World Happiness (draft)

Guillioli, Walter

01 February, 2020

Introduction

What makes us happy? It's about community, family and doing stuff we love. It's also about learning to be at peace with yourself and accept life as it is. The best book I have read on the topic is Happiness: A Guide to Developing Life's Most Important Skill by Matthieu Ricard.
https://www.amazon.com/Happiness-Guide-Developing-Lifes-Important/dp/0316167258

But today let's ake a data science approach to explore it.

This report is written to walk through an example of the lifeycle of a data science project. We will load, explore and prepare data. Then we will use statistics and Machine Learning algorithms to understand why people in some countries are happier than in others. This report is written for a technical audience with a focus on the aspiring data scientist.

Data Overview

We will use the dataset provided by Kaggle. This is the World Happiness Report that was released by the United Nations as is now considered a landmark survey in the state of global happiness. The first release was in 2012 but for this report will use the data for 2019. The data ultimately comes from the Gallup World Poll. For more contest see https://www.kaggle.com/unsdsn/world-happiness.

Load Dataset

First, we load the data and explore the size and structure of the data frame of 156 observations and 9 variables.

```
#Set working directory
#setwd("C:/Users/wguil/OneDrive/Documents/GitHub/world_happiness/")

#Load happiness data for 2019
d2019 <- read.csv("../data/2019.csv", stringsAsFactors = FALSE)

#Make a working copy
d <- d2019

#Size of the data frame
dim(d)
```

```
## [1] 156   9
```

```
#Column names, type of variable and sample values
str(d)
```

```
## 'data.frame':    156 obs. of  9 variables:
##  $ Overall.rank               : int  1 2 3 4 5 6 7 8 9 10 ...
```

```
##  $ Country.or.region           : chr  "Finland" "Denmark" "Norway" "Iceland" ...
##  $ Score                       : num  7.77 7.6 7.55 7.49 7.49 ...
##  $ GDP.per.capita              : num  1.34 1.38 1.49 1.38 1.4 ...
##  $ Social.support              : num  1.59 1.57 1.58 1.62 1.52 ...
##  $ Healthy.life.expectancy     : num  0.986 0.996 1.028 1.026 0.999 ...
##  $ Freedom.to.make.life.choices: num  0.596 0.592 0.603 0.591 0.557 0.572 0.574 0.585 0.584 0.532 ..
##  $ Generosity                  : num  0.153 0.252 0.271 0.354 0.322 0.263 0.267 0.33 0.285 0.244 ...
##  $ Perceptions.of.corruption   : num  0.393 0.41 0.341 0.118 0.298 0.343 0.373 0.38 0.308 0.226 ...
```

```
#Change column names to friendlier and shorter names
colnames(d) <- c("rank", "country", "score", "gpd_pc", "social_support",
                 "life_expectancy", "freedom" , "generosity", "corruption")

#Sample of 10 observations
kable(d[1:10,], row.names = FALSE)
```

rank	country	score	gpd_pc	social_support	life_expectancy	freedom	generosity	corruption
1	Finland	7.769	1.340	1.587	0.986	0.596	0.153	0.393
2	Denmark	7.600	1.383	1.573	0.996	0.592	0.252	0.410
3	Norway	7.554	1.488	1.582	1.028	0.603	0.271	0.341
4	Iceland	7.494	1.380	1.624	1.026	0.591	0.354	0.118
5	Netherlands	7.488	1.396	1.522	0.999	0.557	0.322	0.298
6	Switzerland	7.480	1.452	1.526	1.052	0.572	0.263	0.343
7	Sweden	7.343	1.387	1.487	1.009	0.574	0.267	0.373
8	New Zealand	7.307	1.303	1.557	1.026	0.585	0.330	0.380
9	Canada	7.278	1.365	1.505	1.039	0.584	0.285	0.308
10	Austria	7.246	1.376	1.475	1.016	0.532	0.244	0.226

One data point I would like to have is the continent of each country. I want to compare happiness in let's say Latin America and Europe. I noticed that region is present in the data from 2015 so I will add this to the data frame.

```
#Load happiness data for 2015
d2015 <- read.csv("../data/2015.csv", stringsAsFactors = FALSE)

#Add Region to my dataset by merging by country name
d <- merge(d, d2015[,c("Country", "Region")], by.x = "country",
           by.y = "Country", all.x = TRUE)

#Change col name so every column is lower case for consistency
names(d)[names(d) == "Region"] <- "region"

#Let's look at region names and their count of countries
d %>%
  count(region, sort = TRUE)
```

```
## # A tibble: 11 x 2
##    region                          n
##    <chr>                       <int>
##  1 Sub-Saharan Africa             36
##  2 Central and Eastern Europe     28
```

```
##  3 Latin America and Caribbean        20
##  4 Western Europe                     20
##  5 Middle East and Northern Africa    19
##  6 Southeastern Asia                   9
##  7 Southern Asia                       7
##  8 <NA>                                7
##  9 Eastern Asia                        6
## 10 Australia and New Zealand           2
## 11 North America                       2
```

```
#It seems 7 countries don't have a region so let's see which ones
d[is.na(d$region),]$country
```

```
## [1] "Gambia"            "Namibia"           "North Macedonia"
## [4] "Northern Cyprus"   "Somalia"           "South Sudan"
## [7] "Trinidad & Tobago"
```

```
#Let's add the region to these 7 countries
d[d$country=="Gambia", ]$region <- "Africa"
d[d$country=="Namibia", ]$region <- "Africa"
d[d$country=="North Macedonia", ]$region <- "Central and Eastern Europe"
d[d$country=="Northern Cyprus", ]$region <- "Central and Eastern Europe"
d[d$country=="Somalia", ]$region <- "Africa"
d[d$country=="South Sudan", ]$region <- "Africa"
d[d$country=="Trinidad & Tobago", ]$region <- "Latin America and Caribbean"

#This is not quite what I wanted as I wanted continents, so let's derive a continents column.
#I will use the 7 continents definition but minor adjustments based on areas I want to see
d <- d %>%
  mutate(continent = case_when(region == "Sub-Saharan Africa" ~ "Africa",
                               region == "Middle East and Northern Africa" ~ "Africa",
                               region == "Africa" ~ "Africa",
                               region == "Southeastern Asia" ~ "Asia",
                               region == "Southern Asia" ~ "Asia",
                               region == "Eastern Asia" ~ "Asia",
                               region == "Central and Eastern Europe" ~ "Europe_CEE",
                               region == "Western Europe" ~ "Europe_WE",
                               region == "Latin America and Caribbean" ~ "South America",
                               region == "Australia and New Zealand" ~ "Australasia",
                               region == "North America" ~ "North America"
                               ))

#Let's see what we have
d %>%
  count(continent, sort = TRUE)
```

```
## # A tibble: 7 x 2
##   continent         n
##   <chr>         <int>
## 1 Africa           59
## 2 Europe_CEE       30
## 3 Asia             22
## 4 South America    21
```

