Project #2: Analysis of Restaurant Transactions on Sales

by: Rodrigo Guerra, Isha Bhasin, Shobha Melukote

1. Introduction

a. Set up

b. Quick description of the dataset(s)

- Our data comes from Rodrigo's family's restaurant, Tenoch. It was collected using Square payment processing.
- The dataset includes transaction data from April 2, 2023 to April 8, 2023, which is a week's worth of data from Sunday to Saturday.
- A unique row represents a single transaction at the restaurant
- The dataset is called: **Tenoch_Transactions**
- The variables contained in the dataset include: Date, Time, Gross Sales, Net Sales, Tax, Tip, Total Collected, Source, Card, Cash, NA Drinks, Beer, % Tip, Fees, Net Total, and Dining Option, to name a few.
- This dataset is interesting to us because we're interested in finding out the different factors that affect how much someone pays for their meal at a restaurant.
- **Rodrigo:** I expect a potential relationship to be that: transactions made on the weekend have higher Net Total than those made during the weekdays.
- **Isha:** I expect that the amount tipped will be higher on orders made "For Here" than any other dining option
- **Shobha:** I expect there to potentially be a trend that the amount tipped is higher on weekends than on weekdays

```
# Import datasets
library(readr)
Tenoch_Transactions <- read_csv("Tenoch Transactions.csv")
#View(Tenoch_Transactions)
```

Always a good idea to take a look at the dataset:

```
# Take a look
head(Tenoch_Transactions)
```

```
## # A tibble: 6 × 26
##   Date     Time     Gross Sal…¹ Net S…² Gift …³ Tax   Tip   Total…⁴ Source Card
##   <chr>    <time>   <chr>       <chr>   <chr>   <chr> <chr> <chr>   <chr>  <chr>
## 1 4/8/2023 20:52:45 $4.75       $4.75   $0.00   $0.33 $0.00 $5.08   Regis… $5.08
## 2 4/8/2023 20:45:06 $0.01       $0.01   $0.00   $0.00 $0.00 $0.01   Regis… $0.00
## 3 4/8/2023 20:43:57 $8.95       $8.95   $0.00   $0.63 $0.00 $9.58   Regis… $9.58
## 4 4/8/2023 20:42:11 $0.01       $0.01   $0.00   $0.00 $0.00 $0.01   Regis… $0.00
## 5 4/8/2023 20:40:26 $2.25       $2.25   $0.00   $0.16 $0.00 $2.41   Regis… $2.41
## 6 4/8/2023 20:35:49 $10.95      $10.95  $0.00   $0.77 $0.00 $11.72  Regis… $11.…
## # … with 16 more variables: `Card Entry Methods` <chr>, Cash <chr>,
## #   `Square Gift Card` <chr>, `NA Drinks` <dbl>, Beer <dbl>, `% Tip` <chr>,
## #   Fees <chr>, `Net Total` <chr>, `Card Brand` <chr>, Description <chr>,
## #   Location <chr>, `Dining Option` <chr>, `Customer ID` <chr>,
## #   `Customer Name` <chr>, `Initial Fees` <chr>, `Cash App` <chr>, and
## #   abbreviated variable names ¹`Gross Sales`, ²`Net Sales`,
## #   ³`Gift Card Sales`, ⁴`Total Collected`
```

