資料分析與學習基石

HW1:First visit in Kaggle data

基本資料集描述

資料集名稱:

World Happiness Report

資料集簡介:

世界幸福報告（World Happiness Report）為聯合國為衡量幸福的可持續發展方案，第一份報告於2012發佈，而這份資料集目前共含2015至2022多個國家人民經濟、健康、對政府信賴程度、自認幸福程度等相關資料。

政府、組織和公民社會越來越多列入幸福指標來作為政策、決策的參考項目，該報告因而在全球獲得越來越多認可，跨領域的領先專家-經濟學、心理學、調查分析、國家統計、健康、公共政策等領域的專家-也描述了幸福衡量可以如何有效地用於評估國家的進步。

資料集為CSV格式("年份.csv")

資料集欄位:

Country	國家名稱（共156個)
Region	地區(洲)

Happiness Rank	依據Happines Score 將各國從排名(1~156)
Happiness Score	通過抽樣調查方式，詢問該國人民“從0到10你會如何評價你的幸福感，其中10是最幸福的。”
Economy(GDP per Capita)	人均國內生產總值
Family	國家重視家庭程度
Social Support	政府給予人民的幫助程度
Health (Life Expectancy)	人民預期壽命
Freedom to Make Life Choices	民主與自由選擇程度
Trust (Perception of Government Corruption)	人民對於政府的信賴程度(貪汙方面)
Generosity	人民慷慨程度

資料特性與可能產出:

此資料集從2015年更新至今，可根據分析對象的範圍、時間，或是與其他資料集結合利用，有許多種使用方式，且其缺失率極低，資料相當完整，因此可以持續深入研究多種相關議題。

例如透過觀察資料集中個因素與幸福程度的關係，我們可以得知各因素的影響程度，政府、社會組織等可由此得知需更加關注哪些層面，並針對其規劃政策。

將國家劃分至不同區域，分析出各區域的各個因素影響其幸福程度的關係圖，得知各地區影響最大的因素。

若有國家在某一年的幸福程度上發生劇變，可聚焦在單一國家的歷年資料，觀察其變動因素同時可結合該時間國家發生之重大事件甚至是結合其他資料集，分析出事件與各因素間的關係。

eg. Happiness 2017(Visualization + Prediction

World Happiness Analysis & Covid-19 Correlation

現有分析方法介紹與比較

Notebook 1

Are you Happy?|Analysis on World Happiness Report

目標:

- 視覺化分析此資料集
- 以Happiness Score為dependent variable建立多個regression model
- 得出影響人民幸福度的因素，政府可著重於那些方面增進國家福祉

步驟:

Importing Libraries and Exploring Data

導入會用到的python module

- pandas: 用於表格數據操作
- pyplot, seaborn: 用於資料視覺化
- np: 用於矩陣計算
- sklearn: 用於regression model

```
#load packages
import pandas as pd
import matplotlib.pyplot as plt
%matplotlib inline
import matplotlib as mpl
import seaborn as sns
import matplotlib.pylab as pylab
import numpy as np
import sklearn
import warnings
warnings.filterwarnings('ignore')
print('-'*25)
from subprocess import check_output
print(check_output(["ls", "../input"]).decode("utf8"))
```

Load Data

以pandas讀取csv格式的dataset，簡單觀察歷年資料

```
year_2015=pd.read_csv("../input/world-happiness/2015.csv")
year_2016=pd.read_csv("../input/world-happiness/2016.csv")
year_2017=pd.read_csv("../input/world-happiness/2017.csv")
year_2018=pd.read_csv("../input/world-happiness/2018.csv")
year_2019=pd.read_csv("../input/world-happiness/2019.csv")
```

先取用最新的資料(當時為2019)作分析，之後再結合其他年份做觀察

- **尋找並處理missing data**

```
print('Data columns with null values:',year_2019.isnull().sum
(), sep = '\n')
```

```
Data columns with null values:
Overall rank                    0
Country or region               0
Score                           0
GDP per capita                  0
Social support                  0
Healthy life expectancy         0
Freedom to make life choices    0
Generosity                      0
Perceptions of corruption       0
dtype: int64
```

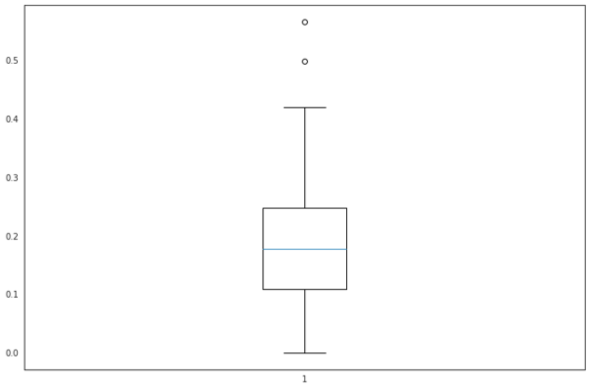
-> 結果並沒有遺失的data

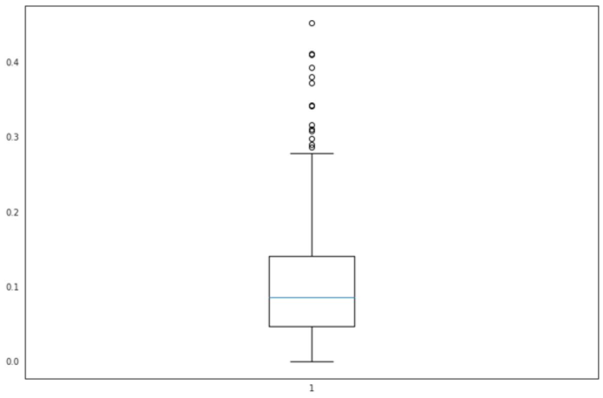
Data Wrangling

- **尋找並處理Outlier(離群值)**

離群值會增加數據的離散度(Variability)，降低其統計檢定力(Statistical Power)，因此，須排除離群值將結果變得具有統計顯著性(statistically significant)

```
year_2019_columns=['Generosity','Perceptions of corruption']
for i in year_2019_columns:
    plt.boxplot(year_2019[i])
    plt.show()
```

