Self Reflection - STA 631

Madan K C

2023-04-28

Objective 1: Describe probability as a foundation of statistical modeling, including inference and maximum likelihood estimation

Probability is essential to statistical modeling because it provides a way to quantify uncertainty and randomness. In statistical inference, probability is used to construct confidence intervals or hypothesis tests to make conclusions about a population based on a sample. Maximum likelihood estimation is a common method of estimating the parameters of a statistical model, and it involves finding the parameter values that make the observed data most likely to have occurred based on probability distributions.

1.1 Simple Linear Regression of Human Freedom Index dataset:

Loading Human Freedom Index dataset from URL and assigning it into a dataframe named hfi:

```r
hfi <- readr::read_csv("https://www.openintro.org/data/csv/hfi.csv")
```

```
## Rows: 1458 Columns: 123
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr   (3): ISO_code, countries, region
## dbl (120): year, pf_rol_procedural, pf_rol_civil, pf_rol_criminal, pf_rol, p...
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```r
head(hfi)
```

```
## # A tibble: 6 x 123
##    year ISO_code countries region pf_rol_procedural pf_rol_civil pf_rol_criminal
##   <dbl> <chr>    <chr>     <chr>              <dbl>        <dbl>           <dbl>
## 1  2016 ALB      Albania   Easte~              6.66         4.55            4.67
## 2  2016 DZA      Algeria   Middl~             NA           NA              NA
## 3  2016 AGO      Angola    Sub-S~             NA           NA              NA
## 4  2016 ARG      Argentina Latin~              7.10         5.79            4.34
## 5  2016 ARM      Armenia   Cauca~             NA           NA              NA
## 6  2016 AUS      Australia Ocean~              8.44         7.53            7.36
## # i 116 more variables: pf_rol <dbl>, pf_ss_homicide <dbl>,
## #   pf_ss_disappearances_disap <dbl>, pf_ss_disappearances_violent <dbl>,
## #   pf_ss_disappearances_organized <dbl>,
## #   pf_ss_disappearances_fatalities <dbl>, pf_ss_disappearances_injuries <dbl>,
## #   pf_ss_disappearances <dbl>, pf_ss_women_fgm <dbl>,
## #   pf_ss_women_missing <dbl>, pf_ss_women_inheritance_widows <dbl>,
## #   pf_ss_women_inheritance_daughters <dbl>, pf_ss_women_inheritance <dbl>, ...
```

1. What are the dimensions of the dataset? What does each row represent?

```
dim(hfi) # shows the dimensions of the dataset, which is 1458*123
```

```
## [1] 1458  123
```

The dim() shows the dimensions of the hfi data frame, which is 1458*123.

The dataset spans a lot of years. We are only interested in data from year 2016.

Create a new R code chunk. Filter the data hfi data frame for year 2016, and Assign the result to a data frame named hfi_2016.

```
library(dplyr)
```

```
##
## Attaching package: 'dplyr'
```

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

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

```
hfi_2016 <- filter(hfi, year == 2016)
```

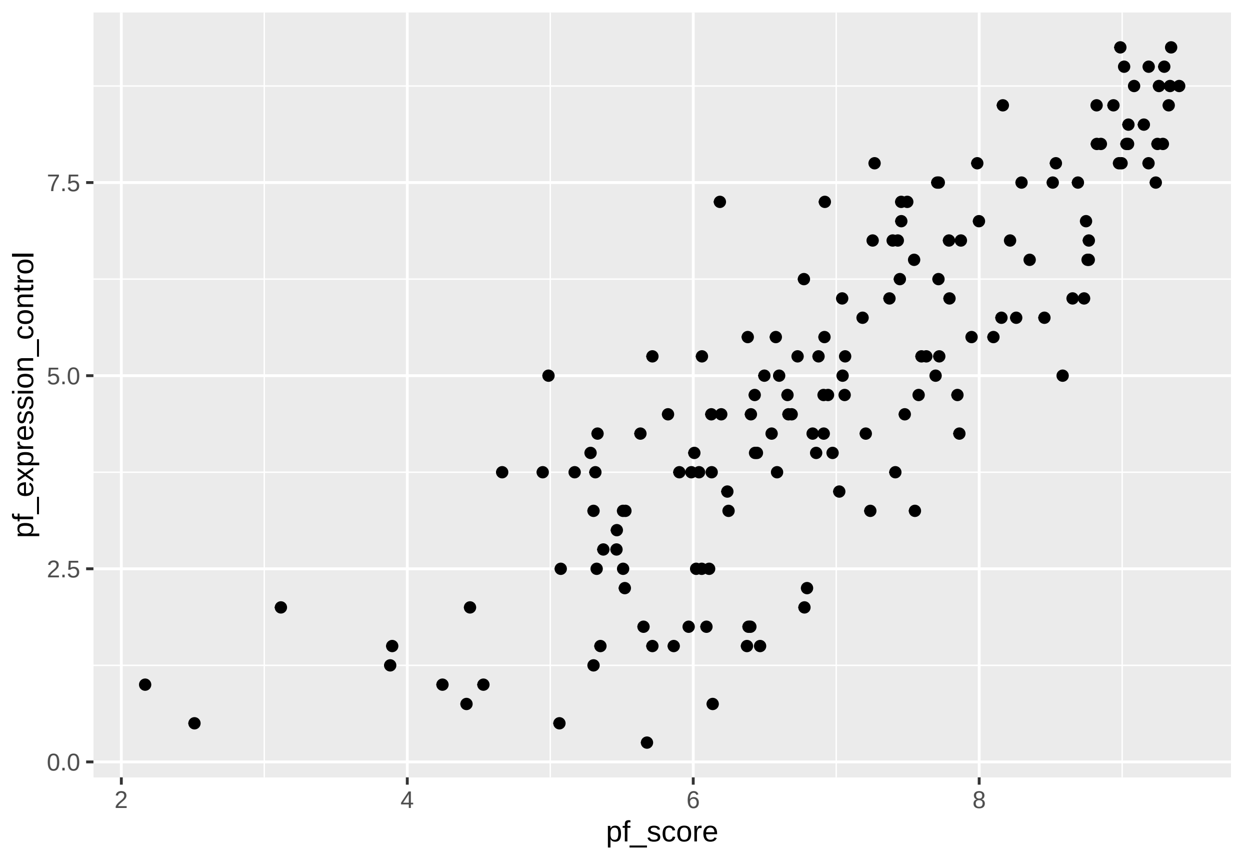
Alternative method using subset to filter HFI data frame for the year 2016

```
hfi_2016 <- subset(hfi, year == 2016)
```

2. What type of plot would you use to display the relationship between the personal freedom score, pf_score, and pf_expression_control? Create a new R code chunk and plot this relationship using the variable pf_expression_control as the predictor.

=> To display the relationship between the personal freedom score and pf_expression_control, we can use a scatter plot with pf_expression_control on the x-axis and pf_score on the y-axis.

```
library(ggplot2)
ggplot(hfi_2016, aes(x = pf_score, y = pf_expression_control)) + geom_point() # In the hfi_2016 data fr
```



3. Does the relationship look linear? If you knew a country's pf_expression_control, or its score out of 10, with 0 being the most, of political pressures and controls on media content, would you be comfortable using a linear model to predict the personal freedom score?

=> From the scatter plot above, we can say that the relationship between pf_expression_control and pf_score is linear. In such scenario, I think if the relationship is linear then linear model can be used to predict the personal freedom score. However, if it is not linear, then we have to consider the variables.

Sum of squared residuals

4. Using statsr::plot_ss, choose a line that does a good job of minimizing the sum of squares. Run the function several times. What was the smallest sum of squares that you got? How does it compare to your neighbour's?

=> The smallest sum of squares that I got is :

Sum of Squares: 102.213

The linear model

```
m1 <- lm(pf_score ~ pf_expression_control, data = hfi_2016)
m1
```

```
##
## Call:
## lm(formula = pf_score ~ pf_expression_control, data = hfi_2016)
##
## Coefficients:
##           (Intercept)  pf_expression_control
##                4.2838                 0.5418
```

The first argument in the function lm() is a formula that takes the form y ~ x. Here it can be read that we want to make a linear model of pf_score as a function of pf_expression_control. The second argument specifies that R should look in the hfi data frame to find the two variables.

The output of lm is an object that contains all of the information we need about the linear model that was just fit. We can access this information using broom::tidy

```
broom::tidy(m1)
```

```
## # A tibble: 2 x 5
##   term                  estimate std.error statistic  p.value
##   <chr>                    <dbl>     <dbl>     <dbl>    <dbl>
## 1 (Intercept)              4.28     0.149       28.8 4.23e-65
## 2 pf_expression_control    0.542    0.0271      20.0 2.31e-45
```

Let's consider this output piece-by-piece. First, the formula used to describe the model is shown at the top, in what's displayed as the "Call". After the formula you find the five-number summary of the residuals. The "Coefficients" table shown next is key; its first column displays the linear model's y-intercept and the coefficient of pf_expression_control. With this table, we can write down the least squares regression line for the linear model:

$$\hat{y} = 4.28 + 0.542 \times pf\_expression\_control$$

Using this equation…

5. Interpret the y-intercept. => The y-intercept of the regression line is 3.725. This means that if the value of pf_expression_control is zero, the predicted value of pf_score would be 3.725.
6. Interpret the slope => The slope of the regression line is 0.606. This means that for every one unit increase in pf_expression_control, the predicted value of pf_score would increase by 0.606 units on average, holding all other variables constant.

Overall model fit

Using your {dplyr} skills, obtain the correlation coefficient between pf_expression_control and pf_score.

```
correlation <- cor(hfi_2016$pf_expression_control, hfi_2016$pf_score)
correlation
```

```
## [1] 0.8450646
```

1. What does this value mean in the context of this model? The value of the correlation coefficient indicates the strength and direction of the linear relationship between the two variables. In this case, a positive correlation coefficient (0.8450646) indicates that higher levels of pf_expression_control are associated with higher levels of pf_score.

```
broom::glance(m1)
```

```
## # A tibble: 1 x 12
##   r.squared adj.r.squared sigma statistic  p.value    df logLik   AIC   BIC
##       <dbl>         <dbl> <dbl>     <dbl>    <dbl> <dbl>  <dbl> <dbl> <dbl>
## 1     0.714         0.712 0.799      400. 2.31e-45     1  -193.  391.  400.
## # i 3 more variables: deviance <dbl>, df.residual <int>, nobs <int>
```

2. What is the value of for this model?

=> The output of glance(m1) provides information on the model fit, including the R-squared value. The R-squared value of the model is 0.7141342 and the adjusted R-squared value is 0.7123476

3. What does this value mean in the context of this model?

=> R-squared represents the proportion of variability in the response variable that is explained by the explanatory variable. In this case, an R-squared value of 0.7141342 means that approximately 71.4% of the variability in pf_score is explained by the linear relationship with pf_expression_control.