Before we start our data analysis, we have to tidy and alter the dataset (specifically the Date and Time columns)

```
# Change the Date column into a date class
Tenoch_Transactions$Date <- as.Date(Tenoch_Transactions$Date, format = "%m/%d/%Y")

# view class of Date
class(Tenoch_Transactions$Date)
```

```
## [1] "Date"
```

```
# view results
head(Tenoch_Transactions$Date)
```

```
## [1] "2023-04-08" "2023-04-08" "2023-04-08" "2023-04-08" "2023-04-08"
## [6] "2023-04-08"
```

```
# Combine the Date and Time columns into one:
Tenoch_Transactions$Date_Time <- paste(Tenoch_Transactions$Date, " T= ", Tenoch_Transact
ions$Time)

# Move Date_Time to the front
Tenoch_Transactions <- Tenoch_Transactions %>%
  select(Date, Time, Date_Time, everything())

Tenoch_Transactions
```

```
## # A tibble: 2,231 × 27
##    Date       Time     Date_Time     Gross…¹ Net S…² Gift …³ Tax   Tip   Total…⁴
##    <date>     <time>   <chr>         <chr>   <chr>   <chr>   <chr> <chr> <chr>
##  1 2023-04-08 20:52:45 2023-04-08  … $4.75   $4.75   $0.00   $0.33 $0.00 $5.08
##  2 2023-04-08 20:45:06 2023-04-08  … $0.01   $0.01   $0.00   $0.00 $0.00 $0.01
##  3 2023-04-08 20:43:57 2023-04-08  … $8.95   $8.95   $0.00   $0.63 $0.00 $9.58
##  4 2023-04-08 20:42:11 2023-04-08  … $0.01   $0.01   $0.00   $0.00 $0.00 $0.01
##  5 2023-04-08 20:40:26 2023-04-08  … $2.25   $2.25   $0.00   $0.16 $0.00 $2.41
##  6 2023-04-08 20:35:49 2023-04-08  … $10.95  $10.95  $0.00   $0.77 $0.00 $11.72
##  7 2023-04-08 20:31:15 2023-04-08  … $13.70  $13.70  $0.00   $0.96 $0.00 $14.66
##  8 2023-04-08 20:29:17 2023-04-08  … $15.00  $15.00  $0.00   $1.05 $2.41 $18.46
##  9 2023-04-08 20:28:30 2023-04-08  … $7.90   $7.90   $0.00   $0.55 $1.00 $9.45
## 10 2023-04-08 20:23:39 2023-04-08  … $13.20  $13.20  $0.00   $0.92 $0.00 $14.12
## # … with 2,221 more rows, 18 more variables: Source <chr>, Card <chr>,
## #   `Card Entry Methods` <chr>, Cash <chr>, `Square Gift Card` <chr>,
## #   `NA Drinks` <dbl>, Beer <dbl>, `% Tip` <chr>, Fees <chr>,
## #   `Net Total` <chr>, `Card Brand` <chr>, Description <chr>, Location <chr>,
## #   `Dining Option` <chr>, `Customer ID` <chr>, `Customer Name` <chr>,
## #   `Initial Fees` <chr>, `Cash App` <chr>, and abbreviated variable names
## #   ¹`Gross Sales`, ²`Net Sales`, ³`Gift Card Sales`, ⁴`Total Collected`
```

```
# Convert character data to POSIXlt date and time
Tenoch_Transactions$Date_Time <- as.POSIXct(Tenoch_Transactions$Date_Time,
                            format="%Y-%m-%d T= %T") #format time

# Format Time as time
Tenoch_Transactions$Time <- format(as.POSIXct(Tenoch_Transactions$Time), "%T")


Tenoch_Transactions
```

```
## # A tibble: 2,231 × 27
##    Date       Time     Date_Time           Gross S…¹ Net S…² Gift …³ Tax   Tip
##    <date>     <chr>    <dttm>              <chr>     <chr>   <chr>   <chr> <chr>
##  1 2023-04-08 20:52:45 2023-04-08 20:52:45 $4.75     $4.75   $0.00   $0.33 $0.00
##  2 2023-04-08 20:45:06 2023-04-08 20:45:06 $0.01     $0.01   $0.00   $0.00 $0.00
##  3 2023-04-08 20:43:57 2023-04-08 20:43:57 $8.95     $8.95   $0.00   $0.63 $0.00
##  4 2023-04-08 20:42:11 2023-04-08 20:42:11 $0.01     $0.01   $0.00   $0.00 $0.00
##  5 2023-04-08 20:40:26 2023-04-08 20:40:26 $2.25     $2.25   $0.00   $0.16 $0.00
##  6 2023-04-08 20:35:49 2023-04-08 20:35:49 $10.95    $10.95  $0.00   $0.77 $0.00
##  7 2023-04-08 20:31:15 2023-04-08 20:31:15 $13.70    $13.70  $0.00   $0.96 $0.00
##  8 2023-04-08 20:29:17 2023-04-08 20:29:17 $15.00    $15.00  $0.00   $1.05 $2.41
##  9 2023-04-08 20:28:30 2023-04-08 20:28:30 $7.90     $7.90   $0.00   $0.55 $1.00
## 10 2023-04-08 20:23:39 2023-04-08 20:23:39 $13.20    $13.20  $0.00   $0.92 $0.00
## # … with 2,221 more rows, 19 more variables: `Total Collected` <chr>,
## #   Source <chr>, Card <chr>, `Card Entry Methods` <chr>, Cash <chr>,
## #   `Square Gift Card` <chr>, `NA Drinks` <dbl>, Beer <dbl>, `% Tip` <chr>,
## #   Fees <chr>, `Net Total` <chr>, `Card Brand` <chr>, Description <chr>,
## #   Location <chr>, `Dining Option` <chr>, `Customer ID` <chr>,
## #   `Customer Name` <chr>, `Initial Fees` <chr>, `Cash App` <chr>, and
## #   abbreviated variable names ¹`Gross Sales`, ²`Net Sales`, …
```

```
# Get the days of the week for each transaction
Tenoch_Transactions$Day <- wday(Tenoch_Transactions$Date_Time, label = TRUE, week_start
= 1)

# Get the numeric days of the week for each transaction
Tenoch_Transactions$Day_Numeric <- wday(Tenoch_Transactions$Date_Time, week_start = 1)


# Move Date_Time to the front
Tenoch_Transactions <- Tenoch_Transactions %>%
  select(Date, Time, Date_Time, Day, everything())


Tenoch_Transactions
```

```
## # A tibble: 2,231 × 29
##    Date       Time     Date_Time           Day   Gross S…¹ Net S…² Gift …³ Tax
##    <date>     <chr>    <dttm>              <ord> <chr>     <chr>   <chr>   <chr>
##  1 2023-04-08 20:52:45 2023-04-08 20:52:45 Sat   $4.75     $4.75   $0.00   $0.33
##  2 2023-04-08 20:45:06 2023-04-08 20:45:06 Sat   $0.01     $0.01   $0.00   $0.00
##  3 2023-04-08 20:43:57 2023-04-08 20:43:57 Sat   $8.95     $8.95   $0.00   $0.63
##  4 2023-04-08 20:42:11 2023-04-08 20:42:11 Sat   $0.01     $0.01   $0.00   $0.00
##  5 2023-04-08 20:40:26 2023-04-08 20:40:26 Sat   $2.25     $2.25   $0.00   $0.16
##  6 2023-04-08 20:35:49 2023-04-08 20:35:49 Sat   $10.95    $10.95  $0.00   $0.77
##  7 2023-04-08 20:31:15 2023-04-08 20:31:15 Sat   $13.70    $13.70  $0.00   $0.96
##  8 2023-04-08 20:29:17 2023-04-08 20:29:17 Sat   $15.00    $15.00  $0.00   $1.05
##  9 2023-04-08 20:28:30 2023-04-08 20:28:30 Sat   $7.90     $7.90   $0.00   $0.55
## 10 2023-04-08 20:23:39 2023-04-08 20:23:39 Sat   $13.20    $13.20  $0.00   $0.92
## # … with 2,221 more rows, 21 more variables: Tip <chr>,
## #   `Total Collected` <chr>, Source <chr>, Card <chr>,
## #   `Card Entry Methods` <chr>, Cash <chr>, `Square Gift Card` <chr>,
## #   `NA Drinks` <dbl>, Beer <dbl>, `% Tip` <chr>, Fees <chr>,
## #   `Net Total` <chr>, `Card Brand` <chr>, Description <chr>, Location <chr>,
## #   `Dining Option` <chr>, `Customer ID` <chr>, `Customer Name` <chr>,
## #   `Initial Fees` <chr>, `Cash App` <chr>, Day_Numeric <dbl>, and …
```

```
# Separate the transactions between Morning, Afternoon, and Evening
Tenoch_Transactions <-Tenoch_Transactions %>%
  mutate(time_of_day = case_when(hour(Date_Time) >= '11:00:00' & hour(Date_Time) < '17:0
0:00' ~ "Afternoon",
                                 hour(Date_Time) >= '17:00:00' & hour(Date_Time) < '23:0
0:00' ~ "Evening",
                                 .default='Morning'))

Tenoch_Transactions
```

```
## # A tibble: 2,231 × 30
##    Date       Time     Date_Time           Day   Gross S…¹ Net S…² Gift …³ Tax
##    <date>     <chr>    <dttm>              <ord> <chr>     <chr>   <chr>   <chr>
##  1 2023-04-08 20:52:45 2023-04-08 20:52:45 Sat   $4.75     $4.75   $0.00   $0.33
##  2 2023-04-08 20:45:06 2023-04-08 20:45:06 Sat   $0.01     $0.01   $0.00   $0.00
##  3 2023-04-08 20:43:57 2023-04-08 20:43:57 Sat   $8.95     $8.95   $0.00   $0.63
##  4 2023-04-08 20:42:11 2023-04-08 20:42:11 Sat   $0.01     $0.01   $0.00   $0.00
##  5 2023-04-08 20:40:26 2023-04-08 20:40:26 Sat   $2.25     $2.25   $0.00   $0.16
##  6 2023-04-08 20:35:49 2023-04-08 20:35:49 Sat   $10.95    $10.95  $0.00   $0.77
##  7 2023-04-08 20:31:15 2023-04-08 20:31:15 Sat   $13.70    $13.70  $0.00   $0.96
##  8 2023-04-08 20:29:17 2023-04-08 20:29:17 Sat   $15.00    $15.00  $0.00   $1.05
##  9 2023-04-08 20:28:30 2023-04-08 20:28:30 Sat   $7.90     $7.90   $0.00   $0.55
## 10 2023-04-08 20:23:39 2023-04-08 20:23:39 Sat   $13.20    $13.20  $0.00   $0.92
## # … with 2,221 more rows, 22 more variables: Tip <chr>,
## #   `Total Collected` <chr>, Source <chr>, Card <chr>,
## #   `Card Entry Methods` <chr>, Cash <chr>, `Square Gift Card` <chr>,
## #   `NA Drinks` <dbl>, Beer <dbl>, `% Tip` <chr>, Fees <chr>,
## #   `Net Total` <chr>, `Card Brand` <chr>, Description <chr>, Location <chr>,
## #   `Dining Option` <chr>, `Customer ID` <chr>, `Customer Name` <chr>,
## #   `Initial Fees` <chr>, `Cash App` <chr>, Day_Numeric <dbl>, …
```

```
# Move Date and Time columns to the front
Tenoch_Transactions <- Tenoch_Transactions %>%
  select(Date, Time, Date_Time, Day, time_of_day, everything())
```

```
# Make the same time_of_day variable, but numeric
Tenoch_Transactions <-Tenoch_Transactions %>%
  mutate(time_of_day_num = case_when(hour(Date_Time) >= '11:00:00'
                                     & hour(Date_Time) < '17:00:00' ~ 2,
                                   hour(Date_Time) >= '17:00:00'
                                   & hour(Date_Time) < '23:00:00' ~ 3,
                                   .default=1))

Tenoch_Transactions
```

```
## # A tibble: 2,231 × 31
##    Date       Time     Date_Time           Day   time_…¹ Gross…² Net S…³ Gift …⁴
##    <date>     <chr>    <dttm>              <ord> <chr>   <chr>   <chr>   <chr>
##  1 2023-04-08 20:52:45 2023-04-08 20:52:45 Sat   Evening $4.75   $4.75   $0.00
##  2 2023-04-08 20:45:06 2023-04-08 20:45:06 Sat   Evening $0.01   $0.01   $0.00
##  3 2023-04-08 20:43:57 2023-04-08 20:43:57 Sat   Evening $8.95   $8.95   $0.00
##  4 2023-04-08 20:42:11 2023-04-08 20:42:11 Sat   Evening $0.01   $0.01   $0.00
##  5 2023-04-08 20:40:26 2023-04-08 20:40:26 Sat   Evening $2.25   $2.25   $0.00
##  6 2023-04-08 20:35:49 2023-04-08 20:35:49 Sat   Evening $10.95  $10.95  $0.00
##  7 2023-04-08 20:31:15 2023-04-08 20:31:15 Sat   Evening $13.70  $13.70  $0.00
##  8 2023-04-08 20:29:17 2023-04-08 20:29:17 Sat   Evening $15.00  $15.00  $0.00
##  9 2023-04-08 20:28:30 2023-04-08 20:28:30 Sat   Evening $7.90   $7.90   $0.00
## 10 2023-04-08 20:23:39 2023-04-08 20:23:39 Sat   Evening $13.20  $13.20  $0.00
## # … with 2,221 more rows, 23 more variables: Tax <chr>, Tip <chr>,
## #   `Total Collected` <chr>, Source <chr>, Card <chr>,
## #   `Card Entry Methods` <chr>, Cash <chr>, `Square Gift Card` <chr>,
## #   `NA Drinks` <dbl>, Beer <dbl>, `% Tip` <chr>, Fees <chr>,
## #   `Net Total` <chr>, `Card Brand` <chr>, Description <chr>, Location <chr>,
## #   `Dining Option` <chr>, `Customer ID` <chr>, `Customer Name` <chr>,
## #   `Initial Fees` <chr>, `Cash App` <chr>, Day_Numeric <dbl>, …
```

```
# Move Date and Time columns to the front
Tenoch_Transactions <- Tenoch_Transactions %>%
  select(Date, Time, Date_Time, Day, time_of_day, everything())
```

```
# Drop Dollar Signs
Tenoch_Transactions$`Gross Sales` = as.numeric(gsub("\\$", "", Tenoch_Transactions$`Gros
s Sales`))

Tenoch_Transactions$`Net Sales` = as.numeric(gsub("\\$", "", Tenoch_Transactions$`Net Sa
les`))

Tenoch_Transactions$`Gift Card Sales` = as.numeric(gsub("\\$", "", Tenoch_Transactions$`
Gift Card Sales`))

Tenoch_Transactions$Tax = as.numeric(gsub("\\$", "", Tenoch_Transactions$Tax))

Tenoch_Transactions$Tip = as.numeric(gsub("\\$", "", Tenoch_Transactions$Tip))

Tenoch_Transactions$`Total Collected` = as.numeric(gsub("\\$", "", Tenoch_Transactions$`
Total Collected`))

Tenoch_Transactions$Card = as.numeric(gsub("\\$", "", Tenoch_Transactions$Card))

Tenoch_Transactions$Cash = as.numeric(gsub("\\$", "", Tenoch_Transactions$Cash))

Tenoch_Transactions$`Square Gift Card` = as.numeric(gsub("\\$", "", Tenoch_Transactions
$`Square Gift Card`))

Tenoch_Transactions$`Net Total` = as.numeric(gsub("\\$", "", Tenoch_Transactions$`Net To
tal`))

Tenoch_Transactions$`Cash App` = as.numeric(gsub("\\$", "", Tenoch_Transactions$`Cash Ap
p`))

#view(Tenoch_Transactions)
```

```
# Drop Parentheses on Fees and Initial Fees
Tenoch_Transactions$Fees = gsub("[()]", "", Tenoch_Transactions$Fees)
Tenoch_Transactions$`Initial Fees` = gsub("[()]", "", Tenoch_Transactions$`Initial Fees
`)

# Change % Tip column
Tenoch_Transactions$`% Tip` <- as.numeric(sub("%", "",Tenoch_Transactions$`% Tip`,fixed=
TRUE))/100


#view(Tenoch_Transactions)
```

```
# Drop Dollar Signs
Tenoch_Transactions$Fees = as.numeric(gsub("\\$", "", Tenoch_Transactions$Fees))

Tenoch_Transactions$`Initial Fees` = as.numeric(gsub("\\$", "", Tenoch_Transactions$`Ini
tial Fees`))

#view(Tenoch_Transactions)
```

```
# Remove NA Values
#Tenoch_Transactions <- Tenoch_Transactions %>%
  #na.omit()

#view(Tenoch_Transactions)
```

Now our data is tidy and everything looks good, so we can proceed to the next step.

c. Define a research question

Something we want to explore is the different factors that affect the amount of money paid in a transaction.

Research Question:

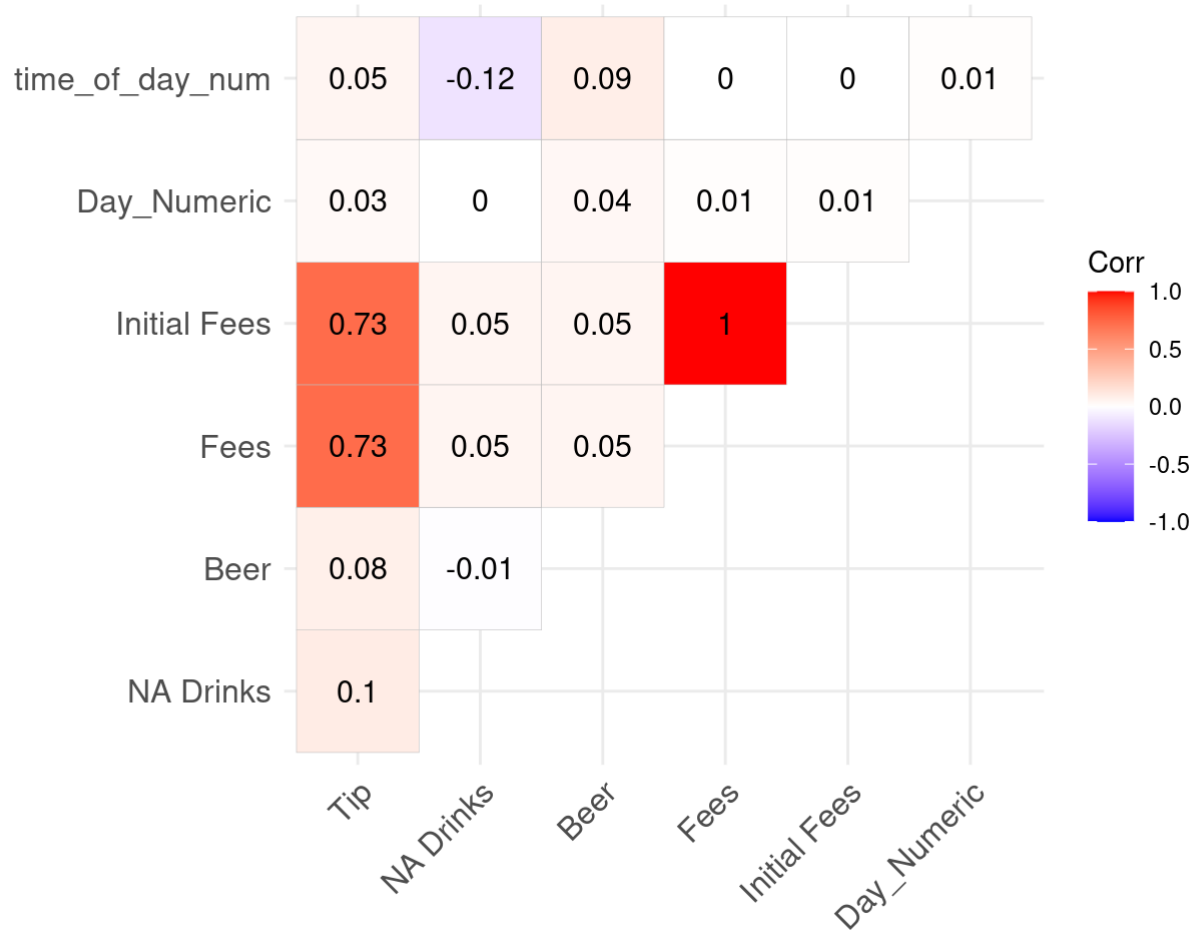
- **Rodrigo:** How does the day of the week of a transaction affect the average Net Total?
- **Isha:** Does dining option affect the tip amount?
- **Shobha:** Does the amount tipped per transaction depend on the day of the week?

