Gender Parity Machine Learning Model

Erika Di Donato

20/12/2020

Contents

1 Introduction

Gender parity, which is critical for the development and prosperity of all economies, refers to the proportional representation of men and women in society.

The World Economic Forum (WEF) publishes an annual Gender Gap Report assigning an index and rank to each country. These metrics are meant to measure relative gaps in four key areas: health, education, economics and politics.

As an annual report, it can guage progress and create global awareness of the challenges that emerge from those gender gaps.

According to the World Economic Forum Global Gender Gap Report of 2020, gender parity will not be attained for another 99.5 years.

1.1 Goal of the Project

The goal of this project is to create a machine learning model combining the WEF Gender Gap Data with other global indicators to predict the Gender Gap Score and better reveal why gender dispartities still exist.

By taking a more granular approach, we should be able to further analyze contributing factors to this imbalance and hightlight opportunities for investment towards narrowing the gap.

1.2 Dataset and Variables

1.2.1 Global_Gender_Gap_2013

Our first dataset to consider is the WEF Global Gender Gap Report of 2013. It contains 136 observations with the following 12 variables:

- Country (136 Countries Represented)
- ISO3
- Overall Rank
- Overall Score
- Economic Participation and Opportunity Rank
- Economic Participation and Opportunity Score
- Educational Attainment Rank
- Educational Attainment Score
- Health and Survival Rank
- Health and Survival Score
- Political Empowerment Rank
- Political Empowerment Score

1.2.2 Population_Indicators

The Population_Indicators dataset consists of 4984 observations on Population annual rate of increase (percent), Total fertility rate (children per women), Infant mortality for both sexes (per 1,000 live births), Maternal mortality ratio (deaths per 100,000 population), Life expectancy at birth for both sexes (years), Life expectancy at birth for males (years), Life expectancy at birth for females (years), between 2010 and 2020 with the following 7 variables:

- Region/Country/Area (numeric identifier)
- X (Name of Region/Country/Area)
- Year
- Series
- Value
- Footnotes (Data refers to a 5-year period preceding the reference year.)
- Source

1.2.3 Ratio_of_girls_to_boys_in_school

This dataset reports 2921 observations on the Ratio of girls to boys in primary, secondary, and tertiary education between 1995 and 2018 across the world with the following 7 variables:

- Region/Country/Area (numeric identifier)
- X (Name of Region/Country/Area)
- Year
- Series
- Value
- Footnotes
- Source

1.2.4 Seats_held_by_women_in_parliament

This dataset contains national and regional data on Women in National Parliament between 2010 and 2020 with 1959 obeservations and the following 9 variables:

- Region/Country/Area (numeric identifier)
- X (Name of Region/Country/Area)
- Year
- Series
- Last Election Date
- Last Election Date footnote
- Value
- Footnotes
- Source

1.3 Key Steps

Our datasets were downloaded, combined and divided into the following subsets:

- Training Set 58.8%
- Testing Set 20.6% (used for assessing training models)
- Validation Set 20.6% (final hold-out test set)

These ratios were chosen to:

- Ensure the distributions in the test and validation sets are similar to the training set
- Mitigate the inherent risk with smaller datasets of overfitting our model

Once our dataset was split, we cleaned, transformed and ran imputations for missing data.

We then performed some data exploration and created visualizations to gain further insights into the data.

In this exploration we checked for outliers, multicollinearity, and heteroskedasticity.

These steps led us to a modelling structure in which we trained our data on the 58.6% of data and tested on 20.6%.

Models included:

- Naive (as a baseline) $y_i = \mu + \epsilon_i$
- Linear Regression $y = \mu + b_1 + b_2 + ... + b_n + \epsilon_i$
- Recurssive Partitioning Model (rpart)
- Random Forest Model

Once we identified our best-performing model, we tuned it and validated the results on our final 20.6% of data.

These results and model performance were evaluated using the RMSE (root-mean-square error) which measures the error of a model in predicting quantitative data.

$$RMSE(g) = \sqrt{\frac{1}{n} \sum_{i=1..n} (y_i - g(x_i))^2}$$

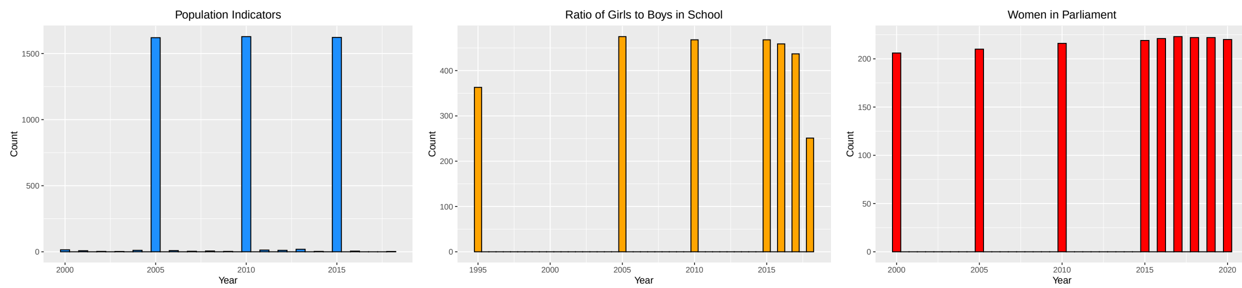
2 Methods/Analysis

Our process begins with the file download, environment setup, and dataset creation.

We wanted to create our own combined dataset from the 3 separate UN files to later join with the Global Gender Gap File.

On inspection, we observe there are reporting gaps in the UN files so we use the average of the years 2010-2015, before combining them to our Gender Gap file. This time period was chosen to reflect the relative socio-political climate of 2013 and recognizes some of these features could be either leading or lagging indicators.

Also note, we have created a unique dataset in this setup.



2.1 Process

Now that we have one combined file (GGG_2013), we split it into our training, test, and validation sets.

```
# Validation set will be 20.6% of GGG_2013 data (p = .2 achieves this due to rounding)
set.seed(3, sample.kind = "Rounding") # if using R 3.5 or earlier, use `set.seed(3)`
val_index <- createDataPartition(
  y = GGG_2013$Overall.Score, times = 1, p = 0.2, list = FALSE)
training_master <- GGG_2013[-val_index,]
temp <- GGG_2013[val_index,]

# Make sure Overall.Score in validation set are also in GGG_2013 set
validation <- temp %>%
      semi_join(GGG_2013, by = "Overall.Score")

# Add rows removed from validation set back into GGG_2013 set
removed <- anti_join(temp, validation)
GGG_2013 <- rbind(GGG_2013, removed)

rm(val_index, temp, removed)

# Test set will be 20.6% of GGG_2013 data data (p = 0.25 of training_master achieves this)
set.seed(5, sample.kind = "Rounding") # if using R 3.5 or earlier, use `set.seed(5)`
test_index <- createDataPartition(
  y = training_master$Overall.Score, times = 1, p = 0.25, list = FALSE)
```

```r
train <- training_master[-test_index,]
temp <- training_master[test_index,]

# Make sure Overall.Score in validation set are also in training_master set
test <- temp %>%
      semi_join(training_master, by = "Overall.Score")

# Add rows removed from vtest set back into training_master set
removed <- anti_join(temp, test)
training_master <- rbind(training_master, removed)

rm(test_index, temp, removed, GGG_2013, training_master)
```

2.1.1 Data Cleaning

Let's observe the train set.

```r
# Summary statistics
dim(train)
```

```
## [1] 80 23
```

```r
names(train)
```

```
##  [1] "Country"
##  [2] "ISO3"
##  [3] "Overall.Rank"
##  [4] "Overall.Score"
##  [5] "Economic.Participation.and.Opportunity.Rank"
##  [6] "Economic.Participation.and.Opportunity.Score"
##  [7] "Educational.Attainment.Rank"
##  [8] "Educational.Attainment.Score"
##  [9] "Health.and.Survival.Rank"
## [10] "Health.and.Survival.Score"
## [11] "Political.Empowerment.Rank"
## [12] "Political.Empowerment.Score"
## [13] "Infant mortality for both sexes (per 1,000 live births)"
## [14] "Life expectancy at birth for both sexes (years)"
## [15] "Life expectancy at birth for females (years)"
## [16] "Life expectancy at birth for males (years)"
## [17] "Maternal mortality ratio (deaths per 100,000 population)"
## [18] "Population annual rate of increase (percent)"
## [19] "Total fertility rate (children per women)"
## [20] "Ratio of girls to boys in primary education"
## [21] "Ratio of girls to boys in secondary education"
## [22] "Ratio of girls to boys in tertiary education"
## [23] "Seats held by women in national parliament, as of February (%)"
```

```r
anyNA(train)
```

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

Our train set has 80 observations with 23 variables. Rank can be expressed as an ordered score and the ISO3 column doesn't give us any new information so we can remove those columns.

