Untitled

R Markdown

This is an R Markdown document. Markdown is a simple formatting syntax for authoring HTML, PDF, and MS Word documents. For more details on using R Markdown see http://rmarkdown.rstudio.com.

When you click the **Knit** button a document will be generated that includes both content as well as the output of any embedded R code chunks within the document. You can embed an R code chunk like this:

```
df1 = read.csv('C:\\Users\\guesm\\Documents\\Methods of Data Science\\2020.csv')
df2 = read.csv('C:\\Users\\guesm\\Documents\\Methods of Data Science\\2021.csv')
```

##Data Cleaning

renaming the data columns for both data sets

```
names(df1)<- c('Country.name', 'Regional.indicator', 'Ladder.score', 'Standard.error.of.ladder.score',
names(df1)
```

```
##  [1] "Country.name"
##  [2] "Regional.indicator"
##  [3] "Ladder.score"
##  [4] "Standard.error.of.ladder.score"
##  [5] "upperwhisker"
##  [6] "lowerwhisker"
##  [7] "Logged.GDP.Per.capita"
##  [8] "Social.support"
##  [9] "Healthy.life.expectancy"
## [10] "Freedom.to.make.life.choices"
## [11] "Generosity"
## [12] "Perceptions.of.corruption"
## [13] "Ladder.score.in.dystopia"
## [14] "Explained.by.log.GDP.Per.Capita"
## [15] "Explained.by.social.support"
## [16] "Explained.by.healthy.life.expectancy"
## [17] "Explained.by.freedom.to.make.choices"
## [18] "Explained.by.Generosity"
## [19] "Explained.by.perceptions.of.corruptions"
## [20] "Dystopia.residual"
```

```
names(df2)<- c('Country.name', 'Regional.indicator', 'Ladder.score', 'Standard.error.of.ladder.score',
names(df2)
```

```
##  [1] "Country.name"
##  [2] "Regional.indicator"
##  [3] "Ladder.score"
```

```
##  [4] "Standard.error.of.ladder.score"
##  [5] "upperwhisker"
##  [6] "lowerwhisker"
##  [7] "Logged.GDP.Per.capita"
##  [8] "Social.support"
##  [9] "Healthy.life.expectancy"
## [10] "Freedom.to.make.life.choices"
## [11] "Generosity"
## [12] "Perceptions.of.corruption"
## [13] "Ladder.score.in.dystopia"
## [14] "Explained.by.log.GDP.Per.Capita"
## [15] "Explained.by.social.support"
## [16] "Explained.by.healthy.life.expectancy"
## [17] "Explained.by.freedom.to.make.choices"
## [18] "Explained.by.Generosity"
## [19] "Explained.by.perceptions.of.corruptions"
## [20] "Dystopia.residual"
```

deleting unnesscary columns for both data sets. removing the same columns from each dataset. This is for data base number 1.

```
df1<-subset(df1, select = -c(Standard.error.of.ladder.score))

df1 <- subset(df1, select = -c(upperwhisker))

df1 <- subset(df1, select = -c(lowerwhisker))

df1 <- subset(df1, select = -c(Ladder.score.in.dystopia))

df1 <- subset(df1, select = -c(Explained.by.log.GDP.Per.Capita))

df1 <- subset(df1, select = -c(Explained.by.social.support))

df1 <- subset(df1, select = -c(Explained.by.healthy.life.expectancy))

df1 <- subset(df1, select = -c(Explained.by.freedom.to.make.choices))

df1 <- subset(df1, select = -c(Explained.by.Generosity))

df1 <- subset(df1, select = -c(Explained.by.perceptions.of.corruptions))

df1 <- subset(df1, select = -c(Dystopia.residual))

df1 <- subset(df1, select = -c(Country.name))

df1 <- subset(df1, select = -c(Regional.indicator))
```

This code block deletes columns we do not need from column from dataset 2.

