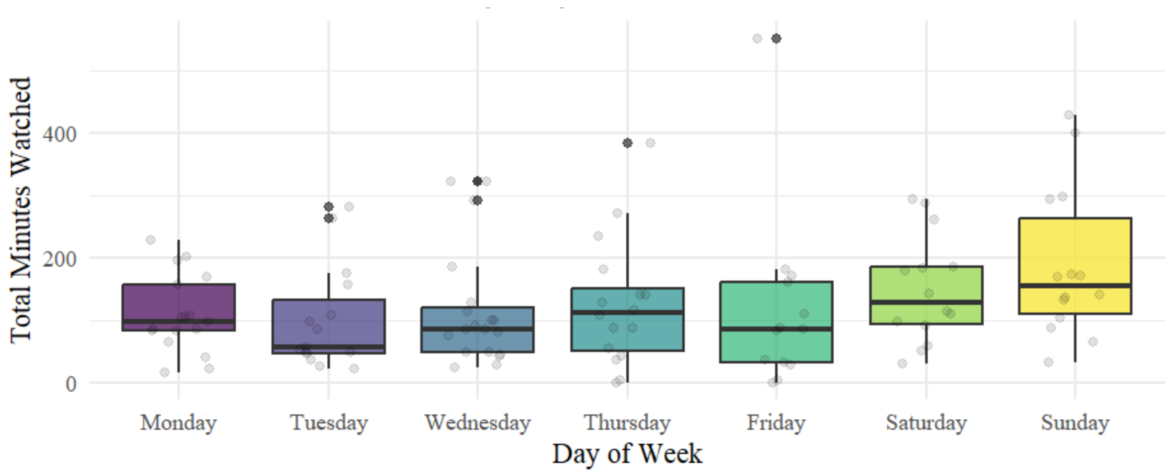


The heatmap in Figure 2 revealed expected patterns, such as large gaps with 0 average minutes watched during “school hours” on weekdays, and slightly more uniform watching on the weekend. There are also higher intensities in the early morning hours (1-3AM), which I believe is due to Netflix autoplaying after I fall asleep.

Next, I generated some plots to get an overall idea of what the viewing data looked like across the days of the week. I first created a boxplot to compare the distribution of total minutes watched across the days of the week, as seen in Figure 3.

Figure 3

Boxplot of minutes watched by day of week

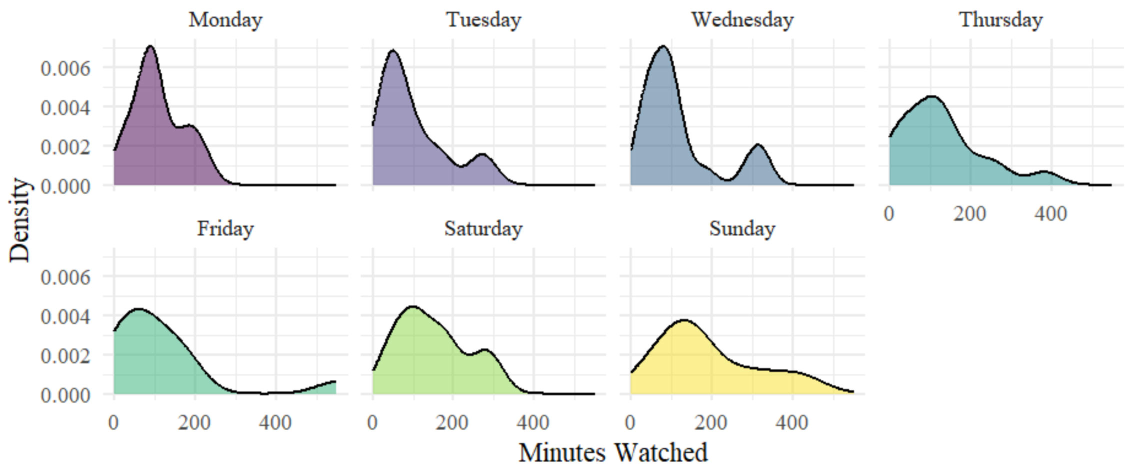


Note: Data includes only "active" days

The spread of data was noticeably larger on weekends (Friday, Saturday, Sunday), indicating highly variable behavior. Weekdays showed tighter interquartile ranges.

