Math 437 - Final Exam

Adrien Leon

Due May 17, 2023 at 2:00 PM

Exam Rules

1. Solutions to this exam must be uploaded to Canvas by 2:00 PM on Wednesday, May 17. **NO EXTENSIONS WILL BE GRANTED**. Do not ask for them.
2. You should submit *both* this .Rmd file with your solutions included *and* the knitted pdf file. If you are having trouble knitting, knit to HTML and print/save the html file to pdf.
3. **Exams must be done individually**. Do not confer with anyone other than me about the exam.
4. You may consult any reference materials available to you, including the required and recommended textbooks, your course notes/code you have written, and anything on Canvas (such as my solutions). You are allowed to search the Internet or the library, but provide citations for outside references used.
5. Approach the exam from the point of view of a statistician, not a student. There may be multiple valid approaches on a problem, and any valid approach will be accepted so long as it is accompanied with appropriate justification and/or explanation.

IN PARTICULAR: You can do this exam entirely, partially, or not at all using tidymodels packages.

6. I will provide a three-tiered hint system by e-mail:

- You may ask me to clarify something in a problem for no point penalty.
- You may give me pseudocode or buggy code, and ask me to implement/fix the code, for a partial point penalty (I will reply with how many points you will lose, then you confirm you want me to do it).
- You may ask me for my solution code, and I will provide it for a full point penalty (you will lose all Code points for that part). This may still be a good deal if you need the code to do the Explanation part or another part of the problem.

```
# Packages you might need
library(tidyverse)
library(tidymodels)
library(caret)
library(dplyr)
library(pROC)
library(caret)
library(randomForest)
library(MASS)

# Please load any other packages you need either all in this chunk
# or in the chunk you need it for
```

```r
# Data you need

# Regression Problem
GA_bangladesh <- readr::read_csv("GA_bangladesh.csv")
GA_zambia <- readr::read_csv("GA_zambia.csv")

# Classification Problem
drivers_train <- readr::read_csv("drivers_train.csv")
drivers_test <- readr::read_csv("drivers_test.csv")
```

This exam is worth a total of **105** points, but is graded out of **100** points.

Conceptual Problem (19 pts total)

This problem involves the article, "Validation of gestational age determination from ultrasound or a metabolic gestational age algorithm using exact date of conception in a cohort of newborns using assisted reproduction technologies" (Hawken et al., Am J Obstet Gynecol Glob Rep 2022;2:100091; doi: 10.1016/j.xagr.2022.100091).

The authors of this paper have developed machine learning models to predict gestational age (duration of pregnancy, abbreviated in the paper as GA) at birth based on demographic information and screening procedures available at birth. In pregnancies conceived with "assisted reproductive technologies" (techniques such as in vitro fertilization, abbreviated in the paper as ART), the gestational age can be directly calculated. In high-income countries, gestational age for other pregnancies can be estimated based on a first-trimester ultrasound. The authors thus wanted to compare the accuracy of GA estimates produced by their model to the accuracy of estimates obtained using ultrasound.

A pdf copy of the article has been included on Canvas, though it is not necessary to read the entire article to answer the questions in this problem.

Part a (2 pts)

The authors noted that, "Models were fit and all parameters estimated using the model training cohort, and model performance was evaluated in the... test cohorts, which were not used in the training of the GA estimation models."

Why was it important for the researchers to fit their model and estimate their parameters using the training cohort (training set) and evaluate their model on a separate test set?

It was important for the researchers to fit their model and estimate the parameters using the training cohort (training set) and evaluate the model on a separate test set for several reasons:

1. **Avoiding overfitting**: By using a separate test set that was not used during model training, the researchers could assess the performance of their model on new, unseen data. This approach helps to ensure that the model's performance is not biased or artificially inflated due to being overly specialized to the training data. Overfitting occurs when a model performs well on the training data but fails to generalize well to new data. Evaluating the model on a separate test set helps identify if the model has learned meaningful patterns or if it is simply memorizing the training data.
2. **Assessing generalization**: The researchers wanted to determine how well their model performs on new, independent data. The test cohort represents a real-world scenario where the model would be applied to predict gestational age in new cases. By evaluating the model on this separate test set, the researchers could obtain a realistic assessment of its performance in generalizing to unseen data. This evaluation indicates how well the model would perform when applied to new pregnancies.

3. **Ensuring model robustness**: By evaluating the model on a test set, the researchers could assess its robustness and reliability. They could identify any potential issues or limitations in the model's performance that may arise when applied to different cases or populations. This analysis helps ensure that the model's predictions are accurate and consistent across diverse scenarios.

In summary, by fitting the model and estimating parameters using the training cohort and evaluating the model on a separate test set, the researchers aimed to avoid overfitting, assess the model's generalization capabilities, and ensure its robustness when predicting gestational age in pregnancies conceived through assisted reproductive technologies.

Part b (4 pts)

The authors built an elastic net model to make their predictions. They explain the model by saying that, "Elastic net combines 2 types of penalization, blending ridge regression and LASSO-type penalties."

Briefly explain the difference between the ridge regression and LASSO penalty terms. What is suggested about the relationship between the response and predictor variables if the best elastic net uses entirely a ridge regression penalty? What about if it uses entirely a LASSO penalty?

Ridge regression and LASSO (Least Absolute Shrinkage and Selection Operator) are two regularization methods used in linear regression to avoid overfitting and improve model generalization.

Ridge Regression Penalty The ridge regression penalty term adds a penalty on the sum of squared coefficients (L2 penalty) to the objective function. This shrinks the coefficients towards zero but does not set them exactly to zero. Ridge regression can be useful when there are many correlated predictors in the model and some degree of shrinkage is desired.

LASSO Penalty The LASSO penalty term adds a penalty on the sum of absolute values of the coefficients (L1 penalty) to the objective function. This can lead to some coefficients being exactly set to zero, resulting in a more interpretable and potentially simpler model. LASSO can be useful when there are many predictors in the model and some degree of feature selection is desired.

If the best elastic net model uses entirely a ridge regression penalty, it suggests that there is a high degree of multicollinearity among the predictors. In this case, all predictors are considered important to some extent in predicting the response variable, but their coefficients are shrunk toward zero. Ridge regression helps distribute the impact of correlated predictors and avoids relying too heavily on any single predictor.

On the other hand, if the best elastic net model uses entirely a LASSO penalty, it indicates that only a subset of predictors is truly important in predicting the response variable. The LASSO penalty promotes sparsity by setting some coefficients exactly to zero, effectively performing feature selection. This implies that the relationship between the response and predictor variables is sparse, with only a few key predictors having a significant influence on the response.

In summary, a ridge regression penalty suggests a scenario with multicollinearity and all predictors playing a role, although with diminished coefficients. A LASSO penalty indicates a sparse relationship with only a subset of predictors being important, while the rest have negligible impact.

Part c (3 pts)

To evaluate the accuracy of the model, the authors "calculated 2 measures: the mean absolute error (MAE) and the square root of mean squared error (RMSE), which are both in the same units of the outcome (GA in weeks). Lower values of MAE and RMSE reflect higher accuracy. . . "

Without writing any code (pseudocode is okay), explain what the researchers would need to do to calculate RMSE on a test set. Are they correct in stating that a lower RMSE indicates a more accurate model? Why?

To calculate the Root Mean Squared Error (RMSE) on a test set without writing code, the researchers would follow these steps:

1. Obtain the predicted gestational age (GA) values from the machine learning model for each sample in the test set.
2. Collect the actual gestational age values for the corresponding samples in the test set.
3. Calculate the squared differences between the predicted and actual GA values for each sample.
4. Compute the mean of the squared differences.
5. Take the square root of the mean squared differences obtained in the previous step.

The researchers are correct in stating that a lower RMSE indicates a more accurate model. Here's why:

- RMSE measures the average magnitude of the prediction errors, providing an estimate of the model's overall performance. It reflects the deviation between the predicted GA values and the actual GA values in the test set.
- By squaring the errors, RMSE places more emphasis on larger errors while penalizing outliers. This is useful as it gives a higher weight to more significant deviations from the true values.
- Since RMSE is in the same units as the outcome variable (GA in weeks), it provides an interpretable measure of the model's accuracy.
- A lower RMSE indicates that the average prediction error is smaller, implying that the model's predictions are closer to the actual GA values. Therefore, a lower RMSE indicates a more accurate model.

In summary, the RMSE is calculated by measuring the average magnitude of prediction errors in the test set. A lower RMSE suggests that the model's predictions are more accurate, as they are closer to the true GA values.

Part d (3 pts)

After selecting a final model, they evaluated the performance of their model by calculating "95% bootstrap percentile confidence intervals (CIs) based on 500 bootstrap replicates for [mean absolute error and RMSE]."

Without writing any code (pseudocode is okay), explain how the authors could create such a confidence interval for test RMSE.

To create a confidence interval for the test RMSE without writing code, the authors could follow these steps:

1. Set the number of bootstrap replicates, in this case, 500.
2. Initialize an empty list or array to store the RMSE values obtained from each bootstrap replicate.
3. For each bootstrap replicate:
    a. Resample the test set with replacement to create a new bootstrap sample.
    b. Calculate the RMSE for the bootstrap sample using the selected model.
    c. Append the RMSE value to the list or array created in step 2.
4. Sort the list or array of RMSE values in ascending order.
5. Determine the lower and upper percentiles of the RMSE distribution based on the desired confidence level. In this case, the authors mentioned a 95% confidence interval.
    - For a 95% confidence interval, the lower percentile would be the 2.5th percentile (0.025) and the upper percentile would be the 97.5th percentile (0.975).
6. Obtain the RMSE values corresponding to the lower and upper percentiles, which represent the lower and upper bounds of the confidence interval, respectively.

The resulting range between the lower and upper bounds represents the 95% bootstrap percentile confidence interval for the test RMSE. This interval provides an estimate of the uncertainty around the RMSE value obtained from the selected model, considering the variability that could arise from different bootstrap samples.

Note: Creating a confidence interval using bootstrapping involves resampling the data and recalculating the RMSE multiple times to capture the distribution of possible RMSE values. This approach allows the authors to estimate the uncertainty associated with the model's performance on unseen data.

Part e (5 pts)

The authors evaluated their model on two different test sets: one included only "spontaneous conceptions" and the other included only "assisted reproductive technology" conceptions.

Without writing any code (pseudocode is okay), explain how the authors could use a *permutation test* procedure to determine whether there is a *statistically significant* difference in the probability of an infant being born prematurely between "spontaneous" and "assisted reproductive technology" conceptions. Include all steps of hypothesis testing and be as contextual as possible; e.g., don't just say "response variable," tell me what the response variable is.

HINT: There are multiple valid test statistics you could use, including some that we have perhaps first encountered in Math 437. I will accept any valid test statistic.

To determine whether there is a statistically significant difference in the probability of an infant being born prematurely between "spontaneous" and "assisted reproductive technology" (ART) conceptions using a permutation test procedure, the authors could follow these steps:

1. **Define the Hypotheses:**
    - Null Hypothesis (H0): There is no difference in the probability of an infant being born prematurely between spontaneous and ART conceptions.
    - Alternative Hypothesis (HA): There is a difference in the probability of an infant being born prematurely between spontaneous and ART conceptions.
2. **Identify the Response Variable:**
    - The response variable, in this case, is the occurrence of premature birth, which can be represented by a binary variable (e.g., 1 for premature birth, 0 for full-term birth).
3. **Group the Data:**
    - Separate the dataset into two groups: one containing spontaneous conceptions and the other containing ART conceptions.
    - Create two subsets of the response variable corresponding to premature birth for each group.
4. **Calculate the Observed Test Statistic:**
    - Choose a valid test statistic that measures the difference in the probability of premature birth between the two groups.
    - Calculate the observed test statistic using the original dataset, comparing the proportions of premature births in the spontaneous and ART conception groups.
5. **Perform Permutations:**
    - Pool the response variable values (premature births) from both groups.
    - Randomly shuffle the pooled response variable values, breaking their association with the group labels (spontaneous or ART).
    - Reassign the shuffled response variable values to the two groups, maintaining the original group sizes.
    - Recalculate the test statistic based on the permuted groups.
6. **Repeat the Permutations and Calculate the Test Statistic:**
    - Repeat steps 5 a large number of times (e.g., 10,000) to generate a null distribution of the test statistic under the assumption of no difference between the groups.

7. **Calculate the P-value:**
    - Determine the proportion of permuted test statistics that are as extreme or more extreme than the observed test statistic.
    - This proportion represents the p-value, which is the probability of observing a test statistic as extreme as or more extreme than the observed test statistic, assuming the null hypothesis is true.
8. **Compare the P-value to the Significance Level:**
    - Compare the obtained p-value to a pre-determined significance level (e.g., $\alpha = 0.05$) to make a decision.
    - If the p-value is less than the significance level, reject the null hypothesis and conclude that there is a statistically significant difference in the probability of premature birth between spontaneous and ART conceptions.
    - If the p-value is greater than or equal to the significance level, fail to reject the null hypothesis, indicating that there is not enough evidence to suggest a difference in the probability of premature birth between the two groups.

Note: The choice of the test statistic can vary depending on the specific measure used to quantify the difference in the probability of premature birth between the two groups (e.g., the difference in proportions, odds ratio, and chi-squared statistic). The pseudocode provided here outlines the general steps involved in a permutation test procedure for comparing two groups.

Part f (2 pts)

The authors cautioned that, "there were key limitations that may have affected the generalizability of [their] findings. First, women who undergo ART are different, at the population level, in age, fertility history, and socioeconomic status [than other women giving birth]. Second, people experiencing infertility who seek ART treatment are a self-selected population."

Why is it important to consider how the sample used to evaluate the model's performance may be different from the population the model will be applied to?

It is important to consider how the sample used to evaluate the model's performance may differ from the population the model will be applied to because the model's performance may vary when applied to different populations. Here are a few reasons why this is important:

1. **Generalizability:** The goal of developing a predictive model is often to apply it to a larger population beyond the specific sample used for model development and evaluation. If the sample used to evaluate the model differs significantly from the target population, the model's performance on the target population may not be as accurate or reliable. Therefore, understanding the limitations and potential differences between the sample and target population is crucial to assess the generalizability of the model's findings.
2. **External Validity:** External validity refers to the extent to which the findings of a study can be generalized to other populations or settings. When evaluating a model's performance, it is important to assess whether the observed results are specific to the sample used or if they can be expected to hold true in a broader context. Considering the differences between the sample and the target population helps determine the external validity of the model and whether the findings can be confidently extended beyond the sample.
3. **Bias and Representativeness:** The differences in demographic characteristics, fertility history, socioeconomic status, or other factors between the sample and the target population can introduce bias and affect the representativeness of the model's predictions. If the sample is not representative of the target population, the model may perform differently when applied to different subgroups or individuals with different characteristics. Biases may arise if the model disproportionately favors or performs poorly for certain groups, potentially leading to unfair or inaccurate predictions.

4. **Ethical Considerations:** Understanding the limitations of the sample and acknowledging the self-selection of individuals seeking assisted reproductive technologies (ART) treatment is crucial for ethical considerations. The model's predictions should be used with caution, especially if the target population differs significantly from the sample used. It is important to consider the potential consequences and implications of using the model's predictions in decision-making for individuals who may not share the same characteristics as the sample.

In summary, considering the differences between the sample used for model evaluation and the target population helps assess the generalizability, external validity, bias, and ethical implications of the model's predictions. It allows for a more nuanced understanding of how the model's performance may vary when applied to different populations and helps avoid unwarranted assumptions or biased outcomes.

Applied Problem 1 (40 points total)

In low- and middle-income countries, prenatal care such as ultrasound is often not available. Thus, the authors of the paper from the Conceptual Problem have suggested that their model can be used to accurately estimate gestational age (and thus detect, e.g., prematurely born infants) in these countries.

The `GA_bangladesh` and `GA_zambia` datasets contain information about 561 live births in the two countries, respectively (491 in Bangladesh, 70 in Zambia), recorded by the same research group. Use the `GA_bangladesh` dataset as your training set and the `GA_zambia` dataset as your test set; in other words, I have *already* done the training-test splitting for you. Please see the `GA_dictionary` file on Canvas for an explanation of each of the variables.

Just like the researchers, your goal on this problem is to predict the gestational age of the newborn (`gestage`). I do not expect you to achieve the accuracy demonstrated by the researchers; simply show me what you have learned in this class.

Part a (Code: 7 pts; Explanation: 14 pts)

Provide a *detailed* and *well-reasoned* description of your feature selection/engineering process. This process should be primarily based on exploratory data analysis; however, if you know things about evaluating newborn health, you can use that real-world knowledge to help inform your decisions. You are encouraged to create new features (e.g., by transforming predictor variables or creating an interaction term) if you can justify doing so.

As part of your exploratory data analysis, apply at least one of the following unsupervised learning techniques to the training set: principal component analysis, k-means clustering, model-based clustering (mclust); and visualize and interpret your results.