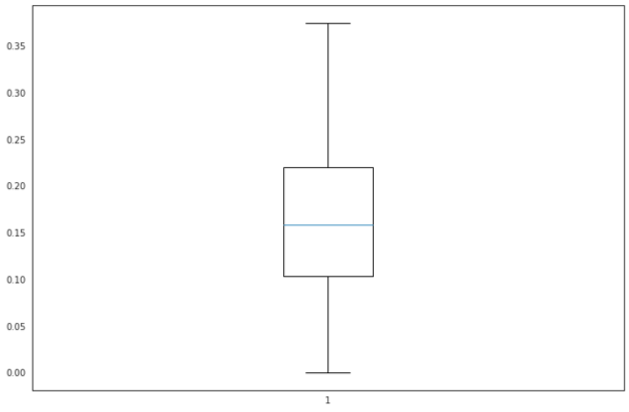


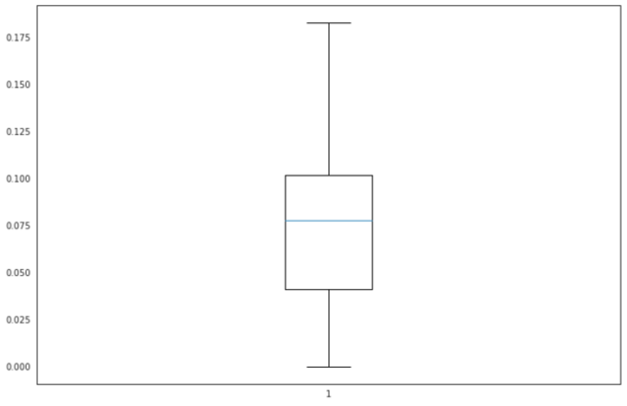


找到了具有離群值的兩個類別，利用四分位距將離群值過濾掉 (利用quantile()計算出特定趴數的累積值)

```
Q1 = year_2019['Generosity'].quantile(0.25)
Q3 = year_2019['Generosity'].quantile(0.75)
IQR = Q3 - Q1    #IQR is interquartile range.

filter = (year_2019['Generosity'] >= Q1 - 1.5 * IQR) & (year_20
19['Generosity'] <= Q3 + 1.2 *IQR)
year_2019_1=year_2019.loc[filter]

Q1 = year_2019_1['Perceptions of corruption'].quantile(0.25)
Q3 = year_2019_1['Perceptions of corruption'].quantile(0.75)
IQR = Q3 - Q1    #IQR is interquartile range.

filter = (year_2019_1['Perceptions of corruption'] >= Q1 - 1.5
* IQR) & (year_2019_1['Perceptions of corruption'] <= Q3 + 0.6
*IQR)
cleaned_2019=year_2019_1.loc[filter]
```

```
year_2019_columns=['Generosity','Perceptions of corruption']
for i in year_2019_columns:
    plt.boxplot(cleaned_2019[i])
    plt.show()
```





Feature Engineering

先將各類別歷年資料分別串接

```
# 5a. Happiness score(year-wise)
data_1=[year_2019['Country or region'],year_2015['Happiness Score'],year_2016['Happiness Score'],year
_2017['Happiness.Score'],
       year_2018['Score'],year_2019['Score']]
headers_1=["Country","2015","2016","2017","2018","2019"]
score = pd.concat(data_1, axis=1, keys=headers_1,join='inner')

# 5b. GDP per capita score (year-wise)
data_2=[year_2019['Country or region'],year_2015['Economy (GDP per Capita)'],year_2016['Economy (GDP
per Capita)'],year_2017['Economy..GDP.per.Capita.'],
       year_2018['GDP per capita'],year_2019['GDP per capita']]
gdp = pd.concat(data_2, axis=1, keys=headers_1,join='inner')

# 5c. Health Life Expectancy score (year-wise)
data_3=[year_2019['Country or region'],year_2015['Health (Life Expectancy)'],year_2016['Health (Life
Expectancy)'],year_2017['Health..Life.Expectancy.'],
       year_2018['Healthy life expectancy'],year_2019['Healthy life expectancy']]
life_exp = pd.concat(data_3, axis=1, keys=headers_1,join='inner')

# 5d. Freedom score (year-wise)
data_4=[year_2019['Country or region'],year_2015['Freedom'],year_2016['Freedom'],year_2017['Freedo
m'],
       year_2018['Freedom to make life choices'],year_2019['Freedom to make life choices']]
freedom = pd.concat(data_4, axis=1, keys=headers_1,join='inner')

# 5d. Generosity score (year-wise)
data_5=[year_2019['Country or region'],year_2015['Generosity'],year_2016['Generosity'],year_2017['Gen
erosity'],
       year_2018['Generosity'],year_2019['Generosity']]
generosity = pd.concat(data_5, axis=1, keys=headers_1,join='inner')
```

再將所有類別整合

```
# Year 2019
year_2019.columns = ["rank","region","score",
                     "gdp_per_capita","social_support","healthy_life_expectancy",
                    "freedom_to_life_choice","generosity","corruption_perceptions"]

# Year 2018
year_2018.columns = ["rank","region","score",
                     "gdp_per_capita","social_support","healthy_life_expectancy",
                    "freedom_to_life_choice","generosity","corruption_perceptions"]

pd.set_option('display.width', 500)
pd.set_option('display.expand_frame_repr', False)

# Year 2017
year_2017.drop(["Whisker.high","Whisker.low",
            "Family","Dystopia.Residual"],axis=1,inplace=True)

year_2017.columns =  ["region","rank","score",
                     "gdp_per_capita","healthy_life_expectancy",
                    "freedom_to_life_choice","generosity","corruption_perceptions"]
```

```
# Year 2016
year_2016.drop(['Region','Lower Confidence Interval','Upper Confidence Interval',
              "Family",'Dystopia Residual'],axis=1,inplace=True)

year_2016.columns = ["region","rank","score",
                     "gdp_per_capita","healthy_life_expectancy",
                    "freedom_to_life_choice","corruption_perceptions","generosity"]

# Year 2015
year_2015.drop(["Region",'Standard Error', 'Family', 'Dystopia Residual'],axis=1,inplace=True)
year_2015.columns = ["region", "rank", "score",
                        "gdp_per_capita","healthy_life_expectancy",
                        "freedom_to_life_choice", "corruption_perceptions", "generosity"]

#Adding year to all the dataframe
year_2015["year"] = 2015
year_2016["year"] = 2016
year_2017["year"] = 2017
year_2018["year"] = 2018
year_2019["year"] = 2019

final_happy = year_2015.append([year_2016,year_2017,year_2018,year_2019])
final_happy=final_happy.drop(['freedom_to_life_choice','freedom_to_life_choice','social_support'],axi
s=1)
final_happy.dropna(inplace=True)
final_happy.head()
```

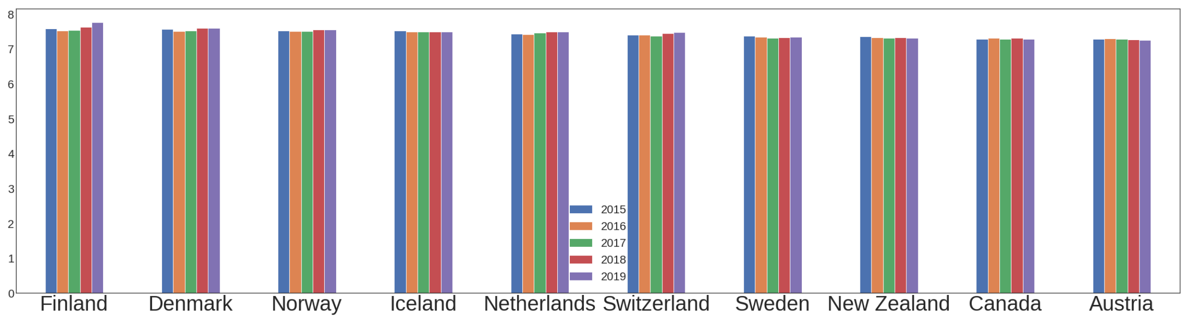
	region	rank	score	gdp_per_capita	healthy_life_expectancy	corruption_perceptions	generosity	year
0	Switzerland	1	7.587	1.39651	0.94143	0.41978	0.29678	2015
1	Iceland	2	7.561	1.30232	0.94784	0.14145	0.43630	2015
2	Denmark	3	7.527	1.32548	0.87464	0.48357	0.34139	2015
3	Norway	4	7.522	1.45900	0.88521	0.36503	0.34699	2015
4	Canada	5	7.427	1.32629	0.90563	0.32957	0.45811	2015