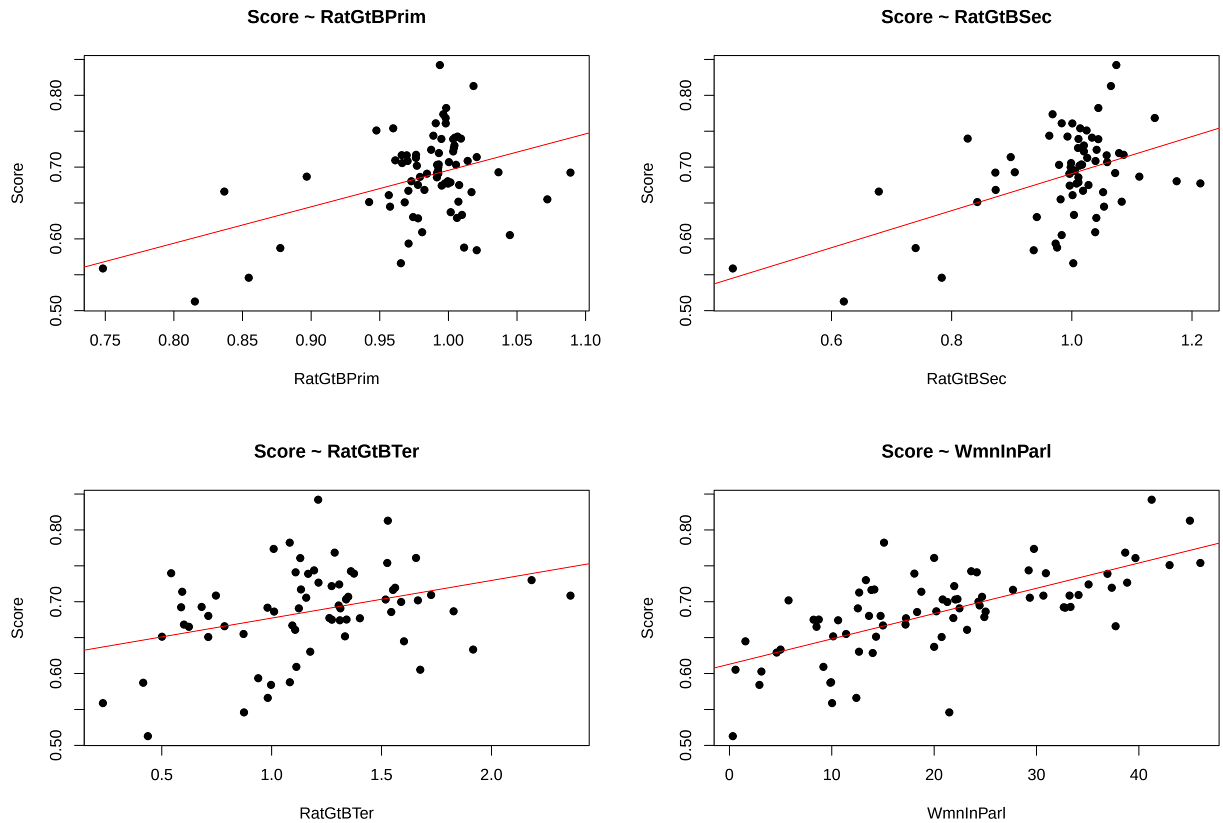
```
##       Country    Overall.Score    Economic.Participation.and.Opportunity.Score
##  Angola*   : 1   Min.   :0.5128   Min.   :0.2508
```

```
##  Argentina: 1   1st Qu.:0.6517   1st Qu.:0.5648
##  Austria  : 1   Median :0.6913   Median :0.6660
##  Bahamas  : 1   Mean   :0.6838   Mean   :0.6384
##  Bahrain  : 1   3rd Qu.:0.7177   3rd Qu.:0.7284
##  Barbados : 1   Max.   :0.8421   Max.   :0.8307
##  (Other)  :74
##  Educational.Attainment.Score Health.and.Survival.Score
##  Min.   :0.5311               Min.   :0.9312
##  1st Qu.:0.9503               1st Qu.:0.9694
##  Median :0.9917               Median :0.9754
##  Mean   :0.9518               Mean   :0.9721
##  3rd Qu.:0.9979               3rd Qu.:0.9793
##  Max.   :1.0000               Max.   :0.9796
##
##  Political.Empowerment.Score
##  Min.   :0.0099
##  1st Qu.:0.0769
##  Median :0.1432
##  Mean   :0.1729
##  3rd Qu.:0.2672
##  Max.   :0.6162
##
##  Infant mortality for both sexes (per 1,000 live births)
##  Min.   : 2.110
##  1st Qu.: 6.231
##  Median :15.623
##  Mean   :22.704
##  3rd Qu.:32.597
##  Max.   :91.228
##
##  Life expectancy at birth for both sexes (years)
##  Min.   :50.89
##  1st Qu.:67.92
##  Median :74.18
##  Mean   :71.69
##  3rd Qu.:76.99
##  Max.   :82.24
##
##  Life expectancy at birth for females (years)
##  Min.   :52.19
##  1st Qu.:70.33
##  Median :76.75
##  Mean   :74.26
##  3rd Qu.:80.20
##  Max.   :84.83
##
##  Life expectancy at birth for males (years)
##  Min.   :49.60
##  1st Qu.:65.11
##  Median :71.30
##  Mean   :69.16
##  3rd Qu.:75.35
##  Max.   :79.89
##
```

```
##  Maternal mortality ratio (deaths per 100,000 population)
##  Min.   :  2.918
##  1st Qu.: 12.164
##  Median : 35.205
##  Mean   :127.547
##  3rd Qu.:142.987
##  Max.   :946.208
##
##  Population annual rate of increase (percent)
##  Min.   :-1.1985
##  1st Qu.: 0.5406
##  Median : 1.2752
##  Mean   : 1.5558
##  3rd Qu.: 2.3666
##  Max.   : 7.0255
##
##  Total fertility rate (children per women)
##  Min.   :1.245
##  1st Qu.:1.783
##  Median :2.266
##  Mean   :2.718
##  3rd Qu.:3.028
##  Max.   :6.582
##
##  Ratio of girls to boys in primary education
##  Min.   :0.7483
##  1st Qu.:0.9710
##  Median :0.9926
##  Mean   :0.9813
##  3rd Qu.:1.0040
##  Max.   :1.0890
##  NA's   :3
##  Ratio of girls to boys in secondary education
##  Min.   :0.4357
##  1st Qu.:0.9779
##  Median :1.0106
##  Mean   :0.9853
##  3rd Qu.:1.0409
##  Max.   :1.2140
##  NA's   :12
##  Ratio of girls to boys in tertiary education
##  Min.   :0.2321
##  1st Qu.:0.9810
##  Median :1.1930
##  Mean   :1.1816
##  3rd Qu.:1.3746
##  Max.   :2.3596
##  NA's   :11
##  Seats held by women in national parliament, as of February (%)
##  Min.   : 0.3322
##  1st Qu.:12.5744
##  Median :20.4728
##  Mean   :20.7887
##  3rd Qu.:29.3196
```

```
##   Max.    :46.0079
##   NA's    :2
```

We will rename our columns, scale some of the observations, and create some scatterplots to see if there is a meaningful way to treat our NA's (missing data).



We observe a relationship between our Indicators and Score so when filling in the NA's, we use Stochastic Regression Imputation. This method uses the known data and adds a random error term to the predicted value. This reproduces our missing values better that just using Mean Imputation and minimizes overestimation of the correlation between values that can happen with Deterministic Regression Imputation.

```
# Stochastic regression imputation
imp <- mice(train, method = "norm", m = 1) # Impute data
```

```
##
##   iter imp variable
##    1    1  RatGtBPrim*  RatGtBSec*  RatGtBTer*  WmnInParl*
##    2    1  RatGtBPrim*  RatGtBSec*  RatGtBTer*  WmnInParl*
##    3    1  RatGtBPrim*  RatGtBSec*  RatGtBTer*  WmnInParl*
##    4    1  RatGtBPrim*  RatGtBSec*  RatGtBTer*  WmnInParl*
##    5    1  RatGtBPrim*  RatGtBSec*  RatGtBTer*  WmnInParl*
```

```
train <- complete(imp) # Store data

# Check for any missing values
anyNA(train)
```

```
## [1] FALSE
```

2.1.2 Data Exploration and Visualizations

We now have a complete training dataset we can explore further.

