What makes a country happy?

Adam Wilson

May 7, 2019

Introduction & Background

This is no easy question to answer, but we will do our best using the information provided from the Wikipedia article titled “World Happiness Report”. You can find the url in the ‘References’ section at the very end of this report.

This project and report is for the **Data Science: Capstone** (**PH125.9x**) course, by **HarvardX**, on https://www.edx.org/ (https://www.edx.org/).

We will be focusing on the 2018 report, which shows the international rankings of countries listed in descending order of their *happiness*. It averages the scores over 2015-2017 for 156 countries.

The scores are obtained as follows: respondents are asked to think of a ladder, with the best possible life being a 10, and the worst possible life being a 0. Respondents are then asked to rate their own life on this 0-10 ladder.

These scores are correlated with various life factors that vary among countries. These are:

- **GDP Per Capita**: This is calculated by using the natural log of a country’s citizens’ purchasing power, adjusted to 2011 international dollars. The natural log is used rather than direct GDP per capita as this form fits the data better.
- **Social Support**: Respondents are asked “If you were in trouble, do you have relatives or friends you can count on to help you whenever you need them?”. The response is binary, either “yes” (1) or “no” (0). The national average of these binary responses is calculated. (Note that despite supposedly being an average of binaries, there are many values on the table between 1 and 2. I was unable to decipher the reasoning behind this. However, this should not disrupt our analysis.)
- **Healthy Life Expectancy**: This is the ratio of people that life in ‘good health’ for a given age. A time series of healthy life expectancy is constructed based on data from the World Heath Organization (WHO) and World Development Indicators (WDI). We generate the ratios of healthy life expectancy to raw life expectancy.
- **Freedom to Make Life Choices**: This is the national average of binary responses to the question “Are you satisfied with your freedom to choose what to do with your life?”.
- **Generosity**: This is the residual of regressing the national average of responses to the question “Have you donated money to charity in the post month?”, on GDP per capita.
- **Perceptions of Corrution**: This is the average of binary answers to two questions: “Is corruption widespread throughout the government?” and “Is corruption widespread within business?”.

Our goal is to compare the scores to these other six variables, determine exactly how correlated they are, and find out if these variables are sufficient for determining happiness.

Methodology

This analysis is done using the **R** programming language.

First we load the necessary packages. We retrieve the html code of our Wikipedia article of interest, and assign the required table, the *2018 report*, into a dataframe. Then we edit this dataframe to make it more analysis-friendly.

Next we move on to observing our data. We create density plots for *score* and the 6 independent variables. We then plot *score* against these variables, one at a time. We then have a look at the correlation coefficients between *score* and our other variables, and create a graph to show these visually.

Next we will perform machine learning algorithms on our data. We create training and testing sets of our data, representing 80% and 20% of our data respectively. We perform our algorithms on the training set and then see how accurate it is on predicting scores in the testing set. We will judge the effectiveness of our algorithms based on their resulting residual mean squared error (RMSE) and accuracy. We judge a score to be sufficiently accurate if it is within 0.5 points of the actual score. First we see how accurate it will be to simply guess the average score from our training set and apply it to our testing set. We will then use linear regression, followed by k-nearest neighbors (KNN), followed by the random forests algorithm.

Analysis & Results

First, let’s look at the top ten entries of our table.

```
head(tab,10)
```

```
##    Overall Rank Country/Region Score GDP per capita Social support
## 1             1        Finland 7.632          1.305          1.592
## 2             2         Norway 7.594          1.456          1.582
## 3             3        Denmark 7.555          1.351          1.590
## 4             4        Iceland 7.495          1.343          1.644
## 5             5    Switzerland 7.487          1.420          1.549
## 6             6    Netherlands 7.441          1.361          1.488
## 7             7         Canada 7.328          1.330          1.532
## 8             8    New Zealand 7.324          1.268          1.601
## 9             9         Sweden 7.314          1.355          1.501
## 10           10      Australia 7.272          1.340          1.573
##    Healthy life expectancy Freedom to make life choices Generosity
## 1                    0.874                        0.681      0.192
## 2                    0.861                        0.686      0.286
## 3                    0.868                        0.683      0.284
## 4                    0.914                        0.677      0.353
## 5                    0.927                        0.660      0.256
## 6                    0.878                        0.638      0.333
## 7                    0.896                        0.653      0.321
## 8                    0.876                        0.669      0.365
## 9                    0.913                        0.659      0.285
## 10                   0.910                        0.647      0.361
##    Perceptions of corruption
## 1                      0.393
## 2                      0.340
## 3                      0.408
## 4                      0.138
## 5                      0.357
## 6                      0.295
## 7                      0.291
## 8                      0.389
## 9                      0.383
## 10                     0.302
```