Figure 4

Density distribution of minutes watched by day of week



Finally, the density distribution plot in Figure 4 confirmed that the data is right-skewed. The majority of days cluster around 60–90 minutes, with a long tail extending out to more extreme days of 300+ minutes.

IV. Statistical Methodology

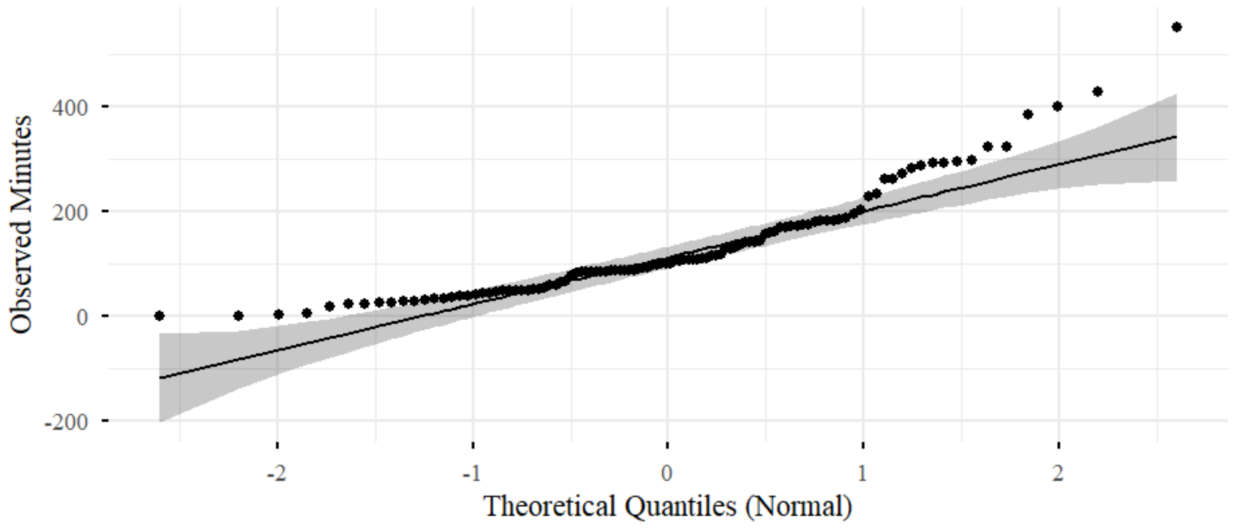
A. Assumption Checking

Before selecting a statistical test to compare my viewing intensity across days of the week, I checked the assumptions required for a standard One-Way ANOVA.

For normality, I performed the Shapiro-Wilk test on the Total_Minutes variable. The test returned a p-value < 0.05 (indicating non-normality), and the Q-Q plot showed significant deviation from the diagonal line. This confirms that my viewing data is not normally distributed. This was visually confirmed with a Q-Q plot, shown below. In the plot, the observed data deviates from the line, confirming non-normality.

Figure 5

Q-Q Plot of Viewing Minutes



Because the assumption of normality was violated and the data contained significant outliers, I decided that a One-Way ANOVA was not the appropriate tool. Instead, I selected the Kruskal-Wallis Rank Sum Test. This is a non-parametric alternative that compares median ranks rather than means and is robust to outliers.

B. Hypotheses

H_0: There is no difference in the median active watch time across the 7 days of the week.

H_a: At least one day of the week has a different median active watch time than the others.

C. Results

The Kruskal-Wallis test yielded the following results:

```
                    Kruskal-Wallis rank sum test

              data:  Total_Minutes by Day_of_Week
Kruskal-Wallis chi-squared = 9.8798, df = 6, p-value = 0.1298
```

Since the p-value (0.1298) is greater than the standard significance level of 0.05, we fail to reject the null hypothesis. This means that, statistically speaking, there is no significant difference in the intensity of my viewing across the days of the week. While the boxplots might show some visual variation, the differences are not strong enough to rule out random chance.

