### Q1.

## a.

# Monthly total bill of all three stores

df <- read.csv("Coffee\_Shop\_Data\_cleaned.csv")

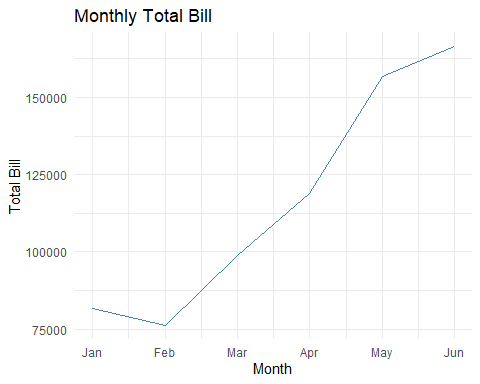
library(dplyr)

##   
## 载入程辑包：'dplyr'

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

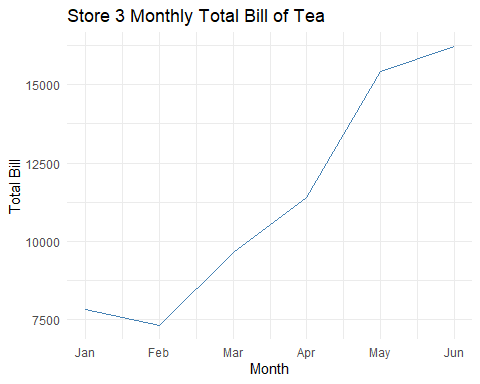
## The following objects are masked from 'package:base':  
##   
## intersect, setdiff, setequal, union

monthly\_total\_bill <- df %>%  
 group\_by(Month) %>%  
 summarise(Total\_Bill = sum(Total\_Bill))  
  
library(ggplot2)  
ggplot(monthly\_total\_bill, aes(x = Month, y = Total\_Bill)) +  
 geom\_line(color = "steelblue") +   
 labs(x = "Month", y = "Total Bill", title = "Monthly Total Bill") +  
 scale\_x\_continuous(breaks = 1:12, labels = c("Jan", "Feb", "Mar", "Apr", "May", "Jun",  
 "Jul", "Aug", "Sep", "Oct", "Nov", "Dec")) +  
 theme\_minimal()

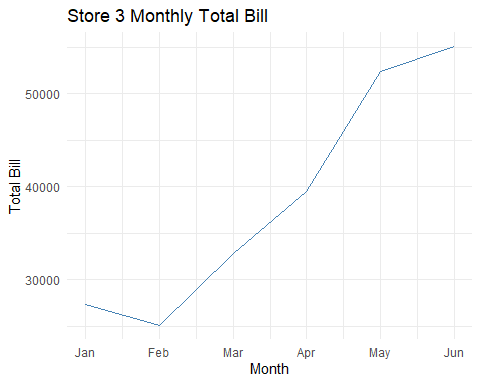


# Monthly total bills of Tea of store 3

Tea\_3\_df <- subset(df, product\_category == "Tea" & store\_id == 3)  
  
library(dplyr)  
monthly\_total\_bill <- Tea\_3\_df %>%  
 group\_by(Month) %>%  
 summarise(Total\_Bill = sum(Total\_Bill))  
  
library(ggplot2)  
ggplot(monthly\_total\_bill, aes(x = Month, y = Total\_Bill)) +  
 geom\_line(color = "steelblue") +   
 labs(x = "Month", y = "Total Bill", title = "Store 3 Monthly Total Bill of Tea") +  
 scale\_x\_continuous(breaks = 1:12, labels = c("Jan", "Feb", "Mar", "Apr", "May", "Jun",  
 "Jul", "Aug", "Sep", "Oct", "Nov", "Dec")) +  
 theme\_minimal()