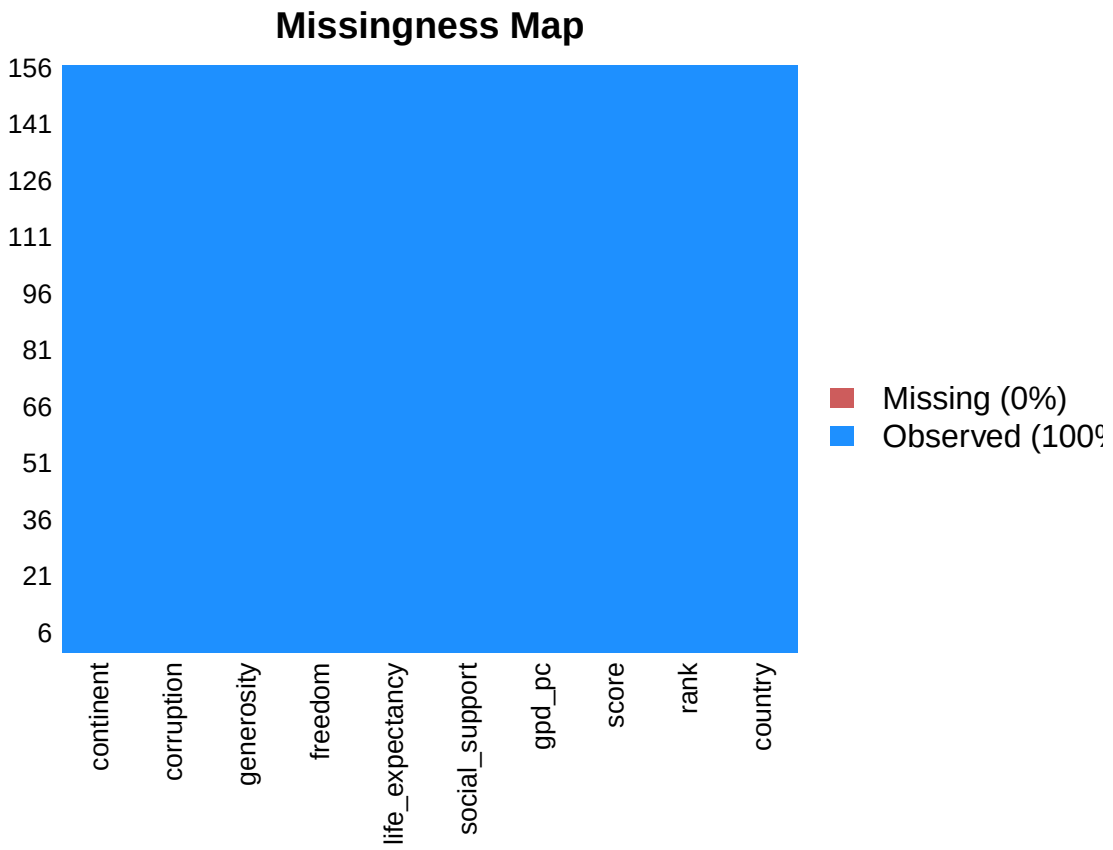
```
## 5 Europe_WE        20
## 6 Australasia       2
## 7 North America     2
```

```
#Drop region since
d <- subset(d, select = -c(region))
```

Missing Values

A key part is validating if there are any missing values in the information. If there is we need to address this. A nice option is the missing values plot from the Amelia package. Fortunately there are not missing values in our data.

```
#Plot missing values
missmap(d)
```



Univariate Data Exploration

It is very important to understand what each column of data is and what type of data we are dealing with. So here we double click on each variable to understand it with summary statistics and plots.

I like to get a list of the variables and it's type and then explore 1x1 since it's a small dataset.

```r
str(d)
```

```
## 'data.frame':    156 obs. of  10 variables:
##  $ country        : chr  "Afghanistan" "Albania" "Algeria" "Argentina" ...
##  $ rank           : int  154 107 88 47 116 11 10 90 37 125 ...
##  $ score          : num  3.2 4.72 5.21 6.09 4.56 ...
##  $ gpd_pc         : num  0.35 0.947 1.002 1.092 0.85 ...
##  $ social_support : num  0.517 0.848 1.16 1.432 1.055 ...
##  $ life_expectancy: num  0.361 0.874 0.785 0.881 0.815 ...
##  $ freedom        : num  0 0.383 0.086 0.471 0.283 0.557 0.532 0.351 0.536 0.527 ...
##  $ generosity     : num  0.158 0.178 0.073 0.066 0.095 0.332 0.244 0.035 0.255 0.166 ...
##  $ corruption     : num  0.025 0.027 0.114 0.05 0.064 0.29 0.226 0.182 0.11 0.143 ...
##  $ continent      : chr  "Asia" "Europe_CEE" "Africa" "South America" ...
```

a) country: the country name should be unique so let's double check no duplicates exist.

```r
sum(duplicated(d$country))
```

```
## [1] 0
```

b) rank: the country rank should go from 1 to 156 and should be unique, let's double check.

```r
summary(d$rank)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##    1.00   39.75   78.50   78.50  117.25  156.00
```

```r
sum(duplicated(d$rank))
```