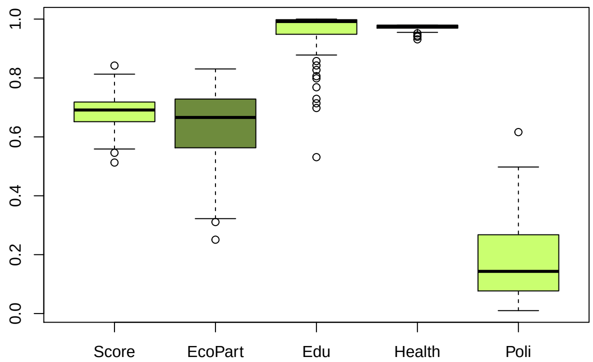
```r
# Review new table statistics
summary(train)
```

```
##       Country       Score           EcoPart             Edu        
##  Angola*  : 1   Min.   :0.5128   Min.   :0.2508   Min.   :0.5311  
##  Argentina: 1   1st Qu.:0.6517   1st Qu.:0.5648   1st Qu.:0.9503  
##  Austria  : 1   Median :0.6913   Median :0.6660   Median :0.9917  
##  Bahamas  : 1   Mean   :0.6838   Mean   :0.6384   Mean   :0.9518  
##  Bahrain  : 1   3rd Qu.:0.7177   3rd Qu.:0.7284   3rd Qu.:0.9979  
##  Barbados : 1   Max.   :0.8421   Max.   :0.8307   Max.   :1.0000  
##  (Other)  :74                                                      
##      Health            Poli           InfMortRt        LifExpBoth   
##  Min.   :0.9312   Min.   :0.0099   Min.   : 2.110   Min.   :50.89  
##  1st Qu.:0.9694   1st Qu.:0.0769   1st Qu.: 6.231   1st Qu.:67.92  
##  Median :0.9754   Median :0.1432   Median :15.623   Median :74.18  
##  Mean   :0.9721   Mean   :0.1729   Mean   :22.704   Mean   :71.69  
##  3rd Qu.:0.9793   3rd Qu.:0.2672   3rd Qu.:32.597   3rd Qu.:76.99  
##  Max.   :0.9796   Max.   :0.6162   Max.   :91.228   Max.   :82.24  
##                                                                    
##    LifExpFem       LifExpMal       MatMortRt          PopIncRt      
##  Min.   :52.19   Min.   :49.60   Min.   : 0.2918   Min.   :-1.1985  
##  1st Qu.:70.33   1st Qu.:65.11   1st Qu.: 1.2164   1st Qu.: 0.5406  
##  Median :76.75   Median :71.30   Median : 3.5205   Median : 1.2752  
##  Mean   :74.26   Mean   :69.16   Mean   :12.7547   Mean   : 1.5558  
##  3rd Qu.:80.20   3rd Qu.:75.35   3rd Qu.:14.2987   3rd Qu.: 2.3666  
##  Max.   :84.83   Max.   :79.89   Max.   :94.6208   Max.   : 7.0255  
##                                                                     
##    TotFertRt       RatGtBPrim       RatGtBSec        RatGtBTer     
##  Min.   :1.245   Min.   :0.7483   Min.   :0.4357   Min.   :0.2321  
##  1st Qu.:1.783   1st Qu.:0.9701   1st Qu.:0.8731   1st Qu.:0.6669  
##  Median :2.266   Median :0.9921   Median :1.0010   Median :1.1269  
##  Mean   :2.718   Mean   :0.9751   Mean   :0.9306   Mean   :1.0791  
##  3rd Qu.:3.028   3rd Qu.:1.0036   3rd Qu.:1.0349   3rd Qu.:1.3427  
##  Max.   :6.582   Max.   :1.0890   Max.   :1.2140   Max.   :2.3596  
##                                                                    
##    WmnInParl      
##  Min.   : 0.3322  
##  1st Qu.:12.1495  
##  Median :20.1070  
##  Mean   :20.2772  
##  3rd Qu.:29.2632  
##  Max.   :46.0079  
## 
```

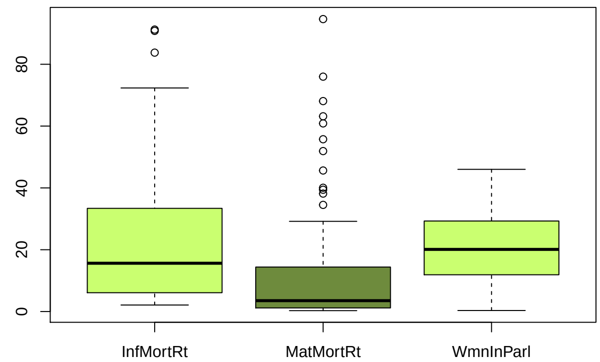
2.1.2.1 Outliers

The following boxplots describe some of the variation in our data. We clearly have some outliers. Particularly in Chad with a very high Maternal Mortality rate of 94.6208 per 10,000 population (scaled).

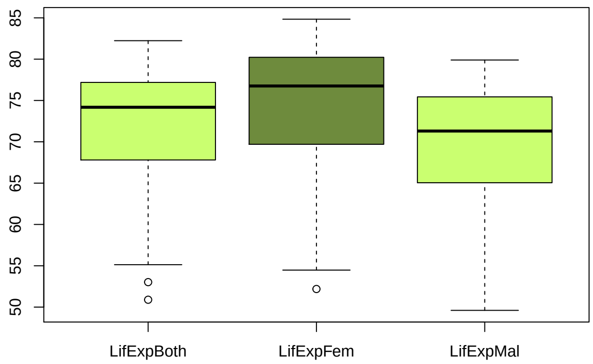
As a rule, outliers should only be removed if it suspected there are some errors with collecting, reporting, or processing the data. We have already scaled this column while we were cleaning the data so we will move on despite some datapoints residing well outside the range of the other values for the sample.