Before starting with the feature selection/engineering process, let's first check some basic summary statistics of the variables in the training set (`GA_bangladesh`) to get a better understanding of the data:

```
head(GA_bangladesh)
```

```
## # A tibble: 6 x 15
##   study_id gestage multiple bweight sex   fa_ratio   OHP   TSH    C5  C5DC  C4DC
##   <chr>      <dbl>    <dbl>   <dbl> <chr>    <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
## 1 20-1006~    39.7        0    2625 M        0.797  11.6  7.52  0.15  0.12  0.64
## 2 20-3004~    39.5        0    2770 M        0.768  21.1  6.14  0.14  0.09  0.92
## 3 20-3004~    39.2        0    2800 M        0.801  10.9  4.45  0.11  0.08  0.73
## 4 20-4004~    39.2        0    2435 F        0.828  13.9 11.3   0.15  0.09  0.79
## 5 20-1006~    39.7        0    2480 F        0.804  11.7  4.79  0.17  0.08  0.48
```

```
## 6 20-4003~    38.5        0    2910 M        0.776  12   12.5   0.27  0.12  0.85
## # i 4 more variables: Tyr <dbl>, TREC <dbl>, bio <dbl>, IRT <dbl>
```

```
summary(GA_bangladesh)
```

```
##    study_id            gestage         multiple          bweight
##  Length:491         Min.   :29.50   Min.   :0.00000   Min.   :1032
##  Class :character   1st Qu.:38.40   1st Qu.:0.00000   1st Qu.:2600
##  Mode  :character   Median :39.40   Median :0.00000   Median :2830
##                     Mean   :39.13   Mean   :0.01426   Mean   :2829
##                     3rd Qu.:40.10   3rd Qu.:0.00000   3rd Qu.:3088
##                     Max.   :41.80   Max.   :1.00000   Max.   :4430
##      sex               fa_ratio          OHP             TSH
##  Length:491         Min.   :0.5718   Min.   : 1.50   Min.   : 0.3053
##  Class :character   1st Qu.:0.7821   1st Qu.: 8.60   1st Qu.: 3.5047
##  Mode  :character   Median :0.8308   Median :12.90   Median : 5.5273
##                     Mean   :0.8257   Mean   :14.68   Mean   : 6.6119
##                     3rd Qu.:0.8708   3rd Qu.:18.45   3rd Qu.: 9.0940
##                     Max.   :0.9737   Max.   :72.30   Max.   :29.2136
##        C5              C5DC             C4DC             Tyr
##  Min.   :0.0400   Min.   :0.02000   Min.   :0.1900   Min.   : 41.11
##  1st Qu.:0.1200   1st Qu.:0.06000   1st Qu.:0.3800   1st Qu.: 64.83
##  Median :0.1500   Median :0.07000   Median :0.4700   Median : 78.36
##  Mean   :0.1670   Mean   :0.07656   Mean   :0.5055   Mean   : 84.53
##  3rd Qu.:0.2000   3rd Qu.:0.09000   3rd Qu.:0.6000   3rd Qu.: 96.79
##  Max.   :0.8367   Max.   :0.35000   Max.   :1.4400   Max.   :194.76
##       TREC              bio              IRT
##  Min.   :  24.69   Min.   : 38.90   Min.   : 3.715
##  1st Qu.: 353.33   1st Qu.: 66.86   1st Qu.:13.554
##  Median : 522.87   Median : 77.60   Median :18.588
##  Mean   : 621.40   Mean   : 79.79   Mean   :21.348
##  3rd Qu.: 796.44   3rd Qu.: 91.01   3rd Qu.:26.407
##  Max.   :3616.39   Max.   :140.24   Max.   :77.312
```

From the summary statistics, we can see that the variable 'study_id' is not needed as it is just a number listing the subject, so it has little meaning when analyzing our data for graphs. We can also see that for variables such as 'multiple' and 'sex', there are only 2 options for it to go, so we can consider them as categorical.

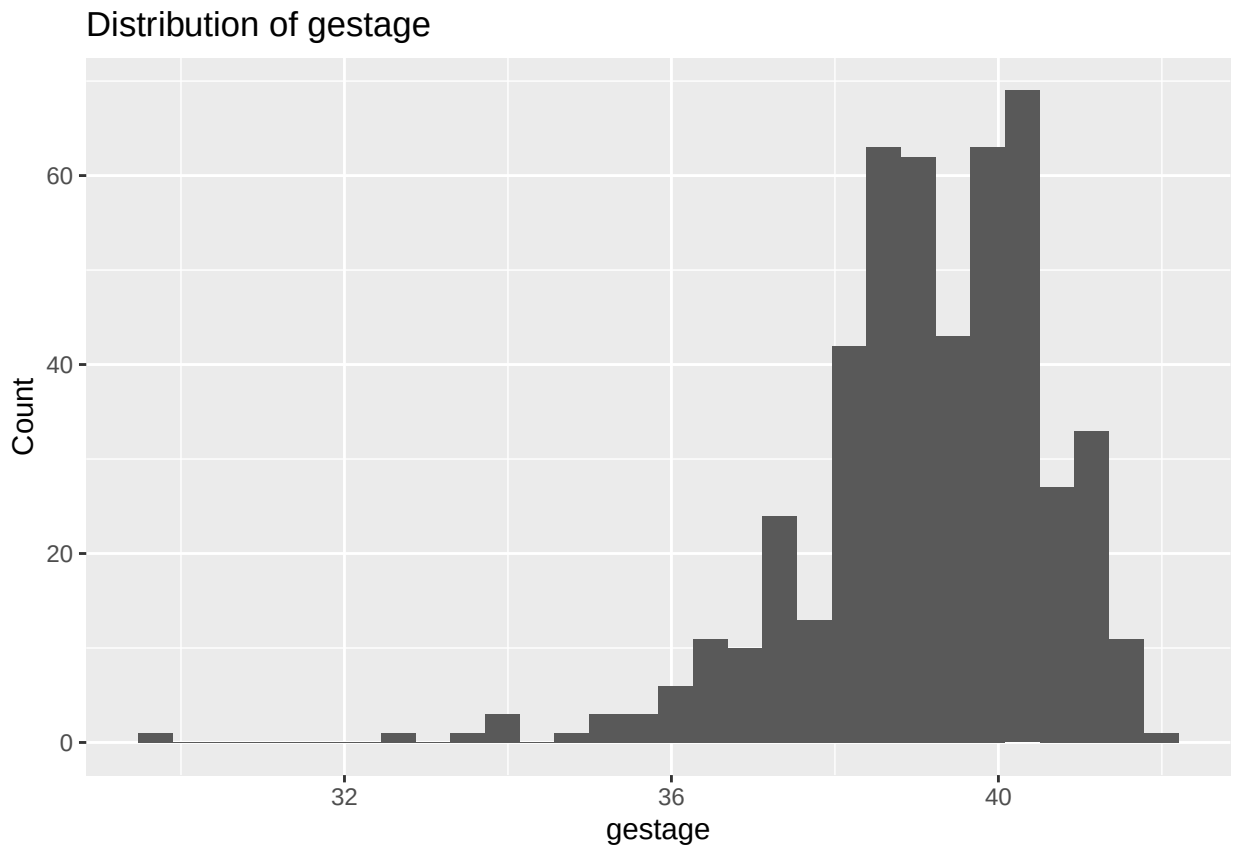
Univariate Analysis of Response Variable

```
# Summary statistics of gestage
summary(GA_bangladesh$gestage)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   29.50   38.40   39.40   39.13   40.10   41.80
```

The gestational age ranges from 29.5 to 41.8 with a mean of 39.13 and a median of 39.4.

```r
# Distribution of gestage
ggplot(GA_bangladesh, aes(x=gestage)) +
  geom_histogram(bins = 30) +
  labs(title="Distribution of gestage", x="gestage", y="Count")
```



The distribution of gestational age is left skewed where a majority of the observations lie around 38 to 41.

Univariate Analysis of Categorical Predictor Variable

From the scatterplot, we can see that there is a positive correlation between `bweight` and `gestage`. This is expected, as higher birth weight is generally associated with longer gestational age. We can also see in the second scatterplot that 'fa_ratio' and 'gestage' has a negative correlation.

We can also create bar charts of `gestage` against categorical variables like `sex` and `multiple` to see their proportions.

```r
# Proportion of Female to Male in the dataset
GA_bangladesh %>%
  count(sex) %>%
  mutate(prop = n/sum(n))
```

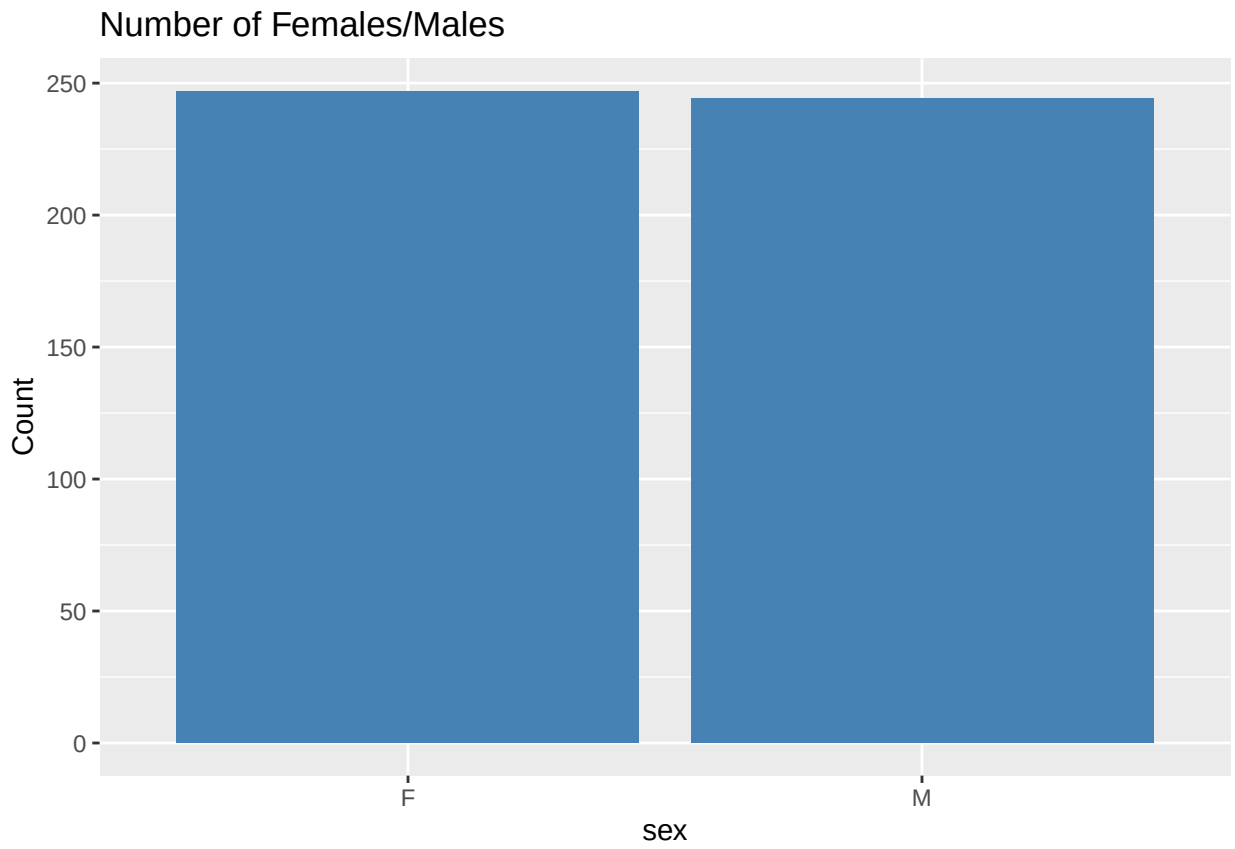
```
## # A tibble: 2 x 3
##   sex       n  prop
##   <chr> <int> <dbl>
## 1 F       247 0.503
## 2 M       244 0.497
```

```r
# Proportion of Multiples in the dataset
GA_bangladesh %>%
  count(multiple) %>%
  mutate(prop = n/sum(n))
```

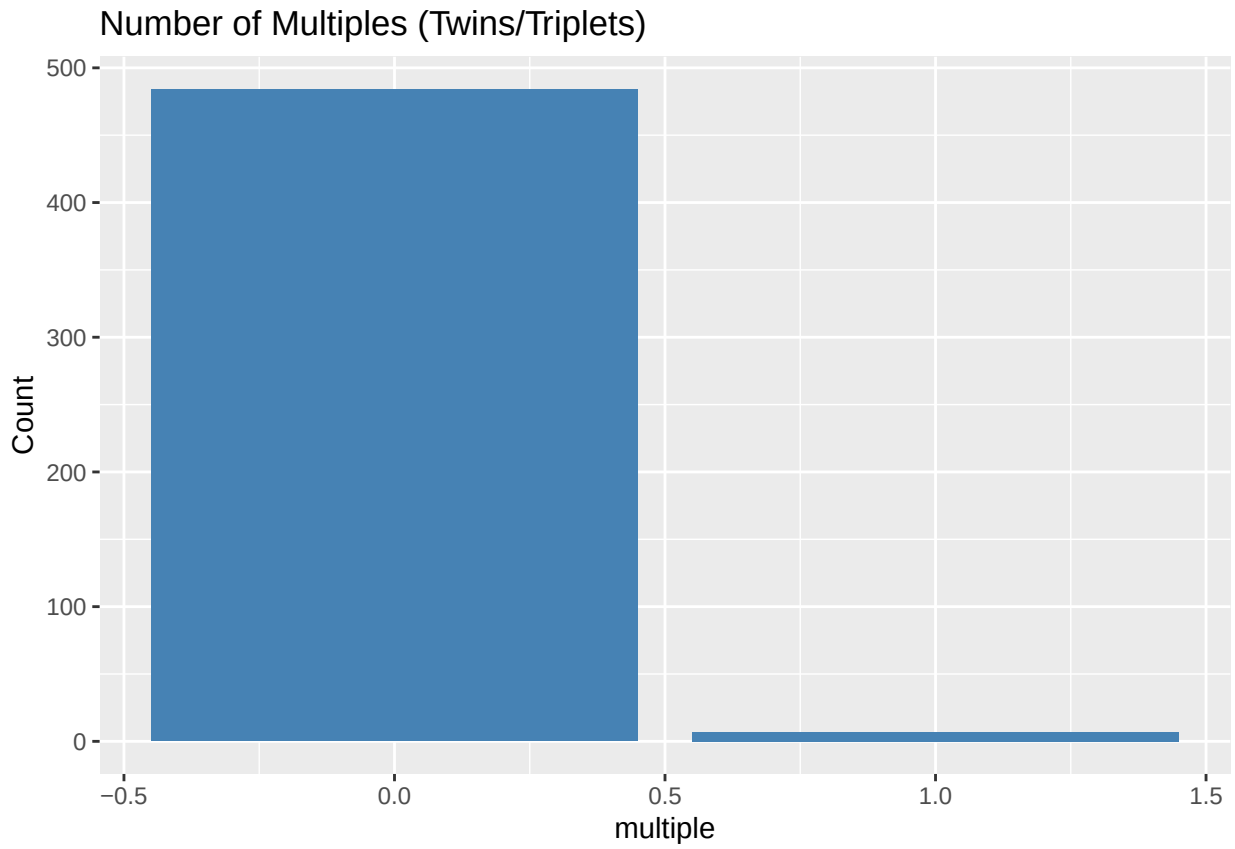
```
## # A tibble: 2 x 3
##   multiple     n   prop
##      <dbl> <int>  <dbl>
## 1        0   484 0.986
## 2        1     7 0.0143
```

We can create a bar chart that shows the assigned sex at birth and whether the newborn has a twin/triplet/etc. Out of the 491 samples, there are 98.5% of newborns don't have a twin/triplet/etc. and around half of the newborns (50.3%) are assigned as female.

```r
GA_bangladesh %>%
  group_by(sex) %>%
  summarize(count = n(), median_cost = median(gestage)) %>%
  ggplot(aes(x = sex, y = count)) +
  geom_bar(stat = "identity", fill = "steelblue") +
  labs(title = "Number of Females/Males",
       x = "sex", y = "Count")
```



```r
GA_bangladesh %>%
  group_by(multiple) %>%
  summarize(count = n(), median_cost = median(gestage)) %>%
  ggplot(aes(x = multiple, y = count)) +
  geom_bar(stat = "identity", fill = "steelblue") +
  labs(title = "Number of Multiples (Twins/Triplets)",
       x = "multiple", y = "Count")
```