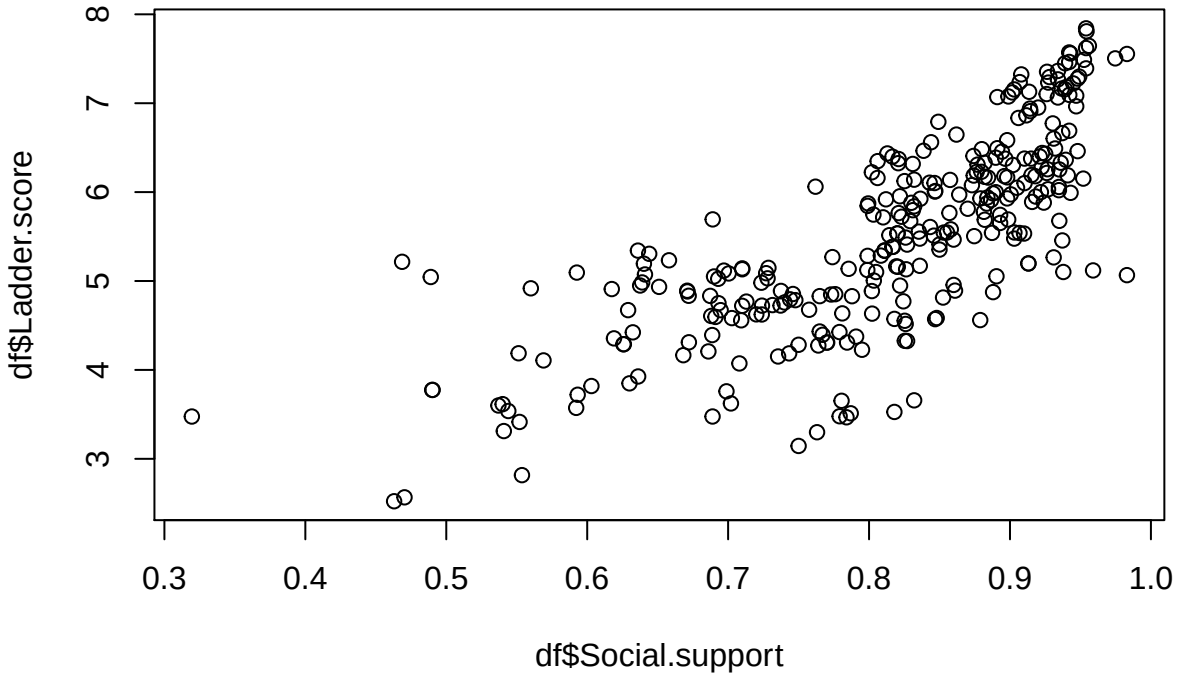
```
df2<-subset(df2, select = -c(Standard.error.of.ladder.score))

df2 <- subset(df2, select = -c(upperwhisker))
```

```r
df2 <- subset(df2, select = -c(lowerwhisker))

df2 <- subset(df2, select = -c(Ladder.score.in.dystopia))

df2 <- subset(df2, select = -c(Explained.by.log.GDP.Per.Capita))

df2 <- subset(df2, select = -c(Explained.by.social.support))

df2 <- subset(df2, select = -c(Explained.by.healthy.life.expectancy))

df2 <- subset(df2, select = -c(Explained.by.freedom.to.make.choices))

df2 <- subset(df2, select = -c(Explained.by.Generosity))

df2 <- subset(df2, select = -c(Explained.by.perceptions.of.corruptions))

df2 <- subset(df2, select = -c(Dystopia.residual))

df2 <- subset(df2, select = -c(Country.name))

df2 <- subset(df2, select = -c(Regional.indicator))
```