2. Exploratory Data Analysis

Create a correlation matrix:

```
## Get all the numeric variables in our dataset
Tenoch_Numeric <- Tenoch_Transactions %>%
    select(where(is.numeric))


# Source:
# https://stackoverflow.com/questions/5863097/selecting-only-numeric-columns-from-a-data
-frame

# Remove irrelevant variables
Tenoch_Numeric <- Tenoch_Numeric %>%
  select(-`Gift Card Sales`, -`Square Gift Card`, - `Cash App`)

#Tenoch_Numeric
```

```
## Use the ggcorrplot to visualize the correlation matrix
ggcorrplot(cor(Tenoch_Numeric),
           type = "upper", # upper diagonal
           lab = TRUE) # print values
```



We can use the correlation coefficient to describe the strength and direction of the relationship between pair of variables.

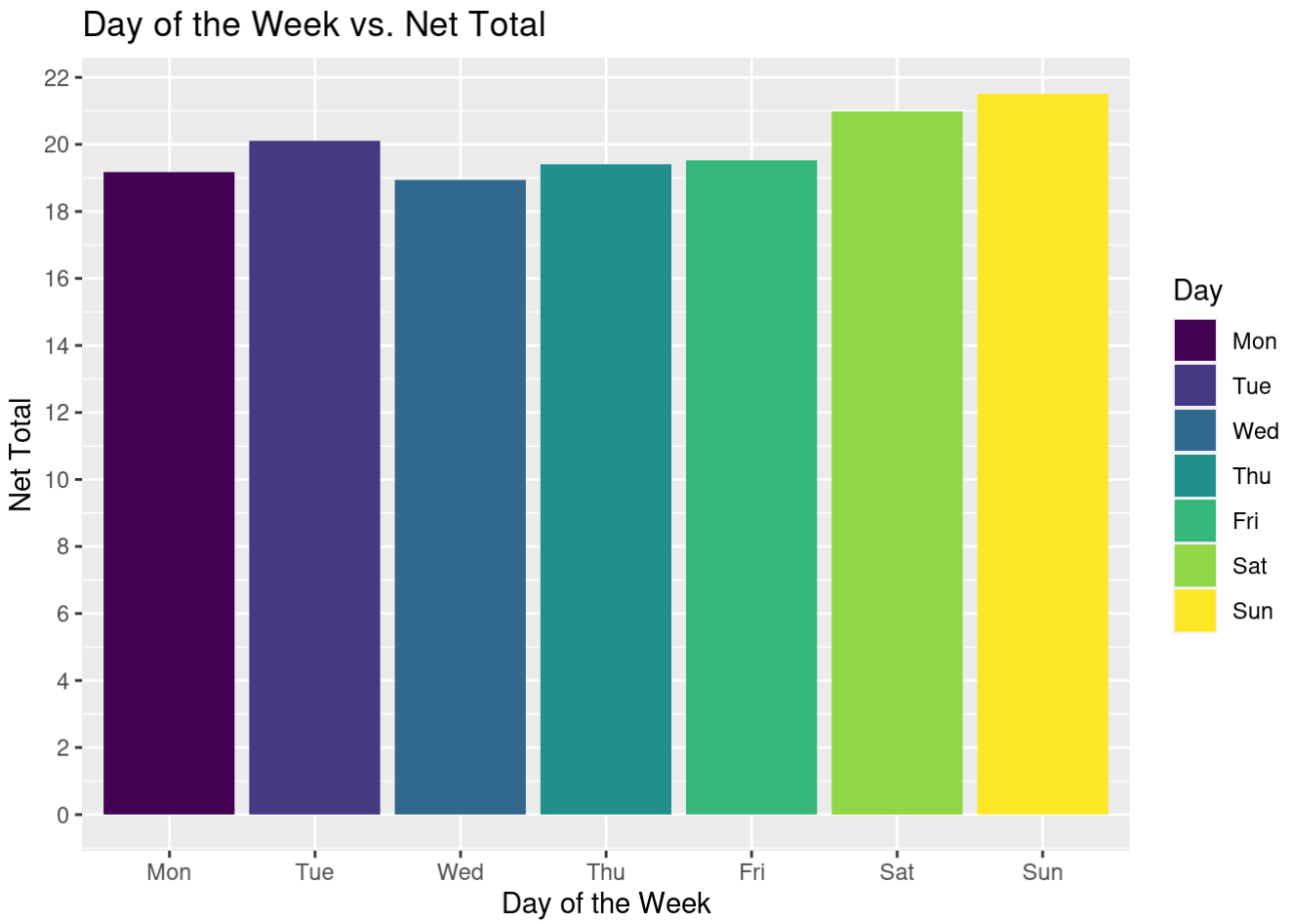
The **variables that are the most correlated are**: Initial Fees and Tip; Fees and Tip; and Initial Fees and Fees.

The **variables that are the least correlated are**: Initial Fees and NA Drinks; Fee and NA Drinks; Initial Fees and Beer; Fees and Beer; and Beer and NA Drinks

Create visualizations to investigate relationships:

Rodrigo's visualizations:

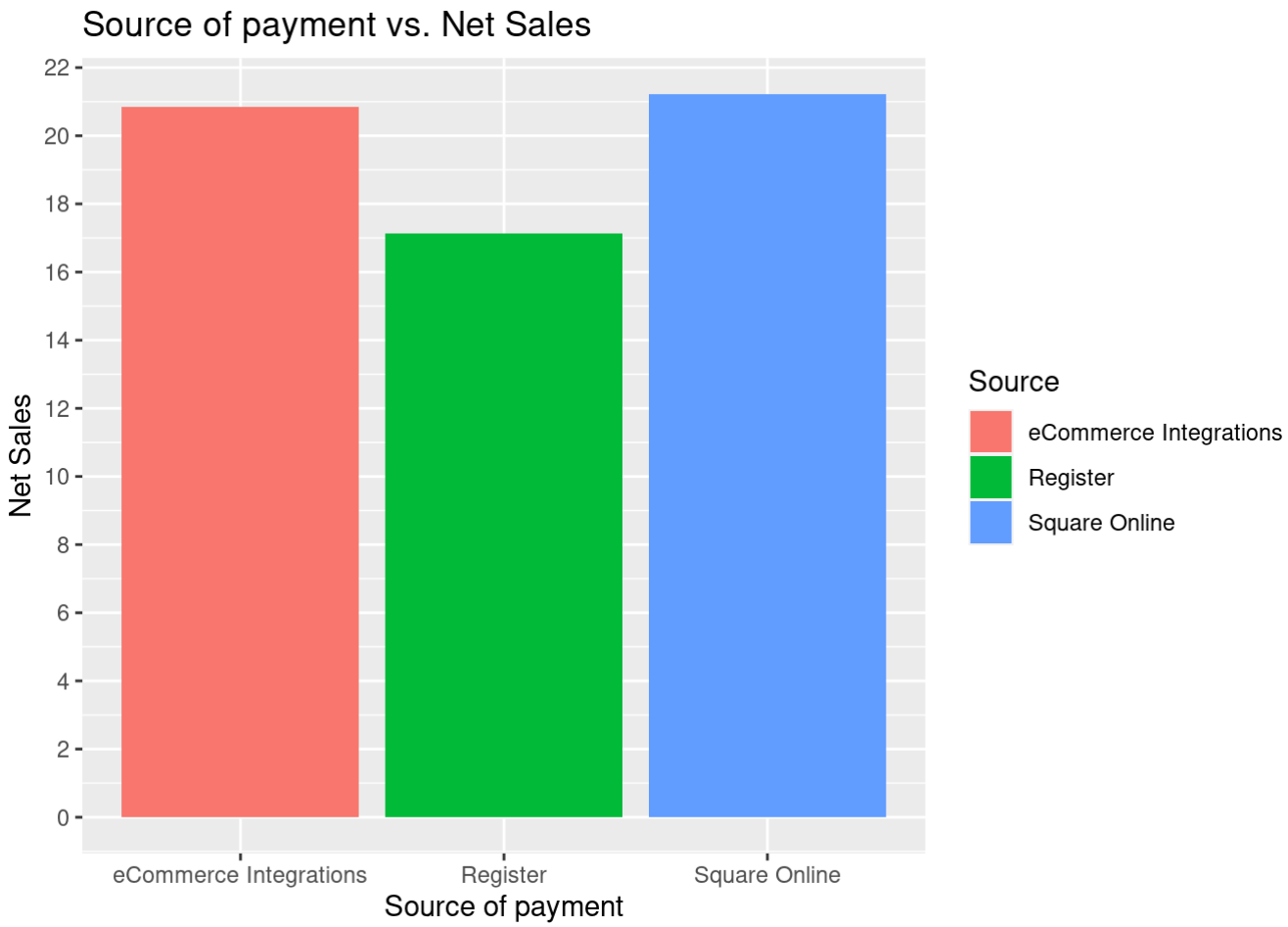
```
# Visualization 1: This plot shows the relationship between the day of the week and the
average Net Total collected
Tenoch_Transactions %>%
  group_by(Day) %>%

  # Create a bar graph showing the day of the week vs. the net total
  ggplot(aes(y = `Net Total`, x = Day, fill = Day)) +
  geom_bar(stat = "summary", fun = "mean") + # use a stat function

  scale_y_continuous(breaks = seq(0, 30, 2)) +
  labs(title = "Day of the Week vs. Net Total",   # adjust labels
       x = "Day of the Week", y = "Net Total")
```



This plot shows the relationship between the day of the week and the averagte Net Total collected per transaction. We can see from the plot that the average total amount of money collected from the restaurant is the lowest on Wednesday, and the average total amount of money collected is the highest on Saturday and Sunday.