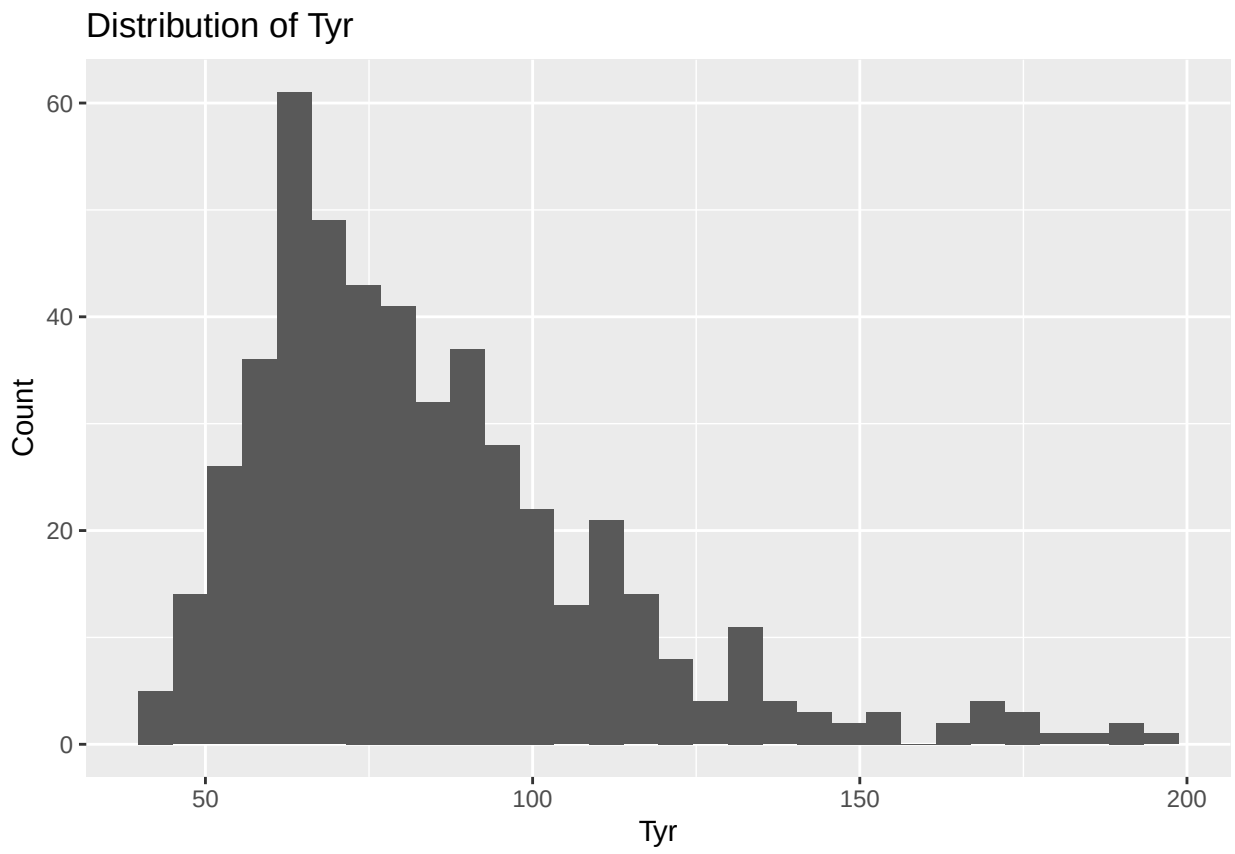


Univariate Analysis of Quantitative Predictor Variables

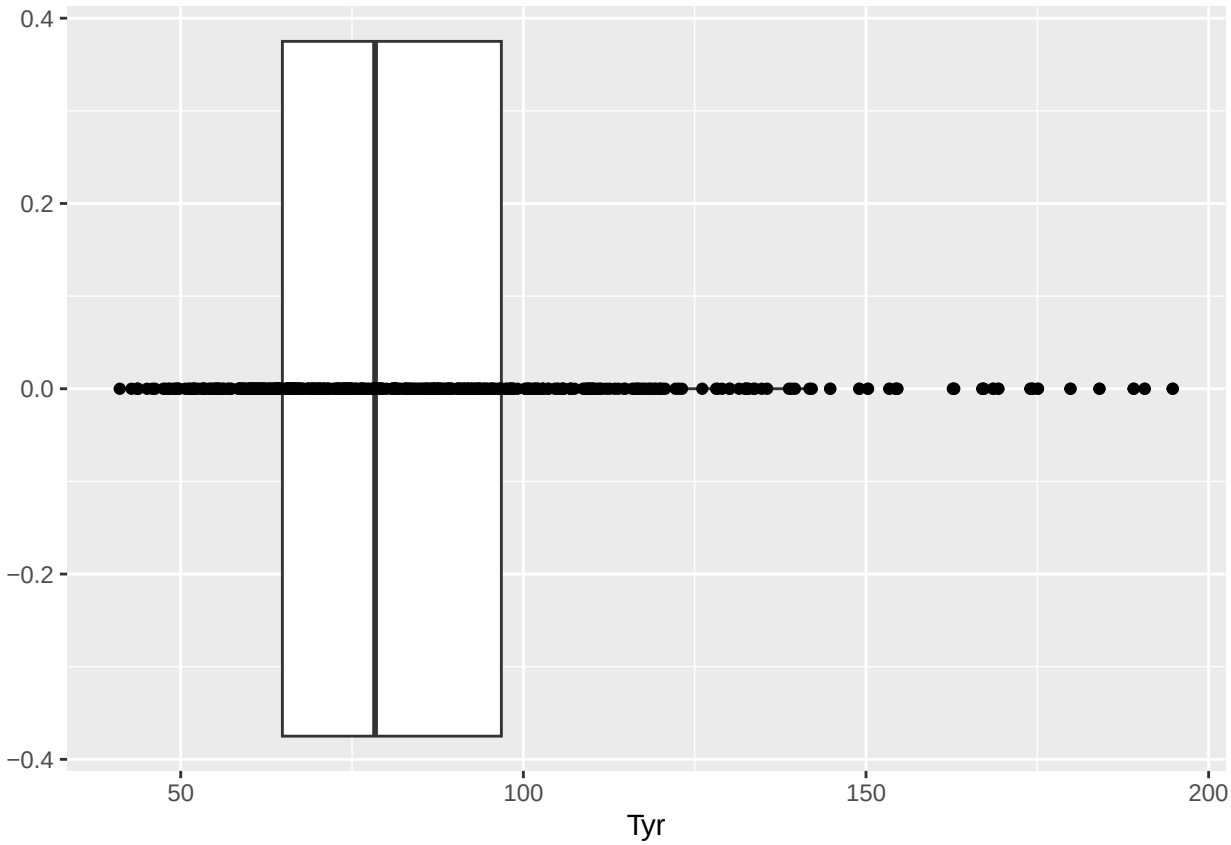
```r
# Summary statistics of Tyr
summary(GA_bangladesh$Tyr)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   41.11   64.83   78.36   84.53   96.79  194.76
```

```r
# Distribution of Tyr
ggplot(GA_bangladesh, aes(x=Tyr)) +
  geom_histogram(bins = 30) +
  labs(title="Distribution of Tyr", x="Tyr", y="Count")
```




```
ggplot(data = GA_bangladesh,
mapping = aes(
y = Tyr
)) + geom_boxplot() +
geom_point(x = 0) + coord_flip()
```

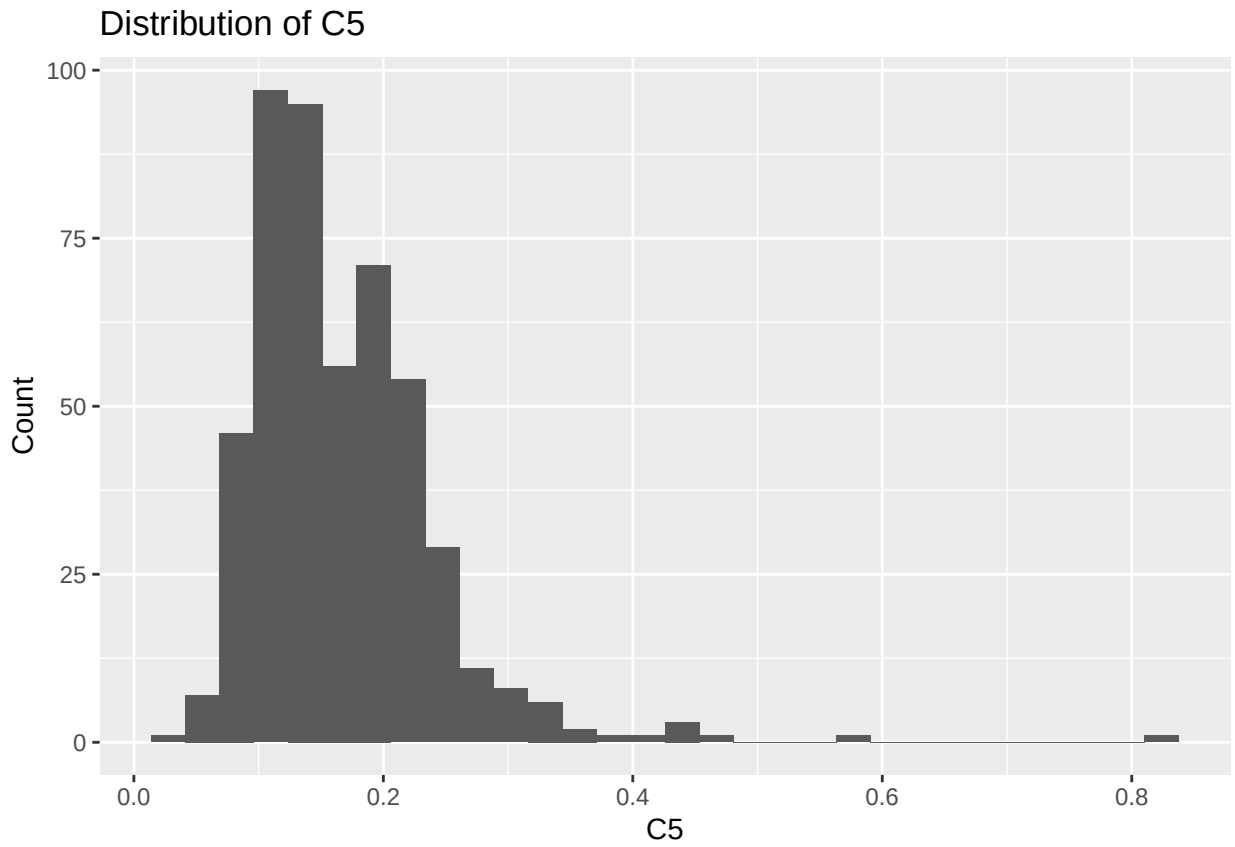


The 'Tyr' variable ranges from 41.11 to 194.76 with a mean of 84.53 and a median of 78.36. The distribution appears to be relatively skewed right with outliers to the right.

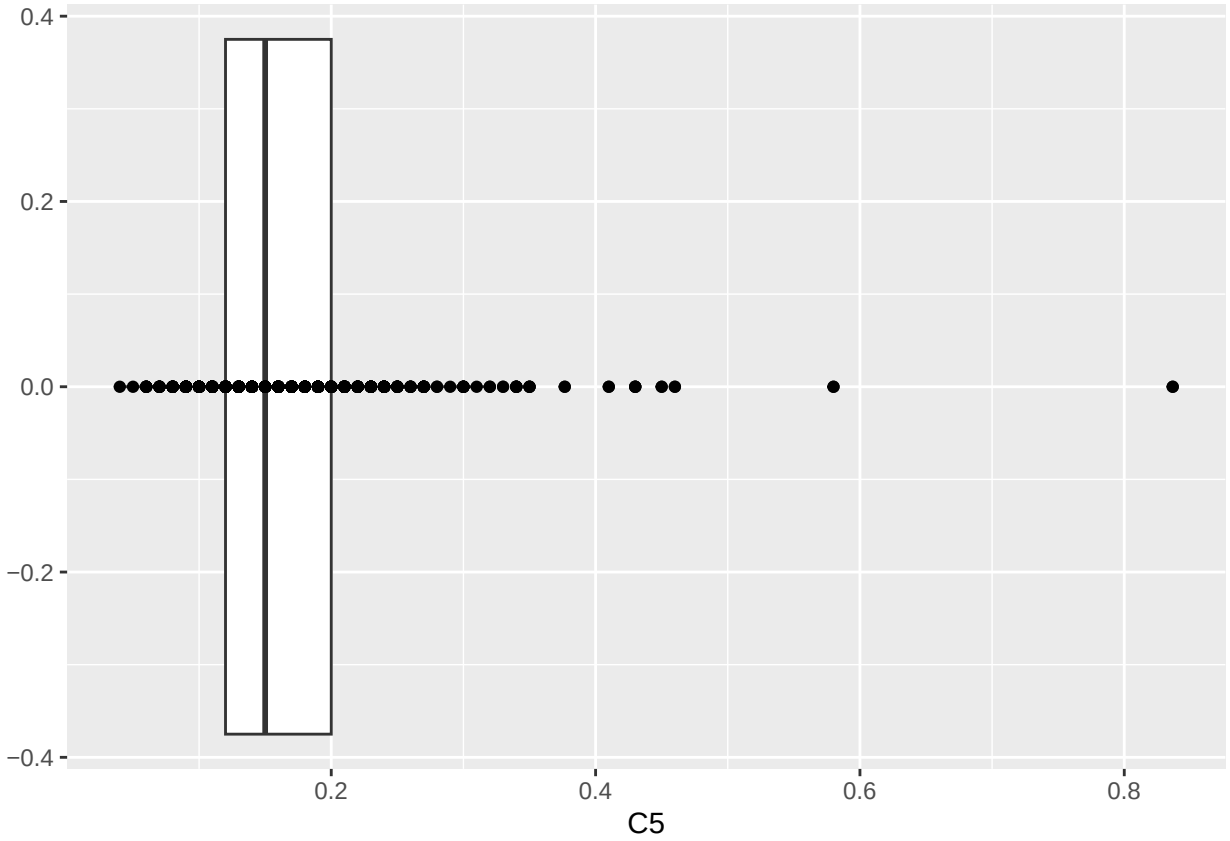
```
# Summary statistics of C5
summary(GA_bangladesh$C5)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##  0.0400  0.1200  0.1500  0.1670  0.2000  0.8367
```

```
# Distribution of C5
ggplot(GA_bangladesh, aes(x=C5)) +
  geom_histogram(bins = 30) +
  labs(title="Distribution of C5", x="C5", y="Count")
```




```
ggplot(data = GA_bangladesh,
mapping = aes(
y = C5
)) + geom_boxplot() +
geom_point(x = 0) + coord_flip()
```

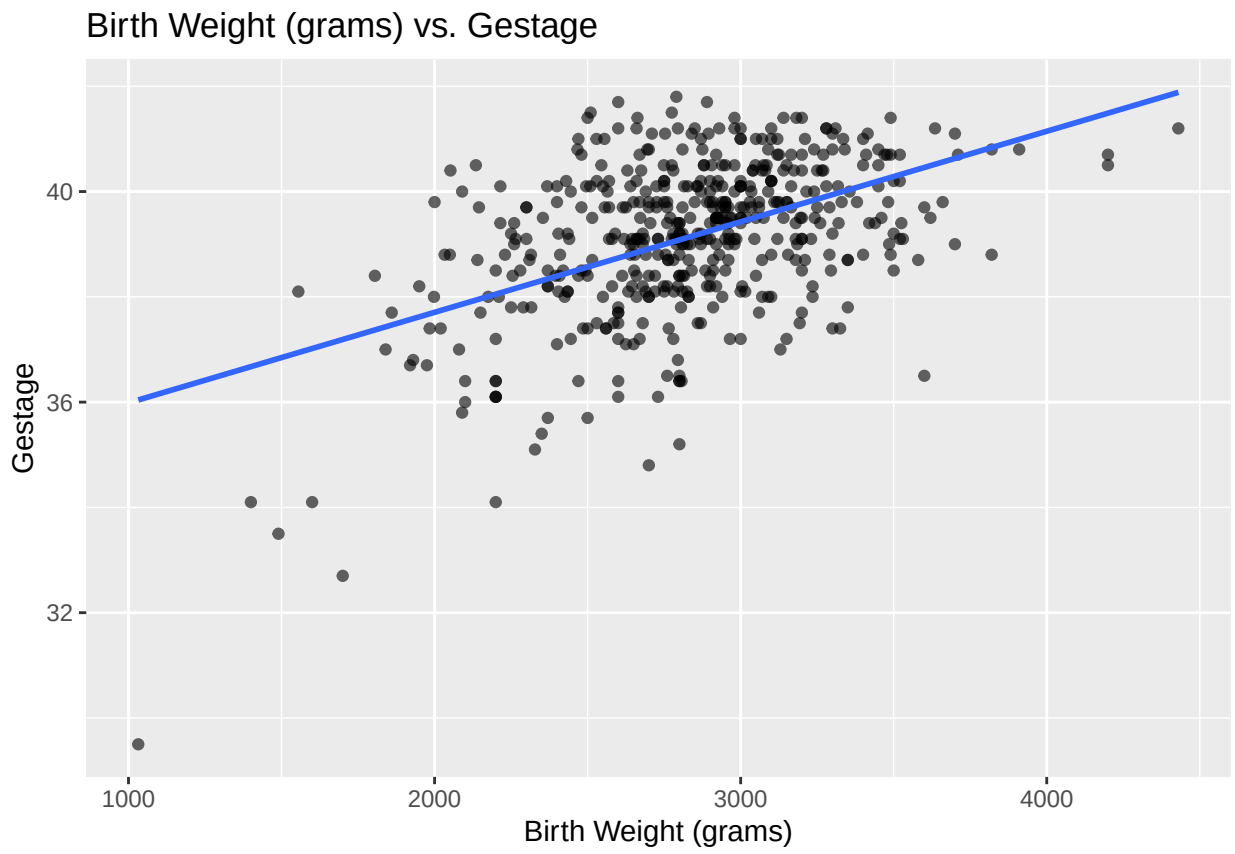


The 'C5' variable ranges from 0.0400 to 0.8367 with a mean of 0.1670 and a median of 0.1500. The distribution appears to be relatively skewed right with outliers to the right.

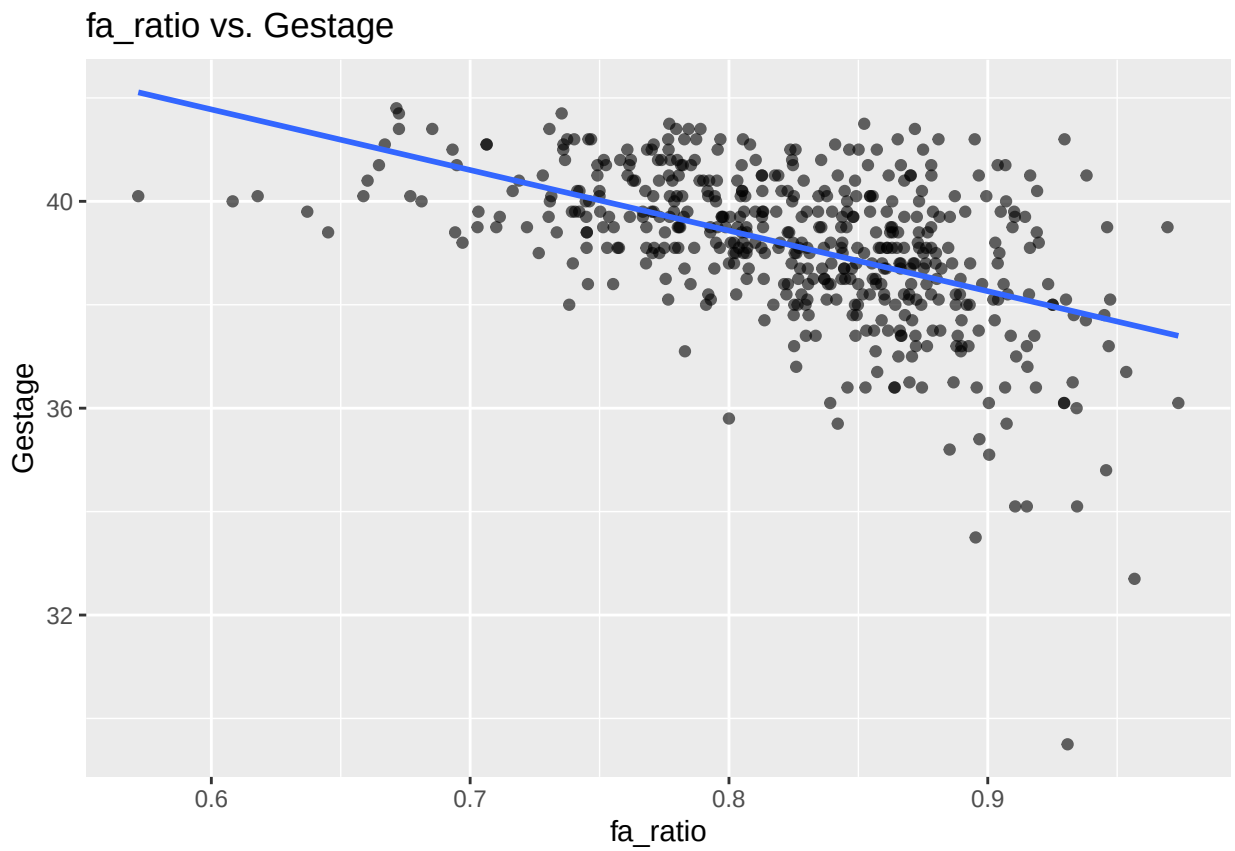
Bivariate Analysis of Quantitative Predictor Variables with Response Variable

Since we are interested in predicting the gestational age of newborns, we can explore the relationship between `gestage` and the other variables in the training set. We can create scatterplots of `gestage` against each variable to visualize their relationships. For example, the following code creates a scatterplot of `gestage` against `bweight`, 'gestage' against 'fa_ratio', and 'gestage' against 'Tyr'.