Data Visualization

1.Top 10 Happy Countries

利用head()取得score項目的前十項，並畫出長條圖，這裡若是不知道前十國家是哪幾個應該可用資料集中的region項目取得

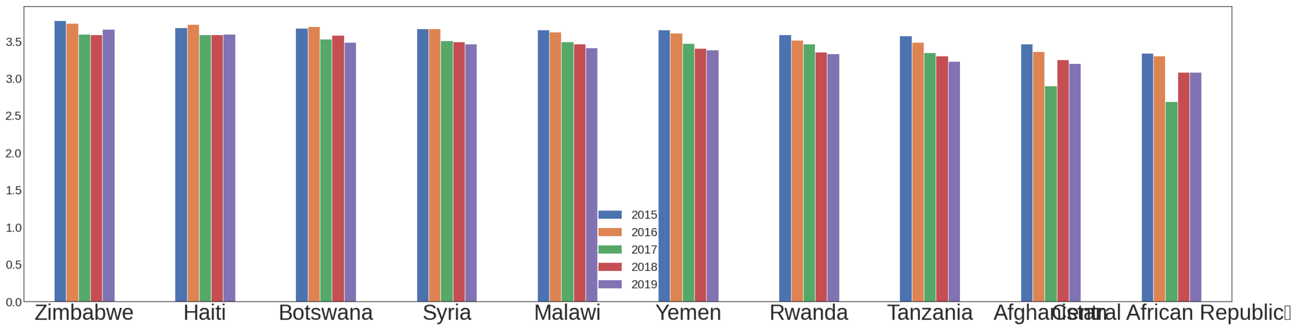
```
sns.set(font_scale = 2)
plt.style.use('seaborn-white')
labels = ['Finland','Denmark','Norway','Iceland','Netherlands','Switzerland','Sweden','New Zealan
d','Canada','Austria']
top_10=score.head(10)
ax=top_10.plot.bar(rot=0)
ax.set_xticklabels(labels)
ax.set_xticklabels(ax.get_xticklabels(), fontsize=40)
fig = plt.gcf()
fig.set_size_inches(40,10)
```



- Finland在四年間有持續上升的趨勢，而Austria則是逐年下降，其他國家無太大變化
- 各國在2017幾乎小幅下降或是持平

2.Last 10 Happy Countries

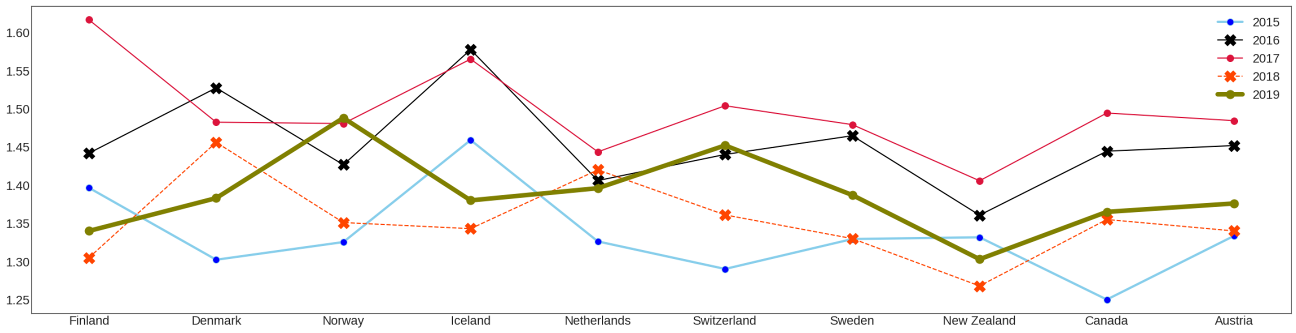
方法與上方相同，改為利用score.tail()



- 四年間，大部分國家幾乎呈現持續下降
- 各國皆在2017有最低數值，但除了Africa和Afghanistan，其他國家隔年皆有上升

3.Top 10 GDP Performances

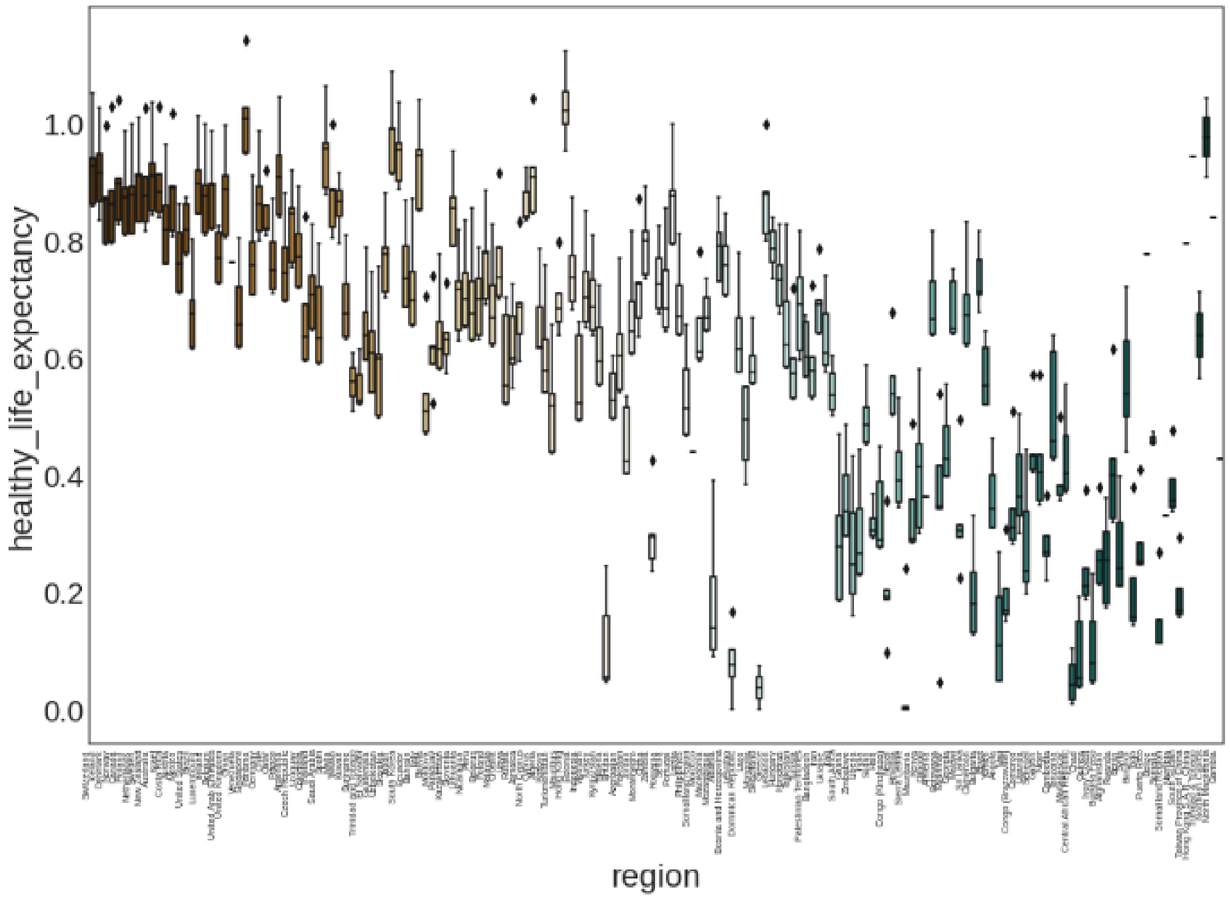
```
gdp_10=gdp.head(10)
plt.plot( 'Country', '2015', data=gdp_10, marker='o', markerfacecolor='blue', markersize=12, color='s
kyblue', linewidth=4)
plt.plot( 'Country', '2016', data=gdp_10, marker='X', color='black', markersize=20, linewidth=2)
plt.plot( 'Country', '2017', data=gdp_10, marker='o', color='crimson', markersize=12, linewidth=2, li
nestyle='solid', label="2017")
plt.plot( 'Country', '2018', data=gdp_10, marker='X', color='orangered', markersize=20, linewidth=2,
linestyle='dashed', label="2018")
plt.plot( 'Country', '2019', data=gdp_10, marker='o', color='olive', markersize=15, linewidth=8, line
style='solid', label="2019")
plt.legend()
fig = plt.gcf()
fig.set_size_inches(40,10)
```



- 整體來看所有國家在2018有顯著較差的GDP
- 2017年時各國有較高的GDP，然而由上方的結果此時各國的幸福程度卻是較低的 → 推測GDP不影響幸福程度嗎?

