Final Project

Dahye Chung, Donguk Yoo, Hanseung Jang, Sanghyun Lee, Jungyoon Choi, Seokyeong Park, Semin Seo

2023-07-19

```
library(tidyverse)
```

```
## -- Attaching core tidyverse packages ------------------------ tidyverse 2.0.0 --
## v dplyr     1.1.2     v readr     2.1.4
## v forcats   1.0.0     v stringr   1.5.0
## v ggplot2   3.4.2     v tibble    3.2.1
## v lubridate 1.9.2     v tidyr     1.3.0
## v purrr     1.0.1
## -- Conflicts ------------------------------------------ tidyverse_conflicts() --
## x dplyr::filter() masks stats::filter()
## x dplyr::lag()    masks stats::lag()
## i Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to beco
```

```
library(broom)
library(tidyverse)
library(tidyr)
library(dplyr)
```

###Part 9

```
library(tidyr)
library(ggplot2)
library(dplyr)
library(readr)
library(class)
library(caret)

Sleep_health_and_lifestyle_dataset <- read_csv(file = "Sleep_health_and_lifestyle_dataset.csv"
  col_types = cols(
    'Person ID' = col_character(),
    'Age' = col_double(),
    'Sleep Duration' = col_double(),
    'Stress Level' = col_double(),
    'Physical Activity Level' = col_double(),
    'Quality of Sleep' = col_double(),
```

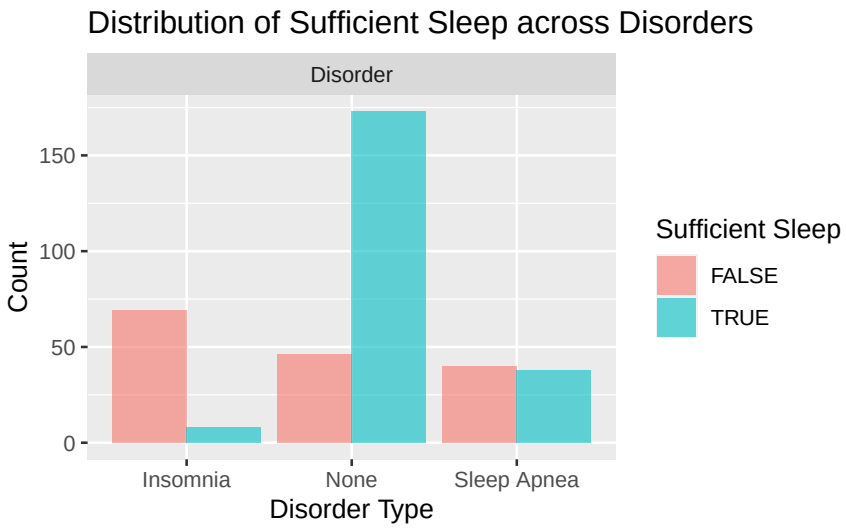
```
    'BMI Category' = col_character(),
    'Blood Pressure' = col_character(),
    'Heart Rate' = col_double(),
    'Daily Steps' = col_double(),
    'Sleep Disorder' = col_character()
  ))

Sleep_health_and_lifestyle_dataset_renamed <- Sleep_health_and_lifestyle_dataset %>%
  rename(ID = 'Person ID',
         Duration = 'Sleep Duration',
         Stress = 'Stress Level',
         Physical = 'Physical Activity Level',
         Quality = 'Quality of Sleep',
         BMI = 'BMI Category',
         BPressure = 'Blood Pressure',
         HRate = 'Heart Rate',
         DSteps = 'Daily Steps',
         Disorder = 'Sleep Disorder')


sleep_data <- Sleep_health_and_lifestyle_dataset_renamed %>%
    mutate(sufficient_sleep = Duration >= 7.0)
```

