Final Project 439: Happiness Prediction

Aidan Ashrafi

12/2/2023

Question 1

```
library(readr)
happiness <- read.csv("/Users/aidanashrafi/Downloads/Happiness_2018.csv")
```

```
head(happiness)
```

```
##       country year life_ladder log_GDP_per_capita social_support
## 1 Afghanistan 2018       2.694              7.631          0.508
## 2     Albania 2018       5.004              9.497          0.684
## 3     Algeria 2018       5.043              9.370          0.799
## 4   Argentina 2018       5.793             10.032          0.900
## 5     Armenia 2018       5.062              9.490          0.814
## 6   Australia 2018       7.177             10.801          0.940
##   healthy_life_expectancy_at_birth freedom_to_make_life_choices generosity
## 1                           53.575                         0.374     -0.091
## 2                           69.075                         0.824      0.007
## 3                           66.300                         0.583     -0.151
## 4                           67.050                         0.846     -0.214
## 5                           66.825                         0.808     -0.169
## 6                           70.825                         0.916      0.143
##   perceptions_of_corruption positive_affect negative_affect
## 1                     0.928           0.385           0.405
## 2                     0.899           0.592           0.319
## 3                     0.759           0.534           0.293
## 4                     0.855           0.732           0.321
## 5                     0.677           0.535           0.455
## 6                     0.405           0.706           0.187
```

For each country, there is a happiness score (life_ladder). There are 8 numerical feature variables: log_GDP_per_capita, social_support, healthy_life_expectancy_at_birth, freedom_to_make_life_choices, generosity, perceptions_of_corruption, positive_affect, and negative_affect.

```
summary(happiness)
```

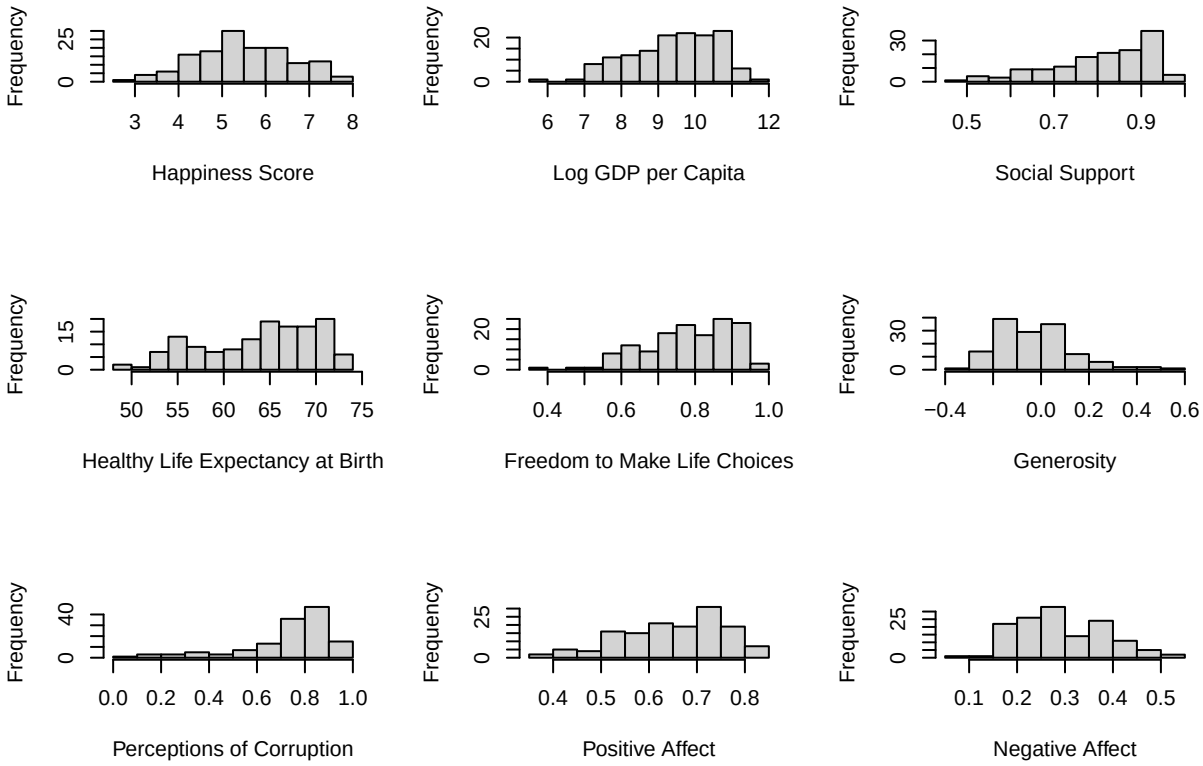
```
##    country               year       life_ladder    log_GDP_per_capita
##  Length:141         Min.   :2018   Min.   :2.694   Min.   : 5.935
##  Class :character   1st Qu.:2018   1st Qu.:4.769   1st Qu.: 8.540
##  Mode  :character   Median :2018   Median :5.465   Median : 9.552
##                     Mean   :2018   Mean   :5.499   Mean   : 9.391
##                     3rd Qu.:2018   3rd Qu.:6.249   3rd Qu.:10.346
##                     Max.   :2018   Max.   :7.858   Max.   :11.645
##
```

```
##  social_support   healthy_life_expectancy_at_birth freedom_to_make_life_choices
##  Min.   :0.4850   Min.   :49.30                    Min.   :0.3740
##  1st Qu.:0.7400   1st Qu.:59.20                    1st Qu.:0.7153
##  Median :0.8410   Median :65.21                    Median :0.7955
##  Mean   :0.8122   Mean   :63.89                    Mean   :0.7838
##  3rd Qu.:0.9090   3rd Qu.:68.66                    3rd Qu.:0.8762
##  Max.   :0.9840   Max.   :73.97                    Max.   :0.9700
##                   NA's   :3                        NA's   :1
##    generosity       perceptions_of_corruption positive_affect  negative_affect
##  Min.   :-0.33800   Min.   :0.0970            Min.   :0.3790   Min.   :0.0930
##  1st Qu.:-0.15100   1st Qu.:0.6910            1st Qu.:0.5770   1st Qu.:0.2155
##  Median :-0.05500   Median :0.7940            Median :0.6650   Median :0.2850
##  Mean   :-0.02876   Mean   :0.7346            Mean   :0.6526   Mean   :0.2929
##  3rd Qu.: 0.06100   3rd Qu.:0.8520            3rd Qu.:0.7375   3rd Qu.:0.3600
##  Max.   : 0.50900   Max.   :0.9520            Max.   :0.8410   Max.   :0.5440
##                     NA's   :8                 NA's   :2        NA's   :2
```

There is nothing unusual from the summary. However, there are some missing values. There are 3 NAs in healthy_life_expectancy_at_birth, 1 NA in freedom_to_make_life_choices, 8 NAs in perceptions_of_corruption, 2 NAs in positive_affect, and 2 NAs in negative_affect. This comes to a total of 16 missing values. However, since there are 1551, this is only 1% of all values in the dataset, so it is not a significant amount of missing data.

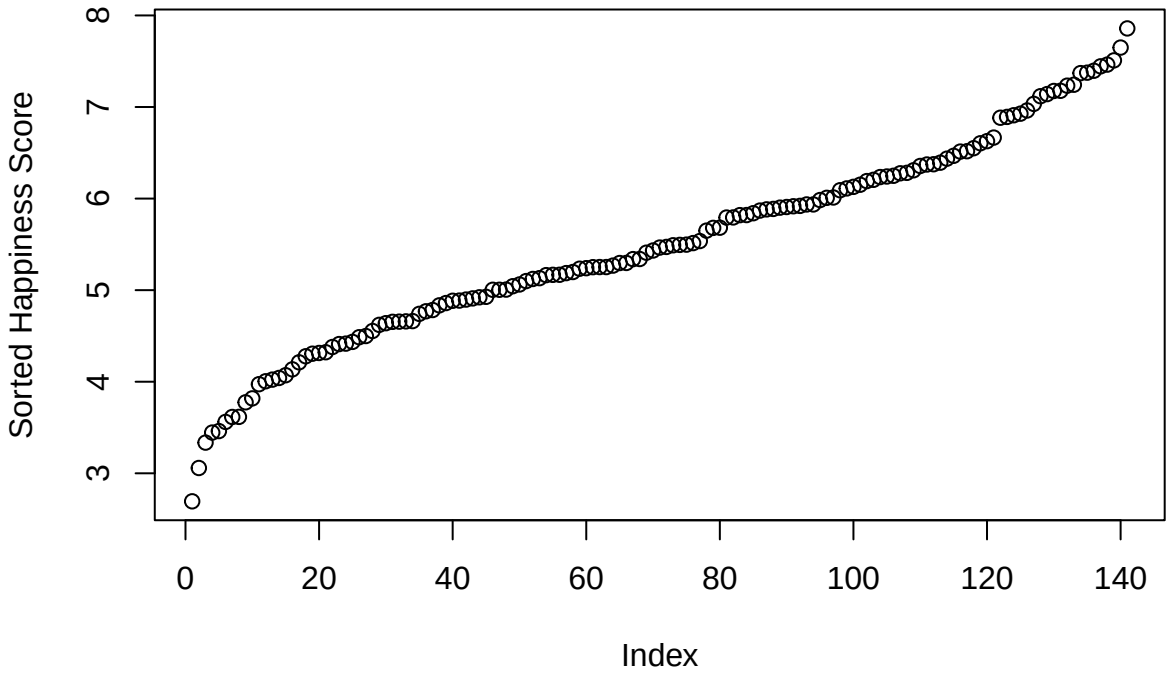
```r
par(mfrow = c(3, 3))

hist(happiness$life_ladder,xlab="Happiness Score",main="")
hist(happiness$log_GDP_per_capita,xlab="Log GDP per Capita",main="")
hist(happiness$social_support,xlab="Social Support",main="")
hist(happiness$healthy_life_expectancy_at_birth,xlab="Healthy Life Expectancy at Birth",main="")
hist(happiness$freedom_to_make_life_choices,xlab="Freedom to Make Life Choices",main="")
hist(happiness$generosity,xlab="Generosity",main="")
hist(happiness$perceptions_of_corruption,xlab="Perceptions of Corruption",main="")
hist(happiness$positive_affect,xlab="Positive Affect",main="")
hist(happiness$negative_affect,xlab="Negative Affect",main="")
```

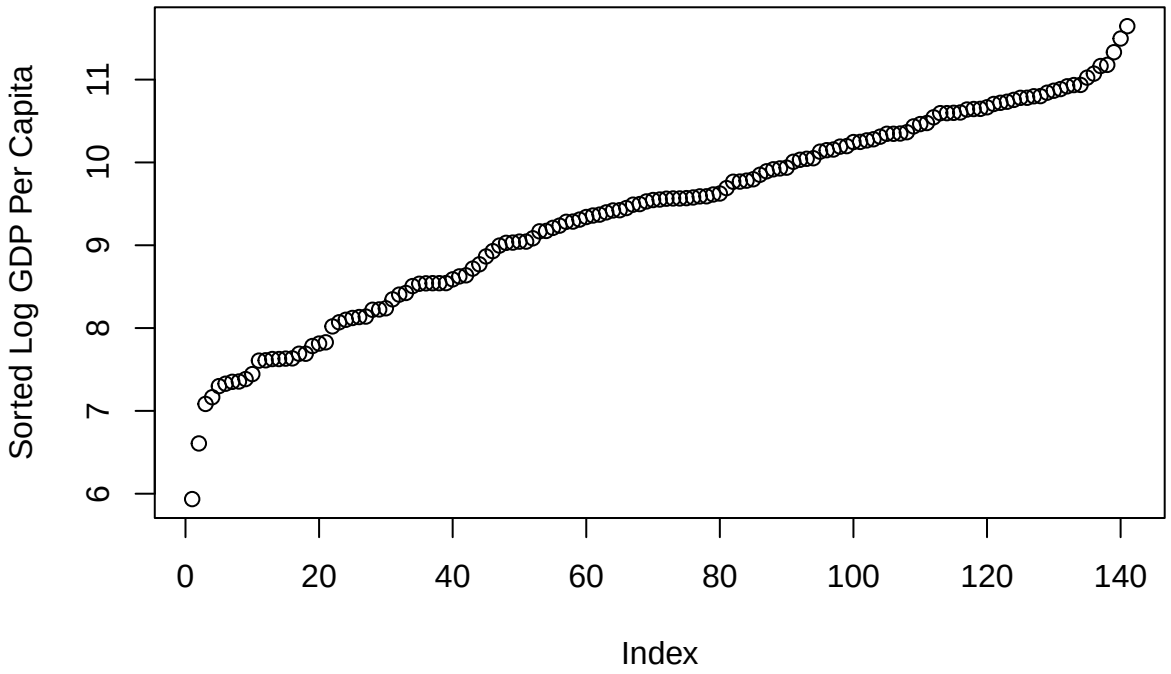


Happiness is normal distributed with an average around 5.5. All other variables are skewed right except generosity which is skewed left.The skew does not seem to be extreme for most variables except for perception of corruption which has large frequency at high levels of corruption perceptions.

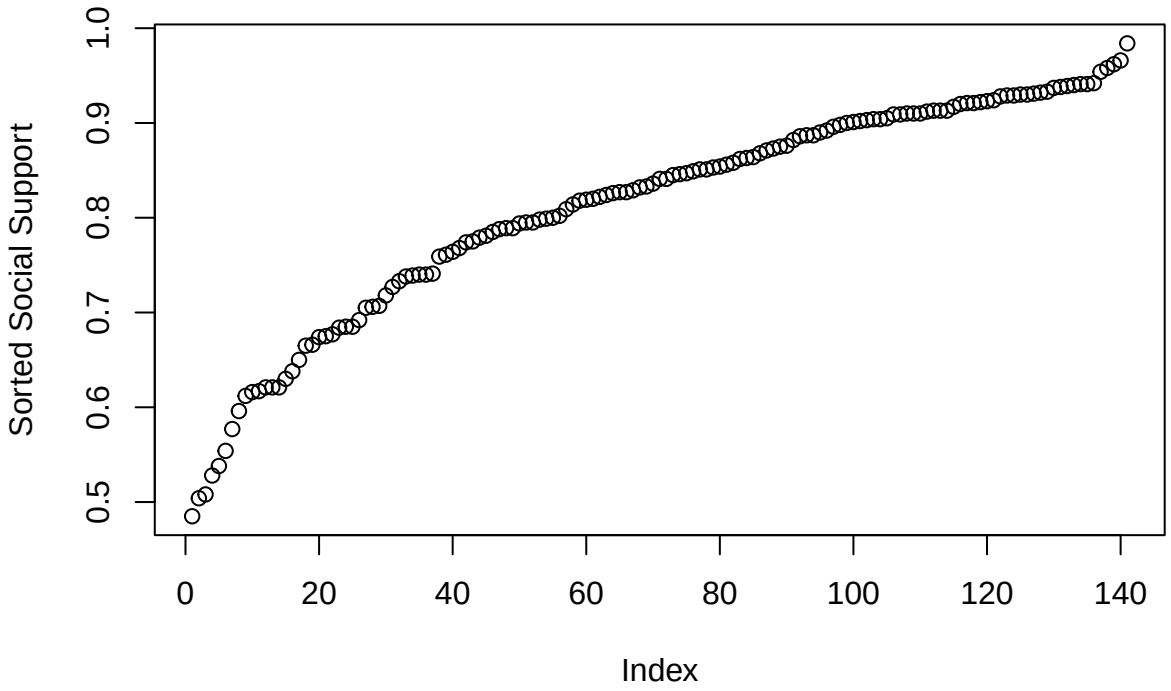
```
plot(sort(happiness$life_ladder),ylab="Sorted Happiness Score")
```



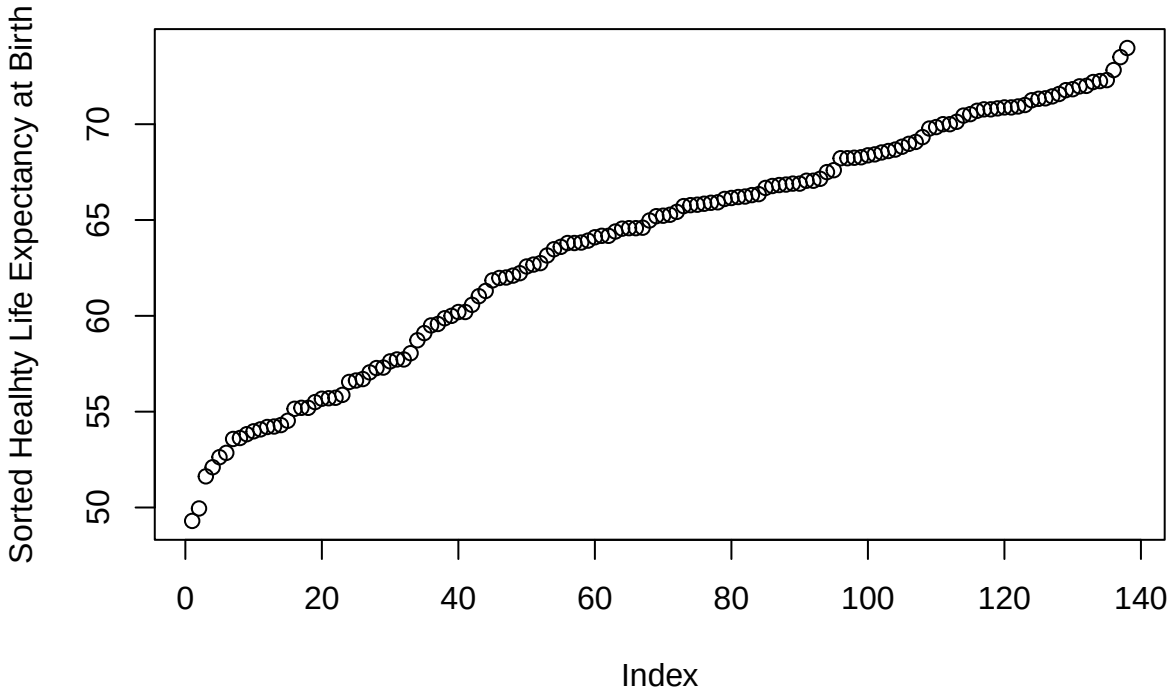
```r
plot(sort(happiness$log_GDP_per_capita),ylab="Sorted Log GDP Per Capita")
```



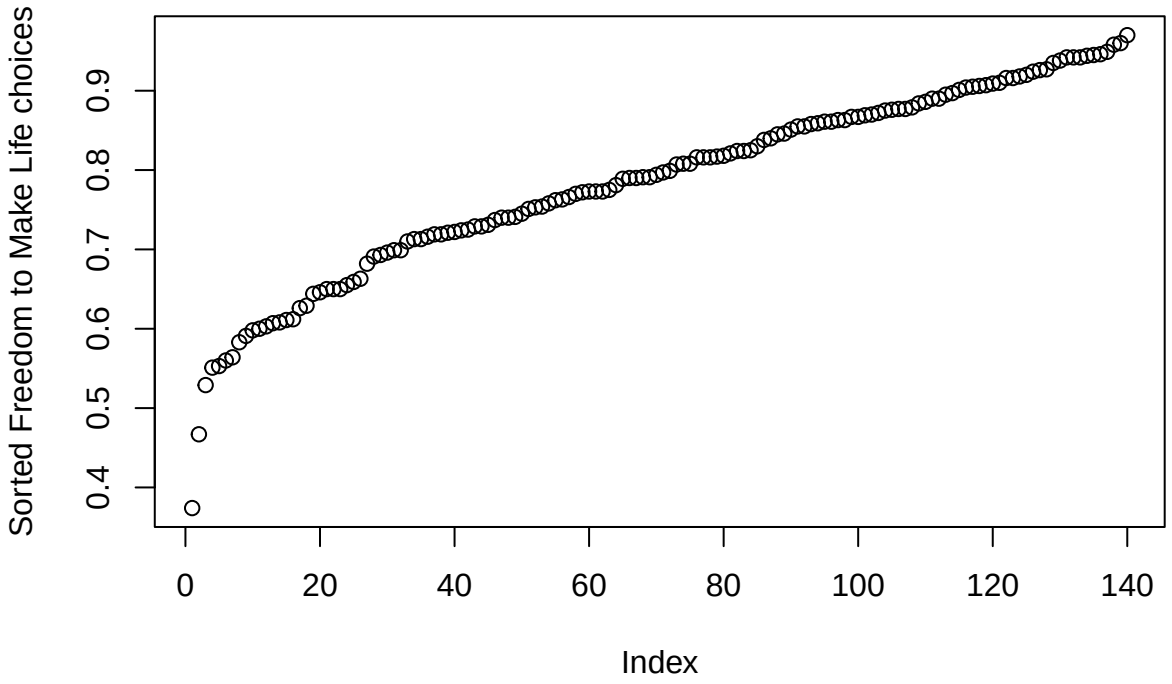

```r
plot(sort(happiness$social_support),ylab="Sorted Social Support")
```



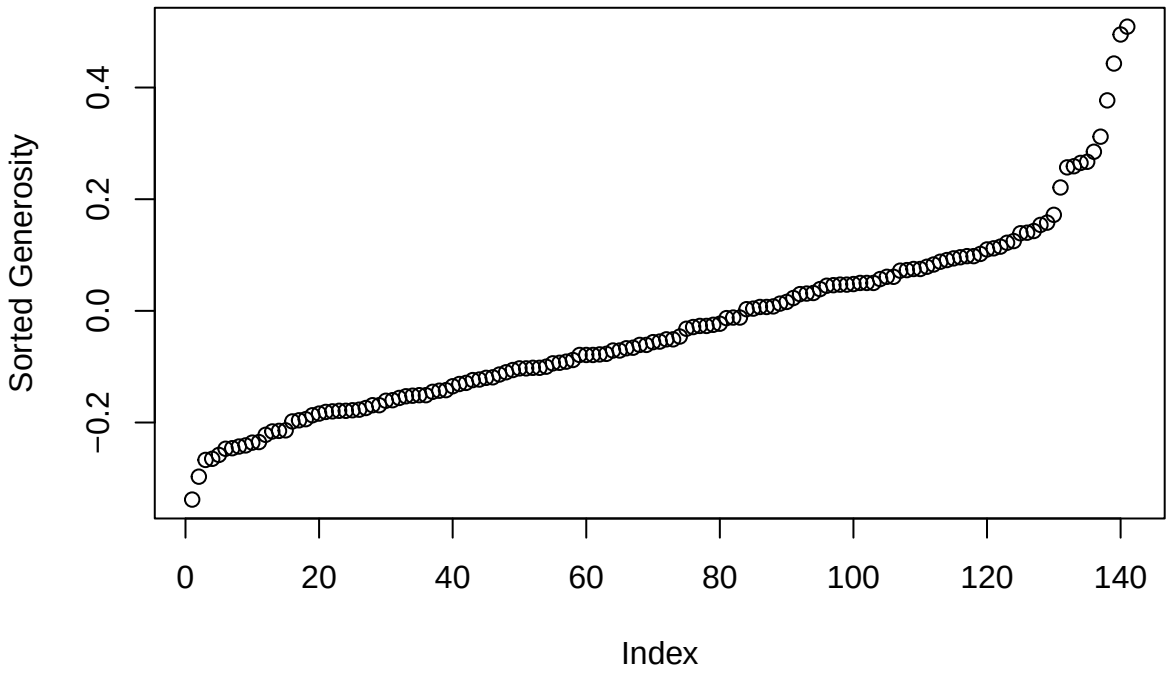

```r
plot(sort(happiness$healthy_life_expectancy_at_birth),ylab="Sorted Healhty Life Expectancy at Birth")
```