4. Fit a new model that uses pf_expression_control to predict hf_score, or the total human freedom score. Using the estimates from the R output, write the equation of the regression line. What does the slope tell us in the context of the relationship between human freedom and the amount of political pressure on media content?

```
m2 <- lm(hf_score ~ pf_expression_control, data = hfi_2016)
summary(m2)
```

```
##
## Call:
## lm(formula = hf_score ~ pf_expression_control, data = hfi_2016)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.68164 -0.45467  0.05692  0.46699  1.88128
##
## Coefficients:
##                       Estimate Std. Error t value Pr(>|t|)
## (Intercept)            5.05340    0.12293   41.11   <2e-16 ***
## pf_expression_control  0.36843    0.02236   16.48   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.6595 on 160 degrees of freedom
## Multiple R-squared:  0.6291, Adjusted R-squared:  0.6268
## F-statistic: 271.4 on 1 and 160 DF,  p-value: < 2.2e-16
```

=> The equation of the regression line for the new model is:

hf_score = 5.0534 + 0.36843 * pf_expression_control

This means that for every 1 unit increase in pf_expression_control, we expect a country's mean human freedom score to increase by 0.36843 units, on average. The intercept (5.0534) represents the estimated mean human freedom score for countries with a pf_expression_control value of 0. The slope (0.36843) represents the estimated change in the mean human freedom score for a 1-unit increase in pf_expression_control.

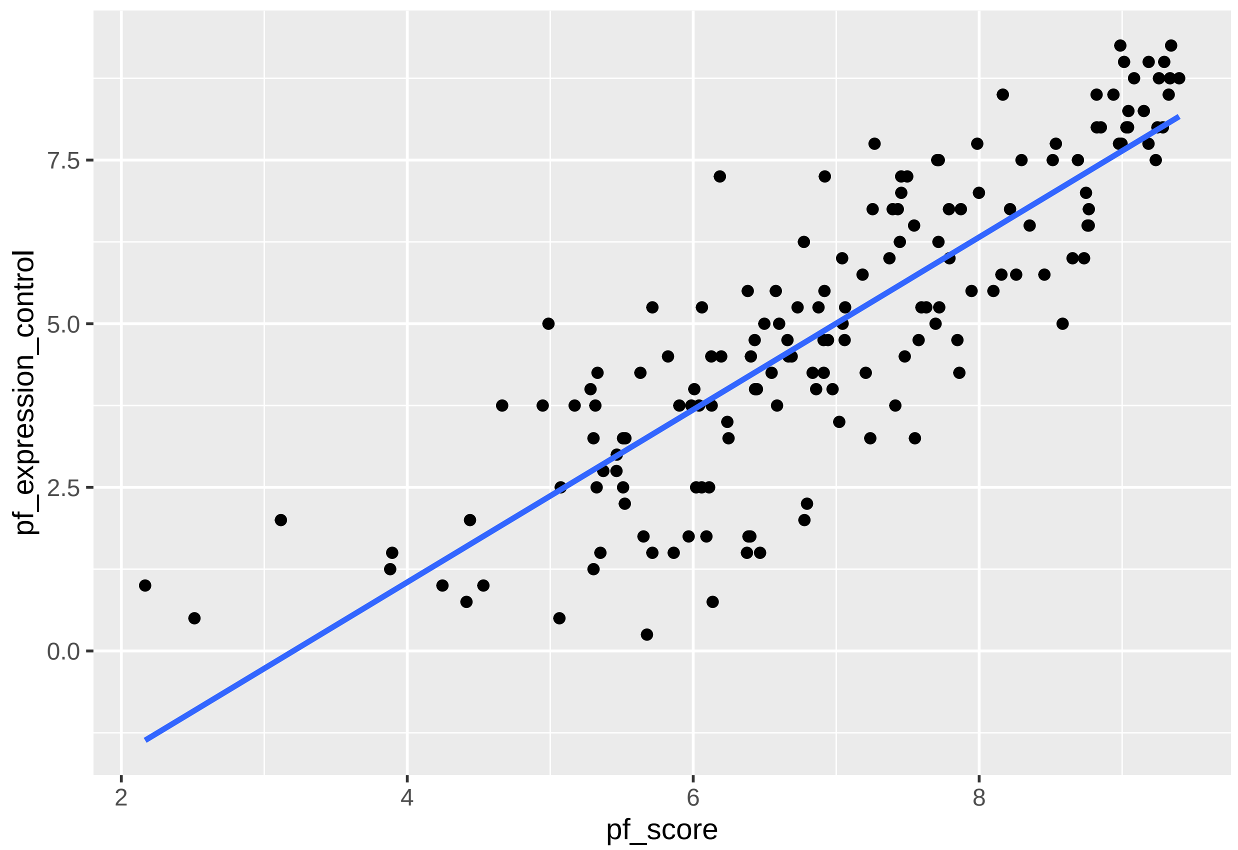
Prediction and prediction errors

I will first create a scatterplot with the least squares line for m1 laid on top.

Copy-and-paste the entire code chunk of the scatterplot you created in Day 1 of this activity below. Add a layer to this (remember how {ggplot2} represents adding various data layers to plots) that shows a smooth line geometry. In this layer, be sure to specify the method as "lm" and do not display confidence intervals around your bands (hint: look at the help documentation for the layer you added).

```
library(ggplot2)
ggplot(hfi_2016, aes(x = pf_score, y = pf_expression_control)) +
  geom_point() +
  geom_smooth(method = "lm", se = FALSE)
```

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



6. If someone saw the least squares regression line and not the actual data, how would they predict a country's personal freedom school for one with a 3 rating for pf_expression_control?

=> To predict a country's personal freedom score for one with a 3 rating for pf_expression_control, we can use the equation of the regression line obtained earlier:

hf_score = 5.05340 + 0.36843 * pf_expression_control

Substituting the value of pf_expression_control as 3, we get:

hf_score = 5.05340 + 0.36843 * 3 = 6.15869

So the predicted hf_score for a country with a 3 rating for pf_expression_control is 6.15869.

7. Is this an overestimate or an underestimate, and by how much? In other words, what is the individual residual value for this prediction?

=> For countries with a pf_expression_control of 0 (those with the largest amount of political pressure on media content), we expect their mean personal freedom score to be 4.28.

For every 1 unit increase in pf_expression_control, we expect a country's mean personal freedom score to increase 0.542 units.

For pf_expression_control as 3, the hf_score is around 4.28 +2*0.542 = 5.364.

As the actual value is lower than the predicted value (5.364<6.15869), this is underestimate.

Model diagnostics

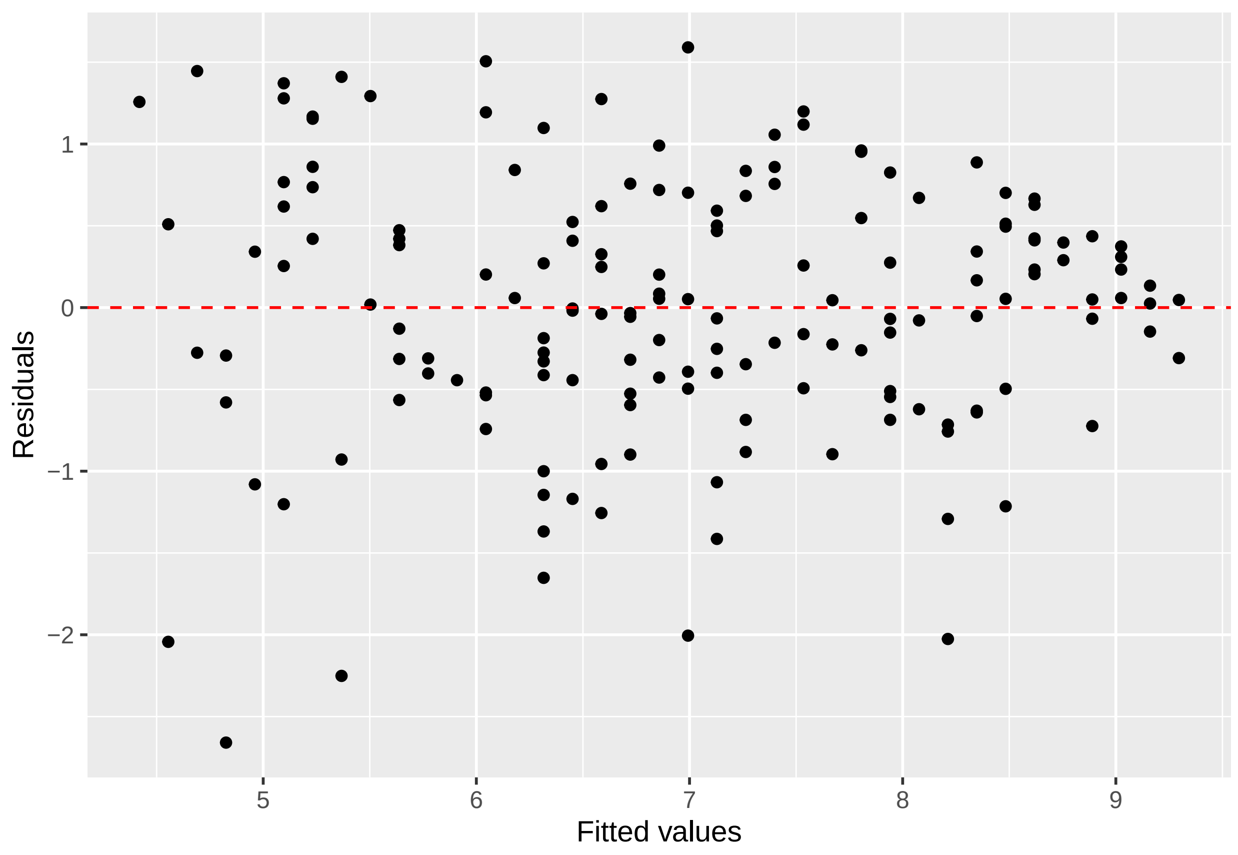
To assess whether the linear model is reliable, we should check for (1) linearity, (2) nearly normal residuals, and (3) constant variability. Note that the normal residuals is not really necessary for all models (sometimes we simply want to describe a relationship for the data that we have or population-level data, where statistical inference is not appropriate/necessary).

In order to do these checks we need access to the fitted (predicted) values and the residuals. We can use broom::augment to calculate these.

```
library(broom)
m1_aug <- augment(m1)
m1_aug
```

```
## # A tibble: 162 x 8
##    pf_score pf_expression_control .fitted .resid    .hat .sigma  .cooksd
##       <dbl>                 <dbl>   <dbl>  <dbl>   <dbl>  <dbl>    <dbl>
##  1     7.60                  5.25    7.13  0.468 0.00625  0.801 0.00108
##  2     5.28                  4       6.45 -1.17  0.00729  0.796 0.00792
##  3     6.11                  2.5     5.64  0.473 0.0133   0.801 0.00239
##  4     8.10                  5.5     7.26  0.836 0.00648  0.799 0.00359
##  5     6.91                  4.25    6.59  0.326 0.00679  0.801 0.000573
##  6     9.18                  7.75    8.48  0.701 0.0150   0.800 0.00594
##  7     9.25                  8       8.62  0.628 0.0166   0.800 0.00531
##  8     5.68                  0.25    4.42  1.26  0.0319   0.795 0.0422
##  9     7.45                  7.25    8.21 -0.758 0.0121   0.799 0.00556
## 10     6.14                  0.75    4.69  1.45  0.0268   0.793 0.0463
## # i 152 more rows
## # i 1 more variable: .std.resid <dbl>
```

```
ggplot(data = m1_aug, aes(x = .fitted, y = .resid)) +
  geom_point() +
  geom_hline(yintercept = 0, linetype = "dashed", color = "red") +
  xlab("Fitted values") +
  ylab("Residuals")
```



8. Is there any apparent pattern in the residuals plot? What does this indicate about the linearity of the relationship between the two variables?