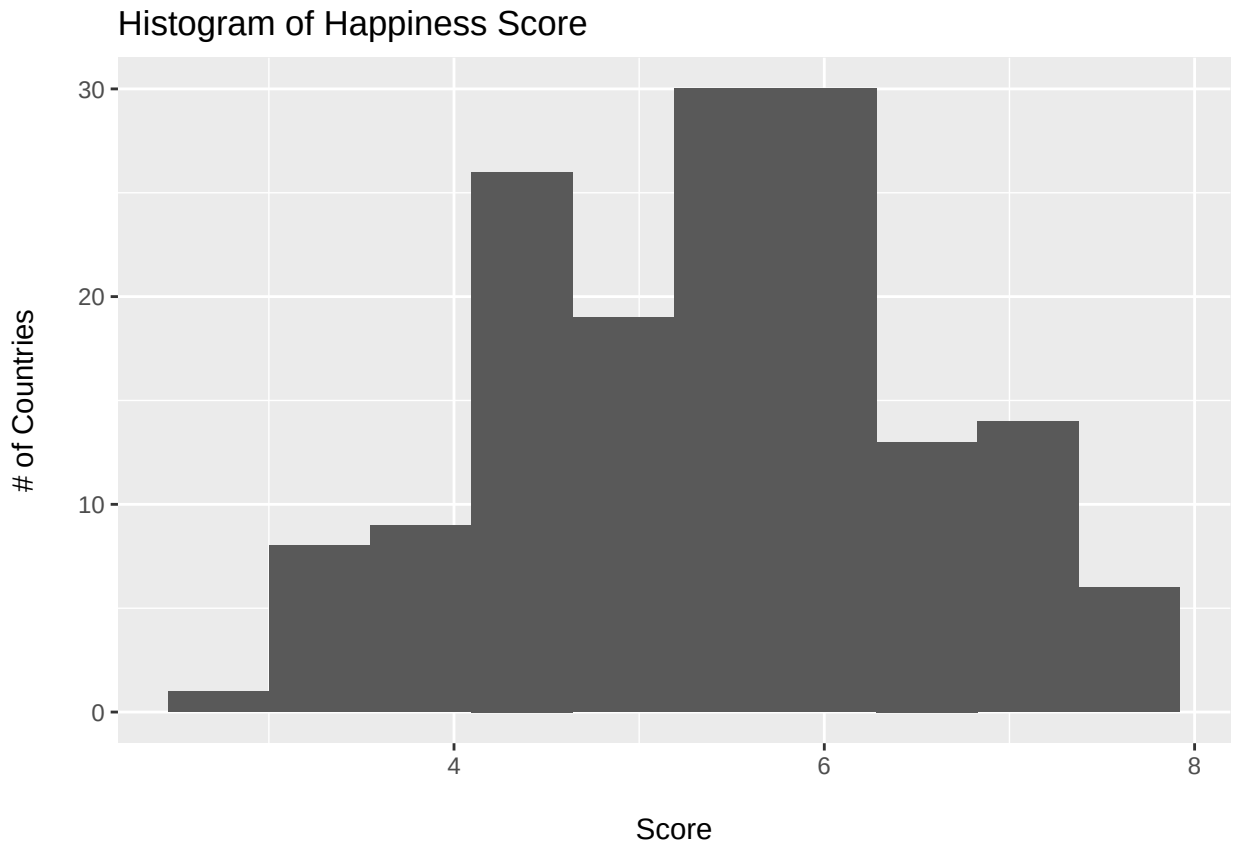
```
## [1] 0
```

c) score: this is the happiness score and the main variable of interest. It ranges from 2.853 to 7.769 and has a very normal distribution, which is really a good thing since we will predict this variable later using Machine Learning algorithms. And some of them, like linear regression performs better on "normal" data.

```r
summary(d$score)
```

```
##    Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
##   2.853   4.545   5.380   5.407   6.184   7.769
```

```r
ggplot(d, aes(score)) +
  geom_histogram(bins = 10) +
  ggtitle("Histogram of Happiness Score") +
  xlab("\nScore") +
  ylab("# of Countries\n")
```



c) gdp, social_support, life_expectancy, freedmon, generosity and corruption: doing what we did for score and the previous two variables is very useful but can become very tedious if we have a long dataset. Ultimately I am just interested in getting to know these variables better. Two complimentary ways to do this is getting the summary statistics to get the min, max and median values for example. Another way is plotting the histograms to get a sense of how the data is distributed.

As we can see from the summary statistics, we see all variables range from 0 to some real number. In the case of gpd the max is 1.684 and in the case of corruption it ranges from 0.453 for example.

```
#variables to explore
eda_variables <- c("gpd_pc", "social_support", "life_expectancy",
                   "freedom", "generosity", "corruption")

#raw summary statistics
summary(d[,eda_variables])
```