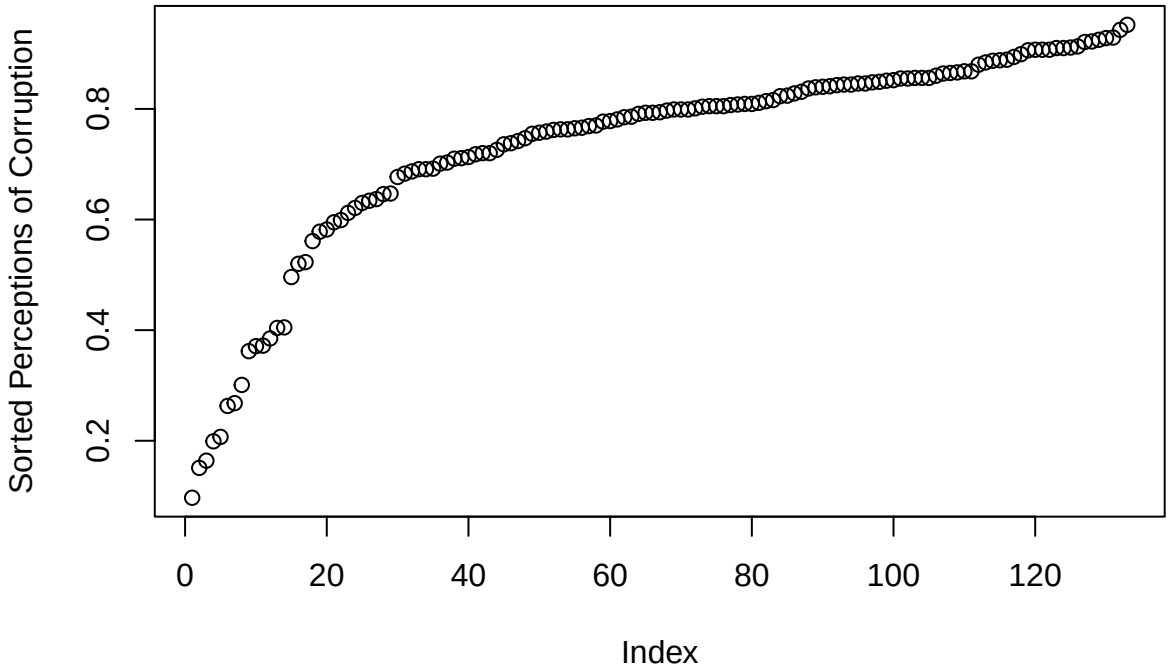

```
plot(sort(happiness$freedom_to_make_life_choices),ylab="Sorted Freedom to Make Life choices")
```



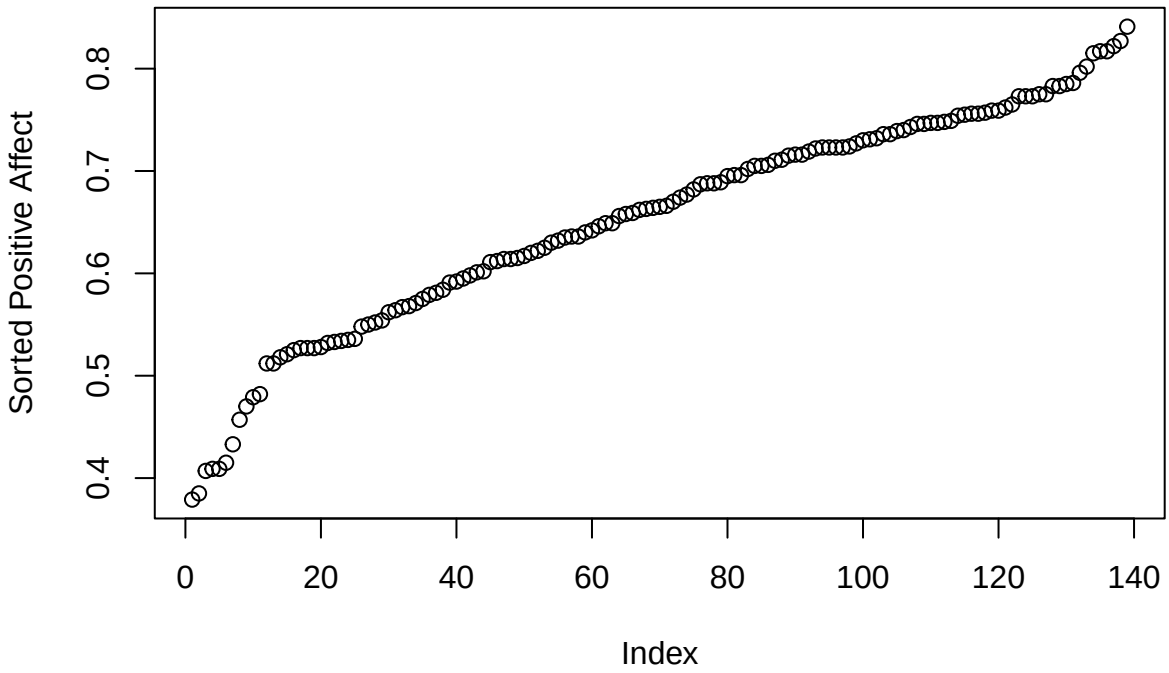

```
plot(sort(happiness$generosity),ylab="Sorted Generosity")
```



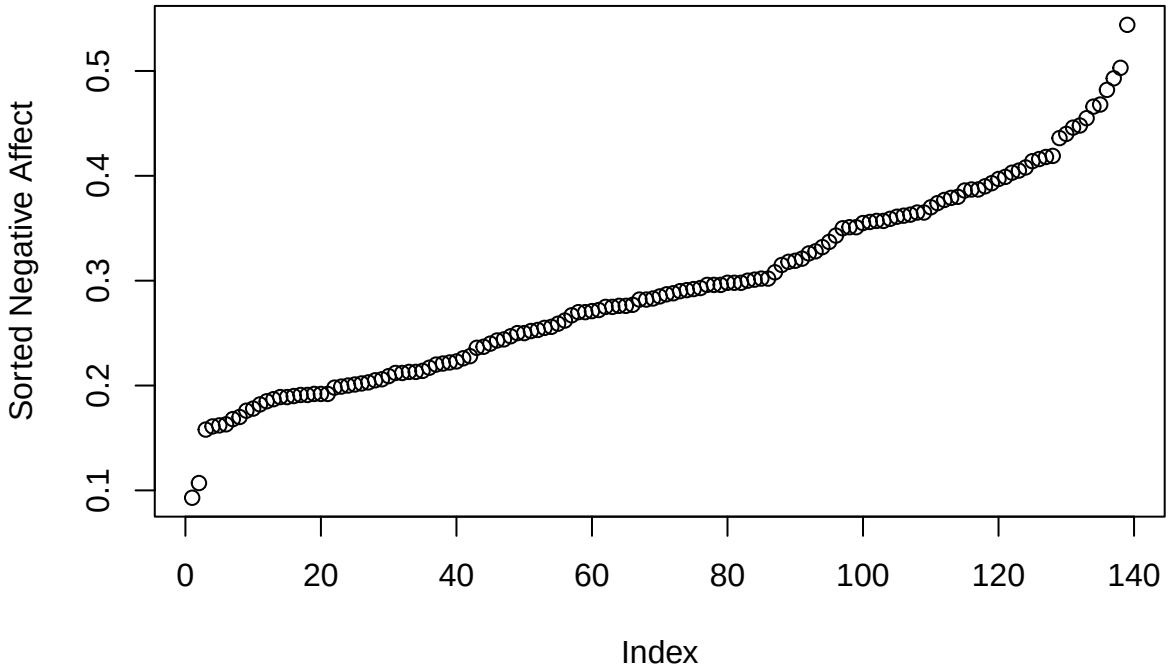

```
plot(sort(happiness$perceptions_of_corruption),ylab="Sorted Perceptions of Corruption")
```




```
plot(sort(happiness$positive_affect),ylab="Sorted Positive Affect")
```




```
plot(sort(happiness$negative_affect),ylab="Sorted Negative Affect")
```



Looking at the individual cases within the distribution of the feature variables, most variables do not have any outliers. The negative affect variable has 2 cases that are very low compared to the rest. More analysis must be done to determine if these cases are outliers.

Question 2

```
mymodel <- lm(life_ladder~.-country-year, data=happiness)
summary(mymodel)
```

```
##
## Call:
## lm(formula = life_ladder ~ . - country - year, data = happiness)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.66321 -0.33003  0.02126  0.29709  1.66671
##
## Coefficients:
##                                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)                      -4.62111    0.87144  -5.303 5.28e-07 ***
## log_GDP_per_capita                0.35544    0.08478   4.192 5.31e-05 ***
## social_support                    2.83854    0.72502   3.915 0.000150 ***
## healthy_life_expectancy_at_birth  0.03672    0.01448   2.535 0.012536 *
## freedom_to_make_life_choices      0.78692    0.55499   1.418 0.158812
## generosity                        0.13951    0.34495   0.404 0.686613
## perceptions_of_corruption        -0.76970    0.30685  -2.508 0.013462 *
## positive_affect                   2.11173    0.56563   3.733 0.000290 ***
## negative_affect                   2.47878    0.72805   3.405 0.000901 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.5326 on 120 degrees of freedom
##   (12 observations deleted due to missingness)
## Multiple R-squared:  0.7829, Adjusted R-squared:  0.7684
## F-statistic: 54.09 on 8 and 120 DF,  p-value: < 2.2e-16
```

We removed the country and year features since country is informational and year is constant for all cases. The R^2 is 0.7684, meaning that 76.84% of the variability in happiness score is explained by the model. The F-statistic is 54.09 with a very low p-value (< 2.2e-16), indicating that the model is statistically significant. This means that at least one of the predictor variables is significantly related to the response variable.

GDP: This feature variable is statistically significant at a p-value of <0.001. When the log of GDP per capita increases by 1, the happiness score increases by 0.36. This makes intuitive sense as you would expect a country with higher GDP to have happier citizens.

Social Support: This feature variable is statistically significant at a p-value of <0.001. When the social support rating increases by 1, the happiness score increases by 2.84. This makes intuitive sense as you would expect when people have those they feel like they can rely on, they will be happier.

Life Expectancy: This feature variable is statistically significant at a p-value of 0.05. When the life expectancy increases by 1, the happiness score increases by 0.04. This is slightly surprising as you would expect a higher life expectancy to have a high impact in a happier population. However, it is still a positive coefficient, so it does still lead to happiness, just at a lesser extent.

Freedom: This feature variable is not statistically significant at a p-value of 0.1. When the freedom score increases by 1, the happiness score increases by 0.79. This makes intuitive sense as you would expect when people have freedom to make their own life choices, they will be happier.

Generosity: This feature variable is not statistically significant at a p-value of 0.1. When the freedom score increases by 1, the happiness score increases by 0.14. This makes intuitive sense as you would expect when people are generous towards others, they will be happier.

Perceptions of Corruption: This feature variable is statistically significant at a p-value of 0.1. When the perception of corruption score increases by 1, the happiness score decreases by 0.77. This makes intuitive sense as you would expect that when people feel there is corruption within the government and business, they will be less happy.

Positive Affect: This feature variable is statistically significant at a p-value of <0.001. When the positive affect score increases by 1, the happiness score increases by 2.11. This makes intuitive sense as you would expect when people have higher averages of laugh, enjoyment and doing interesting things, they will be happier.

Negative Affect: This feature variable is statistically significant at a p-value of <0.001. When the negative affect score increases by 1, the happiness score increases by 2.11. This is very surprising since you would expect when people have higher averages of worry, sadness and anger, they are less happy

Question 3

```
library(readr)
Data <- read_csv("/Users/aidanashrafi/Downloads/Happiness_2018.csv")
```

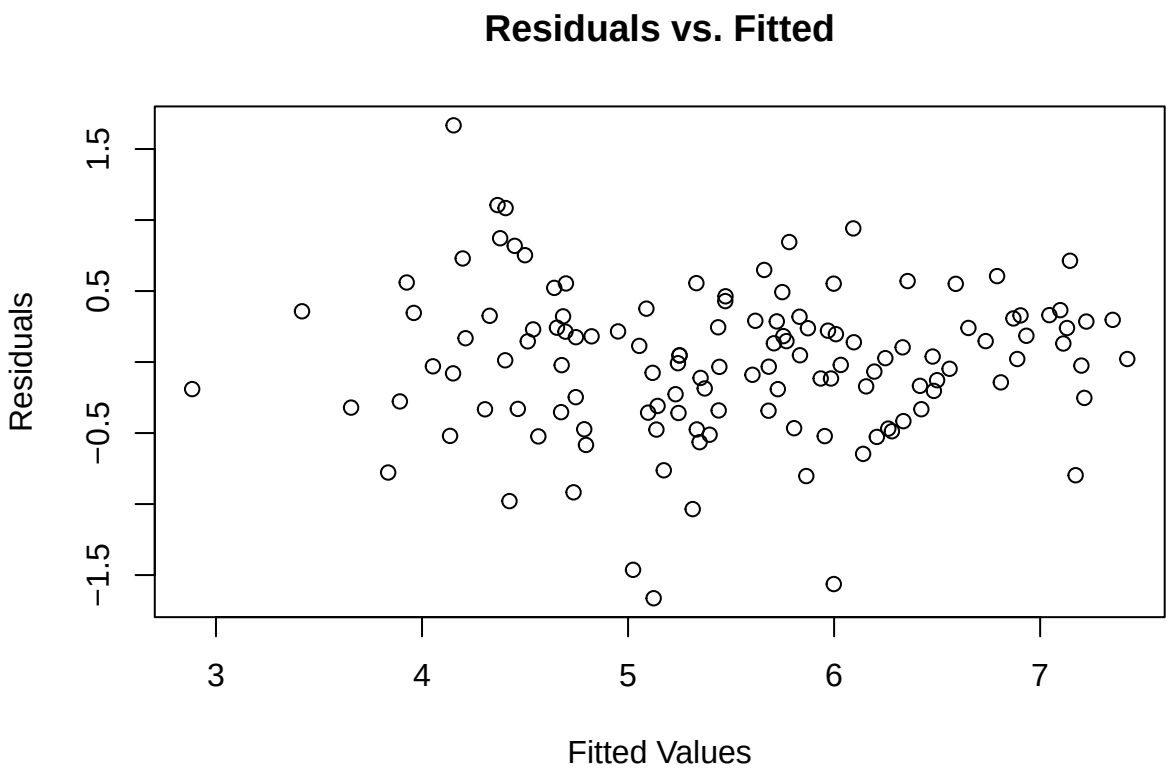
```
## Rows: 141 Columns: 11
## -- Column specification --------------------------------------------------------
## Delimiter: ","
## chr  (1): country
## dbl (10): year, life_ladder, log_GDP_per_capita, social_support, healthy_lif...
##
## i Use `spec()` to retrieve the full column specification for this data.
## i Specify the column types or set `show_col_types = FALSE` to quiet this message.
```

```
mymodel <- lm(life_ladder ~ log_GDP_per_capita + social_support + healthy_life_expectancy_at_birth + fr
```

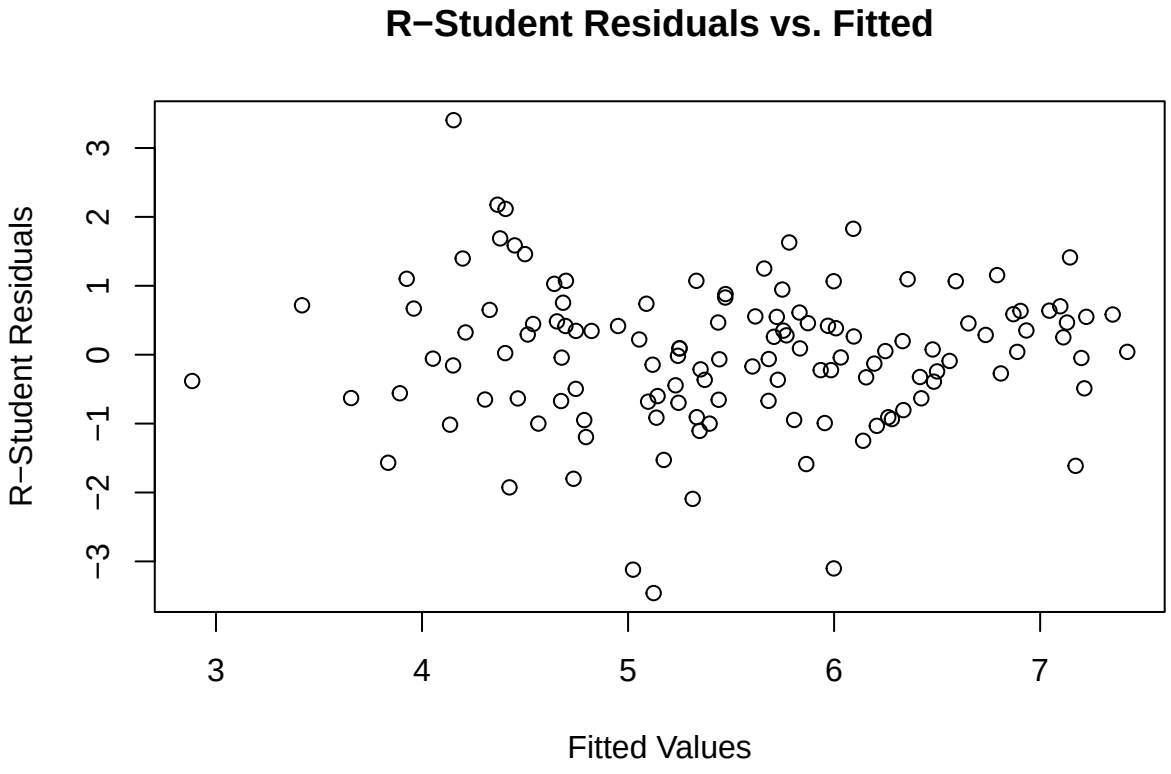
3 (a)

When checking the constant variance assumption for the errors we wish to make sure they are unbiased and homoscedastic. For biasedness, the points on the graph of fitted values vs. errors should be centered around a horizontal band and not exhibiting any trend such as positive linear, negative linear, etc. For scedasticity, the points should all be evenly spaced around this horizontal band. Otherwise, the constant variance assumption may not hold. Pictured below are two graphs: both have the fitted happiness values on the horizontal axis. The first graph has the regular residuals on the vertical axis, while the second has a more precise r-student residual plotted.

```
plot(fitted(mymodel), resid(mymodel), main = "Residuals vs. Fitted", xlab = 'Fitted Values', ylab = 'Re
```




```
plot(fitted(mymodel), rstudent(mymodel), main = "R-Student Residuals vs. Fitted", xlab = 'Fitted Values
```



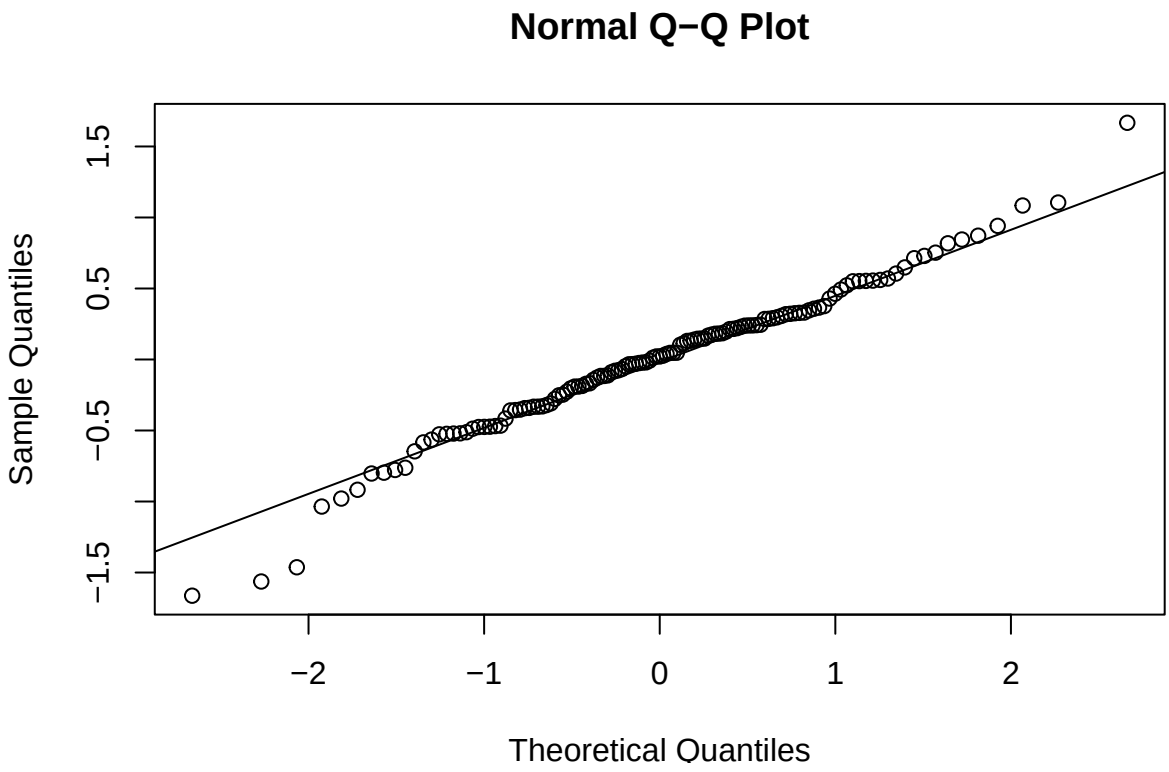
The graphs are shaped in the same manner, so they will both have the same characteristics. Both clumps of fitted-residual points are centered around a horizontal band at 0. Therefore, the errors seem to be unbiased. The scedasticity is a bit more difficult to interpret. It looks as if the errors in the 4-5 range of the fitted values are a bit more spread out than the errors in the 5-7 range in both cases. However, the spacing does

not seem so concerning so as to make us question the constant variance assumption for the errors.

3 (b)

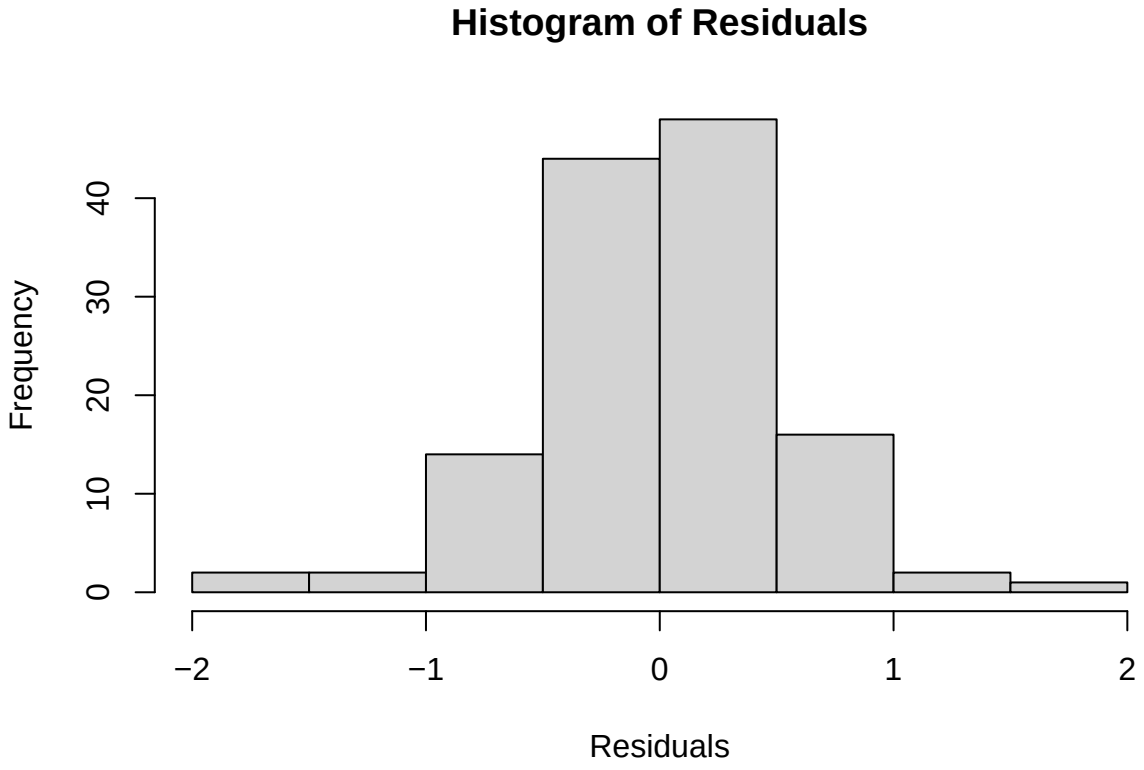
There are numerous ways to check the normality assumption for the errors. The top way is the qq-plot, where the points should fall right along a hypothetical qq-line, with some deviation in both tails. Attached below is the qq-plot of our residuals:

```
qqnorm(resid(mymodel))
qqline(resid(mymodel))
```



Aside from the typical trailing off of the points in the tails, the residuals seem to be normally distributed. It is possible there is a slight S-shape to the graph, which would suggest a short-tailed distribution, but it looks to conform to the normality assumption for the most part. Looking at the histogram for the residuals below, we can see that, for the most part, this assumption is confirmed.

```
hist(resid(mymodel), main = "Histogram of Residuals", xlab = "Residuals")
```



The last way we can check the normality of our errors is using the Shapiro-Wilk normality test. Using the shapiro.test(resid(mymodel)) command in R we get a p-value. We want this p-value to be large so we fail to reject the null hypothesis, as the null hypothesis is that the distribution of the errors is normal. Interestingly, we got a p-value of 0.039 for the Shapiro test, which is quite small and would lead us to reject the hypothesis that the errors are normal.