This agrees with what I predicted!

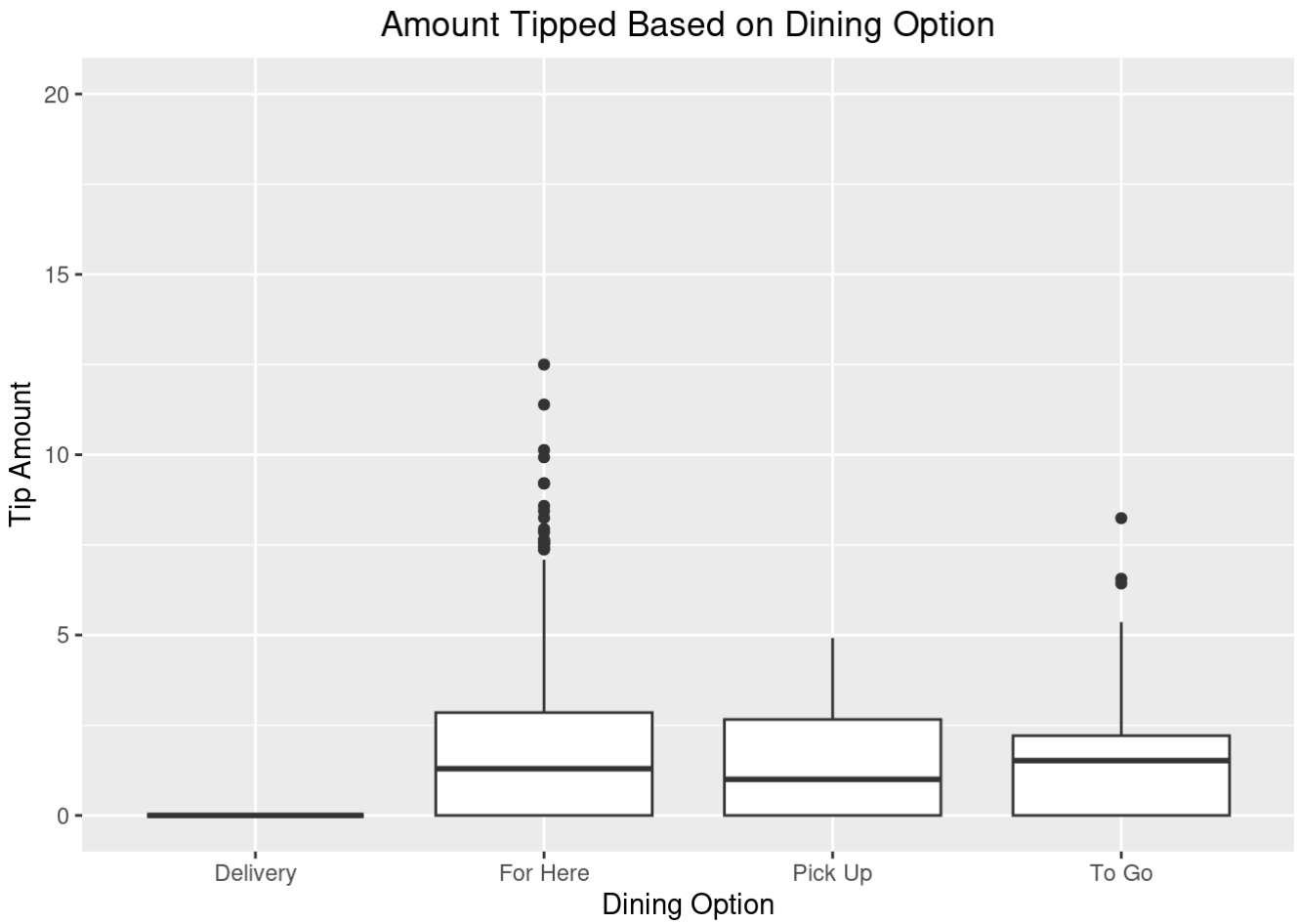
```
# Visualization 2: This plot shows the relationship between the source of payment and th
e Net Sales
Tenoch_Transactions %>%
  group_by(Source) %>%
  filter(Source != "Invoices", Source != "Point of Sale") %>% # filter out Invoices and
Point of Sale because they're not relevant or significant

  # Create a bar graph showing the day of the week vs. the net total
  ggplot(aes(y = `Net Sales`, x = Source, fill = Source)) +
  geom_bar(stat = "summary", fun = "mean") + # use a stat function

  scale_y_continuous(breaks = seq(0, 30, 2)) +
  labs(title = "Source of payment vs. Net Sales",   # adjust labels
       x = "Source of payment", y = "Net Sales")
```



This plot shows the relationship between the source of payment and the average Net Sales per transaction. We can see from the plot that the average net sales from the restaurant is the highest when people pay online through Square and eCommerce (Uber Eats, Grubhub, etc.), and the average net sales is the lowest when people pay at the register.

Isha's visualizations:

```
# Visualization 1: This plot shows the relationship between distribution of tip based on
dining option

#Remove NA values
Tenoch_Transactions_1 <- Tenoch_Transactions %>%
  na.omit()

Tenoch_Transactions_1 %>%
ggplot(aes(x = `Dining Option`, y = Tip)) +
  geom_boxplot() +
  ylim(0, 20) +
  labs(x = "Dining Option", y = "Tip Amount") +
  ggtitle("Amount Tipped Based on Dining Option") +
  theme(plot.title = element_text(hjust = 0.5))
```



Box Plot Explanation -

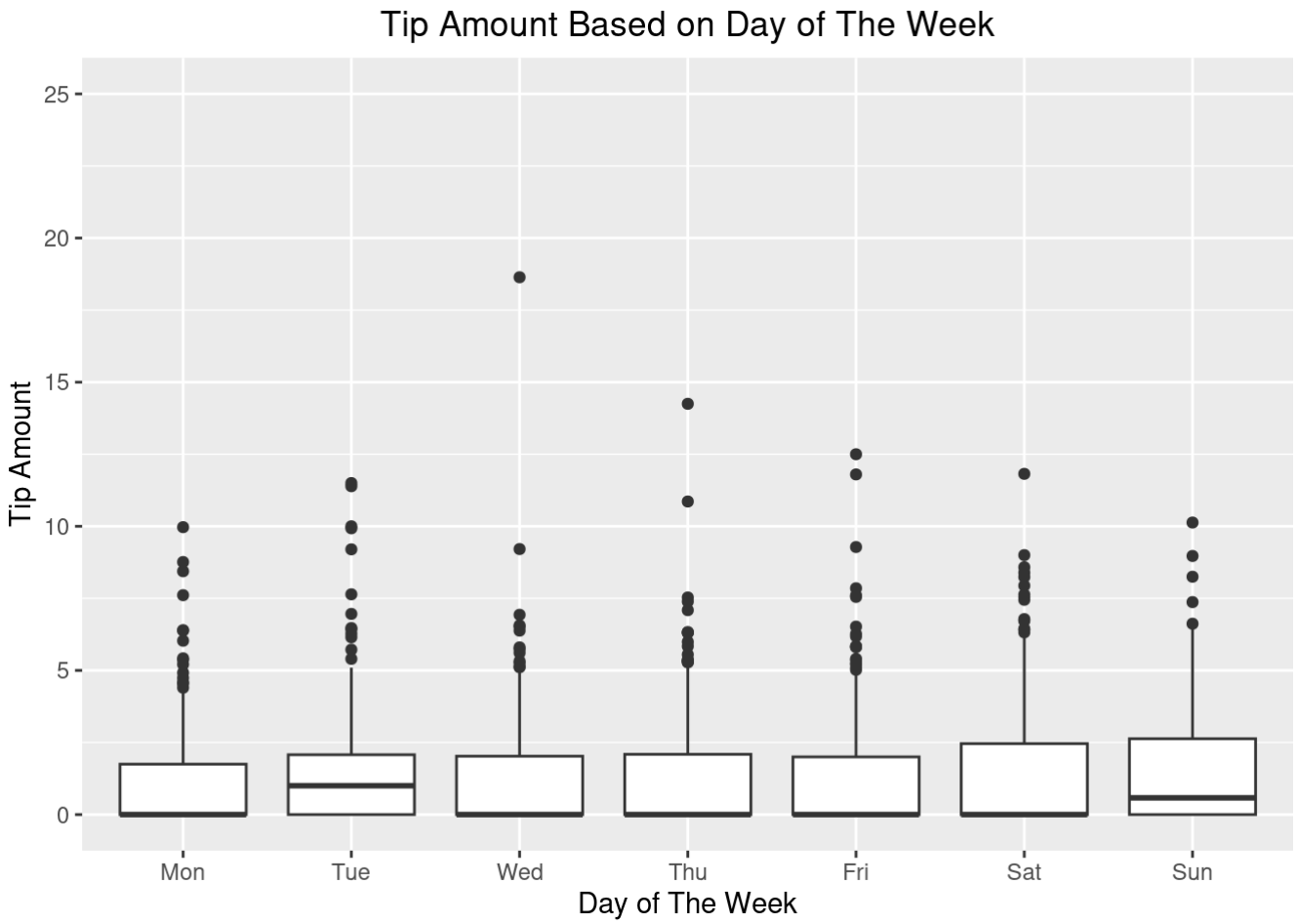
This first plot is a box plot depicting the amount tipped based on dining option. The four dining options are Delivery, For Here, Pick Up, and To Go. Based on this box plot, the median amount tipped is the highest amongst people who place orders “For Here” or “To Go”. The tip amount for Delivery is zero, and the tip amount for “Pick Up” orders has a lower median than the other two dining options. Moreover, there are significant outliers for tip amount in the “For Here” option, with some orders getting up to 12.5% in tip. This may be due to some customers receiving exceptional service when dining in.

```
# Visualization 2: This plot shows the relationship between distribution of total and ti
p based on dining option
Tenoch_Transactions_1 %>%
ggplot(aes(x = `Total Collected`, y = Tip, color = as.factor(`Dining Option`))) +
  geom_point() +
  xlim(0, 150) +
  ylim(0, 20) +
  labs(x = "Total Collected", y = "Tip Amount", title = "Total Collected and Tip Amount
By Dining Option", color = "Dining Option") +
  theme(plot.title = element_text(hjust = 0.5))
```

This plot is a scatterplot that represents that total amount collected and the tip amount colored by dining option. Based on this graph, there seems to be a positive correlation between the total collected and tip amount. As the total amount collected increases, the tip amount also increases. It was interesting to see that only the For Here and To Go orders had data visible on this graph, which means that there probably isn't a lot of data available for the other dining options. Moreover, this graph shows that some people did not pay any tip amount despite the total amount collected being high (over 100). These outliers can probably be attributed to customers who did not receive good service, so they did not tip any amount. Besides these outliers, the trend of tip amount increasing with an increasing total amount collected is pretty consistent.