4.Life Expectancy across Countries

```
ax=sns.boxplot(y='healthy_life_expectancy', x='region',
               data=final_happy,
               palette="BrBG",
               dodge=False)
ax.set_xticklabels(ax.get_xticklabels(), fontsize=7,rotation=90, ha="right")
fig = plt.gcf()
fig.set_size_inches(15,10)
```

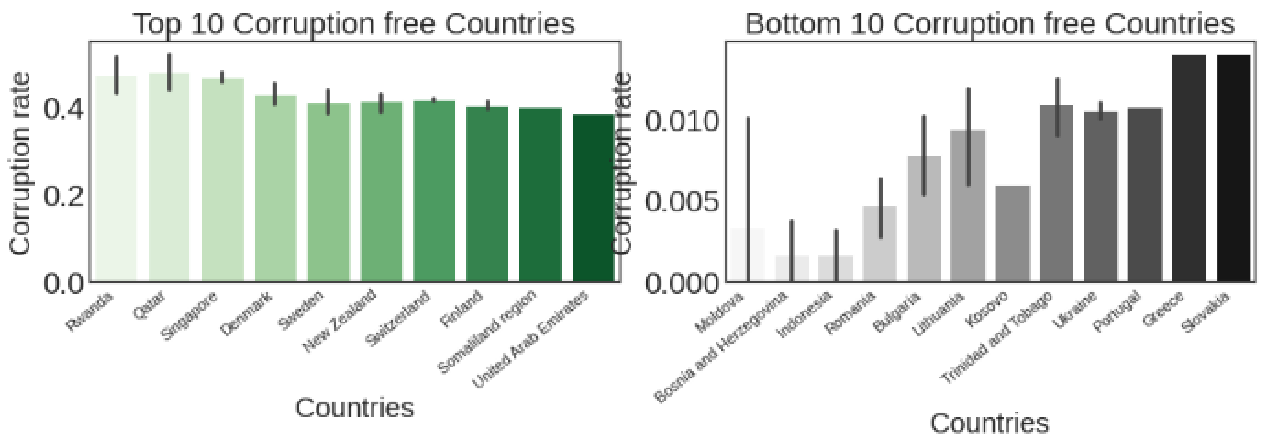


- 國家依序排列後，預期壽命呈現下降趨勢
- 最後一個國家Gambia的預期壽命很高，其其幸福程度卻是最低的

5.Top vs Bottom- Corruption free countries

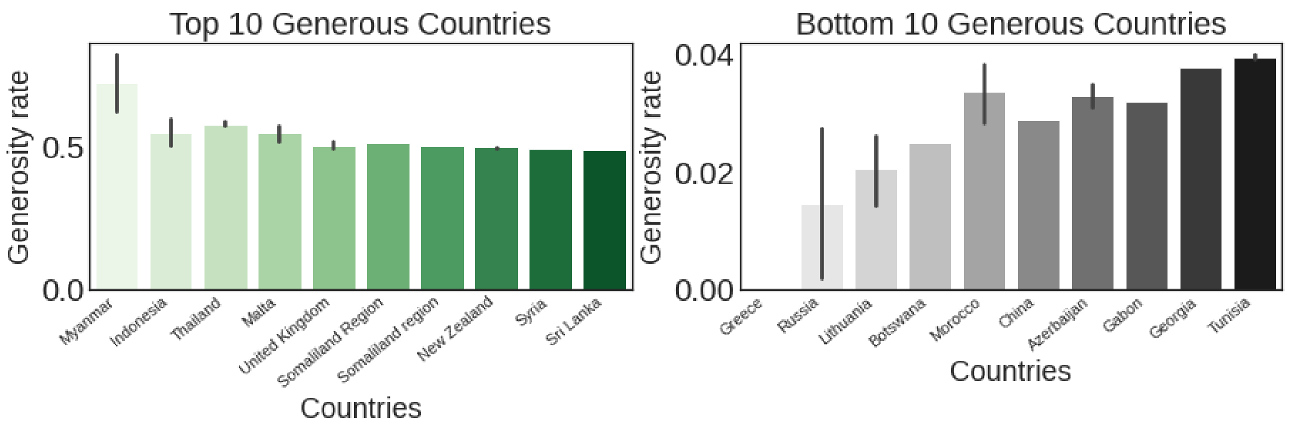
```
plt.figure(figsize = (15, 7))
plt.style.use('seaborn-white')
plt.subplot(2,2,1)
top_10_corrupt=final_happy[['region', 'corruption_perceptions']].sort_values(by = 'corruption_percept
ions',ascending = False).head(32)
ax=sns.barplot(x="region", y="corruption_perceptions", data=top_10_corrupt, palette="Greens")
ax.set_xticklabels(ax.get_xticklabels(),fontsize=11,rotation=40, ha="right")
ax.set_title('Top 10 Corruption free Countries',fontsize= 22)
ax.set_xlabel('Countries',fontsize = 20)
ax.set_ylabel('Corruption rate', fontsize = 20)

plt.subplot(2,2,2)
bot_10_corrupt=final_happy[['region', 'corruption_perceptions']].sort_values(by = 'corruption_percept
ions',ascending = True).head(32)
ax=sns.barplot(x="region", y="corruption_perceptions", data=bot_10_corrupt,palette="Greys")
ax.set_xticklabels(ax.get_xticklabels(),fontsize=11, rotation=40, ha="right")
ax.set_title('Bottom 10 Corruption free Countries',fontsize= 22)
ax.set_xlabel('Countries',fontsize = 20)
ax.set_ylabel('Corruption rate', fontsize = 20)
```

```
Text(0, 0.5, 'Corruption rate')
```



- 雖然Rwanda的經濟較差，卻是在前幾高的
- Qatar和Singapore高於Rwanda可推測是GDP增長的原因

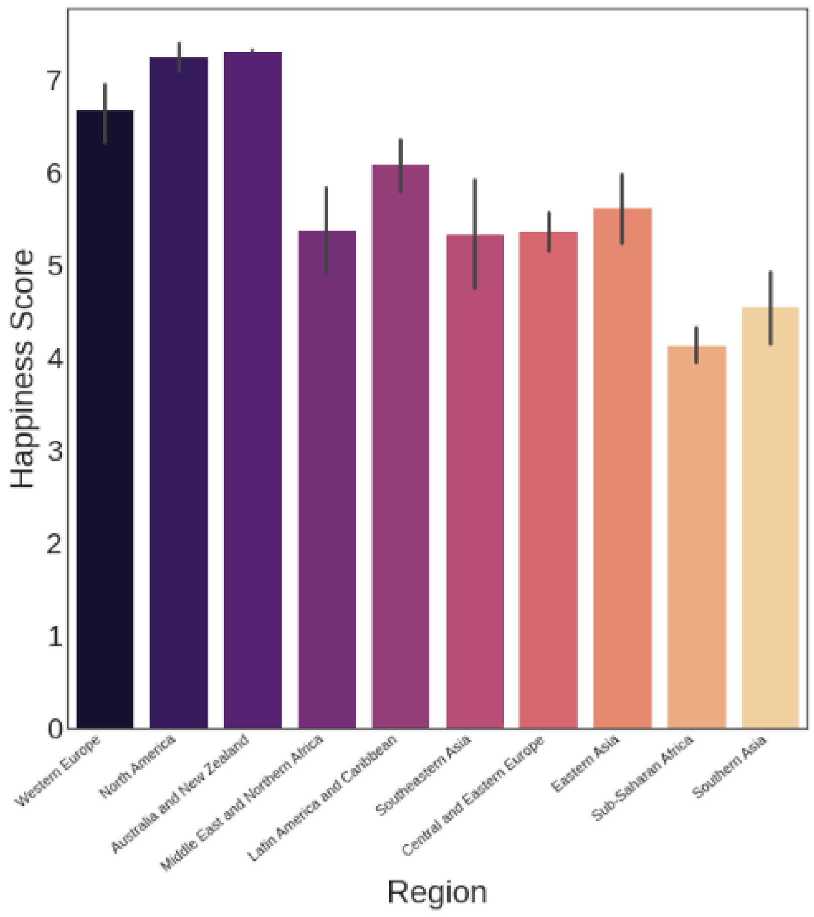
6.Top vs Bottom- Generous countries



7.Happiness Score Based on Region- Bar plot(2016)

```
ax=sns.barplot(x="Region", y="Happiness Score", data=year_2016_copy, palette="magma")
ax.set_xticklabels(ax.get_xticklabels(),fontsize=11, rotation=40, ha="right")
fig = plt.gcf()
```