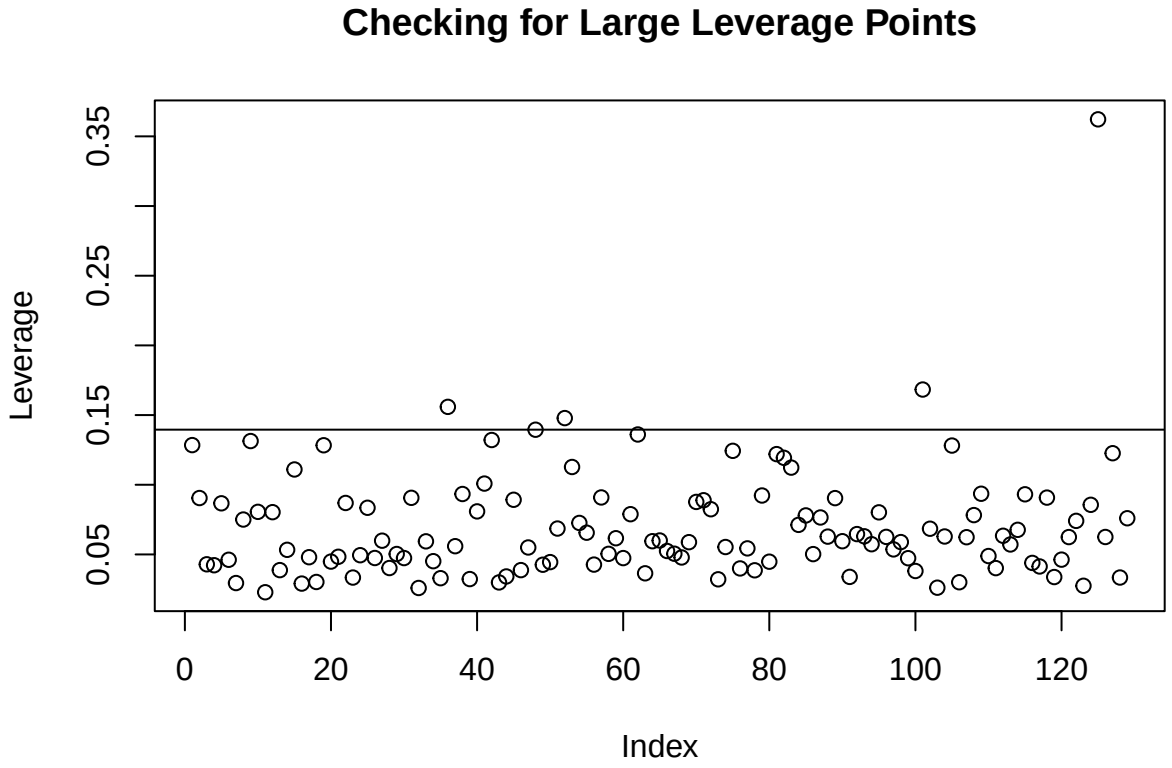
```
shapiro.test(resid(mymodel))
```

```
##
##  Shapiro-Wilk normality test
##
## data:  resid(mymodel)
## W = 0.9786, p-value = 0.039
```

3 (c)

Large leverage points are points that are far outside the "centroid" of all the points, denoted as x-bar. This point would dramatically expand the convex hull that encompasses all of the leverage points and could have too much influence on the model. We can plot these points using the plot() function on lm.influence(mymodel)$hat. As a rule of thumb, a "large" leverage point will be greater than 2 * ([k+1] /n). In our case, this value is 2 * (9/129), or 0.139. The resulting plot is below, with a line for this value:

```
plot(lm.influence(mymodel)$hat, main = "Checking for Large Leverage Points", ylab = "Leverage")
abline(h=2*(9/129))
```



There seem to be 4 or so points that are above this line, with one extremely large point –in comparison to the others– in the top right. We can figure out which ones specifically by asking R to search for them.

```
leverage <- lm.influence(mymodel)$hat
leverage[which.max(abs(leverage))]
```

```
##    137
## 0.3622
```

```
leverage[leverage>(2*(9/129))]
```

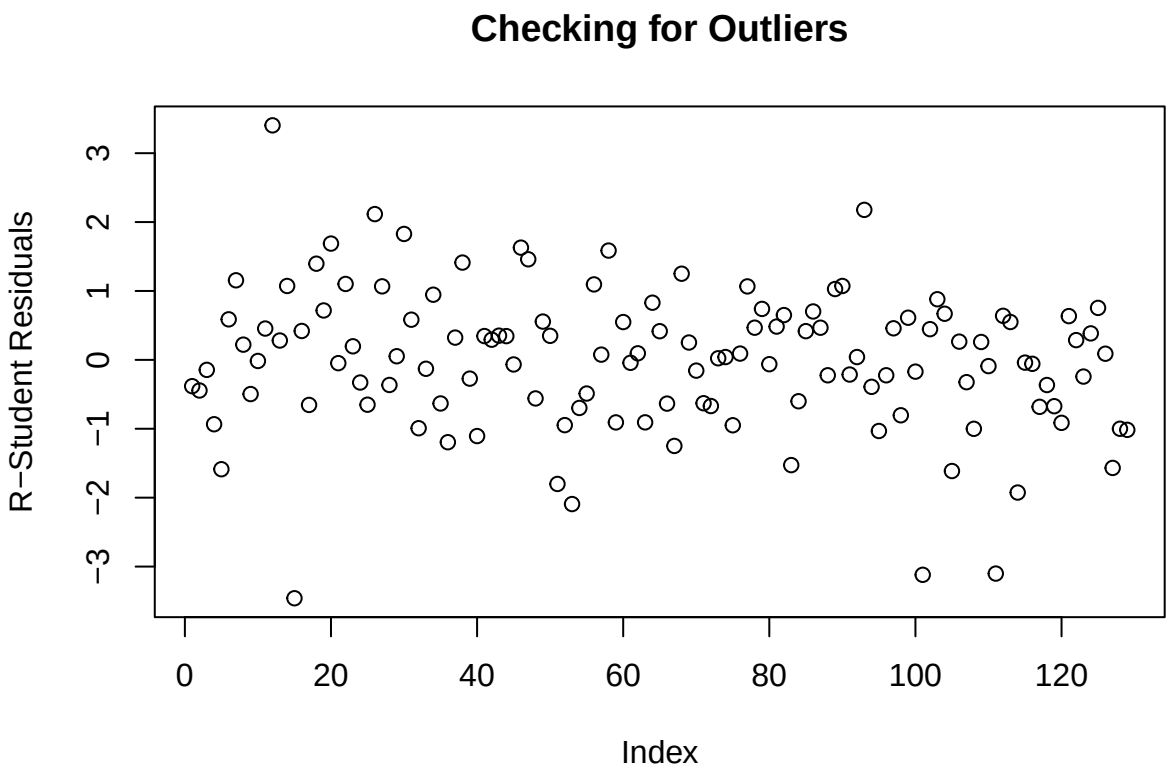
```
##        39        55       107       137
## 0.1558946 0.1478289 0.1683276 0.3622000
```

So, the maximum leverage point is 0.3622, corresponding to Venezuela. The other three points that are above our rule-of-thumb delineator are Eswatini, Indonesia and Rwanda.

3 (d)

To check for potential outliers we can use the R-student residuals for each observation (which we also used in 3a to check the error constant variance assumption). A plot for these r-student residuals is below:

```
plot(rstudent(mymodel), main = 'Checking for Outliers', ylab = 'R-Student Residuals')
```



There seem to be a couple of points that are of concern but, so long as the maximum r-student residual is proven to not be an outlier, the other ones will not be. We can define a variable jack as rstudent(mymodel) and use a similar process that we did in the previous question. The line jack[which.max(abs(jack))] gives us the r-student (jackknife) residual with the highest absolute value.

```
jack <- rstudent(mymodel)
jack[which.max(abs(jack))]
```

```
##        15
## -3.459753
```

This value is -3.46, corresponding to the 15th data point (Botswana) in our data. Now, in order to make sure we do not erroneously claim that points are outliers, we make our critical value that we compare these residuals to extremely high. The Bonferroni critical value we will use is defined by: t [alpha/(2n), n-p-1]. Given that our n is 129 and our p is 9, our desired critical value can be summoned using qt(1-0.05/(2*129), 119).

```
qt(1-(0.05/(2*129)), 119)
```

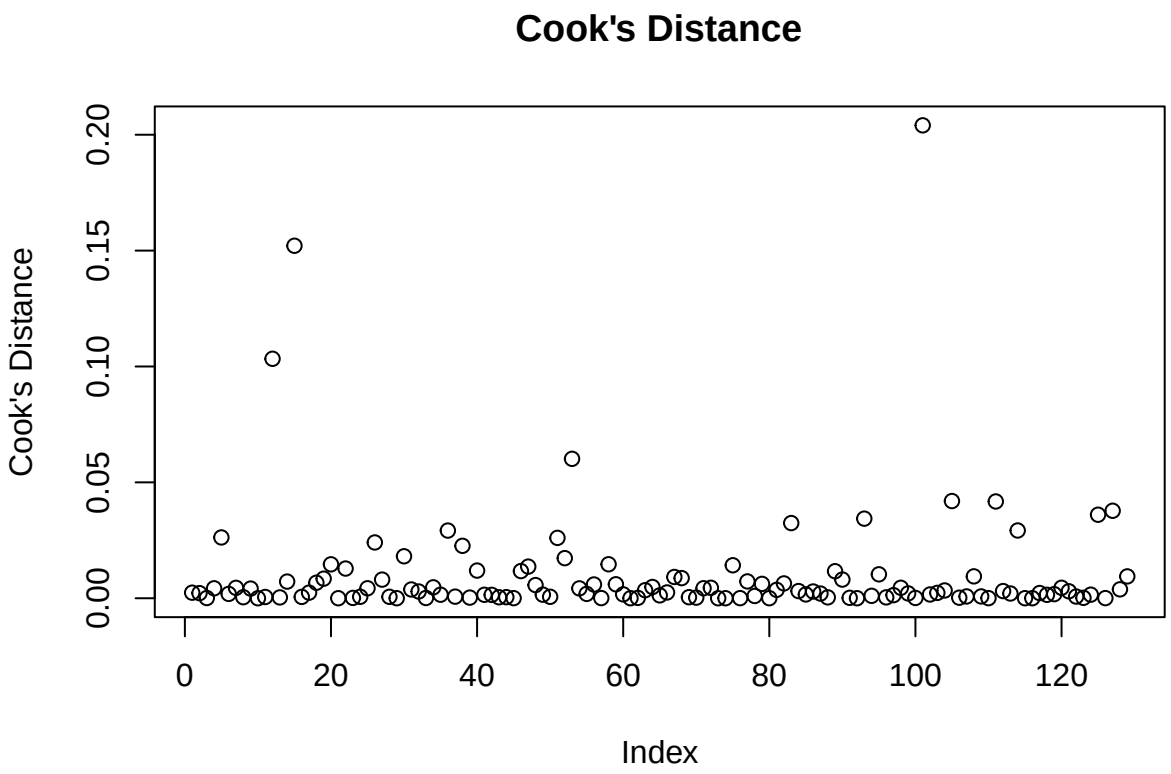
```
## [1] 3.652358
```

This returns 3.65, which is larger than the absolute value of -3.46, and therefore we fail to reject the null hypothesis that all the points come from the same model. There are no outliers.

3 (e)

The manner in which we will check for influential points is using Cook's Distance. As a rule of thumb, if the maximum Cook's Distance for a point is under 0.5, there are no significantly influential points. Attached below is a plot of these points corresponding to their index.

```
cook <- cooks.distance(mymodel)
plot(cook, main = "Cook's Distance", ylab = "Cook's Distance")
```



The maximum Cook's Distance is the point roughly near the 100th index. Now, this point seems to be in the 0.2 area and therefore is within the range of our rule-of-thumb. Just for the sake of thoroughness, we will find out the exact distance and what this point is.

```
cook[which.max(abs(cook))]
```

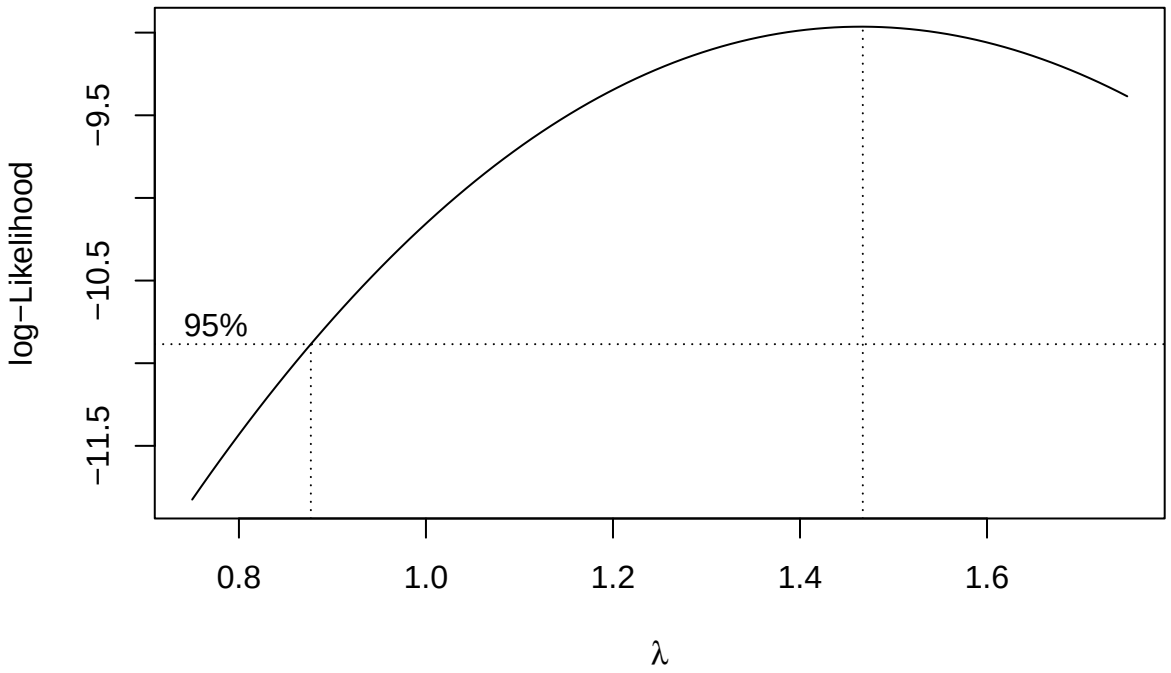
```
##       107
## 0.2040419
```

This returns a value of 0.2040419, corresponding to the 107th data point (Rwanda). There do not seem to be any influential points.

Question 4

Because all of our response data is positive, we can apply a Box-Cox transformation. To see what might be best to set our lambda to, we should plot the Box-Cox line and see where the confidence interval falls. Pictured below is the plot.

```
library(MASS)
boxcox(mymodel, plotit = T, lambda = seq(0.75, 1.75, by = 0.1))
```



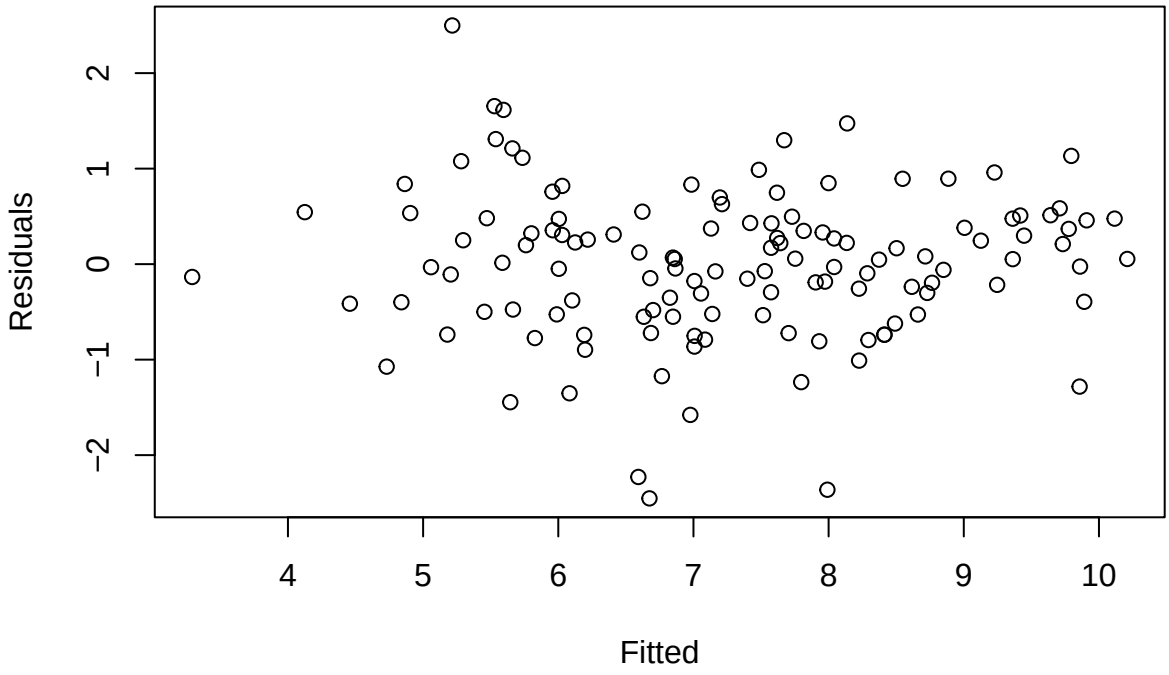
The 95% confidence interval seems to lie from 0.87 to 1.45 or so. 1 is within this interval and is therefore plausible, so there is not a huge need to apply a Box-Cox transformation. However, because 1.16 is the middle of this interval, we will create a new model with our response variable raised to the power of 1.16 and see if the diagnostics in 3 (a) and (b) improve. Attached below is the summary of our new model, the plots of our new model's fitted values up against our new model's residuals and jackknife/R-student residuals, and a qq-plot for our new model:

```
newmodel <- lm((life_ladder)^1.16 ~ log_GDP_per_capita + social_support + healthy_life_expectancy_at_bi
summary(newmodel)
```

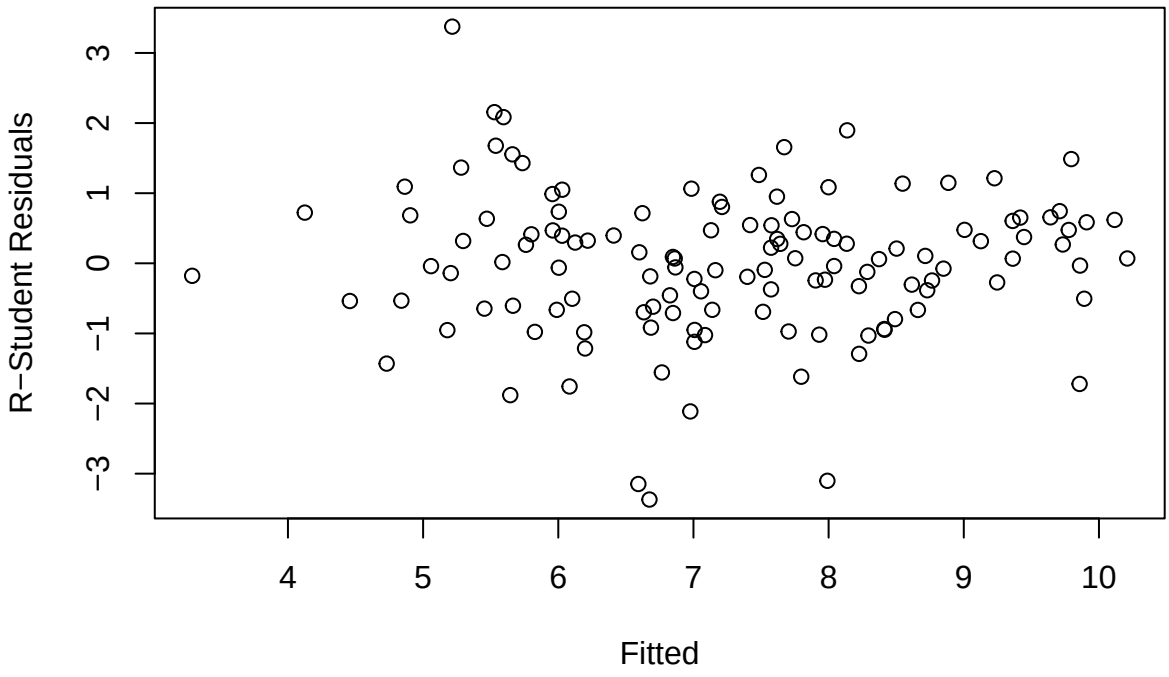
```
##
## Call:
## lm(formula = (life_ladder)^1.16 ~ log_GDP_per_capita + social_support +
##     healthy_life_expectancy_at_birth + freedom_to_make_life_choices +
##     generosity + perceptions_of_corruption + positive_affect +
##     negative_affect, data = Data)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2.45292 -0.48120  0.05215  0.47351  2.49901
##
## Coefficients:
##                                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)                       -7.92308    1.31670  -6.017 1.98e-08 ***
## log_GDP_per_capita                 0.54033    0.12811   4.218 4.81e-05 ***
## social_support                     4.26160    1.09546   3.890 0.000165 ***
## healthy_life_expectancy_at_birth   0.05546    0.02189   2.534 0.012567 *
## freedom_to_make_life_choices       1.13674    0.83856   1.356 0.177775
## generosity                         0.22453    0.52121   0.431 0.667392
## perceptions_of_corruption         -1.27559    0.46363  -2.751 0.006856 **
## positive_affect                    3.20988    0.85464   3.756 0.000268 ***
## negative_affect                    3.70012    1.10004   3.364 0.001033 **
## ---
```

```
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.8048 on 120 degrees of freedom
##   (12 observations deleted due to missingness)
## Multiple R-squared:  0.7853, Adjusted R-squared:  0.771
## F-statistic: 54.86 on 8 and 120 DF,  p-value: < 2.2e-16
```

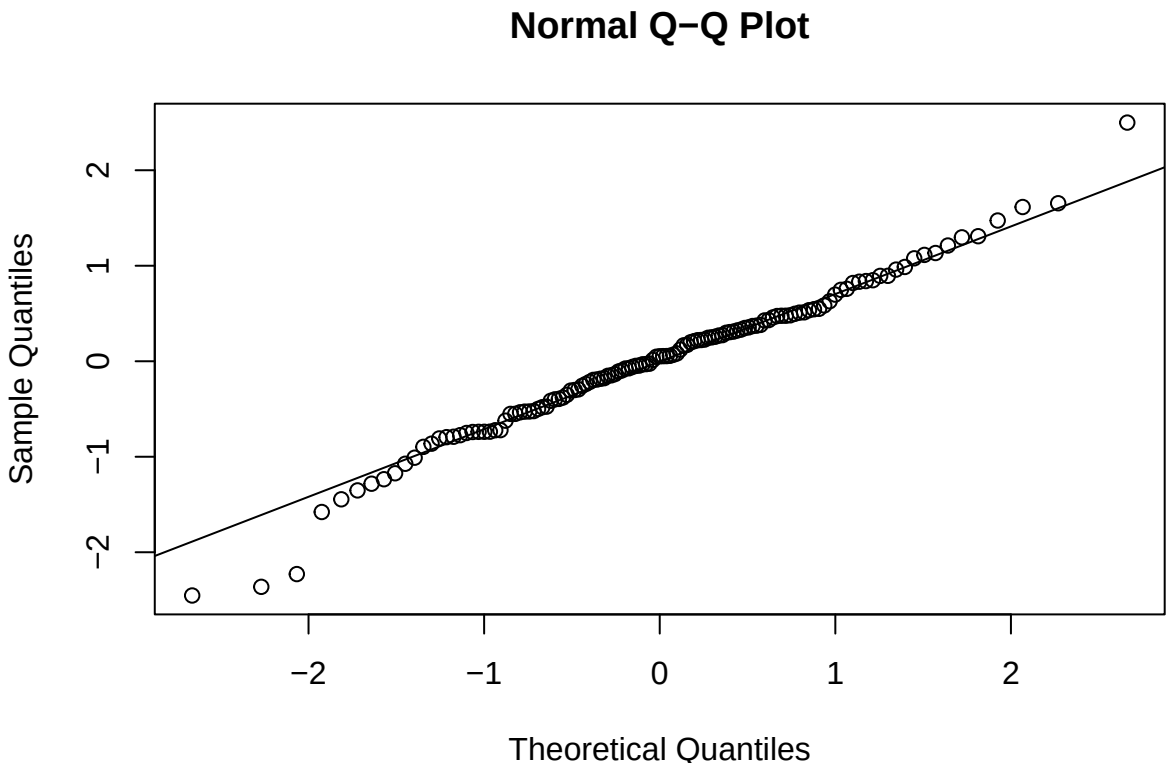
```
plot(fitted(newmodel), resid(newmodel), xlab = 'Fitted', ylab = 'Residuals')
```




```
plot(fitted(newmodel), rstudent(newmodel), xlab = 'Fitted', ylab = 'R-Student Residuals')
```



```
qqnorm(resid(newmodel))
qqline(resid(newmodel))
```



Compared to the plots we generated for 3a, there does not seem to be a significantly more homoscedastic distribution of the errors here. With regards to the qq-plot, again, it does not seem to be significantly more normal than the previous model.