Shobha's visualizations:

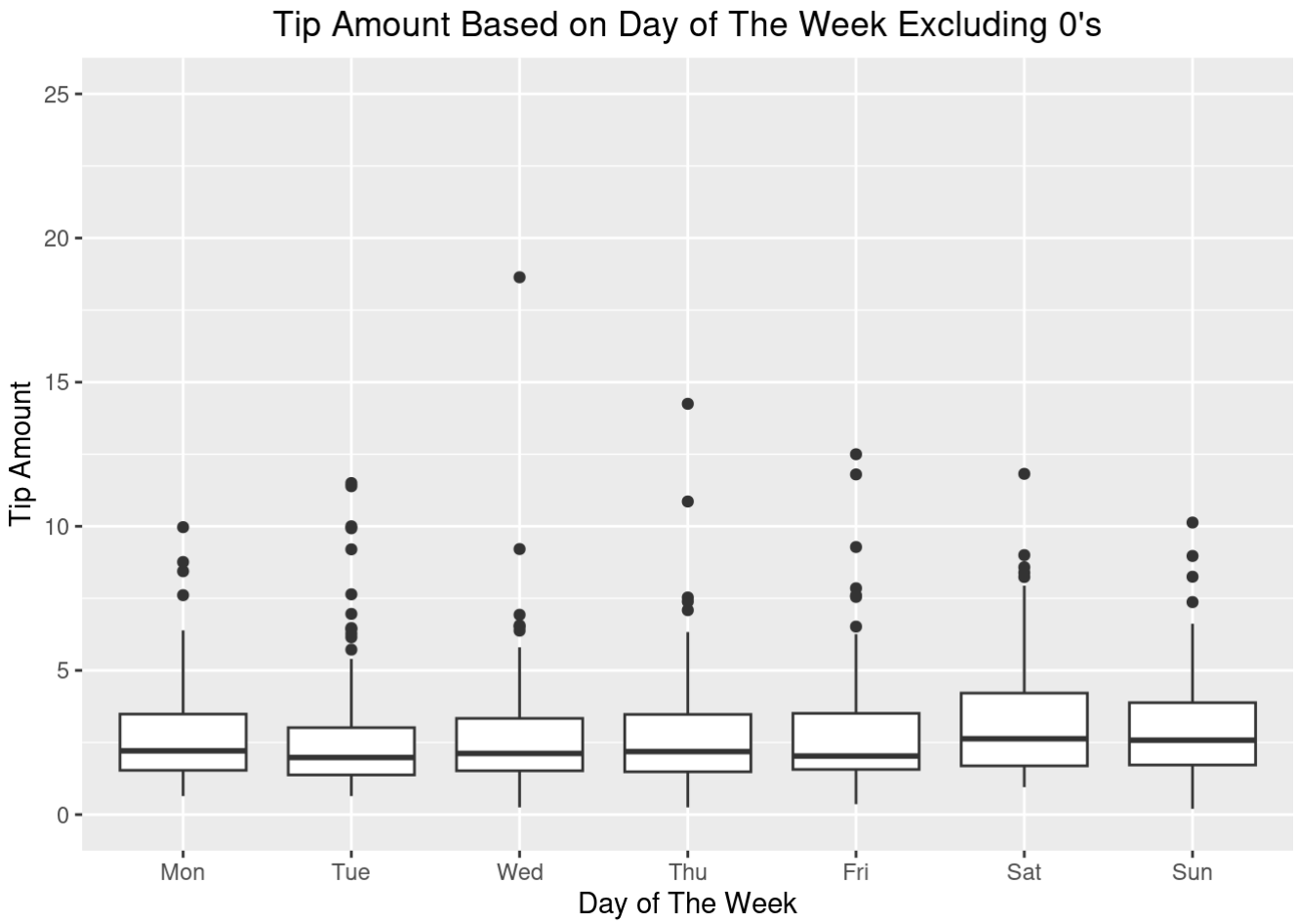
```
# Visualization 1: This plot shows the relationship between distribution based and Day
ggplot(Tenoch_Transactions, aes(x = Day, y = Tip)) +
  geom_boxplot() +
  ylim(0, 25) +
  labs(x = "Day of The Week", y = "Tip Amount") +
  ggtitle("Tip Amount Based on Day of The Week") +
  theme(plot.title = element_text(hjust = 0.5))
```




```
#Create new dataset without 0 values for tip
Tenoch_Transactions_0 <- Tenoch_Transactions %>%
  filter(Tip != 0)
```

This box plot represents the tip amount on different days of the week. Tuesday has the highest median and Sunday has the second highest. However, the medians of the other days of the week are all 0. This somewhat skews the results, so the 0's could be removed to give a more accurate visualization of the distribution of existing tip values for those days.

```
# Visualization 2: This plot shows the relationship between tip distribution based on da
y excluding tip amounts of 0
ggplot(Tenoch_Transactions_0, aes(x = Day, y = Tip)) +
  geom_boxplot() +
  ylim(0, 25) +
  labs(x = "Day of The Week", y = "Tip Amount") +
  ggtitle("Tip Amount Based on Day of The Week Excluding 0's") +
  theme(plot.title = element_text(hjust = 0.5))
```



This box plot represents the tip amount on different days of the week excluding tip amounts of $0. There isn't a huge amount of variation between the different days of the week, but Saturday has the highest median tip amount, closely followed by Sunday and Monday. Since Saturday and Sunday seem to have the highest median tip amounts, it can be concluded that the median tip amount is higher on weekends than it is on weekdays.

3. Prediction and Cross-Validation

a. Train the model

Rodrigo's model: Linear

```
## Use the entire dataset to fit one of these models
# Linear Regression: lm(outcome ~ predictor1 + predictor2, data = fulldata)

Tenoch_Transactions <- Tenoch_Transactions %>%
  mutate(Day_Bin =  if_else(Day == 'Sat' | Day == 'Sun', 1, 0), family = "binomial")


# Fit the model
fit_lin <- lm(Day_Bin ~ `Net Total` + time_of_day_num, data = Tenoch_Transactions)

# Take a look at the model summary
summary(fit_lin)
```

```
##
## Call:
## lm(formula = Day_Bin ~ `Net Total` + time_of_day_num, data = Tenoch_Transactions)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -0.7971 -0.2752 -0.2644  0.6929  0.7587
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)      0.2611908  0.0367820   7.101 1.66e-12 ***
## `Net Total`      0.0014474  0.0006017   2.405   0.0162 *
##  [ reached getOption("max.print") -- omitted 1 row ]
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.4464 on 2223 degrees of freedom
##   (5 observations deleted due to missingness)
## Multiple R-squared:  0.002668,   Adjusted R-squared:  0.001771
## F-statistic: 2.973 on 2 and 2223 DF,  p-value: 0.05133
```

```
Tenoch_Transactions
```

```
## # A tibble: 2,231 × 33
##    Date       Time     Date_Time           Day   time_…¹ Gross…² Net S…³ Gift …⁴
##    <date>     <chr>    <dttm>              <ord> <chr>     <dbl>   <dbl>   <dbl>
##  1 2023-04-08 20:52:45 2023-04-08 20:52:45 Sat   Evening    4.75    4.75       0
##  2 2023-04-08 20:45:06 2023-04-08 20:45:06 Sat   Evening    0.01    0.01       0
##  3 2023-04-08 20:43:57 2023-04-08 20:43:57 Sat   Evening    8.95    8.95       0
##  4 2023-04-08 20:42:11 2023-04-08 20:42:11 Sat   Evening    0.01    0.01       0
##  5 2023-04-08 20:40:26 2023-04-08 20:40:26 Sat   Evening    2.25    2.25       0
##  6 2023-04-08 20:35:49 2023-04-08 20:35:49 Sat   Evening   11.0    11.0        0
##  7 2023-04-08 20:31:15 2023-04-08 20:31:15 Sat   Evening   13.7    13.7        0
##  8 2023-04-08 20:29:17 2023-04-08 20:29:17 Sat   Evening   15      15          0
##  9 2023-04-08 20:28:30 2023-04-08 20:28:30 Sat   Evening    7.9     7.9        0
## 10 2023-04-08 20:23:39 2023-04-08 20:23:39 Sat   Evening   13.2    13.2        0
## # … with 2,221 more rows, 25 more variables: Tax <dbl>, Tip <dbl>,
## #   `Total Collected` <dbl>, Source <chr>, Card <dbl>,
## #   `Card Entry Methods` <chr>, Cash <dbl>, `Square Gift Card` <dbl>,
## #   `NA Drinks` <dbl>, Beer <dbl>, `% Tip` <dbl>, Fees <dbl>,
## #   `Net Total` <dbl>, `Card Brand` <chr>, Description <chr>, Location <chr>,
## #   `Dining Option` <chr>, `Customer ID` <chr>, `Customer Name` <chr>,
## #   `Initial Fees` <dbl>, `Cash App` <dbl>, Day_Numeric <dbl>, …
```

```
# Logistic Regression: lm(outcome ~ predictor1 + predictor2, data = fulldata, family =
"binomial")
# k-nearest-neighbor: knn3(outcome ~ predictor1 + predictor2, data = fulldata, k = 5)
# Decision tree: rpart(outcome ~ predictor1 + predictor2, data = fulldata, method = "cla
ss")
```

```
# Calculate RMSE of regression model
sqrt(mean(resid(fit_lin)^2))
```

```
## [1] 0.4461102
```

```
# Calculate R^2 of regression model
summary(fit_lin)$r.squared
```

```
## [1] 0.002667971
```

- **The value of the RMSE is 0.4460072**. This is quite low, meaning that the model fits the dataset very well.
- The value of R^2, which reports the percentage of variation in the response variable that can be explained by the predictor variables, **is 0.002505424**; this is quite low, meaning that the model doesn't fit the dataset very well.
- This values contradict each other, so this may not be the best model to use.