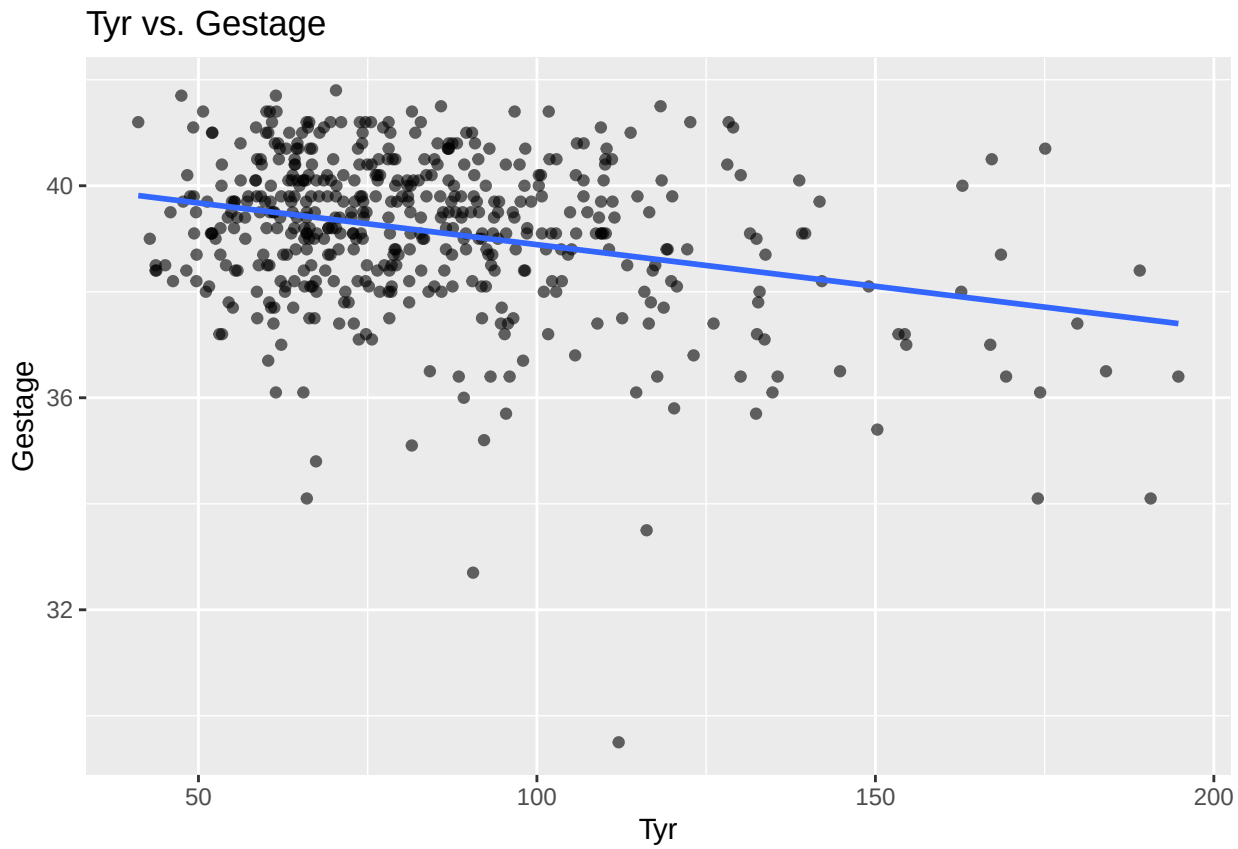
```
# Scatter plot of Birth Weight (grams) vs. Gestage
ggplot(GA_bangladesh, aes(x = bweight, y = gestage)) +
  geom_point(alpha=0.6) +
  geom_smooth(method = "lm", se = FALSE) +
  labs(title="Birth Weight (grams) vs. Gestage", x="Birth Weight (grams)", y="Gestage")
```




```
# Scatter plot of fa_ratio vs. Gestage
ggplot(GA_bangladesh, aes(x = fa_ratio, y = gestage)) +
  geom_point(alpha=0.6) +
  geom_smooth(method = "lm", se = FALSE) +
  labs(title="fa_ratio vs. Gestage", x="fa_ratio", y="Gestage")
```



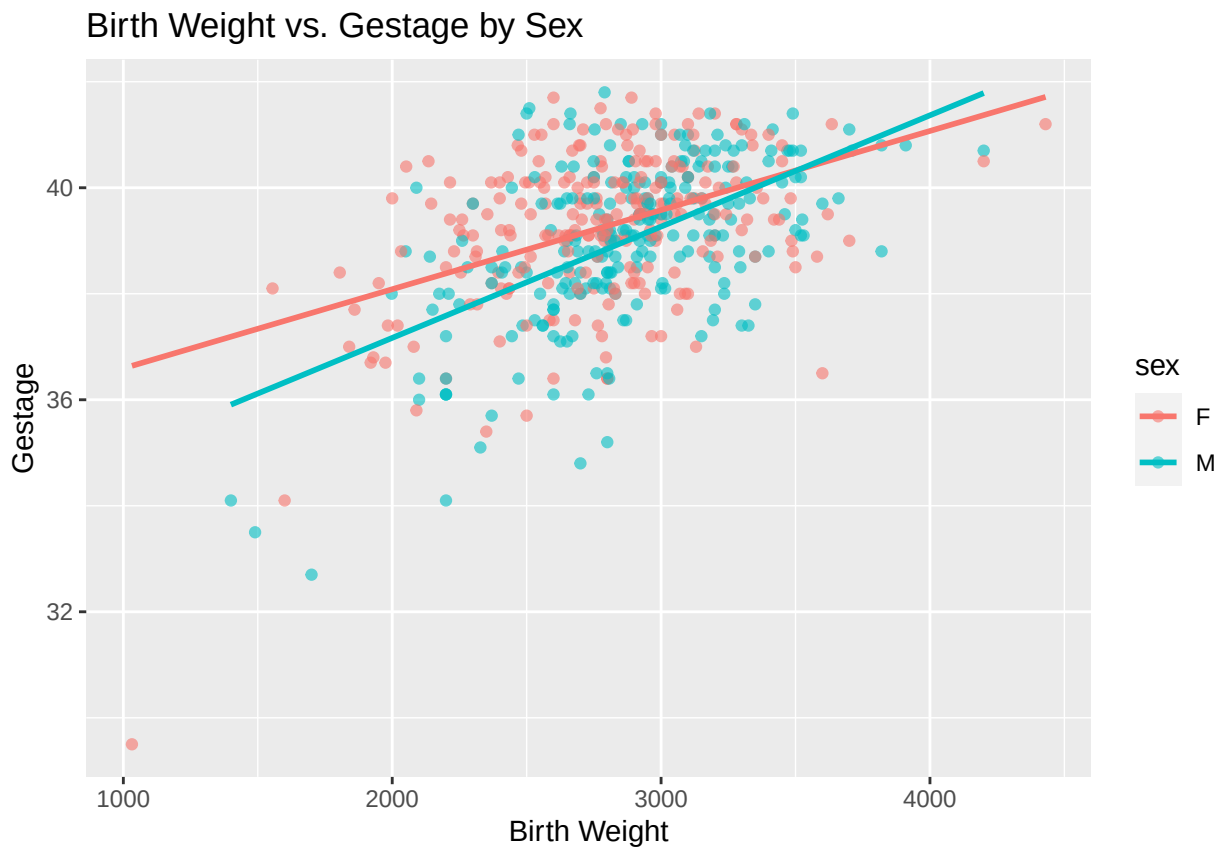

```
# Scatter plot of Tyr vs. Gestage
ggplot(GA_bangladesh, aes(x = Tyr, y = gestage)) +
  geom_point(alpha=0.6) +
  geom_smooth(method = "lm", se = FALSE) +
  labs(title="Tyr vs. Gestage", x="Tyr", y="Gestage")
```



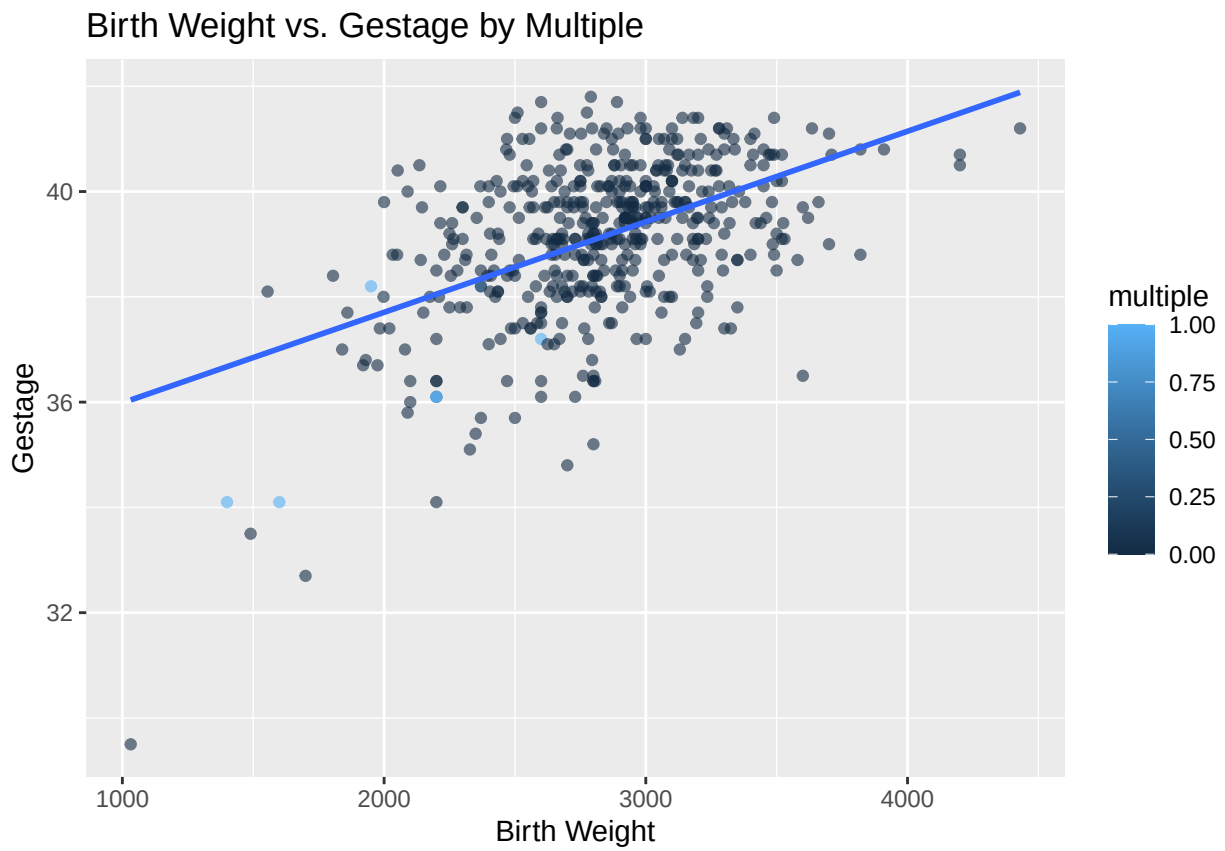
The scatter plots show a positive association between `bweight` and `gestage`. As the birth weight (grams) increases, the gestational age increases as well. And for our second scatter plot, there's a negative association between 'fa_ratio' and 'gestage'. As the 'fa_ratio' increases, the gestational age decreases. The same goes for our third scatterplot between 'Tyr' and 'gestage' indicating that there's a negative association between the two albeit less steep than 'fa_ratio' and 'gestage'.

Trivariate Analysis of Quantitative Predictor Variables with Response Variable and Categorical Predictor Variable

```
# Scatter plot of birth weight vs. gestage by Sex
ggplot(GA_bangladesh, aes(x=bweight, y=gestage, color=sex)) +
  geom_point(alpha=0.6) +
  geom_smooth(method = "lm", se = FALSE) +
  labs(title="Birth Weight vs. Gestage by Sex", x="Birth Weight", y="Gestage")
```




```
# Scatter plot of birth weight vs. gestage by Multiple
ggplot(GA_bangladesh, aes(x=bweight, y=gestage, color=multiple)) +
  geom_point(alpha=0.6) +
  geom_smooth(method = "lm", se = FALSE) +
  labs(title="Birth Weight vs. Gestage by Multiple", x="Birth Weight", y="Gestage")
```



The scatter plot shows a positive association between `Birth Weight` and `gestage` for both assigned Males and Females and whether they're multiples or not. However, the slope of the line is steeper for being assigned Male than Female, suggesting that the relationship may be stronger for Males. For multiples, it's hard to say seeing how only 7 of our samples are categorized as having a multiple ('1').

Based on these initial explorations, we can consider the following features for our model:

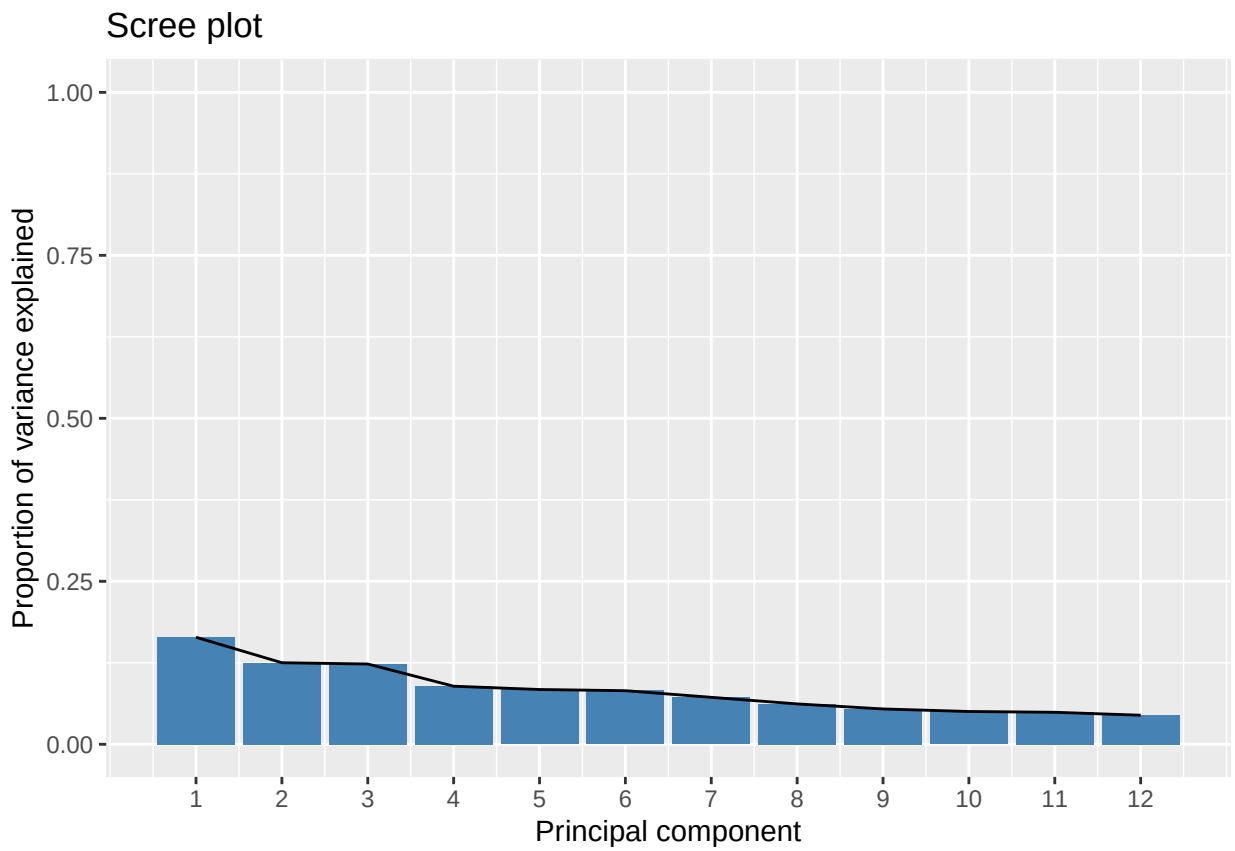
- `bweight`: Birth weight is a strong predictor of gestational age, as evidenced by the positive correlation in the scatterplot. We might also consider transforming `bweight` using a logarithmic or square root function to account for the nonlinear relationship between birth weight and gestational age.
- 'sex': it was shown that newborns assigned as male have a steeper, positive slope of the response variable gestage over newborns assigned as female, but not by much.
- `multiple`: Multiples have a different gestational age distribution than singletons, so including this variable in the model might improve its predictive accuracy.
- `fa_ratio`: Although the units of measurement are unknown, `fa_ratio` might be a useful predictor of gestational age due to its relationship with fetal development.
- 'Tyr': After doing some research, tyrosine is an amino acid the body makes which is an essential component for the production of several brain chemicals called neurotransmitters so this could an important feature to include as well.
- PCA: We can also perform PCA on the training set to identify any linear combinations of the variables that explain a large amount of variation in the data. We can then use the principal components (PCs) as features in our model.

