WorldHappiness_Report_2021

Ayman Tuffaha

9/4/2021

About this document (R Markdown Document)

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.

Introduction

This project part of the capstone project of the EdX course 'HarvardX: PH125.9x Data Science: Capstone'. It's created to take a deep analysis and to employs regression analysis in order to study the happiness of countries, and check the effect of particular factors of social society affects the happiness score, like population, generosity, GDP per capita, social support, health life expectancy, and freedom to make life choices.

The dataset I used at this project include happiness index for the year 2021 according to UN. [Happiness Index according to 2021 data] (https://www.kaggle.com/muhammedabdulazeem/worlds-happiness-countries-2021) and alos I have used [World Happiness Report 2021] (https://www.kaggle.com/ajaypalsinghlo/world-happiness-report-2021?select=world-happiness-report.csv) The data being used has 151 countries listed with 10 columns. These columns are happiness rank, country name, overall happiness score, and seven factors that aid in defining the happiness score: population, GDP per capita, Social Support, Life Expectancy, Freedom, Generosity, and Perceptions of corruption.

Some Notes: 1- Happiness Rank(lower the value, better the happiness rank) 2- Happiness score(higher the value, better the happiness score. Ranges from 0-8) 3- Population of that particular country.

The report is organized as follows: second, the data set is dissected and visualized in the **Data** section. Third, the data is partitioned and the three modeling methods are shown in the **Methods** section. Fourth, the three models are compared in the **Results** section. Finally, the **Conclusion** section summarizes and compares the results.

A glimpse of our data is shown below:

```
## Rows: 151
## Columns: 10
## $ happinessRank                <int> 1, 2, 3, 4, 5, 6, 7, 8, 9, 10,
11, 12, 13~
## $ country                      <chr> "Finland", "Denmark",
"Switzerland", "Ice~
## $ score                        <dbl> 7.809, 7.646, 7.560, 7.504,
7.488, 7.449,~
## $ pop2021                      <dbl> 5548.360, 5813.298, 8715.494,
343.353, 54~
## $ Logged.GDP.per.capita        <dbl> 10.775, 10.933, 11.117, 10.878,
11.053, 1~
## $ Social.support               <dbl> 0.954, 0.954, 0.942, 0.983,
0.954, 0.942,~
## $ Healthy.life.expectancy      <dbl> 72.000, 72.700, 74.400, 73.000,
73.300, 7~
## $ Freedom.to.make.life.choices <dbl> 0.949, 0.946, 0.919, 0.955,
0.960, 0.913,~
## $ Generosity                   <dbl> -0.098, 0.030, 0.025, 0.160,
0.093, 0.175~
## $ Perceptions.of.corruption    <dbl> 0.186, 0.179, 0.292, 0.673,
0.270, 0.338,~
```

Data

The qualitative structure and the general relationship of the data is described above. The data found here is quantitatively normal. A histogram of the happiness scores shows an approximately normal distribution and descriptive statistics support this claim.


```
summary(data$score)

##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   2.567   4.700   5.510   5.467   6.228   7.809
```

The histogram is right-leaning. This can be shown quantitively by comparing the mean score (NA) and the median score (). A mean greater than the median signifies that there are more values larger that the middle. The range of scores is (2.853, 7.769). The three highest and lowest observations are displayed below:

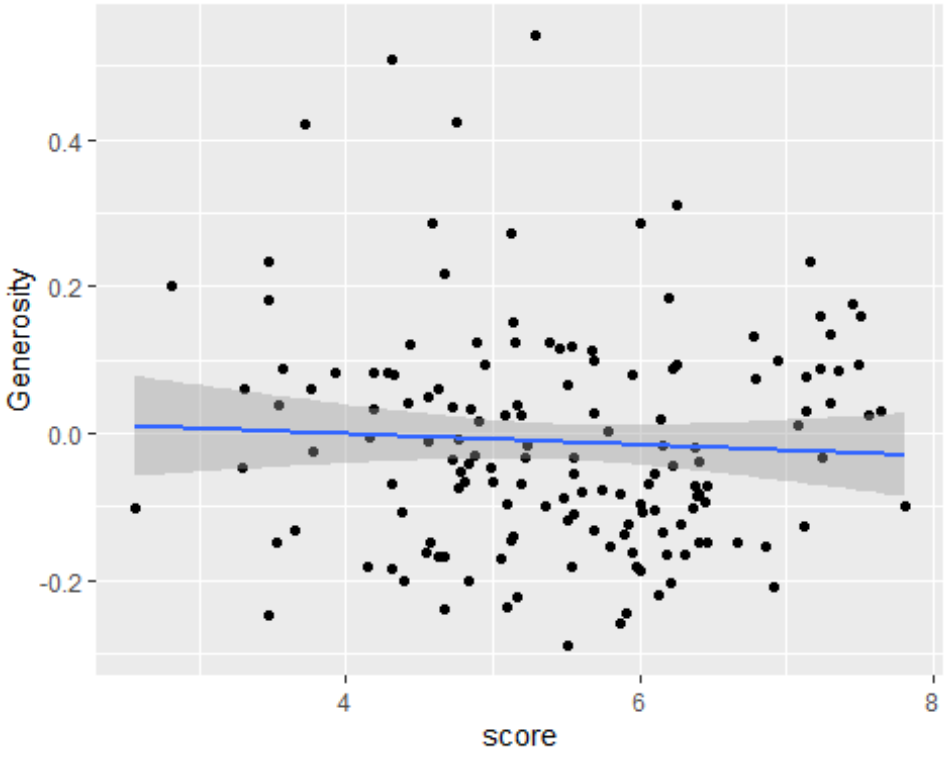
```
##   happinessRank     country score
## 1             1     Finland 7.809
## 2             2     Denmark 7.646
## 3             3 Switzerland 7.560

##   happinessRank     country score
## 1           150      Rwanda 3.312
## 2           151    Zimbabwe 3.299
## 3           152 South Sudan 2.817
## 4           153 Afghanistan 2.567
```