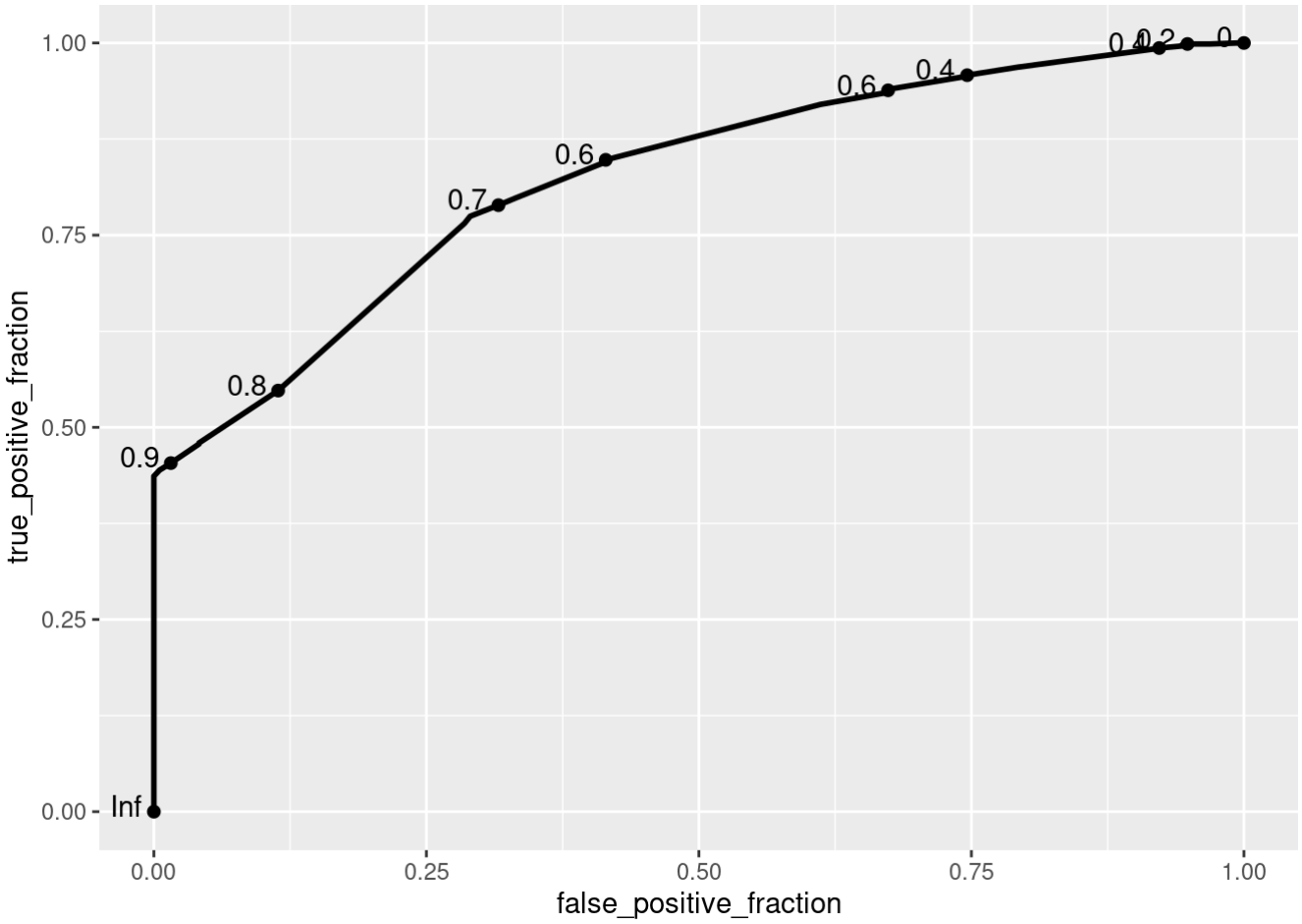
Isha's model:

```
## Use the entire dataset to fit one of these models
# k-nearest-neighbor: knn3(outcome ~ predictor1 + predictor2, data = fulldata, k = 5)
# Fit the model
#Omit NA values + make binary response variable
Tenoch_Transactions_NA <- Tenoch_Transactions %>%
  na.omit()
TT_B <- Tenoch_Transactions_NA %>%
  mutate(`Dining Option` = ifelse(`Dining Option` == "For Here", 1, 0))

# Make Valid Column Names
colnames(TT_B) <- make.names(colnames(TT_B))

#Create KNN dataset
TT_kNN <- knn3(Dining.Option ~ Total.Collected + Tip,
               data = TT_B,
               k = 5)

# Consider the decision tree classifier
TT_tree <- rpart(Dining.Option ~ Total.Collected + Tip, # model
                 data = TT_B, # data
                 method = "class") # classification

#Make predictions
predict(TT_tree, TT_B) %>%
  as.data.frame %>%
  head
```

```
##           0         1
## 1 0.2018828 0.7981172
## 2 0.2018828 0.7981172
## 3 0.2018828 0.7981172
## 4 0.2018828 0.7981172
## 5 0.2018828 0.7981172
##  [ reached 'max' / getOption("max.print") -- omitted 1 rows ]
```

Options for finding AUC is building a ROC curve:

```
#KNN ROC curve
ROC <- ggplot(TT_B) +
  geom_roc(aes(d = Dining.Option, m = predict(TT_kNN, TT_B)[,2]), n.cuts = 10)

ROC
```




```
# Value of AUC for KNN model
calc_auc(ROC)
```

```
##   PANEL group       AUC
## 1     1    -1 0.8233079
```

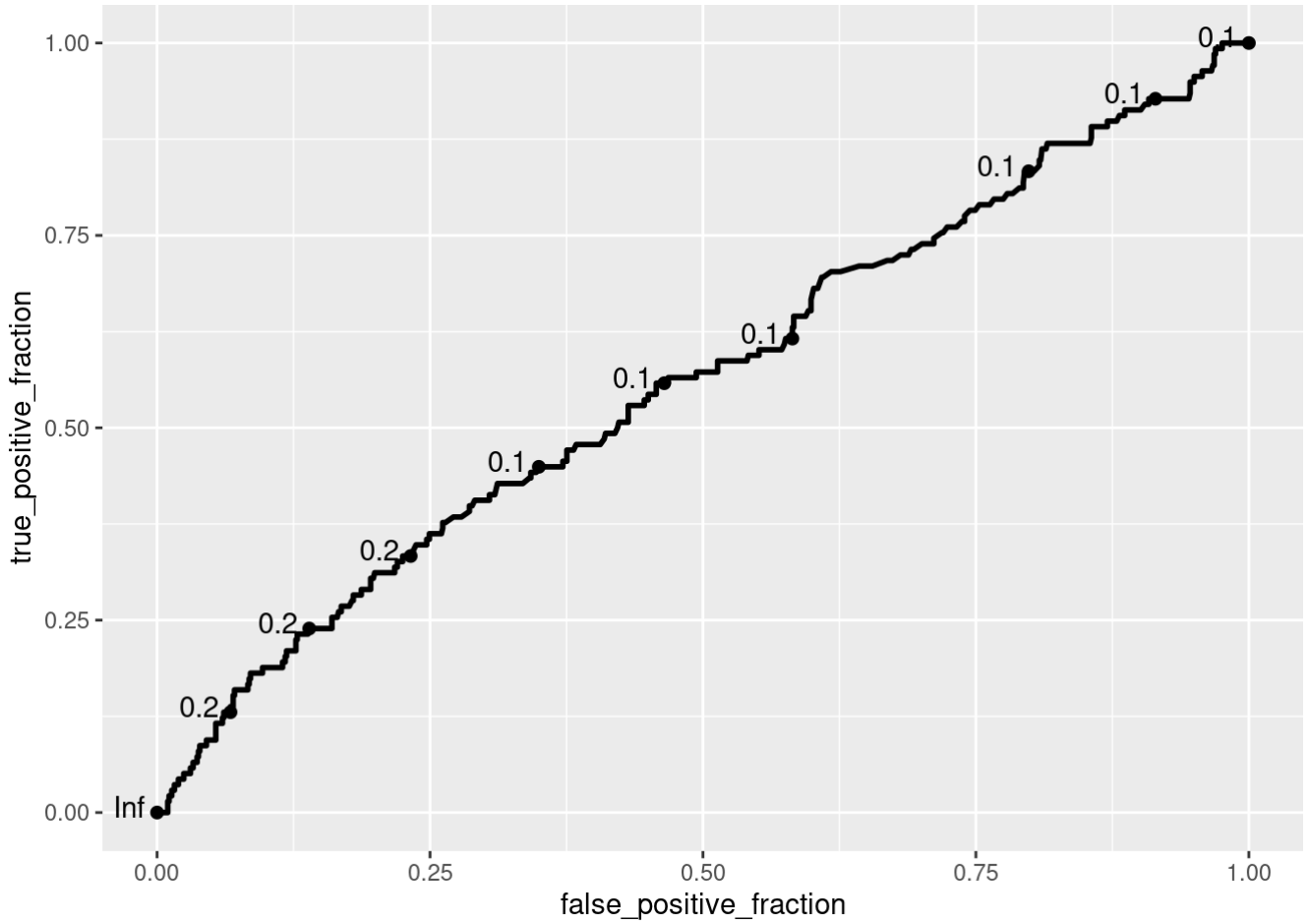
The AUC value is 0.8233079. This is pretty high, which means that the classifier seems to perform fairly well and makes good predictions.

Shobha's model:

```
# Logistic Regression: lm(outcome ~ predictor1 + predictor2, data = fulldata, family =
"binomial")
# Fit the model

# Create log regression dataset
TT_B1 <- Tenoch_Transactions_NA %>%
  mutate(Day = ifelse(Day == "Sat", 1, 0))

# Fit log regression model
TT_log <- glm(Day ~ `Total Collected` + Tip, data = TT_B1, family = "binomial")

# Take a look at the model summary
summary(TT_log)
```

```
##
## Call:
## glm(formula = Day ~ `Total Collected` + Tip, family = "binomial",
##     data = TT_B1)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.4616  -0.5672  -0.5416  -0.5329   2.0329
##
## Coefficients:
##                    Estimate Std. Error z value Pr(>|z|)
## (Intercept)       -1.833393   0.146103 -12.549   <2e-16 ***
## `Total Collected` -0.002240   0.008223  -0.272    0.785
##  [ reached getOption("max.print") -- omitted 1 row ]
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 789.24  on 955  degrees of freedom
## Residual deviance: 786.75  on 953  degrees of freedom
## AIC: 792.75
##
## Number of Fisher Scoring iterations: 4
```

```
#ROC for log regression
ROC <- TT_B1 %>%
  # Make predictions
  mutate(predictions = predict(TT_log, type = "response")) %>%
  ggplot() +
  geom_roc(aes(d = Day, m = predictions), n.cuts = 10)

ROC
```




```
# Value of AUC for log regression model
calc_auc(ROC)
```

```
##   PANEL group       AUC
## 1     1    -1 0.5607172
```

The AUC value is 0.5607172. This is neither high nor low, which means that the classifier seems to have mediocre performance and doesn't make great predictions.

b. Perform cross-validation

Rodrigo's model: Linear cross-validation

```
# Take Non-nmeric variables out of dataset
#Tenoch_Transactions_3 <- Tenoch_Transactions %>%
  #select(-)
```

```
# Choose number of folds:
k = 10

# Randomly order rows in the dataset:
data <- Tenoch_Transactions[sample(nrow(Tenoch_Transactions)), ]

# Create k folds:
folds <- cut(seq(1:nrow(data)), breaks = k, labels = FALSE)

# Initialize a vector to keep track of the performance:
perf_k <- NULL

# Use a for loop to get performance for each test set:
for(i in 1:k){
  # Create train and test sets
    train_not_i <- data[folds != i, ]
    test_i <- data[folds == i, ]

  # Train model on train set (all but fold i)
transactions_log <- fit_lin

  # Test model on test set (fold i)
 predict_i <- data.frame(
    predictions = predict(transactions_log, newdata = test_i, type = "response"),
    Day_Bin = test_i$Day_Bin)

  # Consider the ROC curve for the test dataset
    ROC <- ggplot(predict_i) + geom_roc(aes(d = Day_Bin, m = predictions))


  # Get performance for fold i (AUC)
    perf_k[i] <- calc_auc(ROC)$AUC

}

#Average performance:
mean(perf_k)
```

```
## [1] 0.5445452
```

Our average performance is **0.5473923**. This means that the predictive outcomes are accurate around 54.74% of the time.

How well does your classifier predict new observations? Are there any potential signs of overfitting?