V. Conclusions

A. Summary

This project aimed to quantify my personal streaming habits and determine if I binge-watch more on certain days. The descriptive analysis confirmed that I am a "TV Show Person" (93.7% preference) who favors long-running shows like Gilmore Girls and Grey's Anatomy.

However, the statistical inference provided a surprising insight. Contrary to my assumption that I watch significantly more on weekends, the Kruskal-Wallis test (p = 0.1298) indicates that my viewing intensity is consistent across all days of the week.

This finding is contingent on my methodology of filtering for "Active Days." While my decision to watch might vary (frequency), on a day that I decide to watch Netflix, I tend to watch for a similar duration regardless of the day of week.

B. Limitations

Sample size is a limitation of these results, as my viewing data only begins in April 2025. I would like to redo this analysis in 6 months or so when I have a full year of data. Another limitation is high variance. As noted in the boxplots, the spread of data is very wide. This high variance within each day makes it harder for statistical tests to detect differences between days. Finally, a major limitation of this analysis is the exclusion of other streaming service data. While I attempted to request and download my data from Hulu and Disney Plus as well, the requests were not accepted by the due date of this project.

C. Future Work

If I were to continue this, there are many questions I thought of throughout that would be interesting to investigate. I would love to cross-reference this dataframe with my academic calendar (exam dates) to see if stress correlates with increased viewing (procrastination) or decreased viewing (studying). It would also be fun to package this somehow and allow others to upload their Netflix data and see a visual "Netflix Wrapped."

Appendix

GitHub:

https://github.com/ellie-burton/NetflixWrappedR

R Code:

```r
#===========================================================================
# MATH 452 PROJECT: NETFLIX VIEWING HABITS ANALYSIS
# Ellie Burton
# Description:
# The script first imports and cleans raw Netflix data. It then generates
# exploratory visualizations to illustrate viewing patterns. Finally,
# the code checks assumptions and performs a Kruskal-Wallis rank sum test
# to determine if viewing intensity varies significantly by day of the
week.
#===========================================================================

# --- STEP 1: LOAD LIBRARIES ---
if(!require(tidyverse)) install.packages("tidyverse")
if(!require(lubridate)) install.packages("lubridate")
if(!require(ggpubr)) install.packages("ggpubr") # For nice QQ plots

library(tidyverse)
library(lubridate)
library(ggpubr)

theme_set(theme_minimal(base_family = "serif"))


# --- STEP 2: LOAD AND CLEAN DATA ---
# Ensure 'ViewingActivity.csv' is in your working directory
df <- read_csv("ViewingActivity.csv")

# Clean column names (spaces become dots, e.g., "Start Time" ->
"Start.Time")
colnames(df) <- make.names(colnames(df))

# Initial Filter & Transformation (Session Level)
session_df <- df %>%
  # 1. Remove non-content rows (Hooks, Trailers, etc.)
  filter(!Supplemental.Video.Type %in%
c("HOOK","TRAILER","BONUS_VIDEO","TEASER_TRAILER","TUTORIAL","RECAP")) %>%
  # 2. Time Conversions
  mutate(
    Start_DateTime = ymd_hms(Start.Time),
    Date = as_date(Start_DateTime),
    Duration_Minutes = period_to_seconds(hms(Duration)) / 60
  ) %>%
  # 3. Filter invalid rows
  filter(!is.na(Start_DateTime)) %>%
  # 4. PARSE TITLES
  # Logic: Movies usually have 0 or 1 colon. TV shows usually have 2+
(Show: Season: Episode)
  mutate(
    Type = ifelse(str_count(Title, ":") >= 2, "TV Show", "Movie"),
    # Extract just the Show Name (everything before the first colon)
    Show_Name = str_split_fixed(Title, ":", 3)[,1]
  )

# --- STEP 2.5: CURIOSITY QUESTIONS (Exploratory Analysis) ---
print("--- CURIOSITY CHECK 1: TOTAL WATCH TIME ---")
total_hours <- sum(session_df$Duration_Minutes, na.rm = TRUE) / 60
total_days <- total_hours / 24
cat(sprintf("Total Hours Watched: %.2f\n", total_hours))
cat(sprintf("Total Days of Life Spent on Netflix: %.2f\n", total_days))

print("--- CURIOSITY CHECK 2: TOP 5 BINGED SHOWS ---")
top_shows <- session_df %>%
  filter(Type == "TV Show") %>%
  group_by(Show_Name) %>%
  summarise(Hours_Watched = sum(Duration_Minutes) / 60) %>%
  arrange(desc(Hours_Watched)) %>%
  slice(1:5)
print(top_shows)

print("--- CURIOSITY CHECK 3: MOVIES VS TV SHOWS RATIO ---")
content_ratio <- session_df %>%
  group_by(Type) %>%
  summarise(Total_Minutes = sum(Duration_Minutes)) %>%
  mutate(Percentage = round(Total_Minutes / sum(Total_Minutes) * 100, 1))
print(content_ratio)


# --- STEP 3: AGGREGATION (ACTIVE DAYS ONLY) ---
# Goal: Calculate TOTAL minutes watched per active day.

daily_df <- session_df %>%
  group_by(Date) %>%
  summarise(Total_Minutes = sum(Duration_Minutes)) %>%
  mutate(
    # Re-derive Weekday and Month from the Date
    Day_of_Week = wday(Date, label = TRUE, abbr = FALSE),
    Month = month(Date, label = TRUE, abbr = FALSE),
    # Ensure factor order is correct
    Day_of_Week = factor(Day_of_Week, levels = c("Monday", "Tuesday",
"Wednesday", "Thursday", "Friday", "Saturday", "Sunday"))
  )

# --- STEP 3.5: DATE RANGE CHECK ---
min_date <- min(daily_df$Date)
max_date <- max(daily_df$Date)

print("--- DATA TIMELINE ---")
cat(sprintf("Start Date: %s\n", min_date))
cat(sprintf("End Date:   %s\n", max_date))
cat(sprintf("Total Days Spanned: %s\n", max_date - min_date))

# --- STEP 3.6: MONTHLY ACTIVITY TABLE ---
# Counts how many individual things (rows) were watched per month
print("--- ROW COUNT PER MONTH ---")
monthly_counts <- session_df %>%
  mutate(Month_Year = format(Date, "%Y-%m")) %>%
  group_by(Month_Year) %>%
  summarise(
    Items_Watched = n(),
    Total_Hours = sum(Duration_Minutes) / 60
  ) %>%
  arrange(Month_Year)

print(monthly_counts)

# --- STEP 3.7: RECORD BREAKING DAY ---
# Finds the single day with the absolute highest viewing time
print("--- MOST ACTIVE DAY EVER ---")
record_day <- daily_df %>%
  arrange(desc(Total_Minutes)) %>%
  slice(1) %>%
  mutate(Total_Hours = Total_Minutes / 60)