 # Monthly total bills of of store 3

store\_3\_df <- subset(df, store\_id == 3)  
  
library(dplyr)  
monthly\_total\_bill <- store\_3\_df %>%  
 group\_by(Month) %>%  
 summarise(Total\_Bill = sum(Total\_Bill))  
  
library(ggplot2)  
ggplot(monthly\_total\_bill, aes(x = Month, y = Total\_Bill)) +  
 geom\_line(color = "steelblue") +   
 labs(x = "Month", y = "Total Bill", title = "Store 3 Monthly Total Bill") +  
 scale\_x\_continuous(breaks = 1:12, labels = c("Jan", "Feb", "Mar", "Apr", "May", "Jun",  
 "Jul", "Aug", "Sep", "Oct", "Nov", "Dec")) +  
 theme\_minimal()

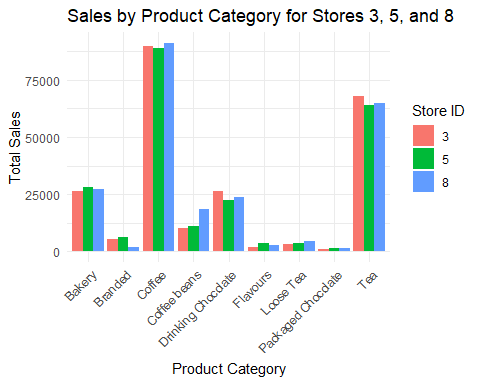


## b.

library(dplyr)  
sales\_by\_category <- df %>%  
 group\_by(store\_id, product\_category) %>%  
 summarise(Total\_Sales = sum(Total\_Bill))

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

library(ggplot2)  
ggplot(sales\_by\_category, aes(x = product\_category, y = Total\_Sales, fill = factor(store\_id))) +  
 geom\_bar(stat = "identity", position = "dodge") +  
 labs(x = "Product Category", y = "Total Sales", title = "Sales by Product Category for Stores 3, 5, and 8") +  
 scale\_fill\_discrete(name = "Store ID") +  
 theme\_minimal() +  
 theme(axis.text.x = element\_text(angle = 45, hjust = 1))

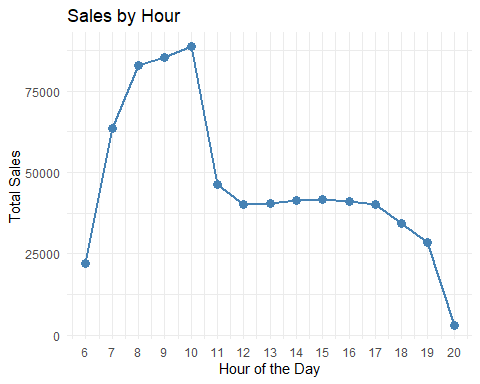


## c.

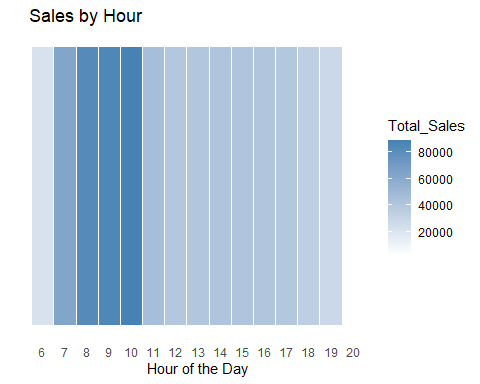
# line chart

library(dplyr)  
sales\_by\_hour <- df %>%  
 group\_by(Hour) %>%  
 summarise(Total\_Sales = sum(Total\_Bill))  
  
library(ggplot2)  
ggplot(sales\_by\_hour, aes(x = Hour, y = Total\_Sales)) +  
 geom\_line(color = "steelblue", size = 1) +  
 geom\_point(color = "steelblue", size = 3) +  
 labs(x = "Hour of the Day", y = "Total Sales", title = "Sales by Hour") +  
 scale\_x\_continuous(breaks = seq(0, 23, by = 1)) +  
 theme\_minimal()

