Homework - 1

Alex: Identify a data set of interest to you with a quantitative variable you wish to predict and/or model. This outcome variable must not be categorical. Use your own kNN algorithm to predict/model the value of your variable of interest using at least one categorical and one quantitative variable, preferably more.

Amuna:

I have always looked forward to having a family of my own one day, and I hope to have both a daughter and a son. Fueled by this excitement, I was motivated to build an algorithm that predicts the height of my future children.

For this project, I used a dataset collected by Professor Myers, which includes the heights of Middlebury economics students along with their parents' heights over the past ten years. However, this dataset comes with some caveats. The majority of Middlebury economics students are male and many participate in sports, which introduces potential bias into the predictions. Since male and athletic individuals tend to be taller on average, the model may not fully generalize to a broader population. Despite these limitations, the dataset provides a sufficient foundation for me to develop a k-Nearest Neighbors algorithm to estimate the height of my future children. Of course, I will need to find a partner first.

Variables of Interest: Quantitative (1): Father's Height Quantitative (2): Mother's Height Categorical (1): Sex of the Offspring Outcome (1): Offspring Height

Reading in our dataset

```
library(tidyverse)
```

```
── Attaching core tidyverse packages ──────────────────────── tidyverse 2.0.0 ──
✔ dplyr     1.1.4     ✔ readr     2.1.5
✔ forcats   1.0.0     ✔ stringr   1.5.1
✔ ggplot2   3.5.1     ✔ tibble    3.2.1
✔ lubridate 1.9.4     ✔ tidyr     1.3.1
✔ purrr     1.0.2
── Conflicts ────────────────────────────────────────── tidyverse_conflicts() ──
✖ dplyr::filter() masks stats::filter()
✖ dplyr::lag()    masks stats::lag()
ℹ Use the conflicted package (<http://conflicted.r-lib.org/>) to force all conflicts to
become errors
```

```
library(haven)
height_data <- read_dta("~/Downloads/heights.dta")
```

Simple Peek Into the Dataset: Now, let's look at a scatterplot that shows the sample distribution of our dataset across different female and male offspring. This helps me visualize how the dataset looks like. The shading for male offspring appears lighter than for female offspring, indicating that males are, on average, taller than females, even if their parents have similar heights.

```
height_data |>
  ggplot() +
  geom_point(aes(x=fheight, y=mheight, color = height)) +
  facet_wrap(~sex) +
  scale_color_viridis_c()
```

Now let's get started with building the algorithm. Removed the "year" column because we don't need it.

```
height_data <- subset(height_data, select = -year)
```

We need a way to mathematically calculate the categorical variable to make predictions, so I'm assigning binary numbers to two sexes. Female is 0 and male is 1.

```
height_data_binary <- height_data |>
  mutate(sex_binary = case_when(str_detect(sex, "Female") ~ 0,
                                str_detect(sex, "Male") ~ 1))
```

Now, all our variables are numeric; however, if we directly plug them into the distance formula, we will not get any meaningful answers because sex is assigned only 0 and 1, whereas parents' heights are in inches, perhaps ranging between numbers between 60 and 80. Since the weightage of different variables is arbitrary right now, we need to systemize and scale them to make realistic calculations. Let's scale all the numbers to range from 0 to 1, meaning the shortest height will be assigned 0 and the tallest height will be assigned 1. Below, I am finding the shortest and tallest heights for both fathers and mothers. The tallest person is 78 inches tall, and the shortest person is 55 inches tall, so 78 inches will be 1 and 55 inches will be 0. Everyone else will be assigned a value between 0 and 1 based on their height.

As for sex, since it is only coded as 0 and 1, male offspring would have significantly greater weight in the calculations compared to females, whose value is 0. To address this, we need to assign values on the same scale. It feels arbitrary to assign 0.2 to females and 0.5 to males without a basis, so instead, I find the average height of female and male offspring and scale it using the same method as parents' heights. This ensures the assignment is based on some rationale rather than random values. As a result, all our variables are scaled between 0 and 1. I found that the average male offspring height is 71.22503 inches, which is assigned 0.7054360, and the average female offspring height is 65.86296 inches, which is assigned 0.4723025 according to the scale I created.

```
height_data_binary |>
  summarize(mheight_min = min(mheight),
            mheight_max = max(mheight),
            fheight_min = min(fheight),
            fheight_max = max(fheight))
```

```
# A tibble: 1 × 4
  mheight_min mheight_max fheight_min fheight_max
        <dbl>       <dbl>       <dbl>       <dbl>
1          55          73          62          78
```

```
height_min <- 55
height_max <- 78

height_data_averagesex <- height_data_binary |>
  mutate(
    average_height = case_when(
      sex_binary == 0 ~ mean(height[sex_binary == 0]),
      sex_binary == 1 ~ mean(height[sex_binary == 1])
    )
  )

height_data_scaled <- height_data_averagesex |>
  mutate(mheight_scaled = (mheight - height_min)/(height_max - height_min),
         fheight_scaled = (fheight - height_min)/(height_max - height_min),
         sex_scaled = (average_height - height_min)/(height_max - height_min))
```