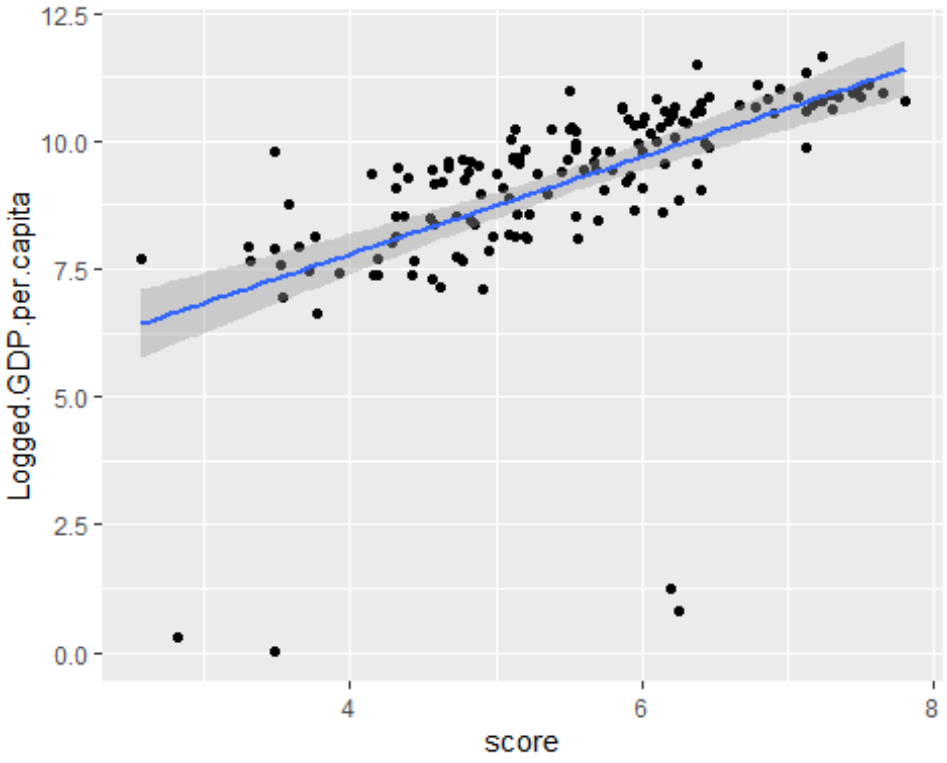
Remember that low happiness scores tend to be more like miserable societies. This means **lower happiness scores tend to have lower factor scores**. Thus, through deduction, high factor scores have a positive effect and low factor scores have a negative effect on the happiness score. Ultimately, determining the correlation of each factor to each other is of utmost interest. The `ggcorrplot` library is used to probe this:

All factors are at least somewhat related (above zero) with the partial exception of `pop2021` and `Generosity` and `Perceptions.of.corruption` . These three factors seems to have less of an effect on the happiness score with a correlation of around zero, thus, *removing the `pop2021` and `Generosity` and `Perceptions.of.corruption`term in the model may improve the accuracy*. The relationships between specific variables can be portrayed more directly by plotting them and determining the line of best fit. For instance, compare `Score` versus `Generosity` and `Score` versus `GDP per capita`:

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

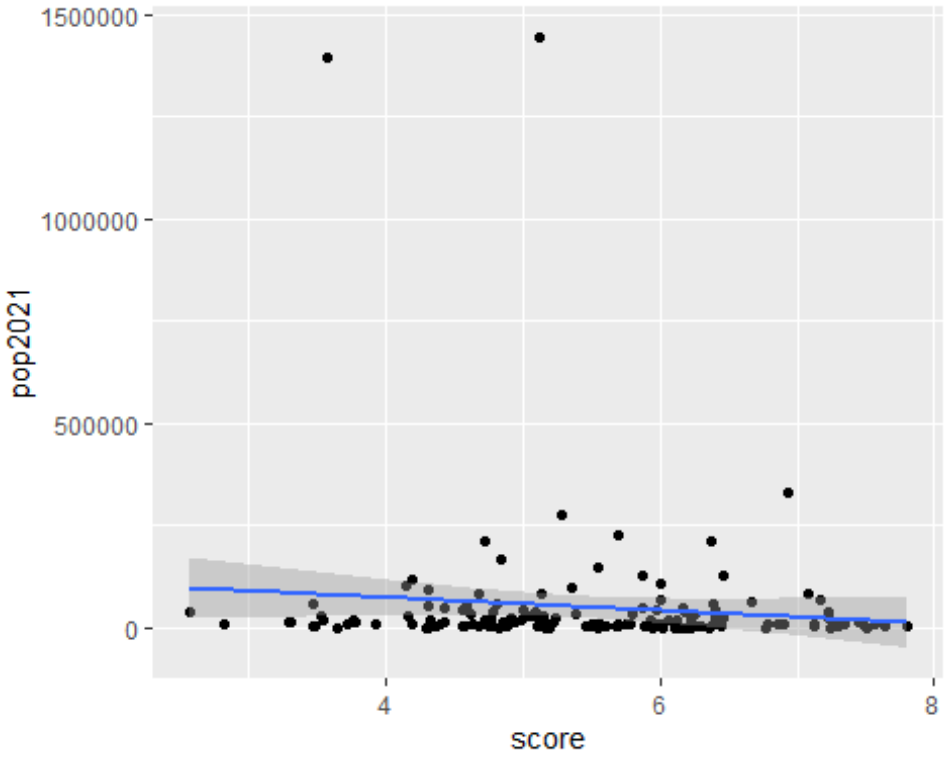



```
## `geom_smooth()` using formula 'y ~ x'
```



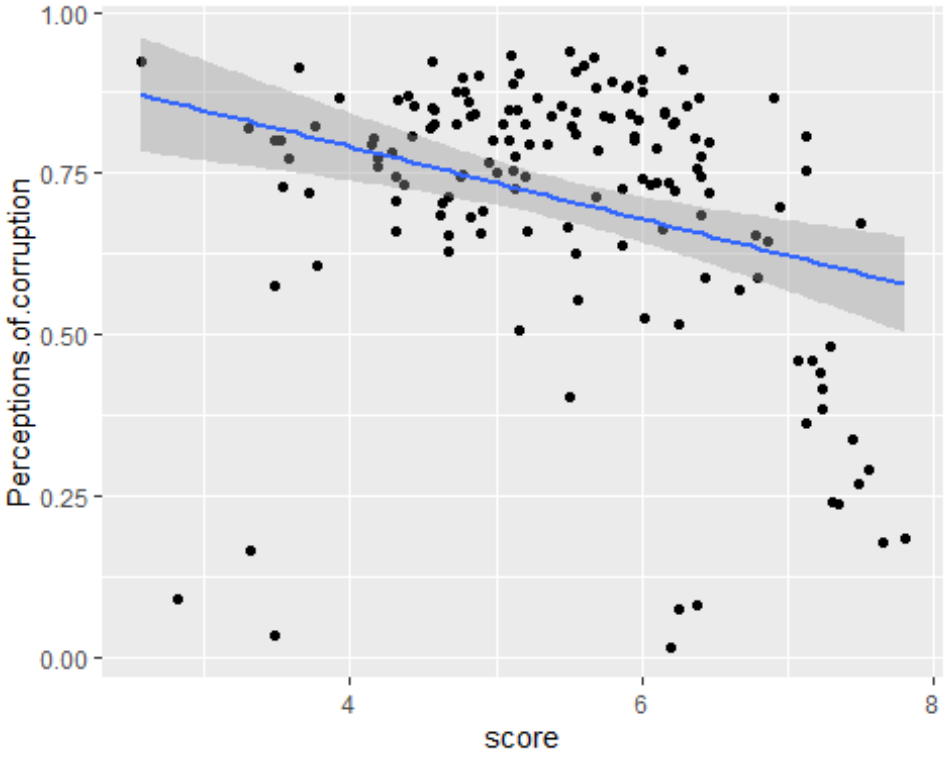
Let's compare `Score` versus population `pop2021`:

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



Let's compare `Score` versus `Perceptions.of.corruption`:

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



In the next section, we build several models to determine the best way to predict a happiness score.