=> There is random scatter and large residual value for low fitted values and the random scatter is decreasing as the fitted values are increased.

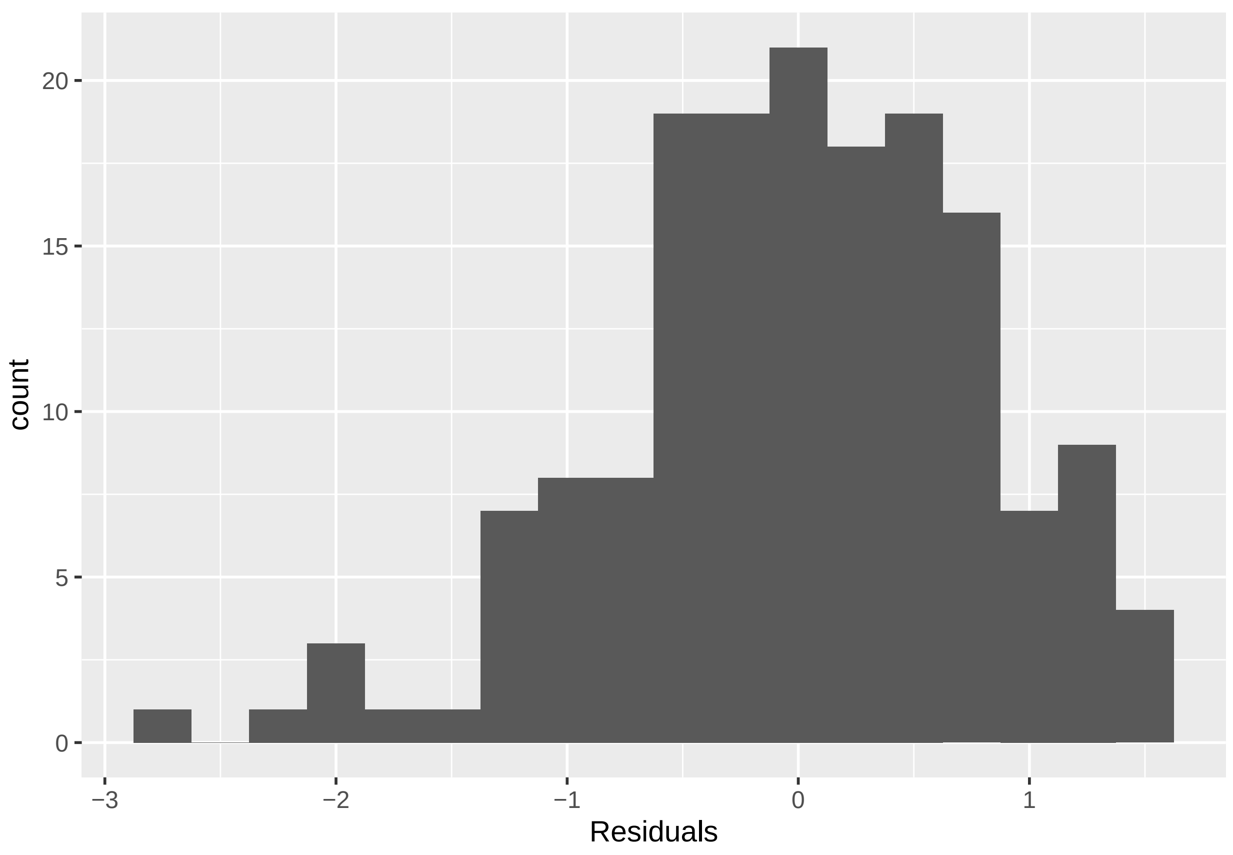
Additionally, we can say that there is the linear relationship between the two variables.

Nearly normal residuals:

To check this condition, we can look at a histogram of the residuals.

Create a new R code chunk and type the following code.

```r
ggplot(data = m1_aug, aes(x = .resid)) +
  geom_histogram(binwidth = 0.25) +
  xlab("Residuals")
```



9. Based on the histogram, does the nearly normal residuals condition appear to be violated? Why or why not?

=> As histogram is approximately bell-shaped and centered around 0, I think the nearly normal residuals condition has not been violated.

Constant variability:

10. Based on the residuals vs. fitted plot, does the constant variability condition appear to be violated? Why or why not?

=> We can see that the spread of residuals appears to be fairly consistent across all levels of fitted values, with the majority of the histograms showing a roughly symmetric and unimodal distribution of residuals. Therefore, we might conclude that the condition of constant variability is met.

1.2 Multiple Linear Regression

The data we're working with is from the OpenIntro site: https://www.openintro.org/data/csv/hfi.csv

Create a new R code chunk to read in the linked CSV file. Rather than downloading this file, uploading to RStudio, then reading it in, explore how to load this file directly from the provided URL with readr::read_csv ({readr} is part of {tidyverse} so you do not need to load/library it separately). Assign this data set into a data frame named hfi (short for "Human Freedom Index").

```r
hfi <- readr::read_csv("https://www.openintro.org/data/csv/hfi.csv")
```

```
## Rows: 1458 Columns: 123
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr   (3): ISO_code, countries, region
## dbl (120): year, pf_rol_procedural, pf_rol_civil, pf_rol_criminal, pf_rol, p...
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```r
head(hfi)
```

```
## # A tibble: 6 x 123
##    year ISO_code countries region pf_rol_procedural pf_rol_civil pf_rol_criminal
##   <dbl> <chr>    <chr>     <chr>              <dbl>        <dbl>           <dbl>
## 1  2016 ALB      Albania   Easte~              6.66         4.55            4.67
## 2  2016 DZA      Algeria   Middl~             NA           NA              NA
## 3  2016 AGO      Angola    Sub-S~             NA           NA              NA
## 4  2016 ARG      Argentina Latin~              7.10         5.79            4.34
## 5  2016 ARM      Armenia   Cauca~             NA           NA              NA
## 6  2016 AUS      Australia Ocean~              8.44         7.53            7.36
## # i 116 more variables: pf_rol <dbl>, pf_ss_homicide <dbl>,
## #   pf_ss_disappearances_disap <dbl>, pf_ss_disappearances_violent <dbl>,
## #   pf_ss_disappearances_organized <dbl>,
## #   pf_ss_disappearances_fatalities <dbl>, pf_ss_disappearances_injuries <dbl>,
## #   pf_ss_disappearances <dbl>, pf_ss_women_fgm <dbl>,
## #   pf_ss_women_missing <dbl>, pf_ss_women_inheritance_widows <dbl>,
## #   pf_ss_women_inheritance_daughters <dbl>, pf_ss_women_inheritance <dbl>, ...
```

Is this an observational study or an experiment?

=> Based on the description on the Human Freedom Index page, this is an observational study.

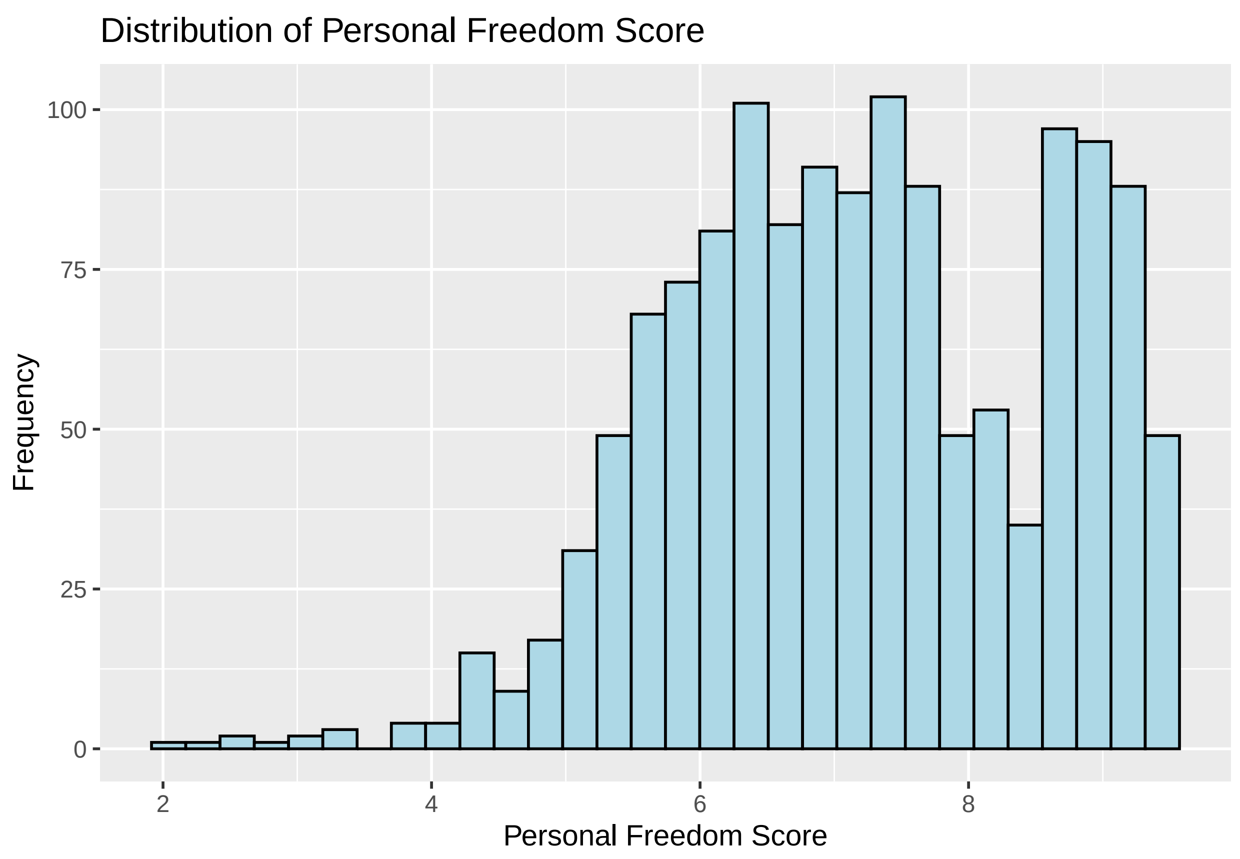
2. You will need to create appropriate univariate graphs to help in answering this:

Describe the distribution of pf_score Is the distribution skewed? Are there any other interesting/odd features (outliers, multiple peaks, etc.)? What does that tell you about countries' personal freedoms? Is this what you expected to see? Why, or why not?