We can use a Shapiro-Wilkes test and see if the p-value is substantially larger than the one we obtained in the previous model:

```
shapiro.test(resid(newmodel))
```

```
##
##  Shapiro-Wilk normality test
##
## data:  resid(newmodel)
## W = 0.97989, p-value = 0.05229
```

This p-value is fairly larger than the original p-value of 0.039 from a percentage perspective, but, at an alpha = 0.1 level of significance, would still lead to a rejection of the null hypothesis.

Therefore, we conclude that applying a Box-Cox transformation to our model would make it conform to the Gauss-Markov assumptions slightly better, but not enough to justify changing it over, which would make the model much harder to interpret.

Question 5

5. explore the possibility of serial correlation in the errors and applying a generalized least squares estimator as a way to correct it. You may need to sort the cases in ascending order of time, of the values of a predictor, or other way that makes sense (e.g., by zip code if appropriate, etc.).

```
happiness_with_na <- read.csv("/Users/aidanashrafi/Downloads/Happiness_2018.csv")
happiness = na.omit(happiness_with_na)
full_model = lm(life_ladder~.-country-year, data=happiness)
head(happiness)
```

```
##       country year life_ladder log_GDP_per_capita social_support
## 1 Afghanistan 2018       2.694              7.631          0.508
## 2     Albania 2018       5.004              9.497          0.684
## 3     Algeria 2018       5.043              9.370          0.799
## 4   Argentina 2018       5.793             10.032          0.900
## 5     Armenia 2018       5.062              9.490          0.814
## 6   Australia 2018       7.177             10.801          0.940
##   healthy_life_expectancy_at_birth freedom_to_make_life_choices generosity
## 1                           53.575                         0.374     -0.091
## 2                           69.075                         0.824      0.007
## 3                           66.300                         0.583     -0.151
## 4                           67.050                         0.846     -0.214
## 5                           66.825                         0.808     -0.169
## 6                           70.825                         0.916      0.143
##   perceptions_of_corruption positive_affect negative_affect
## 1                     0.928           0.385           0.405
## 2                     0.899           0.592           0.319
## 3                     0.759           0.534           0.293
## 4                     0.855           0.732           0.321
## 5                     0.677           0.535           0.455
## 6                     0.405           0.706           0.187
```

```
summary(happiness)
```

```
##    country               year       life_ladder    log_GDP_per_capita
##  Length:129         Min.   :2018   Min.   :2.694   Min.   : 5.935
##  Class :character   1st Qu.:2018   1st Qu.:4.783   1st Qu.: 8.535
##  Mode  :character   Median :2018   Median :5.472   Median : 9.552
##                     Mean   :2018   Mean   :5.510   Mean   : 9.375
##                     3rd Qu.:2018   3rd Qu.:6.241   3rd Qu.:10.346
##                     Max.   :2018   Max.   :7.858   Max.   :11.645
##  social_support   healthy_life_expectancy_at_birth freedom_to_make_life_choices
##  Min.   :0.4850   Min.   :49.30                    Min.   :0.3740
##  1st Qu.:0.7380   1st Qu.:58.05                    1st Qu.:0.7130
##  Median :0.8410   Median :65.22                    Median :0.7910
##  Mean   :0.8089   Mean   :63.83                    Mean   :0.7802
##  3rd Qu.:0.9090   3rd Qu.:68.83                    3rd Qu.:0.8750
##  Max.   :0.9660   Max.   :73.97                    Max.   :0.9700
##    generosity      perceptions_of_corruption positive_affect  negative_affect
##  Min.   :-0.3380   Min.   :0.0970            Min.   :0.3790   Min.   :0.1070
##  1st Qu.:-0.1420   1st Qu.:0.6910            1st Qu.:0.5790   1st Qu.:0.2210
##  Median :-0.0510   Median :0.7940            Median :0.6650   Median :0.2870
##  Mean   :-0.0272   Mean   :0.7337            Mean   :0.6537   Mean   :0.2957
##  3rd Qu.: 0.0720   3rd Qu.:0.8520            3rd Qu.:0.7400   3rd Qu.:0.3610
##  Max.   : 0.5090   Max.   :0.9520            Max.   :0.8410   Max.   :0.5440
```

```
summary(full_model)
```

```
##
## Call:
```

```
## lm(formula = life_ladder ~ . - country - year, data = happiness)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.66321 -0.33003  0.02126  0.29709  1.66671
##
## Coefficients:
##                                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)                      -4.62111    0.87144  -5.303 5.28e-07 ***
## log_GDP_per_capita                0.35544    0.08478   4.192 5.31e-05 ***
## social_support                    2.83854    0.72502   3.915 0.000150 ***
## healthy_life_expectancy_at_birth  0.03672    0.01448   2.535 0.012536 *
## freedom_to_make_life_choices      0.78692    0.55499   1.418 0.158812
## generosity                        0.13951    0.34495   0.404 0.686613
## perceptions_of_corruption        -0.76970    0.30685  -2.508 0.013462 *
## positive_affect                   2.11173    0.56563   3.733 0.000290 ***
## negative_affect                   2.47878    0.72805   3.405 0.000901 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.5326 on 120 degrees of freedom
## Multiple R-squared:  0.7829, Adjusted R-squared:  0.7684
## F-statistic: 54.09 on 8 and 120 DF,  p-value: < 2.2e-16
```

```
ordered_happiness1 = happiness[order(happiness$log_GDP_per_capita),]
fit1 <- lm(life_ladder~.-country-year, data=ordered_happiness1)
e <- resid(fit1)
(rhoest = cor(e[-1],e[-129]))
```

```
## [1] 0.042776
```

```
V<-matrix(1,129,129)
V<-rhoest^(abs(row(V)-col(V)))
C<-chol(V)
tildey<-solve(t(C))%*% ordered_happiness1$life_ladder
X<-model.matrix(fit1)
tildeX<-solve(t(C))%*%X
fit2<-lm(tildey~tildeX-1)
e2<-resid(fit2)
(rhoest2<-cor(e2[-1],e2[-129]))
```

```
## [1] 0.004798293
```

```
ordered_happiness2 = happiness[order(happiness$social_support),]
fit1 <- lm(life_ladder~.-country-year, data=ordered_happiness2)
e <- resid(fit1)
(rhoest = cor(e[-1],e[-129]))
```

```
## [1] 0.04446802
```

```
V<-matrix(1,129,129)
V<-rhoest^(abs(row(V)-col(V)))
C<-chol(V)
tildey<-solve(t(C))%*% ordered_happiness2$life_ladder
X<-model.matrix(fit1)
tildeX<-solve(t(C))%*%X
fit2<-lm(tildey~tildeX-1)
```

```
e2<-resid(fit2)
(rhoest2<-cor(e2[-1],e2[-129]))
```

```
## [1] 0.008045709
```

```
ordered_happiness3 = happiness[order(happiness$healthy_life_expectancy_at_birth),]
fit1 <- lm(life_ladder~.-country-year, data=ordered_happiness3)
e <- resid(fit1)
(rhoest = cor(e[-1],e[-129]))
```

```
## [1] 0.04658441
```

```
V<-matrix(1,129,129)
V<-rhoest^(abs(row(V)-col(V)))
C<-chol(V)
tildey<-solve(t(C))%*% ordered_happiness3$life_ladder
X<-model.matrix(fit1)
tildeX<-solve(t(C))%*%X
fit2<-lm(tildey~tildeX-1)
e2<-resid(fit2)
(rhoest2<-cor(e2[-1],e2[-129]))
```

```
## [1] 0.0006724701
```

```
ordered_happiness4 = happiness[order(happiness$freedom_to_make_life_choices),]
fit1 <- lm(life_ladder~.-country-year, data=ordered_happiness4)
e <- resid(fit1)
(rhoest = cor(e[-1],e[-129]))
```

```
## [1] 0.07931481
```

```
V<-matrix(1,129,129)
V<-rhoest^(abs(row(V)-col(V)))
C<-chol(V)
tildey<-solve(t(C))%*% ordered_happiness4$life_ladder
X<-model.matrix(fit1)
tildeX<-solve(t(C))%*%X
fit2<-lm(tildey~tildeX-1)
e2<-resid(fit2)
(rhoest2<-cor(e2[-1],e2[-129]))
```

```
## [1] 0.02482933
```

```
ordered_happiness5 = happiness[order(happiness$generosity),]
fit1 <- lm(life_ladder~.-country-year, data=ordered_happiness5)
e <- resid(fit1)
(rhoest = cor(e[-1],e[-129]))
```

```
## [1] 0.0944983
```

```
V<-matrix(1,129,129)
V<-rhoest^(abs(row(V)-col(V)))
C<-chol(V)
tildey<-solve(t(C))%*% ordered_happiness5$life_ladder
X<-model.matrix(fit1)
tildeX<-solve(t(C))%*%X
fit2<-lm(tildey~tildeX-1)
e2<-resid(fit2)
```

```
(rhoest2<-cor(e2[-1],e2[-129]))
```

```
## [1] 0.01449754
```

```
ordered_happiness6 = happiness[order(happiness$perceptions_of_corruption),]
fit1 <- lm(life_ladder~.-country-year, data=ordered_happiness6)
e <- resid(fit1)
(rhoest = cor(e[-1],e[-129]))
```

```
## [1] 0.06253767
```

```
V<-matrix(1,129,129)
V<-rhoest^(abs(row(V)-col(V)))
C<-chol(V)
tildey<-solve(t(C))%*% ordered_happiness6$life_ladder
X<-model.matrix(fit1)
tildeX<-solve(t(C))%*%X
fit2<-lm(tildey~tildeX-1)
e2<-resid(fit2)
(rhoest2<-cor(e2[-1],e2[-129]))
```

```
## [1] -0.00571686
```

```
ordered_happiness7 = happiness[order(happiness$positive_affect),]
fit1 <- lm(life_ladder~.-country-year, data=ordered_happiness7)
e <- resid(fit1)
(rhoest = cor(e[-1],e[-129]))
```

```
## [1] 0.140463
```

```
V<-matrix(1,129,129)
V<-rhoest^(abs(row(V)-col(V)))
C<-chol(V)
tildey<-solve(t(C))%*% ordered_happiness7$life_ladder
X<-model.matrix(fit1)
tildeX<-solve(t(C))%*%X
fit2<-lm(tildey~tildeX-1)
e2<-resid(fit2)
(rhoest2<-cor(e2[-1],e2[-129]))
```

```
## [1] 0.01462196
```

```
ordered_happiness8 = happiness[order(happiness$negative_affect),]
fit1 <- lm(life_ladder~.-country-year, data=ordered_happiness8)
e <- resid(fit1)
(rhoest = cor(e[-1],e[-129]))
```

```
## [1] 0.01036088
```

```
V<-matrix(1,129,129)
V<-rhoest^(abs(row(V)-col(V)))
C<-chol(V)
tildey<-solve(t(C))%*% ordered_happiness8$life_ladder
X<-model.matrix(fit1)
tildeX<-solve(t(C))%*%X
fit2<-lm(tildey~tildeX-1)
e2<-resid(fit2)
(rhoest2<-cor(e2[-1],e2[-129]))
```

```
## [1] 0.001364813
```

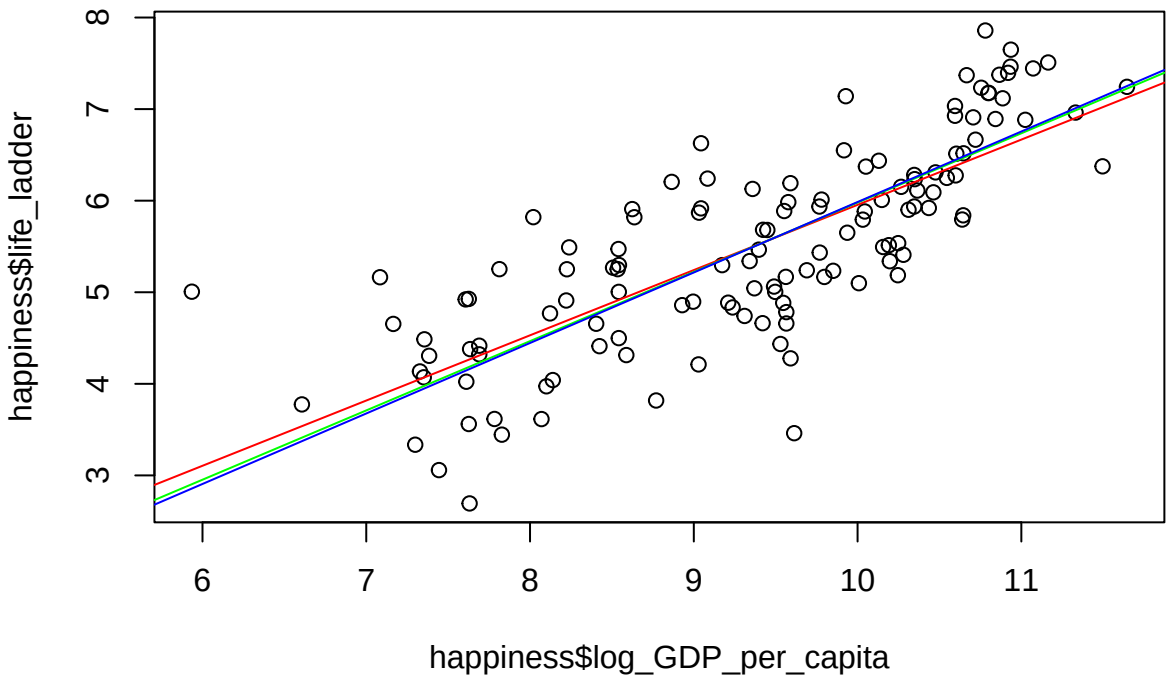
There is no serial correation in any of the predictors due to the correlation of the errors of full model and the generalized least squares model being close to 0 or no correlation. Even though in all cases the correlation got smaller, the correlation was not big enough in the first place to need correcting considering the largest correlaion was for positive affect at .14. Therefore a generalized least squares model is not necessary for this data.

These findings make sence since all of that data were taken from the same year and there is no other way to order the data.

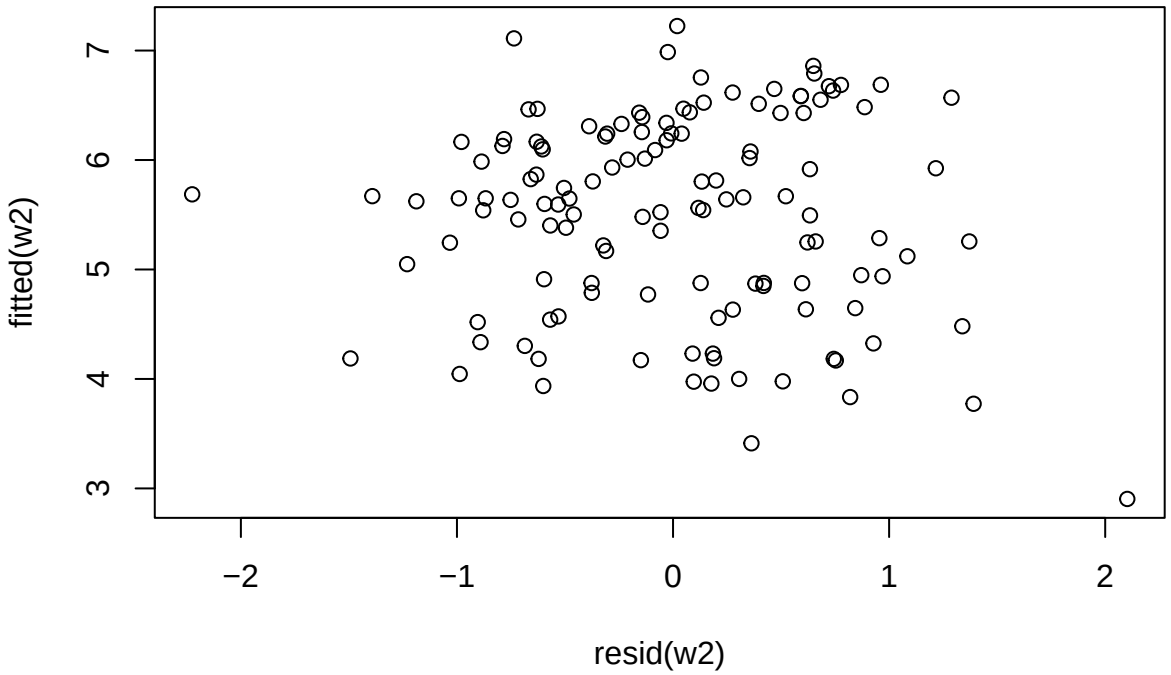
Question 6

6. Explore the possibility that the variance of the errors depend linearly on a specific predictor variable and apply weighted least-squares estimation along the lines of Example 3 in Topic 15.

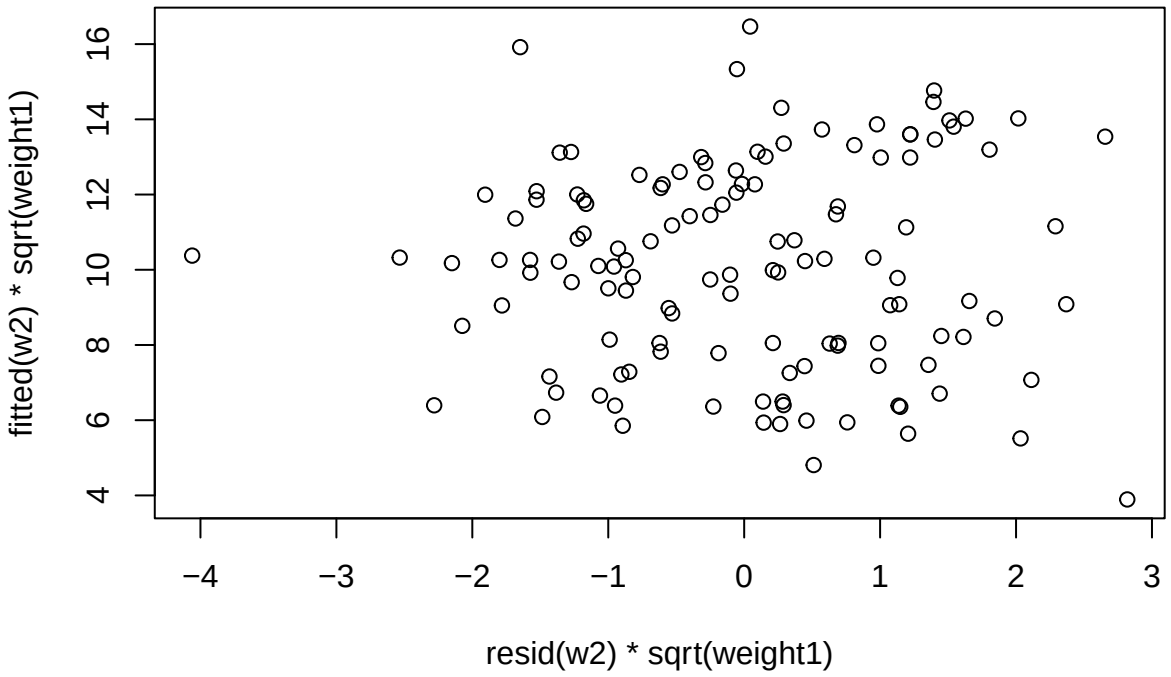
```
weight0 = rep(1,129)
w1 = lm(life_ladder~log_GDP_per_capita, data = happiness, weights = weight0)
sd1 = lm(abs(w1$resid)~log_GDP_per_capita, data = happiness)
weight1 = (1/sd1$fitted)^2
w2 = lm(life_ladder~log_GDP_per_capita, data = happiness, weights = weight1)
sd2 = lm(abs(w2$resid)~log_GDP_per_capita, data = happiness)
weight2 = (1/sd2$fitted)^2
w3 = lm(life_ladder~log_GDP_per_capita, data = happiness, weights = weight2)
plot(happiness$log_GDP_per_capita, happiness$life_ladder)
abline(coef(w1), col = "red")
abline(coef(w2), col = "green")
abline(coef(w3), col = "blue")
```



```r
plot(resid(w2), fitted(w2))
```

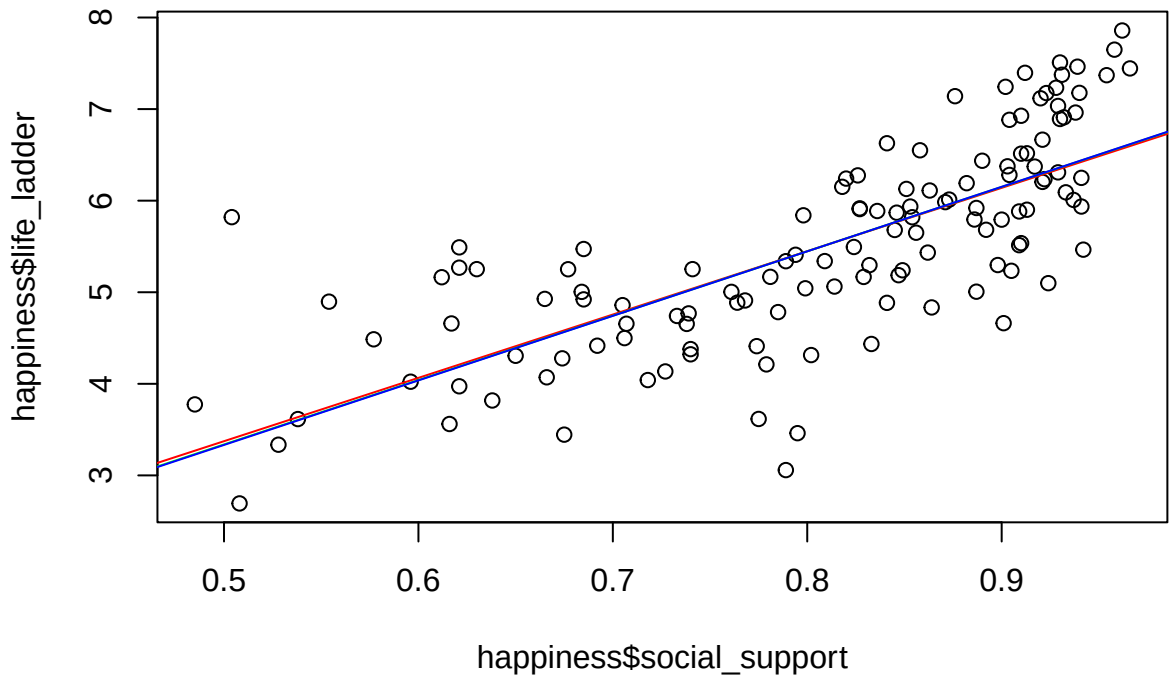



```r
plot(resid(w2)*sqrt(weight1), fitted(w2)*sqrt(weight1))
```




```r
weight0 = rep(1,129)
w1 = lm(life_ladder~social_support, data = happiness, weights = weight0)
sd1 = lm(abs(w1$resid)~social_support, data = happiness)
weight1 = (1/sd1$fitted)^2
w2 = lm(life_ladder~social_support, data = happiness, weights = weight1)
sd2 = lm(abs(w2$resid)~social_support, data = happiness)
weight2 = (1/sd2$fitted)^2
```