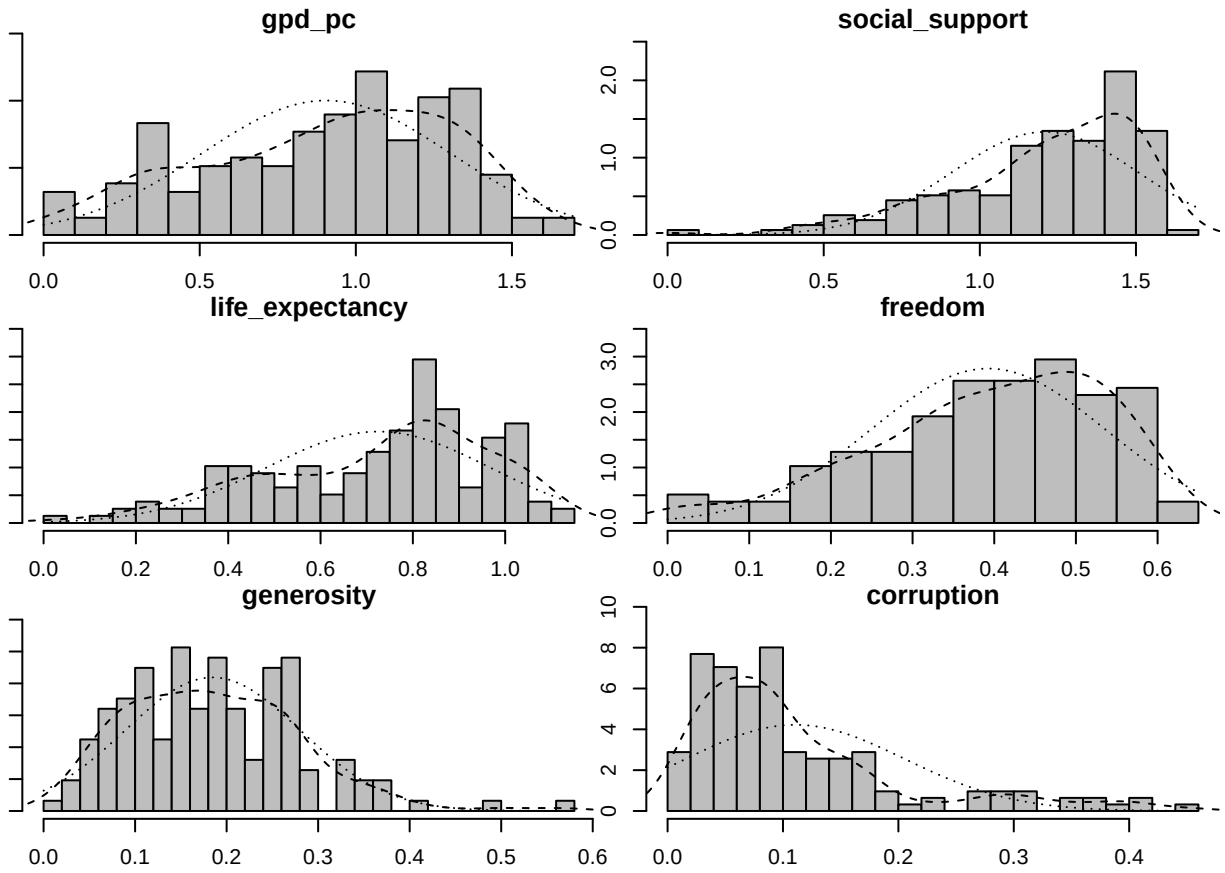
```
##      gpd_pc       social_support  life_expectancy     freedom
##  Min.   :0.0000   Min.   :0.000   Min.   :0.0000   Min.   :0.0000
##  1st Qu.:0.6028   1st Qu.:1.056   1st Qu.:0.5477   1st Qu.:0.3080
##  Median :0.9600   Median :1.272   Median :0.7890   Median :0.4170
##  Mean   :0.9051   Mean   :1.209   Mean   :0.7252   Mean   :0.3926
##  3rd Qu.:1.2325   3rd Qu.:1.452   3rd Qu.:0.8818   3rd Qu.:0.5072
##  Max.   :1.6840   Max.   :1.624   Max.   :1.1410   Max.   :0.6310
##    generosity       corruption
##  Min.   :0.0000   Min.   :0.0000
##  1st Qu.:0.1087   1st Qu.:0.0470
```

```
##  Median :0.1775   Median :0.0855
##  Mean   :0.1848   Mean   :0.1106
##  3rd Qu.:0.2482   3rd Qu.:0.1412
##  Max.   :0.5660   Max.   :0.4530
```

But let's plot this now to see how the data is distributed for these variables.

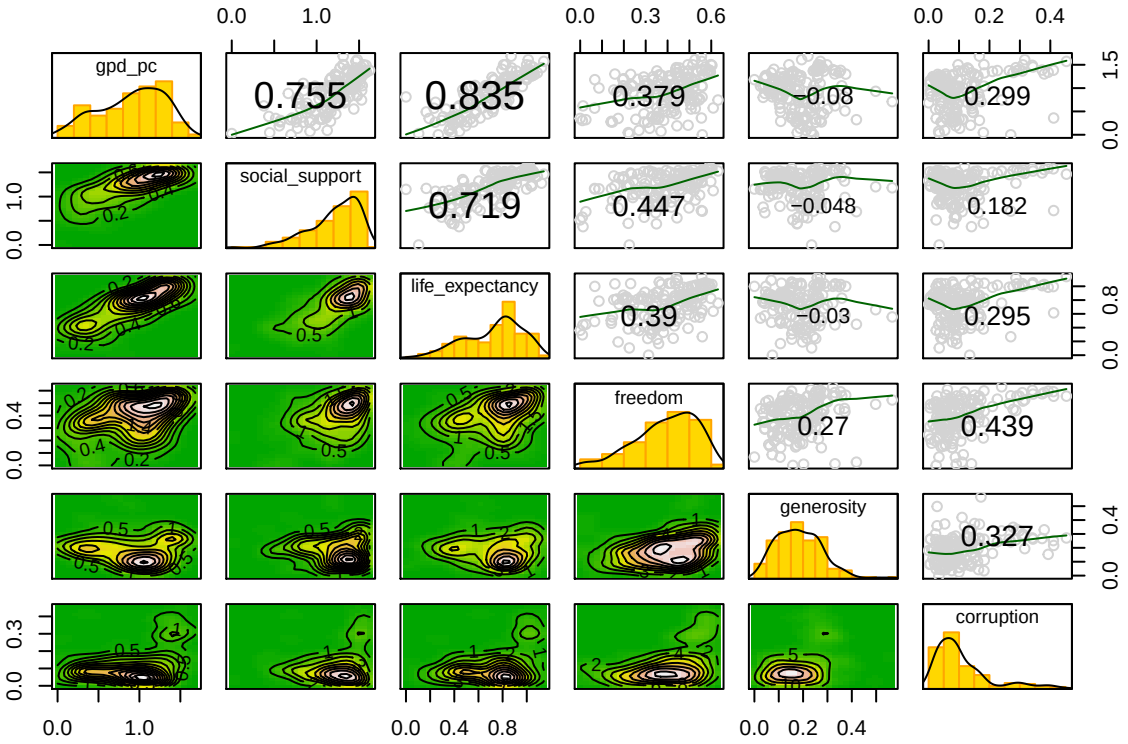
As we can see GDP, freedom and generosity are almost normally distributed. The others skew a bit. We might explore at the end of ths paper if changing the scale of some of these variables could help us predict the happiness score. But let's leave that out for now.

```
multi.hist(d[,eda_variables], bcol = "gray")
```



