Problem Set 4

Matt Krasnow

Due Friday, October 4, 2024 at 11:59pm

Problem set policies. *Please provide concise, clear answers for each question while making sure to fully explain your reasoning. For problems asking for calculations, show your work in addition to clearly indicating the final answer. For problems involving* `R`*, be sure to include the code and output in your solution.*

Please submit the PDF of your knit solutions to Gradescope and be sure to assign which pages of your solution correspond to each problem. Make sure that the PDF is fully readable to the graders; e.g., make sure that lines don't run off the page margin.

We encourage you to discuss problems with other students (and, of course, with the teaching team), but you must write your final answer in your own words. Solutions prepared "in committee" are not acceptable. If you do collaborate with classmates on a problem, please list your collaborators on your solution. Be aware that simply copying answers found online, whether human-generated or machine-generated, is a violation of the Honor Code.

Question 1

Consider a simple linear regression, with an intercept and one predictor.

(a) Write down the design matrix $\mathbf{X}$ and calculate the 2 x 2 matrix $(\mathbf{X}^T\mathbf{X})^{-1}$.

The design matrix of a linear model is a matrix representation of the data.

The linear model is represented as:

$$\vec{Y} = \vec{X}\vec{\beta} + \vec{\epsilon}$$

where $\vec{Y}$ is the observed responses (vector)

$\vec{X}$ is the design matrix

$\vec{\beta}$ is the vector of regression coefficients

$\vec{\epsilon}$ is the error vector (variation not explained by the model)

So the design matrix is:

$$\vec{X} = \begin{pmatrix} 1 & x_1 \\ 1 & x_2 \\ \vdots & \vdots \\ 1 & x_n \end{pmatrix}$$

$$\vec{X}^T = \begin{pmatrix} 1 & 1 & 1 & \cdots & 1 \\ x_1 & x_2 & x_3 & \cdots & x_n \end{pmatrix}$$

$$\left(\vec{X}^T\vec{X}\right)^{-1} = \begin{pmatrix} n & \sum x_i \\ \sum x_i & \sum x_i^2 \end{pmatrix}^{-1}$$

$$\vec{A} = \begin{pmatrix} n & \sum x_i \\ \sum x_i & \sum x_i^2 \end{pmatrix}^{-1}$$

$$\vec{A}^{-1} = \frac{1}{ad - bc}\begin{pmatrix} d & -b \\ -c & a \end{pmatrix}$$

$$\left(\vec{X}^T\vec{X}\right)^{-1} = \frac{1}{n\sum x_i^2 - (\sum x_i)^2}\begin{pmatrix} \sum x_i^2 & -\sum x_i \\ -\sum x_i & n \end{pmatrix}$$

(b) Show that the vector $\hat{\vec{\beta}} = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\vec{Y}$ provides the usual least squares estimates of the intercept and the slope.

Using the solution from part A, we know the OLS condition is minimizing:

$$\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 = ||\vec{Y} - \vec{X}\vec{\beta}||^2$$

where $||\cdot||$ denotes the length of the vector

$$SSR = \left(\vec{Y} - \vec{X}\vec{\beta}\right)^T\left(\vec{Y} - \vec{X}\vec{\beta}\right)$$

$$= \vec{Y}^T\vec{Y} - 2\vec{Y}^T\vec{X}\vec{\beta} + \vec{\beta}^T\vec{X}^T\vec{X}\vec{\beta}$$

We are trying to minimize $\vec{\beta}$ with respect to $\vec{\beta}$ in OLS, so let's take the partial derivative and solve for 0.

$$\frac{\partial}{\partial\vec{\beta}} = 0 - 2\vec{X}^T\vec{Y} + 2\vec{X}^T\vec{X}\vec{\beta}$$

Solve for $\vec{\beta}$:

$$\vec{X}^T\vec{X}\vec{\beta} = \vec{X}^T\vec{Y}$$

Left multiply both sides by $\left(\vec{X}^T\vec{X}\right)^{-1}$

$$\vec{\beta} = \left(\vec{X}^T\vec{X}\right)^{-1}\vec{X}^T\vec{Y}$$

This is our least squares estimate for $\vec{\beta}$ in this model.

(c) Show that the diagonal elements of the 2 x 2 matrix $\sigma^2(\mathbf{X}^T\mathbf{X})^{-1}$ provide the usual variances of the least squares estimates of the intercept and the slope for a simple linear regression.