=>

```r
library(ggplot2)
ggplot(hfi, aes(pf_score)) +
  geom_histogram(fill = "lightblue", color = "black") +
  labs(x = "Personal Freedom Score", y = "Frequency",
       title = "Distribution of Personal Freedom Score")
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```

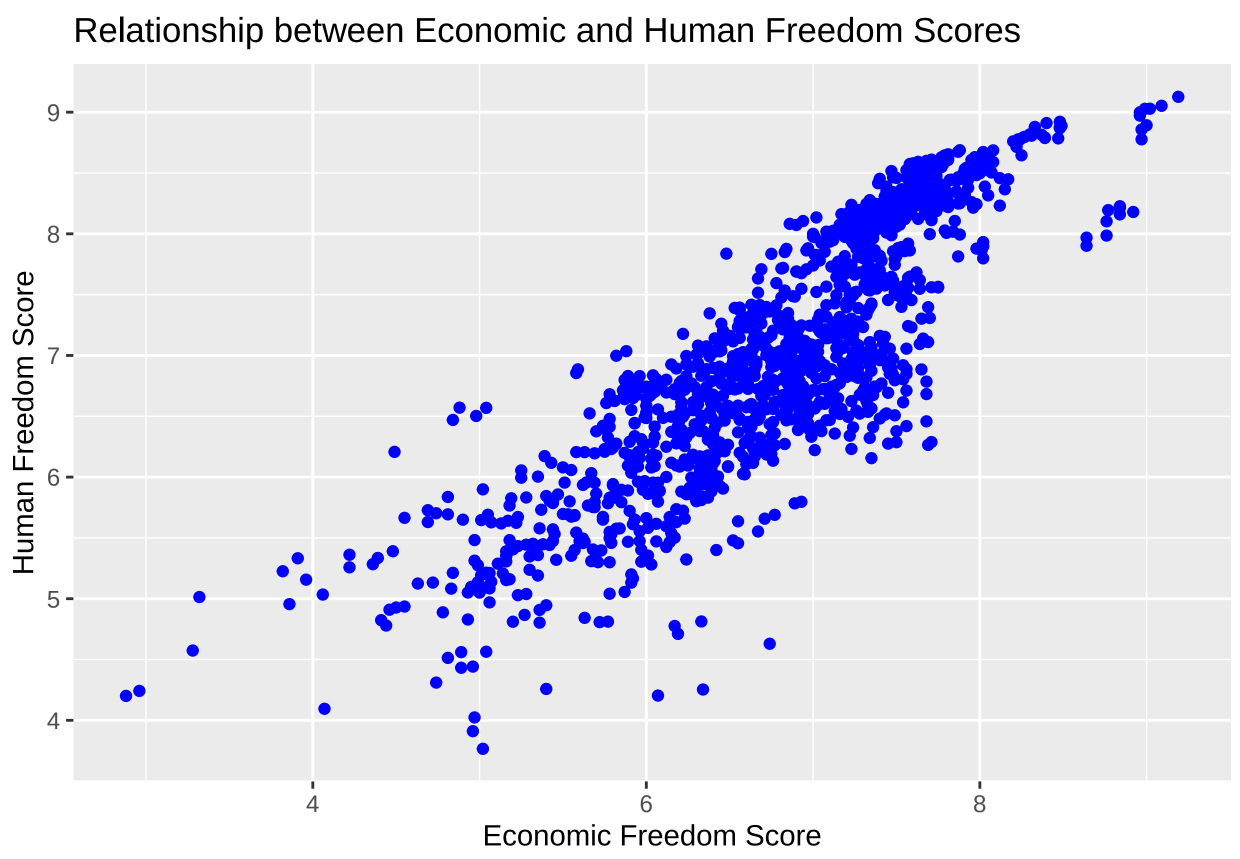


From the histogram, we can see that the distribution of pf_score is roughly symmetric, with a peak around 7.5. There are a few countries with very low personal freedom scores, which appear as outliers in the plot. This tells us that while most countries have relatively high personal freedom scores, there are a few countries where personal freedoms are severely restricted.

3. Excluding pf_score, select two other numeric variables (hint: look for or designations) and describe their relationship with each other using an appropriate visualization.

=> To describe the relationship between two other numeric variables, we can create a scatterplot. For example, let's look at the relationship between ef_score (economic freedom score) and hf_score (human freedom score):

```
ggplot(hfi, aes(ef_score, hf_score)) +
  geom_point(color = "blue") +
  labs(x = "Economic Freedom Score", y = "Human Freedom Score",
       title = "Relationship between Economic and Human Freedom Scores")
```



From the scatterplot, we can see that there is a strong positive relationship between economic freedom score and human freedom score. Countries with higher economic freedom scores tend to also have higher human freedom scores. This is what we might expect, as economic freedoms are often considered to be an important component of overall human freedom.

Pairwise relationships

Review how you described this model in Activity 2. - What were your parameter estimates (i.e., the βs)? How did you interpret these and what did they imply for this scenario? - How good of a fit was this model? What did you use to assess this?

For this activity, we will begin using the two other scores variables (i.e., hf_score and ef_score) to describe the patterns in pf_score. Take a moment to think about what this previous sentence means:

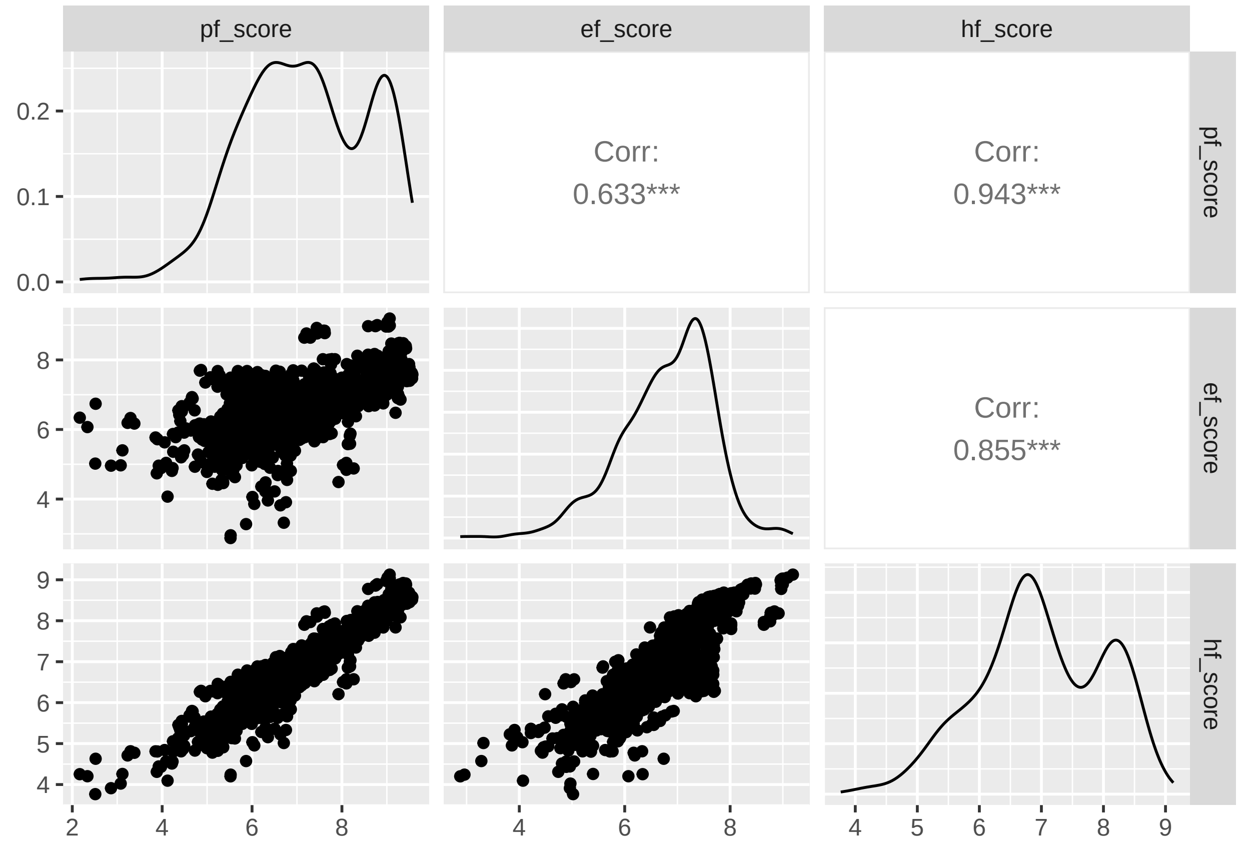
What does this mean from a statistical point of view? What does this mean from a "real world" point of view? Now, we will obtain graphical and numerical summaries to describe the pairwise relationships.

Create a new R code chunk and type the following, then run your code chunk or knit your document.

```
library(GGally)
```

```
## Registered S3 method overwritten by 'GGally':
##   method from
##   +.gg   ggplot2
```

```
library(dplyr)
hfi %>%
  select(ends_with("_score")) %>%
  ggpairs()
```



Note that a warning message (really a list of warning messages) will display in your Console and likely under your R code chunk when you knit this report. To suppress warning messages from displaying after this specific R code chunk, add the follow inside the curly brackets ({r }) at the top of your R code chunk (notice the preceding comma): , warning=FALSE.

1. For each pair of variables, how would you describe the relationship graphically? Do any of the relationships look linear? Are there any interesting/odd features (outliers, non-linear patterns, etc.)?

=>

When creating a scatterplot matrix using the ggpairs function, we can examine the pairwise relationships between all numeric variables in the hfi dataset. The graphical descriptions of the pairwise relationships are as follows:

The relationship between pf_score and hf_score appears to be linear, with a positive slope.

The relationship between pf_score and ef_score is weakly linear.

The relationship between hf_score and ef_score is linear, with a positive slope.

2. For each pair of variables, how would you describe the relationship numerically?

=>

The correlation coefficient between pf_score and hf_score is 0.943, indicating a strong positive linear relationship.

The correlation coefficient between pf_score and ef_score is 0.633, indicating a moderate positive linear relationship.

The correlation coefficient between hf_score and ef_score is 0.855, indicating a strong positive linear relationship.

Aside, if we use both hf_score and ef_score to describe the patterns in pf_score, hopefully you noticed that hf_score and ef_score are collinear (correlated). Essentially, this means that adding more than one of these variables to the model would not add much value to the model. We will talk more on this issue in Activity 4 (other considerations in regression models). For the time being, we will simply continue ahead.

The multiple linear regression model

You will now fit the following model:

Create a new R code chunk and type the following, then run your code chunk or knit your document.