Now, it's time to use the Euclidean distance formula to find the nearest neighbors to make predictions. By inputting the mother's height, father's height, and sex, the function identifies the nearest neighbors based on the

chosen ‘k’ value. If k equals 3, the function finds the three closest neighbors. Then, we take the average offspring height of these three points to produce the final prediction.

```
mheight_input <- 0.10869565
fheight_input <- 0.7391304
sex_input <- 0.7054360

k<- 3

height_data_scaled |>
  mutate(distance = sqrt((mheight_input - mheight_scaled)^2 + (fheight_input -
         fheight_scaled)^2 + (sex_input - sex_scaled)^2)) |>
  arrange(distance) |>
  head(k) |>
  summarize(average_height_k = mean(height)) |>
  pull(average_height_k)
```

```
[1] 67.66667
```

Currently, my dataset has many rows that I don’t need since we will only be using the scaled version, let’s get rid of the ones we don’t need.

```
height_data_useful <- height_data_scaled[, c("height", "mheight_scaled", "fheight_scaled",
        "sex_scaled")]
```

We are also ready to write our knn function.

```
my_knn_haha <- function(height_data_haha, mheight_input, fheight_input, sex_input, k = 17)
  {
  height_data_haha |>
  mutate(distance = sqrt((mheight_input - mheight_scaled)^2 + (fheight_input -
         fheight_scaled)^2 + (sex_input - sex_scaled)^2)) |>
  arrange(distance) |>
  head(k) |>
  summarize(average_height_k = mean(height)) |>
  pull(average_height_k)
}

my_knn_haha(height_data_useful, 0.8, 0.7, 0.5, k=17)
```

```
[1] 68.42647
```

In order to determine the accuracy of my algorithm, we will randomly split the data into two sections: training and testing.

```
set.seed(1)
training_data_rows <- sample(1:nrow(height_data_useful),
                             size = nrow(height_data_useful)/2)

training_data3 <- height_data_useful[training_data_rows, ]
testing_data3 <- height_data_useful[-training_data_rows, ]
```

Alex: Justify explicitly the value(s) of k you found to be ‘optimal’.

Amuna: Now, let’s write a for-loop for our training and testing data. Because predicting exact offspring heights is rare and challenging, a good algorithm here should aim for predictions within 2–3 inches, so a simple true-or-false check isn’t appropriate. Instead, I’m using the mean absolute error, which averages how far predictions deviate from actual values. If this average is high, we are in trouble. On the other hand, minimizing this value