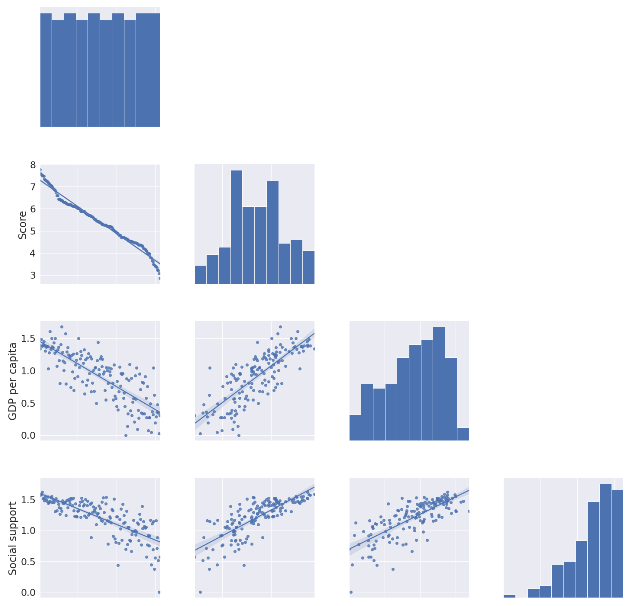
```
fig.set_size_inches(10,10)
```

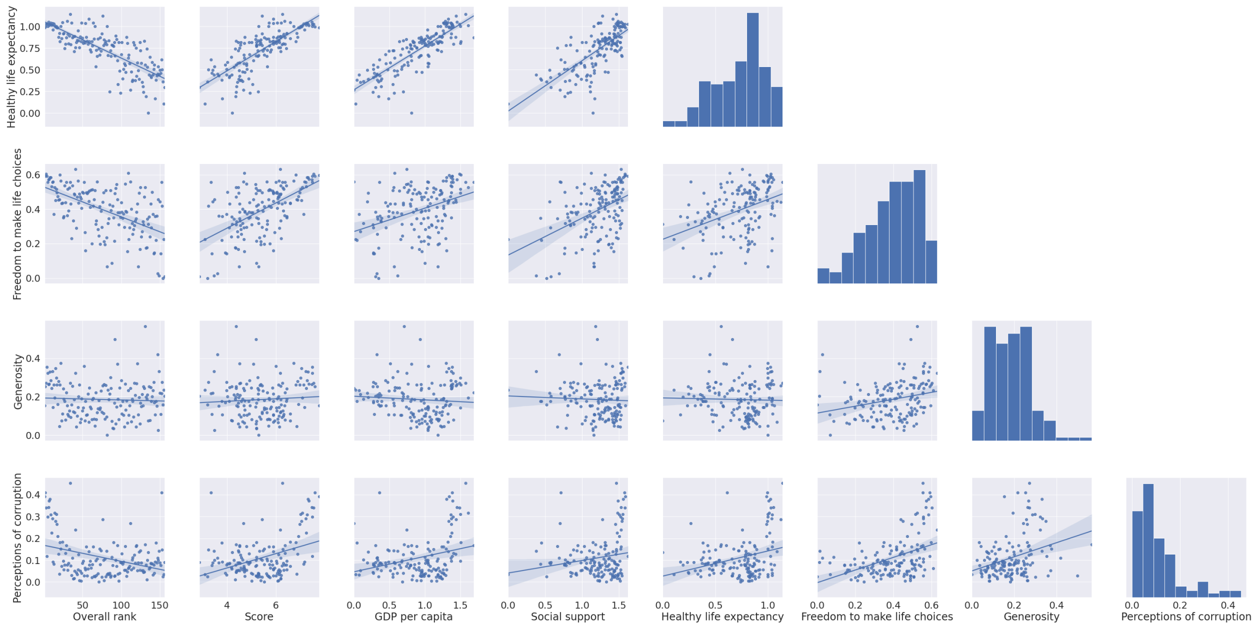


- 儘管西歐國家在幸福程度前十名佔據了不少，但北美與紐澳卻累積了更高的幸福程度
- 如預想的非洲因貧窮與飢荒問題位居最後

8.Relationship between factors-Pair plot(2019)

```
sns.set(font_scale = 2)
sns.pairplot(year_2019_copy,size=5,corner=True,kind="reg")
```

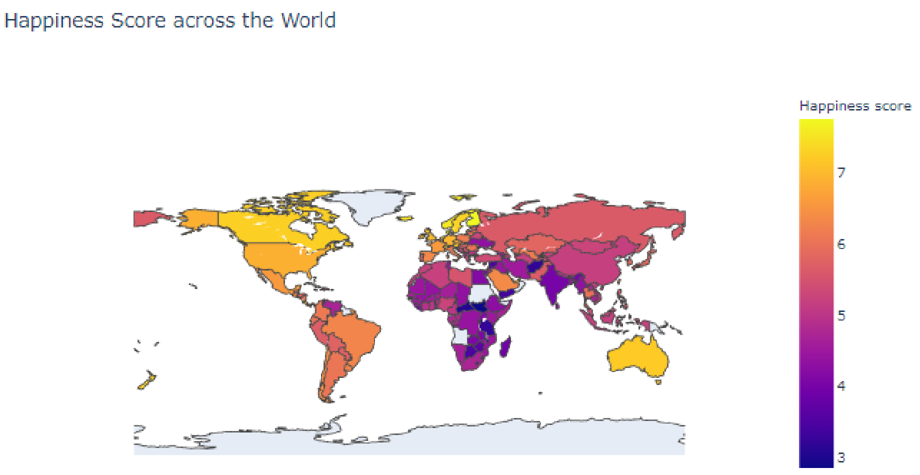




- 儘管西歐國家在幸福程度前十名佔據了不少，但北美與紐澳卻累積了更高的幸福程度
- 如預想的非洲因貧窮與飢荒問題位居最後

9.Happiness Around the World

```
#Inspired from: https://www.kaggle.com/mshinde10/predicting-world-happiness
import plotly.graph_objs as go
from plotly.offline import iplot

map_happy = dict(type = 'choropleth',
           locations = year_2019_copy['Country or region'],
           locationmode = 'country names',
           z = year_2019_copy['Score'],
           text = year_2019_copy['Country or region'],
           colorbar = {'title':'Happiness score'})

layout = dict(title = 'Happiness Score across the World',
             geo = dict(showframe = False, projection = {'type': 'equirectangular'}))

choromap3 = go.Figure(data = [map_happy], layout=layout)
iplot(choromap3)
```