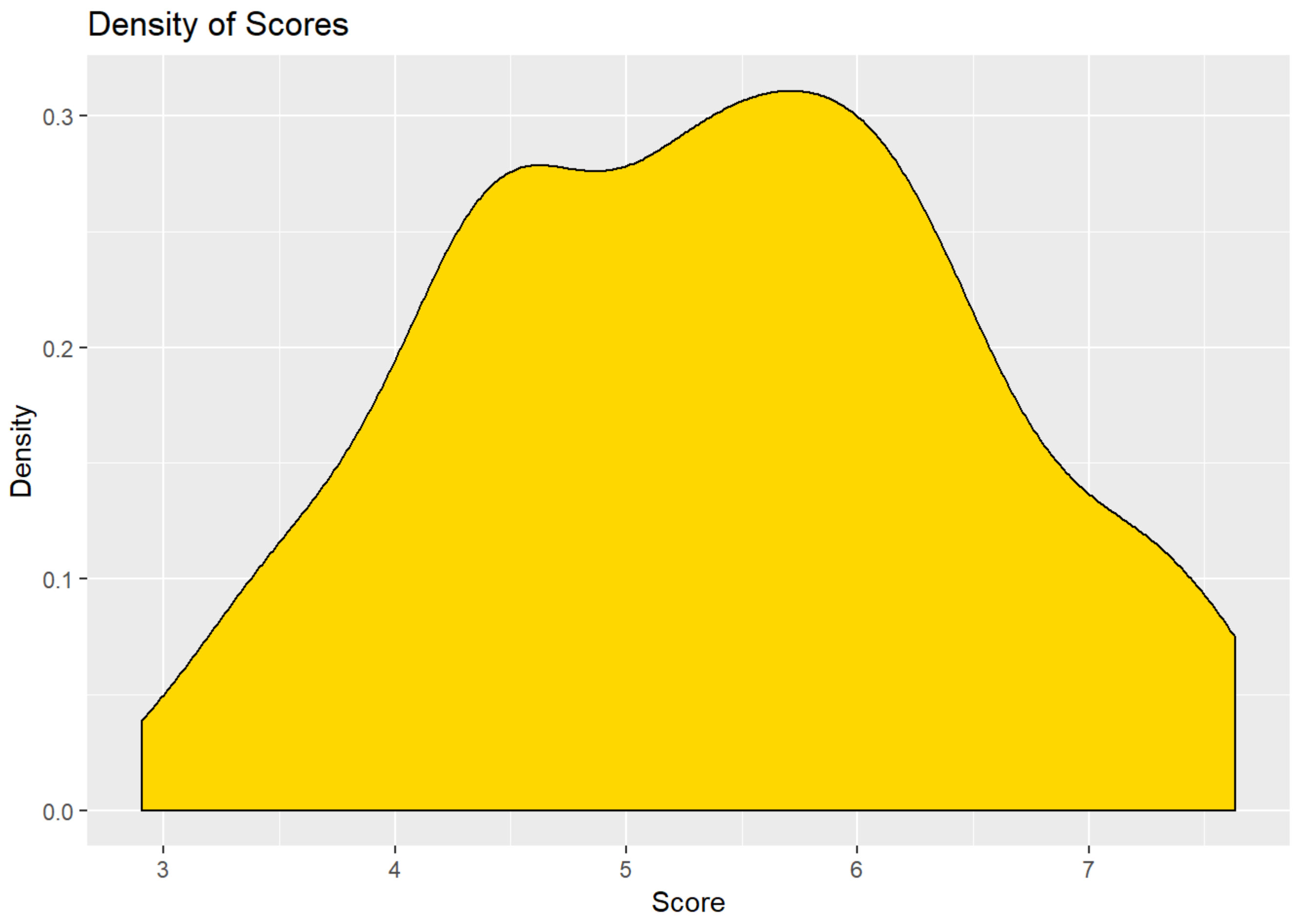
We make our dataframe more friendly for analysis, and observe its structure.