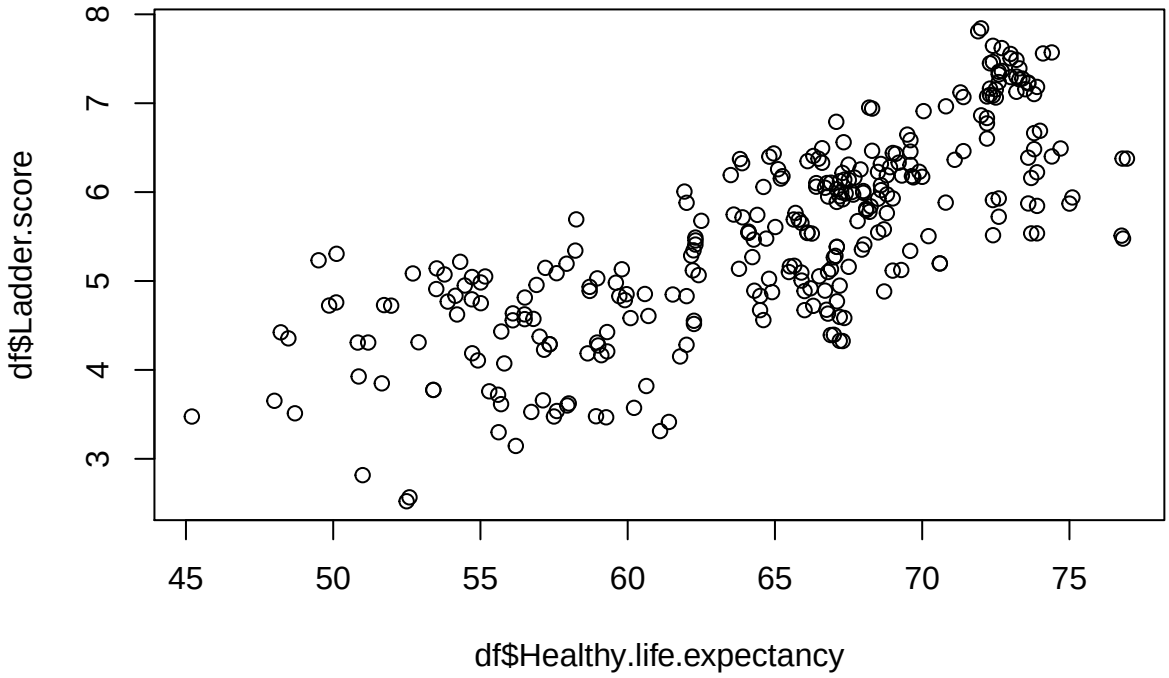
```r
df = rbind(df1,df2)
```

EDA

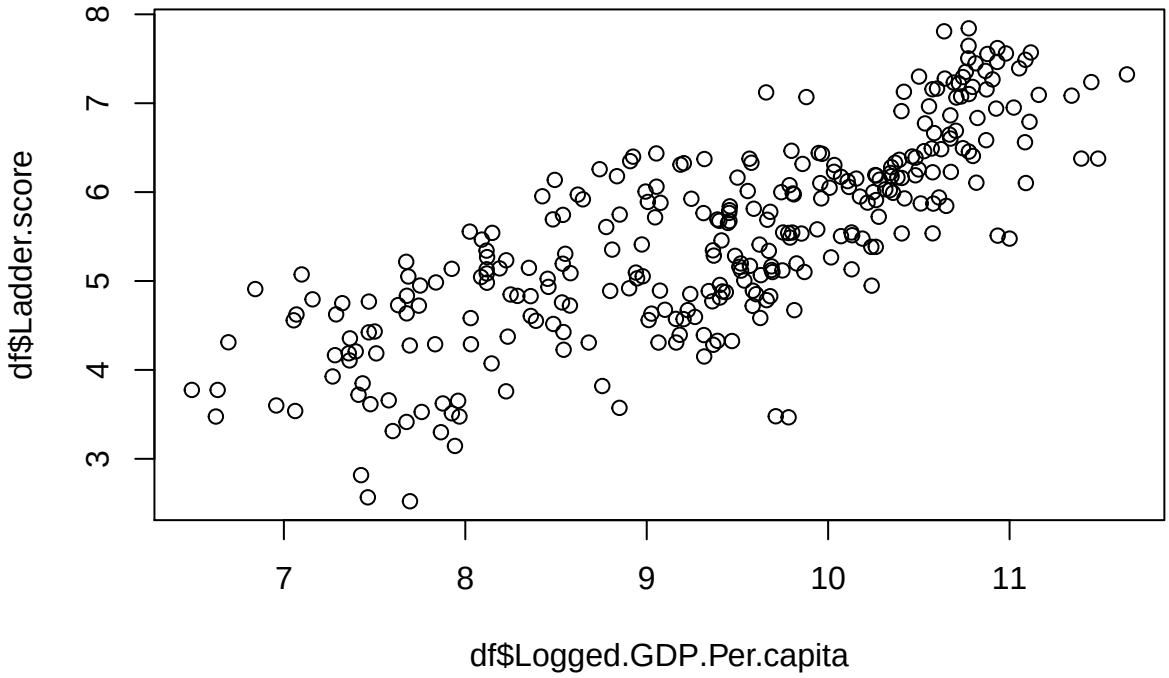
```r
plot(x = df$Social.support, y = df$Ladder.score)
```