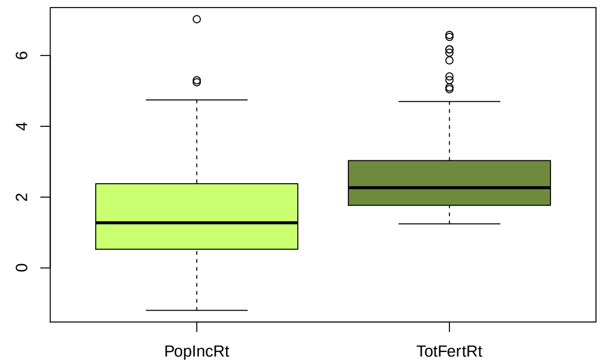
UN Data



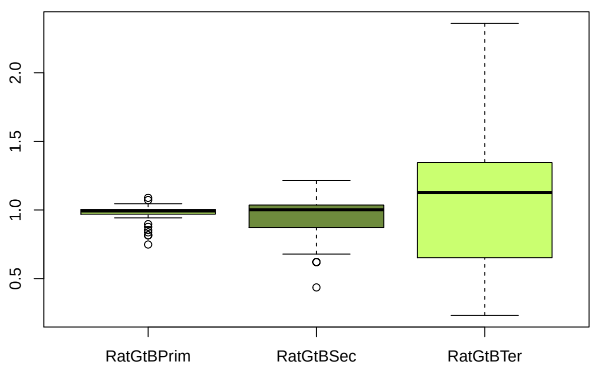
Mortality Rates and Political Power



Life Expectancy



Population Growth and Fertility Rates



Ratio of Girls to Boys in School

2.1.2.2 Test for Multicollinearity

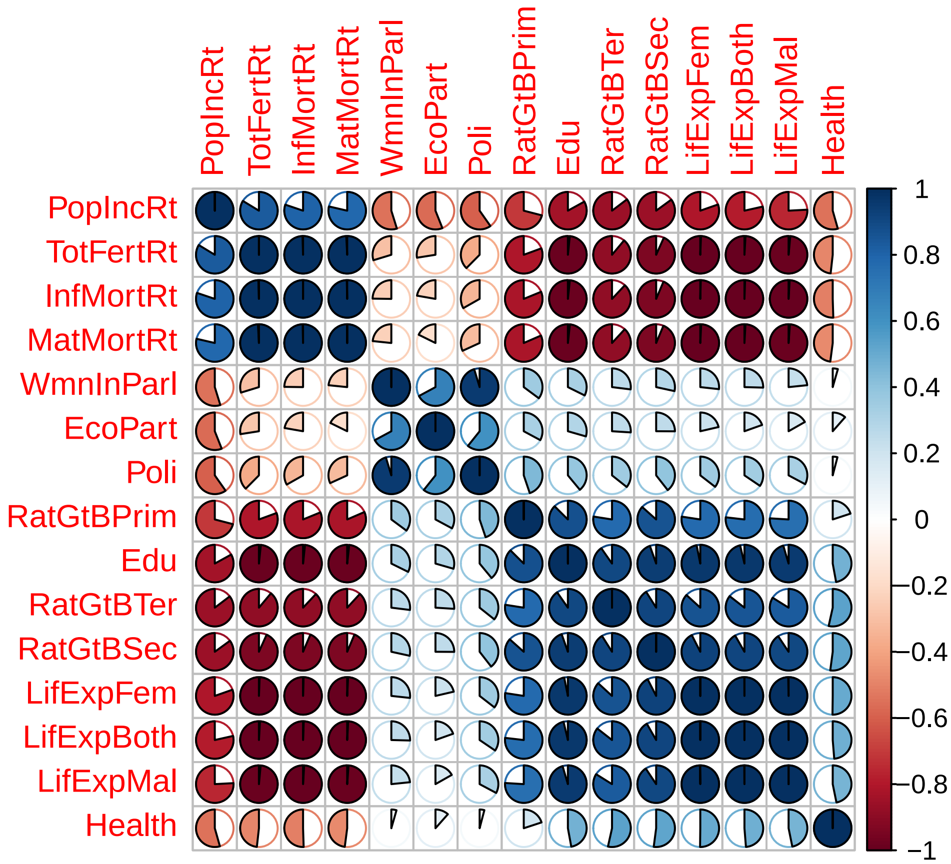
First we examine the correlation in our data to check for multicollinearity.

```
# Create correlation matrix: cordata
cordata = train[,c(3:17)]
corr <- round(cor(cordata), 3)
corr
```

```
##            EcoPart   Edu Health  Poli InfMortRt LifExpBoth LifExpFem
## EcoPart      1.000 0.183  0.078 0.332    -0.092      0.059     0.079
## Edu          0.183 1.000  0.179 0.160    -0.843      0.709     0.740
## Health       0.078 0.179  1.000 0.004    -0.235      0.196     0.224
```

```
## Poli          0.332  0.160  0.004  1.000    -0.106      0.184      0.184
## InfMortRt    -0.092 -0.843 -0.235 -0.106     1.000     -0.924     -0.937
## LifExpBoth    0.059  0.709  0.196  0.184    -0.924      1.000      0.992
## LifExpFem     0.079  0.740  0.224  0.184    -0.937      0.992      1.000
## LifExpMal     0.024  0.665  0.159  0.175    -0.895      0.991      0.966
## MatMortRt     0.049 -0.816 -0.132 -0.047     0.891     -0.855     -0.868
## PopIncRt     -0.300 -0.378 -0.202 -0.259     0.394     -0.337     -0.398
## TotFertRt    -0.128 -0.795 -0.102 -0.107     0.907     -0.852     -0.870
## RatGtBPrim    0.190  0.583 -0.073  0.267    -0.451      0.321      0.325
## RatGtBSec     0.108  0.584  0.267  0.245    -0.584      0.566      0.585
## RatGtBTer     0.107  0.529  0.240  0.170    -0.469      0.427      0.460
## WmnInParl     0.407  0.222  0.044  0.782    -0.070      0.119      0.135
##            LifExpMal MatMortRt PopIncRt TotFertRt RatGtBPrim RatGtBSec
## EcoPart        0.024     0.049   -0.300    -0.128      0.190     0.108
## Edu            0.665    -0.816   -0.378    -0.795      0.583     0.584
## Health         0.159    -0.132   -0.202    -0.102     -0.073     0.267
## Poli           0.175    -0.047   -0.259    -0.107      0.267     0.245
## InfMortRt     -0.895     0.891    0.394     0.907     -0.451    -0.584
## LifExpBoth     0.991    -0.855   -0.337    -0.852      0.321     0.566
## LifExpFem      0.966    -0.868   -0.398    -0.870      0.325     0.585
## LifExpMal      1.000    -0.828   -0.254    -0.816      0.310     0.535
## MatMortRt     -0.828     1.000    0.380     0.885     -0.408    -0.559
## PopIncRt      -0.254     0.380    1.000     0.534     -0.174    -0.392
## TotFertRt     -0.816     0.885    0.534     1.000     -0.330    -0.505
## RatGtBPrim     0.310    -0.408   -0.174    -0.330      1.000     0.536
## RatGtBSec      0.535    -0.559   -0.392    -0.505      0.536     1.000
## RatGtBTer      0.382    -0.450   -0.392    -0.454      0.330     0.513
## WmnInParl      0.093    -0.049   -0.259    -0.092      0.185     0.130
##            RatGtBTer WmnInParl
## EcoPart        0.107     0.407
## Edu            0.529     0.222
## Health         0.240     0.044
## Poli           0.170     0.782
## InfMortRt     -0.469    -0.070
## LifExpBoth     0.427     0.119
## LifExpFem      0.460     0.135
## LifExpMal      0.382     0.093
## MatMortRt     -0.450    -0.049
## PopIncRt      -0.392    -0.259
## TotFertRt     -0.454    -0.092
## RatGtBPrim     0.330     0.185
## RatGtBSec      0.513     0.130
## RatGtBTer      1.000     0.111
## WmnInParl      0.111     1.000
```

We can also observe these relationships more closely in a Correlation Matrix



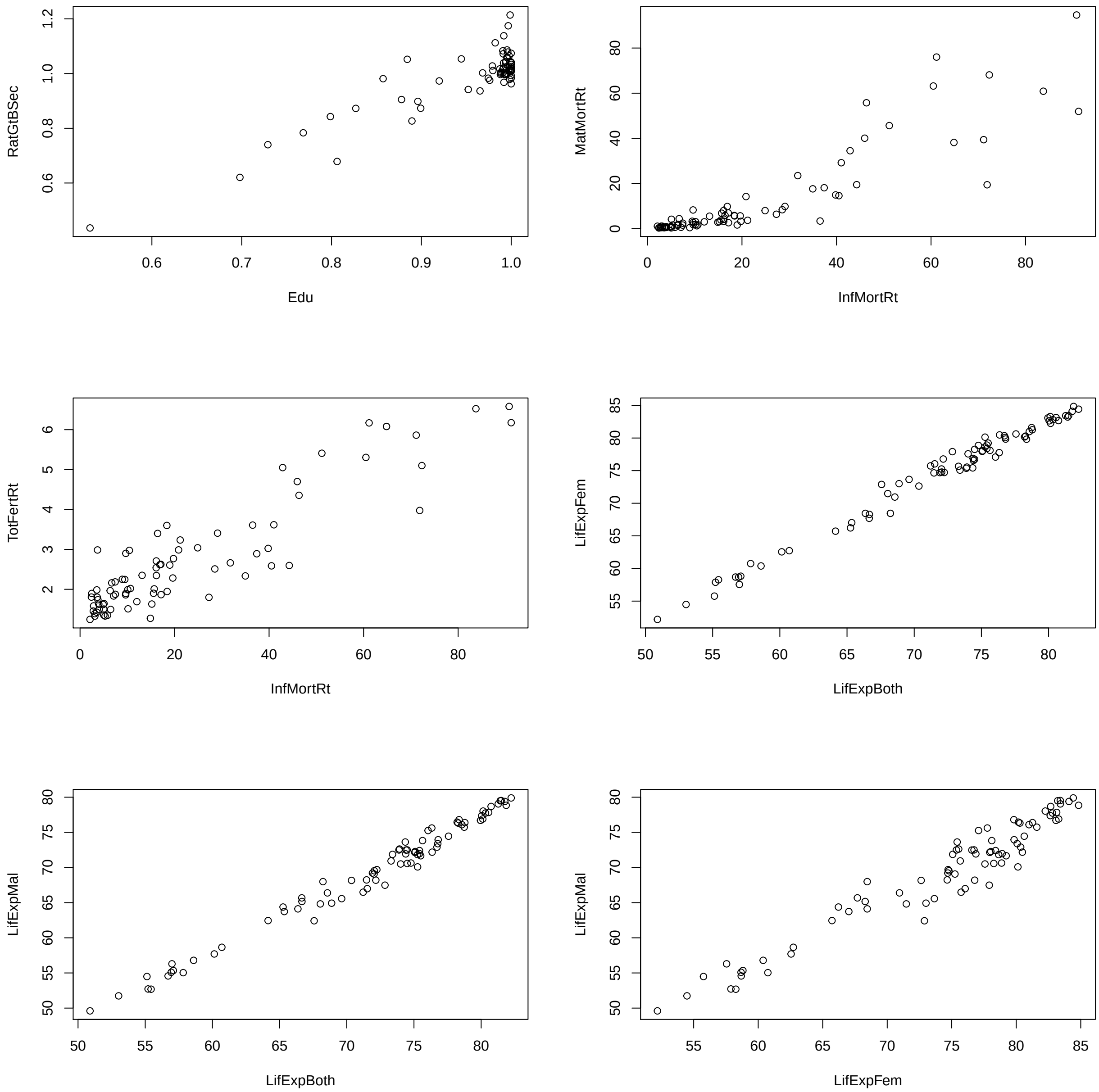
The output above shows the presence of strong positive correlations between the:

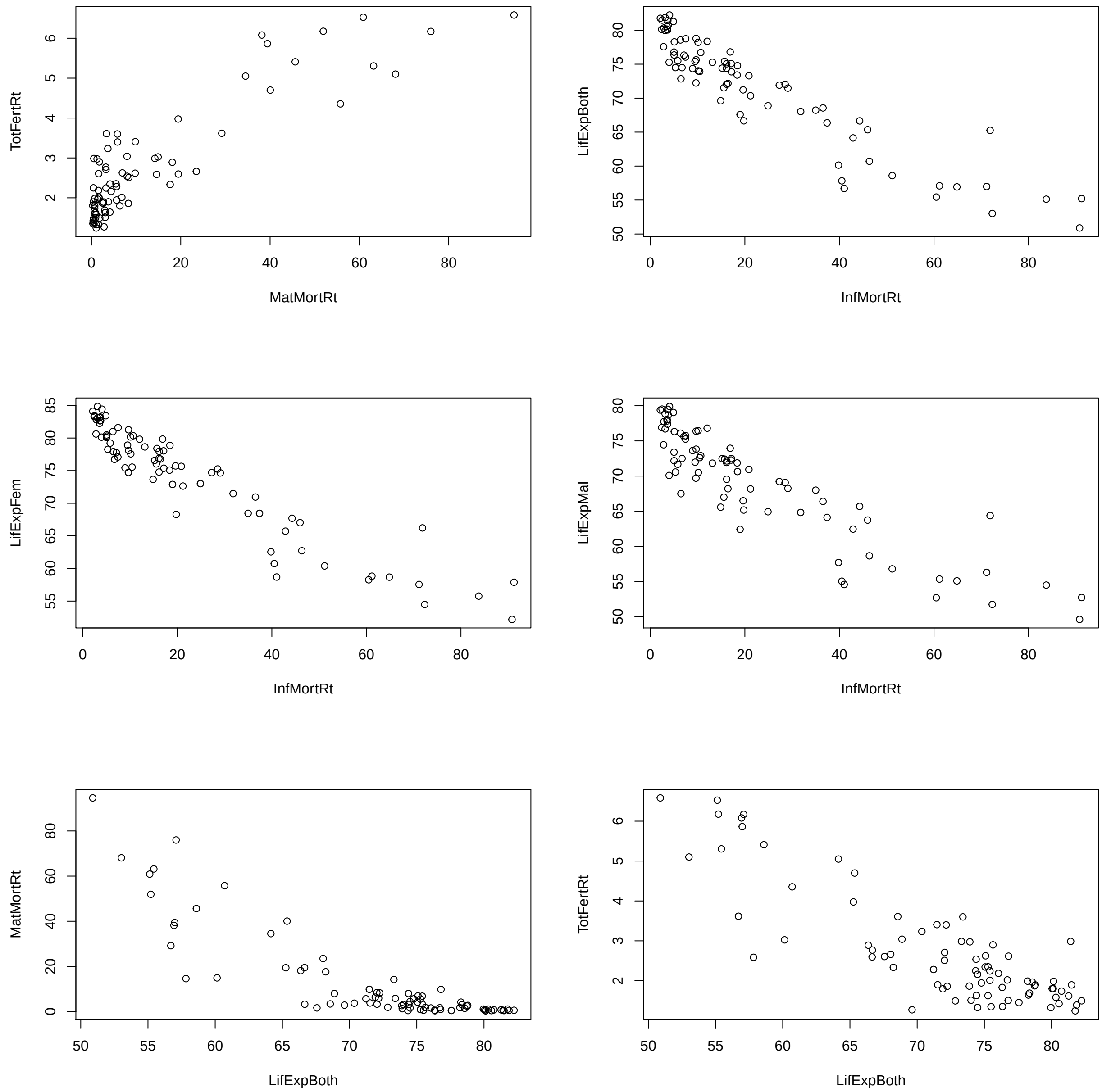
- Edu and RatGtBSec (positive)
- InfMortRt and MatMortRt (positive)
- InfMortRt and TotFertRt (positive))
- LifExpBoth and LifExpFem (positive)
- LifExpBoth and LifExpMal (positive)
- LifExpFem and LifExpMal (positive)
- MatMortRt and TotFertRt (positive)

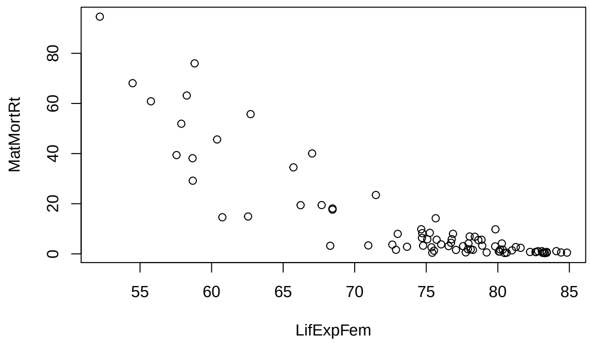
It also shows strong negative correlations between:

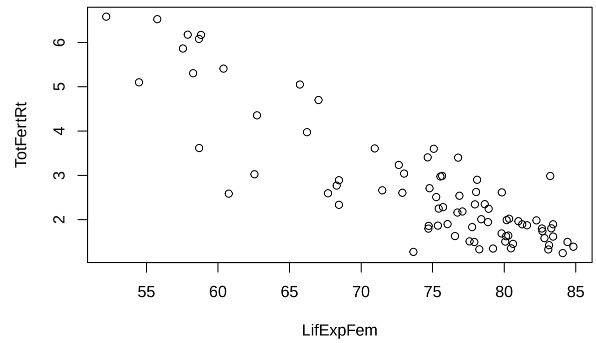
- InfMortRt and LifExpBoth (negative)
- InfMortRt and LifExpFem (negative)
- InfMortRt and LifExpMal (negative)
- LifExpBoth and MatMortRt (negative)
- LifExpBoth and TotFertRt (negative)
- LifExpFem and MatMortRt (negative)
- LifExpFem and TotFertRt (negative)

We'll plot the correlated data and keep these factors in mind when selecting features.