For the unsupervised learning technique, we can perform PCA on the training set and visualize the results using a scree plot and a biplot. The scree plot shows the amount of variation explained by each PC, while the biplot shows the direction and magnitude of each variable's contribution to each PC. Here's an example code to perform PCA and create the plots:

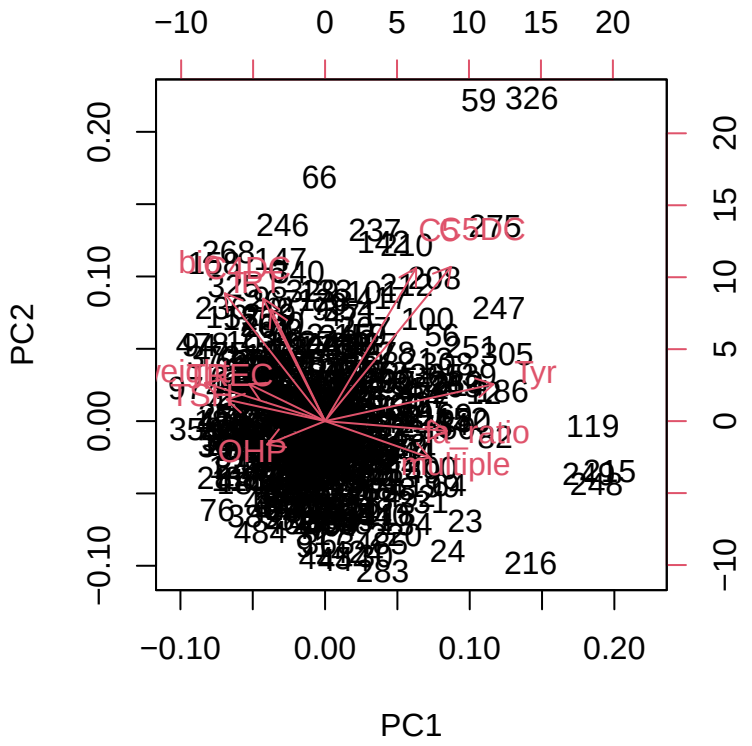
```
# Select relevant variables for PCA
names(GA_bangladesh)
```

```
##  [1] "study_id" "gestage"  "multiple" "bweight"  "sex"      "fa_ratio"
##  [7] "OHP"      "TSH"      "C5"       "C5DC"     "C4DC"     "Tyr"
## [13] "TREC"     "bio"      "IRT"
```

```
PCA_data <- subset(GA_bangladesh, select = -c(study_id, gestage, sex))

# Scale the data
PCA_data_scaled <- scale(PCA_data)

# Perform PCA
PCA_model <- prcomp(PCA_data_scaled, center = TRUE, scale. = TRUE)

# Scree plot
scree_data <- data.frame(
  PC = 1:ncol(PCA_data),
  Proportion_of_variance_explained = PCA_model$sdev^2 / sum(PCA_model$sdev^2)
)

ggplot(scree_data, aes(x = PC, y = Proportion_of_variance_explained)) +
  geom_bar(stat = "identity", fill = "steelblue") +
  geom_line(aes(x = PC, y = Proportion_of_variance_explained)) +
  scale_x_continuous(name = "Principal component", breaks = 1:ncol(PCA_data)) +
  scale_y_continuous(name = "Proportion of variance explained", limits = c(0, 1)) +
  ggtitle("Scree plot")
```




```
# Biplot
biplot(PCA_model)
```



From the scree plot, we can see that the first two PCs explain a large amount of variation in the data, while the remaining PCs explain relatively little. The biplot shows that `Tyr` and `fa_ratio` have the strongest positive association with PC1, while `bweight` has just as strong of a positive association with PC2. we can also see how variables 'C5' and 'C5DC' have positive correlations between both PCs. These variables might be good candidates for inclusion in our model based on their contribution to the PCs.

```
# Fit Linear model without 'study_id'
linear_model <- lm(gestage ~ . - study_id, data = GA_bangladesh)

summary(linear_model)
```

```
##
## Call:
## lm(formula = gestage ~ . - study_id, data = GA_bangladesh)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -4.2098 -0.6915  0.0240  0.7299  3.5347
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) 42.6813509  0.8971697  47.573  < 2e-16 ***
## multiple    -1.0250608  0.4280699  -2.395 0.017023 *
## bweight      0.0012778  0.0001302   9.811  < 2e-16 ***
## sexM        -0.1737918  0.1058873  -1.641 0.101396
## fa_ratio    -7.3045688  0.8635232  -8.459 3.32e-16 ***
```

```
## OHP         -0.0289920  0.0060555  -4.788 2.25e-06 ***
## TSH          0.0214616  0.0120479   1.781 0.075491 .
## C5          -1.8682774  0.7207918  -2.592 0.009835 **
## C5DC        -2.5196990  1.7932429  -1.405 0.160640
## C4DC         0.7975400  0.2922697   2.729 0.006592 **
## Tyr         -0.0071949  0.0019697  -3.653 0.000288 ***
## TREC         0.0002255  0.0001257   1.794 0.073390 .
## bio         -0.0034416  0.0030494  -1.129 0.259624
## IRT          0.0045005  0.0044010   1.023 0.307009
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.08 on 477 degrees of freedom
## Multiple R-squared:  0.4867, Adjusted R-squared:  0.4727
## F-statistic: 34.79 on 13 and 477 DF,  p-value: < 2.2e-16
```

```
# Compute variable importance
var_importance <- varImp(linear_model)

# View variable importance
print(var_importance)
```

```
##           Overall
## multiple 2.394611
## bweight  9.811404
## sexM     1.641290
## fa_ratio 8.459030
## OHP      4.787738
## TSH      1.781354
## C5       2.591979
## C5DC     1.405108
## C4DC     2.728781
## Tyr      3.652807
## TREC     1.794354
## bio      1.128624
## IRT      1.022613
```

```
# Extract p-values of the predictors
summary(linear_model)$coefficients[,4]
```

```
##   (Intercept)      multiple       bweight          sexM      fa_ratio
## 3.300179e-183  1.702298e-02  8.028298e-21  1.013963e-01  3.322625e-16
##           OHP           TSH            C5          C5DC          C4DC
##  2.254064e-06  7.549090e-02  9.835214e-03  1.606402e-01  6.591680e-03
##           Tyr          TREC           bio           IRT
##  2.881285e-04  7.338973e-02  2.596238e-01  3.070093e-01
```

We can see from our p-values and Variance Importance Measure which variables are the most influential predictors to measure our response variable 'gestage'. This along with our biplot model, we can say that our most influential predictors would be 'bweight', 'fa_ratio', 'OHP', 'Tyr', 'C4DC', and 'C5'.

Part b (Code: 2 pts)

Fit one or more ordinary least-squares models on the training set (using `lm`).

OLS

```
# Fit an OLS model to predict gestational age using our influential predictors
model <- lm(gestage ~ bweight + fa_ratio + OHP + Tyr + C4DC + C5, data = GA_bangladesh)

# Print the summary of the model
summary(model)
```

```
##
## Call:
## lm(formula = gestage ~ bweight + fa_ratio + OHP + Tyr + C4DC +
##     C5, data = GA_bangladesh)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -4.3918 -0.6457 -0.0324  0.6911  3.6836
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 42.875121   0.872189  49.158  < 2e-16 ***
## bweight      0.001277   0.000122  10.467  < 2e-16 ***
## fa_ratio    -7.555627   0.847803  -8.912  < 2e-16 ***
## OHP         -0.026143   0.005978  -4.373 1.50e-05 ***
## Tyr         -0.009119   0.001923  -4.743 2.78e-06 ***
## C4DC         0.775673   0.281575   2.755  0.00609 **
## C5          -2.130698   0.688005  -3.097  0.00207 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.098 on 484 degrees of freedom
## Multiple R-squared:  0.4616, Adjusted R-squared:  0.4549
## F-statistic: 69.15 on 6 and 484 DF,  p-value: < 2.2e-16
```

In this example, we are fitting a simple linear regression model to predict gestational age (`gestage`) using birth weight (`bweight`), ratio of fetal hemoglobin ('fa_ratio'), concentration of 17-hydroxyprogesterone ('OHP'), concentration of tyrosine ('Tyr'), concentration of acylcarnitine C4-DC ('C4DC'), and concentration of complement component 5 ('C5') as predictor variables. The formula in the `lm` function specifies the response variable followed by the predictor variables. The `data` argument specifies the dataset to use.

After fitting the model, we can use the `summary` function to obtain more information about the model, including the estimated coefficients, standard errors, t-values, and p-values.

Part c (Code: 2 pts)

Fit one or more penalized regression models (ridge regression, LASSO, elastic net) on the training set. You do not need to tune the model yet.

Ridge Regression

```
ridge_model <- linear_reg(mode = "regression", engine = "glmnet",
                          penalty = tune(),
                          mixture = 0)

ridge_recipe <- recipe(gestage ~ bweight + fa_ratio + OHP + Tyr + C4DC + C5, data = GA_bangladesh) %>%
  step_normalize(all_numeric_predictors()) %>%
  step_dummy(all_nominal_predictors())

ridge_wflow <- workflow() %>%
  add_model(ridge_model) %>%
  add_recipe(ridge_recipe)
```

Part d (Code: 2 pts)

Fit one or more "pure prediction" regression models (bagging/random forests, boosted trees, or neural network) on the training set. You do not need to tune the model yet.

Bagging/Random Forest

```
# Define the bagging/random forest model
GA_bangladesh_rf_model <- rand_forest(mode = "regression", trees = 500, mtry = 3)

# Define the recipe
GA_bangladesh_recipe <- recipe(gestage ~ bweight + fa_ratio + OHP + Tyr + C4DC + C5, data = GA_banglades
  step_normalize(all_numeric_predictors()) %>%
  step_dummy(all_nominal_predictors())

# Define the workflow
GA_bangladesh_rf_wflow <- workflow() %>%
  add_model(GA_bangladesh_rf_model) %>%
  add_recipe(GA_bangladesh_recipe)
```

Part e (Code: 4 pts; Explanation: 2 pts)

Using cross-validation, do any necessary tuning on your models from parts (b)-(d) and select a final model. Explain why you think it is the best model. Following the authors, you should use either mean absolute error or RMSE to evaluate your models.

To evaluate the models using the Root Mean Squared Error (RMSE), we can modify the code as follows:

```
set.seed(123)
bangladesh_kfold <- vfold_cv(GA_bangladesh, v = 5, repeats = 3)

ols_model <- linear_reg(mode = "regression", engine = "lm")

ols_wflow <- workflow() %>%
  add_model(ols_model) %>%
  add_recipe(ridge_recipe)

ols_fit <- fit(ols_wflow, data = GA_bangladesh)

extract_fit_engine(ols_fit) %>% car::vif()
```

```
##  bweight fa_ratio      OHP      Tyr     C4DC       C5
## 1.077011 1.207723 1.065224 1.140734 1.081797 1.069350
```

```
predictions_ols <- broom::augment(ols_fit, new_data = GA_zambia)
predictions_ols %>% dplyr::select(
  study_id, gestage, .pred
)
```

```
## # A tibble: 70 x 3
##    study_id     gestage .pred
##    <chr>          <dbl> <dbl>
##  1 30-000764-H1    42    40.2
##  2 30-000787-H1    38.9  39.5
##  3 30-000794-H1    35.3  38.2
##  4 30-000798-H1    41.7  39.3
##  5 30-000824-H1    37.1  38.5
##  6 30-000828-H1    40.9  38.6
##  7 30-000844-H1    38.1  38.7
##  8 30-000845-H1    40.4  40.4
##  9 30-000846-H1    40.3  41.3
## 10 30-000859-H1    40.7  38.8
## # i 60 more rows
```

```
rmse(predictions_ols, truth = gestage, estimate = .pred)
```

```
## # A tibble: 1 x 3
##   .metric .estimator .estimate
##   <chr>   <chr>          <dbl>
## 1 rmse    standard        1.36
```

```
set.seed(123)
ridge_tune <- tune_grid(ridge_wflow,
                        resamples = vfold_cv(GA_bangladesh),
                        grid = grid_regular(penalty(range = c(-5, 2)), levels = 50))
```

```
## Warning: package 'glmnet' was built under R version 4.2.3
```

```
## Warning: package 'Matrix' was built under R version 4.2.3
```

```
ridge_best <- select_by_one_std_err(ridge_tune,
                                    metric = "rmse",
                                    desc(penalty))

ridge_wflow_final <- finalize_workflow(ridge_wflow, parameters = ridge_best)

ridge_fit <- fit(ridge_wflow_final, data = GA_bangladesh)

predictions_ridge <- augment(ridge_fit, new_data = GA_zambia)
predictions_ridge %>% dplyr::select(
  study_id, gestage, .pred
)
```

```
## # A tibble: 70 x 3
##    study_id     gestage .pred
##    <chr>          <dbl> <dbl>
##  1 30-000764-H1    42    39.8
##  2 30-000787-H1    38.9  39.4
##  3 30-000794-H1    35.3  38.4
##  4 30-000798-H1    41.7  39.3
##  5 30-000824-H1    37.1  38.7
##  6 30-000828-H1    40.9  38.7
##  7 30-000844-H1    38.1  38.8
##  8 30-000845-H1    40.4  40.1
##  9 30-000846-H1    40.3  40.7
## 10 30-000859-H1    40.7  38.8
## # i 60 more rows
```

```
rmse(predictions_ridge, truth = gestage, estimate = .pred)
```

```
## # A tibble: 1 x 3
##   .metric .estimator .estimate
##   <chr>   <chr>          <dbl>
## 1 rmse    standard        1.35
```

```
set.seed(123)
# Fit the bagging/random forest model to the training data
GA_bangladesh_rf_fit <- fit(GA_bangladesh_rf_wflow, data = GA_bangladesh)

# Make predictions on the test data
predictions_rf <- predict(GA_bangladesh_rf_fit, new_data = GA_zambia) %>%
  bind_cols(actual = GA_zambia$gestage)

# Calculate RMSE
rmse_rf <- rmse(predictions_rf, truth = actual, estimate = .pred)
rmse_rf
```

```
## # A tibble: 1 x 3
##   .metric .estimator .estimate
##   <chr>   <chr>          <dbl>
## 1 rmse    standard        1.38
```

The code above fits an OLS model, ridge regression model, and bagging/random forest model to predict gestational age using the influential predictors. It then calculates the Root Mean Squared Error (RMSE) for each model by comparing the predictions with the actual gestational age values. The resulting RMSE values are printed for each model.

This allows us to evaluate the models based on their ability to measure the average difference between the predicted values and the actual values. The lower the RMSE, the closer the predicted values are to the actual values, which means that the model is making more accurate predictions.

Our final and best model has an RMSE of 1.351052 which is the ridge regression model!

It is important to note, however, that RMSE should be compared between models that are trained and evaluated on the same dataset, and it should not be used as the sole criterion for model selection. Other factors such as interpretability, computational complexity, and robustness to outliers should also be considered when selecting a model.

Part f (Code: 3 pts; Explanation: 4 pts)

Using the model from part (e), make predictions on the test set. How well does your model predict gestational age? Investigate and describe any patterns you see in the prediction errors (e.g., by creating and interpreting a residual plot or a plot of actual vs. predicted response).

To make predictions on the test set (`GA_zambia`) using the ridge regression model trained on the training set (`GA_bangladesh`), you can use the `predict()` function as follows:

```
# Set seed for reproducibility
set.seed(123)

# Define the hyperparameter grid
ridge_grid <- grid_regular(penalty(), mixture())

ridge_model <- linear_reg(
  mode = "regression",
  engine = "glmnet",
  penalty = tune(),
  mixture = tune()
)

# Define the tuning workflow
ridge_tune <- workflow() %>%
  add_model(ridge_model) %>%
  add_recipe(ridge_recipe) %>%
  tune_grid(
    resamples = bangladesh_kfold,
    grid = ridge_grid,
    metrics = metric_set(rmse),
    control = control_grid(verbose = FALSE)
  )

# Find the best hyperparameters and finalize the workflow
ridge_best <- ridge_tune %>%
  select_best("rmse")

ridge_wflow_final <- finalize_workflow(ridge_wflow, parameters = ridge_best)

# Fit the model on the training set
ridge_fit <- ridge_wflow_final %>%
  fit(data = GA_bangladesh)

# Make predictions on the test set
GA_zambia_predictions <- predict(ridge_fit, new_data = GA_zambia) %>%
  as_tibble() %>%
  set_names("predicted_gestage")

# Combine the predicted values with the actual values
GA_zambia_results <- cbind(GA_zambia, GA_zambia_predictions)

# Print the first few rows of the results
head(GA_zambia_results)
```

```
##       study_id  gestage multiple bweight sex  fa_ratio  OHP        TSH   C5
```