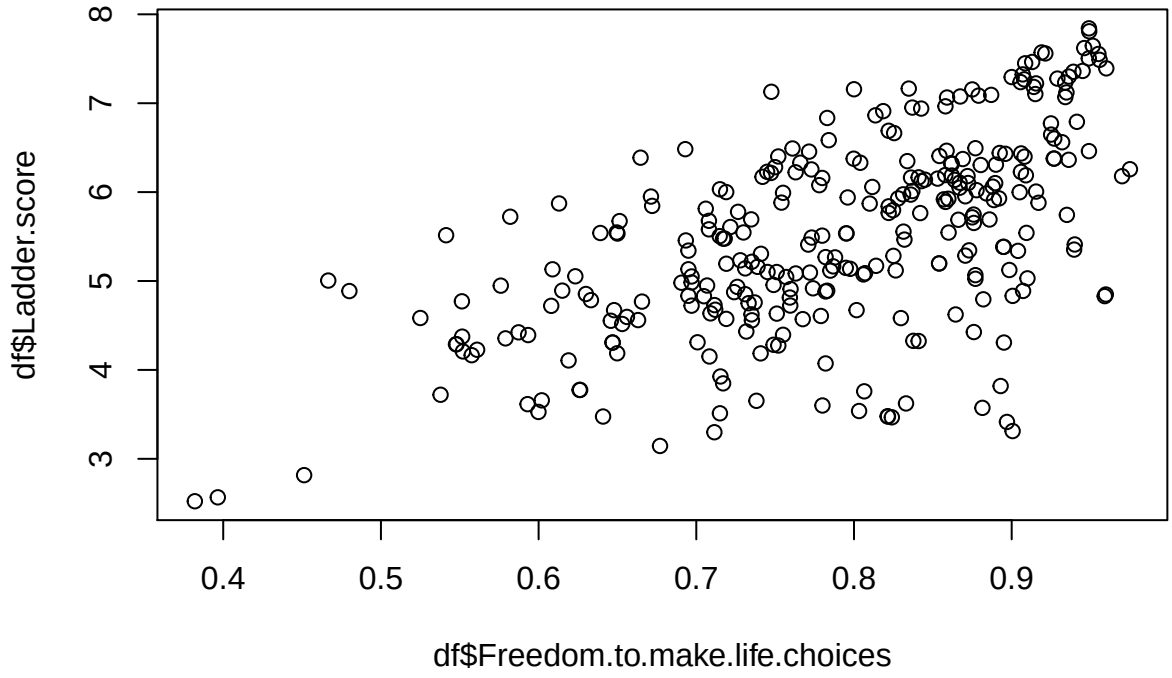

```
plot(x = df$Healthy.life.expectancy, y = df$Ladder.score)
```



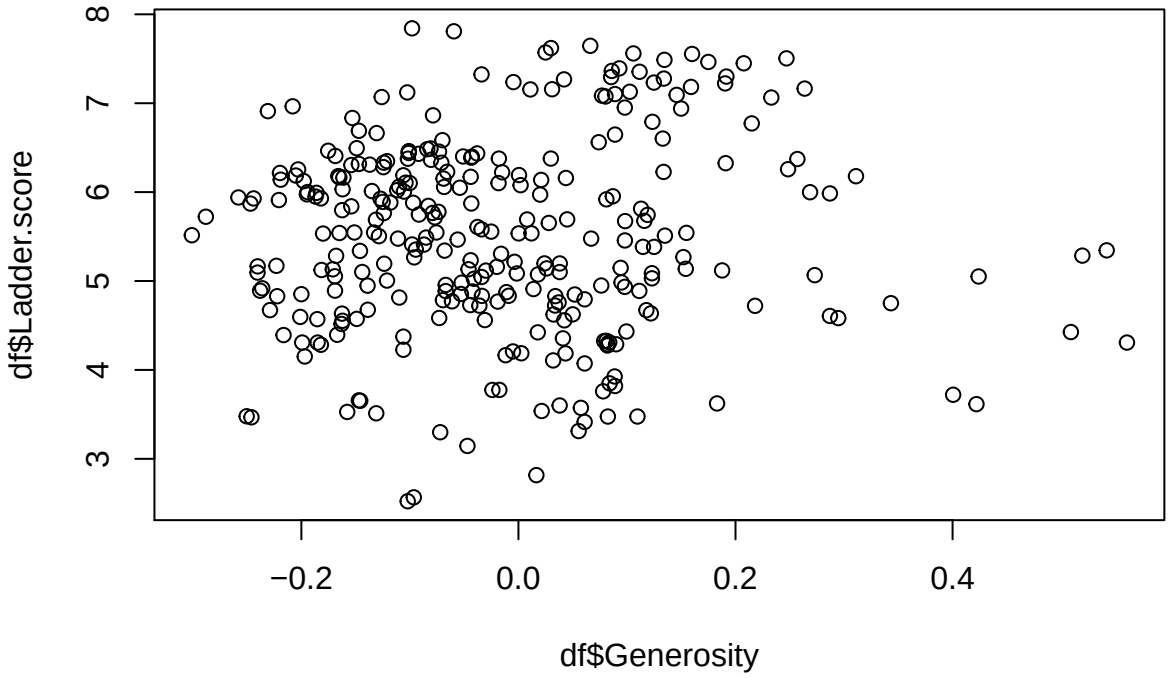

```
plot(x = df$Logged.GDP.Per.capita, y = df$Ladder.score)
```