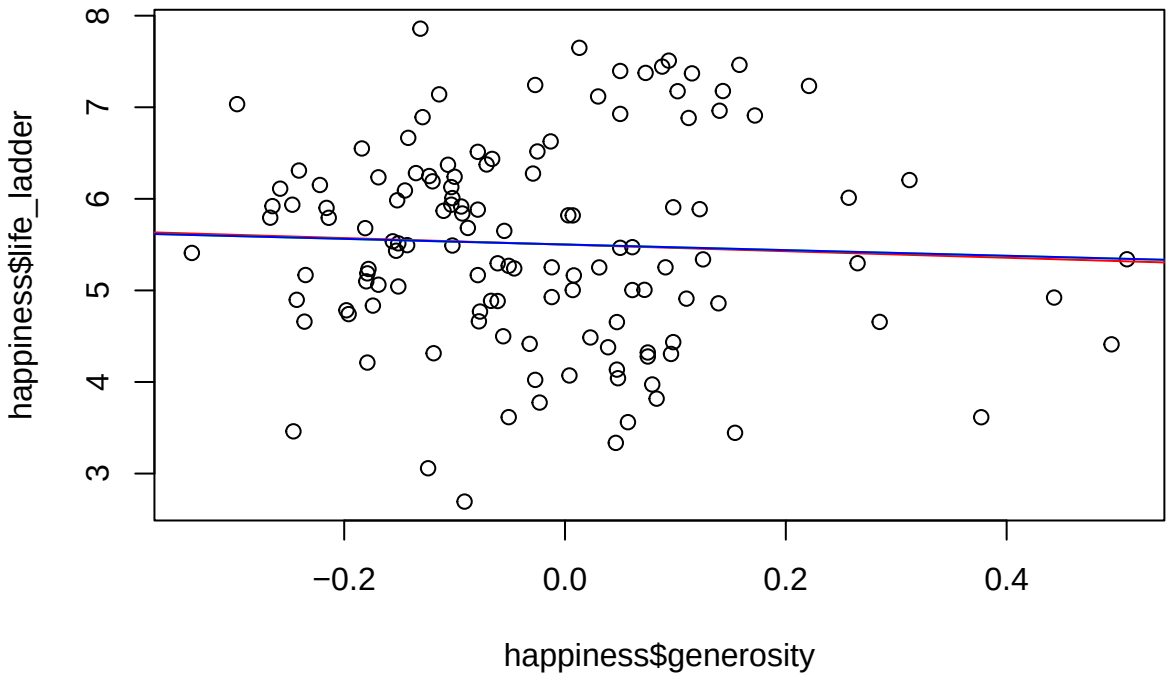
```r
w3 = lm(life_ladder~social_support, data = happiness, weights = weight2)
plot(happiness$social_support, happiness$life_ladder)
abline(coef(w1), col = "red")
abline(coef(w2), col = "green")
abline(coef(w3), col = "blue")
```



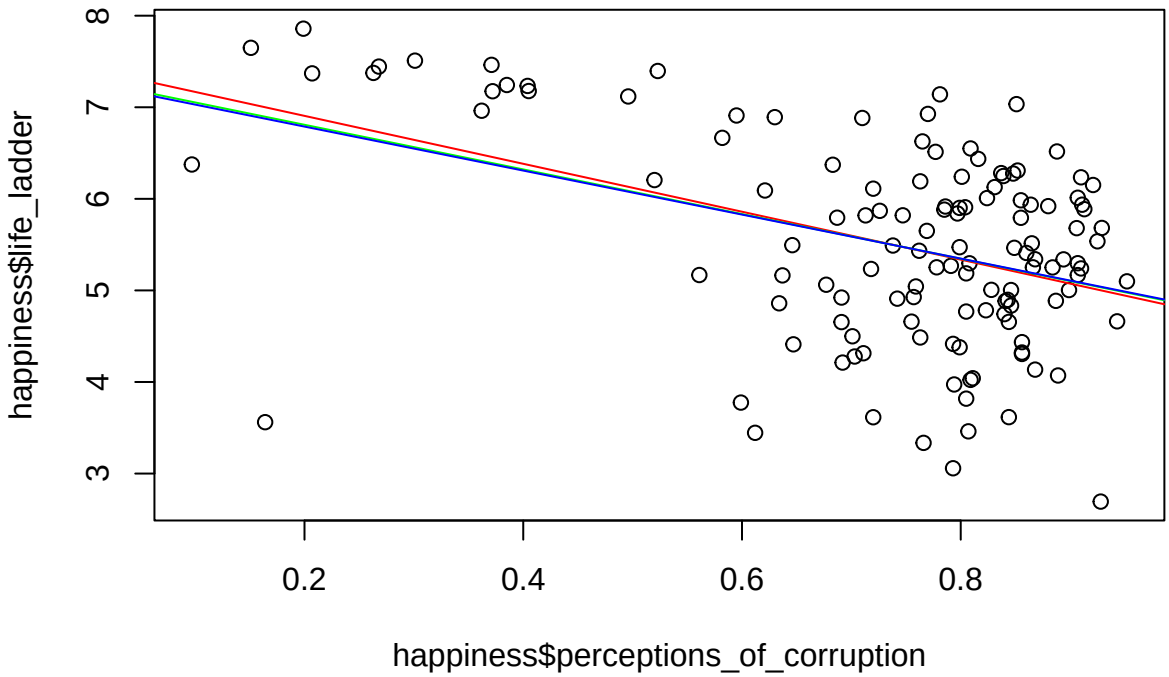

```r
weight0 = rep(1,129)
w1 = lm(life_ladder~healthy_life_expectancy_at_birth, data = happiness, weights = weight0)
sd1 = lm(abs(w1$resid)~healthy_life_expectancy_at_birth, data = happiness)
weight1 = (1/sd1$fitted)^2
w2 = lm(life_ladder~healthy_life_expectancy_at_birth, data = happiness, weights = weight1)
sd2 = lm(abs(w2$resid)~healthy_life_expectancy_at_birth, data = happiness)
weight2 = (1/sd2$fitted)^2
w3 = lm(life_ladder~healthy_life_expectancy_at_birth, data = happiness, weights = weight2)
plot(happiness$healthy_life_expectancy_at_birth, happiness$life_ladder)
abline(coef(w1), col = "red")
abline(coef(w2), col = "green")
abline(coef(w3), col = "blue")
```


```
weight0 = rep(1,129)
w1 = lm(life_ladder~freedom_to_make_life_choices, data = happiness, weights = weight0)
sd1 = lm(abs(w1$resid)~freedom_to_make_life_choices, data = happiness)
weight1 = (1/sd1$fitted)^2
w2 = lm(life_ladder~freedom_to_make_life_choices, data = happiness, weights = weight1)
sd2 = lm(abs(w2$resid)~freedom_to_make_life_choices, data = happiness)
weight2 = (1/sd2$fitted)^2
w3 = lm(life_ladder~freedom_to_make_life_choices, data = happiness, weights = weight2)
plot(happiness$freedom_to_make_life_choices, happiness$life_ladder)
abline(coef(w1), col = "red")
abline(coef(w2), col = "green")
abline(coef(w3), col = "blue")
```

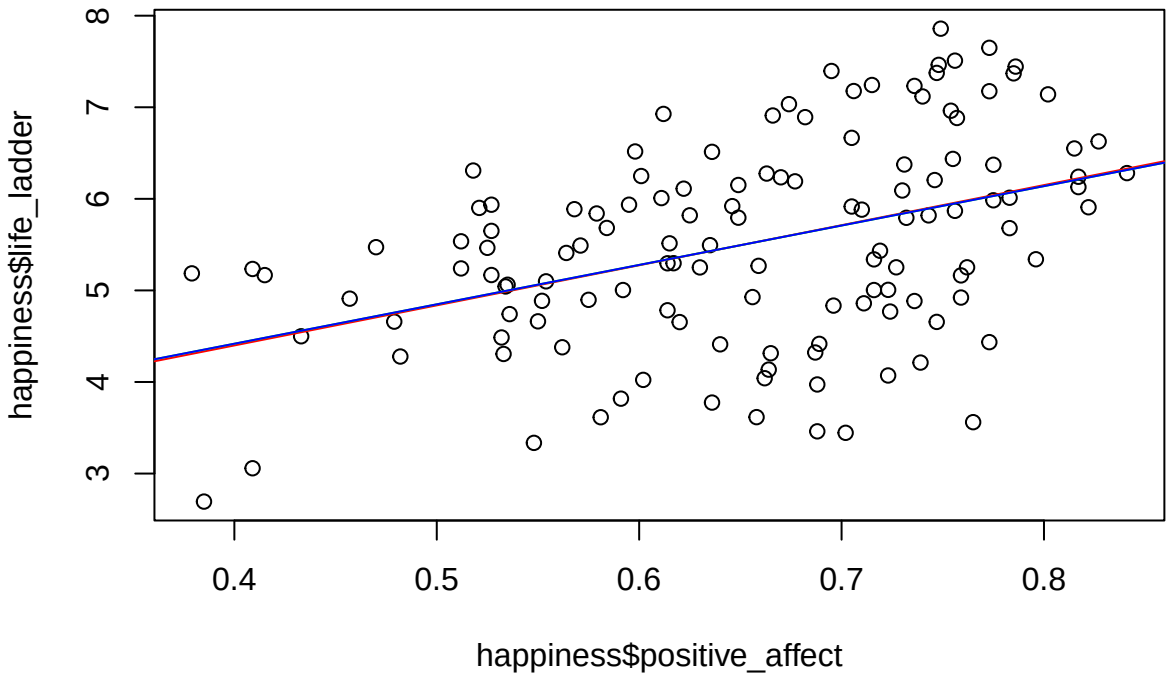

```
weight0 = rep(1,129)
w1 = lm(life_ladder~generosity, data = happiness, weights = weight0)
sd1 = lm(abs(w1$resid)~generosity, data = happiness)
weight1 = (1/sd1$fitted)^2
w2 = lm(life_ladder~generosity, data = happiness, weights = weight1)
sd2 = lm(abs(w2$resid)~generosity, data = happiness)
weight2 = (1/sd2$fitted)^2
w3 = lm(life_ladder~generosity, data = happiness, weights = weight2)
plot(happiness$generosity, happiness$life_ladder)
abline(coef(w1), col = "red")
abline(coef(w2), col = "green")
abline(coef(w3), col = "blue")
```



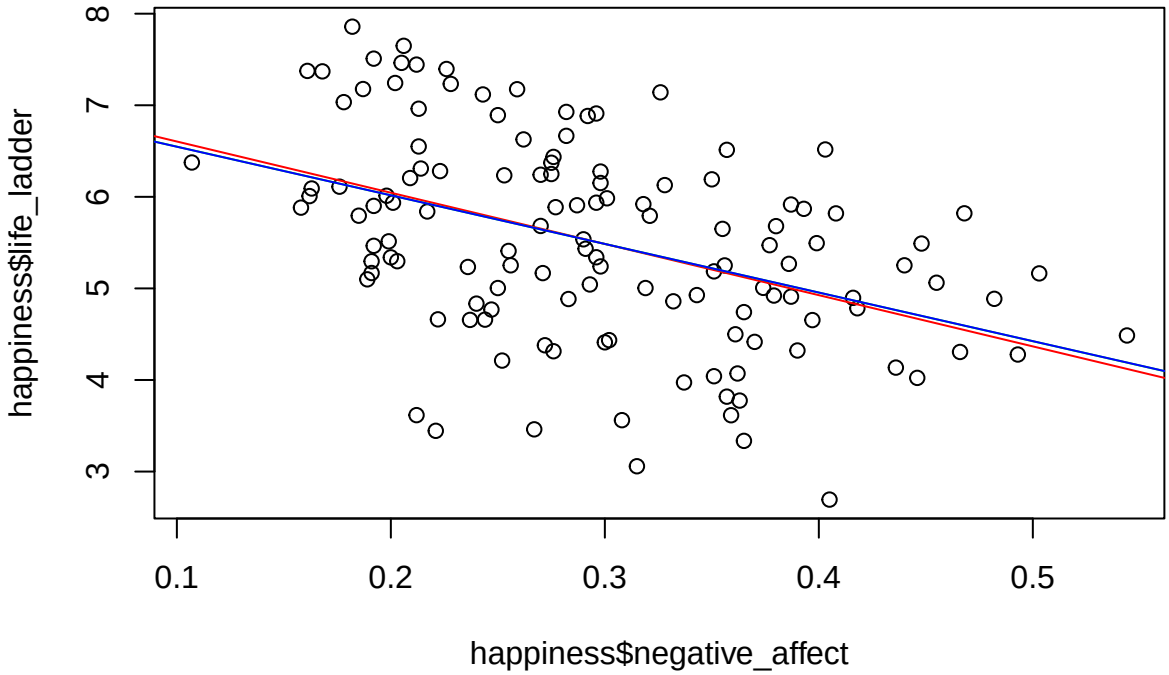

```
weight0 = rep(1,129)
w1 = lm(life_ladder~perceptions_of_corruption, data = happiness, weights = weight0)
sd1 = lm(abs(w1$resid)~perceptions_of_corruption, data = happiness)
weight1 = (1/sd1$fitted)^2
w2 = lm(life_ladder~perceptions_of_corruption, data = happiness, weights = weight1)
sd2 = lm(abs(w2$resid)~perceptions_of_corruption, data = happiness)
weight2 = (1/sd2$fitted)^2
w3 = lm(life_ladder~perceptions_of_corruption, data = happiness, weights = weight2)
plot(happiness$perceptions_of_corruption, happiness$life_ladder)
abline(coef(w1), col = "red")
abline(coef(w2), col = "green")
abline(coef(w3), col = "blue")
```



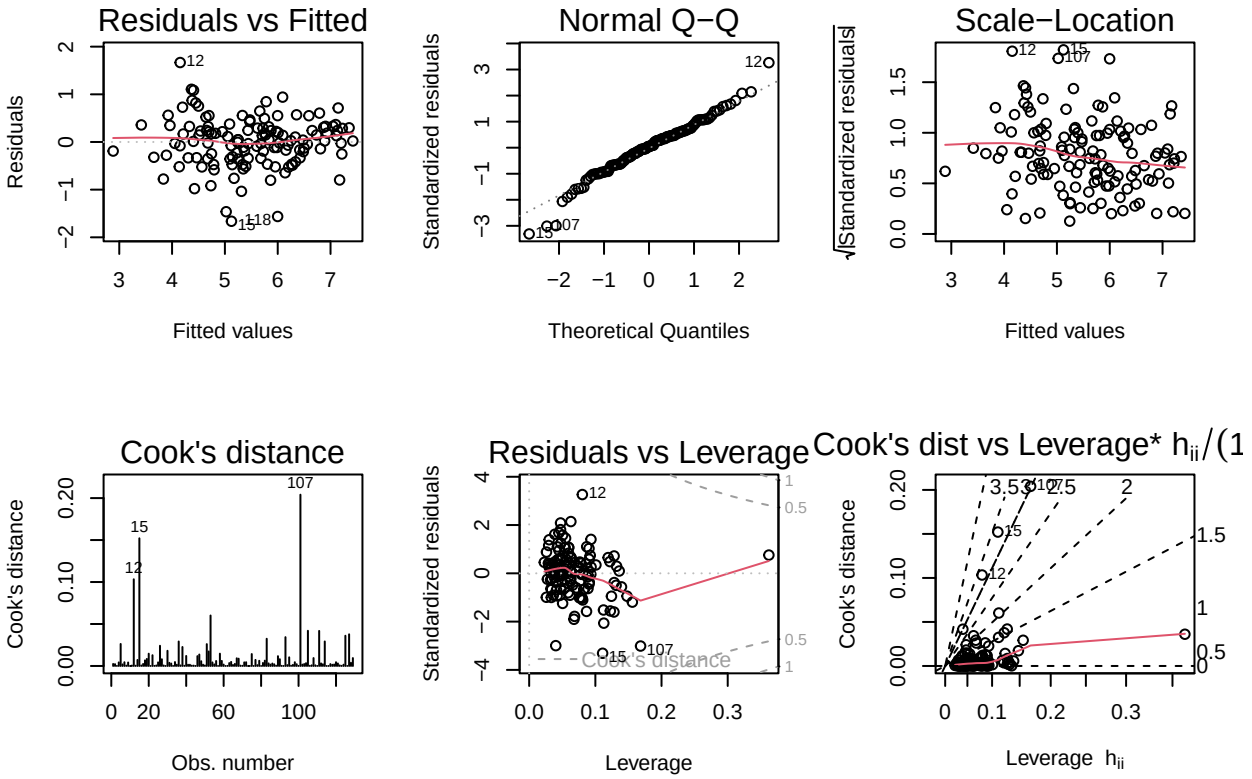

```
weight0 = rep(1,129)
w1 = lm(life_ladder~positive_affect, data = happiness, weights = weight0)
sd1 = lm(abs(w1$resid)~positive_affect, data = happiness)
weight1 = (1/sd1$fitted)^2
w2 = lm(life_ladder~positive_affect, data = happiness, weights = weight1)
sd2 = lm(abs(w2$resid)~positive_affect, data = happiness)
weight2 = (1/sd2$fitted)^2
w3 = lm(life_ladder~positive_affect, data = happiness, weights = weight2)
plot(happiness$positive_affect, happiness$life_ladder)
abline(coef(w1), col = "red")
abline(coef(w2), col = "green")
abline(coef(w3), col = "blue")
```




```
weight0 = rep(1,129)
w1 = lm(life_ladder~negative_affect, data = happiness, weights = weight0)
sd1 = lm(abs(w1$resid)~negative_affect, data = happiness)
weight1 = (1/sd1$fitted)^2
w2 = lm(life_ladder~negative_affect, data = happiness, weights = weight1)
sd2 = lm(abs(w2$resid)~negative_affect, data = happiness)
weight2 = (1/sd2$fitted)^2
w3 = lm(life_ladder~negative_affect, data = happiness, weights = weight2)
plot(happiness$negative_affect, happiness$life_ladder)
abline(coef(w1), col = "red")
abline(coef(w2), col = "green")
abline(coef(w3), col = "blue")
```




```
par(mfrow=c(2,3))
plot(full_model, which=1:6)
```



It is unlikely that the varrience of errors depends linearly on any specific predictor due to most models changing minimally when using iteratively reweighted least squares. The one that changed the most was log GDP per capita. For that predictor, when the residuals vs fitted values were plotted for both weighted and unweighted there was not a significant change in the level of homoskedasitcity.

Question 7

R Code and Comments for Question 7

First, we will read and clean the data to allow for analysis and interpretation.

```
happiness = read.csv("/Users/aidanashrafi/Downloads/Happiness_2018.csv")

model<- lm(life_ladder~ . - country - year,data=happiness)
summary(model)
```

```
##
## Call:
## lm(formula = life_ladder ~ . - country - year, data = happiness)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.66321 -0.33003  0.02126  0.29709  1.66671
##
## Coefficients:
##                                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)                       -4.62111    0.87144  -5.303 5.28e-07 ***
## log_GDP_per_capita                 0.35544    0.08478   4.192 5.31e-05 ***
## social_support                     2.83854    0.72502   3.915 0.000150 ***
## healthy_life_expectancy_at_birth   0.03672    0.01448   2.535 0.012536 *
## freedom_to_make_life_choices       0.78692    0.55499   1.418 0.158812
## generosity                         0.13951    0.34495   0.404 0.686613
## perceptions_of_corruption         -0.76970    0.30685  -2.508 0.013462 *
## positive_affect                    2.11173    0.56563   3.733 0.000290 ***
## negative_affect                    2.47878    0.72805   3.405 0.000901 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.5326 on 120 degrees of freedom
##   (12 observations deleted due to missingness)
## Multiple R-squared:  0.7829, Adjusted R-squared:  0.7684
## F-statistic: 54.09 on 8 and 120 DF,  p-value: < 2.2e-16
```

```
matrix <- model.matrix(model)
rank_matrix<- t(matrix)%*% matrix
rank_model_matrix <- qr(rank_matrix)$rank
num_cols_X <- ncol(rank_matrix)
rank_model_matrix == num_cols_X
```

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

As discussed in class, R adjusts and imposes extra conditions on Beta Hat to make it unique as R has to make sure there is unique OLSE to be able to provide an output, problems with multicollinearity will not be found using this method.

Conducting a Test of Significance

```
reduced_model <- lm(life_ladder ~ 1, data = happiness)
anova(reduced_model)
```

```
## Analysis of Variance Table
##
## Response: life_ladder
```

```
##            Df Sum Sq Mean Sq F value Pr(>F)
## Residuals 140 165.51  1.1823
```

```
anova(model)
```

```
## Analysis of Variance Table
##
## Response: life_ladder
##                                   Df Sum Sq Mean Sq  F value    Pr(>F)
## log_GDP_per_capita                 1 94.978  94.978 334.7743 < 2.2e-16 ***
## social_support                     1  7.890   7.890  27.8092 6.009e-07 ***
## healthy_life_expectancy_at_birth   1  2.715   2.715   9.5712 0.0024593 **
## freedom_to_make_life_choices       1  7.687   7.687  27.0937 8.119e-07 ***
## generosity                         1  0.560   0.560   1.9727 0.1627464
## perceptions_of_corruption          1  1.846   1.846   6.5061 0.0120085 *
## positive_affect                    1  3.797   3.797  13.3847 0.0003782 ***
## negative_affect                    1  3.289   3.289  11.5919 0.0009013 ***
## Residuals                        120 34.045   0.284
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

This output shows us that at least one of the predictors is significant therefore contradicting that each individual predictor is not significant.

Now we wil conduct sensitivity analysis to see if small deviations in the dependent variable will drastically change beta coefficient significance, beta coefficient sign and strength, and overall goodness-of-fit.

```
min<- min(happiness$life_ladder)
max<- max(happiness$life_ladder)
mean<- mean(happiness$life_ladder)
sd<- sd(happiness$life_ladder)
mean - 2*sd >= min
```

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

```
max - 2*sd >= mean
```

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

The fact that the mean - 2 SD is still greater than the min gives insight into what the scalar in front of rnorm should be. So SD = 1.08, but mean - 2 SD >= min, so our scalar is ok to deviate the response within the range +- [0,0.75] and be considered a small change in the response.