RatGtBSec
0.6
0.8
1.0
1.2
0.6
0.7
0.8
0.9
1.0
Edu
MatMortRt
0
20
40
60
80
0
20
40
60
80
InfMortRt
TotFertRt
2
3
4
5
6
0
20
40
60
80
InfMortRt
LifExpFem
55
60
65
70
75
80
85
50
55
60
65
70
75
80
LifExpBoth
LifExpMal
50
55
60
65
70
75
80
50
55
60
65
70
75
80
LifExpBoth
LifExpMal
50
55
60
65
70
75
80
55
60
65
70
75
80
85
LifExpFem


TotFertRt
MatMortRt
LifExpBoth
InfMortRt
LifExpFem
InfMortRt
LifExpMal
InfMortRt
MatMortRt
LifExpBoth
TotFertRt
LifExpBoth



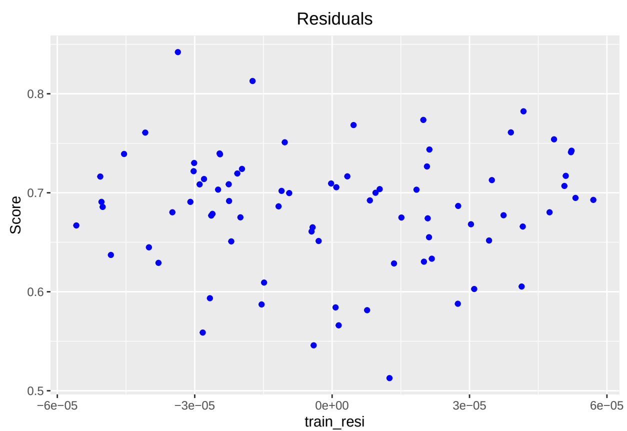


2.1.2.3 Test for Heteroskedasticity

Heteroskedasticity occurs when the variance for all observations in a data set are not the same. Because it is a violation of the ordinary least square assumption there are two main consequences on the least squares estimators:

- The least squares estimator is still a linear and unbiased estimator, but it is no longer best. That is, there is another estimator with a smaller variance.
- The standard errors computed for the least squares estimators are incorrect. This can affect confidence intervals and hypothesis testing that use those standard errors, which could lead to misleading conclusions.

We will test for heteroskedasticity by creating a residual plot of the least squares residuals against the explanatory variable. If there is an evident pattern in the plot, then heteroskedasticity is present.



There is no discernable pattern. We do not have to correct our data for heteroskedasticity.

2.1.3 Insights Gained

Some of the relationships uncovered in the dataset are not suprising. For example, the strong negative correlation between Maternal Mortality Ratio and Life Expectancy at Birth for Females (0.966). We can easily agree that a direct cause of death for women would shorten their life expectancy.

Other relationships were more interesting. Such as the high correlation between Total Fertility Rate with the Infant Mortality (0.907) and Maternal Mortatity (0.885) Rates. Could this be an indication that countries with more access to family planning also have better outcomes for infants and their mothers?

The absence of stronger correlations in other areas also poses some questions. For instance, Health and Survival Scores with Political Empowerment (0.004), and Seats held by women in national parliament (0.054). Is something other than political participation preventing women from enacting change that could make a real difference?

2.2 Modelling

In preparation for modelling, we

- Duplicate any transfortmations we performed in our training set on our test set
- Define RMSE as our evaluation criteria
- Define mu as the Score average

2.2.1 Naive Forecast

For the naive forecast, we simply set all forecasts to be the value of mean Score and assume all variation is due to random error.

$$y_i = \mu + \epsilon_i$$

```
# Naive Forecast Based on Mean Rating
RMSE_naive <- RMSE(test_noCountry$Score, mu)

# Save results in a tibble
RMSE_results = tibble(Method = "Naive Forecast", RMSE = RMSE_naive)
RMSE_results <- RMSE_results %>% mutate_if(is.numeric, ~ round(.,digits = 8))
# Return model RMSE results
RMSE_results
```

Method	RMSE
Naive Forecast	0.0518581

This is a very good result but the Naive Forecast really doesn't predict Country Scores very well. It just sets them all to the average without accounting for any variation in our data.

2.2.2 Linear Regression - Part 1 (All Variables)

Linear regression is a basic and commonly used type of predictive analysis in which we measure the magnitude of the effects of our predictors.

Our Regression Model is of the form:

$$y = \mu + b_1 + b_2 + ... + b_n + \epsilon_i$$

2.2.2.0.1 (All) Linear Regression Feature Selection

Once defining our Model to include all variables, we can make some observations about the feature coefficients.

Specifically, it appears that The Gender Gap Report weighs Economic Participation, Educational Attainment, Health and Survival, and Political Empowerment equally when determining the Overall Score.

Our supplementary features from the UN don't add much additional variation to this formula.

This is not suprising since the Score is calculated on the 4 WEF features.

```r
# Define the Linear Model
train_lin <- train_noCountry

linModel_formula = Score ~ .
linModel <- lm(linModel_formula, data = train_lin)

# Observe the coeffficients
LinCoefficient <- round(summary(linModel)$coefficients, 5)
LinCoefficient
```

```
##              Estimate Std. Error    t value Pr(>|t|)
## (Intercept) -0.00037    0.00057   -0.64830  0.51911
## EcoPart      0.24998    0.00004 6166.96854  0.00000
## Edu          0.24991    0.00012 2067.77552  0.00000
## Health       0.25016    0.00050  501.43222  0.00000
## Poli         0.25003    0.00006 4464.45929  0.00000
## InfMortRt    0.00000    0.00000    0.07451  0.94084
## LifExpBoth  -0.00001    0.00006   -0.12707  0.89928
## LifExpFem    0.00001    0.00003    0.34325  0.73254
## LifExpMal    0.00000    0.00003   -0.01359  0.98920
## MatMortRt    0.00000    0.00000    0.56783  0.57214
## PopIncRt     0.00000    0.00000   -0.08602  0.93172
## TotFertRt    0.00000    0.00001    0.33573  0.73817
## RatGtBPrim   0.00015    0.00011    1.35158  0.18127
## RatGtBSec   -0.00002    0.00004   -0.70147  0.48555
## RatGtBTer   -0.00001    0.00001   -1.01469  0.31407
## WmnInParl    0.00000    0.00000   -0.90523  0.36874
```

2.2.2.0.2 (All) Linear Regression Training

Once we reduce the dimensions to those with a p-VALUE < 0.05, we recognize this model doesn't offer any new information and would just returns the WEF score.

```r
# Define the Linear Model with Dimension Reduction
train_lin <- train_noCountry
linModel_formula = Score ~ EcoPart + Edu + Health + Poli
linModel <- lm(linModel_formula, data = train_lin)

# Observe the coeffficients
LinCoefficient <- round(summary(linModel)$coefficients, 5)
LinCoefficient
```

```
##              Estimate Std. Error    t value Pr(>|t|)
## (Intercept) -0.00021    0.00038   -0.55775  0.57868
## EcoPart      0.24999    0.00003 7962.95597  0.00000
## Edu          0.24999    0.00005 5460.85099  0.00000
## Health       0.25023    0.00039  634.18062  0.00000
## Poli         0.25001    0.00003 7637.79234  0.00000
```

2.2.2.0.3 (All) Linear Regression Forecast on the Test Set

Note: this is a Trivial Solution

```r
# Linear Model Forecast
predicted_Score <- predict(linModel, test_noCountry)
```

```r
RMSE_model_LinearT <- RMSE(predicted_Score, test_noCountry$Score)

# Save results in a tibble
RMSE_results <- bind_rows(RMSE_results,
                          tibble(Method = "Full Linear Model (Trivial Solution)",
                          RMSE = RMSE_model_LinearT))

# Return model RMSE results
RMSE_results <- RMSE_results %>% mutate_if(is.numeric, ~ round(.,digits = 8))
RMSE_results
```

Method	RMSE
Naive Forecast	0.0518581
Full Linear Model (Trivial Solution)	0.0000371

2.2.3 Linear Regression - Part 2 (UN Data)

What happens if we remove the WEF data and express our Scores as a function of only the UN data?

2.2.3.1 (UN Data) Linear Regression Feature Selection

By removing the WEF Variables we express Score as a function of only the UN data.

```r
# Define the new Linear Model
train_lin <- train_noCountry[, c(1, 6:16)]
linModel_formula = Score ~.
linModel <- lm(linModel_formula, data = train_lin)

# Observe the coeffficients
LinCoefficient <- round(summary(linModel)$coefficients, 5)
LinCoefficient
```

```
##             Estimate Std. Error  t value Pr(>|t|)
## (Intercept)  0.43075    0.17211  2.50276  0.01473
## InfMortRt   -0.00069    0.00068 -1.01262  0.31483
## LifExpBoth   0.13321    0.05043  2.64137  0.01024
## LifExpFem   -0.06544    0.02537 -2.57924  0.01207
## LifExpMal   -0.06797    0.02553 -2.66205  0.00968
## MatMortRt    0.00126    0.00048  2.61926  0.01086
## PopIncRt     0.00318    0.00447  0.71212  0.47883
## TotFertRt   -0.01603    0.00836 -1.91766  0.05936
## RatGtBPrim   0.20346    0.09070  2.24316  0.02815
## RatGtBSec    0.03598    0.03267  1.10121  0.27469
## RatGtBTer    0.00975    0.01007  0.96836  0.33629
## WmnInParl    0.00292    0.00035  8.22621  0.00000
```