Our usual variances of the least squares estimates of the $\vec{\beta}$ in OLS are:

$$\hat{\beta}_0 \sim N\left(\beta_0, \sigma^2\left(\frac{1}{n} - \frac{\bar{x}^2}{S_{xx}}\right)\right)$$

$$\hat{\beta}_1 \sim N\left(\beta_1, \frac{\sigma^2}{S_{xx}}\right)$$

$$\text{if } \epsilon_i \sim N(0,\sigma^2) \text{ and } S_{xx} = \sum_{i=1}^{n}(x_i - \bar{x})^2$$

Show that the variances are:

$$\frac{\sigma^2}{S_{xx}} \text{ and } \sigma^2\left(\frac{1}{n} + \frac{\bar{x}^2}{S_{xx}}\right)$$

for the diagonal elements of $\sigma^2\left(\vec{X}^T\vec{X}\right)^{-1}$

$$\left(\vec{X}^T\vec{X}\right)^{-1} = \frac{1}{n\sum x_i^2 - \left(\sum x_i\right)^2}\begin{pmatrix} n & -\sum x_i \\ -\sum x_i & \sum x_i^2 \end{pmatrix}$$

Simplifying and isolating the diagonals:

$$a_{11} = \frac{\sigma^2\sum x_i^2}{n\sum x_i^2 - \left(\sum x_i\right)^2}$$

$$a_{22} = \frac{\sigma^2 n}{n\sum x_i^2 - \left(\sum x_i\right)^2} = \frac{\sigma^2}{S_{xx}}$$

Solve for a_{22} to show that:

$$n\sum x_i^2 - \left(\sum x_i\right)^2 = \sum_{i=1}^{n}(x_i - \bar{x})^2$$

$$\sum x_i^2 - \frac{1}{n}\left(\sum x_i\right)^2 = S_{xx}$$

This is the variance of $\hat{\beta}_1$

$$a_{11} = \frac{\sigma^2 n}{n\sum x_i^2 - \left(\sum x_i\right)^2}$$

$$a_{11} = \frac{\sigma^2}{nS_{xx}}$$

$$S_{xx} = \sum (x_i - \bar{x})^2 = \sum x_i^2 - \frac{1}{n}\left(\sum x_i\right)^2$$

$$\text{Thus } a_{11} = \sigma^2\left(\frac{1}{n} + \frac{\bar{x}^2}{S_{xx}}\right)$$

Thus the diagonals equal the variance.

(d) A second predictor is being considered for inclusion in the model (X_2). Under what conditions will its presence in the model have no effect on the estimates of β_0 and β_1?

d) Under this model, we know

$$\beta = \begin{bmatrix}\beta_0 \\ \beta_1 \\ \beta_2\end{bmatrix}$$

This gives us a new design matrix:

$$\mathbf{X} = \begin{bmatrix}1 & X_{11} & X_{21} \\ 1 & X_{12} & X_{22} \\ \vdots & \vdots & \vdots \\ 1 & X_{1n} & X_{2n}\end{bmatrix}$$

Intuitively, if X_2

is correlated with existing predictors or the response variable, it will impact the value of the other predictors.

For X_2 to have no impact on β_0 and β_1, it must be orthogonal to the other vectors.

This means

$$\mathbf{X}^T X_2 = \mathbf{0}$$

This means that $\sum X_2 = 0$ because the first column in $\mathbf{X}$ is all 1s.

This shows that it will not be correlated with the other predictors.

Now, we need to show that it is uncorrelated with $\hat{y}$.

$$\text{Thus: } \quad \text{Corr}\left[\hat{y}, \mathbf{X}\middle| X_1\right] = 0$$

Thus, the conditions are:

$$\text{Corr}\left[\hat{y}, \mathbf{X}\middle| X_1\right] = 0$$

$$\mathbf{X}^T X_2 = 0$$

$$\sum_{i=1}^{n} X_{2i} = 0$$

Question 2

In this problem you will code up your own linear regression calculations in R. In parts (a)-(g) you will attempt to replicate the results of `lm()` using the `GaltonFamilies` dataset in the `HistData` package. Specifically, in Lecture 7, we fit the following model:

$$\vec{Y} = \beta_0 + \beta_1\vec{x}_1 + \beta_2\vec{x}_2 + \beta_3\vec{x}_3 + \vec{\varepsilon} \qquad \vec{\varepsilon} \sim \mathcal{N}(\vec{0}, \sigma^2 I)$$

where:

Y = child height

x_1 = midparent height = $\frac{\text{father height} + 1.08 \times \text{mother height}}{2}$