- 大部分亞洲與非洲地區有較低的幸福程度
- 美國地區有相對較高的幸福程度

Regression model - Happiness Score

Declare the dependent(y) and independent variable(x)

```
# Dataframe for regression model
reg_model=year_2019_copy.drop(['Overall rank','Country or region'],axis=1)
# Outcome and Explanatory variables
x = reg_model.drop(['Score'],axis=1)
y = reg_model['Score']
```

Regression model

multiple linear regression

```
from sklearn.linear_model import LinearRegression
reg = LinearRegression()
reg.fit(x,y)
print("Regression coefficients:",reg.coef_)
print("Regression intercept:",reg.intercept_)
print("R square value:",reg.score(x,y))
```

```
Regression coefficients: [0.77537163 1.12419158 1.07814273 1.45483237 0.48978335 0.97228022]
Regression intercept: 1.7952202293072812
R square value: 0.7791638079594221
```

R平方為0.78，R平方代表從獨立變數X可以解釋依變數Y變異的比例，是一種衡量回歸模型表現的指標

Finding Adjusted R-square

因為有許多項，因此檢查調整後R平方獲得忽略一些較無關項目的結果

(參考資料: [回歸分析的R平方與調整後R平方])

```
r2 = reg.score(x,y)
n = x.shape[0]
p = x.shape[1]

adjusted_r2 = 1-(1-r2)*(n-1)/(n-p-1)
print("Adjusted R square value:",adjusted_r2)
```

```
Adjusted R square value: 0.7702710753940296
```

與原R平方相差不多，表示model十分適配

```
from sklearn.feature_selection import f_regression
f_regression(x,y)
p_values = f_regression(x,y)[1]
```

```
reg_summary = pd.DataFrame(data = x.columns.values, columns=['Features'])
reg_summary ['Coefficients'] = reg.coef_
reg_summary ['p-values'] = p_values.round(3)
reg_summary
```

	Features	Coefficients	p-values
0	GDP per capita	0.775372	0.000
1	Social support	1.124192	0.000
2	Healthy life expectancy	1.078143	0.000
3	Freedom to make life choices	1.454832	0.000
4	Generosity	0.489783	0.347
5	Perceptions of corruption	0.972280	0.000

Freedom and Social support影響較大，Generosity則較低

R square- Train test Split score

```
from sklearn.linear_model import LinearRegression
from sklearn.model_selection import train_test_split
from sklearn.metrics import r2_score
X_train, X_test, y_train, y_test = train_test_split(x, y, test_size = 0.2, random_state = 0)
regressor = LinearRegression()
regressor.fit(X_train, y_train)
y_pred = regressor.predict(X_test)
score=r2_score(y_test,y_pred)
print("R2 score:",score)
```

```
R2 score: 0.6498014441077297
```

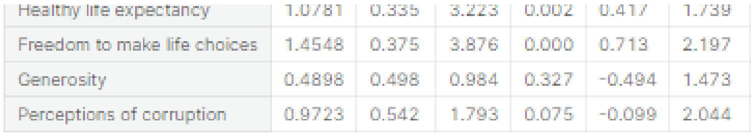
Regression using OLS

```
import statsmodels.api as sm
x1 = sm.add_constant(x)
results = sm.OLS(y,x1).fit()
results.summary()
```

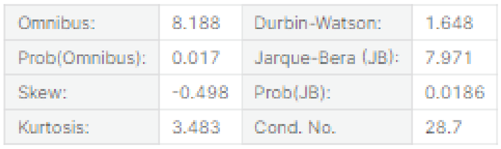
OLS Regression Results

Dep. Variable:	Score	R-squared:	0.779
Model:	OLS	Adj. R-squared:	0.770
Method:	Least Squares	F-statistic:	87.62
Date:	Fri, 29 May 2020	Prob (F-statistic):	2.40e-46
Time:	11:46:40	Log-Likelihood:	-119.76
No. Observations:	156	AIC:	253.5
Df Residuals:	149	BIC:	274.9
Df Model:	6		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
const	1.7952	0.211	8.505	0.000	1.378	2.212
GDP per capita	0.7754	0.218	3.553	0.001	0.344	1.207
Social support	1.1242	0.237	4.745	0.000	0.656	1.592



Healthy life expectancy	1.0781	0.335	3.223	0.002	0.417	1.739
Freedom to make life choices	1.4548	0.375	3.876	0.000	0.713	2.197
Generosity	0.4898	0.498	0.984	0.327	-0.494	1.473
Perceptions of corruption	0.9723	0.542	1.793	0.075	-0.099	2.044



Omnibus:	8.188	Durbin-Watson:	1.648
Prob(Omnibus):	0.017	Jarque-Bera (JB):	7.971
Skew:	-0.498	Prob(JB):	0.0186
Kurtosis:	3.483	Cond. No.	28.7

- F- Statistic: 87% and probability of nearly 0 as shown that the whole model explains our data well and it is significant
- R Square and Adjusted R square: Our adjusted R squared value is 0.77 which doesn't vary much from the R square value(0.77) which shows that the model is fitted well despite additional features
- T statistic: From the t test values the Generosity value(0.984) is very low and it has very low power and hinders the model performance and the p value(0.327) is higher than 0.05 so that feature is significant.

Notebook 2

Demographic Trend analysis on Suicide rate

發想:

由"Suicide in happy places"此一理論發想，其內容是在說當一個不快樂的人身處在其他人都很幸福的環境中會加劇他的痛苦，進而造成自殺的可能。

目標:

- 探討幸福程度與自殺率間的關係

結合World Happiness Report, Suicide Rates Overview 1985 to 2016兩個資料集

write in R

Importing Libraries

```
library(dplyr)
library(ggplot2)
library(gganimate)
library(gapminder)
library(countrycode)
library(ggplot2)
library(gganimate)
library(grid)
library(ggrepel)
library(scales)
library(ggcorrplot)
```

Data Cleansing and Transformation

Data Cleansing

- Generation Feature has been removed as it is merely serves the purpose of age feature and it has many conflicts within itself.
- Countries which have less propotion of data are removed
- Human Development index is removed from suicide data set as there are many missing values.
- Age feature format was corrected throughout the data

Data Transformation

- Continent was added to the dataset using the ‘countrycode’ package
- For the year 2015 Happiness and suicide data are merged and even HDI is included from happiness dataset.
- As we are mainly dealing with the suicide rates we have eliminated the additional countries which are tagged from World happiness data.
- categorical variables are turned to factors.

```
#Data Cleaning
#data <- read.csv('../input/suicide-rates-overview-1985-to-2016/master.csv')
data_s<- read.csv('../input/suicide-rates-overview-1985-to-2016/master.csv')
data_h<- read.csv('../input/world-happiness/2015.csv')
data_s<-data_s[-c(8,9)] # Drropping  columns which has missing values almost everywhere.

data_s<-data_s %>%
  rename(
    Country = country,
    Year = year,
    Sex= sex,
    Age = age,
    Pop = population,
    S_N = suicides_no,
    GDP = gdp_for_year....,
    GDP_C = gdp_per_capita....,
    Gen = generation
    )

#Reordering age as aa Ordinal variable:
data_s$Age <- factor(data_s$Age,
                     levels = c("5-14 years",
                                "15-24 years",
                                "25-34 years",
                                "35-54 years",
                                "55-74 years",
                                "75+ years"))

#We can see from data that 2016 column is mostly NA so filtering the same
data_s <- data_s %>%
          filter(Year != 2016)

# 3) TIDYING DATAFRAME
data_s$Age <- gsub(" years", "", data_s$Age)
data_s$Sex <- ifelse(data_s$Sex == "male", "Male", "Female")

# Adding continent to data:
data_s$Continent <- countrycode(sourcevar = data_s[, "Country"],
                              origin = "country.name",
                              destination = "continent")
```