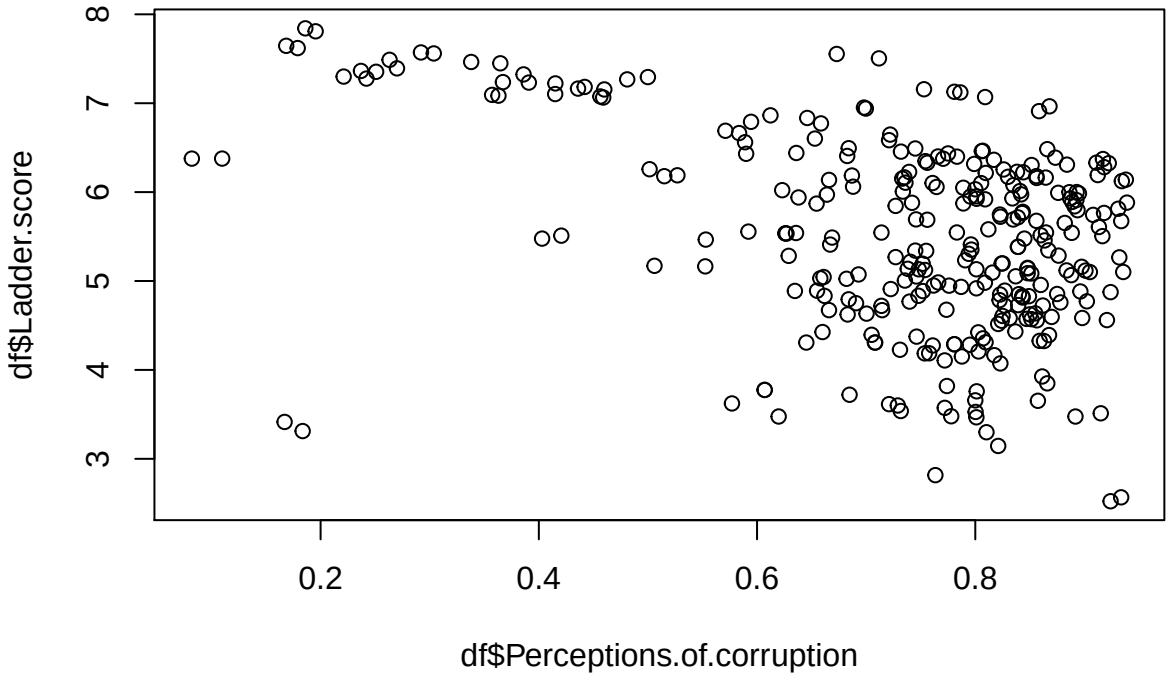

```
plot(x = df$Freedom.to.make.life.choices, y = df$Ladder.score)
```




```
plot(x = df$Generosity, y = df$Ladder.score)
```




```
plot(x = df$Perceptions.of.corruption, y = df$Ladder.score)
```