Methods

Model 1: The Sum of Factors

The first model is seemingly the most obvious and was proposed on a discussion forum from the Kaggle website. It states that the "perfect" prediction model is to simply take the sum of all factors as the happiness score. This model is attempted with one caveat: a "standard dystopia score" was discovered in earlier happiness reports and was given the value 1.85. This value is added to our predicted scores as well because each factor is a ranking of how much *better* the country is than the standard dystopia. Note the use Root Mean Square Error (RMSE) as a success indicator. This choice is further explained in the results section.

```
# find  predicted score by sum method and calculate the corresponding
RMSE
sum_model <- data %>% mutate(pred_score = pop2021 +
                                Logged.GDP.per.capita +
                                Social.support +
                                Healthy.life.expectancy +
                                Freedom.to.make.life.choices +
                                Generosity +
                                Perceptions.of.corruption +
                                1.85,
                              RMSE = RMSE(score, pred_score))
```

```
# show top results of the summation model
sum_model %>%
  filter(happinessRank <= 5) %>%
  select(happinessRank, country, score, pred_score, RMSE)

##   happinessRank     country score pred_score   RMSE
## 1             1     Finland 7.809   5634.976 174391
## 2             2     Denmark 7.646   5900.890 174391
## 3             3 Switzerland 7.560   8805.039 174391
## 4             4     Iceland 7.504    431.852 174391
## 5             5      Norway 7.488   5554.110 174391

# calculate the missing dystopian residuals
sum_model <- sum_model %>% mutate(residual = score - pred_score)
# show top results of the summation model
sum_model %>%
  filter(happinessRank <= 5) %>%
  select(happinessRank,  country, score, pred_score, RMSE, residual)

##   happinessRank     country score pred_score   RMSE  residual
## 1             1     Finland 7.809   5634.976 174391 -5627.167
## 2             2     Denmark 7.646   5900.890 174391 -5893.244
## 3             3 Switzerland 7.560   8805.039 174391 -8797.479
## 4             4     Iceland 7.504    431.852 174391  -424.348
## 5             5      Norway 7.488   5554.110 174391 -5546.622
```

Model 2: The 2021 GLM Model

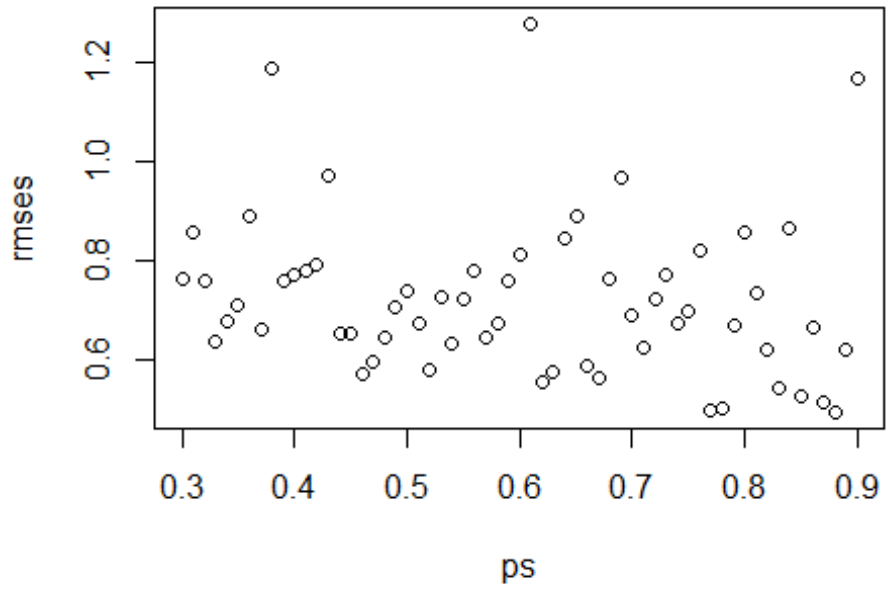
Before our first linear regression model is applied, the data must be partitioned into a training and test set. This step is common when employing machine learning algorithms that require a check on the goodness of fit. It reduces the probability of overfitting to our training data at the expense of our prediction model. This was not completed for our sum of factors model because a model did not need to be trained, the equation was simply `sum(data$[factors])`.