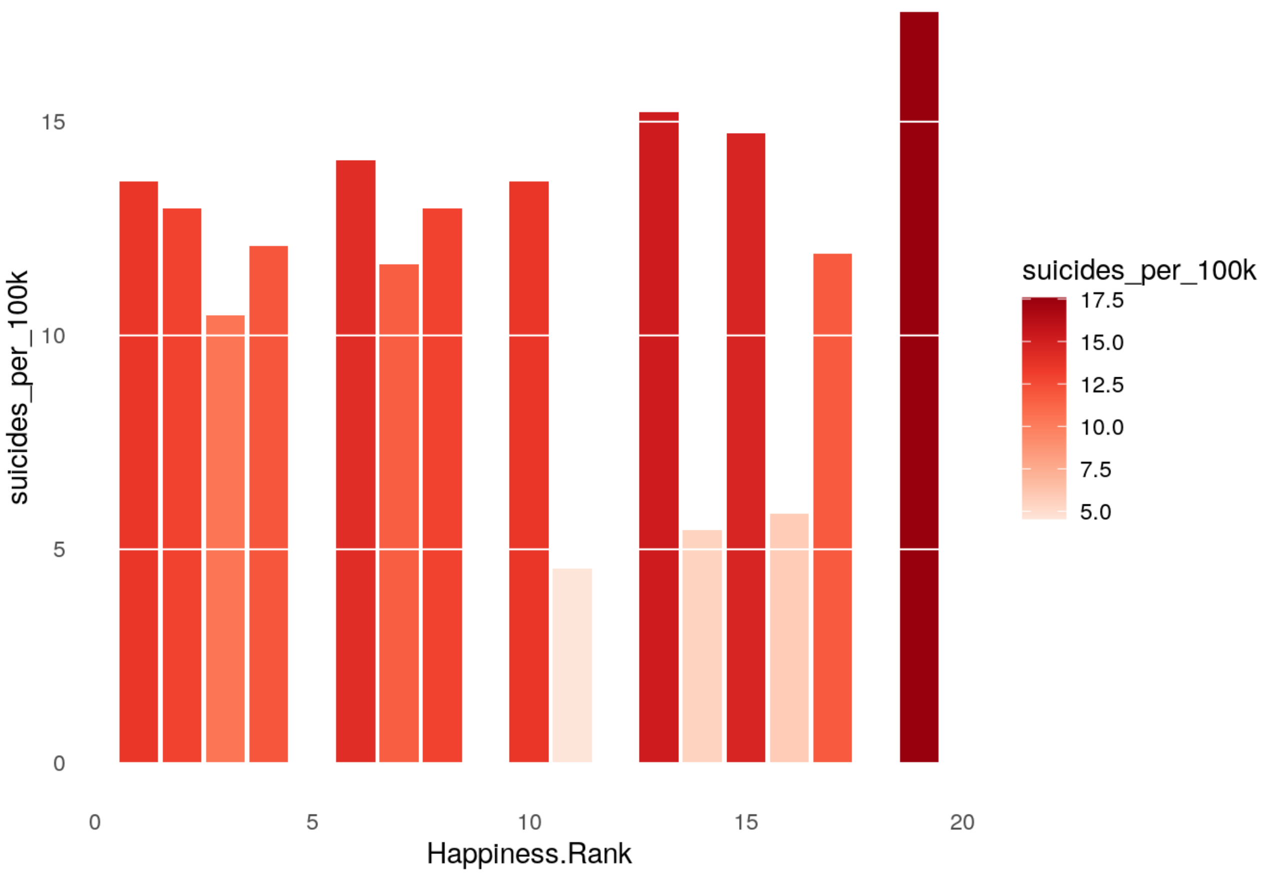
Trend Analysis

此部分為自殺率方面的分析，與World Happiness Report較無關，因此不多加贅述

Relation between Suicide and Happiness

Happiness vs Suicide Rate Over the years

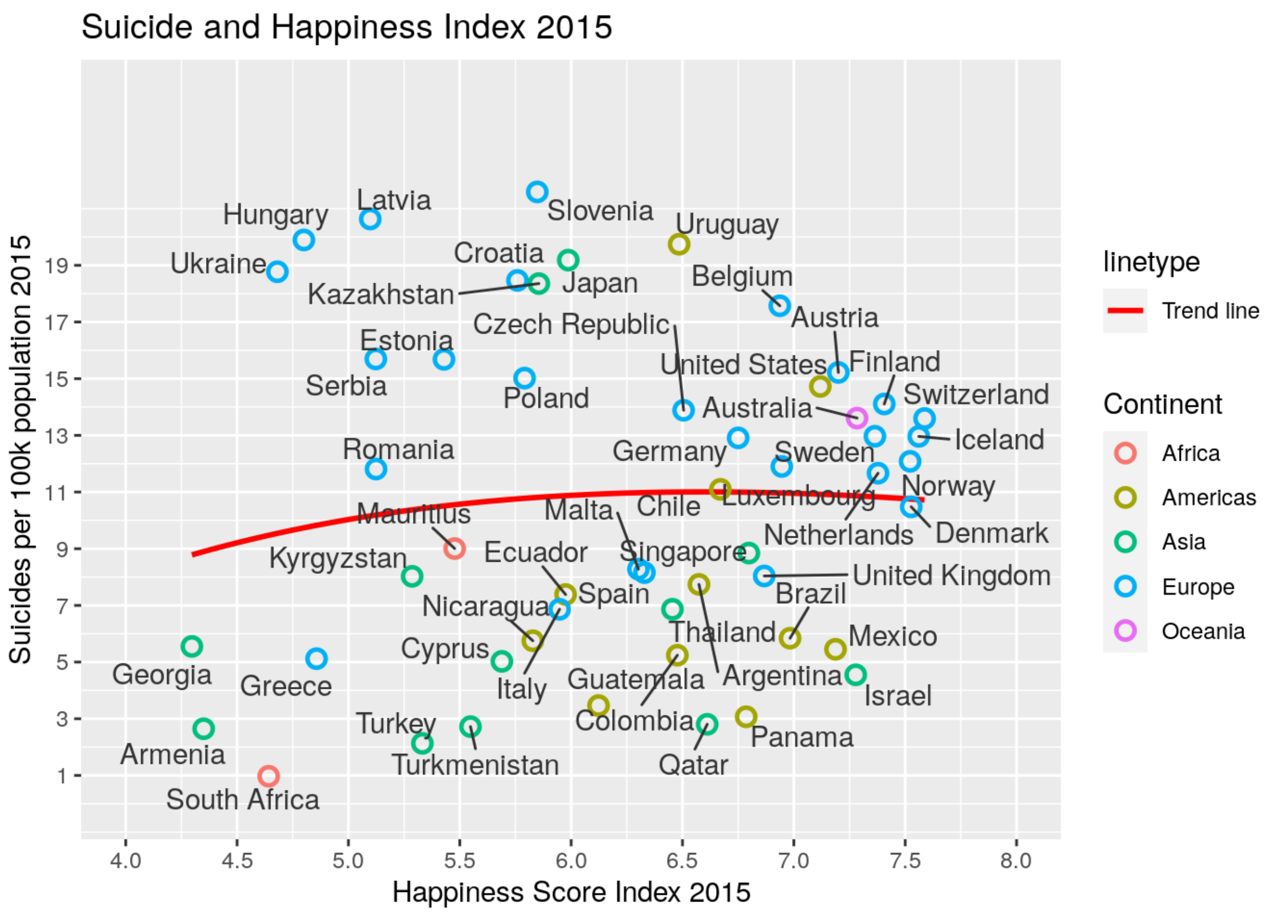
```r
combine4<-combine3[ which(combine3$Happiness.Rank < 21), ]

p <- ggplot(combine4, aes(Happiness.Rank, suicides_per_100k, fill = suicides_per_100k)) +
  geom_col() +
  scale_fill_distiller(palette = "Reds", direction = 1) +
  theme_minimal() +
  theme(
    panel.grid = element_blank(),
    panel.grid.major.y = element_line(color = "white"),
    panel.ontop = TRUE
  )
p #+ #transition_states(Happiness.Rank, wrap = FALSE) +
```