x_2 = an indicator of male gender

x_3 = birth order

(a) Start by fitting the model in R with `lm()` and saving the output as an object named `lm.out`.

```
if (!requireNamespace("HistData", quietly = TRUE)) {
  install.packages("HistData")
}

if (!requireNamespace("dplyr", quietly = TRUE)) {
  install.packages("dplyr")
}
```

```
library(HistData)

# Prepare the data
data(GaltonFamilies)
GaltonFamilies$gender <- ifelse(GaltonFamilies$gender == "male", 1, 0)
GaltonFamilies$midparentHeight <- (GaltonFamilies$father + 1.08 * GaltonFamilies$mother) / 2

# Fit the linear regression model
lm.out <- lm(childHeight ~ midparentHeight + gender + childNum, data = GaltonFamilies)

# Display the summary of the model
summary(lm.out)
```

```
##
## Call:
## lm(formula = childHeight ~ midparentHeight + gender + childNum,
##     data = GaltonFamilies)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -9.2057 -1.3216  0.1892  1.2687  8.6431
##
## Coefficients:
```

```
##                 Estimate Std. Error t value Pr(>|t|)
## (Intercept)     22.03832    2.59276    8.50   <2e-16 ***
## midparentHeight  0.63684    0.03704   17.20   <2e-16 ***
## gender           4.08629    0.16339   25.01   <2e-16 ***
## childNum        -0.41004    0.03467  -11.83   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2.024 on 930 degrees of freedom
## Multiple R-squared:  0.6812, Adjusted R-squared:  0.6801
## F-statistic: 662.3 on 3 and 930 DF,  p-value: < 2.2e-16
```

(b) Use matrix algebra in R to manually compute the least squares estimates of the β coefficients. Extract the coefficients from your `lm()` object and show that your estimates match these exactly.

```
X <- model.matrix(lm.out)
y <- GaltonFamilies$childHeight

beta_hat <- solve(t(X) %*% X) %*% t(X) %*% y

lm_coef <- coef(lm.out)

comparison <- data.frame(
  Manual = beta_hat,
  LM = lm_coef,
  Difference = beta_hat - lm_coef
)

# Display the results
print(comparison)
```

```
##                     Manual         LM    Difference
## (Intercept)     22.0383196 22.0383196  3.693401e-11
## midparentHeight  0.6368378  0.6368378 -3.895773e-13
## gender           4.0862864  4.0862864 -1.127987e-13
## childNum        -0.4100434 -0.4100434 -3.458345e-14
```

```
all.equal(as.numeric(beta_hat), as.numeric(lm_coef))
```

```
## [1] TRUE
```

(c) Compute the residual standard error using matrix algebra in R. Extract the estimate from your `lm()` object and show that they match exactly.

```
X <- model.matrix(lm.out)
y <- GaltonFamilies$childHeight

y_hat <- X %*% beta_hat

residuals <- y - y_hat
```

```r
RSS <- sum(residuals^2)

n <- nrow(X)  # number of observations
p <- ncol(X)  # number of parameters (including intercept)
df <- n - p   # degrees of freedom

RSE_manual <- sqrt(RSS / df)

RSE_lm <- summary(lm.out)$sigma

cat("Manually computed RSE:", RSE_manual, "\n")
```

```
## Manually computed RSE: 2.024301
```

```r
cat("lm() computed RSE:", RSE_lm, "\n")
```

```
## lm() computed RSE: 2.024301
```

```r
cat("Difference:", RSE_manual - RSE_lm, "\n")
```

```
## Difference: -3.996803e-15
```

```r
all.equal(RSE_manual, RSE_lm)
```

```
## [1] TRUE
```

(d) Use matrix algebra in R to manually compute the variance-covariance matrix of your β estimates. Extract the variance-covariance matrix from your `lm()` object and show that your version matches R's exactly.

```r
X <- model.matrix(lm.out)

X_tX_inv <- solve(t(X) %*% X)

var_cov_manual <- (RSE_manual^2) * X_tX_inv

var_cov_lm <- vcov(lm.out)

cat("Manually computed variance-covariance matrix:\n")
```

```
## Manually computed variance-covariance matrix:
```

```r
print(var_cov_manual)
```

```
##                (Intercept) midparentHeight        gender      childNum
## (Intercept)     6.72238246   -0.0957468344 -0.0652304968 -0.0161958369
## midparentHeight -0.09574683    0.0013716079  0.0005724215  0.0001471095
## gender          -0.06523050    0.0005724215  0.0266966323  0.0033094587
## childNum        -0.01619584    0.0001471095  0.0033094587  0.0012021605
```

```r
cat("\nlm() computed variance-covariance matrix:\n")
```

```
##
## lm() computed variance-covariance matrix:
```

```
print(var_cov_lm)
```

```
##                 (Intercept) midparentHeight        gender      childNum
## (Intercept)      6.72238246   -0.0957468344 -0.0652304968 -0.0161958369
## midparentHeight -0.09574683    0.0013716079  0.0005724215  0.0001471095
## gender          -0.06523050    0.0005724215  0.0266966323  0.0033094587
## childNum        -0.01619584    0.0001471095  0.0033094587  0.0012021605
```

```
cat("\nDifference:\n")
```

```
##
## Difference:
```

```
print(var_cov_manual - var_cov_lm)
```

```
##                   (Intercept) midparentHeight        gender      childNum
## (Intercept)     -2.618350e-11    3.748529e-13  1.681433e-13  4.223011e-14
## midparentHeight  3.748113e-13   -5.365717e-15 -2.406278e-15 -6.041446e-16
## gender           1.703498e-13   -2.437829e-15 -1.172673e-15 -2.862294e-16
## childNum         4.275053e-14   -6.116527e-16 -2.849283e-16 -7.480995e-17
```

```
all.equal(as.vector(var_cov_manual), as.vector(var_cov_lm))
```

```
## [1] TRUE
```