Another cool chart that can be used to explore distribution of data **and** relationships with each other is this one. This starts to show the correlation with each other but let's not get ahead of ourselves as we will see that later.

```
kdepairs(d[,eda_variables])
```



Methods and Results

In this section we explore in more detail what variables could help explain and predict the happiness score. We beging exploring numerical correlations and then we explore the difference by continent.

As a last step we build ML models to actually predict the happiness score and we compare those models.

Correlations and Prediction Potential of Happiness

Ultimately we want to use all the data provided to predict the happiness score and we will get to that. But before we need to understand our data and see which variables on their own have a strong correlation with the happiness score. We are also interested in variables that have strong correlation with each other since that might create some noise in our ML model down the road and we might need to address it.

A quick first step is to plot a correlation matrix and the values will range from [-1, 1] where extremes represent strong positive/negative correlation.

```
#variables to explore correlations
corr_variables <- c("score", eda_variables)

res <- cor(d[,corr_variables])
kable(round(res,2))
```

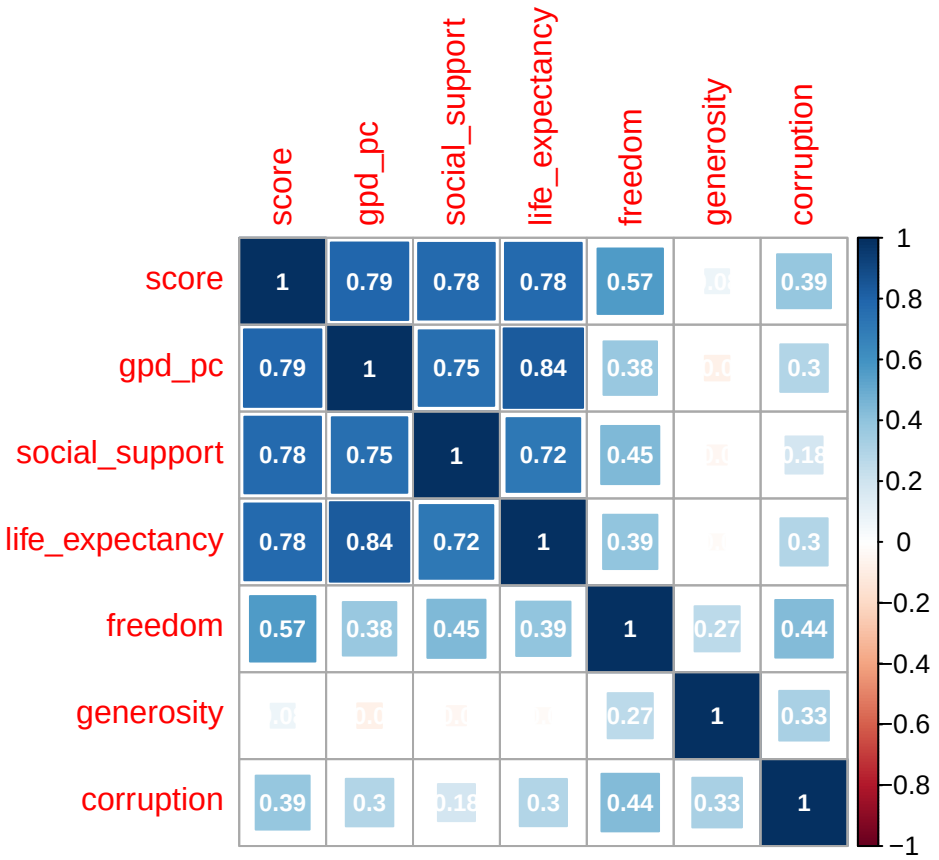
	score	gpd_pc	social_support	life_expectancy	freedom	generosity	corruption
score	1.00	0.79	0.78	0.78	0.57	0.08	0.39
gpd_pc	0.79	1.00	0.75	0.84	0.38	-0.08	0.30
social_support	0.78	0.75	1.00	0.72	0.45	-0.05	0.18
life_expectancy	0.78	0.84	0.72	1.00	0.39	-0.03	0.30
freedom	0.57	0.38	0.45	0.39	1.00	0.27	0.44
generosity	0.08	-0.08	-0.05	-0.03	0.27	1.00	0.33
corruption	0.39	0.30	0.18	0.30	0.44	0.33	1.00

When looking at the score column we see that GDP, social_support and life_expectancy have the highest correlation and therefore the highest predictive potential. Generosity doesn't seem likely to help predict the score.

Also important to highlight the correlation of GPD with social_support and life_expectancy. This is known as multi-collinearity and might cause issues later so we need to be careful when interpreting and addressing.