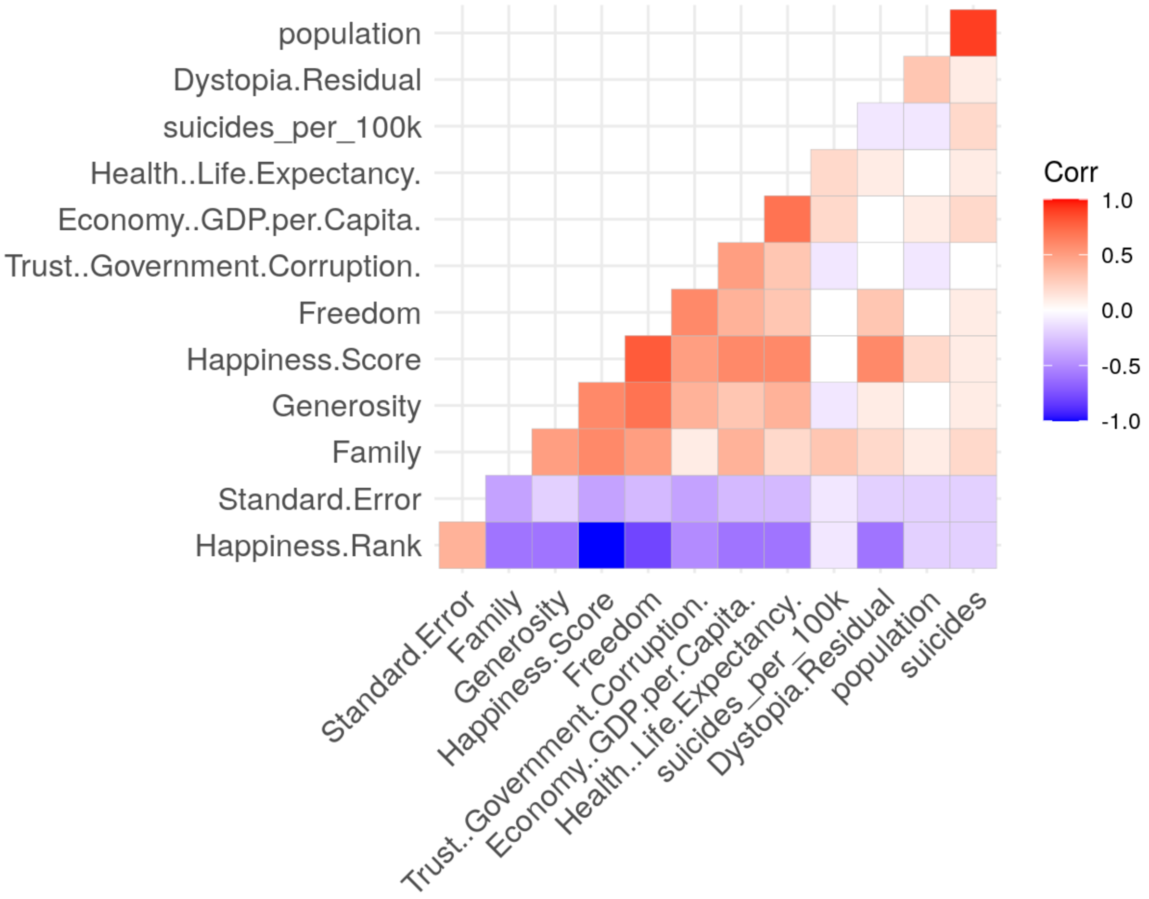
幸福程度排名最後的自殺率仍最高

Happiness vs Suicide Rate with the countries

```
pc1 <- ggplot(combine3, aes(x=Happiness.Score, y =suicides_per_100k, color = Continent ))

pc2 <- pc1 +
  geom_smooth(mapping = aes(linetype = "Trend line"),
            method = "lm",
            formula = y ~ x + log(x), se = FALSE,
            color = "red")
pc3 <- pc2 + geom_point(shape = 1, size = 2.5, stroke = 1.25)
pc4 <- pc3 +
   geom_text_repel(aes(label = Country),
                  color = "gray20",
                  force = 10)

pc5 <- pc4 +
  scale_x_continuous(name = "Happiness Score Index 2015",
                     limits = c(4, 8),
                     breaks = seq(4, 8,by = 0.5)) +
  scale_y_continuous(name = "Suicides per 100k population 2015",
                     limits = c(0, 25),
                     breaks = seq(1, 20,by = 2)) +
  ggtitle("Suicide and Happiness Index 2015")
pc5
```



```
combine3<-combine3[-c(1,5,16)]
corr <- round(cor(combine3), 1)


ggcorrplot(corr, hc.order = TRUE, type = "lower")
```



以上取了21個國家的幸福程度排名以及將其自殺率與平均自殺率比較

可以發現幸福程度越高的其自殺率有稍微降低的趨勢，但仍舊不足以證明該理論

而最後一張關係圖可發現Population、Economy等項目與自殺率相關

大部分亞洲與美洲國家位於趨勢線上，即使這些國家的幸福程度排名十分前面，他們的自殺率較高

現有分析方法產出與比較

Notebook1 除了將資料視覺化，明確指出可影響幸福程度的項目，能夠進一步探討政策等著重方面規劃，此外，採用了機器學習算法對數據進行預測，並比較了各種模型的效果，以便更好地了解影響幸福指數的因素。

Notebook2 將理論做驗證，雖然由幸福程度與自殺率的關係圖乍看可以認同理論，但仍需進一步觀察各項間的關係，而透過觀察又可以再發現影響自殺率與幸福程度的共同因素，利用這兩個資料集可以更深入地探討自殺率和社會因素、心理因素之間的關係，甚至使用更多的機器學習模型來預測自殺率的趨勢和影響因素。

兩者皆都關注了人類幸福和健康等問題，但是它們的方法和焦點有所不同。Notebook1將資料整理後完整分析其自身項目中的關係，因此相較於Notebook2，在World Happiness Report的資料處理上注意了資料缺失、離群值等問題，較全面與嚴謹，而Notebook2則是關注在全球自殺率與幸福度等其他因素之間的關係，直接取其所需部分。

Notebook1 是以python撰寫，Notebook2 是以R語言撰寫

Insight

這份資料集含有大量的資訊，可以依據對分析對象聚焦程度不同，而有多樣的使用方式與產出，對於第一次接觸資料分析的我來說，雖然資訊量龐大，卻有很大的幫助，尤其是Notebook1在將資料視覺化時，不只是學到了這些視覺化的方法，同時可以去思考資料間的關係應該用哪種圖去表達最清晰。

此外，在瀏覽各個現有方法時，看到將國家劃分至不同區域，分析出各區域的各個因素影響其幸福程度的關係圖，得知各地區影響最大的因素，我認為Notebok1上也可以運用其分類，再比較各地區幸福程度，可推測出產生人民幸福感的條件順序(類似馬斯洛需求層次理論的概念)，得知增進幸福程度的最佳步驟。

除了分析資料集本身外，也有許多利用此資料集與其他資料集整合的方式，而最令我印象深刻的莫過於Notebook2中將曾經聽過的理論作驗證而做資料分析，讓我發現資料分析原來不是只有從資料中觀察整合獲得結果結論，也可以試著從現有的結論發散，尋找資料間的關聯性，再試著探索更多新的發現。