(e) Manually recreate the table given by `summary(lm.out)$coefficients`. That is, in addition to the β coefficients you computed above, compute the standard errors, t-statistics and p-values, and organize the data in an attractive tabular format that exactly matches the results given by R.

```
# Part (e): Manually recreate the summary table

# Extract standard errors from variance-covariance matrix
std_errors <- sqrt(diag(var_cov_manual))

# Calculate t-statistics
t_values <- beta_hat / std_errors

# Calculate p-values
p_values <- 2 * pt(-abs(t_values), df = df)

# Create the manual coefficients table
coef_table_manual <- data.frame(
  Estimate = beta_hat,
  `Std. Error` = std_errors,
  `t value` = t_values,
  `Pr(>|t|)` = p_values,
  check.names = FALSE
)
```

```r
# Assign row names to match lm() output
rownames(coef_table_manual) <- names(lm_coef)

# Define the p-value formatting function
format_pval <- function(p) {
  if (is.na(p) || !is.numeric(p)) return(NA)
  if (p < 2.2e-16) return("< 2e-16")  # Adjusted to match R's minimal p-value
  if (p < 0.001) return("< 0.001")
  if (p < 0.1) return(sprintf("%.3f", p))
  return(sprintf("%.2f", p))
}

# Apply formatting to p-values
coef_table_manual$`Pr(>|t|)` <- sapply(coef_table_manual$`Pr(>|t|)`, format_pval)

# Display manual and lm() coefficient tables
cat("Manual Coefficient Table:\n")
```

```
## Manual Coefficient Table:
```

```r
print(coef_table_manual)
```

```
##                   Estimate Std. Error   t value Pr(>|t|)
## (Intercept)     22.0383196 2.59275577   8.49996  < 2e-16
## midparentHeight  0.6368378 0.03703523  17.19546  < 2e-16
## gender           4.0862864 0.16339104  25.00924  < 2e-16
## childNum        -0.4100434 0.03467219 -11.82629  < 2e-16
```

```r
cat("\nlm() Coefficient Table:\n")
```

```
##
## lm() Coefficient Table:
```

```r
print(summary(lm.out)$coefficients)
```

```
##                   Estimate Std. Error   t value      Pr(>|t|)
## (Intercept)     22.0383196 2.59275577   8.49996  7.476310e-17
## midparentHeight  0.6368378 0.03703523  17.19546  9.407542e-58
## gender           4.0862864 0.16339104  25.00924 5.554353e-106
## childNum        -0.4100434 0.03467219 -11.82629  3.661584e-30
```

```r
# Check numerical equivalence (excluding formatted p-values)
manual_numeric <- coef_table_manual[, 1:3]
lm_numeric <- summary(lm.out)$coefficients[, 1:3]
cat("\nNumerical comparison of Estimate, Std. Error, and t value:\n")
```

```
##
## Numerical comparison of Estimate, Std. Error, and t value:
```

```r
print(all.equal(as.matrix(manual_numeric), as.matrix(lm_numeric)))
```

```
## [1] TRUE
```

```
# Compare p-values (after removing formatting for numerical comparison)
manual_p_values <- ifelse(coef_table_manual$`Pr(>|t|)` == "< 2e-16", 0, as.numeric(coef_table_m
r_p_values <- summary(lm.out)$coefficients[, 4]
cat("\nP-value comparison:\n")
```

```
##
## P-value comparison:
```

```
print(data.frame(Manual = manual_p_values, R = r_p_values))
```

```
##                  Manual             R
## (Intercept)           0  7.476310e-17
## midparentHeight       0  9.407542e-58
## gender                0 5.554353e-106
## childNum              0  3.661584e-30
```

```
print(all.equal(manual_p_values, r_p_values))
```

```
## [1] "names for current but not for target"
```

(f) Manually construct a 95% confidence interval for the average height of sons born to 5'8'' fathers and 5'5'' mothers who have 3 older siblings. Show that your interval exactly matches that given by the `predict()` function in R.

$$\text{Father's height} = 68\,\text{in}$$

$$\text{Mother's height} = 65\,\text{in}$$

$$\text{mid parent} = 69.1\,\text{in}$$

$$\text{gender } X_2 = 1 \text{ (b/c sons)}$$

$$\text{Birth order } X_3 = 4 \text{ (given there are 3 other siblings)}$$

Design vector

$$\vec{X} = \Big[1, \text{midparent}, \text{gender}, \text{birth order}\Big] = \Big[1, 69.1, 1, 4\Big]$$