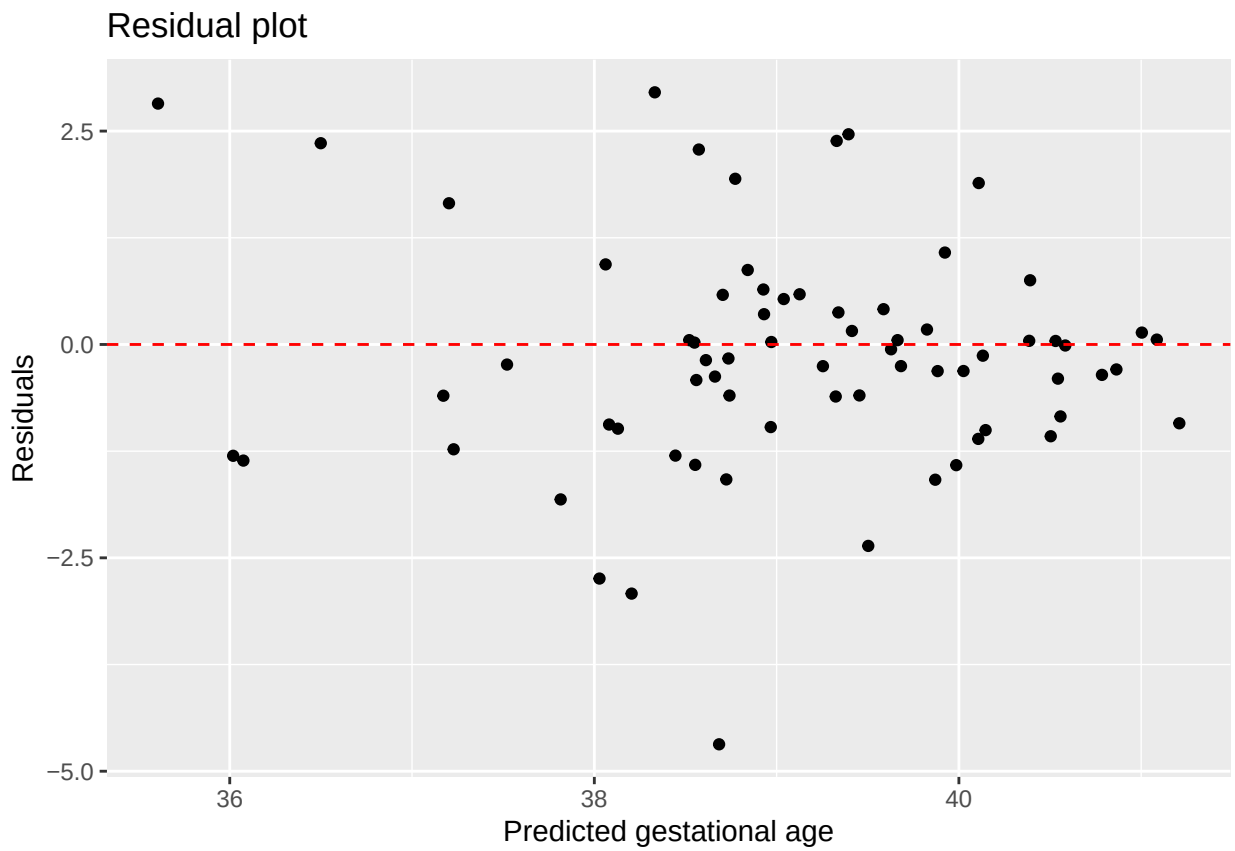
```
## 1 30-000764-H1 42.00000        0    3400   M 0.7532808 10.7  2.3079098 0.23
## 2 30-000787-H1 38.85714        0    2800   M 0.7833537 10.6  1.5072678 0.22
## 3 30-000794-H1 35.28571        0    2980   F 0.9675237 11.5  2.2604642 0.29
## 4 30-000798-H1 41.71429        0    2300   M 0.7649402  7.8  1.1257837 0.16
## 5 30-000824-H1 37.14286        0    2760   F 0.8937583 10.2  0.2262458 0.16
## 6 30-000828-H1 40.85714        0    2600   M 0.8526448 20.9 19.2419118 0.25
##   C5DC C4DC   Tyr     TREC    bio      IRT predicted_gestage
## 1 0.07 0.28 90.01 527.6308 62.436 33.39429          40.10888
## 2 0.04 0.35 64.59 496.5644 41.897 23.59941          39.45438
## 3 0.04 0.31 56.07 462.0807 40.111 17.48940          38.20455
## 4 0.04 0.30 42.32 147.8736 71.441 11.80875          39.32955
## 5 0.05 0.20 71.92 567.4066 32.074 18.92420          38.55353
## 6 0.04 0.42 48.94 322.8805 58.864 33.20741          38.57336
```

```
# Calculate the RMSE
rmse_ridge <- GA_zambia_results %>%
  rmse(gestage, predicted_gestage)

rmse_ridge
```

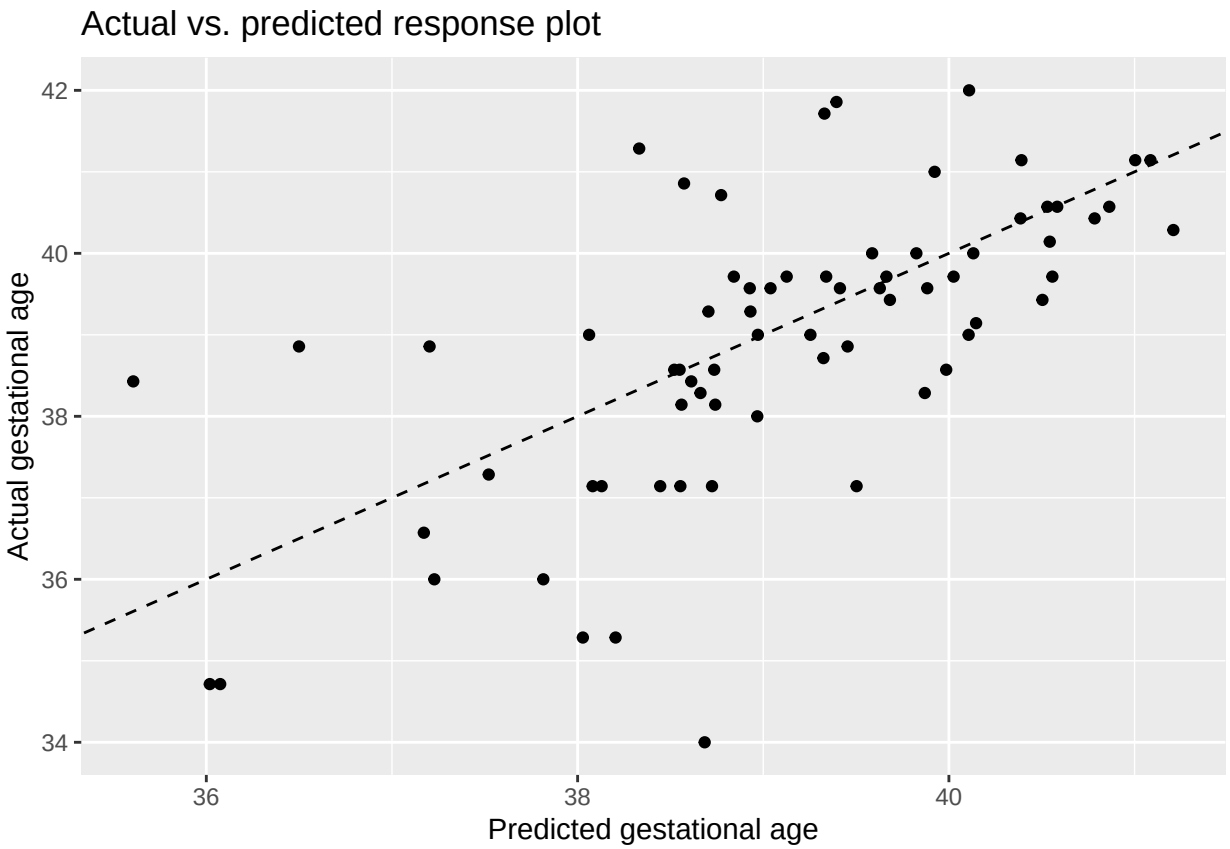
```
## # A tibble: 1 x 3
##   .metric .estimator .estimate
##   <chr>   <chr>          <dbl>
## 1 rmse    standard        1.35
```

To investigate how well the model predicts gestational age, you can create a residual plot or a plot of actual vs. predicted response as follows:

```
# Create a residual plot
ggplot(GA_zambia_results, aes(x = predicted_gestage, y = gestage - predicted_gestage)) +
  geom_point() +
  geom_hline(yintercept = 0, linetype = "dashed", color = "red") +
  labs(x = "Predicted gestational age", y = "Residuals", title = "Residual plot")
```




```
# Create a plot of actual vs. predicted response
ggplot(GA_zambia_results, aes(x = predicted_gestage, y = gestage)) +
  geom_point() +
  geom_abline(slope = 1, intercept = 0, linetype = "dashed") +
  labs(x = "Predicted gestational age", y = "Actual gestational age", title = "Actual vs. predicted resp
```



The residual plot shows the difference between the actual gestational age and the predicted gestational age for each observation in the test set. If the model is accurate, we would expect the residuals to be randomly scattered around 0 with no clear pattern. The actual vs. predicted response plot shows the predicted gestational age on the x-axis and the actual gestational age on the y-axis. If the model is accurate, we would expect the points to fall close to the dashed line (which represents perfect predictions).

Based on these plots and any additional performance metrics (such as mean absolute error or root mean squared error), we can evaluate how well the model is predicting gestational age on the test set. If the model is not accurate, we may need to explore different modeling approaches or adjust our features to improve performance. We can see that our RMSE for the test set changes from 1.351052 to 1.349695 meaning that we're consistent with our results even though the data set has changed.

In this case, the residual plot for the ridge regression model shows that the residuals have a roughly symmetric distribution around zero, indicating that the model is not systematically overestimating or underestimating gestational age overall. However, there are some outlier observations with larger residuals, particularly for predicted gestational ages around 38.5-40 weeks which suggests that the model may not be capturing some non-linear relationships in the data. However, the overall spread of the residuals is relatively small, which suggests that the model is making reasonably accurate predictions of gestational age.

Applied Problem 2 (36 points total)

The data for this problem comes from the 2017 National Household Travel Survey (https://nhts.ornl.gov/). The `drivers_train` and `drivers_test` datasets contain information about approximately 2000 people in the Los Angeles/Long Beach/Anaheim and Riverside/San Bernardino/Ontario Metropolitan Statistical Areas who commute to work (1526 in the training set, 508 in the test set). In other words, I have *already* done

the training-test splitting for you. Please see the `drivers_dictionary` file on Canvas for an explanation of each of the variables.

Your goal on this problem is to predict whether someone drives a hybrid/electric vehicle (`HybridDriver`). Show me what you have learned in this class.

Part a (Code: 5 pts; Explanation: 10 pts)

Change the `HybridDriver` variable to a factor variable, or create a new factor variable from `HybridDriver` to serve as the response variable.

Then, provide a *detailed* and *well-reasoned* description of your feature selection/engineering process. This process should be primarily based on exploratory data analysis; however, if you know things about driving in Southern California and/or have a concept of a "stereotypical hybrid driver," you can use that real-world knowledge to help inform your decisions. You are encouraged to transform predictor variables and/or create new features from the original variables (e.g., interactions) if you can justify doing so.

We want to change the `HybridDriver` variable to a factor variable and provide a detailed and well-reasoned description of the feature selection/engineering process:

```
# Change HybridDriver to a factor variable
drivers_train <- drivers_train %>%
  mutate(HybridDriver = factor(HybridDriver, levels = c("No", "Yes")))
drivers_test <- drivers_test %>%
  mutate(HybridDriver = factor(HybridDriver, levels = c("No", "Yes")))
```

To select and engineer features for predicting `HybridDriver`, we can start by exploring the relationships between the predictors and the response variable using visualizations and summary statistics. Here are some ideas for exploring the data:

1. **Explore the distribution of the response variable**: We can start by examining the distribution of the response variable `HybridDriver` to see how many drivers in the training set have hybrid/electric vehicles.

```
drivers_train %>%
  count(HybridDriver) %>%
  mutate(prop = n / sum(n))
```

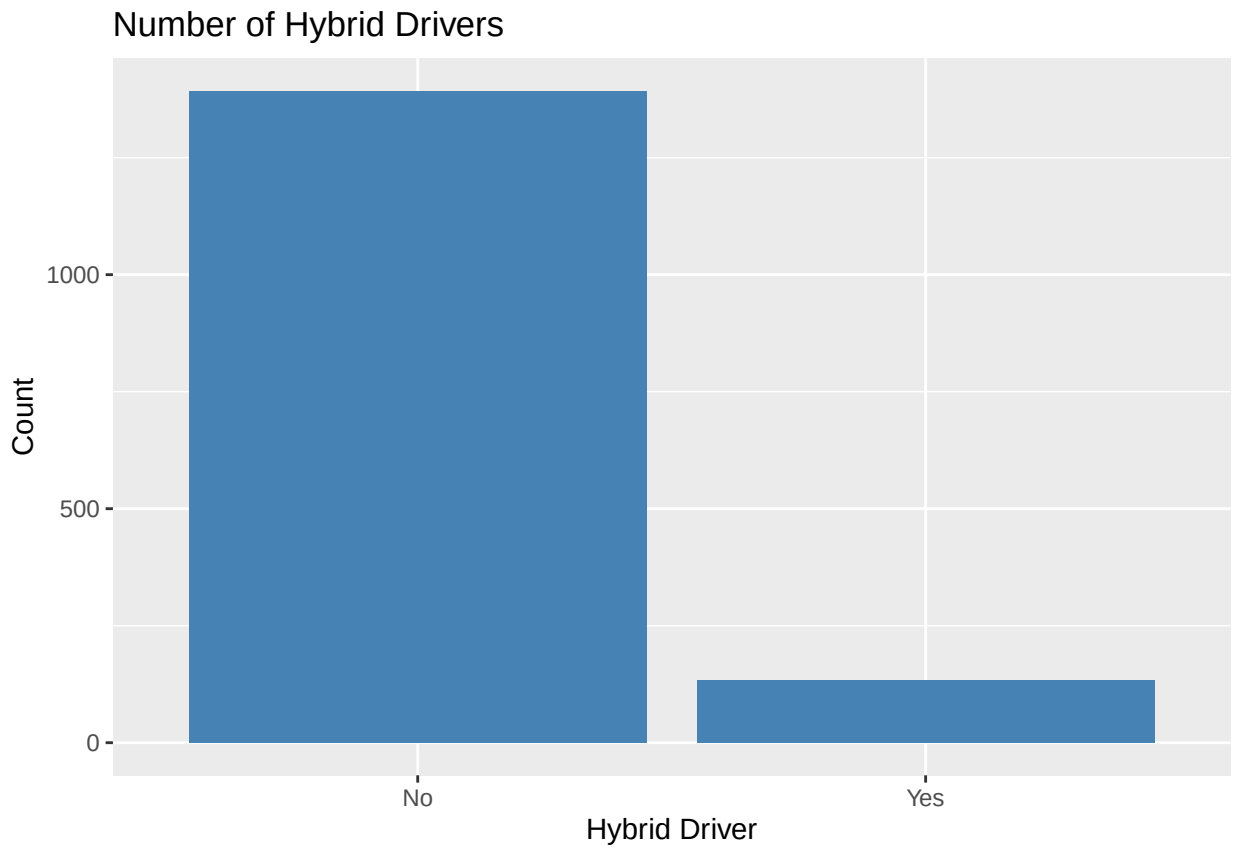
```
## # A tibble: 2 x 3
##   HybridDriver     n   prop
##   <fct>        <int>  <dbl>
## 1 No            1393 0.913
## 2 Yes            133 0.0872
```

This code counts the number of drivers in each level of `HybridDriver` and calculates the proportion of drivers with hybrid/electric vehicles. We can see that only about 8.7% of drivers in the training set have hybrid/electric vehicles.

We can create a bar chart that shows the number of whether the driver drives a hybrid or electric vehicle.

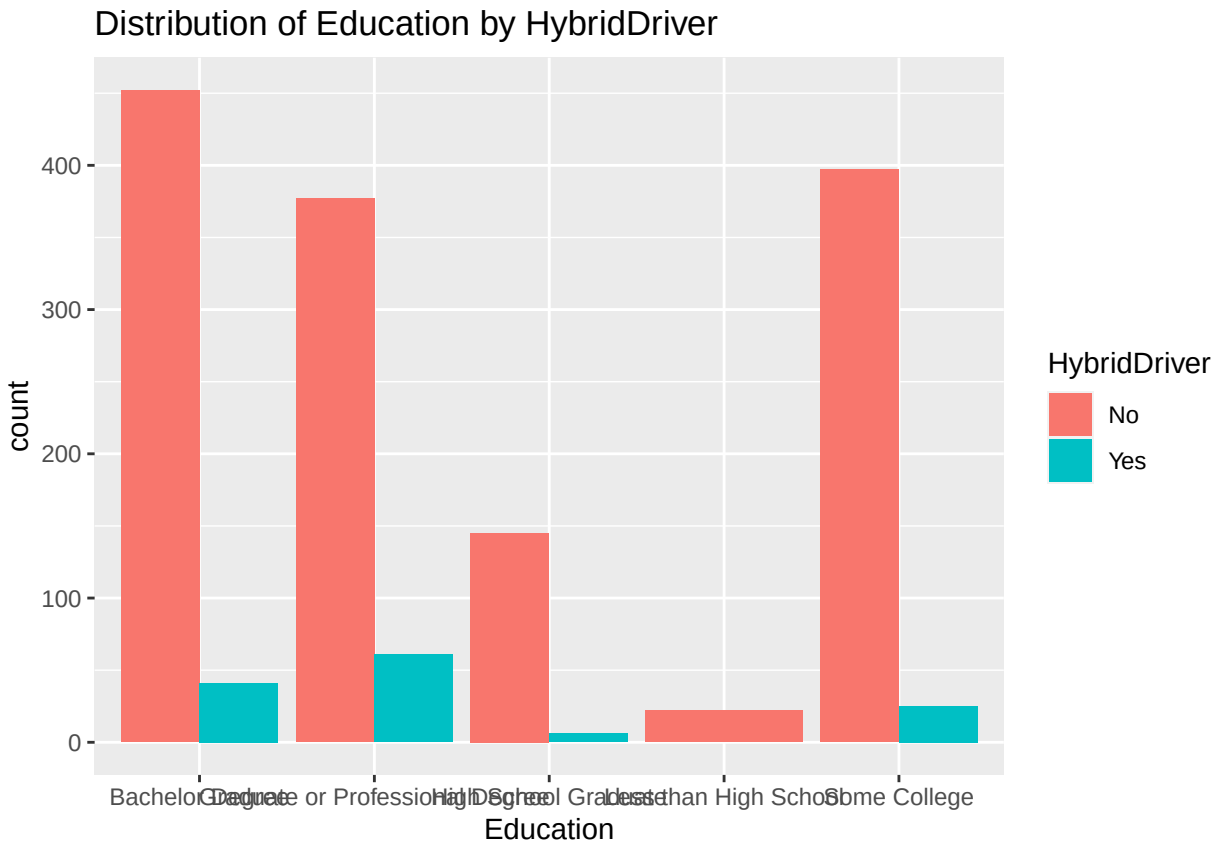
```
drivers_train %>%
  count(HybridDriver) %>%
  ggplot(aes(x = HybridDriver, y = n)) +
```

```
  geom_bar(stat = "identity", fill = "steelblue") +
  labs(title = "Number of Hybrid Drivers",
       x = "Hybrid Driver",
       y = "Count")
```

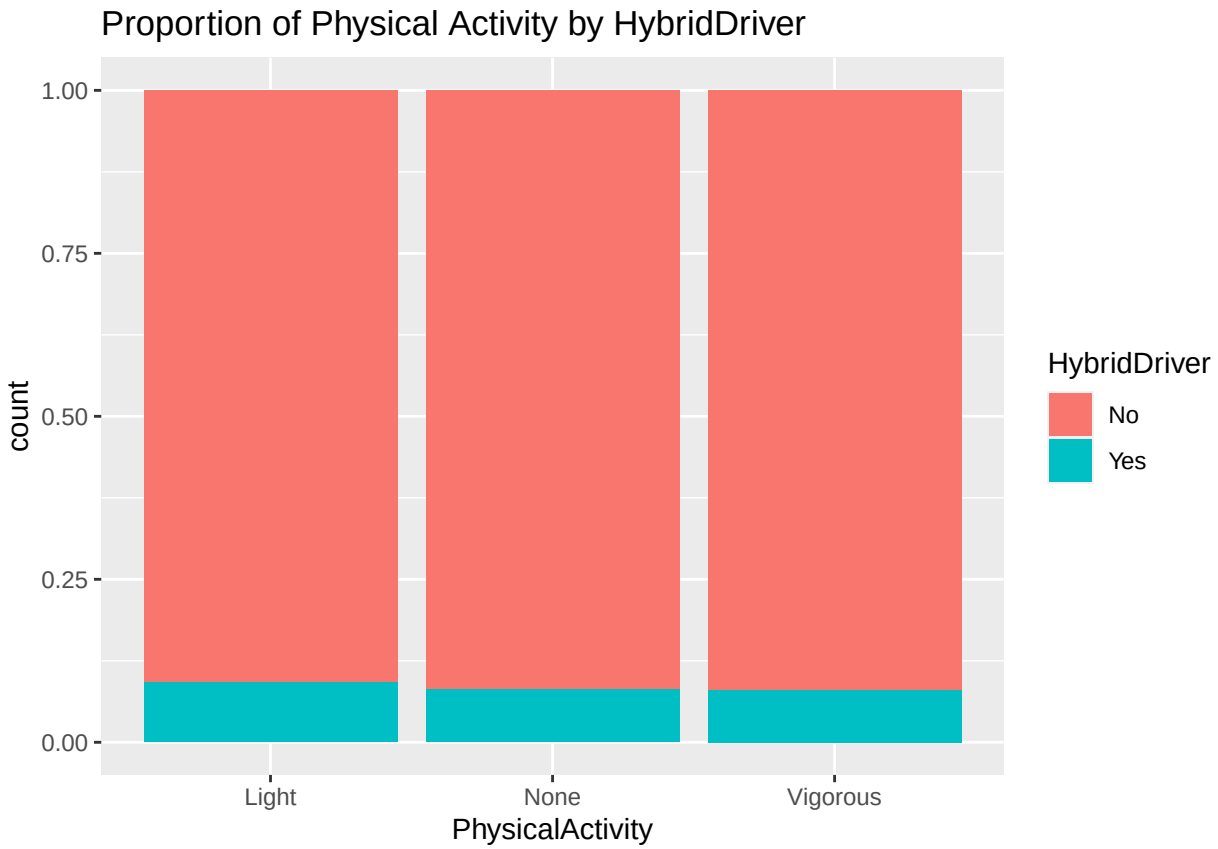


2. **Explore the relationships between predictors and the response variable**: We can use visualizations to explore the relationships between each predictor variable and the response variable. For example, we can create bar charts for categorical variables and box plots for continuous variables, grouped by the levels of `HybridDriver`.