```
library(broom)
m_hr_ef <- lm(pf_score ~ hf_score + ef_score, data = hfi)
tidy(m_hr_ef)
```

```
## # A tibble: 3 x 5
##   term         estimate std.error statistic p.value
##   <chr>           <dbl>     <dbl>     <dbl>   <dbl>
## 1 (Intercept)  1.46e-11  1.50e-10  9.78e- 2   0.922
## 2 hf_score     2.00e+ 0  3.63e-11  5.52e+10   0
## 3 ef_score    -1.00e+ 0  4.21e-11 -2.38e+10   0
```

6. Using your output, write the complete estimated equation for this model. Remember in Activity 2 that this looked like:

y_hat = 1.464213e-11 + (2.000000e+00)*hf_score + (-1.000000e+00)*ef_score + Error

7. For each of the estimated parameters (the y-intercept, and slope associated with each explanatory variable - three total), interpret these values in the context of this problem. That is, what do they mean for a "non-data" person?

=>

The intercept (or y-intercept) is estimated to be 1.46e-11. This represents the expected value of the pf_score when both hf_score and ef_score are equal to 0. In other words, if a player had no heart function score and no exercise function score, their predicted physical function score would be very close to 0.

The estimated coefficient for hf_score is 2.0. This means that for every unit increase in the hf_score, the predicted pf_score will increase by 2 units, holding ef_score constant. Therefore, a player with a higher heart function score would be expected to have a higher physical function score, all else being equal.

The estimated coefficient for ef_score is -1.0. This means that for every unit increase in the ef_score, the predicted pf_score will decrease by 1 unit, holding hf_score constant. Therefore, a player with a higher exercise function score would be expected to have a lower physical function score, all else being equal.

Challenge: 3-D plots

In ISL, the authors provided a 3-D scatterplot with a plane that represents the estimated model. Do some internet sluething to minimally produce a 3-D scatterplot (you do not need to include the plane). Ideally, this would be something that plays nicely with (looks simlilar to) {ggplot2}.

Create a new R code chunk and add your code to create this plot. =>

```
#library(plotly)
#plot_ly(data = hfi, x = ~hf_score, y = ~ef_score, z = ~pf_score, type = "scatter3d", mode = "markers")
```

The 3-D scatterplot allows us to visualize the relationship between three variables, which is not possible with the GGally::ggpairs output. It also allows us to rotate and interact with the plot, which can help us gain a better understanding of the relationship between the variables. However, it can be difficult to see the relationship between the variables with a 3-D scatterplot, especially if the data points are overlapping or if there are a large number of data points. The GGally::ggpairs output allows us to see the relationship between all pairs of variables in one plot, making it easier to see patterns and relationships between the variables. However, it can be difficult to interpret when there are a large number of variables or if there are many outliers in the data

```
#library(plotly)
#plotly::ggplotly(data = hfi, x = ~hf_score, y = ~ef_score, z = ~pf_score, type = "scatter3d", mode = "
```

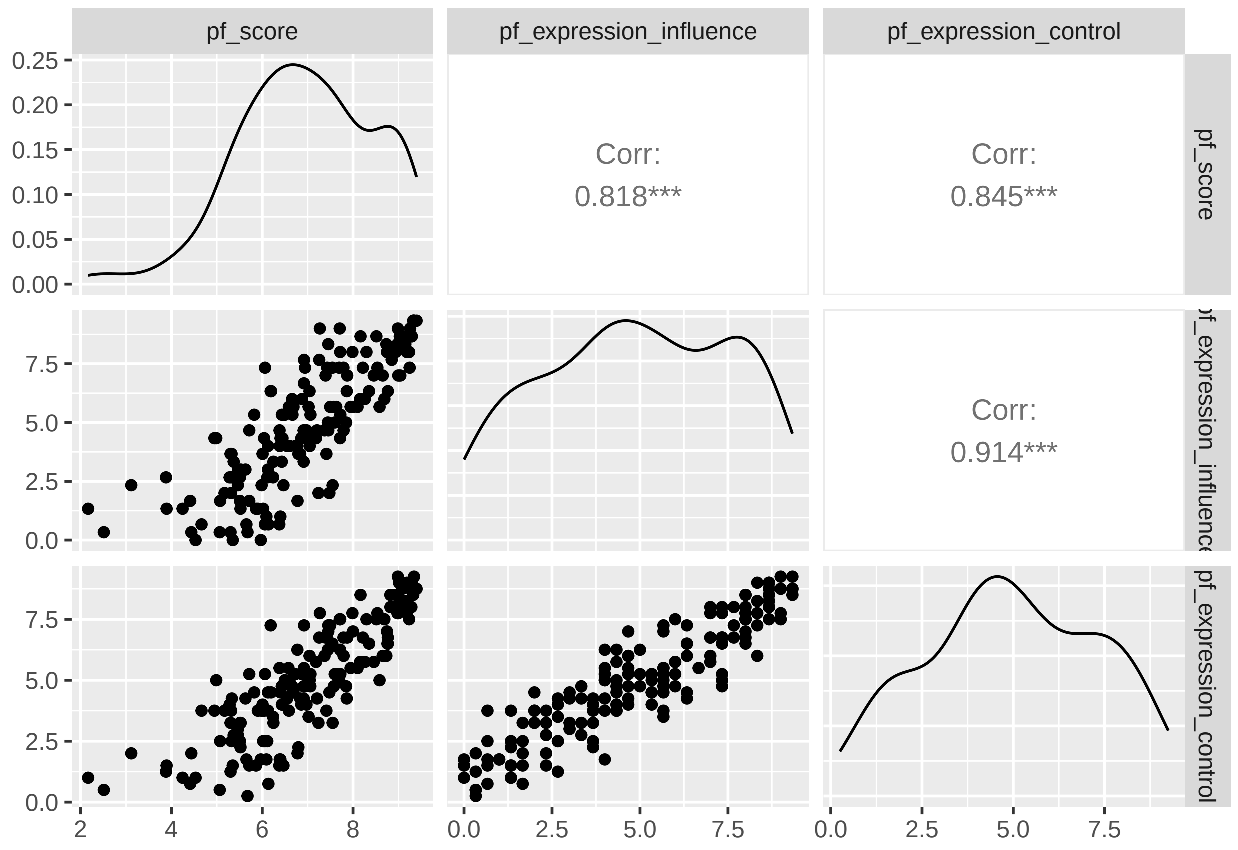
Is One Predictor Useful?

Filtering the data of the year 2016 only

```
hfi_2016 <- filter(hfi, year == 2016)
```

Review any visual patterns and then fit the MLR model

```
# review any visual patterns
hfi_2016 %>%
  select(pf_score, pf_expression_influence, pf_expression_control) %>%
  ggpairs()
```



```r
#fit the mlr model
m_pf <- lm(pf_score ~ pf_expression_influence + pf_expression_control, data = hfi_2016)
tidy(m_pf)
```

```
## # A tibble: 3 x 5
##   term                    estimate std.error statistic  p.value
##   <chr>                      <dbl>     <dbl>     <dbl>    <dbl>
## 1 (Intercept)                4.33     0.147      29.4  3.02e-66
## 2 pf_expression_influence    0.155    0.0574      2.70 7.68e- 3
## 3 pf_expression_control      0.380    0.0655      5.81 3.29e- 8
```

The correlation values means that the variables (pf_expression_influence and pf_expression_control) are strongly correlated to pf_score.

###Is there a relationship between the response and predictors? We visually assessed if there appeared to be a linear relationship between the response and each predictor variable (along with providing a measurement of the strength and direction of that relationship if it is linear). Now we would like to assess if this relationship is "note worthy". That is, is there overwhelming evidence that at least one of the slopes associated with this MLR model exists (or is not zero). If a slope is zero (e.g., if beta1 = 0), that would indicate no relationship.

We can do this by testing the overall model. The statistical hypotheses that we are testing are: H0 : beta1 = beta2 = 0 Ha : At least one beta_j is not zero, for j =1, 2

From your reading, this hypothesis test is performed using the F-statistic which has two degrees of freedom associated with it.

Create a new R code chunk and type the following code.

```r
summary(m_pf)
```

```
##
## Call:
## lm(formula = pf_score ~ pf_expression_influence + pf_expression_control,
##     data = hfi_2016)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2.74651 -0.46378  0.05426  0.55326  1.62682
##
## Coefficients:
##                         Estimate Std. Error t value Pr(>|t|)
## (Intercept)              4.32622    0.14697  29.436  < 2e-16 ***
## pf_expression_influence  0.15487    0.05736   2.700  0.00768 **
## pf_expression_control    0.38036    0.06545   5.811 3.29e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.784 on 159 degrees of freedom
## Multiple R-squared:  0.7267, Adjusted R-squared:  0.7232
## F-statistic: 211.4 on 2 and 159 DF,  p-value: < 2.2e-16
```

The p-value is less than alpha (0.05), so we should reject the null hypothesis. Therefore, at least one of the variables in this model is useful, so either pf_expression_influence or pf_expression_control.

The standard error of the two variables differ by .008.

Deciding on Important Variables

The 'pf_expression_control' has a lower p-value so I think it is more important than 'pf_expression_influence'.

Model Fit

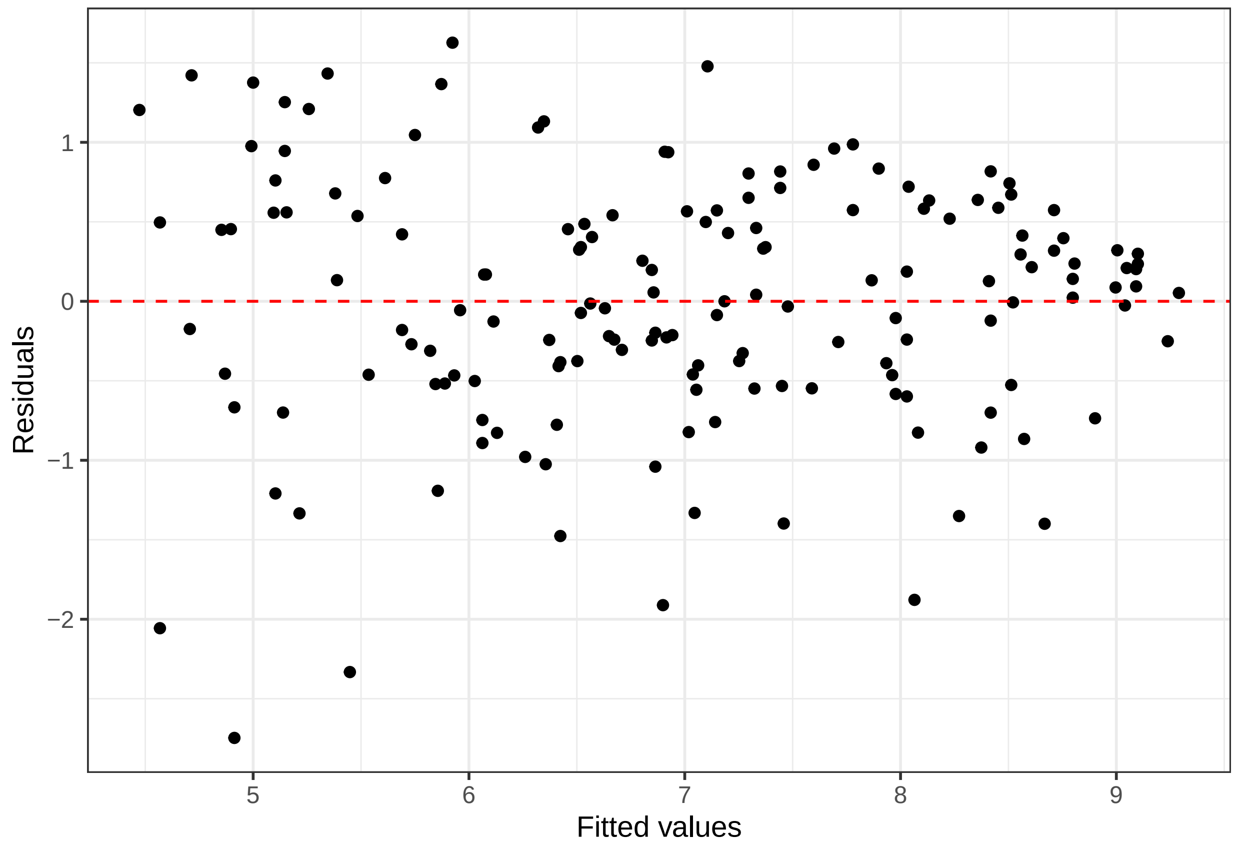
```r
glance(m_pf)
```