Thus

$$\hat{Y} = \vec{X}\vec{\beta} = \beta_0 + 69.1\beta_1 + \beta_2 + 4\beta_3$$

Standard error

$$SE = \sqrt{\vec{X}^T \left(\vec{X}^T \vec{X}\right)^{-1} \vec{X} RSE}$$

$RSE =$ residual standard error

$t^*[\text{critical value}] = t_{.975,df}$

Thus

$$\hat{Y} \pm t^* \cdot SE$$

Now for the R component:

```r
# Part (f): Manually construct a 95% confidence interval

# Ensure previous parts (a)-(e) have been executed, so variables like lm.out,
# beta_hat, var_cov_manual, and RSE_manual are available.

# Step 1: Define the new data point
father_height_inches <- 5 * 12 + 8  # 5'8'' -> 68 inches
mother_height_inches <- 5 * 12 + 5  # 5'5'' -> 65 inches

midparentHeight_new <- (father_height_inches + 1.08 * mother_height_inches) / 2
gender_new <- 1  # Male
childNum_new <- 4  # 3 older siblings imply child number 4

# Display the new predictor values
cat("New Data Point:\n")
```

```
## New Data Point:
```

```r
cat("Father's Height (in):", father_height_inches, "\n")
```

```
## Father's Height (in): 68
```

```r
cat("Mother's Height (in):", mother_height_inches, "\n")
```

```
## Mother's Height (in): 65
```

```r
cat("Midparent Height:", midparentHeight_new, "\n")
```

```
## Midparent Height: 69.1
```

```r
cat("Gender (Male=1):", gender_new, "\n")
```

```
## Gender (Male=1): 1
```

```r
cat("Child Number:", childNum_new, "\n\n")
```

```
## Child Number: 4
```

```r
# Step 2: Create the design vector with intercept
X_new <- c(1, midparentHeight_new, gender_new, childNum_new)

# Step 3: Calculate the predicted child height
y_hat <- as.numeric(X_new %*% beta_hat)

cat("Predicted Child Height (Manual Calculation):", y_hat, "inches\n\n")
```

```
## Predicted Child Height (Manual Calculation): 68.48993 inches
```

```r
# Step 4: Calculate the standard error of the prediction
# var_cov_manual is (RSE^2) * (X'X)^-1
# SE = sqrt(X_new %*% var_cov_manual %*% X_new)
SE_y_hat <- sqrt(as.numeric(X_new %*% var_cov_manual %*% X_new))

# Step 5: Determine the critical t-value for 95% confidence
alpha <- 0.05
df <- n - p  # Degrees of freedom from previous parts
t_critical <- qt(1 - alpha/2, df)

cat("Standard Error of Prediction:", SE_y_hat, "\n")
```

```
## Standard Error of Prediction: 0.1102145
```

```r
cat("Critical t-value for 95% CI:", t_critical, "\n\n")
```

```
## Critical t-value for 95% CI: 1.962518
```

```r
# Step 6: Construct the 95% confidence interval
lower_bound <- y_hat - t_critical * SE_y_hat
upper_bound <- y_hat + t_critical * SE_y_hat

cat("Manual 95% Confidence Interval: [", lower_bound, ", ", upper_bound, "] inches\n\n")
```

```
## Manual 95% Confidence Interval: [ 68.27363 ,  68.70622 ] inches
```

```r
# Step 7: Use R's predict() function to obtain the confidence interval
# Create a new data frame matching the model's predictors
new_data <- data.frame(
  midparentHeight = midparentHeight_new,
  gender = gender_new,
  childNum = childNum_new
)
```

```r
# Obtain the confidence interval using predict()
predict_lm <- predict(lm.out, newdata = new_data, interval = "confidence", level = 0.95)

cat("R's predict() Function 95% Confidence Interval:\n")
```

```
## R's predict() Function 95% Confidence Interval:
```

```r
print(predict_lm)
```

```
##        fit      lwr      upr
## 1 68.48993 68.27363 68.70622
```

```r
# Step 8: Compare Manual and predict() Intervals
comparison <- data.frame(
  Manual_Lower = lower_bound,
  Predict_Lower = predict_lm[,"lwr"],
  Manual_Upper = upper_bound,
  Predict_Upper = predict_lm[,"upr"]
)

cat("\nComparison of Manual and predict() Confidence Intervals:\n")
```

```
##
## Comparison of Manual and predict() Confidence Intervals:
```

```r
print(comparison)
```

```
##   Manual_Lower Predict_Lower Manual_Upper Predict_Upper
## 1     68.27363      68.27363     68.70622      68.70622
```

```r
# Verify that the intervals match (allowing for minor numerical differences)
all_close <- all.equal(
  c(lower_bound, upper_bound),
  c(predict_lm[,"lwr"], predict_lm[,"upr"]),
  tolerance = 1e-8
)

if (isTRUE(all_close)) {
  cat("\nThe manually constructed confidence interval matches the predict() function's interval
} else {
  cat("\nThere are differences between the manual interval and predict() function's interval.\n
  print(all_close)
}
```

```
##
## The manually constructed confidence interval matches the predict() function's interval exact
```

(g) Manually construct a prediction interval for a son born to a 5'8'' father and a 5'5'' mother who has 3 older siblings. Show that your interval exactly matches that given by the `predict()` function R.