✓ Perform 5 or 10-fold cross-validation with the same model using the train()function. Report the average performance of the model across your k folds (if a group project, each group members performs cross-validation for their model).

✓ Discuss the results in a paragraph. How well does your classifier predict new observations? Are there any potential signs of overfitting? (if a group project, compare the performance of the different models).

Isha's model: kNN cross-validation

```
# This didn't work for us, returned an error and we could't figure out the solution
# Choose number of folds:
#k = 10

# Randomly order rows in the dataset:
#data <- TT_B[sample(nrow(TT_B)), ]

# Create k folds:
#folds <- cut(seq(1:nrow(data)), breaks = k, labels = FALSE)

# Initialize a vector to keep track of the performance:
#perf_k <- NULL

# Use a for loop to get performance for each test set:
#for(i in 1:k){
  # Create train and test sets
 #   train_not_i <- data[folds != i, ]
  #  test_i <- data[folds == i, ]

  # Train model on train set (all but fold i)
#kNN_train <- TT_tree

  # Test model on test set (fold i)
 #predict_i <- data.frame(
  #  predictions = predict(kNN_train, newdata = test_i, type = "response"),
   # `Dining.Option` = test_i$`Dining.Option`)


  # Consider the ROC curve for the test dataset
    #ROC <- ggplot(predict_i) + geom_roc(aes(d = `Dining.Option`, m = predictions))


  # Get performance for fold i (AUC)
  #  perf_k[i] <- calc_auc(ROC)$AUC

#}

#Average performance:
#mean(perf_k)
```

Shobha's model: Logistic cross-validation

```
# Choose number of folds:
k = 10

# Randomly order rows in the dataset:
data <- TT_B1[sample(nrow(TT_B1)), ]

# Create k folds:
folds <- cut(seq(1:nrow(data)), breaks = k, labels = FALSE)

# Initialize a vector to keep track of the performance:
perf_k <- NULL

# Use a for loop to get performance for each test set:
for(i in 1:k){
  # Create train and test sets
    train_not_i <- data[folds != i, ]
    test_i <- data[folds == i, ]

  # Train model on train set (all but fold i)
transactions_log <- TT_log

  # Test model on test set (fold i)
 predict_i <- data.frame(
    predictions = predict(transactions_log, newdata = test_i, type = "response"),
    Day = test_i$Day)

  # Consider the ROC curve for the test dataset
    ROC <- ggplot(predict_i) + geom_roc(aes(d = Day, m = predictions))


  # Get performance for fold i (AUC)
    perf_k[i] <- calc_auc(ROC)$AUC

}

#Average performance:
mean(perf_k)
```

```
## [1] 0.5596274
```

Our average performance is **0.5607172**. This means that the predictive outcomes are accurate around 56.07% of the time.

Overall, the linear model is fairly accurate, although the RMSE and R^2 are contradictory. The kNN model is the best all-around, with an AUC value of 0.8233079; which is pretty high,

meaning that the classifier seems to perform fairly well and makes good predictions. The logistic regression model is in the middle, with average performance all round.

4 & 5. Dimensionality reduction and Clustering

Let's investigate some patterns in your dataset: are the observations "naturally" forming different groups?

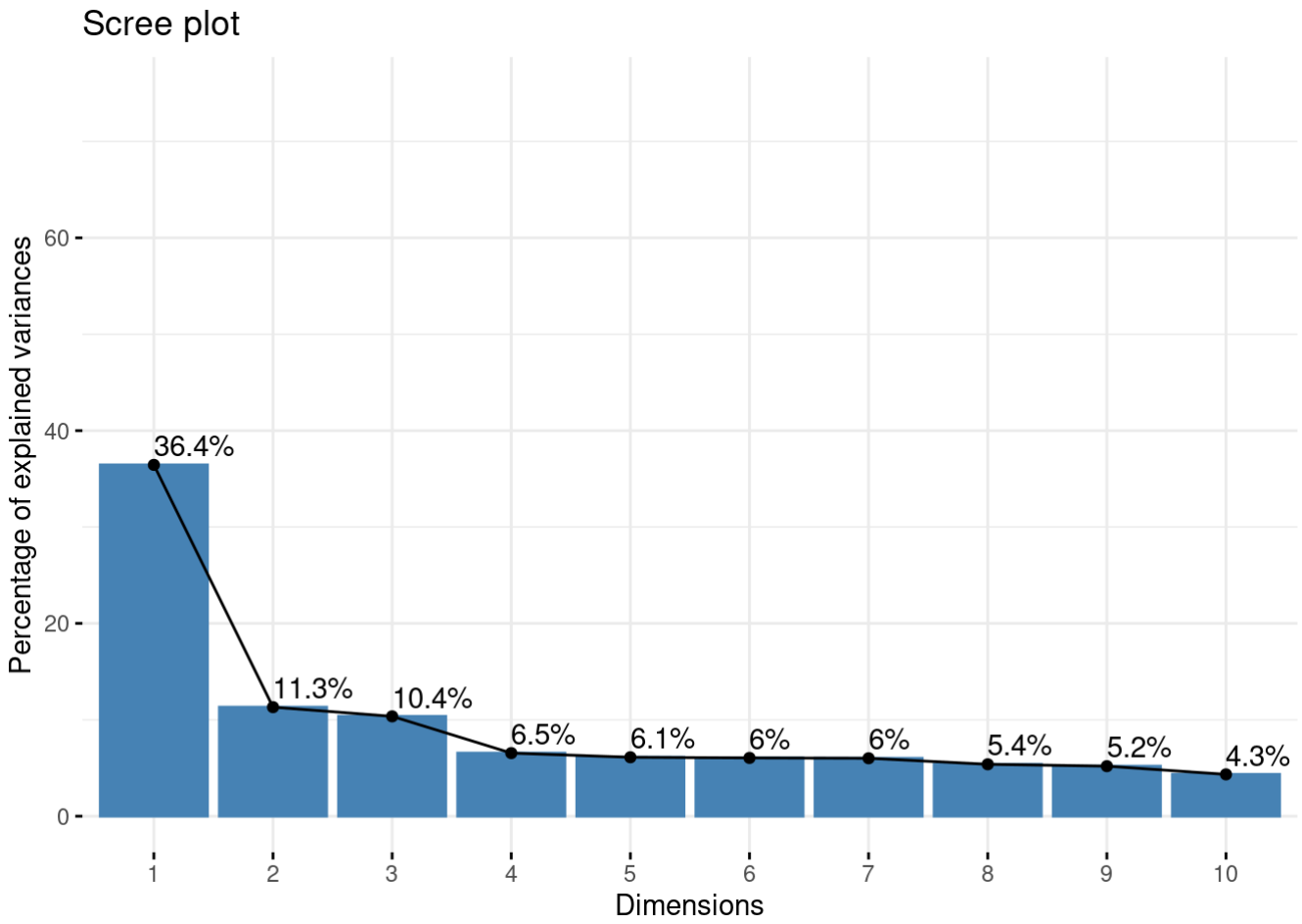
```
# Perform PCA with prcomp()

# Scale and save as new dataset
Tenoch_Transactions_Scaled <- Tenoch_Transactions %>%
  mutate(across(where(is.numeric), scale))

#Look at scaled data
head(Tenoch_Transactions_Scaled)
```

```
## # A tibble: 6 × 33
##   Date       Time     Date_Time           Day   time_o…¹ Gross…² Net S…³ Gift …⁴
##   <date>     <chr>    <dttm>              <ord> <chr>      <dbl>   <dbl>   <dbl>
## 1 2023-04-08 20:52:45 2023-04-08 20:52:45 Sat   Evening   -0.955  -0.955 -0.0346
## 2 2023-04-08 20:45:06 2023-04-08 20:45:06 Sat   Evening   -1.30   -1.30  -0.0346
## 3 2023-04-08 20:43:57 2023-04-08 20:43:57 Sat   Evening   -0.647  -0.647 -0.0346
## 4 2023-04-08 20:42:11 2023-04-08 20:42:11 Sat   Evening   -1.30   -1.30  -0.0346
## 5 2023-04-08 20:40:26 2023-04-08 20:40:26 Sat   Evening   -1.14   -1.14  -0.0346
## 6 2023-04-08 20:35:49 2023-04-08 20:35:49 Sat   Evening   -0.501  -0.501 -0.0346
## # … with 25 more variables: Tax <dbl[,1]>, Tip <dbl[,1]>,
## #   `Total Collected` <dbl[,1]>, Source <chr>, Card <dbl[,1]>,
## #   `Card Entry Methods` <chr>, Cash <dbl[,1]>, `Square Gift Card` <dbl[,1]>,
## #   `NA Drinks` <dbl[,1]>, Beer <dbl[,1]>, `% Tip` <dbl[,1]>, Fees <dbl[,1]>,
## #   `Net Total` <dbl[,1]>, `Card Brand` <chr>, Description <chr>,
## #   Location <chr>, `Dining Option` <chr>, `Customer ID` <chr>,
## #   `Customer Name` <chr>, `Initial Fees` <dbl[,1]>, `Cash App` <dbl[,1]>, …
```

```
#Remove non-numeric columns
TT1 <- Tenoch_Transactions_Scaled[ , unlist(lapply(Tenoch_Transactions_Scaled,
                                   is.numeric))] %>%
  na.omit()

#Perform PCA with prcomp()
pca_TT <- TT1 %>%
  prcomp()

names(pca_TT)
```

```
## [1] "sdev"     "rotation" "center"   "scale"    "x"
```