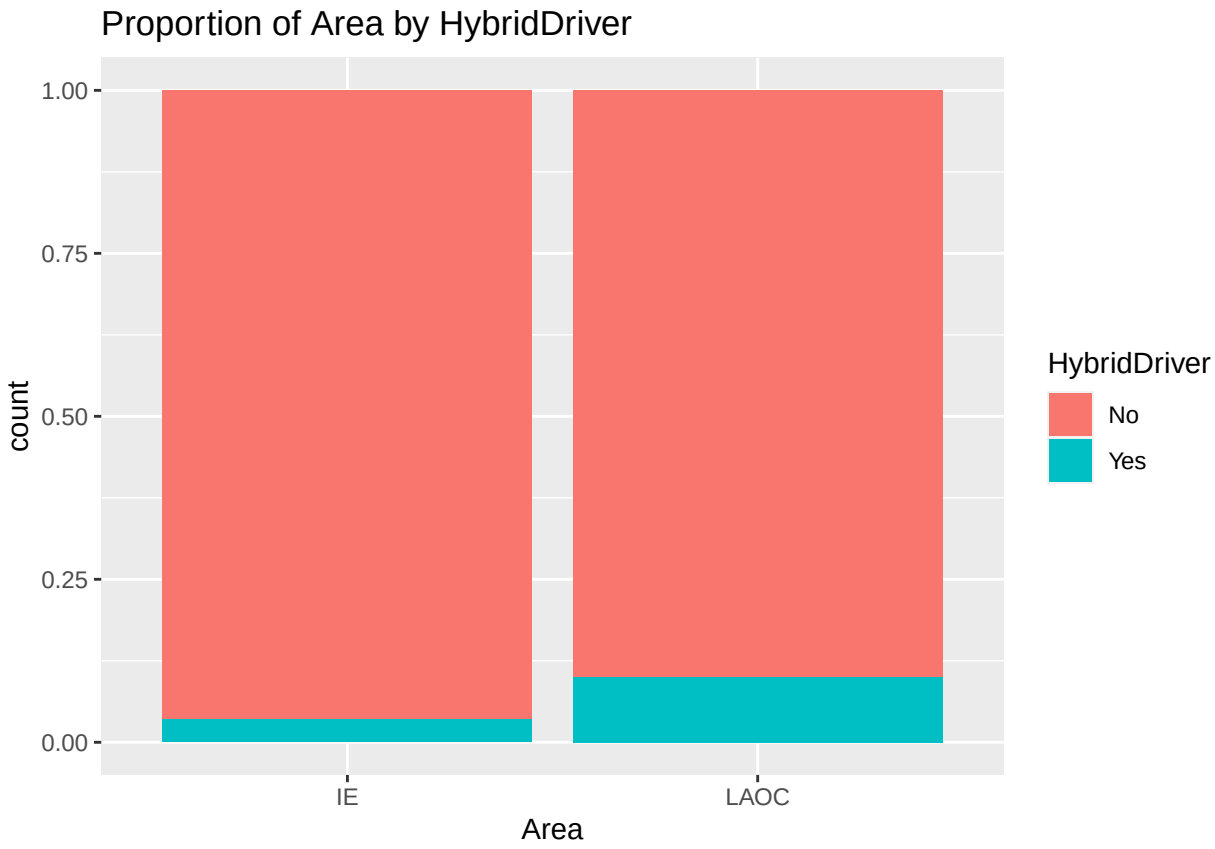
```
# Bar chart of Education by HybridDriver
drivers_train %>%
  ggplot(aes(x = Education, fill = HybridDriver)) +
  geom_bar(position = "dodge") +
  labs(title = "Distribution of Education by HybridDriver")
```



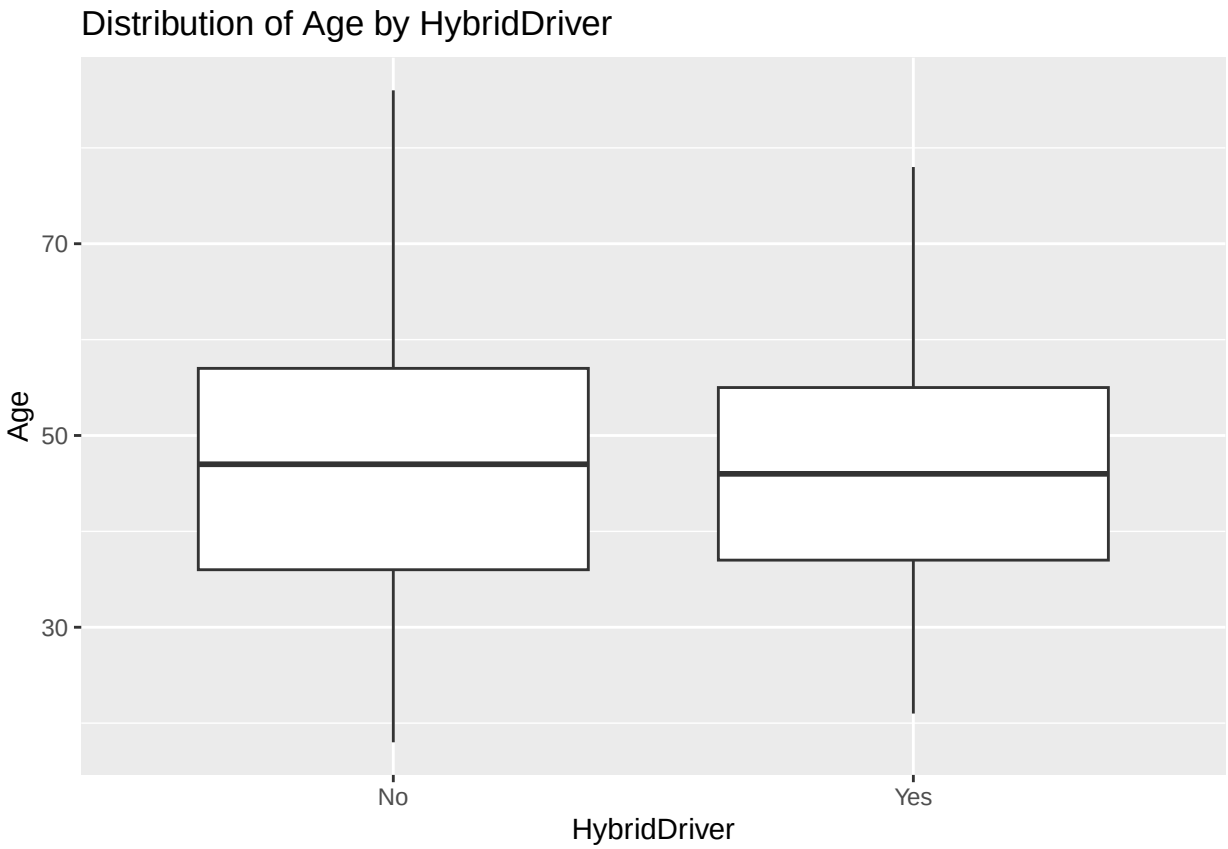

```
# Proportion of Physical Activity by Hybrid Driver
drivers_train %>%
  ggplot(aes(x = PhysicalActivity, fill = HybridDriver)) +
  geom_bar(position = "fill") +
  labs(title = "Proportion of Physical Activity by HybridDriver")
```




```
# Proportion of Area by Hybrid Driver
drivers_train %>%
  ggplot(aes(x = Area, fill = HybridDriver)) +
  geom_bar(position = "fill") +
  labs(title = "Proportion of Area by HybridDriver")
```



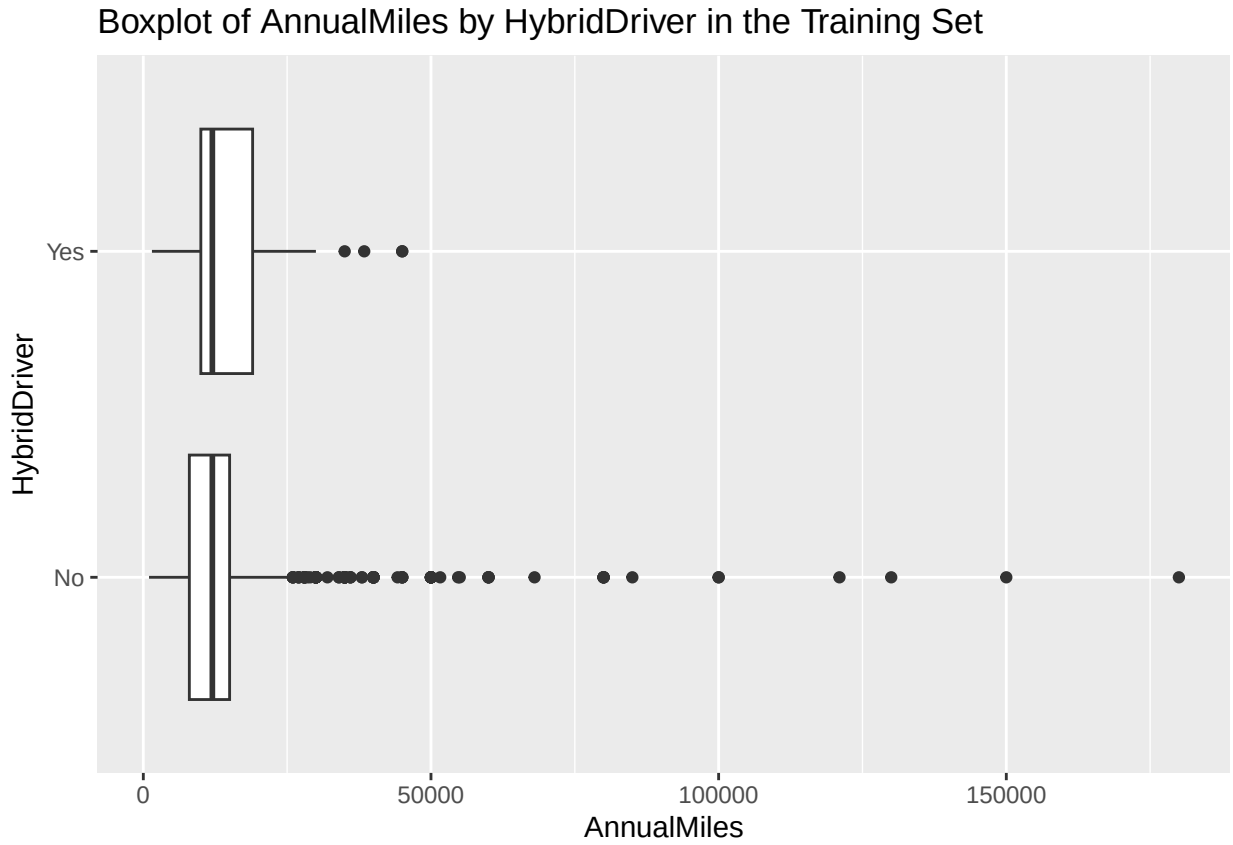

```
# Box plot of Age by HybridDriver
drivers_train %>%
  ggplot(aes(x = HybridDriver, y = Age)) +
  geom_boxplot() +
  labs(title = "Distribution of Age by HybridDriver")
```



These plots can help us identify which predictor variables may be important for predicting `HybridDriver`. For example, we can see that drivers with graduate or professional degrees are more likely to have hybrid/electric vehicles than those with less education. As for the second plot, physical activity has little say in making a difference in whether a person drives a hybrid or electric vehicle. For the 3rd plot, we can see that there's a slightly higher proportion of residents in Los Angeles/Long Beach/Anaheim that own a Hybrid than people in Riverside/San Bernardino/Ontario. We can also see that younger drivers tend to have hybrid/electric vehicles more than older drivers.

Bivariate Analysis of Quantitative Predictor Variables with Response Variable

```
# Bivariate analysis of 'AnnualMiles' and 'HybridDriver'
ggplot(drivers_train, aes(x = AnnualMiles, y = HybridDriver)) +
  geom_boxplot() +
  labs(title = "Boxplot of AnnualMiles by HybridDriver in the Training Set",
       x = "AnnualMiles",
       y = "HybridDriver")
```

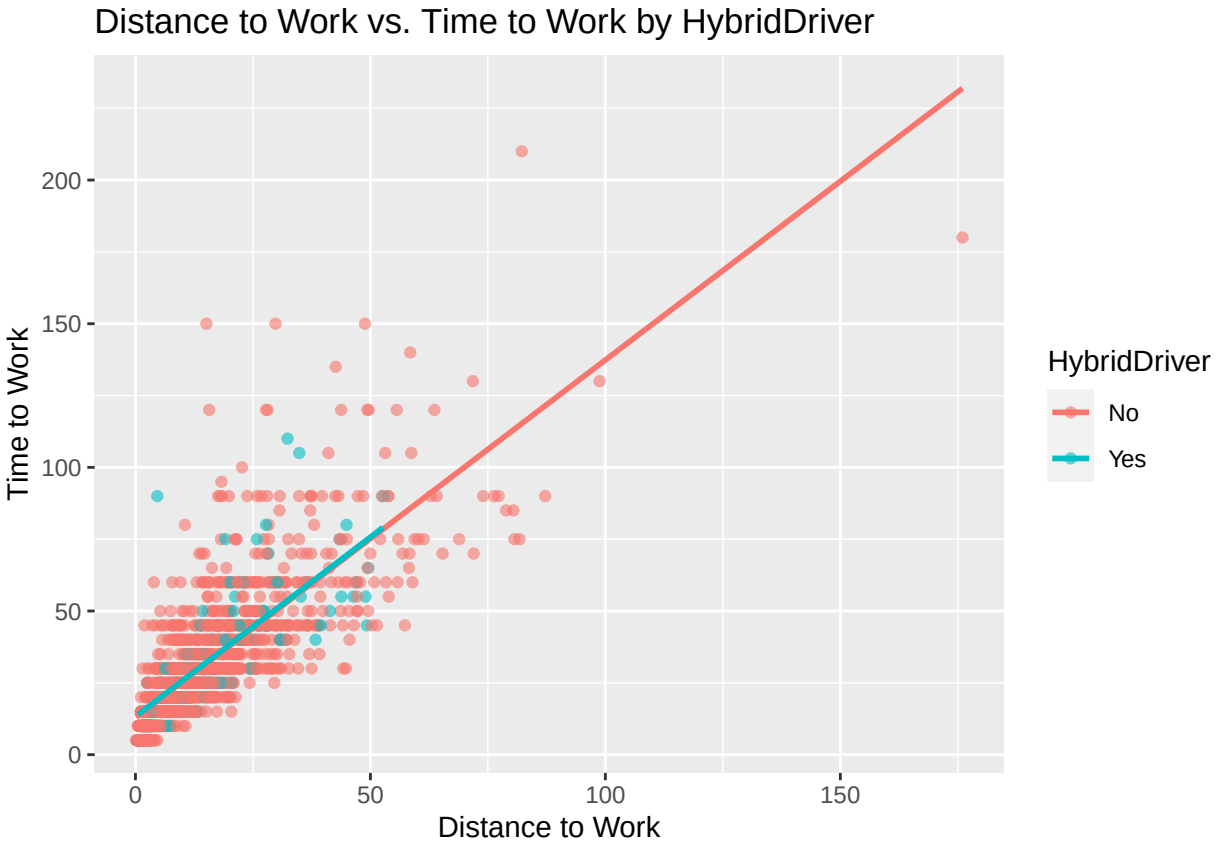


From the bivariate analysis, we can see that the numerical variable 'AnnualMiles' has some differences in their distribution between whether the driver drives a hybrid/electric vehicle or not. We can see there's more outliers to the right for drivers who don't drive a Hybrid but this can also be because their sample size is over 10x more than Hybrid drivers.