Multiple Linear Regression

```
df$Ladder.score = as.numeric(df$Ladder.score)
str(df)
```

```
## 'data.frame':    302 obs. of  7 variables:
##  $ Ladder.score                : num  7.81 7.65 7.56 7.5 7.49 ...
##  $ Logged.GDP.Per.capita       : num  10.6 10.8 11 10.8 11.1 ...
##  $ Social.support              : num  0.954 0.956 0.943 0.975 0.952 ...
##  $ Healthy.life.expectancy     : num  71.9 72.4 74.1 73 73.2 ...
##  $ Freedom.to.make.life.choices: num  0.949 0.951 0.921 0.949 0.956 ...
##  $ Generosity                  : num  -0.0595 0.0662 0.1059 0.2469 0.1345 ...
##  $ Perceptions.of.corruption   : num  0.195 0.168 0.304 0.712 0.263 ...
```

```
library(caTools)
set.seed(123)
split = sample.split(df$Ladder.score, SplitRatio = 0.8)
training_set = subset(df, split == TRUE)
test_set = subset(df, split == FALSE)
```

making the multiple regression model

```
regressor = lm(formula = Ladder.score ~ Logged.GDP.Per.capita + Social.support + Healthy.life.expectanc
               data = training_set)
```

```r
summary(regressor)
```

```
##
## Call:
## lm(formula = Ladder.score ~ Logged.GDP.Per.capita + Social.support +
##     Healthy.life.expectancy + Freedom.to.make.life.choices +
##     Generosity + Perceptions.of.corruption, data = training_set)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.73710 -0.29520  0.08116  0.34831  1.43395
##
## Coefficients:
##                               Estimate Std. Error t value Pr(>|t|)
## (Intercept)                   -1.72101    0.49513  -3.476 0.000607 ***
## Logged.GDP.Per.capita          0.20939    0.06747   3.103 0.002150 **
## Social.support                 2.51410    0.51985   4.836 2.40e-06 ***
## Healthy.life.expectancy        0.04009    0.01049   3.821 0.000170 ***
## Freedom.to.make.life.choices   1.63880    0.40631   4.033 7.45e-05 ***
## Generosity                     0.42141    0.25873   1.629 0.104711
## Perceptions.of.corruption     -0.89517    0.24292  -3.685 0.000284 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.5561 on 234 degrees of freedom
## Multiple R-squared:  0.7471, Adjusted R-squared:  0.7406
## F-statistic: 115.2 on 6 and 234 DF,  p-value: < 2.2e-16
```

Beginning backwards elimination. this step removing Freedom.to.make.life.choices because it has the highest p value, and is larger than a sig level of .5

```r
regressor = lm(formula = Ladder.score ~ Logged.GDP.Per.capita + Social.support + Healthy.life.expectanc
               data = training_set)
```

```r
summary(regressor)
```

```
##
## Call:
## lm(formula = Ladder.score ~ Logged.GDP.Per.capita + Social.support +
##     Healthy.life.expectancy + Generosity + Perceptions.of.corruption,
##     data = training_set)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.55038 -0.32440  0.06997  0.35345  1.58615
##
## Coefficients:
##                             Estimate Std. Error t value Pr(>|t|)
## (Intercept)                 -1.00690    0.47718  -2.110  0.03591 *
## Logged.GDP.Per.capita        0.21011    0.06963   3.018  0.00283 **
## Social.support               3.05684    0.51818   5.899 1.27e-08 ***
## Healthy.life.expectancy      0.04480    0.01076   4.164 4.40e-05 ***
```

```
## Generosity                  0.71258    0.25640   2.779  0.00589 **
## Perceptions.of.corruption -1.13298    0.24319  -4.659 5.33e-06 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.5739 on 235 degrees of freedom
## Multiple R-squared:  0.7295, Adjusted R-squared:  0.7238
## F-statistic: 126.8 on 5 and 235 DF,  p-value: < 2.2e-16
```

more elimination. Taking away the variable perceptions.of.corruption, because it is now has the highest p value, and it is above a .5 sig level.

```
regressor = lm(formula = Ladder.score ~ Logged.GDP.Per.capita + Social.support + Healthy.life.expectanc
               data = training_set)
```

```
summary(regressor)
```

```
##
## Call:
## lm(formula = Ladder.score ~ Logged.GDP.Per.capita + Social.support +
##     Healthy.life.expectancy + Generosity, data = training_set)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.52552 -0.37844  0.03287  0.42871  1.63594
##
## Coefficients:
##                          Estimate Std. Error t value Pr(>|t|)
## (Intercept)              -2.52613    0.36332  -6.953 3.48e-11 ***
## Logged.GDP.Per.capita     0.25220    0.07201   3.502 0.000552 ***
## Social.support            2.76481    0.53647   5.154 5.40e-07 ***
## Healthy.life.expectancy   0.05300    0.01107   4.788 2.98e-06 ***
## Generosity                1.04602    0.25678   4.074 6.32e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.5985 on 236 degrees of freedom
## Multiple R-squared:  0.7045, Adjusted R-squared:  0.6995
## F-statistic: 140.7 on 4 and 236 DF,  p-value: < 2.2e-16
```