2.2.3.2 (UN Data) Linear Regression Training

Once we reduce the dimensions to remove the highly correlated data we can redefine the model with our training set.

```r
# Define the new Linear Model with Dimension Reduction
train_lin <- train_noCountry[, c(1, 6:16)]
linModel_formula = Score ~ LifExpBoth + PopIncRt + RatGtBPrim + RatGtBTer
linModel <- lm(linModel_formula, data = test_noCountry)
```

```
# Observe the coeffficients
LinCoefficient <- round(summary(linModel)$coefficients, 5)
LinCoefficient
```

```
##             Estimate Std. Error  t value Pr(>|t|)
## (Intercept)  0.64878    0.23103  2.80817  0.00998
## LifExpBoth  -0.00103    0.00111 -0.92244  0.36588
## PopIncRt    -0.01286    0.01096 -1.17372  0.25252
## RatGtBPrim   0.04534    0.24163  0.18763  0.85281
## RatGtBTer    0.06715    0.02815  2.38562  0.02567
```

2.2.3.3 (UN Data) Linear Regression Forecast on the Test Set

We can now run the linear regression model on the test set.

```
# New Linear Model Forecast
predicted_Score <- predict(linModel, test_noCountry)

RMSE_model_LinearUN <- RMSE(predicted_Score, test_noCountry$Score)

# Save results in a tibble
RMSE_results <- bind_rows(RMSE_results,
                          tibble(Method = "Reduced Linear Model (UN Data)",
                          RMSE = RMSE_model_LinearUN))

# Return model RMSE results
RMSE_results <- RMSE_results %>% mutate_if(is.numeric, ~ round(.,digits = 8))
RMSE_results
```

Method	RMSE
Naive Forecast	0.0518581
Full Linear Model (Trivial Solution)	0.0000371
Reduced Linear Model (UN Data)	0.0411649

Our Reduced Linear Model performs better than the Naive Forecast.

2.2.4 Recursive Partitioning (rpart)

Through recursive partitioning we create a regression tree that will further explore our data for important features and make decisions based on that information to split our observations. All the split points are based on one variable at a time.

2.2.4.1 rpart Training

Managing multicollinearity through feature selection is redundant with the rpart model. In a standard recursive partitioning tree that considers all predictors, each split is based on the most powerful predictor available.

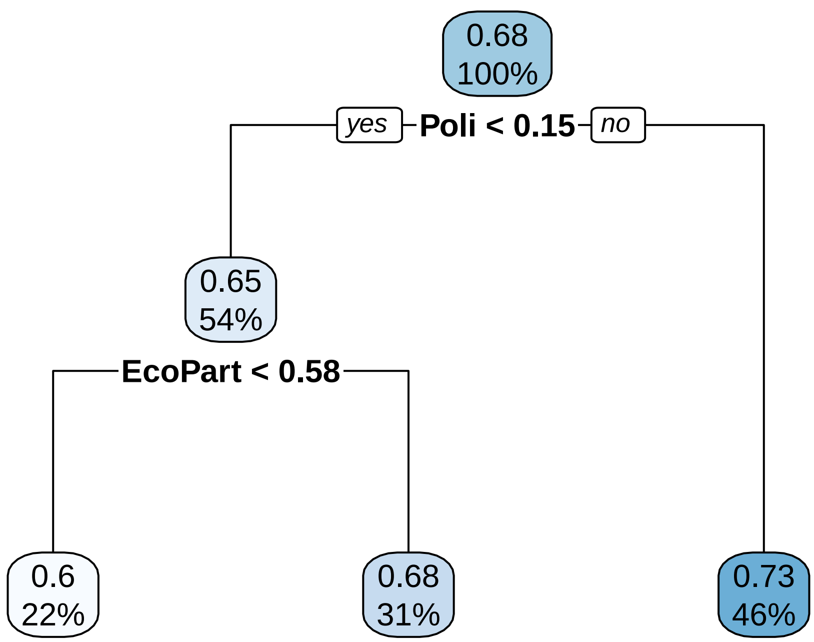
We construct the model will all variables as predictors.

```
# rpart Training
train_rpart <- train_noCountry
test_rpart <- test_noCountry
set.seed(100)
```

```
rpartMod <- train(Score ~ ., data = train_rpart, method = "rpart")
rpartImp <- varImp(rpartMod)
print(rpartImp)
```

```
## rpart variable importance
##
##              Overall
## EcoPart      100.00
## Poli          78.42
## WmnInParl     78.07
## TotFertRt     35.34
## PopIncRt      33.49
## Edu           32.96
## LifExpBoth    30.51
## Health         0.00
## MatMortRt      0.00
## RatGtBTer      0.00
## InfMortRt      0.00
## RatGtBSec      0.00
## LifExpMal      0.00
## LifExpFem      0.00
## RatGtBPrim     0.00
```

```
rpart.plot(rpartMod$finalModel)
```



2.2.4.2 rpart Forecast on the Test Set

We can now run the rpart model on the test set.

```
# rpart Model Forecast
predicted_Score <- predict(rpartMod$finalModel, test_rpart)

RMSE_model_rpart <- RMSE(predicted_Score, test_rpart$Score)

# Save results in a tibble
RMSE_results <- bind_rows(RMSE_results,
                          tibble(Method = "rpart Model",
                          RMSE = RMSE_model_rpart))

# Return model RMSE results
RMSE_results <- RMSE_results %>% mutate_if(is.numeric, ~ round(.,digits = 8))
RMSE_results
```

Method	RMSE
Naive Forecast	0.0518581
Full Linear Model (Trivial Solution)	0.0000371
Reduced Linear Model (UN Data)	0.0411649
rpart Model	0.0412000

While the Recursive Partitioning Model performed better than the Naive Model, it did not perform as well as our Reduced Linear Model (UN Data).

2.2.5 Random Forest

The next model to consider is the Random Forest Model. It is based on generating a large number of decision trees (default = 500), each constructed using a different subset of our training set.

2.2.5.1 Random Forest Training

While there could be traces of the impacts of collinearity in this model, the way it uses feature importance over a large number of decision trees will mitigate this risk.

Our training model is initially defined as using all the possible input variables (mtry = seq(1:15)) to create our decision trees and we can observe the RMSE-minimizing mtry from the model output.

```
# Define Random Forest Model
set.seed(7, sample.kind = "Rounding")

fit_rtree <- train(Score ~., data = train_noCountry, method = "rf",
                        tuneGrid = data.frame(mtry = seq(1:15)), importance = TRUE)
print(fit_rtree)
```