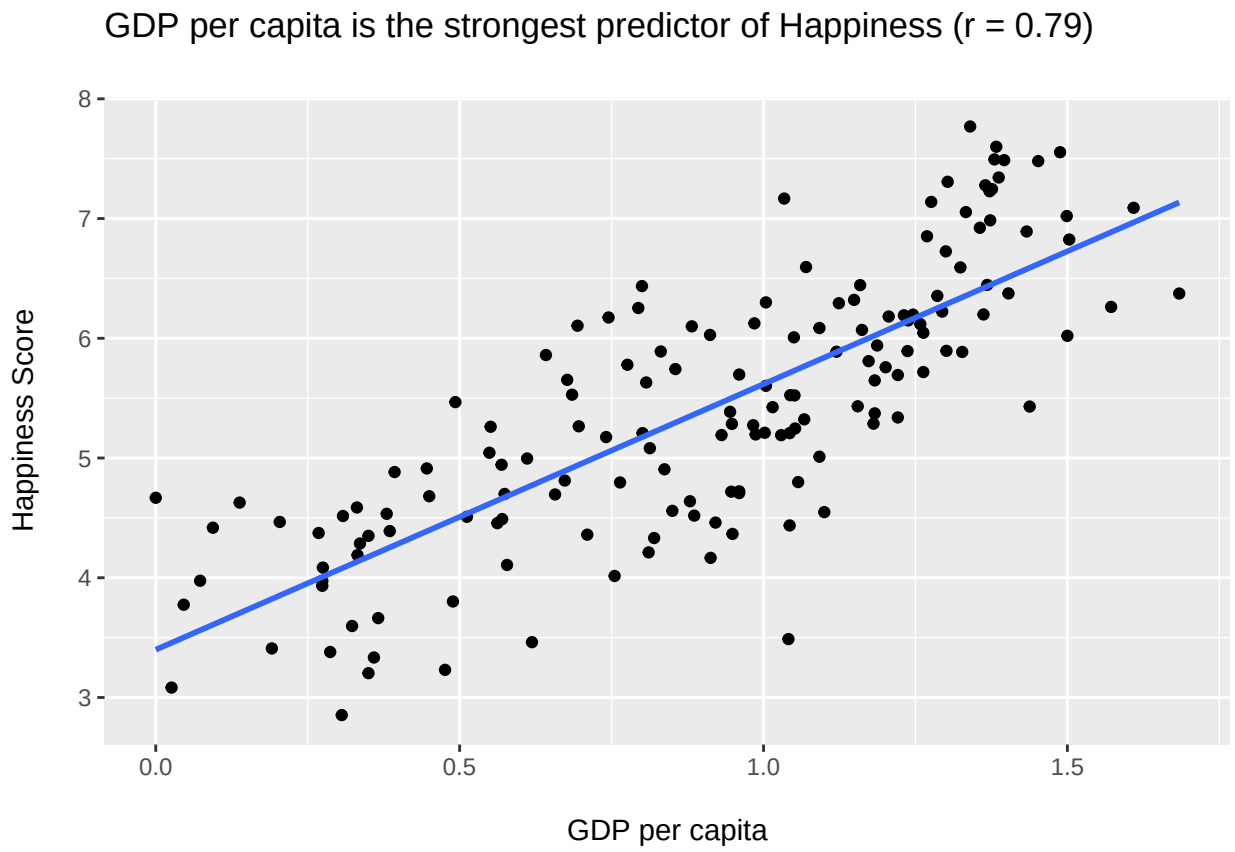
As you can see this is getting tedious and we only have a handful of variables. An easier way to see this information is with a correlation plot.

```
corrplot(res,
         #title = "\nCorrelation Plot of Happiness Score and numerical predictors",
         method = "square",
         addgrid.col = "darkgray",
         addCoef.col = "white",  number.cex = 0.75)
```



Since GDP per capita is the predictor with strongest correlation to the happiness score let's take a look at it. One way to do this is to plot all the points "draw a regression line" on the plot.

```
ggplot(d, aes(x=gpd_pc, y=score)) +
  geom_point() +
  geom_smooth(method=lm, se=FALSE) +
  ggtitle("GDP per capita is the strongest predictor of Happiness (r = 0.79)\n") +
  labs(y="Happiness Score\n", x = "\nGDP per capita")
```



What about continent?

We couldn't include continent as predictor about because it is not numerical so we can't get a correlation. A good thing to do is explore how the values of our variable of interest (happiness score) varies by values of region. I like to see the median and the different shape of the data.

We can see Western Europe, North American and Australasia leading the pack with the highest scores while Africa, Asia and Central & Eastern Europe leading behind. I guess that is no surprise.

```
#Median by continent
kable(d %>%
  group_by(continent) %>%
  summarise(median = median(score), n = n()) %>%
  arrange(desc(n)))
```

continent	median	n
Africa	4.5160	59
Europe_CEE	5.5660	30
Asia	5.1915	22
South America	6.1050	21
Europe_WE	7.0375	20
Australasia	7.2675	2
North America	7.0850	2