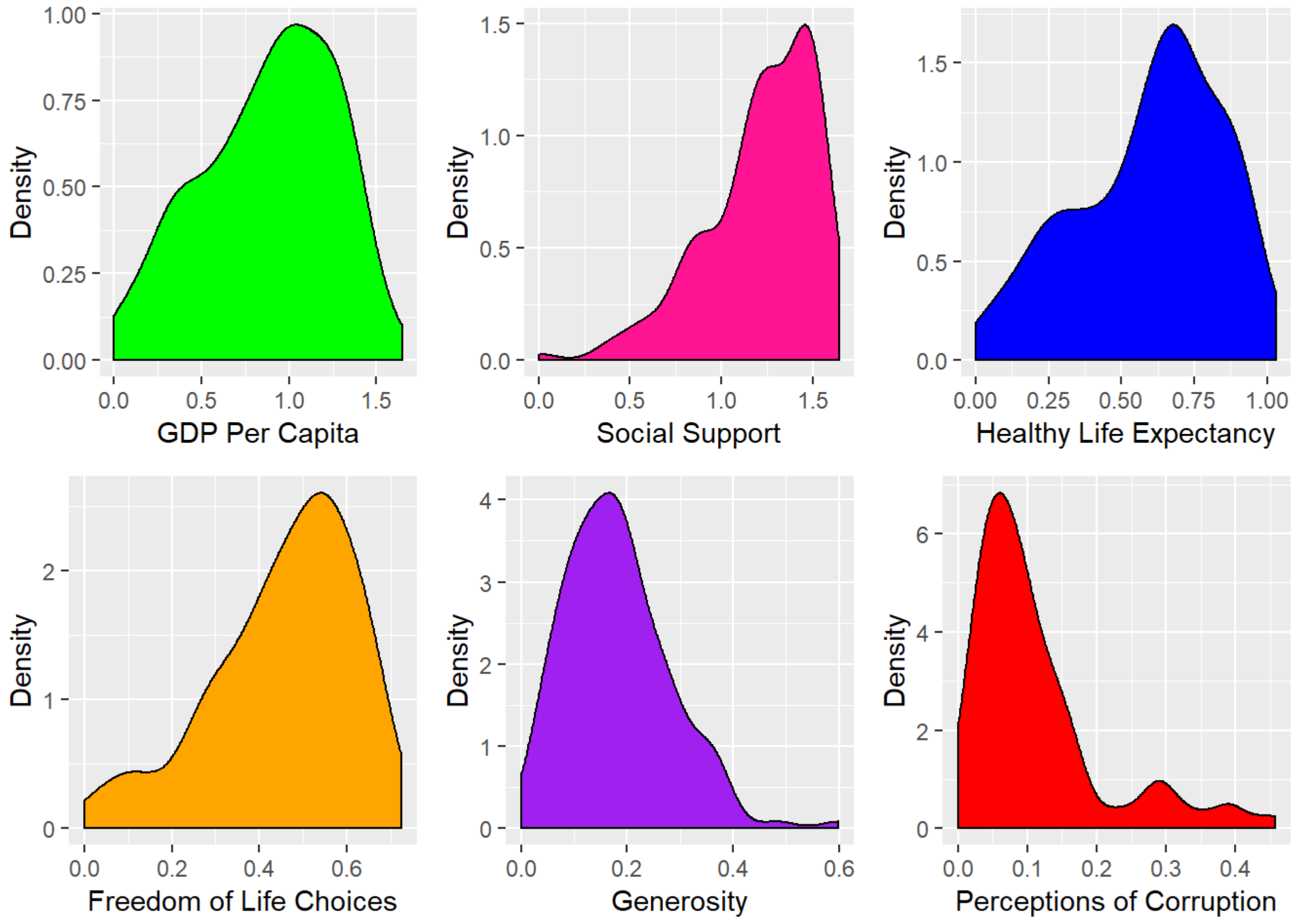
```
str(ranks)
```

```
## 'data.frame':    156 obs. of  9 variables:
##  $ rank                   : int  1 2 3 4 5 6 7 8 9 10 ...
##  $ country                : chr  "Finland" "Norway" "Denmark" "Iceland" ...
##  $ score                  : num  7.63 7.59 7.55 7.5 7.49 ...
##  $ gdp_per_capita         : num  1.3 1.46 1.35 1.34 1.42 ...
##  $ social_support         : num  1.59 1.58 1.59 1.64 1.55 ...
##  $ healthy_life_expectancy: num  0.874 0.861 0.868 0.914 0.927 0.878 0.896 0.876 0.913 0.91
...
##  $ life_choices_freedom   : num  0.681 0.686 0.683 0.677 0.66 0.638 0.653 0.669 0.659 0.647
...
##  $ generosity             : num  0.192 0.286 0.284 0.353 0.256 0.333 0.321 0.365 0.285 0.361
...
##  $ corruption_perceptions : chr  "0.393" "0.340" "0.408" "0.138" ...
```

We see that all of our variables are numerical except for *corruption_perceptions*, which has been defined as a list of characters. This is because there is a value labelled 'N/A' within this variable's column. We remove this country from our analysis.