```
sensitivity_model_1<- lm(life_ladder + 0.75*rnorm(141)~ . - country - year, data=happiness)
sensitivity_model_2<- lm(life_ladder + 0.5*rnorm(141)~ . - country - year, data=happiness)
summary(sensitivity_model_1)
```

```
##
## Call:
## lm(formula = life_ladder + 0.75 * rnorm(141) ~ . - country -
##     year, data = happiness)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.91354 -0.49819 -0.01422  0.61307  2.10867
##
## Coefficients:
##                                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)                     -4.7175567  1.3869063  -3.401 0.000911 ***
```

```
## log_GDP_per_capita                0.4371581  0.1349354   3.240 0.001548 **
## social_support                    1.1657596  1.1538729   1.010 0.314384
## healthy_life_expectancy_at_birth  0.0418757  0.0230522   1.817 0.071780 .
## freedom_to_make_life_choices      0.2655112  0.8832665   0.301 0.764239
## generosity                        0.0006655  0.5489960   0.001 0.999035
## perceptions_of_corruption        -0.3950922  0.4883486  -0.809 0.420095
## positive_affect                   3.3727155  0.9002062   3.747 0.000277 ***
## negative_affect                   1.2544023  1.1586930   1.083 0.281156
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.8477 on 120 degrees of freedom
##   (12 observations deleted due to missingness)
## Multiple R-squared:  0.5921, Adjusted R-squared:  0.5649
## F-statistic: 21.78 on 8 and 120 DF,  p-value: < 2.2e-16
```

```
summary(sensitivity_model_2)
```

```
##
## Call:
## lm(formula = life_ladder + 0.5 * rnorm(141) ~ . - country - year,
##     data = happiness)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2.00397 -0.46536 -0.03607  0.45835  1.55543
##
## Coefficients:
##                                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)                      -4.990960   1.105148  -4.516 1.48e-05 ***
## log_GDP_per_capita                0.452201   0.107522   4.206 5.04e-05 ***
## social_support                    4.183091   0.919457   4.550 1.30e-05 ***
## healthy_life_expectancy_at_birth  0.009923   0.018369   0.540 0.590061
## freedom_to_make_life_choices      0.465929   0.703826   0.662 0.509244
## generosity                        0.228433   0.437464   0.522 0.602510
## perceptions_of_corruption        -0.809453   0.389138  -2.080 0.039643 *
## positive_affect                   2.183388   0.717324   3.044 0.002871 **
## negative_affect                   3.361013   0.923298   3.640 0.000403 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.6755 on 120 degrees of freedom
##   (12 observations deleted due to missingness)
## Multiple R-squared:   0.71,  Adjusted R-squared:  0.6906
## F-statistic: 36.72 on 8 and 120 DF,  p-value: < 2.2e-16
```

```
summary(model)
```

```
##
## Call:
## lm(formula = life_ladder ~ . - country - year, data = happiness)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.66321 -0.33003  0.02126  0.29709  1.66671
```

```
##
## Coefficients:
##                                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)                      -4.62111    0.87144  -5.303 5.28e-07 ***
## log_GDP_per_capita                0.35544    0.08478   4.192 5.31e-05 ***
## social_support                    2.83854    0.72502   3.915 0.000150 ***
## healthy_life_expectancy_at_birth  0.03672    0.01448   2.535 0.012536 *
## freedom_to_make_life_choices      0.78692    0.55499   1.418 0.158812
## generosity                        0.13951    0.34495   0.404 0.686613
## perceptions_of_corruption        -0.76970    0.30685  -2.508 0.013462 *
## positive_affect                   2.11173    0.56563   3.733 0.000290 ***
## negative_affect                   2.47878    0.72805   3.405 0.000901 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.5326 on 120 degrees of freedom
##   (12 observations deleted due to missingness)
## Multiple R-squared:  0.7829, Adjusted R-squared:  0.7684
## F-statistic: 54.09 on 8 and 120 DF,  p-value: < 2.2e-16
```

Comparing 0.5 to 0.75 and the original model gives very important insights and allows us to finally say there is potential for multicollinearity in the dataset. As evidenced in the outputs, small changes in the response affect beta significance, goodness-of-fit, and strengths of beta coefficients. Therefore, we have discovered potential multicollinearity, and now must proceed.

Correlation Matrix Creation and Insights

Now that we have established potential multicollinearity, let's look at the correlation between variables.

```r
variables_of_interest <- c("life_ladder","log_GDP_per_capita", "social_support", "generosity","healthy_l
happiness_subset <- happiness[, variables_of_interest]
# Calculate the correlation matrix, excluding missing values
cor_matrix <- round(cor(happiness_subset, use = "complete.obs"), 3)
print(cor_matrix)
```

```
##                                  life_ladder log_GDP_per_capita social_support
## life_ladder                            1.000              0.778          0.742
## log_GDP_per_capita                     0.778              1.000          0.768
## social_support                         0.742              0.768          1.000
## generosity                            -0.051             -0.260         -0.167
## healthy_life_expectancy_at_birth       0.754              0.840          0.733
## freedom_to_make_life_choices           0.523              0.355          0.396
## perceptions_of_corruption             -0.441             -0.326         -0.222
## positive_affect                        0.423              0.149          0.270
## negative_affect                       -0.448             -0.589         -0.645
##                                  generosity healthy_life_expectancy_at_birth
## life_ladder                          -0.051                            0.754
## log_GDP_per_capita                   -0.260                            0.840
## social_support                       -0.167                            0.733
## generosity                            1.000                           -0.177
## healthy_life_expectancy_at_birth     -0.177                            1.000
## freedom_to_make_life_choices          0.239                            0.331
## perceptions_of_corruption            -0.238                           -0.308
## positive_affect                       0.255                            0.138
## negative_affect                       0.075                           -0.489
##                                  freedom_to_make_life_choices
```

```
## life_ladder                                          0.523
## log_GDP_per_capita                                   0.355
## social_support                                       0.396
## generosity                                           0.239
## healthy_life_expectancy_at_birth                     0.331
## freedom_to_make_life_choices                         1.000
## perceptions_of_corruption                           -0.456
## positive_affect                                      0.592
## negative_affect                                     -0.367
##                                  perceptions_of_corruption positive_affect
## life_ladder                                         -0.441           0.423
## log_GDP_per_capita                                  -0.326           0.149
## social_support                                      -0.222           0.270
## generosity                                          -0.238           0.255
## healthy_life_expectancy_at_birth                    -0.308           0.138
## freedom_to_make_life_choices                        -0.456           0.592
## perceptions_of_corruption                            1.000          -0.343
## positive_affect                                     -0.343           1.000
## negative_affect                                      0.304          -0.250
##                                  negative_affect
## life_ladder                               -0.448
## log_GDP_per_capita                        -0.589
## social_support                            -0.645
## generosity                                 0.075
## healthy_life_expectancy_at_birth          -0.489
## freedom_to_make_life_choices              -0.367
## perceptions_of_corruption                  0.304
## positive_affect                           -0.250
## negative_affect                            1.000
```

```r
# Initialize a vector to store the maximum correlation for each variable
max_cor_for_var <- rep(-Inf, length(variables_of_interest))
# Initialize a vector to store the corresponding variables for max correlation
max_var_for_cor <- rep("", length(variables_of_interest))
# Loop through each variable
for (j in seq_along(variables_of_interest)) {
  # Loop through each element in the correlation matrix for the current variable
  for (i in seq_along(cor_matrix[, j])) {
    # Check if the element is not 1 (assuming you want to exclude self-correlations)
    if (cor_matrix[[i, j]] != 1 && !is.na(cor_matrix[[i, j]])) {
      # Update max_cor_for_var, max_var_for_cor, and max_indices_for_var if the current correlation is
      current_corr_value <- cor_matrix[[i, j]]
      abs_corr_value <- abs(current_corr_value)
      if (abs_corr_value > max_cor_for_var[j]) {
        max_cor_for_var[j] <- abs_corr_value
        max_var_for_cor[j] <- variables_of_interest[i]
        if (current_corr_value < 0) {
          # If the correlation is negative, include the sign in the output
          sign_indicator <- "-"
        } else {
          sign_indicator <- ""
        }
      }
    }
```

```
  }
  # Print the maximum correlation, the corresponding variable on a new line
  cat(sprintf("Variable: %s, Max Correlation: %s%f, Corresponding Variable: %s\n",
              variables_of_interest[j], sign_indicator, max_cor_for_var[j], max_var_for_cor[j]))
}
```

```
## Variable: life_ladder, Max Correlation: 0.778000, Corresponding Variable: log_GDP_per_capita
## Variable: log_GDP_per_capita, Max Correlation: 0.840000, Corresponding Variable: healthy_life_expecta
## Variable: social_support, Max Correlation: 0.768000, Corresponding Variable: log_GDP_per_capita
## Variable: generosity, Max Correlation: -0.260000, Corresponding Variable: log_GDP_per_capita
## Variable: healthy_life_expectancy_at_birth, Max Correlation: 0.840000, Corresponding Variable: log_GD
## Variable: freedom_to_make_life_choices, Max Correlation: 0.592000, Corresponding Variable: positive_a
## Variable: perceptions_of_corruption, Max Correlation: -0.456000, Corresponding Variable: freedom_to_m
## Variable: positive_affect, Max Correlation: 0.592000, Corresponding Variable: freedom_to_make_life_ch
## Variable: negative_affect, Max Correlation: -0.645000, Corresponding Variable: social_support
```

This produces an output with each variable, its max correlation, and the corresponding variable that produces the max correlation. Based on this output, it is evident that log_GDP_per_capita, social_support, and healthy_life_expectancy have a strong positive correlation (> 0.7). Based on this output, it is evident that negative effect and social support have a relatively strong negative correlation (< -0.5). Based on this output, it is evident that positive affect and freedom_to_make_life_choice have a relatively strong correlation (> 0.5). Furthermore, Based on this output, it is evident that life_ladder is strongly positively correlated with log_GDP_per_capita, social_support, and healthy_life_expectancy (> 0.7).

Now, we have a better idea of what is potentially causing our multicollinearity.

We will now conduct Condition Number Test & Variance Inflation Factor to dive deeper into the roots of the problem.

```
x<- model.matrix(model)[,-1];
e<- eigen(t(x)%*%x)
e$val
```

```
## [1] 5.423607e+05 5.569248e+01 6.543028e+00 3.970106e+00 1.748274e+00
## [6] 1.082188e+00 6.218488e-01 4.127828e-01
```

```
condition_numbers<-sqrt(e$val[1]/e$val)
#If the condition number is >30, we have a collinearity problem
condition_numbers
```

```
## [1]    1.0000   98.6838  287.9086  369.6094  556.9797  707.9340  933.9028
## [8] 1146.2604
```

```
#Initialize a numeric variable
condition_numbers_showing_multicollinearity <- numeric()
#Check to see if condition_numbers > 30
for (number in condition_numbers) {
  if (number > 30) {
    condition_numbers_showing_multicollinearity <- c(condition_numbers_showing_multicollinearity, numbe
  }
}
#If there is no output for this, then the model does not have a collinearity problem
condition_numbers_showing_multicollinearity
```

```
## [1]   98.6838  287.9086  369.6094  556.9797  707.9340  933.9028 1146.2604
```

There are 7 output values for the condition number test > 30 out of the 8 output values. Therefore, as several condition numbers are large, problems are being caused by more than just one linear combination. This

agrees with some of our insights from the correlation matrix.

Now we will look at the VIFs

```
#Compute the first variable VIF to make sure it agrees with vif()
a<-summary(lm(x[,1]~x[,-1]))$r.squared
1/(1-a)
```

```
## [1] 4.742125
```

```
#Will check the first variable VIF computation
#If VIF > 10, this indicates high collinearity in the model
#Computes all variables VIF in the model
library(car)
```

```
## Loading required package: carData
```

```
vif_model<- vif(model)
#Initialize a numeric variable
vif_showing_multicollinearity <- numeric()
vif_under_threshold <- numeric()
#Check to see if vif > 10
for (number in vif_model) {
  if (number > 10) {
    vif_showing_multicollinearity <- c(vif_showing_multicollinearity, number)
    }else {
      vif_under_threshold <- c(vif_under_threshold,number)
      vif_showing_multicollinearity <- c(vif_showing_multicollinearity,"No")
    }
  }
vif_showing_multicollinearity
```

```
## [1] "No" "No" "No" "No" "No" "No" "No" "No"
```

```
vif_under_threshold
```

```
## [1] 4.742125 3.340700 3.752970 1.959149 1.337855 1.480502 1.648568 1.883040
```

These outputs suggest that none of the VIFs > 10, so using this test, multicollinearity is not an issue. This leads us to question if the multicollinearity is associated with individual variables as the VIF assesses this or if the multicollinearity is associated with a combination of variables as condition numbers assess. Based on the outputs, we can conclude that the multicollinearity is associated with a combination of variables.

Based on the correlation matrix, and the insights from the VIF and Condition Number Test, Amputation may not be as effective as intended as individual variables are not causing our problem. Further reasoning and evidence to support this are in the conclusion of Question 8.

So, we have assessed multicollinearity, the model clearly exhibits multicollinearity through conclusive results derived from sensitivity analysis, the model's correlation matrix, and the Condition Number Test, but Amputation of individual variables is not a valid solution to address multicollinearity as examples will show at the end of Question 8.

Question 8

So in Question 7, we concluded that there is evidence of multicollinearity that is affecting the predictive power of the model. Now we will apply an exhaustive model selection, using the AIC, BIC, and Mallow's Cp, Adjusted R2, and R-squared.

Adjusted R2

```r
#Initialize a model
model<- lm(life_ladder~ . - country - year,data=happiness)
summary(model)
```

```
##
## Call:
## lm(formula = life_ladder ~ . - country - year, data = happiness)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.66321 -0.33003  0.02126  0.29709  1.66671
##
## Coefficients:
##                                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)                       -4.62111    0.87144  -5.303 5.28e-07 ***
## log_GDP_per_capita                 0.35544    0.08478   4.192 5.31e-05 ***
## social_support                     2.83854    0.72502   3.915 0.000150 ***
## healthy_life_expectancy_at_birth   0.03672    0.01448   2.535 0.012536 *
## freedom_to_make_life_choices       0.78692    0.55499   1.418 0.158812
## generosity                         0.13951    0.34495   0.404 0.686613
## perceptions_of_corruption         -0.76970    0.30685  -2.508 0.013462 *
## positive_affect                    2.11173    0.56563   3.733 0.000290 ***
## negative_affect                    2.47878    0.72805   3.405 0.000901 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.5326 on 120 degrees of freedom
##   (12 observations deleted due to missingness)
## Multiple R-squared:  0.7829, Adjusted R-squared:  0.7684
## F-statistic: 54.09 on 8 and 120 DF,  p-value: < 2.2e-16
```

```r
# Load the leaps package
library(leaps)
#In order for leaps to run, we must omit values that R previously removed itself in calculation (12 obs
happiness<-na.omit(happiness)
happiness_subset <- happiness[, c("life_ladder","log_GDP_per_capita", "social_support", "generosity", "
# Create the predictor matrix (excluding the response variable)
x <- happiness_subset[,c("log_GDP_per_capita", "social_support", "generosity", "healthy_life_expectancy_
# Response variable
y <- happiness_subset$life_ladder
# Perform subset selection using leaps
leaps_model<- leaps(x, y, method = "adjr2")
best_model_index <- which.max(leaps_model$adjr2)
best_model_adjr2 <- leaps_model$adjr2[best_model_index]
best_model_variables <- colnames(x)[leaps_model$which[best_model_index, ]]
best_model_index
```

```
## [1] 59
```

```r
best_model_adjr2
```

```
## [1] 0.7700121
```

```r
best_model_variables
```

```
## [1] "log_GDP_per_capita"               "social_support"
```

```
## [3] "healthy_life_expectancy_at_birth" "freedom_to_make_life_choices"
## [5] "perceptions_of_corruption"        "positive_affect"
## [7] "negative_affect"
```

```
best_model_1 <- lm(life_ladder ~ . - country - year - generosity, data=happiness)
summary(best_model_1)
```

```
##
## Call:
## lm(formula = life_ladder ~ . - country - year - generosity, data = happiness)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.68864 -0.32471  0.03681  0.29617  1.66386
##
## Coefficients:
##                                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)                      -4.58279    0.86328  -5.309 5.08e-07 ***
## log_GDP_per_capita                0.34651    0.08158   4.248 4.27e-05 ***
## social_support                    2.83170    0.72231   3.920 0.000147 ***
## healthy_life_expectancy_at_birth  0.03705    0.01441   2.571 0.011343 *
## freedom_to_make_life_choices      0.82753    0.54394   1.521 0.130776
## perceptions_of_corruption        -0.79740    0.29807  -2.675 0.008502 **
## positive_affect                   2.13475    0.56081   3.807 0.000223 ***
## negative_affect                   2.47608    0.72550   3.413 0.000875 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.5308 on 121 degrees of freedom
## Multiple R-squared:  0.7826, Adjusted R-squared:   0.77
## F-statistic: 62.22 on 7 and 121 DF,  p-value: < 2.2e-16
```

This output shows that the best model based on adjusted R squared amputates generosity out of the model, and has 7 remaining predictors.

RSquared

```
leaps_model_2<- leaps(x, y, method = "r2")
best_model_2_index <- which.max(leaps_model_2$r2)
best_model_2_r2 <- leaps_model_2$r2[best_model_2_index]
best_model_2_variables <- colnames(x)[leaps_model_2$which[best_model_2_index, ]]
best_model_2_index
```

```
## [1] 67
```

```
best_model_2_r2
```

```
## [1] 0.7828855
```

```
best_model_2_variables
```

```
## [1] "log_GDP_per_capita"              "social_support"
## [3] "generosity"                      "healthy_life_expectancy_at_birth"
## [5] "freedom_to_make_life_choices"    "perceptions_of_corruption"
## [7] "positive_affect"                 "negative_affect"
```

```
best_model_2 <- lm(life_ladder ~ . - country - year, data=happiness)
summary(best_model_2)
```

```
##
## Call:
## lm(formula = life_ladder ~ . - country - year, data = happiness)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.66321 -0.33003  0.02126  0.29709  1.66671
##
## Coefficients:
##                                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)                       -4.62111    0.87144  -5.303 5.28e-07 ***
## log_GDP_per_capita                 0.35544    0.08478   4.192 5.31e-05 ***
## social_support                     2.83854    0.72502   3.915 0.000150 ***
## healthy_life_expectancy_at_birth   0.03672    0.01448   2.535 0.012536 *
## freedom_to_make_life_choices       0.78692    0.55499   1.418 0.158812
## generosity                         0.13951    0.34495   0.404 0.686613
## perceptions_of_corruption         -0.76970    0.30685  -2.508 0.013462 *
## positive_affect                    2.11173    0.56563   3.733 0.000290 ***
## negative_affect                    2.47878    0.72805   3.405 0.000901 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.5326 on 120 degrees of freedom
## Multiple R-squared:  0.7829, Adjusted R-squared:  0.7684
## F-statistic: 54.09 on 8 and 120 DF,  p-value: < 2.2e-16
```

This output shows that the best model based on R squared, is actually the full model, with all 8 predictors.

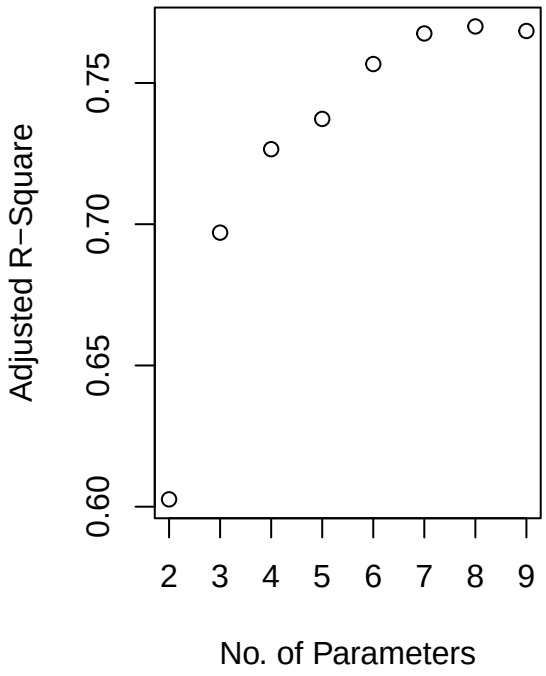
Mallow's Cp

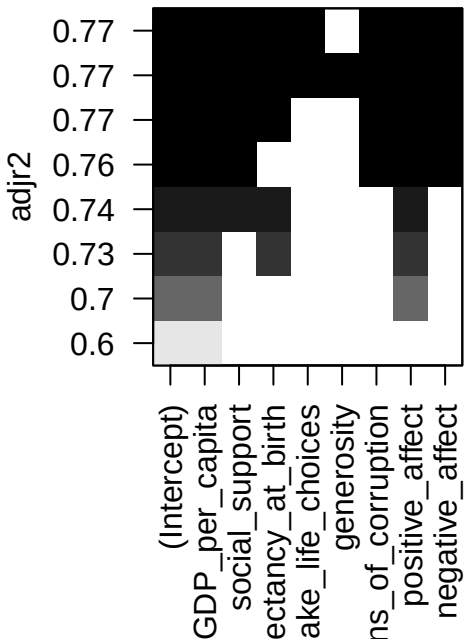
```
#Create a full model
model<- lm(life_ladder~ . - country - year,data=happiness)
b<- regsubsets(life_ladder~ . - country - year,data=happiness, nbest=1)
(rs<-summary(b))
```

```
## Subset selection object
## Call: regsubsets.formula(life_ladder ~ . - country - year, data = happiness,
##     nbest = 1)
## 8 Variables  (and intercept)
##                                   Forced in Forced out
## log_GDP_per_capita                    FALSE      FALSE
## social_support                        FALSE      FALSE
## healthy_life_expectancy_at_birth      FALSE      FALSE
## freedom_to_make_life_choices          FALSE      FALSE
## generosity                            FALSE      FALSE
## perceptions_of_corruption             FALSE      FALSE
## positive_affect                       FALSE      FALSE
## negative_affect                       FALSE      FALSE
## 1 subsets of each size up to 8
## Selection Algorithm: exhaustive
##          log_GDP_per_capita social_support healthy_life_expectancy_at_birth
## 1  ( 1 ) "*"                " "            " "
## 2  ( 1 ) "*"                " "            " "
## 3  ( 1 ) "*"                " "            "*"
## 4  ( 1 ) "*"                "*"            "*"
## 5  ( 1 ) "*"                "*"            " "
```

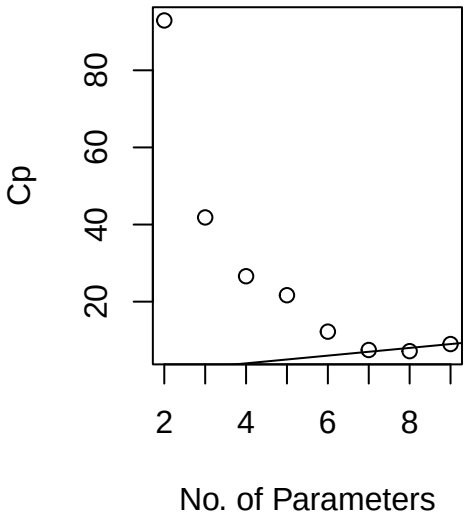
```
## 6  ( 1 ) "*"                 "*"            "*"
## 7  ( 1 ) "*"                 "*"            "*"
## 8  ( 1 ) "*"                 "*"            "*"
##          freedom_to_make_life_choices generosity perceptions_of_corruption
## 1  ( 1 ) " "                          " "        " "
## 2  ( 1 ) " "                          " "        " "
## 3  ( 1 ) " "                          " "        " "
## 4  ( 1 ) " "                          " "        " "
## 5  ( 1 ) " "                          " "        "*"
## 6  ( 1 ) " "                          " "        "*"
## 7  ( 1 ) "*"                          " "        "*"
## 8  ( 1 ) "*"                          "*"        "*"
##          positive_affect negative_affect
## 1  ( 1 ) " "             " "
## 2  ( 1 ) "*"             " "
## 3  ( 1 ) "*"             " "
## 4  ( 1 ) "*"             " "
## 5  ( 1 ) "*"             "*"
## 6  ( 1 ) "*"             "*"
## 7  ( 1 ) "*"             "*"
## 8  ( 1 ) "*"             "*"
```

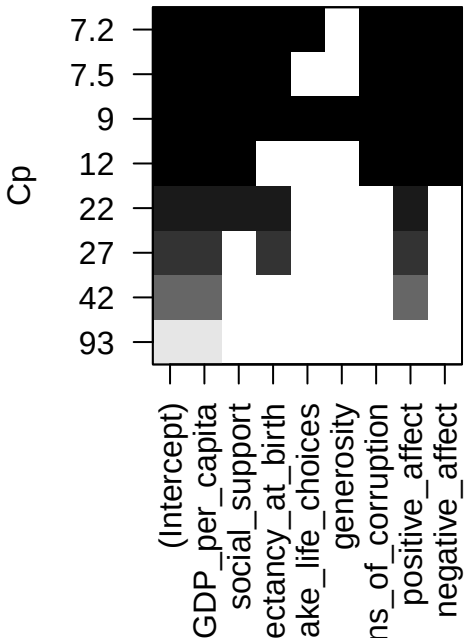
```
par(mfrow=c(1,2))
plot(2:9,rs$adjr2,xlab="No. of Parameters", ylab="Adjusted R-Square")
plot(b,scale="adjr2")
```