The Happiness Report has over 150 country observations and seven factors in which we will condition our model. Given the relatively low number of observations compared to the amount of factors, the model may have a tendency to overfit to the training data by overweighting unimportant variables. This is almost unavoidable when working with low volumes of data. The reality of regression is that you can *always* find a model that fits your training data exactly but which is typically useless for prediction. Keeping this in mind, an optimal training-test data split ratio, that is 70 training:30 test, 80 training:20 test, ..., etc., is first determined.

```
# --- test for an appropriate ratio in data partitioning
# a sequence of p's we want to test
ps <- seq(from=.30, to=.90, by=.01)
# calculate RMSEs for each p value
rmses <- sapply(ps, function(p){
  train_index <- createDataPartition(data$score, times=1, p=p,
```

```
list=FALSE)
  train <- data[train_index,]
  test <- data[-train_index,]
  fit <- glm(score ~ pop2021 +
               Logged.GDP.per.capita +
               Social.support +
               Healthy.life.expectancy +
               Freedom.to.make.life.choices +
               Generosity +
               Perceptions.of.corruption,
             data = train)
  test <- test %>% mutate(pred_score = predict.glm(fit, newdata=test))
  RMSE(test$score, test$pred_score)
})
```

The tested model in the partitioning is explored after this section. Plotting `rmses` versus `p` shows a slight pattern: when you increase the training data size, our RMSE decreases. Intuitively, this makes sense. The data is quite correlated as it has already been shown and the model is being allowed to work with more training data to make better predictions in the test set. From the plot below, the lowest RMSE is 0.4968906 with a ratio of 0.88:0.12.



While useful in achieving a low RMSE, employing only 0.12 percent of our data to test does not leave much in terms of prediction. Ultimately, an arbitrary value of 0.70 is chosen because RMSE seems to become more sporadic after this value.

Future models in the *methods* section use 0.70 as well. This ratio is also kept when the current data is supplemented with more data.

```
# set seed to keep partitioning consistent
set.seed(1, sample.kind = "Rounding")
# ----- Data partitioning -----
train_index <- createDataPartition(data$score, times=1, p=0.70,
list=FALSE)
train <- data[train_index,]
test <- data[-train_index,]
```

With our data partitioned 0.70:0.30, a generalized linear model is fitted using the `caret` package. `score` is predicted using all seven factors.

```
# --- fit our glm model, caret::glm
fit <- glm(score ~ pop2021 +
             Logged.GDP.per.capita +
             Social.support +
             Healthy.life.expectancy +
             Freedom.to.make.life.choices +
             Generosity +
             Perceptions.of.corruption,
           data = train)
# add predicted scores to a 'results' data frame
results <- test %>%
  mutate(pred_score = predict.glm(fit, newdata=test))
```

```
##    happinessRank        country score pred_score
## 1              1        Finland 7.809   7.331346
## 6              6    Netherlands 7.449   7.114962
## 10            10     Luxembourg 7.238   7.039390
## 13            13 United Kingdom 7.165   6.776602
## 14            14         Israel 7.129   6.328248
## 16            16        Ireland 7.129   7.126062
```

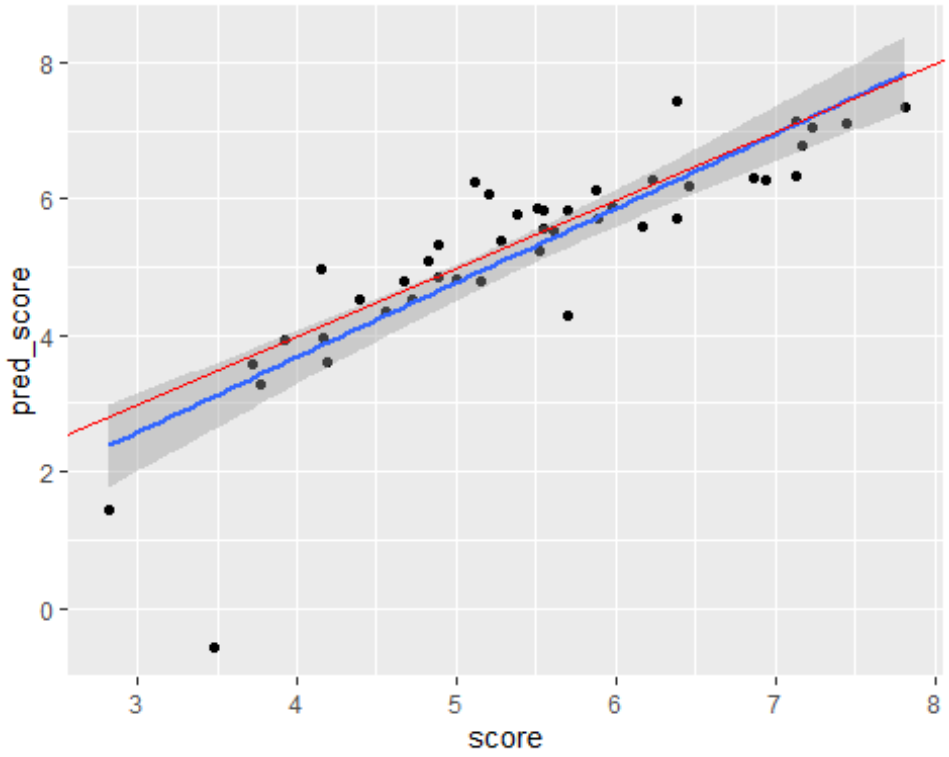
```
##     happinessRank                 country score pred_score
## 136           138                   Egypt 4.151  4.9705247
## 137           139            Sierra Leone 3.926  3.9329431
## 138           140                 Burundi 3.775  3.2946238
## 140           142                   Haiti 3.721  3.5845487
## 147           149 Central African Republic 3.476 -0.5696986
## 150           152             South Sudan 2.817  1.4417747
```