print(record_day)


# --- STEP 4: VISUALIZATION ---
# Visualization 1: Time Series (Line Plot)
p_timeline <- ggplot(daily_df, aes(x = Date, y = Total_Minutes)) +
  geom_line(color = "steelblue", alpha = 0.6) +
  geom_smooth(method = "loess", color = "darkred", se = FALSE) + # Adds a
trend line
  theme_minimal(base_family = "serif") + # Explicitly ensuring Serif
  labs(
    title = "Timeline of My Netflix Addiction",
    subtitle = paste("Daily Watch Time from", min_date, "to", max_date),
    x = "Date",
    y = "Minutes Watched per Day"
  )
print(p_timeline)

# Visualization 2: Boxplot of Daily Consumption
p1 <- ggplot(daily_df, aes(x = Day_of_Week, y = Total_Minutes, fill =
Day_of_Week)) +
  geom_boxplot(alpha = 0.7) +
  geom_jitter(color="black", alpha=0.1, width=0.2) + # Shows individual
active days
  theme_minimal(base_family = "serif") +
  labs(
    title = "Netflix Binge Intensity by Day of Week",
    subtitle = "Total Minutes Watched (Active Days Only)",
    x = "Day of Week",
    y = "Total Minutes Watched"
  ) +
  theme(legend.position = "none")
print(p1)

# Visualization 3: Density Plot (Shows the shape of the data)
p2 <- ggplot(daily_df, aes(x = Total_Minutes, fill = Day_of_Week)) +
  geom_density(alpha = 0.5) +
  facet_wrap(~Day_of_Week, ncol = 4) +
  theme_minimal(base_family = "serif") +
  labs(
    title = "Distribution of Viewing Minutes by Day",
    x = "Minutes Watched",
    y = "Density"
  ) +
  theme(legend.position = "none")
print(p2)

# Visualization 5: Heatmap (Hour vs Day)
# Normalize by "Active Days" to be consistent with ANOVA

# 1. Count how many ACTIVE days exist for each weekday (and convert to
character for join)
active_day_counts <- daily_df %>%
  group_by(Day_of_Week) %>%
  summarise(n_active_days = n()) %>%
  mutate(Day_of_Week = as.character(Day_of_Week)) # FORCE TO CHARACTER

print("--- Active Days per Weekday ---")
print(active_day_counts)

# 2. Prep Heatmap Data
heatmap_data <- session_df %>%
  mutate(
    Day_of_Week = wday(Start_DateTime, label = TRUE, abbr = FALSE),
    Hour = hour(Start_DateTime)
  ) %>%
  # Force to character immediately to avoid factor mismatch during join
  mutate(Day_of_Week = as.character(Day_of_Week)) %>%

  group_by(Day_of_Week, Hour) %>%
  summarise(Total_Minutes = sum(Duration_Minutes), .groups = 'drop') %>%

  # 3. Join the counts (Now both are characters, so it will work!)
  left_join(active_day_counts, by = "Day_of_Week") %>%

  # 4. Calculate Average
  mutate(Average_Minutes = Total_Minutes / n_active_days) %>%

  # 5. NOW we convert back to Factor with REVERSED levels for plotting
  mutate(Day_of_Week = factor(Day_of_Week,
                              levels = rev(c("Monday", "Tuesday",
"Wednesday", "Thursday", "Friday", "Saturday", "Sunday"))))

# Heatmap of AVERAGE INTENSITY (Active Days Normalized)
```

```
p_heatmap_avg <- ggplot(heatmap_data, aes(x = Hour, y = Day_of_Week, fill =
Average_Minutes)) +
  geom_tile(color = "white") +
  scale_fill_viridis_c(option = "plasma") +
  scale_x_continuous(breaks = 0:23) +
  theme_minimal(base_family = "serif") +
  labs(
    title = "Heatmap: Average Active Intensity",
    subtitle = "Minutes watched per hour slot (On days when I actually
watch)",
    x = "Hour of Day",
    y = "Day of Week",
    fill = "Avg Min"
  )
print(p_heatmap_avg)

# --- STEP 5: ASSUMPTION CHECKING (DIAGNOSTICS) ---
# A. Normality Check (Shapiro-Wilk Test)
if(nrow(daily_df) < 5000) {
  shapiro_res <- shapiro.test(daily_df$Total_Minutes)
  print(paste("Shapiro-Wilk Normality Test p-value:", shapiro_res$p.value))
} else {
  print("Sample too large for Shapiro-Wilk. check Q-Q plot.")
}

# Visualization 4: Q-Q Plot
# ggpubr requires specifying the theme argument differently or adding it as
a layer
p3 <- ggqqplot(daily_df$Total_Minutes,
               title = "Q-Q Plot of Viewing Minutes",
               ylab = "Observed Minutes",
               xlab = "Theoretical Quantiles (Normal)",
               ggtheme = theme_minimal(base_family = "serif")) # Ensuring
Serif here too
print(p3)


# --- STEP 6: STATISTICAL TEST (HYPOTHESIS TESTING) ---
# --- Kruskal-Wallis Test (Non-Parametric) ---
kruskal_res <- kruskal.test(Total_Minutes ~ Day_of_Week, data = daily_df)
print("--- Kruskal-Wallis Results (Non-Parametric) ---")
print(kruskal_res)
```