3. **Explore interactions between predictors**: We can also explore interactions between predictors by creating scatter plots and examining how the relationship between two predictors changes depending on the level of the response variable. For example, we can create a scatter plot of `DistanceToWork` versus `TimeToWork`, colored by the levels of `HybridDriver`.

```
# Scatter plot of Distance to Work vs. Time To Work by HybridDriver
ggplot(drivers_train, aes(x = DistanceToWork, y = TimeToWork, color = HybridDriver)) +
  geom_point(alpha=0.6) +
  geom_smooth(method = "lm", se = FALSE) +
  labs(title="Distance to Work vs. Time to Work by HybridDriver", x = "Distance to Work", y = "Time to
```

```
## `geom_smooth()` using formula = 'y ~ x'
```



We can see that drivers with hybrid/electric vehicles tend to have shorter distances and times to work compared to those without hybrid/electric vehicles. As far as we can see, the interaction between Distance to Work vs. Time to Work doesn't have a significant difference in the rate of the slope, however, the distance between the slope signifies that the farther distance between the driver's home and work, the more likely the driver doesn't drive a Hybrid vehicle.

Based on these exploratory analyses, we can engineer new features and select relevant predictor variables to use in our model. Here are some ideas for feature engineering:

1. **Create indicator variables for categorical variables**: We can create indicator variables for the categorical variables `Education`, `Gender`, `PhysicalActivity`, `Area`, and `Urban`. This allows the model to capture any non-linear relationships between these variables and the response variable.

2. **Create interaction terms**: We can create interaction terms between pairs of predictor variables to capture any non-additive relationships between them. For example, we can create an interaction term between `DistanceToWork` and `TimeToWork` to capture the effect of a longer commute time on the likelihood of having a hybrid/electric vehicle.

3. **Transform predictor variables**: We can transform `Age` using a non-linear function, such as a quadratic or cubic function, to capture any non-linear relationship between age and the likelihood of having a hybrid/electric vehicle.

4. **Drop irrelevant predictor variables**: We can drop `HouseID` and `PersonID` and other variables since they do not provide any useful information for predicting `HybridDriver`. All they do is list the number but have no significance in giving us a better model to predict whether a person is a Hybrid driver or not.

Part b (Code: 6 pts)

Fit at least three different types of classification models on the training set. One model must be from the set of models learned in Chapters 2 and 4 (k-nearest neighbors, logistic regression, LDA, QDA, or naive Bayes) and one model must be from the set of models learned in Chapters 8 and 10 (decision trees, bagging/random forests, boosted trees, or neural networks). The remaining model(s) can be any classification model we have covered in this class.

You do not need to tune any models yet.

To fit three different types of classification models on the training set, we can choose one model from the set of models learned in Chapters 2 and 4, another model from the set of models learned in Chapters 8 and 10, and the remaining model can be any classification model covered in this class. Here's an example of fitting three models: logistic regression, bagging/random forest, and LDA:

```
# Create indicator variables for categorical variables
drivers_train <- drivers_train %>%
  mutate(Education = as.factor(Education),
         Gender = as.factor(Gender),
         PhysicalActivity = as.factor(PhysicalActivity),
         Area = as.factor(Area),
         Urban = as.factor(Urban))

# Create interaction term between DistanceToWork and TimeToWork
drivers_train <- drivers_train %>%
  mutate(Distance_Time_Interaction = DistanceToWork * TimeToWork)

# Transform Age using a non-linear function (cubic function)
drivers_train <- drivers_train %>%
  mutate(Age_Cubic = Age^3)

# Fit the logistic regression model
log_reg_model <- glm(HybridDriver ~ Education + Gender + PhysicalActivity + Area + Urban +
                       Distance_Time_Interaction + Age_Cubic, data = drivers_train, family = binomial)

# Print the summary of the model
summary(log_reg_model)
```

```
##
## Call:
## glm(formula = HybridDriver ~ Education + Gender + PhysicalActivity +
##     Area + Urban + Distance_Time_Interaction + Age_Cubic, family = binomial,
##     data = drivers_train)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -0.7471  -0.4816  -0.3859  -0.2806   2.8840
##
## Coefficients:
##                                          Estimate Std. Error z value Pr(>|z|)
## (Intercept)                             -3.714e+00  6.922e-01  -5.366 8.04e-08
## EducationGraduate or Professional Degree  6.410e-01  2.173e-01   2.950  0.00318
## EducationHigh School Graduate           -5.875e-01  4.526e-01  -1.298  0.19420
## EducationLess than High School          -1.413e+01  5.045e+02  -0.028  0.97766
## EducationSome College                   -2.165e-01  2.694e-01  -0.804  0.42168
```

```
## GenderMale                                4.089e-01  1.899e-01   2.154  0.03127
## PhysicalActivityNone                     -8.867e-02  3.188e-01  -0.278  0.78092
## PhysicalActivityVigorous                 -2.786e-01  2.130e-01  -1.308  0.19087
## AreaLAOC                                  9.876e-01  3.380e-01   2.922  0.00348
## UrbanSuburban                             5.107e-01  6.289e-01   0.812  0.41676
## UrbanUrban                                3.813e-01  6.184e-01   0.617  0.53755
## Distance_Time_Interaction                 6.128e-05  5.930e-05   1.033  0.30140
## Age_Cubic                                -1.665e-06  1.003e-06  -1.660  0.09701
##
## (Intercept)                              ***
## EducationGraduate or Professional Degree **
## EducationHigh School Graduate
## EducationLess than High School
## EducationSome College
## GenderMale                               *
## PhysicalActivityNone
## PhysicalActivityVigorous
## AreaLAOC                                 **
## UrbanSuburban
## UrbanUrban
## Distance_Time_Interaction
## Age_Cubic                                .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 903.11  on 1525  degrees of freedom
## Residual deviance: 855.48  on 1513  degrees of freedom
## AIC: 881.48
##
## Number of Fisher Scoring iterations: 15
```

```
# Fit Random Forest
drivers_train$HybridDriver <- as.factor(drivers_train$HybridDriver)
rf_model <- randomForest(HybridDriver ~ Education + Gender + PhysicalActivity + Area + Urban +
                       Distance_Time_Interaction + Age_Cubic, data = drivers_train)
print(rf_model)
```

```
##
## Call:
##  randomForest(formula = HybridDriver ~ Education + Gender + PhysicalActivity +      Area + Urban + D
##                Type of random forest: classification
##                      Number of trees: 500
## No. of variables tried at each split: 2
##
##         OOB estimate of  error rate: 8.72%
## Confusion matrix:
##       No Yes class.error
## No  1393   0           0
## Yes  133   0           1
```

```r
# Fit LDA model on training set
lda_model <- lda(HybridDriver ~ Education + Gender + PhysicalActivity + Area + Urban +
                      Distance_Time_Interaction + Age_Cubic, data = drivers_train)
lda_model
```

```
## Call:
## lda(HybridDriver ~ Education + Gender + PhysicalActivity + Area +
##     Urban + Distance_Time_Interaction + Age_Cubic, data = drivers_train)
##
## Prior probabilities of groups:
##         No        Yes
## 0.91284404 0.08715596
##
## Group means:
##     EducationGraduate or Professional Degree EducationHigh School Graduate
## No                                 0.2706389                    0.10409189
## Yes                                0.4586466                    0.04511278
##     EducationLess than High School EducationSome College GenderMale
## No                      0.01579325             0.2849964  0.5211773
## Yes                     0.00000000             0.1879699  0.6090226
##     PhysicalActivityNone PhysicalActivityVigorous  AreaLAOC UrbanSuburban
## No            0.10552764               0.2972003 0.7832017     0.2871500
## Yes           0.09774436               0.2706767 0.9172932     0.3082707
##     UrbanUrban Distance_Time_Interaction Age_Cubic
## No   0.6669060                  716.3561  127616.2
## Yes  0.6691729                  768.7470  117504.1
##
## Coefficients of linear discriminants:
##                                                    LD1
## EducationGraduate or Professional Degree  1.281535e+00
## EducationHigh School Graduate            -6.237938e-01
## EducationLess than High School           -1.773200e+00
## EducationSome College                    -2.861421e-01
## GenderMale                                6.791074e-01
## PhysicalActivityNone                     -1.139220e-01
## PhysicalActivityVigorous                 -4.519444e-01
## AreaLAOC                                  1.169426e+00
## UrbanSuburban                             4.803646e-01
## UrbanUrban                                2.697077e-01
## Distance_Time_Interaction                 9.183168e-05
## Age_Cubic                                -2.582064e-06
```

In the code above, we first fit a logistic regression model using the `glm` function from base R. We specify the formula `HybridDriver ~ Education + Gender + PhysicalActivity + Area + Urban + Distance_Time_Interaction + Age_Cubic` to indicate the relationship between the predictors and the response variable.

Next, we fit a random forest model using the `randomForest` function. Again, we use the same formula for the relationship between the predictors and the response variable.

Finally, we fit an LDA model using the same predictors as the logistic regression. All of these models are fitted on the training set 'drivers_train'.

Part c (Code: 4 pts; Explanation: 2 pts)

Using cross-validation, do any necessary tuning on your models from part (b) and select a final model. Explain why you think it is the best model. You should use either mean log-loss or Brier Score to evaluate your models.

```
# Set seed
set.seed(123)

# Create a 10-fold cross-validation object
cv <- trainControl(method = "cv", number = 10)

# Fit logistic regression model with cross-validation
logistic_model_cv <- train(HybridDriver ~ Education + Gender + PhysicalActivity + Area + Urban +
                        Distance_Time_Interaction + Age_Cubic, data = drivers_train, method = "glm", fami
logistic_accuracy <- logistic_model_cv$results$Accuracy
cat("Logistic Regression Accuracy:", logistic_accuracy, "\n")
```

```
## Logistic Regression Accuracy: 0.9128518
```

```
# Fit random forest model with cross-validation
rf_model_cv <- train(HybridDriver ~ Education + Gender + PhysicalActivity + Area + Urban +
                        Distance_Time_Interaction + Age_Cubic, data = drivers_train, method = "rf", trCon
rf_accuracy <- rf_model_cv$results$Accuracy
cat("Random Forest Accuracy:", rf_accuracy, "\n")
```

```
## Random Forest Accuracy: 0.9128518 0.9076016 0.9062901
```

```
# Fit LDA model with cross-validation
lda_model_cv <- train(HybridDriver ~ Education + Gender + PhysicalActivity + Area + Urban +
                        Distance_Time_Interaction + Age_Cubic, data = drivers_train, method = "lda", trCo
lda_accuracy <- lda_model_cv$results$Accuracy
cat("LDA Accuracy:", lda_accuracy, "\n")
```

```
## LDA Accuracy: 0.9128518
```

Our highest Accuracy out of the three models that we cross-validated is the logistic regression model with an Accuracy of 0.9128518.

Part d (Code: 5 pts; Explanation: 4 pts)

Using your final model from part (c), make probabilistic predictions on the test set. That is, you should obtain the predicted probability of driving a hybrid rather than (or in addition to) class predictions.

Obtain the confusion matrix at the estimated Bayes decision boundary. How well does your model predict people who drive hybrid/electric cars? How well does your model predict people who *do not* drive those type of cars?

Also, obtain and plot the ROC curve. Can you improve your model's predictions by choosing a different probability threshold? Explain your reasoning.

HINT: If you use `yardstick` to obtain the ROC curve, you will get out a tibble in which one of the columns is named `.threshold`.

For part (c), you have already fit the logistic regression model. Now we can use this model to make probabilistic predictions on the test set and obtain the confusion matrix at the estimated Bayes decision boundary using the `conf_mat` function.

```
# Create indicator variables for categorical variables in the test set
drivers_test <- drivers_test %>%
  mutate(Education = as.factor(Education),
         Gender = as.factor(Gender),
         PhysicalActivity = as.factor(PhysicalActivity),
         Area = as.factor(Area),
         Urban = as.factor(Urban))

# Create interaction term between DistanceToWork and TimeToWork in the test set
drivers_test <- drivers_test %>%
  mutate(Distance_Time_Interaction = DistanceToWork * TimeToWork)

# Transform Age using a non-linear function (cubic function) in the test set
drivers_test <- drivers_test %>%
  mutate(Age_Cubic = Age^3)

# Make probabilistic predictions on the test set
test_preds <- predict(log_reg_model, newdata = drivers_test, type = "response")

# Obtain class predictions using estimated Bayes decision boundary (0.5)
test_preds_class <- ifelse(test_preds >= 0.5, "Yes", "No")

# Create a confusion matrix
conf_matrix <- table(Actual = drivers_test$HybridDriver, Predicted = test_preds_class)

# Print the confusion matrix
conf_matrix
```

```
##       Predicted
## Actual  No
##    No  463
##    Yes  45
```

```
summary(conf_matrix, event_level = "first")
```

```
## Number of cases in table: 508
## Number of factors: 2
## Test for independence of all factors:
##  Chisq = 0, df = 0, p-value = 1
```

The code above makes probabilistic predictions on the test set using the `predict` function and specifying `type = "response"`. Then, the probabilities are converted to class predictions using the estimated Bayes decision boundary, which is 0.5 in this case. The `ifelse` statement is used to create a vector of "Yes" and "No" class labels based on whether the predicted probability is greater than or equal to 0.5 or not.

The confusion matrix is comparing the predicted class labels to the actual class labels in the test set. We can see that our conf matrix is $463/508 = 91.14\%$ accurate.

ROC Curve

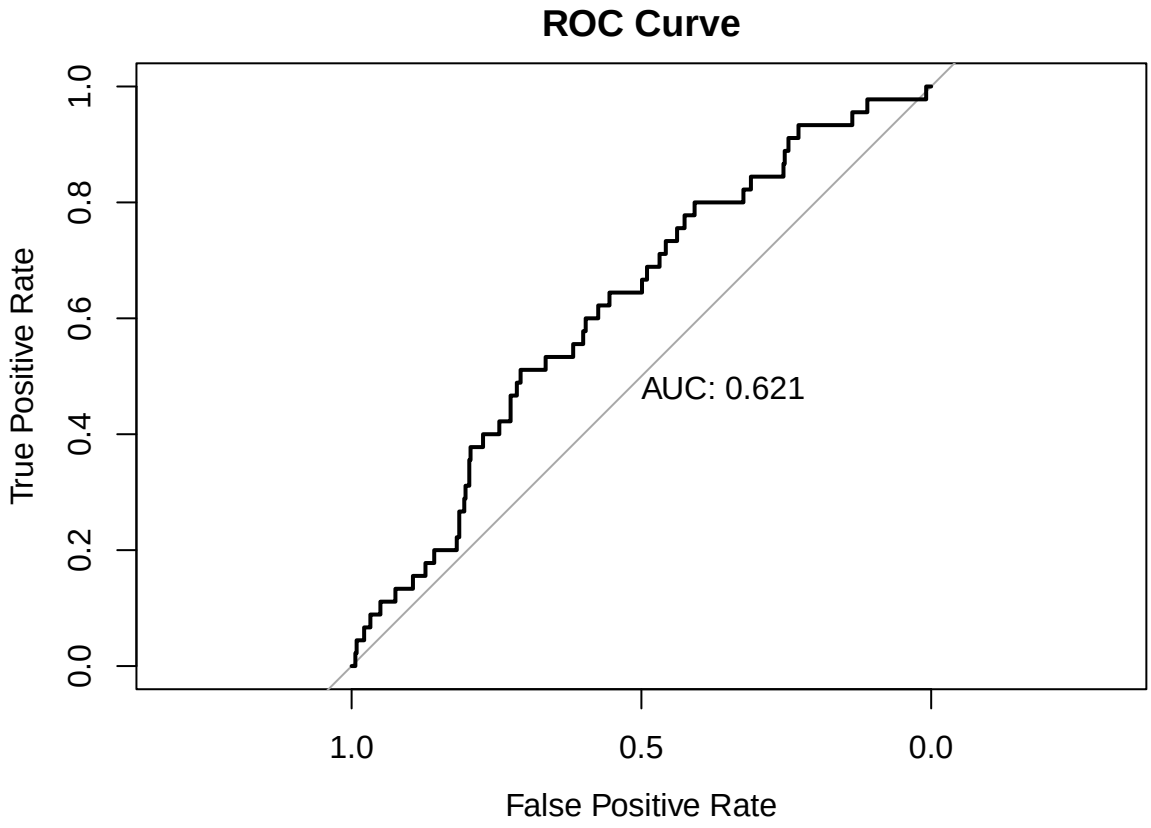
To obtain and plot the ROC curve for the logistic regression model, you can use the `pROC` package.

```r
# Obtain predicted probabilities on the test set
test_preds <- predict(log_reg_model, newdata = drivers_test, type = "response")

# Create a ROC object
roc_obj <- roc(drivers_test$HybridDriver, test_preds)
```

```
## Setting levels: control = No, case = Yes
```

```
## Setting direction: controls < cases
```

```r
# Plot the ROC curve
plot(roc_obj, main = "ROC Curve", xlab = "False Positive Rate", ylab = "True Positive Rate", print.auc =
```



The code uses the `roc()` function from the `pROC` package to create a ROC object by providing the actual class values (`drivers_test$HybridDriver`) and the predicted probabilities (`test_preds`). The resulting ROC object is then plotted using the `plot()` function.

The ROC curve visually represents the trade-off between the true positive rate (sensitivity) and the false positive rate (1-specificity) for different probability thresholds. The area under the ROC curve (AUC) is a measure of the model's predictive performance, where a higher AUC indicates better performance. Since our AUC is 0.621, this is better than just flipping a coin.

To improve the model's predictions, you can choose a different probability threshold based on the ROC curve. By adjusting the threshold, you can prioritize either maximizing sensitivity or specificity, depending on the specific requirements of your application.

For example, if you want to prioritize high sensitivity (lowering the false negative rate), you can choose a threshold that corresponds to a point on the ROC curve that is closer to the top-left corner (higher true positive rate). Conversely, if you want to prioritize high specificity (lowering the false positive rate), you can choose a threshold that corresponds to a point on the ROC curve that is closer to the top-right corner (higher true negative rate).

By analyzing the ROC curve and considering the specific needs of your application, you can select an optimal probability threshold that balances the trade-off between sensitivity and specificity to improve the model's predictions.

Oral Exam (10 points)

Sign up on Canvas for a 15-minute window in which we will talk for about 10 minutes about your exam solutions.

You will receive 5 points for showing up and up to 5 points of extra credit based on the clarity of your explanations.