```
## # A tibble: 1 x 12
##   r.squared adj.r.squared sigma statistic  p.value    df logLik   AIC   BIC
##       <dbl>         <dbl> <dbl>     <dbl>    <dbl> <dbl>  <dbl> <dbl> <dbl>
## 1     0.727         0.723 0.784      211. 1.65e-45     2  -189.  386.  398.
## # i 3 more variables: deviance <dbl>, df.residual <int>, nobs <int>
```

The R-squared value is 0.7266672 which means that this model improves the prediction accuracy by 72.67% over using the average of `pf_score`. The r-squared value of the single linear regression model is 0.7141342 so this multiple linear regression model has an improved accuracy.

###Linearity: You already checked if the relationship between pf_score and pf_expression_influence and between pf_score and pf_expression_control is linear using. We should also verify this condition with a plot of the residuals vs. fitted (predicted) values once we have fit a MLR model.

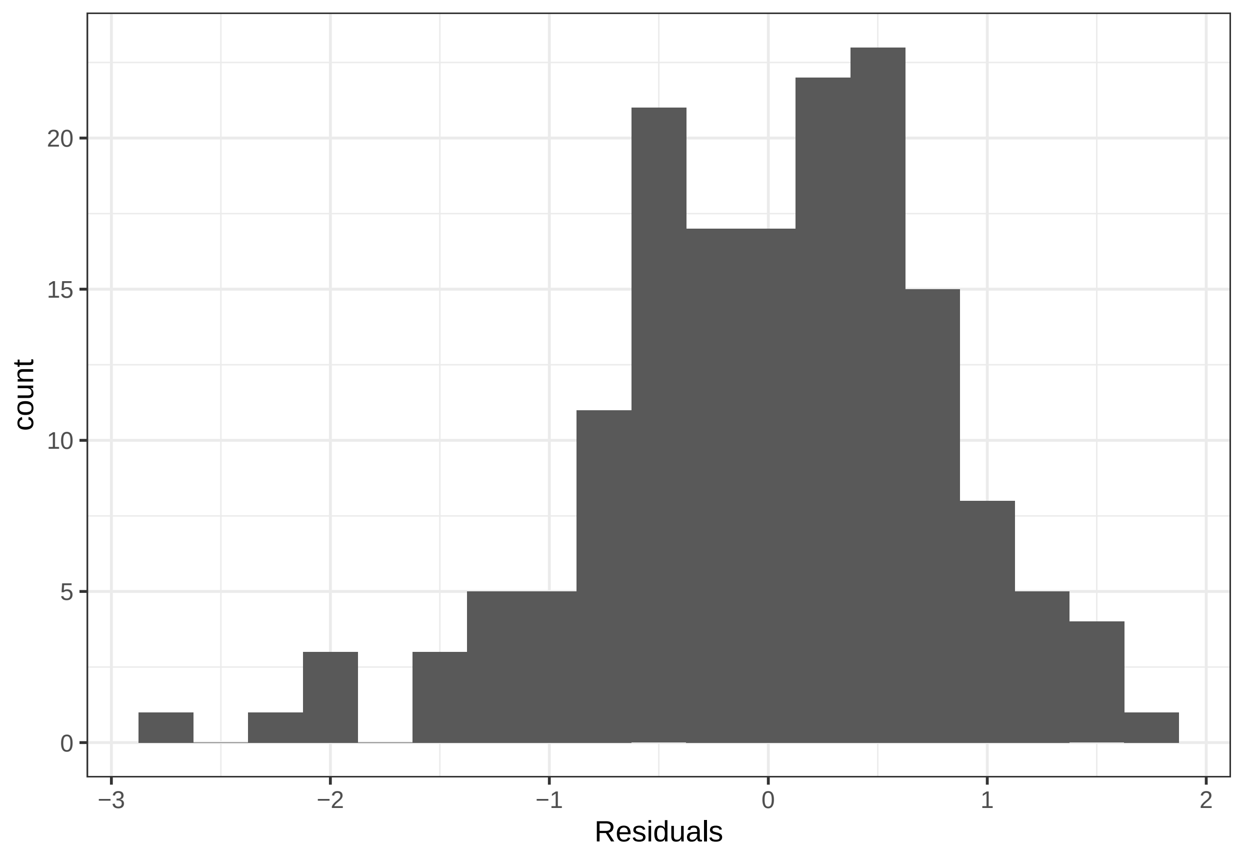
```r
# obtain fitted values and residuals
m_pf_aug <- augment(m_pf)
# plot fitted values and residuals
ggplot(data = m_pf_aug, aes(x = .fitted, y = .resid)) +
  geom_point() +
  geom_hline(yintercept = 0, linetype = "dashed", color = "red") +
  xlab("Fitted values") +
  ylab("Residuals") +
  theme_bw()
```



There is random scatter and large residual value for low fitted values and the random scatter is decreasing as the fitted values are increased.

###Nearly normal residuals: To check this condition, we can look at a histogram of the residuals.

```r
ggplot(data = m_pf_aug, aes(x = .resid)) +
  geom_histogram(binwidth = 0.25) +
  xlab("Residuals") +
  theme_bw()
```



As histogram is approximately bell-shaped and centered around 0, I think the nearly normal residuals condition has not been violated.

Prediction

```
hfi %>%
  filter(countries == "United States" & year == 2016) %>%
  select(pf_score, pf_expression_influence, pf_expression_control)
```

```
## # A tibble: 1 x 3
##   pf_score pf_expression_influence pf_expression_control
##      <dbl>                   <dbl>                 <dbl>
## 1     8.75                       8                     7
```

```
hfi %>%
  filter(countries == "United States" & year == 2016) %>%
  predict(m_pf, .)
```

```
##        1
## 8.227657
```

Objective 2: Determine and apply the appropriate generalized linear model for a specific data context

Customer Churn Prediction

Introduction

In this project, we will be analyzing a telecom churn dataset. The objective of the analysis is to develop a model to predict whether a customer will churn or not. The dataset consists of 7043 observations and 23 variables. The dataset contains a mix of categorical and continuous variables.

Loading the required package and reading the dataset

```
library(readxl)
telecom_data <- read_excel("data/telecom-churn-rate-dataset.xlsx")
#View(telecom_churn_rate_dataset)
```

Check the dimension of the dataset. Get the names of variables in the dataset

```
dim(telecom_data)
```

```
## [1] 7043   23
```

```
names(telecom_data)
```

```
##  [1] "customerID"       "gender"           "SeniorCitizen"    "Partner"
##  [5] "Dependents"       "tenure"           "PhoneService"     "MultipleLines"
##  [9] "InternetService"  "OnlineSecurity"   "OnlineBackup"     "DeviceProtection"
## [13] "TechSupport"      "StreamingTV"      "StreamingMovies"  "Contract"
## [17] "PaperlessBilling" "PaymentMethod"    "MonthlyCharges"   "TotalCharges"
## [21] "numAdminTickets"  "numTechTickets"   "Churn"
```

Get the structure of the dataset.

```
str(telecom_data)
```

```
## tibble [7,043 x 23] (S3: tbl_df/tbl/data.frame)
##  $ customerID      : chr [1:7043] "7590-VHVEG" "5575-GNVDE" "3668-QPYBK" "7795-CFOCW" ...
##  $ gender          : chr [1:7043] "Female" "Male" "Male" "Male" ...
##  $ SeniorCitizen   : num [1:7043] 0 0 0 0 0 0 0 0 0 0 ...
##  $ Partner         : chr [1:7043] "Yes" "No" "No" "No" ...
##  $ Dependents      : chr [1:7043] "No" "No" "No" "No" ...
##  $ tenure          : num [1:7043] 1 34 2 45 2 8 22 10 28 62 ...
##  $ PhoneService    : chr [1:7043] "No" "Yes" "Yes" "No" ...
##  $ MultipleLines   : chr [1:7043] "No phone service" "No" "No" "No phone service" ...
##  $ InternetService : chr [1:7043] "DSL" "DSL" "DSL" "DSL" ...
##  $ OnlineSecurity  : chr [1:7043] "No" "Yes" "Yes" "Yes" ...
##  $ OnlineBackup    : chr [1:7043] "Yes" "No" "Yes" "No" ...
##  $ DeviceProtection: chr [1:7043] "No" "Yes" "No" "Yes" ...
##  $ TechSupport     : chr [1:7043] "No" "No" "No" "Yes" ...
##  $ StreamingTV     : chr [1:7043] "No" "No" "No" "No" ...
##  $ StreamingMovies : chr [1:7043] "No" "No" "No" "No" ...
##  $ Contract        : chr [1:7043] "Month-to-month" "One year" "Month-to-month" "One year" ...
##  $ PaperlessBilling: chr [1:7043] "Yes" "No" "Yes" "No" ...
##  $ PaymentMethod   : chr [1:7043] "Electronic check" "Mailed check" "Mailed check" "Bank transfer (a
##  $ MonthlyCharges  : num [1:7043] 29.9 57 53.9 42.3 70.7 ...
##  $ TotalCharges    : num [1:7043] 29.9 1889.5 108.2 1840.8 151.7 ...
##  $ numAdminTickets : num [1:7043] 0 0 0 0 0 0 0 0 0 0 ...
##  $ numTechTickets  : num [1:7043] 0 0 0 3 0 0 0 0 2 0 ...
##  $ Churn           : chr [1:7043] "No" "No" "Yes" "No" ...
```

There are 17 categorical (character) variables and 6 numeric variables in the dataset.

Find the number of unique values in each variable:

```
library(dplyr)
sapply(telecom_data, n_distinct)
```

```
##       customerID           gender    SeniorCitizen          Partner
##             7043                2                2                2
##       Dependents           tenure     PhoneService    MultipleLines
##                2               73                2                3
##  InternetService   OnlineSecurity     OnlineBackup DeviceProtection
##                3                3                3                3
##      TechSupport      StreamingTV  StreamingMovies         Contract
##                3                3                3                3
## PaperlessBilling    PaymentMethod   MonthlyCharges     TotalCharges
##                2                4             1585             6531
##  numAdminTickets   numTechTickets            Churn
##                6               10                2
```

Check for missing values in the dataset:

```
colSums(is.na(telecom_data))
```

```
##       customerID           gender    SeniorCitizen          Partner
##                0                0                0                0
##       Dependents           tenure     PhoneService    MultipleLines
##                0                0                0                0
##  InternetService   OnlineSecurity     OnlineBackup DeviceProtection
##                0                0                0                0
##      TechSupport      StreamingTV  StreamingMovies         Contract
##                0                0                0                0
## PaperlessBilling    PaymentMethod   MonthlyCharges     TotalCharges
##                0                0                0               11
##  numAdminTickets   numTechTickets            Churn
##                0                0                0
```

The 'TotalCharges' variable shas 11 missing values

Remove the observations having missing values.

```r
telecom_data <- na.omit(telecom_data)
```

Checking if there are any mising values again.

```r
colSums(is.na(telecom_data))
```

```
##       customerID           gender    SeniorCitizen          Partner
##                0                0                0                0
##       Dependents           tenure     PhoneService    MultipleLines
##                0                0                0                0
##  InternetService   OnlineSecurity     OnlineBackup DeviceProtection
##                0                0                0                0
##      TechSupport      StreamingTV  StreamingMovies         Contract
##                0                0                0                0
## PaperlessBilling    PaymentMethod   MonthlyCharges     TotalCharges
##                0                0                0                0
##  numAdminTickets   numTechTickets            Churn
##                0                0                0
```

From this, we found that there are not any missing values.

For the logistic regression, it is appropriate to convert categorical variables to factor. Therefore, converting the categorical variables to factor.