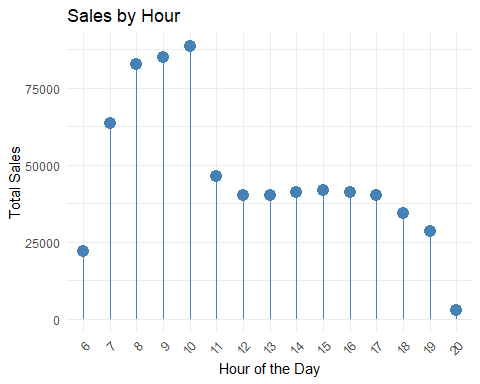
## Warning: Using `size` aesthetic for lines was deprecated in ggplot2 3.4.0.  
## ℹ Please use `linewidth` instead.  
## This warning is displayed once every 8 hours.  
## Call `lifecycle::last\_lifecycle\_warnings()` to see where this warning was  
## generated.

 # heat map

library(ggplot2)  
ggplot(sales\_by\_hour, aes(x = factor(Hour), y = 1, fill = Total\_Sales)) +  
 geom\_tile(color = "white", size = 0.5) +  
 scale\_fill\_gradient(low = "white", high = "steelblue") +  
 labs(x = "Hour of the Day", y = "", title = "Sales by Hour") +  
 theme\_minimal() +  
 theme(axis.text.y = element\_blank(),  
 axis.ticks.y = element\_blank(),  
 panel.grid = element\_blank())

 # lollipop chart

library(ggplot2)  
ggplot(sales\_by\_hour, aes(x = factor(Hour), y = Total\_Sales)) +  
 geom\_point(color = "steelblue", size = 4) +  
 geom\_segment(aes(x = factor(Hour), xend = factor(Hour), y = 0, yend = Total\_Sales), color = "steelblue") +  
 labs(x = "Hour of the Day", y = "Total Sales", title = "Sales by Hour") +  
 theme\_minimal() +  
 theme(axis.text.x = element\_text(angle = 45, hjust = 1))



### Q2.

# install.packages("randomForest")  
library(randomForest)

## randomForest 4.7-1.1

## Type rfNews() to see new features/changes/bug fixes.

##   
## 载入程辑包：'randomForest'

## The following object is masked from 'package:ggplot2':  
##   
## margin

## The following object is masked from 'package:dplyr':  
##   
## combine

library(dplyr)

tea\_sales <- df %>%  
 filter(product\_category == "Tea") %>%  
 group\_by(store\_id, Day.of.Week, Hour) %>%  
 summarise(Total\_Bill = sum(Total\_Bill))

## `summarise()` has grouped output by 'store\_id', 'Day.of.Week'. You can override  
## using the `.groups` argument.

set.seed(123)   
train\_index <- sample(nrow(tea\_sales), 0.7 \* nrow(tea\_sales))  
train\_data <- tea\_sales[train\_index, ]  
test\_data <- tea\_sales[-train\_index, ]

rf\_model <- randomForest(Total\_Bill ~ store\_id + Day.of.Week + Hour,  
 data = train\_data,  
 ntree = 500,  
 mtry = 2,  
 importance = TRUE)

# Make predictions on the test data  
predictions <- predict(rf\_model, newdata = test\_data)  
  
# Calculate the mean squared error (MSE)  
mse <- mean((test\_data$Total\_Bill - predictions)^2)  
print(paste0("Mean Squared Error (MSE): ", mse))

## [1] "Mean Squared Error (MSE): 8019.75246870488"

# Calculate the root mean squared error (RMSE)  
rmse <- sqrt(mse)  
print(paste0("Root Mean Squared Error (RMSE): ", rmse))

## [1] "Root Mean Squared Error (RMSE): 89.5530706827236"

# Calculate the R-squared (R^2)  
r\_squared <- 1 - (sum((test\_data$Total\_Bill - predictions)^2) / sum((test\_data$Total\_Bill - mean(test\_data$Total\_Bill))^2))  
print(paste0("R-squared (R^2): ", r\_squared))