The top five and bottom five observations are shown above compared with the actual score. The `results` data frame is plotted below with a line of best fit in blue and a reference line in red at `y = x`. If the model was to work perfectly, the line of best fit would follow the reference line because each predicted score would be equal to the score.

```
# plot predicted scores vs actual scores
# also plot y = x line
```

```
ggplot(data = results, aes(score, pred_score)) +
  geom_point(color='black') +
  geom_smooth(method = "lm", se = TRUE) +
  geom_abline(color='red')

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



Even though results have been shown, it is worth mentioning the use of RMSE instead of other success indicators. RMSE suggests how close (or far) your predicted values are from the actual data you are attempting to model. The use of a success measure for this model, and others in the methods section, is to understand the accuracy and precision of the model's predictions. For this reason, RMSE is used as a success metric over alternatives. The RMSE of this model is NaN. The coefficients of the fitted model are shown below for completeness:

```
##                   (Intercept)                       pop2021
##                 -9.800153e-01                 -5.426788e-07
##         Logged.GDP.per.capita                Social.support
##                  2.194181e-01                  4.196750e+00
##       Healthy.life.expectancy Freedom.to.make.life.choices
##                  6.270194e-03                  1.818020e+00
##                    Generosity     Perceptions.of.corruption
##                  1.614081e-01                 -1.153200e+00
```

The following equation is the final model equation. Using these coefficients and the following notation predicted score = $\hat{y}$, GDP per capita score = x_{GDP}, Social Support

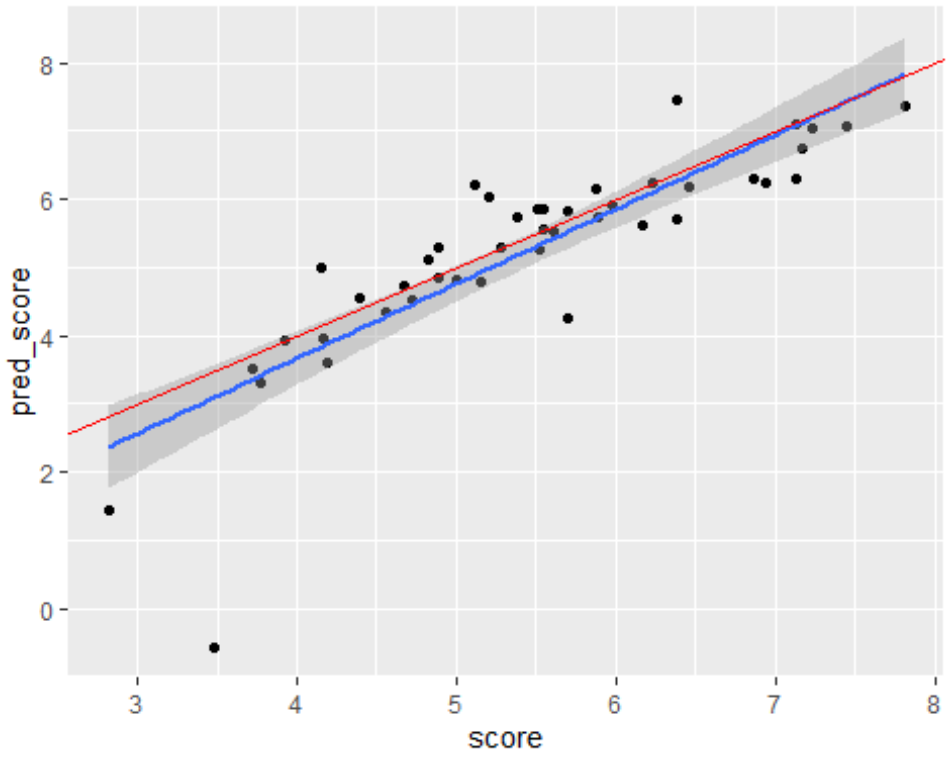
score = x_{SS}, Life Expectancy score = x_{HEA}, Freedom score = x_{FRE}, Generosity score = x_{GEN}, Truth score = x_{TRU}.

Removing Generosity

In an attempt to improve the model, we can remove the generosity component because early evaluations pointed to it being the least correlated factor. The previously partitioned data (`p = 0.70`) is used in this model as well.