more elimination. taking away generosity because it now has the highest p value and is greater than sig value of .5

```
regressor = lm(formula = Ladder.score ~ Logged.GDP.Per.capita + Social.support + Healthy.life.expectanc
               data = training_set)
```

```
summary(regressor)
```

```
##
## Call:
## lm(formula = Ladder.score ~ Logged.GDP.Per.capita + Social.support +
```

```
##     Healthy.life.expectancy + Perceptions.of.corruption, data = training_set)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.69451 -0.36611  0.08114  0.34306  1.58812
##
## Coefficients:
##                            Estimate Std. Error t value Pr(>|t|)
## (Intercept)                -0.63288    0.46428  -1.363   0.1741
## Logged.GDP.Per.capita       0.17952    0.06973   2.575   0.0106 *
## Social.support              3.23877    0.52130   6.213 2.33e-09 ***
## Healthy.life.expectancy     0.04315    0.01089   3.961 9.89e-05 ***
## Perceptions.of.corruption -1.32165    0.23683  -5.581 6.56e-08 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.582 on 236 degrees of freedom
## Multiple R-squared:  0.7206, Adjusted R-squared:  0.7159
## F-statistic: 152.2 on 4 and 236 DF,  p-value: < 2.2e-16
```

more elimination. getting rid of Healthy.life.expectancy for the same reasons as before

```
regressor = lm(formula = Ladder.score ~ Logged.GDP.Per.capita + Social.support + Perceptions.of.corrupti
               data = training_set)
```

```
summary(regressor)
```

```
##
## Call:
## lm(formula = Ladder.score ~ Logged.GDP.Per.capita + Social.support +
##     Perceptions.of.corruption, data = training_set)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -1.97951 -0.41013  0.09092  0.39426  1.59484
##
## Coefficients:
##                            Estimate Std. Error t value Pr(>|t|)
## (Intercept)                 0.23507    0.42182   0.557    0.578
## Logged.GDP.Per.capita       0.35833    0.05476   6.543 3.68e-10 ***
## Social.support              3.67484    0.52510   6.998 2.64e-11 ***
## Perceptions.of.corruption -1.46675    0.24112  -6.083 4.69e-09 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.5997 on 237 degrees of freedom
## Multiple R-squared:  0.7021, Adjusted R-squared:  0.6983
## F-statistic: 186.1 on 3 and 237 DF,  p-value: < 2.2e-16
```

more elimination. getting rid of perceptions. of.corruption

```
regressor = lm(formula = Ladder.score ~ Logged.GDP.Per.capita + Social.support,
               data = training_set)
```

```
summary(regressor)
```

```
##
## Call:
## lm(formula = Ladder.score ~ Logged.GDP.Per.capita + Social.support,
##     data = training_set)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2.11136 -0.46455  0.01637  0.48901  1.66947
##
## Coefficients:
##                       Estimate Std. Error t value Pr(>|t|)
## (Intercept)           -1.48340    0.33610  -4.414 1.54e-05 ***
## Logged.GDP.Per.capita  0.44258    0.05685   7.785 2.13e-13 ***
## Social.support         3.48528    0.56242   6.197 2.52e-09 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.6435 on 238 degrees of freedom
## Multiple R-squared:  0.6555, Adjusted R-squared:  0.6526
## F-statistic: 226.5 on 2 and 238 DF,  p-value: < 2.2e-16
```

more elimination. getting rid of Social.support

```
regressor = lm(formula = Ladder.score ~ Logged.GDP.Per.capita,
               data = training_set)
```

```
summary(regressor)
```

```
##
## Call:
## lm(formula = Ladder.score ~ Logged.GDP.Per.capita, data = training_set)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2.32256 -0.49248  0.05879  0.55666  1.42085
##
## Coefficients:
##                       Estimate Std. Error t value Pr(>|t|)
## (Intercept)           -1.24108    0.35899  -3.457 0.000646 ***
## Logged.GDP.Per.capita  0.71873    0.03796  18.932  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.692 on 239 degrees of freedom
## Multiple R-squared:    0.6,  Adjusted R-squared:  0.5983
## F-statistic: 358.4 on 1 and 239 DF,  p-value: < 2.2e-16
```

predicting test set results.

```
y_pred = predict(regressor, newdata = test_set)
```