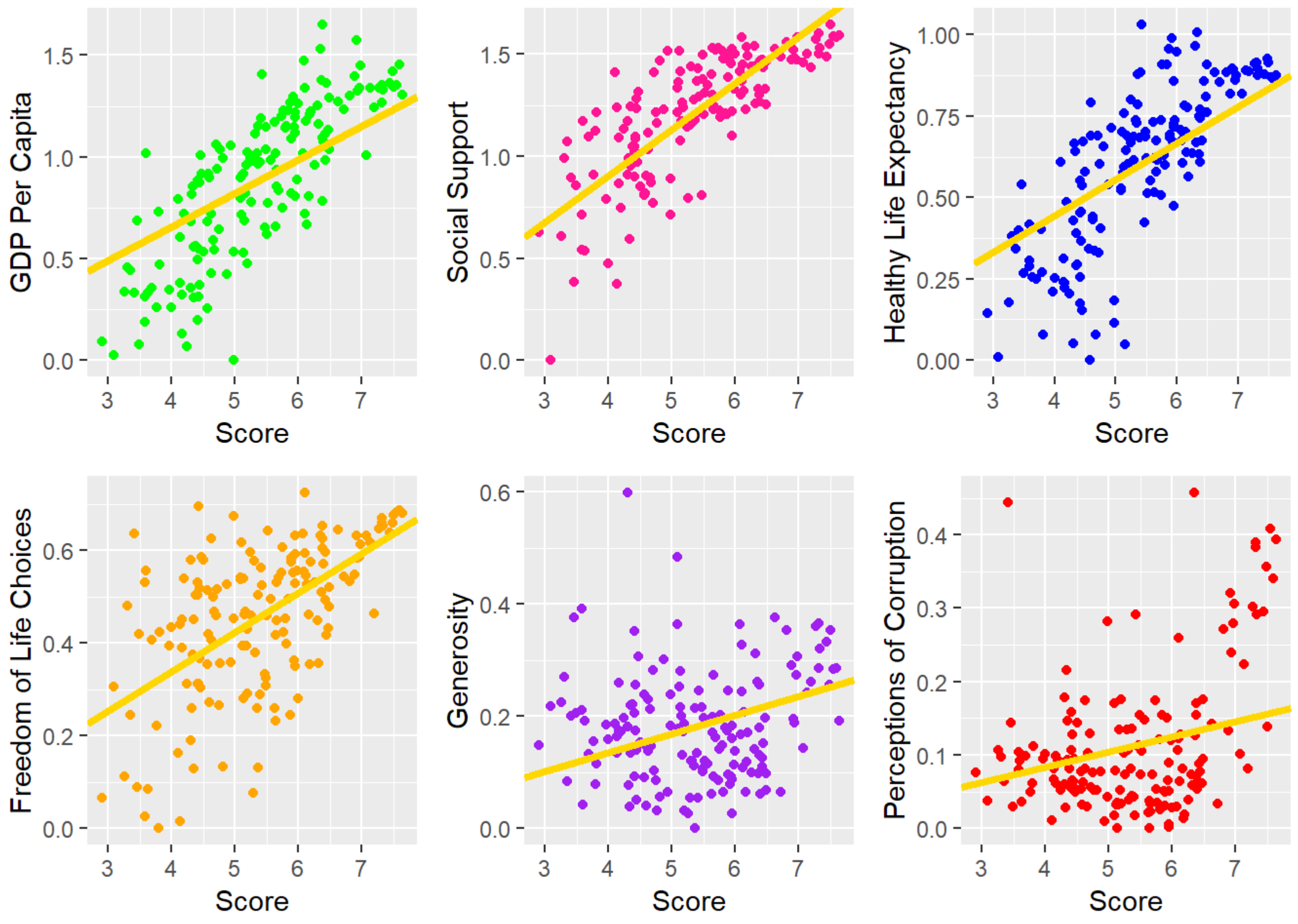
Now let's look at some density plots, first of *score*, and then of our independent variables.