```
# fit model without generosity
fit <- glm(score ~ pop2021 +
             Logged.GDP.per.capita +
             Social.support +
             Healthy.life.expectancy +
             Freedom.to.make.life.choices +
             Perceptions.of.corruption,
           data = train)
# add predicted scores to a 'results' data frame
results <- test %>%
  mutate(pred_score = predict.glm(fit, newdata=test))
# plot predicted scores vs actual scores
ggplot(data = results, aes(score, pred_score)) +
  geom_point(color='black') +
  geom_smooth(method = "lm", se = TRUE) +
  geom_abline(color='red')

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



This model yields a RMSE of NaN. The model coefficients are shown below along with the model equation following the same format as before.

```
##                   (Intercept)                        pop2021
##                 -9.514777e-01                  -5.472381e-07
##         Logged.GDP.per.capita                 Social.support
##                  2.103450e-01                   4.189786e+00
##       Healthy.life.expectancy Freedom.to.make.life.choices
##                  6.892309e-03                   1.863964e+00
##     Perceptions.of.corruption
##                 -1.179317e+00
```

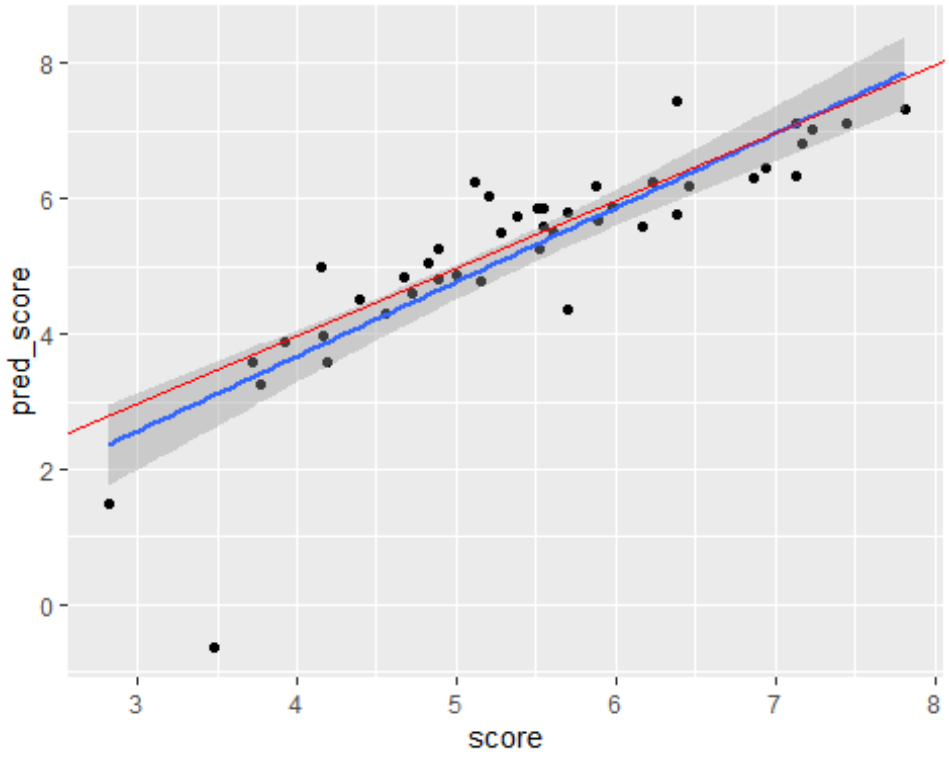
Removing Population pop2021

In an attempt to improve the model, we can remove the population component because early evaluations pointed to it being the least correlated factor. The previously partitioned data (`p = 0.70`) is used in this model as well.

```
# fit model without population
fit1 <- glm(score ~ Logged.GDP.per.capita +
                    Social.support +
                    Healthy.life.expectancy +
                    Freedom.to.make.life.choices +
                    Generosity +
                    Perceptions.of.corruption,
                    data = train)
```

```
# add predicted scores to a 'results' data frame
results <- test %>%
  mutate(pred_score = predict.glm(fit1, newdata=test))
# plot predicted scores vs actual scores
ggplot(data = results, aes(score, pred_score)) +
  geom_point(color='black') +
  geom_smooth(method = "lm", se = TRUE) +
  geom_abline(color='red')

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



This model yields a RMSE of NaN. The model coefficients are shown below along with the model equation following the same format as before.