Use the same process for hte handwritten work as in F

Now for the R component,

```
# Part (g): Manually construct a 95% prediction interval

# Ensure previous parts (a)-(e) have been executed, so variables like lm.out,
# beta_hat, var_cov_manual, and RSE_manual are available.

# Step 1: Define the new data point
father_height_inches <- 5 * 12 + 8  # 5'8'' -> 68 inches
mother_height_inches <- 5 * 12 + 5  # 5'5'' -> 65 inches

midparentHeight_new <- (father_height_inches + 1.08 * mother_height_inches) / 2
gender_new <- 1  # Male
childNum_new <- 4  # 3 older siblings imply child number 4

# Display the new predictor values
cat("New Data Point:\n")
```

```
## New Data Point:
```

```
cat("Father's Height (in):", father_height_inches, "\n")
```

```
## Father's Height (in): 68
```

```
cat("Mother's Height (in):", mother_height_inches, "\n")
```

```
## Mother's Height (in): 65
```

```
cat("Midparent Height:", midparentHeight_new, "\n")
```

```
## Midparent Height: 69.1
```

```
cat("Gender (Male=1):", gender_new, "\n")
```

```
## Gender (Male=1): 1
```

```
cat("Child Number:", childNum_new, "\n\n")
```

```
## Child Number: 4
```

```
# Step 2: Create the design vector with intercept
X_new <- c(1, midparentHeight_new, gender_new, childNum_new)

# Step 3: Calculate the predicted child height
y_hat <- as.numeric(X_new %*% beta_hat)

cat("Predicted Child Height (Manual Calculation):", y_hat, "inches\n\n")
```

```
## Predicted Child Height (Manual Calculation): 68.48993 inches
```

```
# Step 4: Calculate the standard error of prediction
# For prediction interval, SE_pred = sqrt(X_new %*% (X'X)^-1 %*% X_new * RSE^2 + RSE^2)
```

```r
# Since var_cov_manual = (RSE^2) * (X'X)^-1, we can compute as follows:

SE_mean <- sqrt(as.numeric(X_new %*% var_cov_manual %*% X_new))
SE_pred <- sqrt(SE_mean^2 + RSE_manual^2)

# Alternatively, combine steps:
# SE_pred <- RSE_manual * sqrt(1 + X_new %*% (X'X')^-1 %*% X_new)

cat("Standard Error of Prediction:", SE_pred, "\n\n")
```

```
## Standard Error of Prediction: 2.027299
```

```r
# Step 5: Determine the critical t-value for 95% prediction interval
alpha <- 0.05
df <- n - p  # Degrees of freedom from previous parts
t_critical <- qt(1 - alpha/2, df)

cat("Critical t-value for 95% Prediction Interval:", t_critical, "\n\n")
```

```
## Critical t-value for 95% Prediction Interval: 1.962518
```

```r
# Step 6: Construct the 95% prediction interval
lower_bound_pred <- y_hat - t_critical * SE_pred
upper_bound_pred <- y_hat + t_critical * SE_pred

cat("Manual 95% Prediction Interval: [", lower_bound_pred, ", ", upper_bound_pred, "] inches\n"
```

```
## Manual 95% Prediction Interval: [ 64.51132 ,  72.46854 ] inches
```

```r
# Step 7: Use R's predict() function to obtain the prediction interval
# Create a new data frame matching the model's predictors
new_data <- data.frame(
  midparentHeight = midparentHeight_new,
  gender = gender_new,
  childNum = childNum_new
)

# Obtain the prediction interval using predict()
predict_lm_pred <- predict(lm.out, newdata = new_data, interval = "prediction", level = 0.95)

cat("R's predict() Function 95% Prediction Interval:\n")
```

```
## R's predict() Function 95% Prediction Interval:
```

```r
print(predict_lm_pred)
```

```
##        fit      lwr      upr
## 1 68.48993 64.51132 72.46854
```

```r
# Step 8: Compare Manual and predict() Prediction Intervals
comparison_pred <- data.frame(
```

```
  Manual_Lower = lower_bound_pred,
  Predict_Lower = predict_lm_pred[,"lwr"],
  Manual_Upper = upper_bound_pred,
  Predict_Upper = predict_lm_pred[,"upr"]
)

cat("\nComparison of Manual and predict() Prediction Intervals:\n")
```

```
##
## Comparison of Manual and predict() Prediction Intervals:
```

```
print(comparison_pred)
```

```
##   Manual_Lower Predict_Lower Manual_Upper Predict_Upper
## 1     64.51132      64.51132     72.46854      72.46854
```

```
# Step 9: Verify that the intervals match (allowing for minor numerical differences)
all_close_pred <- all.equal(
  c(lower_bound_pred, upper_bound_pred),
  c(predict_lm_pred[,"lwr"], predict_lm_pred[,"upr"]),
  tolerance = 1e-8
)

if (isTRUE(all_close_pred)) {
  cat("\nThe manually constructed prediction interval matches the predict() function's interval
} else {
  cat("\nThere are differences between the manual prediction interval and predict() function's
  print(all_close_pred)
}
```

```
##
## The manually constructed prediction interval matches the predict() function's interval exact
```

(h) Interpret the intervals in parts (f) and (g).