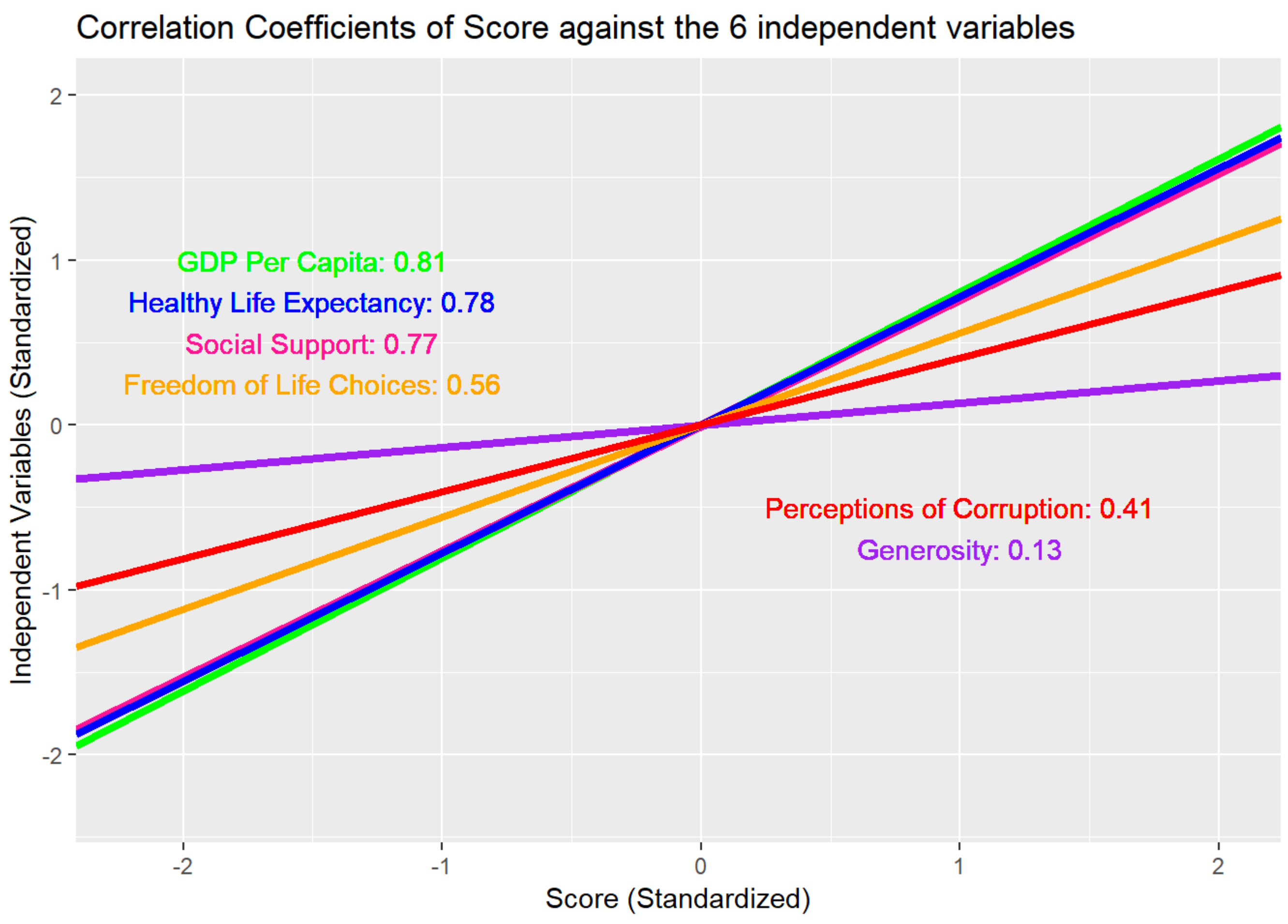


It looks like *score* approximately follows a Normal distribution, but the same cannot be said for our other variables.

Now let’s plot *score* against the other variables.



It looks like *score* has a high correlation with *GDP per capita*, *social support*, *healthy life expectancy*, and *freedom of life choices*, but has a lower correlaton with *generosity* and *perceptions of corruption*. We can confirm this with another graph that shows the correlation coefficients.