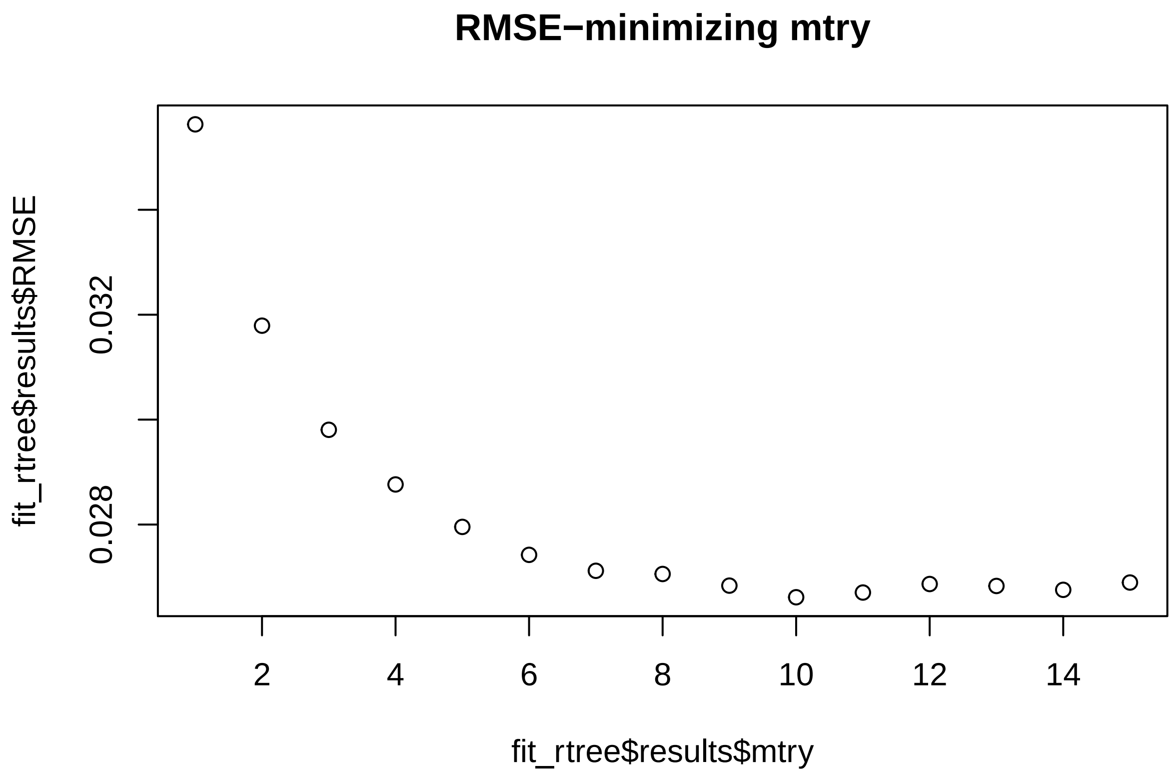
```
## Random Forest
##
## 80 samples
## 15 predictors
##
## No pre-processing
## Resampling: Bootstrapped (25 reps)
## Summary of sample sizes: 80, 80, 80, 80, 80, 80, ...
## Resampling results across tuning parameters:
##
```

```
##   mtry  RMSE        Rsquared   MAE
##    1    0.03562865  0.7889418  0.02689973
##    2    0.03179110  0.8273401  0.02341636
##    3    0.02980617  0.8460229  0.02167382
##    4    0.02876460  0.8516622  0.02079039
##    5    0.02795506  0.8565236  0.02006683
##    6    0.02742246  0.8583060  0.01958510
##    7    0.02711863  0.8611338  0.01935138
##    8    0.02705768  0.8572096  0.01923642
##    9    0.02683580  0.8565554  0.01907778
##   10    0.02661452  0.8579846  0.01892582
##   11    0.02670414  0.8547957  0.01896245
##   12    0.02686631  0.8509408  0.01912482
##   13    0.02682948  0.8494599  0.01903896
##   14    0.02675562  0.8493348  0.01896701
##   15    0.02689498  0.8467021  0.01900944
##
## RMSE was used to select the optimal model using the smallest value.
## The final value used for the model was mtry = 10.
```

```r
imp <- varImp(fit_rtree)
imp
```

```
## rf variable importance
##
##             Overall
## Poli        100.000
## EcoPart      92.565
## Edu          35.236
## WmnInParl    26.479
## TotFertRt    22.508
## InfMortRt    16.732
## LifExpMal    16.568
## PopIncRt     15.149
## LifExpBoth   13.721
## RatGtBPrim   13.393
## LifExpFem    12.275
## MatMortRt    10.414
## RatGtBSec     9.576
## RatGtBTer     1.760
## Health        0.000
```

```r
plot(fit_rtree$results$mtry, fit_rtree$results$RMSE, main = "RMSE-minimizing mtry")
```



2.2.5.2 Random Forest Tuning and Testing

Now that we have observed an optimal mtry = 10, we create the final predictive model and run it on our test set.

```
# Tune and Test Random Forest Model
fit_rtree2 <- train(Score ~., data = train_noCountry, method = "rf",
                        tuneGrid = data.frame(mtry = 10), importance = TRUE)

predicted_Score <- predict(fit_rtree2, test_noCountry)


RMSE_model_rtree2 <- RMSE(predicted_Score, test$Score)

# Save results in a tibble
RMSE_results <- bind_rows(RMSE_results,
                          tibble(Method = "Random Forest Model",
                          RMSE = RMSE_model_rtree2))

# Return model RMSE results
RMSE_results <- RMSE_results %>% mutate_if(is.numeric, ~ round(.,digits = 8))
RMSE_results
```

Method	RMSE
Naive Forecast	0.0518581
Full Linear Model (Trivial Solution)	0.0000371

Method	RMSE
Reduced Linear Model (UN Data)	0.0411649
rpart Model	0.0412000
Random Forest Model	0.0226823

2.3 Model Results

Our Random Forest Model is by far the best-performing model.

We can now prepare our validation set by performing the same operations as we did on the training data:

- Remove and renaming columns to match our training set
- Scale the Maternal Mortality Rate (divide by 10)
- Run stochastic regression imputation for NA's

```
##
##  iter imp variable
##   1   1  RatGtBSec*  RatGtBTer*  WmnInParl*
##   2   1  RatGtBSec*  RatGtBTer*  WmnInParl*
##   3   1  RatGtBSec*  RatGtBTer*  WmnInParl*
##   4   1  RatGtBSec*  RatGtBTer*  WmnInParl*
##   5   1  RatGtBSec*  RatGtBTer*  WmnInParl*
```

```
# Random Forest Validation
predicted_Score <- predict(fit_rtree2, validation)


Final_RMSE_model_rtree2 <- RMSE(predicted_Score, validation$Score)

# Save results in a tibble
RMSE_results <- bind_rows(RMSE_results,
                          tibble(Method = "Final Random Forest Model Validation",
                          RMSE = Final_RMSE_model_rtree2))
```

2.4 Model Performance

Our Final Random Forest Model performed exceptionally well on the validation set with a Final RMSE of 0.023734.

Method	RMSE
Naive Forecast	0.0518581
Full Linear Model (Trivial Solution)	0.0000371
Reduced Linear Model (UN Data)	0.0411649
rpart Model	0.0412000
Random Forest Model	0.0226823
Final Random Forest Model Validation	0.0237340

Random forests are frequently used as "blackbox" models in data science, as they generate reasonable predictions across a wide range of data while requiring very little configuration. Our example was no exception.

3 Conclusion

3.1 Summary and Potential Impact

We can see from the ranked variable importance extracted from our Final Random Forest Model that Political and Economic Participation, Educational Attainment, and various Health indicators are critical indicators for the Gender Gap Score.

As we noted ealy on, the high correlation between Total Fertility Rate with the Infant Mortality (0.907) and Maternal Mortatity (0.885) Rates could be an indication that countries with more access to family planning also have better outcomes for infants and their mothers.

Through better understanding of the contributing factors of these elements, perhaps we can achieve parity sooner than the predicted 99.5 years.

```
## rf variable importance
##
##             Overall
## Poli        100.000
## EcoPart      83.949
## Edu          30.376
## WmnInParl    27.678
## TotFertRt    20.685
## PopIncRt     16.001
## LifExpBoth   15.071
## LifExpMal    14.565
## RatGtBSec    13.274
## LifExpFem    12.776
## InfMortRt    11.503
## MatMortRt     7.889
## RatGtBPrim    6.783
## RatGtBTer     6.782
## Health        0.000
```

3.2 Limitations

Restricted access to metadata and processing power limited further investigation into the changes over time for the relationships explored in this project.

The small sample size also posed a challenge when selecting training and validation splits. Special care was taken to ensure large enough subsets to relect the general distribution of the dataset.

Ideally, information for any specific year or range of years would be available to run deeper analysis on the correlations between Gender Gap, Women in Parliament and other Indicators.

3.3 Future Work

The observations in this dataset lead to more questions about health outcomes for women and infants and how they relate to political empowerment and participation.

Future studies would also involve a more rubust investigation in to the Gender Gap Index and how it relates to the Happiness Index, Gross Domestic Product (GDP) and Gini index (measure representing income inequality).

Identifying the leading and lagging indicators over time would also be of particular interest.

4 References

HDX. (2014–2019, November 14–10). The Global Gender Gap Index 2013 [Detailed rankings]. World Economic Forum. https://data.humdata.org/dataset/29f2f52f-a9c2-4ff9-a99e-42b894dc18e9/resource/a8fe8c72-1359-4ce3-b8cc-03d9e5e85875

United Nations Statistics Division. (2019, September 20). Population growth, fertility, life expectancy and mortality [Trends in Maternal Mortality 1990 - 2015]. World Health Organization (WHO), the United Nations Children's Fund (UNICEF), the United Nations Population Fund (UNFPA), the World Bank and the United Nations Population Division. https://data.un.org/

United Nations Statistics Division. (2020a, September 5–November 5). Ratio of girls to boys in primary, secondary and tertiary levels [Dataset]. United Nations Educational, Scientific and Cultural Organization (UNESCO). https://data.un.org/

United Nations Statistics Division. (2020b, September 5–November 5). Seats held by women in national parliament [Dataset]. Inter-Parliamentary Union (IPU). https://data.un.org/