The confidence interval estimates the range within which the average height of all sons with specified parental heights and birth order is expected to lie, reflecting uncertainty around the mean estimate. In contrast, the prediction interval provides a broader range where the height of an individual son with those characteristics is likely to fall, accounting for both the uncertainty in the mean prediction and the natural variability of individual outcomes. **(i)** Conduct a formal hypothesis test at the $\alpha = 0.05$ level of whether sons born to 5'8'' fathers and 5'5'' mothers who have 3 older siblings are taller on average than daughters born to 6'0'' fathers and 5'10'' mothers who have no older siblings.

(i) Conduct a formal hypothesis test at the $\alpha = 0.05$ level of whether sons born to 5'8'' fathers and 5'5'' mothers who have 3 older siblings are taller on average than daughters born to 6'0'' fathers and 5'10'' mothers who have no older siblings.

Definitions: Let group 1 be the sons born to 5'8'' fathers and 5'5'' mothers who have 3 older siblings Let group 2 be the daughters born to 6'0'' fathers and 5'10'' mothers who have no older siblings

Hypotheses: H_0: The average height of group 1 (sons) is less than or equal to the average height of

group 2 (daughters)

$$\mu_1 \leq \mu_2$$

H_1: The average height of group 1 (sons) is greater than the average height of group 2 (daughters)

$$\mu_1 > \mu_2$$

Groupings: Group 1: Father's height = 68 mother's height = 65 gender indicator = 1 birth order = 4 (3 plus 1) midparent height: 69.1 in

Group 2: Father's height = 72 mother's height = 70 gender indicator = 0 birth order = 1 midparent height: 73.8 in

Design vectors:

$$\vec{X}_1 = \begin{bmatrix} 1 & 69.1 & 1 & 4 \end{bmatrix}$$

$$\vec{X}_2 = \begin{bmatrix} 0 & 73.8 & 0 & 1 \end{bmatrix}$$

Define $\vec{X}_{diff}$ as:

$$\vec{X}_{diff} = \vec{X}_1 - \vec{X}_2 = \begin{bmatrix} 1 & -4.7 & 1 & 3 \end{bmatrix}$$

Now, we can define our linear model as:

$$\vec{Y} = \vec{X}_{diff}\vec{\beta}$$

Calculate the SE of the difference:

$$SE_{diff} = \sqrt{\vec{X}_{diff}(\vec{X}^T\vec{X})^{-1}\vec{X}_{diff}^T} * RSE$$

and our T stat is :

$$t^* = \frac{\vec{\hat{Y}}}{SE_{diff}}$$

WE will reject $ H_0$ if $t > t_{\alpha,\text{df}}$

fail to reject otherwise

Now, for the R component (calculation!)

```r
# Step 1: Define the predictor values for Group 1 (Sons)
father_height1 <- 5 * 12 + 8    # 5'8'' = 68 inches
mother_height1 <- 5 * 12 + 5    # 5'5'' = 65 inches
midparent1 <- (father_height1 + 1.08 * mother_height1) / 2  # 69.1 inches
gender1 <- 1                     # Male
childNum1 <- 4                   # 3 older siblings

# Step 2: Define the predictor values for Group 2 (Daughters)
```

```r
father_height2 <- 6 * 12 + 0    # 6'0'' = 72 inches
mother_height2 <- 5 * 12 + 10   # 5'10'' = 70 inches
midparent2 <- (father_height2 + 1.08 * mother_height2) / 2  # 73.8 inches
gender2 <- 0                    # Female
childNum2 <- 1                  # No older siblings

# Step 3: Create the design vectors
X1 <- c(1, midparent1, gender1, childNum1)  # Group 1
X2 <- c(1, midparent2, gender2, childNum2)  # Group 2

# Step 4: Compute the difference in design vectors
X_diff <- X1 - X2  # [0, -4.7, 1, 3]

# Step 5: Compute the difference in predicted heights
delta_Y_hat <- sum(X_diff * beta_hat)

# Step 6: Compute the standard error of the difference
SE_delta <- sqrt(as.numeric(t(X_diff) %*% var_cov_manual %*% X_diff))

# Step 7: Compute the test statistic
t_stat <- delta_Y_hat / SE_delta

# Step 8: Determine the critical t-value for a one-sided test
alpha <- 0.05
df <- n - p
t_critical <- qt(1 - alpha, df)

# Step 9: Compute the p-value
p_value <- 1 - pt(t_stat, df)

# Step 10: Display the results
cat("Difference in Predicted Heights (Group1 - Group2):", delta_Y_hat, "inches\n")
```

```
## Difference in Predicted Heights (Group1 - Group2): -0.1369816 inches
```

```r
cat("Standard Error of the Difference:", SE_delta, "inches\n")
```

```
## Standard Error of the Difference: 0.2795396 inches
```

```r
cat("Test Statistic (t):", t_stat, "\n")
```

```
## Test Statistic (t): -0.4900257
```

```r
cat("Critical t-value (one-sided, alpha=0.05):", t_critical, "\n")
```

```
## Critical t-value (one-sided, alpha=0.05): 1.646494
```

```r
cat("P-value:", p_value, "\n\n")
```

```
## P-value: 0.6878844
```

```r
# Step 11: Decision
if (t_stat > t_critical) {
  cat("Result: Reject the null hypothesis. There is sufficient evidence to conclude that sons i
} else {
  cat("Result: Fail to reject the null hypothesis. There is insufficient evidence to conclude t
}
```

```
## Result: Fail to reject the null hypothesis. There is insufficient evidence to conclude that
```