Now let’s continue on to our machine learning algorithms. As mentioned before, we split our data into two parts: training set, consisting of 80% of the data, and testing set, consisting of the other 20%.

Method 1: Guessing the average

Before we start our algorithms, let’s see how far we get with just guessing that a score from the testing set will be the **average score** of the training set.

```
m <- mean(train_set$score)
m
```

```
## [1] 5.379049
```

Our mean score is **5.38**. Now let’s see how our *residual mean squared error* (*RMSE*) and *accuracy* fare. As mentioned before, we will consider a predicted score as accurate enough if it is within 0.5 points of the true score.

```
avg_rmse
```

```
## [1] 1.122587
```

```
mean(avg_acc)
```

```
## [1] 0.3125
```

We get an RMSE of **1.12** and accuracy of **0.31**. This means that a given predicted score is an average of 1.12 points away from its true score, and that the predicted score correctly guesses the true score 31% of the time.

Method 2: Linear Regression

We now move on to **linear regression**.

```
fit$coef
```

```
##             (Intercept)          gdp_per_capita          social_support 
##               1.6792876               0.7392670               1.2448642 
## healthy_life_expectancy    life_choices_freedom              generosity 
##               1.2880446               1.3357963               0.4076508 
##  corruption_perceptions 
##               0.6533542
```

This means that, according to our linear model, *score* can be predicted as follows (to two decimal places):

`Score = 1.68 + 0.74G + 1.24S + 1.29H + 1.34F + 0.41G + 0.65C`,

where `G = GDP per capita`, `S = social support`, `H = healthy life expectancy`, `F = freedom of life choices`, `G = generosity`, and `C = perceptions of corruption`.

Now let’s see how this model performs on our testing set.

```
lm_rmse
```

```
## [1] 0.5417422
```

```
mean(lm_acc)
```

```
## [1] 0.65625
```

We have improved our results; we have an RMSE of **0.54**, and an accuracy of **66%**.

Method 3: KNN

Now we move on to the **k-nearest neighbors** (**KNN**) algorithm. This algorithm works by defining a data point in the feature space (of our independent variables) that we wish to define. The **k** is how many of the closest points (in our case, these are the scores) from that data point we wish to consider. The average of these points is what our given data point becomes. We will perform cross-validation to find the *k* that gives us the minimum RMSE.

```
train_knn$bestTune
```

```
##   k
## 4 8
```

This means that our KNN algorithm works best when a given data point considers its **8** closest neighbors (which is the 4th observed number in our algorithm, as we try out every even number from 2 to 100). So what RMSE and accuracy does this give us?

```
knn_rmse
```

```
## [1] 0.537452
```

```
mean(knn_acc)
```

```
## [1] 0.65625
```

Our results are very similar, with an RMSE of **0.54** and an accuracy of **66%**.

Method 4: Random Forests

We move on the final algorithm we consider for this report: the **random forests** algorithm. This algorithm uses **decision trees** (which is, in this context, an algorithm where you move down different paths of a 'tree' based on the value of variables, to arrive at the optimal *score*), but improves performance by average multiple decision trees. We get the following RMSE and accuracy from this method:

```
rf_rmse
```

```
## [1] 0.4957967
```

```
mean(rf_acc)
```

```
## [1] 0.71875
```

This gives us our best result, with an RMSE of **0.50** and accuracy of **72%**.

Let's make our findings more readable by organizing them into simple dataframes.