```
plot(2:9,rs$cp,xlab="No. of Parameters", ylab="Cp")
abline(0,1)
plot(b,scale="Cp")
```





Based on this output, it is clear that the best model based on Mallow's Cp is the model with 6 remaining predictors (7 parameters), amputating generosity and freedom_to_make_life_choices of out of the model.

AIC

```
step_model_aic <- step(model, k = 2, direction = "both", trace = 1)
```

```
## Start:  AIC=-153.84
## life_ladder ~ (country + year + log_GDP_per_capita + social_support +
##     healthy_life_expectancy_at_birth + freedom_to_make_life_choices +
##     generosity + perceptions_of_corruption + positive_affect +
##     negative_affect) - country - year
##
##                                    Df Sum of Sq    RSS     AIC
## - generosity                        1    0.0464 34.091 -155.67
## <none>                                         34.045 -153.84
## - freedom_to_make_life_choices      1    0.5704 34.615 -153.70
## - perceptions_of_corruption         1    1.7852 35.830 -149.25
## - healthy_life_expectancy_at_birth  1    1.8230 35.868 -149.12
## - negative_affect                   1    3.2887 37.334 -143.95
## - positive_affect                   1    3.9544 37.999 -141.67
## - social_support                    1    4.3488 38.394 -140.34
## - log_GDP_per_capita                1    4.9862 39.031 -138.21
##
## Step:  AIC=-155.67
## life_ladder ~ log_GDP_per_capita + social_support + healthy_life_expectancy_at_birth +
##     freedom_to_make_life_choices + perceptions_of_corruption +
##     positive_affect + negative_affect
##
##                                    Df Sum of Sq    RSS     AIC
## <none>                                         34.091 -155.67
## - freedom_to_make_life_choices      1    0.6521 34.744 -155.22
## + generosity                        1    0.0464 34.045 -153.84
## - healthy_life_expectancy_at_birth  1    1.8628 35.954 -150.81
## - perceptions_of_corruption         1    2.0164 36.108 -150.26
```

```
## - negative_affect                   1    3.2819 37.373 -145.81
## - positive_affect                   1    4.0824 38.174 -143.08
## - social_support                    1    4.3302 38.422 -142.24
## - log_GDP_per_capita                1    5.0835 39.175 -139.74
```

This output suggests that the model with the lowest AIC is the model that has 7 predictor variables, amputating generosity out of the model. This output is concurrent with the Adjusted R squared exhaustive model search results.

BIC

```
step_model_bic <- step(model, k = log(nrow(happiness)), direction = "both", trace = 1)
```

```
## Start:  AIC=-128.11
## life_ladder ~ (country + year + log_GDP_per_capita + social_support +
##     healthy_life_expectancy_at_birth + freedom_to_make_life_choices +
##     generosity + perceptions_of_corruption + positive_affect +
##     negative_affect) - country - year
##
##                                   Df Sum of Sq    RSS     AIC
## - generosity                       1    0.0464 34.091 -132.79
## - freedom_to_make_life_choices     1    0.5704 34.615 -130.82
## <none>                                        34.045 -128.11
## - perceptions_of_corruption        1    1.7852 35.830 -126.37
## - healthy_life_expectancy_at_birth 1    1.8230 35.868 -126.24
## - negative_affect                  1    3.2887 37.334 -121.07
## - positive_affect                  1    3.9544 37.999 -118.79
## - social_support                   1    4.3488 38.394 -117.46
## - log_GDP_per_capita               1    4.9862 39.031 -115.33
##
## Step:  AIC=-132.79
## life_ladder ~ log_GDP_per_capita + social_support + healthy_life_expectancy_at_birth +
##     freedom_to_make_life_choices + perceptions_of_corruption +
##     positive_affect + negative_affect
##
##                                   Df Sum of Sq    RSS     AIC
## - freedom_to_make_life_choices     1    0.6521 34.744 -135.21
## <none>                                        34.091 -132.79
## - healthy_life_expectancy_at_birth 1    1.8628 35.954 -130.79
## - perceptions_of_corruption        1    2.0164 36.108 -130.24
## + generosity                       1    0.0464 34.045 -128.11
## - negative_affect                  1    3.2819 37.373 -125.79
## - positive_affect                  1    4.0824 38.174 -123.06
## - social_support                   1    4.3302 38.422 -122.22
## - log_GDP_per_capita               1    5.0835 39.175 -119.72
##
## Step:  AIC=-135.21
## life_ladder ~ log_GDP_per_capita + social_support + healthy_life_expectancy_at_birth +
##     perceptions_of_corruption + positive_affect + negative_affect
##
##                                   Df Sum of Sq    RSS     AIC
## <none>                                        34.744 -135.21
## - healthy_life_expectancy_at_birth 1    1.9164 36.660 -133.14
## + freedom_to_make_life_choices     1    0.6521 34.091 -132.79
## + generosity                       1    0.1282 34.615 -130.82
## - perceptions_of_corruption        1    2.8573 37.601 -129.87
```

```
## - negative_affect                  1    3.1218 37.865 -128.97
## - social_support                   1    4.6403 39.384 -123.89
## - log_GDP_per_capita               1    5.2551 39.999 -121.90
## - positive_affect                  1    7.7041 42.448 -114.23
```

This output suggests that the model with the lowest BIC is the model that has 6 predictor variables, amputating generosity and freedom_to_make_life_choices out of the model. This result agrees with the best model using Mallow's Cp.

CONCLUSION:

Therefore, after conducting tests to assess multicollinearity and exhaustive model search using different selection criteria, the best model using AIC and Adjusted R Squared is the model with 7 predictor variables, amputating generosity out of the model. The best model using BIC and Mallow's Cp is the model with 6 predictor variables, amputating generosity and freedom_to_make_life_choices out of the model. The best model using R Squared is the full model with 8 predictors. Furthermore, the data analysis revealed strong evidence of multicollinearity. However, Amputation is not necessarily suitable for the multicollinearity present in this dataset, and examples will be shown below. The VIFs for each variable were < 10, but the condition numbers were > 30. So, the conclusion is that individual variables are not necessarily causing multicollinearity. This is an issue with a linear combination of variables. Here are the examples to support my claim about the ineffectiveness of amputation.

Example 1

```
#Example 1 showing Amputation was not valid solution
aic_model<- lm(life_ladder ~ . - country - year - generosity, data=happiness)
z<- model.matrix(aic_model)[,-1];
eign<- eigen(t(z)%*%z)
eign$val
```

```
## [1] 5.423606e+05 5.553023e+01 6.283986e+00 2.950917e+00 1.099246e+00
## [6] 6.232123e-01 4.127873e-01
```

```
aic_condition_numbers<-sqrt(eign$val[1]/eign$val)
aic_condition_numbers
```

```
## [1]    1.00000   98.82785  293.78286  428.71191  702.41973  932.88061 1146.25405
```

As we can see from this output, amputation is not a viable solution for addressing multicollinearity.

Example 2

```
#Example 2 showing Amputation was not valid solution
bic_model<- lm(life_ladder~ . - country - generosity - year - freedom_to_make_life_choices,data=happines
y<- model.matrix(bic_model)[,-1];
eig<- eigen(t(y)%*%y)
eig$val
```

```
## [1] 5.422821e+05 5.551507e+01 6.269694e+00 1.819673e+00 1.099147e+00
## [6] 4.130697e-01
```

```
bic_condition_numbers<-sqrt(eig$val[1]/eig$val)
bic_condition_numbers
```

```
## [1]    1.00000   98.83418  294.09623  545.90357  702.40043 1145.77921
```

```
#Example 3 showing Amputation was not valid solution
corr_model<- lm(life_ladder~ . - country - healthy_life_expectancy_at_birth - year,data=happiness)
w<- model.matrix(corr_model)[,-1];
eigw<- eigen(t(w)%*%w)
eigw$val
```

```
## [1] 1.182169e+04 7.783438e+00 4.810803e+00 1.754211e+00 1.093224e+00
## [6] 6.219985e-01 4.751948e-01
```

```
corr_model_condition_numbers<-sqrt(eigw$val[1]/eigw$val)
corr_model_condition_numbers
```

```
## [1]   1.00000  38.97211  49.57137  82.09161 103.98844 137.86215 157.72621
```

```
#Example 4 showing Amputation was not valid solution
corr2_model<- lm(life_ladder~ . - country - healthy_life_expectancy_at_birth - social_support - year,da
wn<- model.matrix(corr2_model)[,-1];
eigwn<- eigen(t(wn)%*%wn)
eigwn$val
```

```
## [1] 1.173622e+04 7.772394e+00 4.781458e+00 1.726500e+00 8.963542e-01
## [6] 6.190683e-01
```

```
corr2_model_condition_numbers<-sqrt(eigwn$val[1]/eigwn$val)
corr2_model_condition_numbers
```

```
## [1]   1.00000  38.85856  49.54319  82.44814 114.42589 137.68762
```

These results support my conclusion that amputation is not a viable solution and furthermore, that the multicollinearity is complex. The multicollinearity is associated with a linear combination of variables that is not easily identifiable.

Question 9

```
happiness <- read.csv("/Users/aidanashrafi/Downloads/Happiness_2018.csv")
```

BACKWARD ELIMINATION:

```
backwardModel <- lm(life_ladder~.-country-year, data=happiness)
summary(backwardModel)
```

```
##
## Call:
## lm(formula = life_ladder ~ . - country - year, data = happiness)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.66321 -0.33003  0.02126  0.29709  1.66671
##
## Coefficients:
##                                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)                      -4.62111    0.87144  -5.303 5.28e-07 ***
## log_GDP_per_capita                0.35544    0.08478   4.192 5.31e-05 ***
## social_support                    2.83854    0.72502   3.915 0.000150 ***
## healthy_life_expectancy_at_birth  0.03672    0.01448   2.535 0.012536 *
## freedom_to_make_life_choices      0.78692    0.55499   1.418 0.158812
## generosity                        0.13951    0.34495   0.404 0.686613
## perceptions_of_corruption        -0.76970    0.30685  -2.508 0.013462 *
## positive_affect                   2.11173    0.56563   3.733 0.000290 ***
## negative_affect                   2.47878    0.72805   3.405 0.000901 ***
## ---
```

```
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.5326 on 120 degrees of freedom
##   (12 observations deleted due to missingness)
## Multiple R-squared:  0.7829, Adjusted R-squared:  0.7684
## F-statistic: 54.09 on 8 and 120 DF,  p-value: < 2.2e-16
```

We start with the full model for backwards elimination and iteratively get rid of the least significant predictor, corresponding to the predictor with the greatest p-value that is still greater than 0.05. This is generosity.

```
backwardModel <- update(backwardModel, life_ladder~.-country-year-generosity)
summary(backwardModel)
```

```
##
## Call:
## lm(formula = life_ladder ~ log_GDP_per_capita + social_support +
##     healthy_life_expectancy_at_birth + freedom_to_make_life_choices +
##     perceptions_of_corruption + positive_affect + negative_affect,
##     data = happiness)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.68864 -0.32471  0.03681  0.29617  1.66386
##
## Coefficients:
##                                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)                      -4.58279    0.86328  -5.309 5.08e-07 ***
## log_GDP_per_capita                0.34651    0.08158   4.248 4.27e-05 ***
## social_support                    2.83170    0.72231   3.920 0.000147 ***
## healthy_life_expectancy_at_birth  0.03705    0.01441   2.571 0.011343 *
## freedom_to_make_life_choices      0.82753    0.54394   1.521 0.130776
## perceptions_of_corruption        -0.79740    0.29807  -2.675 0.008502 **
## positive_affect                   2.13475    0.56081   3.807 0.000223 ***
## negative_affect                   2.47608    0.72550   3.413 0.000875 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.5308 on 121 degrees of freedom
##   (12 observations deleted due to missingness)
## Multiple R-squared:  0.7826, Adjusted R-squared:   0.77
## F-statistic: 62.22 on 7 and 121 DF,  p-value: < 2.2e-16
```

Again, get rid of the predictor with the greatest p-value that is still greater than 0.05. This is freedom_to_make_life_choices.

```
backwardModel <- update(backwardModel, life_ladder~.-country-year-generosity-freedom_to_make_life_choic
summary(backwardModel)
```

```
##
## Call:
## lm(formula = life_ladder ~ log_GDP_per_capita + social_support +
##     healthy_life_expectancy_at_birth + perceptions_of_corruption +
##     positive_affect + negative_affect, data = happiness)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
```

```
## -1.65280 -0.32754  0.04575  0.30626  1.68566
##
## Coefficients:
##                                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)                      -4.17274    0.83089  -5.022 1.75e-06 ***
## log_GDP_per_capita                0.33945    0.07992   4.247 4.23e-05 ***
## social_support                    2.99574    0.71504   4.190 5.29e-05 ***
## healthy_life_expectancy_at_birth  0.03797    0.01445   2.628  0.00968 **
## perceptions_of_corruption        -0.94113    0.28658  -3.284  0.00133 **
## positive_affect                   2.52001    0.48783   5.166 9.36e-07 ***
## negative_affect                   2.36584    0.72405   3.268  0.00141 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.5326 on 123 degrees of freedom
##   (11 observations deleted due to missingness)
## Multiple R-squared:  0.7775, Adjusted R-squared:  0.7666
## F-statistic: 71.63 on 6 and 123 DF,  p-value: < 2.2e-16
```

All p-values are now less than 0.05 (all predictors are significant), so we stop. This is the model we get from backward elimination.

FORWARD SELECTION:

```
forwardModel <- lm(life_ladder~1, data=happiness)
add1(forwardModel, ~log_GDP_per_capita+social_support+healthy_life_expectancy_at_birth+perceptions_of_c
```

```
## Warning in add1.lm(forwardModel, ~log_GDP_per_capita + social_support + : using
## the 129/141 rows from a combined fit
```

```
## Single term additions
##
## Model:
## life_ladder ~ 1
##                                  Df Sum of Sq     RSS     AIC  F value
## <none>                                       156.807  27.181
## log_GDP_per_capita                1    94.978  61.829 -90.873 213.5261
## social_support                    1    86.234  70.573 -73.808 169.8458
## healthy_life_expectancy_at_birth  1    89.058  67.749 -79.076 182.7179
## perceptions_of_corruption         1    30.543 126.264   1.234  33.6242
## positive_affect                   1    28.014 128.793   3.793  30.2339
## negative_affect                   1    31.536 125.271   0.216  34.9917
## freedom_to_make_life_choices      1    42.964 113.843 -12.124  52.4583
## generosity                        1     0.409 156.398  28.845   0.3632
##                                     Pr(>F)
## <none>
## log_GDP_per_capita               < 2.2e-16 ***
## social_support                   < 2.2e-16 ***
## healthy_life_expectancy_at_birth < 2.2e-16 ***
## perceptions_of_corruption        4.304e-08 ***
## positive_affect                  1.779e-07 ***
## negative_affect                  2.450e-08 ***
## freedom_to_make_life_choices     2.744e-11 ***
## generosity                          0.5477
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

We start with only the intercept and iteratively add the most significant predictors, which corresponds to the predictor with the smallest p-value that is still less than 0.05. Since we cannot see further into the decimals of the p-values, we turn to the predictor that has the largest F value out of log_GDP_per_capita, social_support, and healthy_life_expectancy_at_birth. We select log_GDP_per_capita to add to the model.

```
forwardModel <- lm(life_ladder~log_GDP_per_capita, data=happiness)
add1(forwardModel, ~log_GDP_per_capita+social_support+healthy_life_expectancy_at_birth+perceptions_of_c
```

```
## Warning in add1.lm(forwardModel, ~log_GDP_per_capita + social_support + : using
## the 129/141 rows from a combined fit
```

```
## Single term additions
##
## Model:
## life_ladder ~ log_GDP_per_capita
##                                 Df Sum of Sq    RSS      AIC F value    Pr(>F)
## <none>                                       61.829  -90.873
## social_support                   1    7.8897 53.939 -106.483 20.1855 1.473e-05
## healthy_life_expectancy_at_birth  1    5.3046 56.524 -100.444 12.9509 0.0004447
## perceptions_of_corruption         1    6.1659 55.663 -102.425 15.2866 0.0001443
## positive_affect                   1   15.0617 46.767 -124.888 44.4442 5.786e-10
## negative_affect                   1    0.0232 61.805  -88.921  0.0517 0.8204310
## freedom_to_make_life_choices      1   10.9631 50.865 -114.051 29.7433 2.211e-07
## generosity                        1    3.8578 57.971  -97.184  9.1836 0.0029165
##
## <none>
## social_support                   ***
## healthy_life_expectancy_at_birth ***
## perceptions_of_corruption        ***
## positive_affect                  ***
## negative_affect
## freedom_to_make_life_choices     ***
## generosity                       **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Add the predictor with the smallest p-value that is still less than 0.05. This is positive_affect.

```
forwardModel <- lm(life_ladder~log_GDP_per_capita+positive_affect, data=happiness)
add1(forwardModel, ~log_GDP_per_capita+social_support+healthy_life_expectancy_at_birth+perceptions_of_c
```

```
## Warning in add1.lm(forwardModel, ~log_GDP_per_capita + social_support + : using
## the 129/139 rows from a combined fit
```

```
## Single term additions
##
## Model:
## life_ladder ~ log_GDP_per_capita + positive_affect
##                                 Df Sum of Sq    RSS     AIC F value    Pr(>F)
## <none>                                       46.767 -124.89
## social_support                   1    3.6782 43.089 -133.46 11.5242 0.0009023
## healthy_life_expectancy_at_birth  1    4.8938 41.873 -137.15 15.7776 0.0001154
## perceptions_of_corruption         1    1.7687 44.998 -127.86  5.3065 0.0227741
## negative_affect                   1    0.9176 45.849 -125.44  2.7019 0.1025575
## freedom_to_make_life_choices      1    1.6651 45.102 -127.56  4.9839 0.0272313
## generosity                        1    0.6567 46.110 -124.71  1.9226 0.1678515
```

```
##
## <none>
## social_support                   ***
## healthy_life_expectancy_at_birth ***
## perceptions_of_corruption        *
## negative_affect
## freedom_to_make_life_choices     *
## generosity
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Add the predictor with the smallest p-value that is still less than 0.05. This is healthy_life_expectancy_at_birth.

```
forwardModel <- lm(life_ladder~log_GDP_per_capita+positive_affect+healthy_life_expectancy_at_birth, data
add1(forwardModel, ~log_GDP_per_capita+social_support+healthy_life_expectancy_at_birth+perceptions_of_c
```

```
## Warning in add1.lm(forwardModel, ~log_GDP_per_capita + social_support + : using
## the 129/136 rows from a combined fit
```

```
## Single term additions
##
## Model:
## life_ladder ~ log_GDP_per_capita + positive_affect + healthy_life_expectancy_at_birth
##                              Df Sum of Sq    RSS     AIC F value  Pr(>F)
## <none>                                    41.873 -137.15
## social_support                1   1.96389 39.909 -141.34  6.4464 0.01229 *
## perceptions_of_corruption     1   1.42618 40.447 -139.62  4.6191 0.03346 *
## negative_affect               1   0.84514 41.028 -137.78  2.6985 0.10284
## freedom_to_make_life_choices  1   1.33574 40.537 -139.33  4.3166 0.03970 *
## generosity                    1   0.41518 41.458 -136.43  1.3119 0.25414
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Add the predictor with the smallest p-value that is still less than 0.05. This is social_support.

```
forwardModel <- lm(life_ladder~log_GDP_per_capita+positive_affect+healthy_life_expectancy_at_birth+socia
add1(forwardModel, ~log_GDP_per_capita+social_support+healthy_life_expectancy_at_birth+perceptions_of_c
```

```
## Warning in add1.lm(forwardModel, ~log_GDP_per_capita + social_support + : using
## the 129/136 rows from a combined fit
```

```
## Single term additions
##
## Model:
## life_ladder ~ log_GDP_per_capita + positive_affect + healthy_life_expectancy_at_birth +
##     social_support
##                              Df Sum of Sq    RSS     AIC F value   Pr(>F)
## <none>                                    39.909 -141.34
## perceptions_of_corruption     1   2.04378 37.865 -146.12  7.0167 0.009076 **
## negative_affect               1   2.30824 37.601 -147.03  7.9804 0.005475 **
## freedom_to_make_life_choices  1   1.13421 38.775 -143.06  3.8026 0.053324 .
## generosity                    1   0.50034 39.409 -140.97  1.6505 0.201177
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Add the predictor with the smallest p-value that is still less than 0.05. This is negative_affect.

```
forwardModel <- lm(life_ladder~log_GDP_per_capita+positive_affect+healthy_life_expectancy_at_birth+socia
add1(forwardModel, ~log_GDP_per_capita+social_support+healthy_life_expectancy_at_birth+perceptions_of_c
```

```
## Warning in add1.lm(forwardModel, ~log_GDP_per_capita + social_support + : using
## the 129/136 rows from a combined fit
```

```
## Single term additions
##
## Model:
## life_ladder ~ log_GDP_per_capita + positive_affect + healthy_life_expectancy_at_birth +
##     social_support + negative_affect
##                              Df Sum of Sq    RSS     AIC F value   Pr(>F)
## <none>                                    37.601 -147.03
## perceptions_of_corruption     1   2.85734 34.744 -155.22 10.6091 0.001438 **
## freedom_to_make_life_choices  1   1.49305 36.108 -150.26  5.3341 0.022501 *
## generosity                    1   0.64631 36.955 -147.27  2.2561 0.135530
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Add the predictor with the smallest p-value that is still less than 0.05. This is perceptions_of_corruption.

```
forwardModel <- lm(life_ladder~log_GDP_per_capita+positive_affect+healthy_life_expectancy_at_birth+socia
add1(forwardModel, ~log_GDP_per_capita+social_support+healthy_life_expectancy_at_birth+perceptions_of_c
```

```
## Warning in add1.lm(forwardModel, ~log_GDP_per_capita + social_support + : using
## the 129/130 rows from a combined fit
```

```
## Single term additions
##
## Model:
## life_ladder ~ log_GDP_per_capita + positive_affect + healthy_life_expectancy_at_birth +
##     social_support + negative_affect + perceptions_of_corruption
##                              Df Sum of Sq    RSS     AIC F value Pr(>F)
## <none>                                    34.744 -155.22
## freedom_to_make_life_choices  1   0.65212 34.091 -155.67  2.3337 0.1292
## generosity                    1   0.12815 34.615 -153.70  0.4517 0.5028
```

We stop because all p-values exceed 0.05, so the predictors are not significant enough to be added to the model.