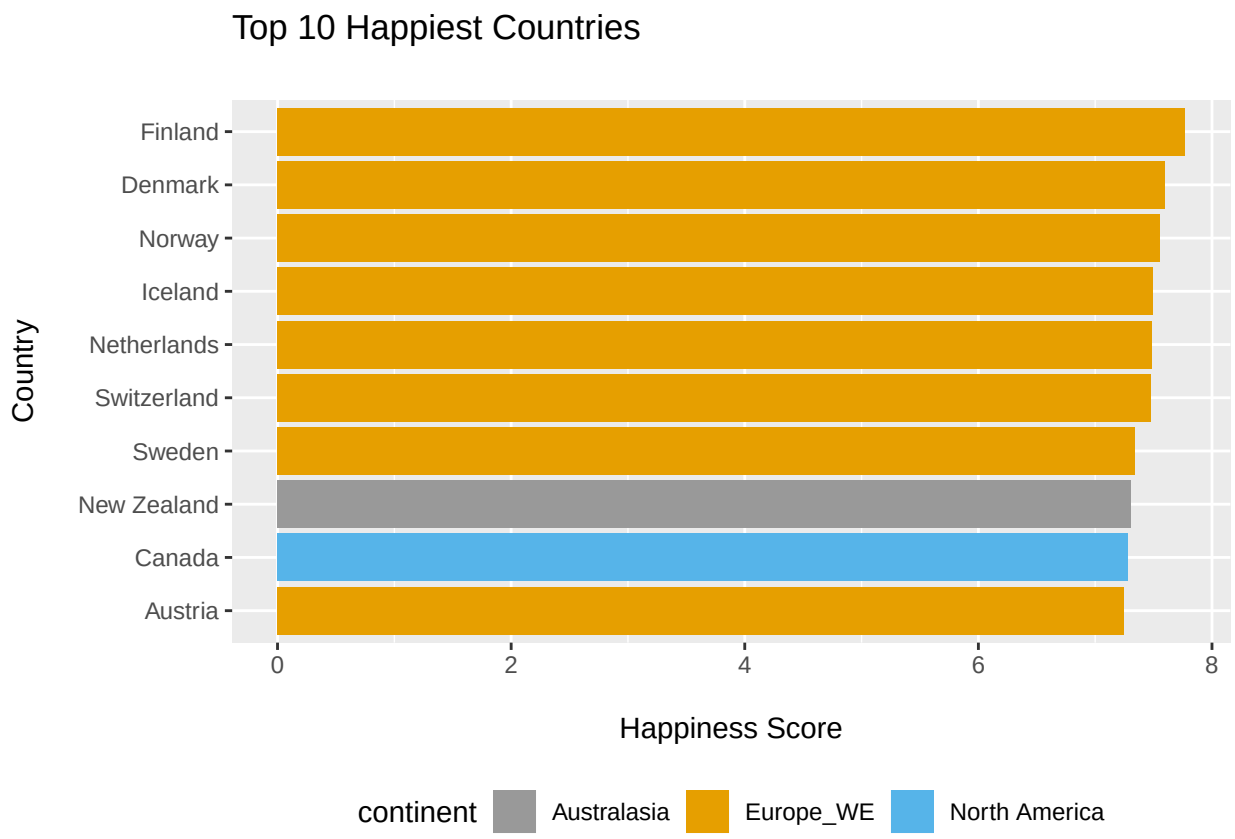
```
#Box plot to see a bit more than just the median
ggplot(d, aes(x = reorder(continent, score, FUN = median), y = score, fill = continent)) +
  geom_boxplot() +
  labs(title="Plot of Happiness Score by Continent\n",
       x="Continent\n", y = "\nHappiness Score") +
  coord_flip()
```

Happiest and saddest countries

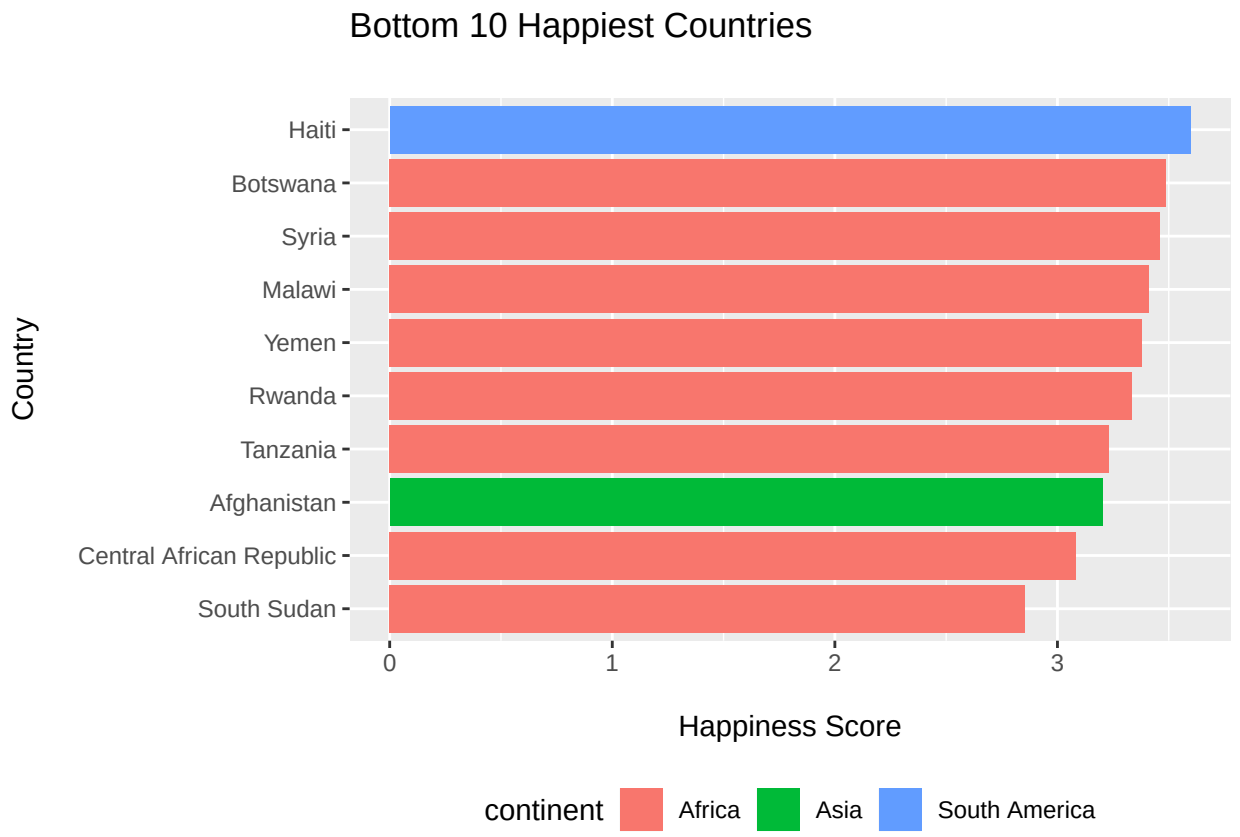
Out of curiosity let's see what are the countries with the top scores for happiness. As expected mostly dominated by Western Europe and North Americ with scores above 7.

```
#Top 10 countries by happiness score
d %>%
```

```
arrange(desc(score)) %>%
slice(1:10) %>%
ggplot(., aes(x = reorder(country, score), y=score, fill = continent)) +
geom_bar(stat="identity") +
labs(title="Top 10 Happiest Countries\n",x="Country\n", y = "\nHappiness Score") +
coord_flip() +
scale_fill_manual(values=c("#999999", "#E69F00", "#56B4E9")) +
theme(legend.position="bottom")
```



But how about the bottom 10? As expected it is mostly Africa and Asia with scores below 4.