Here we compare the true scores from our test set against our predictions:

```
ml_scores
```

```
##    True Score Average Score Difference Linear Regression Difference
## 1        7.63                    -2.25                        -0.64
## 2        7.44                    -2.06                        -0.59
## 3        6.93                    -1.55                        -0.26
## 4        6.49                    -1.11                        -0.01
## 5        6.31                    -0.93                         0.16
## 6        6.26                    -0.88                        -0.49
## 7        6.19                    -0.81                        -0.17
## 8        6.17                    -0.79                        -0.16
## 9        6.11                    -0.73                         0.14
## 10       6.07                    -0.69                         0.05
## 11       5.92                    -0.54                         0.68
## 12       5.88                    -0.50                        -0.10
## 13       5.79                    -0.41                         0.18
## 14       5.66                    -0.28                        -0.35
## 15       5.52                    -0.14                         0.00
## 16       5.43                    -0.05                         1.21
## 17       5.32                     0.06                         0.16
## 18       5.25                     0.13                        -0.47
## 19       5.25                     0.13                         0.49
## 20       4.97                     0.40                        -0.85
## 21       4.88                     0.50                         0.21
## 22       4.81                     0.57                         0.52
## 23       4.67                     0.71                        -0.65
## 24       4.47                     0.91                         1.33
## 25       4.45                     0.93                        -0.18
## 26       4.44                     0.94                         0.71
## 27       4.32                     1.06                         0.45
## 28       3.96                     1.42                        -0.11
## 29       3.63                     1.75                        -0.50
## 30       3.58                     1.80                        -0.15
## 31       3.36                     2.02                         0.83
## 32       3.08                     2.30                        -0.85
##    KNN Difference Random Forests Difference
## 1           -0.28                     -0.42
## 2           -0.28                     -0.46
## 3            0.19                     -0.15
## 4           -0.15                      0.01
## 5            0.06                      0.08
## 6           -0.40                     -0.60
## 7           -0.20                     -0.48
## 8           -0.37                     -0.28
## 9            0.48                      0.18
## 10           0.17                     -0.19
## 11           0.58                      0.55
## 12           0.21                     -0.01
## 13           0.04                      0.10
## 14          -0.10                     -0.36
## 15           0.25                      0.15
## 16           1.48                      1.15
## 17           0.52                      0.15
## 18          -0.67                     -0.40
## 19           0.50                      0.27
## 20         -0.65                      -0.56
## 21         -0.04                      -0.01
## 22          0.48                       0.57
## 23         -0.28                      -0.39
## 24          1.13                       1.32
## 25         -0.24                       0.01
## 26          0.92                       0.45
## 27          0.60                       0.51
## 28          0.08                       0.13
## 29          0.18                       0.08
## 30          0.41                       0.15
## 31          0.84                       0.73
## 32          0.78                       1.05
```

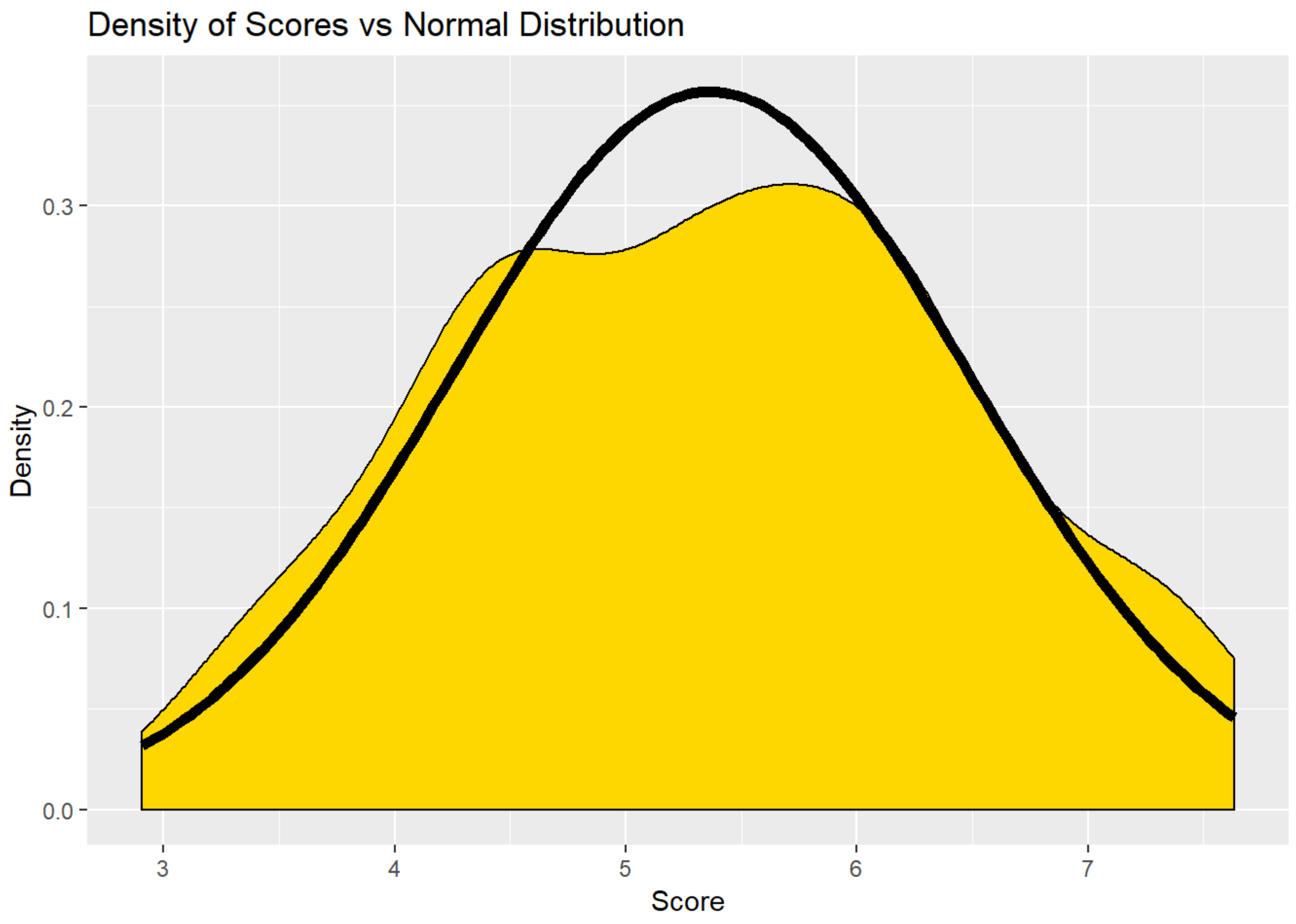
Here we summarize our RMSEs and accuracy of each algorithm:

```
ml_df
```

```
##              Method RMSE Accuracy
## 1  Just the Average 1.12     0.31
## 2 Linear Regression 0.54     0.66
## 3               KNN 0.54     0.66
## 4     Random Forest 0.50     0.72
```