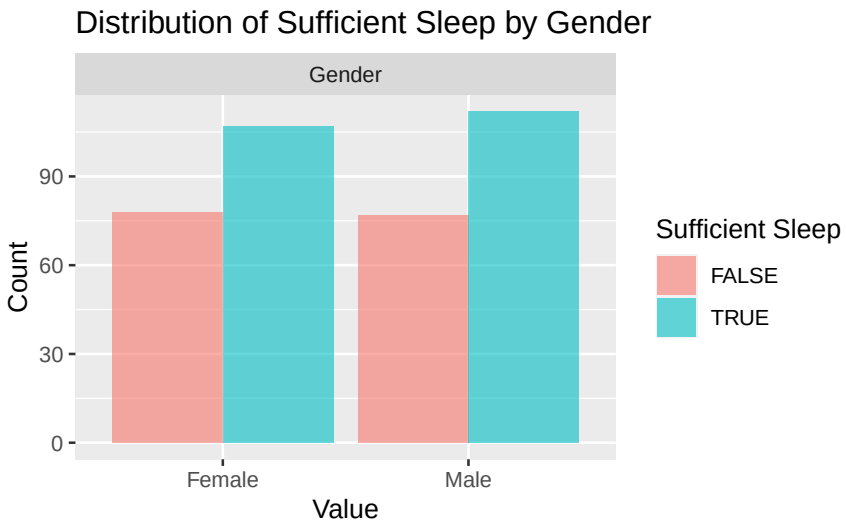
```
sleep_data %>%
  pivot_longer(cols = c(Disorder), names_to = "variable", values_to = "value") %>%
  group_by(variable, value, sufficient_sleep) %>%
  summarise(count = n()) %>%
  ggplot() +
  geom_bar(
    mapping = aes(x = value, y = count, fill = sufficient_sleep),
    position = "dodge",
    alpha = 0.6,
    stat = "identity"
  ) +
  facet_wrap(~ variable, scales = "free") +
  labs(title = "Distribution of Sufficient Sleep across Disorders",
       x = "Disorder Type",
       y = "Count",
       fill = "Sufficient Sleep")
```

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



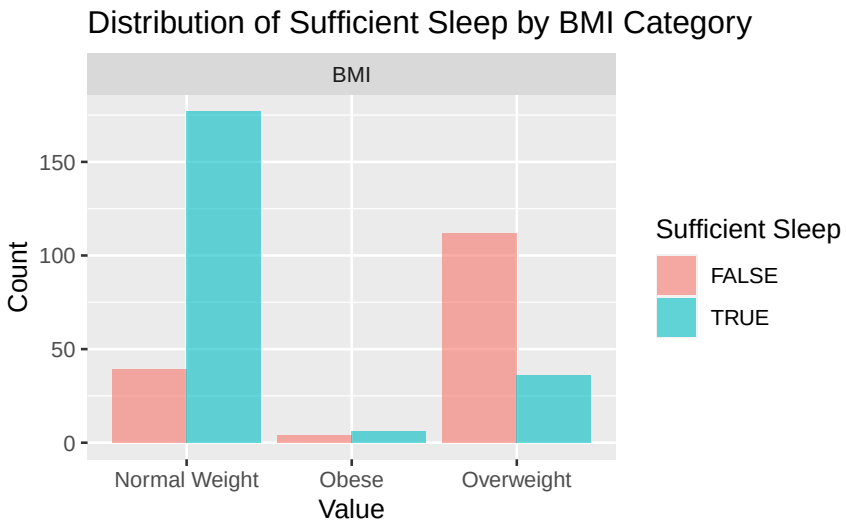

```
sleep_data %>%
  pivot_longer(cols = c(Gender), names_to = "variable", values_to = "value") %>%
  group_by(variable, value, sufficient_sleep) %>%
  summarise(count = n()) %>%
  ggplot() +
  geom_bar(
    mapping = aes(x = value, y = count, fill = sufficient_sleep),
    position = "dodge",
    alpha = 0.6,
    stat = "identity"
  ) +
  facet_wrap(~ variable, scales = "free") +
  labs(title = "Distribution of Sufficient Sleep by Gender",
       x = "Value",
       y = "Count",
       fill = "Sufficient Sleep")
```