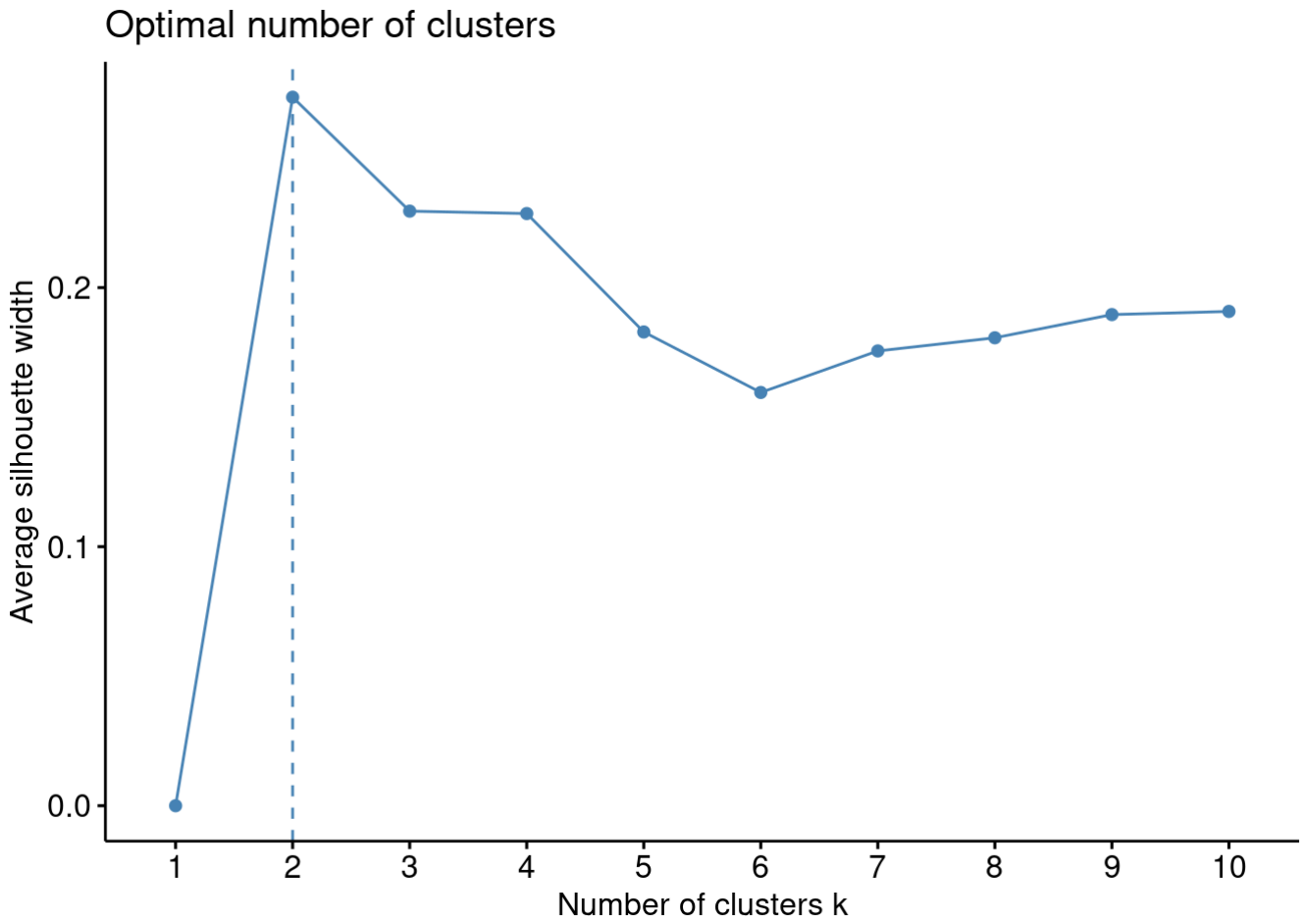
```
#Construct scree plot
fviz_eig(pca_TT, addlabels = TRUE, ylim = c(0, 75))
```



Describe, visualize, and interpret the clusters.

The principle components describe variation in the original data based on linearly uncorrelated variables. The first principle component represents the highest amount of explained variance, with a high value meaning that the component is a good representation of the variables, while a low value means that the component is a poor representation of the variables. The second principle component represents the second highest amount of explained variance, with a high value meaning that the component is a good representation of the variables, while a low value means that the component is a poor representation of the variables. The total amount of variation explained by the first principle component is 45.8% and the amount of variation explained by the second principle component is 11.6%.

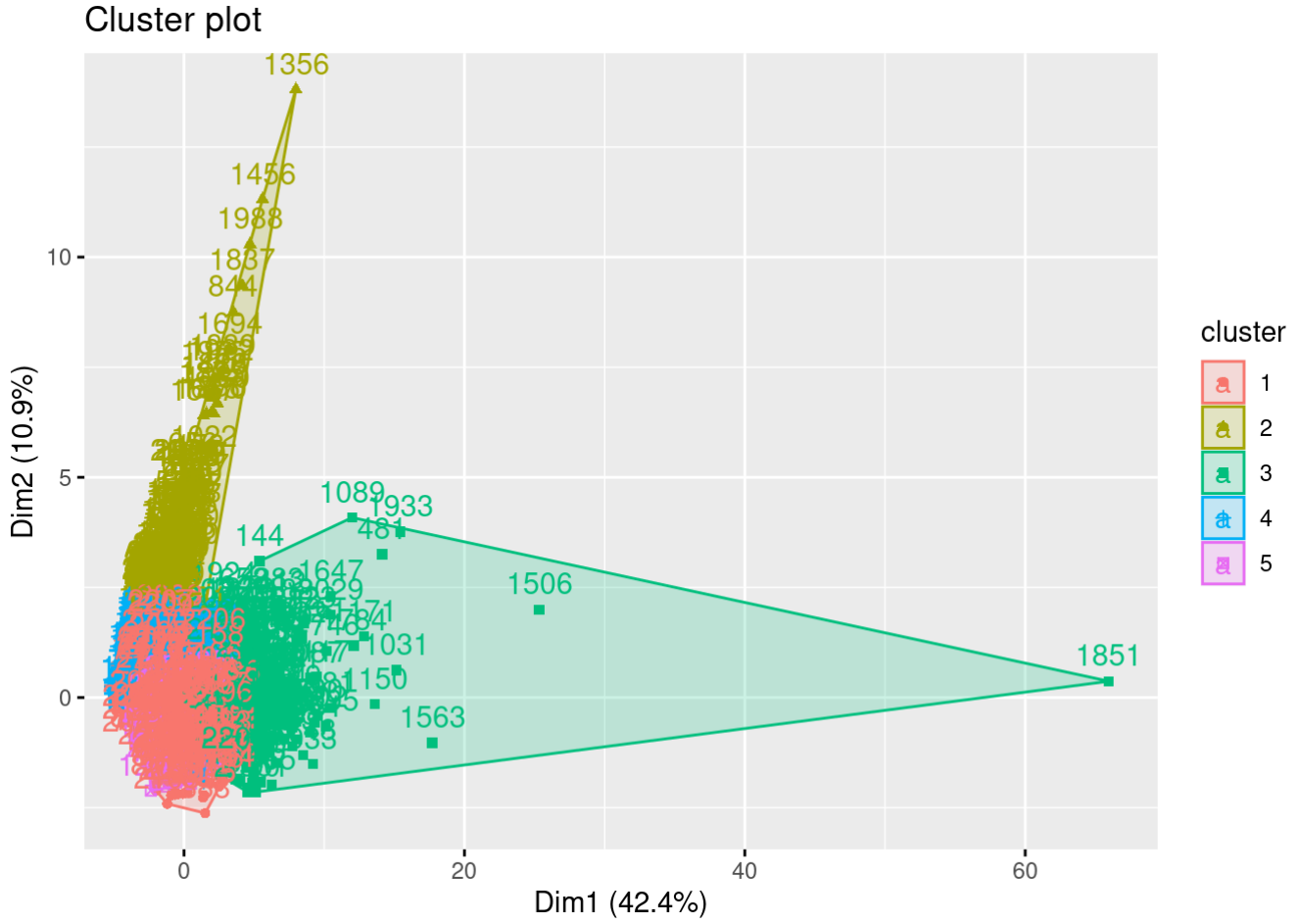
```
# Perform clustering using kmeans() or pam()
# Find number of clusters
fviz_nbclust(TT1, pam, method = "silhouette")
```




```
#Apply PAM with 5 clusters
pam_results <- TT1 %>%
  pam(k = 5)

#Look at resulting object
pam_results
```

```
## Medoids:
##        ID Gross Sales   Net Sales Gift Card Sales          Tax        Tip
##      Total Collected        Card       Cash Square Gift Card  NA Drinks
##            Beer       % Tip        Fees   Net Total Initial Fees    Cash App
##       Day_Numeric time_of_day_num    Day_Bin
##  [ reached getOption("max.print") -- omitted 5 rows ]
## Clustering vector:
##  [1] 1 1 1 1 1 1 1 1 1 2
##  [ reached getOption("max.print") -- omitted 2208 entries ]
## Objective function:
##    build     swap
## 2.461092 2.398477
##
## Available components:
##  [1] "medoids"    "id.med"     "clustering" "objective"  "isolation"
##  [6] "clusinfo"   "silinfo"    "diss"       "call"       "data"
```

```
#Visualize the clusters
fviz_cluster(pam_results, data = Tenoch_Transactions_Scaled)
```



Describe, visualize, and interpret the clusters.

Based on the cluster plot, there is an increasing amount of variance with increasing cluster number. Clusters 1, 2, and 3 are the most centralized with no visible outliers. Clusters 4 and 5 seem to have a few outliers. The values on each axis (corresponding to the first two principle components) for most of the data points in each cluster are fairly low, meaning that they are not represented well by the principle components. The average silhouette width based on the plot is not very high. Values closer to 0 indicate overlapping clusters, so there is likely a fair amount of overlap between the clusters shown on the plot.

6. Discussion

Answer your research questions, reflect on the process, and include acknowledgements.

Something we want to explore is the different factors that affect the amount of money paid in a transaction.

Research Question:

- **Rodrigo:** How does the day of the week of a transaction affect the average Net Total?
- Looking at the analysis, and my first visualization, I can conclude that the average Net Total per transaction is the highest during the weekend and is the lowest on Wednesday.
- The value of the RMSE is 0.4460072, which is quite low, meaning that the model fits the dataset very well.

- The value of R^2 is 0.002505424; this is quite low, meaning that the model doesn't fit the dataset very well.
- Because the RMSE and R^2 contradict each other, my linear model may not be the most accurate to use.
- The average performance of my model is 0.5473923. This means that the predictive outcomes are accurate around 54.74% of the time.
- **Isha:** Does dining option affect the tip amount?
- Looking at the analysis, I can conclude that the tip amount per transaction is higher for customers who order to dine-in or to go than delivery or pickup. This makes sense as the customers who interact with the restaurant staff are probably more likely to tip than customers who don't.
- Looking at my first visualization, the box plot shows that "For Here" and "To Go" orders correspond with the highest tip amounts. They also have the highest median tips.
- My second visualization is a scatterplot, and the dots placed at higher tip amounts correspond with those two dining options as well. Both of these graphs point towards the conclusion that "For Here" and "To Go" dining options have the highest tip amount.
- Moreover, the AUC value based on the kNN prediction model was pretty high (0.8233079), showing that the classifier seems to perform fairly well and is making good predictions.
- **Shobha:** Does the amount tipped per transaction depend on the day of the week?
- Looking at the analysis, I can conclude that the tip amount per transaction is higher on the weekends than on weekdays. This makes sense considering that the average Net total is the highest during the weekends.
- My second visualization shows that the median tip amounts are highest on the weekends. Since the first visualization is skewed by zeroes, it is less helpful to analyze in relation to my research question.
- **The AUC value is 0.5607172**. This is neither high nor low, which means that the classifier seems to have mediocre performance and doesn't make great predictions.

✓ Reflect on the process of conducting this project. What was challenging, what have you learned from the process itself?

- We all thought that conducting this project was quite challenging and took a lot of trial and error. Making sure the data was all tidy and ready for us to analyze was a difficult process. The cross validation was quite problematic to figure out, but we feel that we have a much better understanding of it now that we have completed the project.
- Overall, we learned that data scientists have to make sure that everything is well organized so that their projects are able to work and run properly.

✓ Include acknowledgements for any help received (if a group project, that is where you report the contribution of each member: who did what).

- Rod: Cleaned and Tidied the data, Title and Introduction, correlation matrix, respective visualizations, models, and Cross Validation
- Isha and Shobha: Dimensionality Reduction, Clustering, respective visualizations, models, and Cross Validation

7. Formatting

Remember to knit your file and produce a pdf with multiple pages to upload on Gradescope. It is ok to not follow the structure of the project in order but make sure to identify the pages accordingly.