```r
telecom_data[, sapply(telecom_data, is.character)] <- lapply(telecom_data[, sapply(telecom_data, is.cha
```

Checking whether the categorical variables are converted to factor or not:

```r
str(telecom_data)
```

```
## tibble [7,032 x 23] (S3: tbl_df/tbl/data.frame)
##  $ customerID      : Factor w/ 7032 levels "0002-ORFBO","0003-MKNFE",..: 5366 3954 2559 5525 6501 65
##  $ gender          : Factor w/ 2 levels "Female","Male": 1 2 2 2 1 1 2 1 1 2 ...
##  $ SeniorCitizen   : num [1:7032] 0 0 0 0 0 0 0 0 0 0 ...
##  $ Partner         : Factor w/ 2 levels "No","Yes": 2 1 1 1 1 1 1 1 2 1 ...
##  $ Dependents      : Factor w/ 2 levels "No","Yes": 1 1 1 1 1 1 2 1 1 2 ...
##  $ tenure          : num [1:7032] 1 34 2 45 2 8 22 10 28 62 ...
##  $ PhoneService    : Factor w/ 2 levels "No","Yes": 1 2 2 1 2 2 2 1 2 2 ...
##  $ MultipleLines   : Factor w/ 3 levels "No","No phone service",..: 2 1 1 2 1 3 3 2 3 1 ...
##  $ InternetService : Factor w/ 3 levels "DSL","Fiber optic",..: 1 1 1 1 2 2 2 1 2 1 ...
##  $ OnlineSecurity  : Factor w/ 3 levels "No","No internet service",..: 1 3 3 3 1 1 1 3 1 3 ...
##  $ OnlineBackup    : Factor w/ 3 levels "No","No internet service",..: 3 1 3 1 1 1 3 1 1 3 ...
##  $ DeviceProtection: Factor w/ 3 levels "No","No internet service",..: 1 3 1 3 1 3 1 1 3 1 ...
##  $ TechSupport     : Factor w/ 3 levels "No","No internet service",..: 1 1 1 3 1 1 1 1 3 1 ...
##  $ StreamingTV     : Factor w/ 3 levels "No","No internet service",..: 1 1 1 1 1 3 3 1 3 1 ...
##  $ StreamingMovies : Factor w/ 3 levels "No","No internet service",..: 1 1 1 1 1 3 1 1 3 1 ...
##  $ Contract        : Factor w/ 3 levels "Month-to-month",..: 1 2 1 2 1 1 1 1 1 2 ...
##  $ PaperlessBilling: Factor w/ 2 levels "No","Yes": 2 1 2 1 2 2 2 1 2 1 ...
##  $ PaymentMethod   : Factor w/ 4 levels "Bank transfer (automatic)",..: 3 4 4 1 3 3 2 4 3 1 ...
##  $ MonthlyCharges  : num [1:7032] 29.9 57 53.9 42.3 70.7 ...
##  $ TotalCharges    : num [1:7032] 29.9 1889.5 108.2 1840.8 151.7 ...
##  $ numAdminTickets : num [1:7032] 0 0 0 0 0 0 0 0 0 0 ...
##  $ numTechTickets  : num [1:7032] 0 0 0 3 0 0 0 0 2 0 ...
##  $ Churn           : Factor w/ 2 levels "No","Yes": 1 1 2 1 2 2 1 1 2 1 ...
##  - attr(*, "na.action")= 'omit' Named int [1:11] 489 754 937 1083 1341 3332 3827 4381 5219 6671 ...
##   ..- attr(*, "names")= chr [1:11] "489" "754" "937" "1083" ...
```

Grouping customers by gender and finding the number of customers for each gender:

```
library(magrittr)
library(dplyr)
customer_status <- telecom_data %>% group_by(gender) %>% summarise(num_customers = n())
customer_status
```

```
## # A tibble: 2 x 2
##   gender num_customers
##   <fct>          <int>
## 1 Female          3483
## 2 Male            3549
```

From the table, we found that there are 3483 female customers and 3549 male customers.

As ‘Customer ID’ variable is not related to the regression, so using variables except customerID variable. Removing customerID variable from the dataset:

```
telecom_data <- select(telecom_data, -customerID)
```

Performing logistic regression with all the variables:

```
logistic_reg1 <- glm(Churn ~ ., family = binomial, data = telecom_data)
summary(logistic_reg1)
```

```
##
## Call:
## glm(formula = Churn ~ ., family = binomial, data = telecom_data)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.7730  -0.4598  -0.0797   0.2386   3.8509
##
## Coefficients: (7 not defined because of singularities)
##                                         Estimate Std. Error z value Pr(>|z|)
## (Intercept)                            1.4285935  1.0245216   1.394 0.163197
## genderMale                            -0.0957337  0.0764895  -1.252 0.210718
## SeniorCitizen                          0.2698725  0.1045992   2.580 0.009878 **
## PartnerYes                            -0.0549046  0.0943263  -0.582 0.560519
## DependentsYes                         -0.0930284  0.1081090  -0.861 0.389511
## tenure                                -0.0683092  0.0072093  -9.475  < 2e-16 ***
## PhoneServiceYes                        0.2408302  0.8188660   0.294 0.768680
## MultipleLinesNo phone service                 NA         NA      NA       NA
## MultipleLinesYes                       0.5188060  0.2209888   2.348 0.018892 *
## InternetServiceFiber optic             1.8314326  1.0126636   1.809 0.070524 .
## InternetServiceNo                     -1.9160048  1.0191497  -1.880 0.060108 .
## OnlineSecurityNo internet service             NA         NA      NA       NA
## OnlineSecurityYes                    -0.3581201  0.2254677  -1.588 0.112209
## OnlineBackupNo internet service              NA         NA      NA       NA
## OnlineBackupYes                      -0.2108974  0.2221889  -0.949 0.342529
## DeviceProtectionNo internet service          NA         NA      NA       NA
## DeviceProtectionYes                   0.0071828  0.2227764   0.032 0.974279
## TechSupportNo internet service               NA         NA      NA       NA
## TechSupportYes                       -0.0492909  0.2281475  -0.216 0.828950
## StreamingTVNo internet service               NA         NA      NA       NA
## StreamingTVYes                        0.3565925  0.4135539   0.862 0.388543
## StreamingMoviesNo internet service           NA         NA      NA       NA
## StreamingMoviesYes                    0.3587087  0.4142150   0.866 0.386492
## ContractOne year                     -0.8510954  0.1558515  -5.461 4.74e-08 ***
## ContractTwo year                     -2.4370384  0.3004879  -8.110 5.05e-16 ***
## PaperlessBillingYes                   0.3191673  0.0854006   3.737 0.000186 ***
## PaymentMethodCredit card (automatic) -0.2092096  0.1431671  -1.461 0.143934
## PaymentMethodElectronic check         0.1326731  0.1182345   1.122 0.261812
## PaymentMethodMailed check            -0.2372223  0.1337243  -1.774 0.076069 .
## MonthlyCharges                       -0.0355461  0.0402637  -0.883 0.377327
## TotalCharges                         -0.0002083  0.0000880  -2.367 0.017947 *
## numAdminTickets                      -0.0514773  0.0300145  -1.715 0.086330 .
## numTechTickets                        1.4598345  0.0532417  27.419  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 8143.4  on 7031  degrees of freedom
## Residual deviance: 4257.2  on 7006  degrees of freedom
## AIC: 4309.2
##
## Number of Fisher Scoring iterations: 7
```

There are several values with multicollinearity and insignificantly large p-values.

Considering only the variables having significant p-value. Performing logistic regression with significant variables only:

```
logistic_reg <- glm(Churn ~ SeniorCitizen + tenure + MultipleLines + Contract + PaperlessBilling + Tota
summary(logistic_reg)
```

```
##
## Call:
## glm(formula = Churn ~ SeniorCitizen + tenure + MultipleLines +
##     Contract + PaperlessBilling + TotalCharges + numTechTickets,
##     family = binomial, data = telecom_data)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.9517  -0.5189  -0.0862   0.2388   4.0840
##
## Coefficients:
##                               Estimate Std. Error z value Pr(>|z|)
## (Intercept)                 -2.756e-01  7.308e-02  -3.771 0.000162 ***
## SeniorCitizen                5.774e-01  9.761e-02   5.916 3.30e-09 ***
## tenure                        -9.959e-02  6.643e-03 -14.993  < 2e-16 ***
## MultipleLinesNo phone service  3.416e-01  1.300e-01   2.628 0.008578 **
## MultipleLinesYes               6.188e-01  8.869e-02   6.977 3.01e-12 ***
## ContractOne year              -1.308e+00  1.484e-01  -8.811  < 2e-16 ***
## ContractTwo year              -2.992e+00  2.877e-01 -10.399  < 2e-16 ***
## PaperlessBillingYes            6.563e-01  7.910e-02   8.298  < 2e-16 ***
## TotalCharges                   1.961e-04  7.336e-05   2.674 0.007505 **
## numTechTickets                 1.412e+00  5.122e-02  27.576  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 8143.4  on 7031  degrees of freedom
## Residual deviance: 4579.8  on 7022  degrees of freedom
## AIC: 4599.8
##
## Number of Fisher Scoring iterations: 7
```

Data Partition and Modelling

In this chunk of code, the necessary packages are loaded, and the telecom data is partitioned into a training and testing set. The leave column is created as a factor with two levels, "Churn" and "Not Vhurn" and the "Churn" column is removed. The glm function from caret is used to fit a logistic regression model to the training set. The model's performance is then evaluated using confusion matrices for both the training and testing sets.

```
library(caret)
```

```
## Loading required package: lattice
```

```
data_partition <- telecom_data
#telecom_data$Churn = as.factor(telecom_data$Churn)
data_partition$leave = ifelse(data_partition$Churn == "Yes","Churn","Not Churn")
data_partition$leave = as.factor(data_partition$leave)
data_partition = data_partition %>% dplyr::select(-Churn) #removing the column with numbers, otherwise

set.seed(1)
test.indices = createDataPartition(data_partition$leave, p = 0.2, list = FALSE) #classic 80/20 train-te
test_partition = data_partition[test.indices,]
train_partition = data_partition[-test.indices,]
```

The code fits a generalized linear model (GLM) using the train function from the caret package to predict customer churn based on seven predictor variables. The trained model is then used to generate churn predictions for both the training and test partitions.