```
summary(forwardModel)
```

```
##
## Call:
## lm(formula = life_ladder ~ log_GDP_per_capita + positive_affect +
##     healthy_life_expectancy_at_birth + social_support + negative_affect +
##     perceptions_of_corruption, data = happiness)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.65280 -0.32754  0.04575  0.30626  1.68566
##
## Coefficients:
##                                Estimate Std. Error t value Pr(>|t|)
## (Intercept)                    -4.17274    0.83089  -5.022 1.75e-06 ***
## log_GDP_per_capita              0.33945    0.07992   4.247 4.23e-05 ***
## positive_affect                 2.52001    0.48783   5.166 9.36e-07 ***
```

```
## healthy_life_expectancy_at_birth  0.03797    0.01445   2.628  0.00968 **
## social_support                    2.99574    0.71504   4.190 5.29e-05 ***
## negative_affect                   2.36584    0.72405   3.268  0.00141 **
## perceptions_of_corruption        -0.94113    0.28658  -3.284  0.00133 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.5326 on 123 degrees of freedom
##   (11 observations deleted due to missingness)
## Multiple R-squared:  0.7775, Adjusted R-squared:  0.7666
## F-statistic: 71.63 on 6 and 123 DF,  p-value: < 2.2e-16
```

Thus, this is the model derived from forward selection.

Question 10

We are going to run some model diagnostics on the reduced model derived from backward elimination (though we actually arrive at the same model from backward elimination and forward selection).

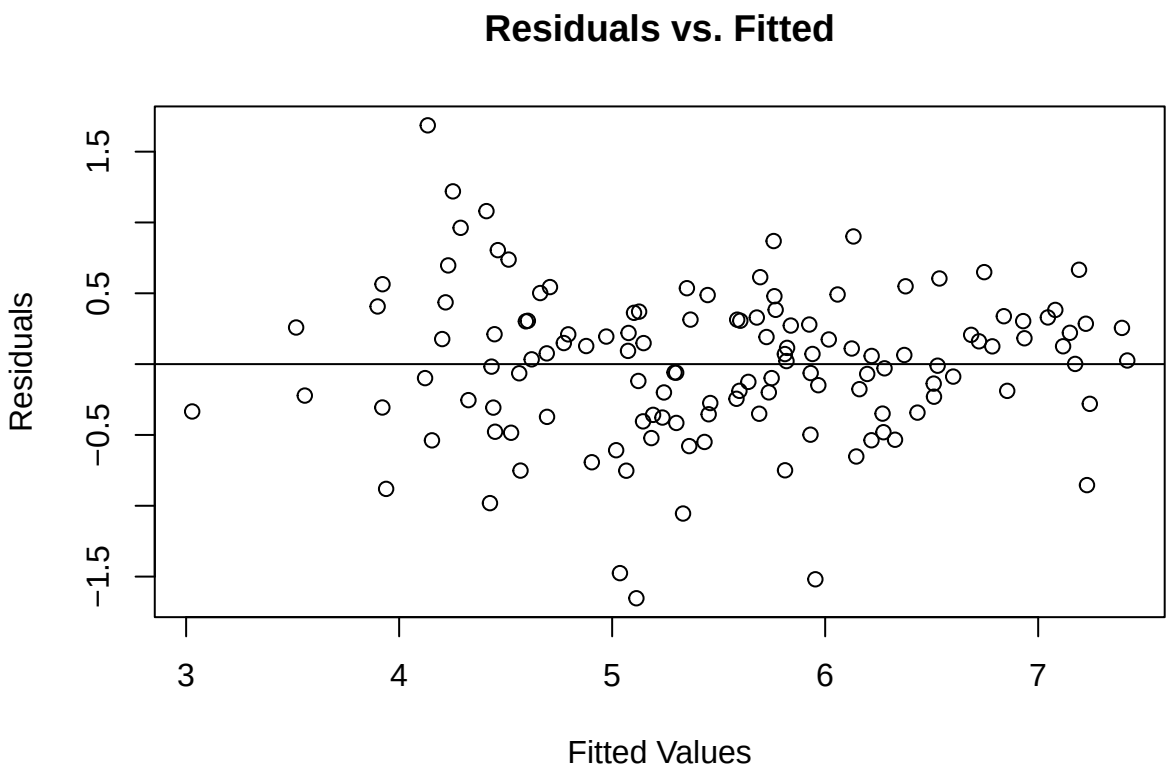
```
summary(backwardModel)
```

```
##
## Call:
## lm(formula = life_ladder ~ log_GDP_per_capita + social_support +
##     healthy_life_expectancy_at_birth + perceptions_of_corruption +
##     positive_affect + negative_affect, data = happiness)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.65280 -0.32754  0.04575  0.30626  1.68566
##
## Coefficients:
##                                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)                      -4.17274    0.83089  -5.022 1.75e-06 ***
## log_GDP_per_capita                0.33945    0.07992   4.247 4.23e-05 ***
## social_support                    2.99574    0.71504   4.190 5.29e-05 ***
## healthy_life_expectancy_at_birth  0.03797    0.01445   2.628  0.00968 **
## perceptions_of_corruption        -0.94113    0.28658  -3.284  0.00133 **
## positive_affect                   2.52001    0.48783   5.166 9.36e-07 ***
## negative_affect                   2.36584    0.72405   3.268  0.00141 **
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.5326 on 123 degrees of freedom
##   (11 observations deleted due to missingness)
## Multiple R-squared:  0.7775, Adjusted R-squared:  0.7666
## F-statistic: 71.63 on 6 and 123 DF,  p-value: < 2.2e-16
```

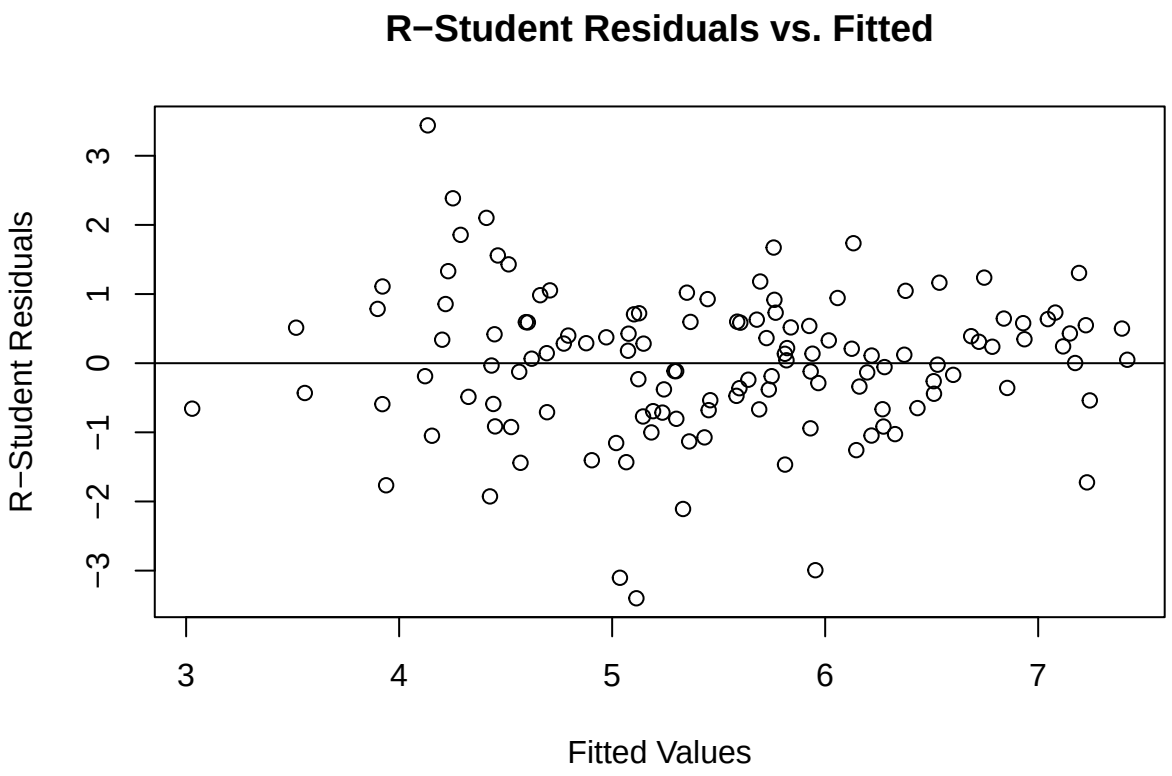
A quick refresher: k=6 p=7 n=130

A

```
plot(fitted(backwardModel), resid(backwardModel), main="Residuals vs. Fitted", xlab="Fitted Values", yla
abline(h=0)
```




```
plot(fitted(backwardModel), rstudent(backwardModel), main="R-Student Residuals vs. Fitted", xlab="Fitte
abline(h=0)
```

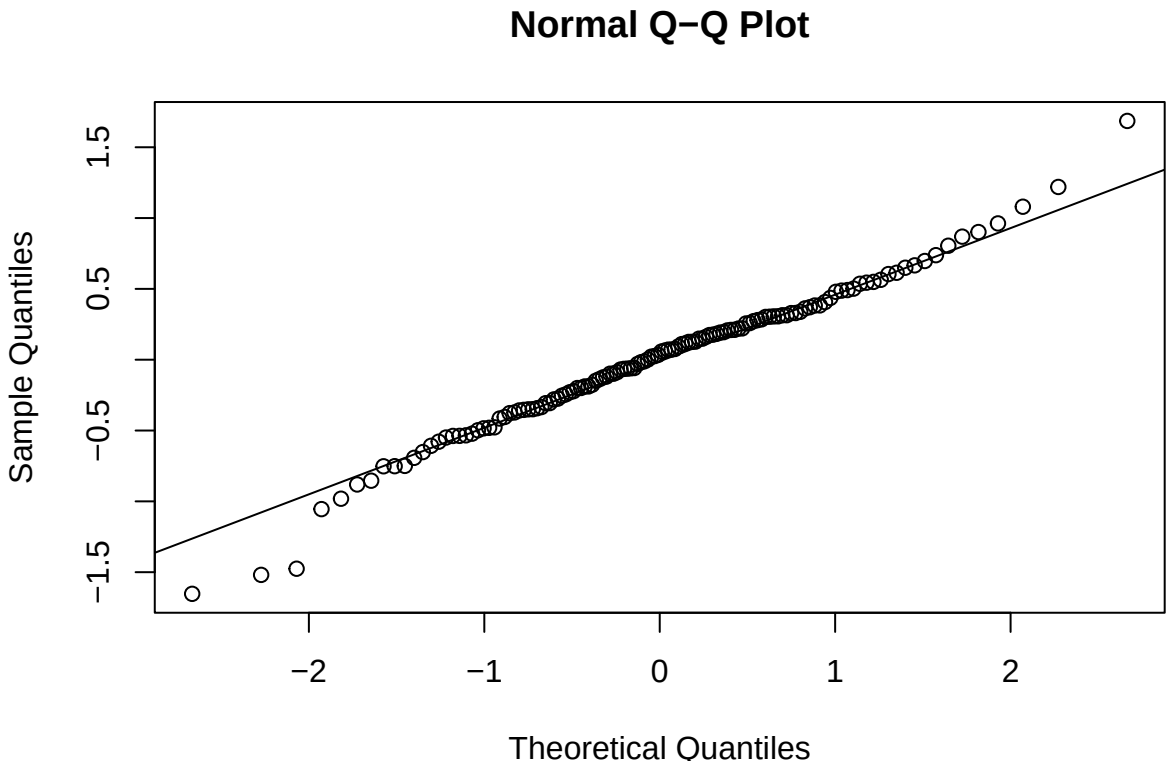


Again, two graphs are pictured. Both have fitted happiness values on the horizontal axis. The first graph has regular residuals on the vertical axis, while the second graph has r-student residuals on the vertical axis.

Again, both graphs show that the points are centered around the horizontal line at 0, indicating unbiasness. In terms of scedasticity, they seem to be fairly evenly spaced around the horizontal band. It is pretty safe to assume that the constant variance assumption holds. Note that there may be some larger variance corresponding with smaller fitted values, but we do not think it is enough to be of concern.

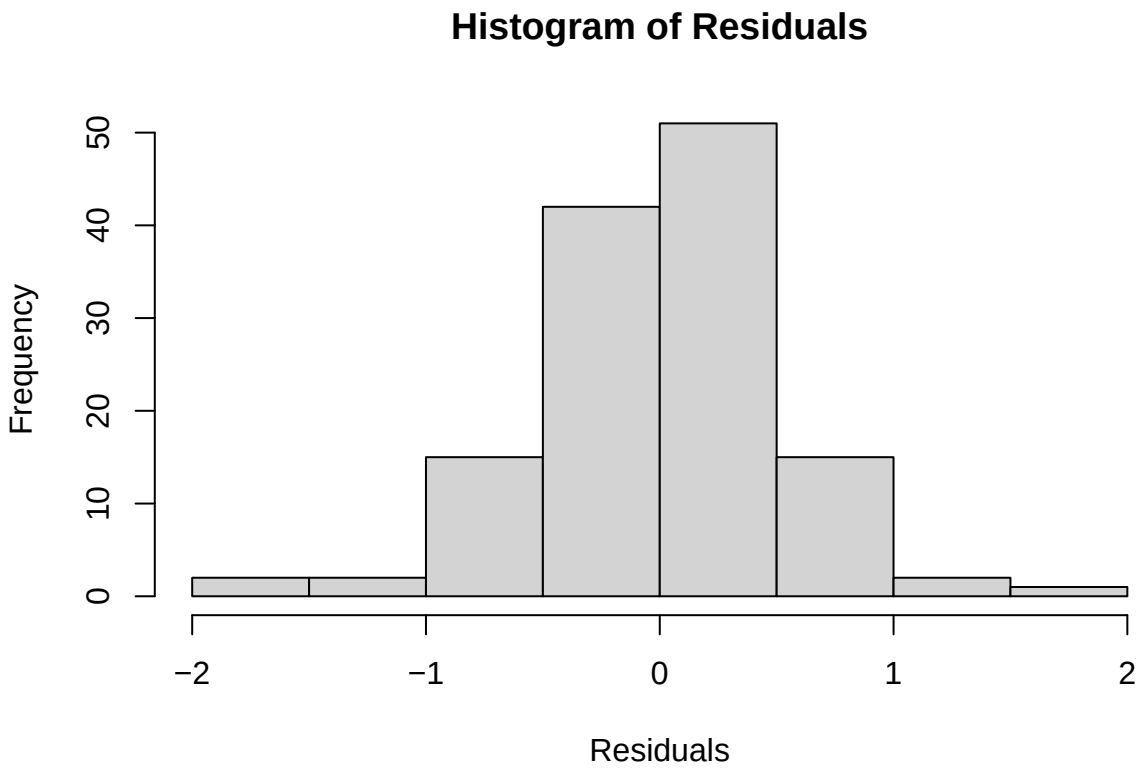
B

```
qqnorm(resid(backwardModel))
qqline(resid(backwardModel))
```



The residuals seem to be pretty normally distributed, as the Q-Q plot follows a pretty linear path. Again, there seems to be some trailing off in the points at the tails, but nothing of great concern. One may also interpret a slight S-curve, which would indicate a shorter tail distribution, but again, does not seem to be too concerning.

```
hist(resid(backwardModel), main="Histogram of Residuals", xlab="Residuals", ylab="Frequency")
```



The histogram is also indicative of the normal nature of the residuals.

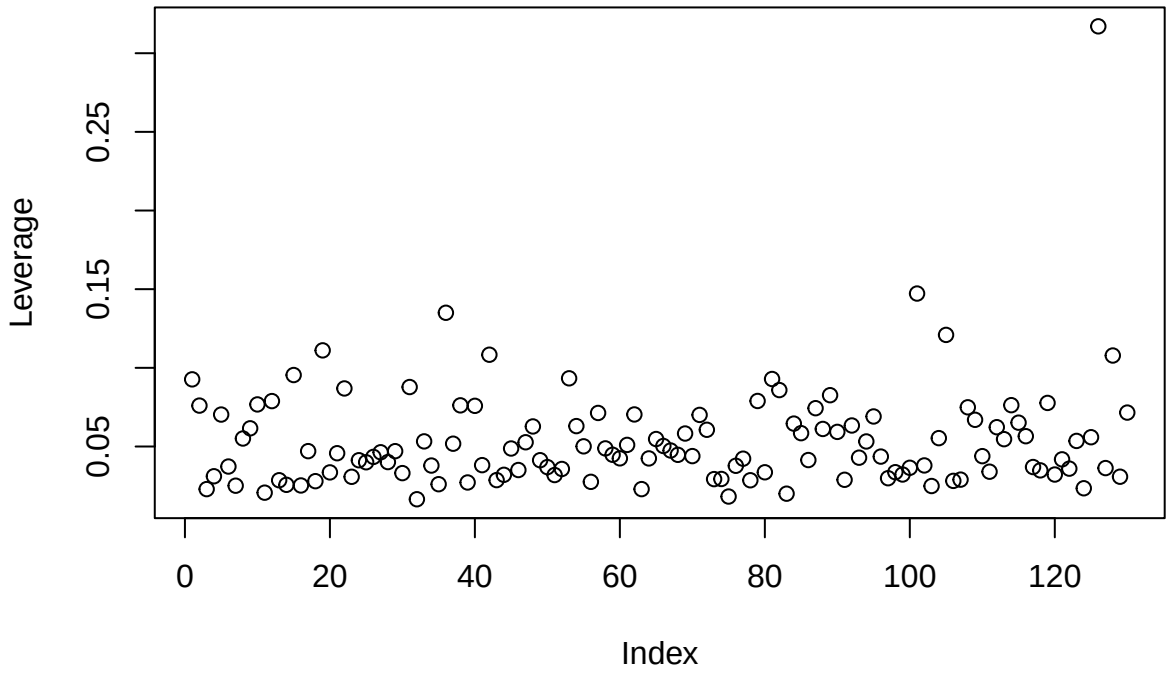
```
shapiro.test(residuals(backwardModel))
```

```
##
##  Shapiro-Wilk normality test
##
## data:  residuals(backwardModel)
## W = 0.98181, p-value = 0.07885
```

The p-value is 0.07885, which is a definite improvement over the 0.039 p-value obtained in the full model above. It is also over the threshold of the normal alpha of 0.05, which is a positive sign. A larger p-value means we do not reject the null hypothesis of normality.

C

```
plot(lm.influence(backwardModel)$hat, main="Checking For Large Leverage Points", xlab="Index", ylab="Le
```



Again, we see a point with an obvious leverage greater than the rest. However, this may not be the only leverage point. We must do some quantitative analysis to see which points have large leverage points. Generally, any point with leverage greater than 2*[(k+1)/n] is considered an outlying X observation (i.e. a point with large leverage).

```
boundary = 2*(7/130)
print(boundary)
```

```
## [1] 0.1076923
```

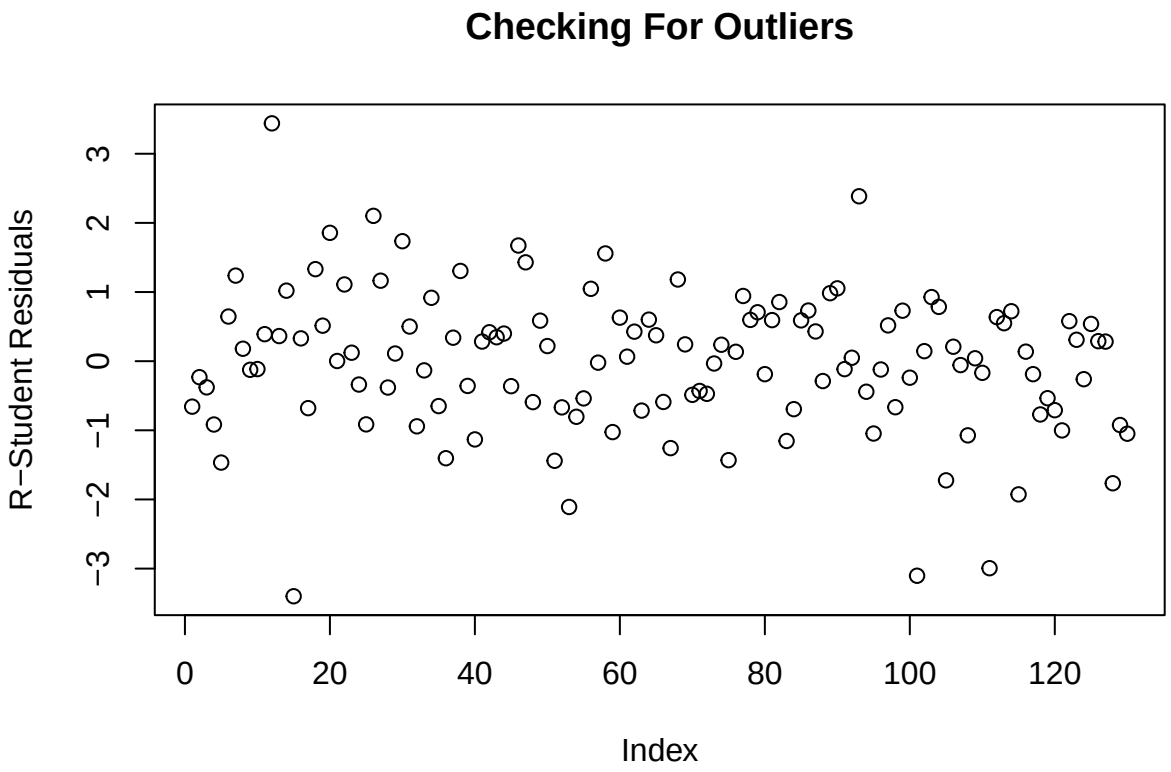
```
leverage = lm.influence(backwardModel)$hat
leverage[which(leverage>boundary)]
```

```
##        19        39        45       107       112       137       139
## 0.1110963 0.1350127 0.1083399 0.1472193 0.1209317 0.3170732 0.1078397
```

While it only looks like one point has a large leverage from the plot, it seems as though seven points have unusually large leverages. As indicated by the numbers, these correspond to countries Burundi, Eswatini, Georgia, Rwanda, Singapore, Venezuela, and Yemen.

D

```
plot(rstudent(backwardModel), main="Checking For Outliers", xlab="Index", ylab="R-Student Residuals")
```



Visually, there seem to be a couple points that could be outliers. Again, we must check quantitatively if these points are actually outliers using r-student residuals.

```
jack = rstudent(backwardModel)
jack[which.max(abs(jack))]
```

```
##       12
## 3.439728
```

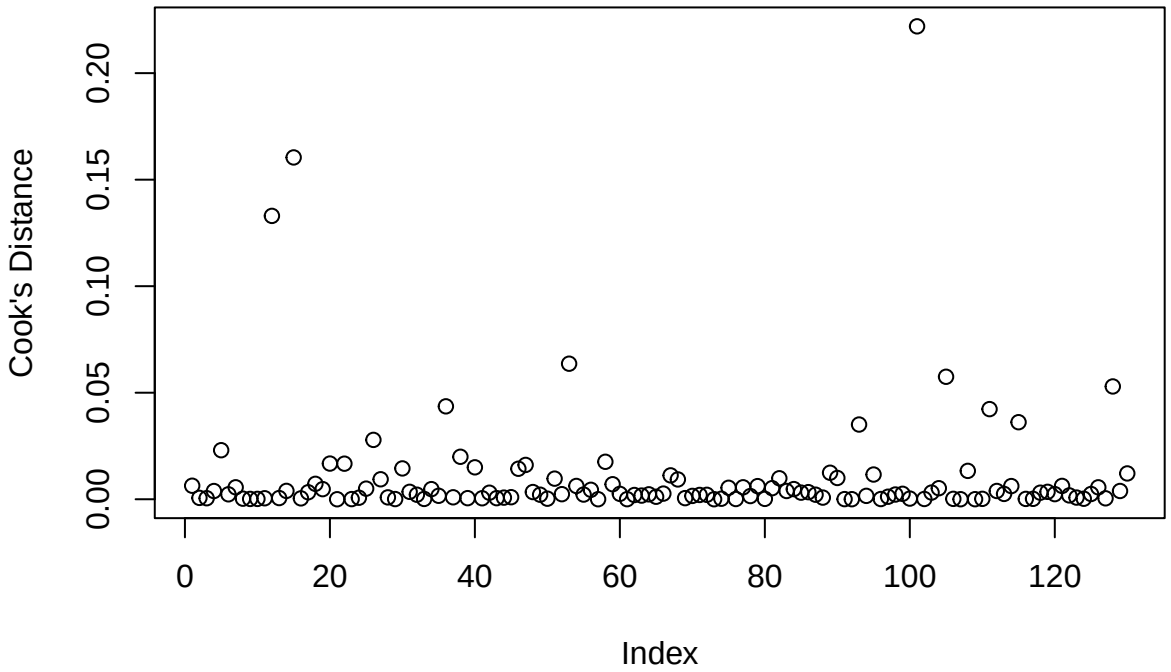
```
qt(1-(.05/(2*130)), 130-7-1)
```

```
## [1] 3.651938
```

We use the Bonferroni correction since we are checking if any points in the data set is an outlier. Since the point with the largest absolute value jack residual is less than the Bonferroni critical value, we fail to reject the null hypothesis that all the points come from the same model. There are no outliers.

E

```
plot(cooks.distance(backwardModel), main="Cook's Distance", xlab="Index", ylab="Cook's Distance")
```



Again, we provide a visual of the Cook's distance of all the points. There seem to be a couple that have a larger distance compared to the other. However, we stick to our rule of thumb that a cook's distance of under 0.5 is not an influential point. From the graph, we can already see that the point with the largest distance is only around 0.2.

```
cook = cooks.distance(backwardModel)
cook[which.max(abs(cook))]
```

```
##       107
## 0.2219613
```

We confirm this by checking the point with the largest Cook's distance. It is less than 0.5 and within our rule of thumb, so there are no influential points.