based on the data, there is not strong enough evidence to reject the null hypothesis; thus we fail to reject the null hypothesis that group 1 is on average taller than group 2

Question 3

Prove the theorem on slide 10 of lecture 7 (given on Wednesday, September 25).

Problem:

Let $\hat{Y}_{n\times 1}$ be a vector with mean μ and covariance matrix Σ.

Let A be an $m \times n$ matrix and $b_{m\times 1}$ be a constant vector. Set $\hat{Z} = A\hat{Y} + b$.

Then: $E[\hat{Z}] = A\mu + b$,

$\text{Cov}[\hat{Z}] = A\Sigma A^T$. Prove that this is true.

Proof:

Proving expectation:

$$E[\hat{Z}] = E[A\hat{Y} + b]$$

$$= E[A\hat{Y}] + E[b]$$

$$= AE[\hat{Y}] + b$$

$$= A\mu + b$$

$$E[\hat{Z}] = A\mu + b$$

Proving for Covariance:

$$\text{Cov}[\hat{Z}] = E[(Z - E[\hat{Z}])(Z - E[\hat{Z}])^T]$$

$$= E[(A\hat{Y} + b - E[\hat{Z}])(A\hat{Y} + b - E[\hat{Z}])^T]$$

We previously established that $E[\hat{Z}] = A\mu + b$

$$= E[(A\hat{Y} + b - (A\mu + b))(A\hat{Y} + b - (A\mu + b))^T]$$

$$= E[A(\hat{Y} - \mu)]E[A(\hat{Y} - \mu)]^T$$

$$= E[A(\hat{Y} - \mu)(\hat{Y} - \mu)^T A^T]$$

$$= AE[(\hat{Y} - \mu)(\hat{Y} - \mu)^T]A^T$$

By definition of the covariance matrix,

$$\Sigma = E[(\hat{Y} - \mu)(\hat{Y} - \mu)^T]$$

Thus,

$$\text{Cov}[\hat{Z}] = A\Sigma A^T$$

Question 4

Prove the theorem on slide 11 of lecture 7 (given on Wednesday, September 25).

(4) Prove:

Suppose that $\vec{Y} \sim \mathcal{N}(\vec{\mu}, \Sigma)$ and set $\vec{Z} = A\vec{Y} + \vec{b}$,

then $\vec{Z} \sim \mathcal{N}(A\vec{\mu} + \vec{b}, A\Sigma A^T)$

Using the result of question 3,

$$\mathbb{E}[\vec{Z}] = A\vec{\mu} + \vec{b}$$

$$\text{Cov}(\vec{Z}) = A\Sigma A^T$$

Identify that $\vec{Y}$ is MVN with mean $\vec{\mu}$ and covariance matrix Σ.

Thus the parameters for the distribution of $\vec{Z}$ must be $\mathbb{E}[\vec{Z}]$ and $\text{Cov}(\vec{Z})$ from the previous problem.

Now we must show that it is also normally distributed (MVN).

We know from the Stat 110 book that:

A linear combination of normals is normally distributed.

Examining $\vec{Z}$, we can identify that $A\vec{Y}+\vec{b}$ is in fact a linear combination since A is a constant matrix and $\vec{b}$ is a constant vector. When we have a constant matrix, we can view it as a linear combination. The addition of b is an affine transformation and by definition does not change the colinearity, preserving the linear combination.

Thus $\vec{Z} \sim$ MVN.

Combining these ideas,

$$\vec{Z} \sim \mathcal{N}(A\vec{\mu} + \vec{b}, A\Sigma A^T)$$