```
model_train = train(leave ~ SeniorCitizen + tenure + MultipleLines + Contract + PaperlessBilling + Tota

predTrain = predict(model_train, train_partition)
predTest = predict(model_train, test_partition)
```

For Training data, the Confusion Matrix:

```
confusionMatrix(predTrain, train_partition$leave, positive = "Churn")
```

```
## Confusion Matrix and Statistics
##
##            Reference
## Prediction  Churn Not Churn
##   Churn       997       367
##   Not Churn   498      3763
##
##                Accuracy : 0.8462
##                  95% CI : (0.8365, 0.8556)
##     No Information Rate : 0.7342
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.5946
##
##  Mcnemar's Test P-Value : 9.864e-06
##
##             Sensitivity : 0.6669
##             Specificity : 0.9111
##          Pos Pred Value : 0.7309
##          Neg Pred Value : 0.8831
##              Prevalence : 0.2658
##          Detection Rate : 0.1772
##    Detection Prevalence : 0.2425
##       Balanced Accuracy : 0.7890
##
##        'Positive' Class : Churn
##
```

We got 84.62% accuracy here.

For Testing data, the Confusion Matrix:

```
confusionMatrix(predTest, test_partition$leave, positive = "Churn")
```

```
## Confusion Matrix and Statistics
##
##            Reference
## Prediction  Churn Not Churn
##   Churn       253        92
##   Not Churn   121       941
##
##                Accuracy : 0.8486
##                  95% CI : (0.8288, 0.867)
##     No Information Rate : 0.7342
##     P-Value [Acc > NIR] : < 2e-16
##
##                   Kappa : 0.6023
##
##  Mcnemar's Test P-Value : 0.05504
##
##             Sensitivity : 0.6765
##            Specificity : 0.9109
##         Pos Pred Value : 0.7333
##         Neg Pred Value : 0.8861
##             Prevalence : 0.2658
##         Detection Rate : 0.1798
##   Detection Prevalence : 0.2452
##      Balanced Accuracy : 0.7937
##
##       'Positive' Class : Churn
##
```

For the testing data, the accuracy is little more than for the training data (84.86%) which is a good prediction.

Cross-validation

In this chunk of code, a 15-fold cross-validation technique is applied to the logistic regression model previously built. The performance of the model is then printed. The model's performance is also evaluated using a confusion matrix on the testing set.

```
train_control <- trainControl(method="cv", number=15) #15-fold cross validation
model_cv <- caret::train(leave ~ SeniorCitizen + tenure + MultipleLines + Contract + PaperlessBilling +
print(model_cv)
```

```
## Generalized Linear Model
##
## 5625 samples
##    7 predictor
##    2 classes: 'Churn', 'Not Churn'
##
## No pre-processing
## Resampling: Cross-Validated (15 fold)
## Summary of sample sizes: 5251, 5250, 5250, 5250, 5250, 5250, ...
## Resampling results:
##
##   Accuracy   Kappa
##   0.8444451  0.5910879
```

We got the accuracy of 84.45% which is less than that of data partition we did before.

Now, finding the accuracy for the test data:

```
predTest.cv <- predict(model_cv, test_partition)
cmTest.cv = confusionMatrix(predTest.cv, test_partition$leave)
cmTest.cv
```

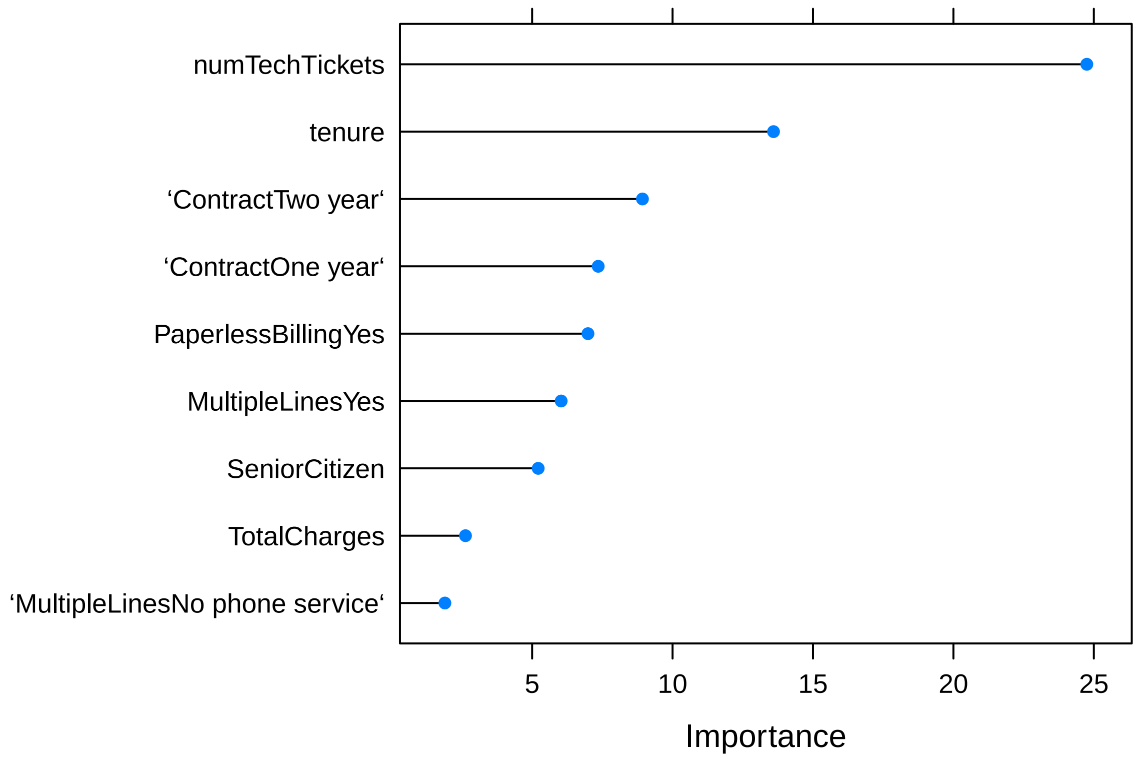
```
## Confusion Matrix and Statistics
##
##            Reference
## Prediction  Churn Not Churn
##   Churn       253        92
##   Not Churn   121       941
##
```

```
##                Accuracy : 0.8486
##                  95% CI : (0.8288, 0.867)
##     No Information Rate : 0.7342
##     P-Value [Acc > NIR] : < 2e-16
##
##                   Kappa : 0.6023
##
##  Mcnemar's Test P-Value : 0.05504
##
##             Sensitivity : 0.6765
##             Specificity : 0.9109
##          Pos Pred Value : 0.7333
##          Neg Pred Value : 0.8861
##              Prevalence : 0.2658
##          Detection Rate : 0.1798
##    Detection Prevalence : 0.2452
##       Balanced Accuracy : 0.7937
##
##        'Positive' Class : Churn
##
```

The accuracy is 84.86% which is equal to that of normal data partition.

Now finding the important variables for the model:

```
importance <- varImp(model_train, scale=FALSE)
plot(importance)
```



Decision Tree

In this chunk of code, the party package is loaded, and a decision tree is built using the ctree function. The model's performance is evaluated using confusion matrices for both the training and testing sets.

```
library(party)
```

```
## Loading required package: grid

## Loading required package: mvtnorm

## Loading required package: modeltools

## Loading required package: stats4

## Loading required package: strucchange

## Loading required package: zoo

##
## Attaching package: 'zoo'

## The following objects are masked from 'package:base':
##
##     as.Date, as.Date.numeric

## Loading required package: sandwich

##
## Attaching package: 'party'

## The following object is masked from 'package:dplyr':
##
##     where
```

```
tree <- ctree(leave~., data = train_partition)
```

```
treePredTrain <- predict(tree, train_partition, type = "response")

confusionMatrix(treePredTrain,train_partition$leave)
```

```
## Confusion Matrix and Statistics
##
##            Reference
## Prediction  Churn Not Churn
##   Churn      1078       337
##   Not Churn   417      3793
##
##                Accuracy : 0.866
##                  95% CI : (0.8568, 0.8748)
##     No Information Rate : 0.7342
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.6506
##
##  Mcnemar's Test P-Value : 0.004015
##
##             Sensitivity : 0.7211
##             Specificity : 0.9184
##          Pos Pred Value : 0.7618
##          Neg Pred Value : 0.9010
##              Prevalence : 0.2658
##          Detection Rate : 0.1916
##    Detection Prevalence : 0.2516
##       Balanced Accuracy : 0.8197
##
##        'Positive' Class : Churn
##
```

We got the 86.6% accuracy using decision tree which is greater than others above.

Finding accuracy on the test data:

```
treePredTest <- predict(tree, test_partition, type = "response")

confusionMatrix(treePredTest,test_partition$leave)
```

```
## Confusion Matrix and Statistics
##
##            Reference
## Prediction  Churn Not Churn
##   Churn       259       100
##   Not Churn   115       933
##
##                Accuracy : 0.8472
##                  95% CI : (0.8273, 0.8656)
##     No Information Rate : 0.7342
##     P-Value [Acc > NIR] : <2e-16
##
##                   Kappa : 0.6034
##
##  Mcnemar's Test P-Value : 0.3397
##
##             Sensitivity : 0.6925
##             Specificity : 0.9032
##          Pos Pred Value : 0.7214
##          Neg Pred Value : 0.8903
##              Prevalence : 0.2658
##          Detection Rate : 0.1841
##    Detection Prevalence : 0.2552
##       Balanced Accuracy : 0.7979
##
##        'Positive' Class : Churn
##
```

The accuracy we got is the largest till now: 84.72%.

Cross Validation: Decision Tree

In this chunk of code, a 10-fold cross-validation technique is applied to the decision tree model previously built. The performance of the model is then printed. The model's performance is also evaluated using a confusion matrix on the testing set.

Overall, the code performs data partitioning, logistic regression, cross-validation, and decision tree modeling on the telecom data set and evaluates the model's performance on the training and testing sets.

```
train_control_tree <- trainControl(method="cv", number=15)
model_tree <- caret::train(leave~., data=train_partition, trControl=train_control_tree, method="ctree")

print(model_tree)
```

```
## Conditional Inference Tree
##
## 5625 samples
##   21 predictor
##    2 classes: 'Churn', 'Not Churn'
##
## No pre-processing
## Resampling: Cross-Validated (15 fold)
## Summary of sample sizes: 5250, 5250, 5250, 5250, 5249, 5250, ...
## Resampling results across tuning parameters:
##
##   mincriterion  Accuracy   Kappa
##   0.01          0.8435433  0.5934086
##   0.50          0.8463896  0.5952389
##   0.99          0.8515490  0.6181477
##
## Accuracy was used to select the optimal model using the largest value.
## The final value used for the model was mincriterion = 0.99.
```

The final model had a mincriterion value of 0.99 and an accuracy of 0.8488913.

```
predTest_tree <- predict(model_tree, test_partition)
tree_cv = confusionMatrix(predTest_tree, test_partition$leave)
tree_cv
```

```
## Confusion Matrix and Statistics
##
##            Reference
## Prediction  Churn Not Churn
##   Churn       260       109
##   Not Churn   114       924
##
##                Accuracy : 0.8415
##                  95% CI : (0.8214, 0.8602)
##     No Information Rate : 0.7342
##     P-Value [Acc > NIR] : <2e-16
```

```
##
##                   Kappa : 0.5922
##
##  Mcnemar's Test P-Value : 0.7888
##
##             Sensitivity : 0.6952
##             Specificity : 0.8945
##          Pos Pred Value : 0.7046
##          Neg Pred Value : 0.8902
##              Prevalence : 0.2658
##          Detection Rate : 0.1848
##    Detection Prevalence : 0.2623
##       Balanced Accuracy : 0.7948
##
##        'Positive' Class : Churn
##
```

However, the prediction on the test data decreases to 84.15%.

Therefore, we are selecting decision tree with data partition into 80/20 (train/test) among logistic regression with data partition and cross validation and decision tree with data partition and cross validation. The selected model has 86.6% accuracy in train and 86.72% accuracy in the test data.