```
##                   (Intercept)         Logged.GDP.per.capita
##                   -0.99973548                    0.22457855
##                Social.support       Healthy.life.expectancy
##                    4.31000148                    0.00694251
## Freedom.to.make.life.choices                    Generosity
##                    1.60182501                    0.19592334
##    Perceptions.of.corruption
##                   -1.18281017
```

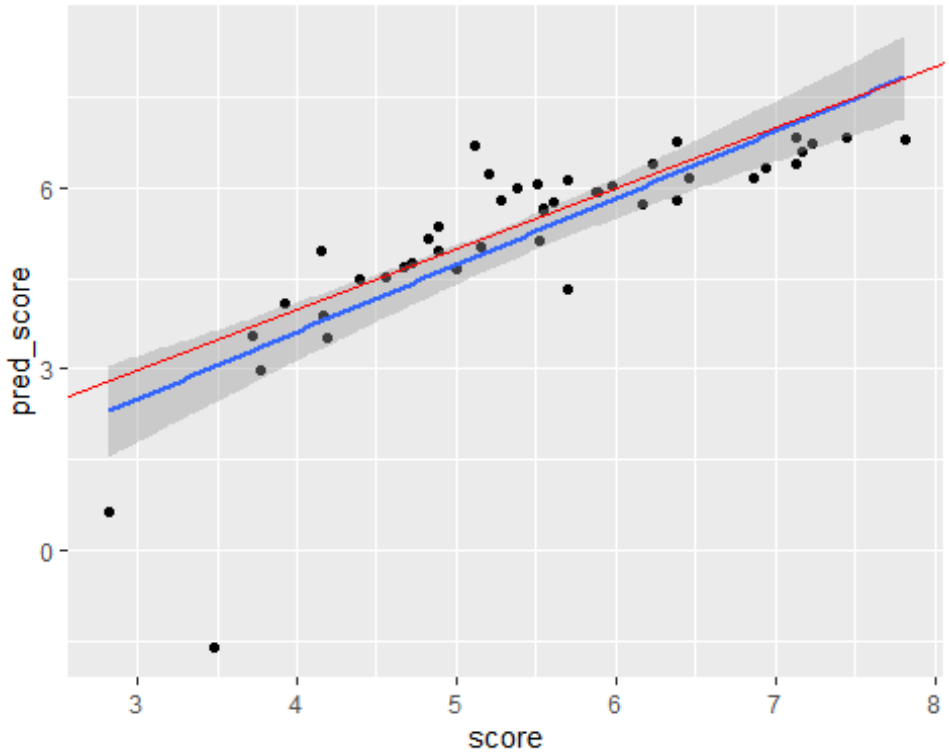
Removing courruptiion Perceptions.of.corruption

In an attempt to improve the model, we can remove the Perceptions.of.corruption component because early evaluations pointed to it being the least correlated factor. The previously partitioned data (`p = 0.70`) is used in this model as well.

```
# fit model without corruption
fit2 <- glm(score ~ pop2021 +
                    Logged.GDP.per.capita +
                    Social.support +
                    Healthy.life.expectancy +
                    Freedom.to.make.life.choices +
                    Generosity,
                    data = train)

# add predicted scores to a 'results' data frame
results <- test %>%
  mutate(pred_score = predict.glm(fit2, newdata=test))
# plot predicted scores vs actual scores
ggplot(data = results, aes(score, pred_score)) +
  geom_point(color='black') +
  geom_smooth(method = "lm", se = TRUE) +
  geom_abline(color='red')

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



This model yields a RMSE of NaN. The model coefficients are shown below along with the model equation following the same format as before.

```
##                   (Intercept)                          pop2021
##                 -2.239205e+00                    -5.900398e-07
##         Logged.GDP.per.capita                   Social.support
##                  2.165616e-01                     4.697929e+00
##       Healthy.life.expectancy Freedom.to.make.life.choices
##                  1.413729e-04                     2.383094e+00
##                    Generosity
##                  4.776454e-01
```

Results

The results of our five models are shown in the table below. It is clearly shown that the best model in terms of RMSE is the sum of factors model. Even though the data seemed incomplete with dystopian residuals, taking the sum of the factor and the standard dystopian scores yield the closest score predictions.

Method	RMSE
Sum of Factors Model	1.74391^{5}
Model 2 RMSE with all factors	0.806
GLM Model - No Generosity 2021	0.808
GLM Model - No Population 2021	0.805
GLM Model - No corruption 2021	0.999

Conclusion

This report felt limited in the amount of data being trained and tested. If more consistent data was available, it is my belief that the summation model would out perform the GLM models. The other thing to consider at this report is different component factors like population of the country or generosity or perceptions of corruption may slightly affects the happiness score as indicated at the above RMSE values based on 3 taken factors: Generosty, number of population, and perseptions of corruption.