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



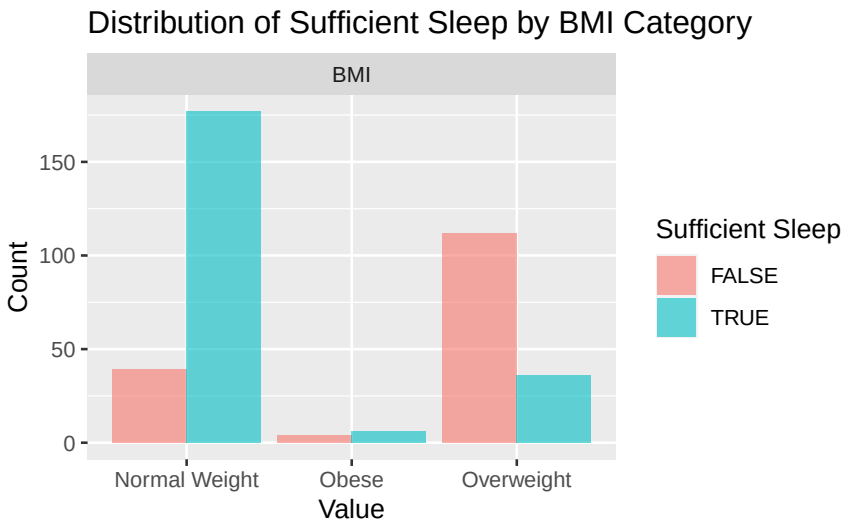

```
sleep_data %>%
  pivot_longer(cols = c(BMI), names_to = "variable", values_to = "value") %>%
  mutate(value = ifelse(value == "Normal", "Normal Weight", value)) %>%
  group_by(variable, value, sufficient_sleep) %>%
  summarise(count = n()) %>%
  ggplot() +
  geom_bar(
    mapping = aes(x = value, y = count, fill = sufficient_sleep),
    position = "dodge",
    alpha = 0.6,
    stat = "identity"
  ) +
  facet_wrap(~ variable, scales = "free") +
  labs(title = "Distribution of Sufficient Sleep by BMI Category",
       x = "Value",
       y = "Count",
       fill = "Sufficient Sleep")
```

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




```
sleep_data %>%
  pivot_longer(cols = c(BMI), names_to = "variable", values_to = "value") %>%
  mutate(value = ifelse(value == "Normal", "Normal Weight", value)) %>%
  group_by(variable, value, sufficient_sleep) %>%
  summarise(count = n()) %>%
  ggplot() +
  geom_bar(
    mapping = aes(x = value, y = count, fill = sufficient_sleep),
    position = "dodge",
    alpha = 0.6,
    stat = "identity"
  ) +
  facet_wrap(~ variable, scales = "free") +
  labs(title = "Distribution of Sufficient Sleep by BMI Category",
       x = "Value",
       y = "Count",
       fill = "Sufficient Sleep")
```

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




```r
mode_gender <- as.character(names(which.max(table(sleep_data$Gender))))
mode_occupation <- as.character(names(which.max(table(sleep_data$Occupation))))
mode_bmi <- as.character(names(which.max(table(sleep_data$BMI))))

sleep_data <- sleep_data %>%
mutate(
  Gender = if_else(is.na(Gender), mode_gender, Gender),
  Occupation = if_else(is.na(Occupation), mode_occupation, Occupation),
  BMI = if_else(is.na(BMI), mode_bmi, BMI)
)
```

```r
sleep_data$sufficient_sleep <- ifelse(sleep_data$Duration >= 7, "Sufficient", "Insufficient")

set.seed(123)
train_indices <- createDataPartition(sleep_data$sufficient_sleep, p = 0.7, list = FALSE)
trainingSet <- sleep_data[train_indices, ]
testSet <- sleep_data[-train_indices, ]
```

```r
trainingSet$sufficient_sleep <- as.factor(trainingSet$sufficient_sleep)
testSet$sufficient_sleep <- as.factor(testSet$sufficient_sleep)

training_Outcomes <- trainingSet$sufficient_sleep
test_Outcomes <- testSet$sufficient_sleep
```

```r
model <- glm(sufficient_sleep ~ Age + Gender + Occupation + Physical + DSteps + BMI, data = tra

predictions <- predict(model, newdata = testSet, type = "response")

threshold <- 0.5
predicted_classes <- as.factor(ifelse(predictions >= threshold, "Sufficient", "Insufficient"))
```

```r
actual_classes <- test_Outcomes
accuracy <- sum(predicted_classes == actual_classes) / length(actual_classes)
print(paste("Accuracy:", accuracy))
```

```
## [1] "Accuracy: 0.981981981981982"
```

```r
model_1_preds <- testSet %>%
  add_predictions(model, type = "response") %>%
  mutate(
    outcome = as.factor(if_else(condition = pred > threshold,
                      "Sufficient", "Insufficient"))
  )
```