## [1] "R-squared (R^2): 0.905851146727362"

importance\_scores <- importance(rf\_model)  
print(importance\_scores)

## %IncMSE IncNodePurity  
## store\_id 61.68627 2651828.2  
## Day.of.Week -11.06519 714362.8  
## Hour 102.86598 15624045.0

new\_data <- data.frame(  
 store\_id = c(3, 5, 8),  
 Day.of.Week = rep("Wednesday", 3),  
 Hour = rep(11, 3)  
)

predictions <- predict(rf\_model, newdata = new\_data)  
  
results <- data.frame(new\_data, Total\_Bill = predictions)  
print(results)

## store\_id Day.of.Week Hour Total\_Bill  
## 1 3 Wednesday 11 657.3200  
## 2 5 Wednesday 11 530.0715  
## 3 8 Wednesday 11 547.5709

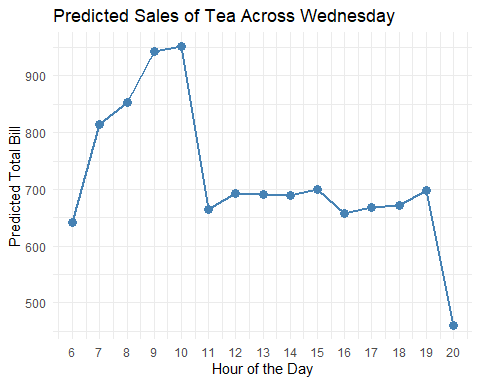
### Extra Credit

new\_data <- data.frame(  
 store\_id = rep(3, 15),   
 Day.of.Week = rep(2,15),   
 Name.Day = ("Wednesday"),  
 product\_catery = ("Tea"),  
 Hour = seq(6, 20)  
)

predictions <- predict(rf\_model, newdata = new\_data)  
results <- data.frame(new\_data, Total\_Bill = predictions)  
results

## store\_id Day.of.Week Name.Day product\_catery Hour Total\_Bill  
## 1 3 2 Wednesday Tea 6 641.1136  
## 2 3 2 Wednesday Tea 7 814.7872  
## 3 3 2 Wednesday Tea 8 853.3714  
## 4 3 2 Wednesday Tea 9 943.6609  
## 5 3 2 Wednesday Tea 10 951.7408  
## 6 3 2 Wednesday Tea 11 664.2715  
## 7 3 2 Wednesday Tea 12 693.6467  
## 8 3 2 Wednesday Tea 13 690.5155  
## 9 3 2 Wednesday Tea 14 689.8835  
## 10 3 2 Wednesday Tea 15 700.4262  
## 11 3 2 Wednesday Tea 16 658.0419  
## 12 3 2 Wednesday Tea 17 667.8849  
## 13 3 2 Wednesday Tea 18 672.1328  
## 14 3 2 Wednesday Tea 19 697.2537  
## 15 3 2 Wednesday Tea 20 460.2027

library(ggplot2)  
# Line plot  
ggplot(results, aes(x = Hour, y = Total\_Bill)) +  
 geom\_line(color = "steelblue", size = 1) +  
 geom\_point(color = "steelblue", size = 3) +  
 labs(x = "Hour of the Day", y = "Predicted Total Bill", title = "Predicted Sales of Tea Across Wednesday") +  
 scale\_x\_continuous(breaks = seq(0, 23, by = 1)) +  
 theme\_minimal()



# Bar plot  
ggplot(results, aes(x = factor(Hour), y = Total\_Bill)) +  
 geom\_bar(stat = "identity", fill = "steelblue") +  
 labs(x = "Hour of the Day", y = "Predicted Total Bill", title = "Predicted Sales of Tea Across Wednesday") +  
 theme\_minimal() +  
 theme(axis.text.x = element\_text(angle = 45, hjust = 1))