Discussion

As mentioned in the analysis, it appears that *score* approximately follows a Normal Distribution. We can confirm this graphically by augmenting our plot, comparing the density curve of *score* to the Normal Distribution curve, with a mean and standard deviation equal to that of the *score* variable.



The same cannot be said for our six independent variables. It appears that *GDP per capita*, *social support*, *healthy life expectancy*, and *freedom of life choices* are skewed to the right, whilst *generosity* and *perception of corruption* are skewed to the left. This is mostly good news, since it shows that most countries' citizens aren't impoverished, have sufficient support from friends and family, will live a large portion of their life relatively healthily, feel that they have have the freedom to pursue their life choices, and do not feel that their government and local businesses are corrupt. It does seem, however, that most respondents aren't particularly charitable.

Observing our correlation coefficients, it appears that *score* correlates highly with *GDP per capita*, *healthy life expectancy*, and *social support*, correlates moderately with *freedom of life choices* and *perceptions of corruption*, and has little correlation with *generosity*. This implies that happiness is majorly affected by income, support from friends and family, and a healthy life. It is minorly affected by freedom to pursue life choices and perceptions of government and business corruption. Being charitable has little to no effect on happiness. How accurate all of this is, however, is explored further through our machine learning algorithms.

Our best machine learning result was using Random Forests, and this gave us an accuracy of 72%. This means that even with our best efforts, we still only accurately predict less than 3 out of every 4 scores. This could either mean that we simply didn't choose an accurate enough machine learning algorithm for our context, or it could mean that a country's happiness goes beyond the variables that we analyzed. For example, respondents believing that they have a "good life" doesn't necessarily mean that they are happy with it.

Conclusion

We set out to see just how accurate the 2018 international report on world happiness was on predicting happiness scores, and I believe we achieved satisfactory results. We managed to show visually the frequencies of our happiness scores as well as the independent variables, and we also compared the scores to these variables. We looked at the how correlated *score* is to our other variables. In our machine learning efforts, we created algorithms in an attempt to predict scores, first by just guessing by using the average score, followed by linear regression, k-nearest neighbors, and random forests. Even our best accuracy from these attempts doesn’t seem accurate enough, implying we should either use different machine learning techniques in the future, or that we are not able to accurately predict happiness with our given variables. If the latter is true, then much more research needs to be done into how to measure happiness.

References

- Wikipedia - World Happiness Report: https://en.wikipedia.org/wiki/World_Happiness_Report (https://en.wikipedia.org/wiki/World_Happiness_Report)
- World Happiness Report 2018: https://worldhappiness.report/ed/2018/ (https://worldhappiness.report/ed/2018/)
- Introduction to Data Science, by Rafael A. Irizarry: https://rafalab.github.io/dsbook/ (https://rafalab.github.io/dsbook/)