Using ML to predict happiness

Now, this is the fun part. Can we actually use the data provided to predict the happiness score? If yes, with how much precision? For this we will be using several ML algorithms and we will be comparing each other.

Doing it all wrong

But first let's do it all wrong. A typical error is to use all the data and predict on the same data. This is not good because we just don't know how well the model would perform with new data. It might just overfit on exisiting data. But let's do it anyway with a simple liner regression. As can be seen and as expected the variables that help predict are GDP, social support, life expectancy and freedom. Generosity and corruption are not necessarily good predictors.

```
#Fit model
lm1 <- lm(score ~ gpd_pc + social_support + life_expectancy + freedom
          + generosity + corruption,
          data=d)

#summary(lm1)
```

But how good is this model? One way to measure this is with Adjusted R2 which basically explains how much of the variance is captured by the model so the higher and closer to 100% the better. In this case we get a decent 77%.

```r
#Adj R2
summary(lm1)$adj.r.squared
```

```
## [1] 0.7702711
```

Doing it right

The right way to do this is using part of the data to build the model and use the remaining data to test the model. The key is to see how good a model can be with new data. A simple way to do this is to do a one time train and one time test of the model. But here I will do this many times using cross-validation approach. I will do linear regression and random forest and will compare each other.

a) Linear Regression

```r
# Define training control
set.seed(97701)
train.control <- trainControl(method = "cv", number = 10)

# Train the model
lm2 <- train(score ~ gpd_pc + social_support + life_expectancy + freedom +
               generosity + corruption,
               data=d,
               method = "lm",
               trControl = train.control)

# Summarize the results
print(lm2)
```

```
## Linear Regression
##
## 156 samples
##   6 predictor
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 142, 140, 141, 140, 140, 141, ...
## Resampling results:
##
##   RMSE       Rsquared   MAE
##   0.5321895  0.7691712  0.427866
##
## Tuning parameter 'intercept' was held constant at a value of TRUE
```

As can be seen this results in an R2 of 0.772. But what is the final model? In linear regression we get an equation with coefficients and these are:

```r
#Get coefficients
lm2$finalModel
```

```
##
## Call:
## lm(formula = .outcome ~ ., data = dat)
```

```
##
## Coefficients:
##     (Intercept)           gpd_pc   social_support  life_expectancy
##          1.7952           0.7754           1.1242           1.0781
##         freedom       generosity       corruption
##          1.4548           0.4898           0.9723
```

b) Random Forest Let's do a Random Forest which is nothing more than several trees.

```
# Train the model
rf1 <- train(score ~ gpd_pc + social_support + life_expectancy + freedom +
               generosity + corruption, #continent
               data=d,
               method = "cforest",
               trControl = train.control)

# Summarize the results
print(rf1)
```

```
## Conditional Inference Random Forest
##
## 156 samples
##   6 predictor
##
## No pre-processing
## Resampling: Cross-Validated (10 fold)
## Summary of sample sizes: 140, 141, 141, 140, 140, 141, ...
## Resampling results across tuning parameters:
##
##   mtry  RMSE       Rsquared   MAE
##   2     0.4960117  0.8102408  0.3880707
##   4     0.4922534  0.8062871  0.3865762
##   6     0.4980470  0.8003798  0.3892507
##
## RMSE was used to select the optimal model using the smallest value.
## The final value used for the model was mtry = 4.
```

As can be seen by the results we get a slight improvement when mtry == 2 for an R2 of 0.820. mtry is simply how many variables are tested at each tree. We can see the main variables selected by Random Forest here.

```
varImp(rf1$finalModel)
```

```
##                    Overall
## gpd_pc          0.350760917
## social_support  0.403160591
## life_expectancy 0.290922968
## freedom         0.068430038
## generosity      0.008300556
## corruption      0.043449951
```

Since it seems Random Forest is a better algorithm let's explore visually how this looks like. In this plot we can see the actual value of the happiness scores vs the predicted ones.

```r
#Make a data frame of the actual score and the prediction
res_df <- as.data.frame(cbind(d$score, predict(rf1$finalModel)))
colnames(res_df) <- c("score_actual", "score_predicted")

#Plot the results
ggplot(res_df, aes(x=score_actual, y=score_predicted)) +
  geom_point() +
  geom_smooth(method=lm, se=FALSE) +
  ggtitle("Actual vs Predicted Happiness Scores\n") +
  labs(y="Predicted Score\n", x = "\nActual Score") +
  xlim(2,8) +
  ylim(2,8)
```

The blue line indicates a perfect fit. The fact that we see points deviating from the line means that there is an error in the prediction that we can't measure in the data we have. That is why we have an R2 of around 82%.

Conclusion

Happiness in the only logical pursuit of life. There are many things we need to do as humans such as having strong connections t our family and communities. We should also spend time doing hobbies we are passionate about and we should work on projects that are important to us (that is why I did this paper). But it seems happiness can be measured.

In this short paper we explored happiness scores of countries in the world and we used data from those countries to explain and predict this score.

It seems that how rich the countries are (GDP) is the most important factor but this is closely followed by how healhy and free people can be in the countries. These could be argued to be related to wealth.

When modeling data we should always remember we are trying to simplify the world in an equation or an algorithm and that is hard to do. So we should treat this models carefuly. George Box said it best **"All models are wrong, but some are useful"**.

References

1. Kaggle. World Happiness Data. https://www.kaggle.com/unsdsn/world-happiness
2. Datahub. Countryand Continent Codes. https://bit.ly/3aZioCS

About the Author

Walter Guillioli is a Data Scientist that loves to solve real life and business problems through data. Walter spent 14+ years at Microsoft helping Marketing teams make better investments and marketing strategies through the use of data.

Walter founded and directs a non-profit called People Saving Animals to help homeless dogs and cats in Central America.

Walter lives in Central Oregon and spends most of his time outside running on trails or spending time with his wife Alejandra and his little troublemaker Diego.

Reach Walter at wguillioli@gmail.com or at https://www.linkedin.com/in/walter-guillioli-69090038/