will mean we are making our algorithm more accurate. I tested all k values from 1 to 25 and found that accuracy began to decline after k = 17, so I chose 17 as the optimal value. Since there are more than 200 observations, I think 17 is a reasonable choice. If there were only 20 observations, k = 17 would be too large and introduce excessive noise.

```
preds <- NULL

for(row in 1:nrow(testing_data3)){
  preds[row] <- my_knn_haha(
                       training_data3,
                       testing_data3[[row,"mheight_scaled"]],
                       testing_data3[[row, "fheight_scaled"]],
                       testing_data3[[row, "sex_scaled"]],
                       k = 17)
}

preds
```

```
  [1] 63.89824 67.91176 64.72941 71.86471 63.95706 72.05882 69.75000 73.98235
  [9] 71.52353 73.21765 73.45294 64.09353 64.35882 70.72647 71.05294 64.83647
 [17] 65.36588 71.26471 63.89824 69.97647 71.52353 74.12412 65.50736 66.25882
 [25] 66.22941 73.93588 73.64176 70.81471 73.40647 68.72941 69.85882 70.53529
 [33] 73.70059 68.52353 64.79941 71.61765 71.38235 71.26471 71.38235 65.48765
 [41] 70.75000 70.30000 69.11765 72.43588 72.64118 71.47059 64.35882 64.41294
 [49] 71.26471 64.83647 65.29941 64.72941 65.09353 67.64118 71.39706 71.93529
 [57] 68.08235 70.72647 71.80588 68.99412 72.25941 73.40647 73.64176 72.31765
 [65] 72.31765 68.72941 71.38235 69.29412 73.98235 72.47059 72.90647 64.27324
 [73] 73.27647 71.86471 71.73529 64.62618 72.96529 71.73529 65.72941 64.54706
 [81] 65.66412 73.98235 70.90294 67.91176 71.75882 72.31765 63.89824 64.01000
 [89] 65.36588 70.35294 70.63824 68.99412 64.18176 70.86765 70.76471 69.71176
 [97] 65.09353 70.70588 70.75000 71.75882 66.84824 66.84824 70.72647 71.75882
[105] 66.01765 65.48353 71.47059 65.92471 68.08235 70.72647 67.91176 70.53529
[113] 70.01471 68.79412 72.31765 70.53529 65.96471 65.51882 69.83824 70.44706
[121] 71.75882 65.13677 67.34706 72.14118 71.44118 70.52059 70.61765 68.77059
[129] 70.90294 67.85030 65.78176 70.90294 72.90647 67.17383 64.83647 70.41176
```

```
mae <- mean(abs(preds - testing_data3$height))
print(mae)
```

```
[1] 1.765705
```

Alex: Provide a 2d or 3d visualization that gives intuition about the regions/surfaces of your kNN algorithm. In other words, visualize how your algorithm is making predictions projected onto a 2d or 3d space. Describe why you chose the variables you did and what your visualization tells you about their relationship with the outcome variable.

Amuna: Since my dataset contains three variables (father's height, mother's height, and offspring sex), I am faceting the graphs by sex and creating two 2D visualizations–one for female offspring and one for male offspring.

In these plots:

Father's height is on the X-axis Mother's height is on the Y-axis Fill is predicted offspring heights The left facet represents female offspring predictions. The right facet represents Male offspring predictions.

I used a grid of all possible parental height combinations. This allows us to apply the k-Nearest Neighbors algorithm to predict offspring height for every possible combination of father's and mother's height across both sexes.

Interpretation of the visualizations:

- The smooth transition of the gradient suggests that our knn algorithm makes predictions based on the nearby data. I think k=17 seems was a good choice.

- The male offspring tend to have higher predicted heights compared to female offspring, as indicated by more yellow shades on the right side of the male facet. On the other hand, the darker purple regions in the female facet are indicative of predictions of shorter heights for female offspring. This shows that our algorithm captures the height differences across box sexes.

- Overall, when both parents are taller, predicted offspring height is also taller (around 72 inches or more). The opposite trend appears for shorter parents, where predictions lean toward shorter offspring heights. This pattern aligns with the concept of assortative mating, which suggests that people tend to choose partners with similar physical traits, including height.

- The color gradient shifts more noticeably along the X-axis than the Y-axis, suggesting that father's height has a stronger influence on the offspring's predicted height, especially for male offspring.

```
library(ggplot2)

every_person_ever <- expand.grid(mheight_scaled = seq(0, 1, by = 0.01),
                                 fheight_scaled = seq(0, 1, by = 0.01),
                                 sex_scaled = c(0.5, 0.7))

preds2 <- numeric(nrow(every_person_ever))

for (row in 1:nrow(every_person_ever)) {
  preds2[row] <- my_knn_haha(training_data3,
                             every_person_ever[row, "mheight_scaled"],
                             every_person_ever[row, "fheight_scaled"],
                             every_person_ever[row, "sex_scaled"],
                             k = 17)
}

sex_label <- as_labeller(c("0.5" = "Female", "0.7" = "Male"))

every_person_ever |>
  mutate(predicted_height = preds2) |>
  ggplot() +
  geom_tile(aes(x=fheight_scaled,y = mheight_scaled, fill = predicted_height)) +
   labs(title = "k-Nearest Neighbor (k=17)",
        x = "Father's Height (Scaled)",
        y = "Mother's Height (Scaled)",
        fill = "Predicted Offspring Height") +
  facet_wrap(~sex_scaled, labeller = sex_label) +
  scale_fill_viridis_c()
```

Thank you for following along with me so far. This was great. I can't believe I did this!!!