World Happiness ML - Model

ML - Model made by Harsh Vashisth on World Happiness analysis of 2019. This model comprises the World Happiness score of 156 countries along with a regression model of an average score of above 70 out of 100 by providing test data again and again with varying weights. The data used to train this model is based on the statistics of 2019.

Introduction -

The **World Happiness Report** is a publication of the *United Nations Sustainable Development Solutions Network*. It contains articles and rankings of national happiness, based on respondent ratings of their own lives, which the report also correlates with various (quality of) life factors. As of March 2021, Finland had been ranked the happiest country in the world four times in a row.

Data is collected from people in over 150 countries. Each variable measured reveals a populated-weighted average score on a scale running from 0 to 10 that is tracked over time and compared against other countries.

These variables currently include -

- real GDP per capita
- social support
- healthy life expectancy
- freedom to make life choices
- generosity
- perceptions of corruption

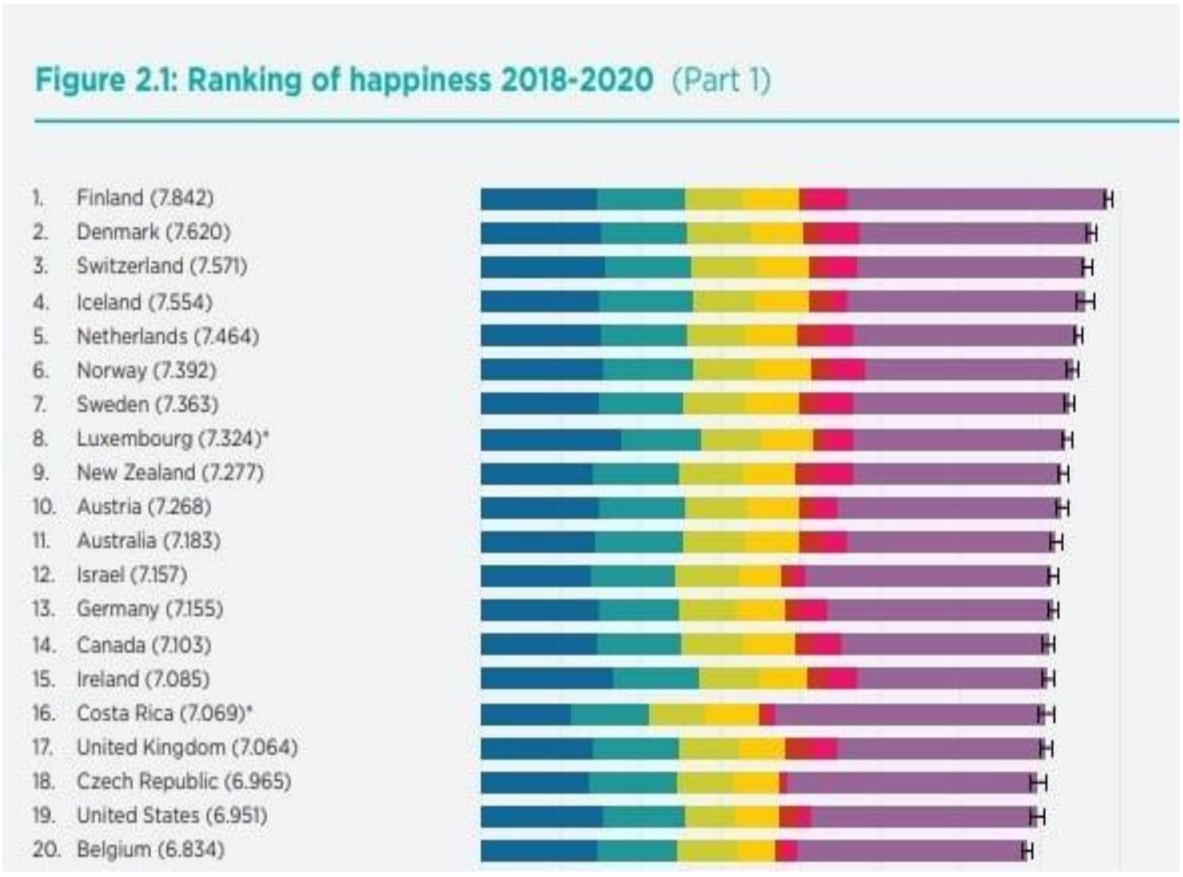
Each country is also compared against a hypothetical nation called Dystopia. Dystopia represents the lowest national averages for each key variable and is, along with residual error, used as a regression benchmark. The six metrics are used to explain the estimated extent to which each of these factors contribute to increasing life satisfaction when compared to the hypothetical nation of Dystopia, but they themselves do not have an impact on the total score reported for each country.

The 2019 report features the happiness score averaged over the years 2016–2018. As per the 2019 Happiness Index, Finland is the happiest country in the world. Denmark, Norway, Iceland and Netherlands hold the next top positions. The report was published on 20 March 2019 by the UN.

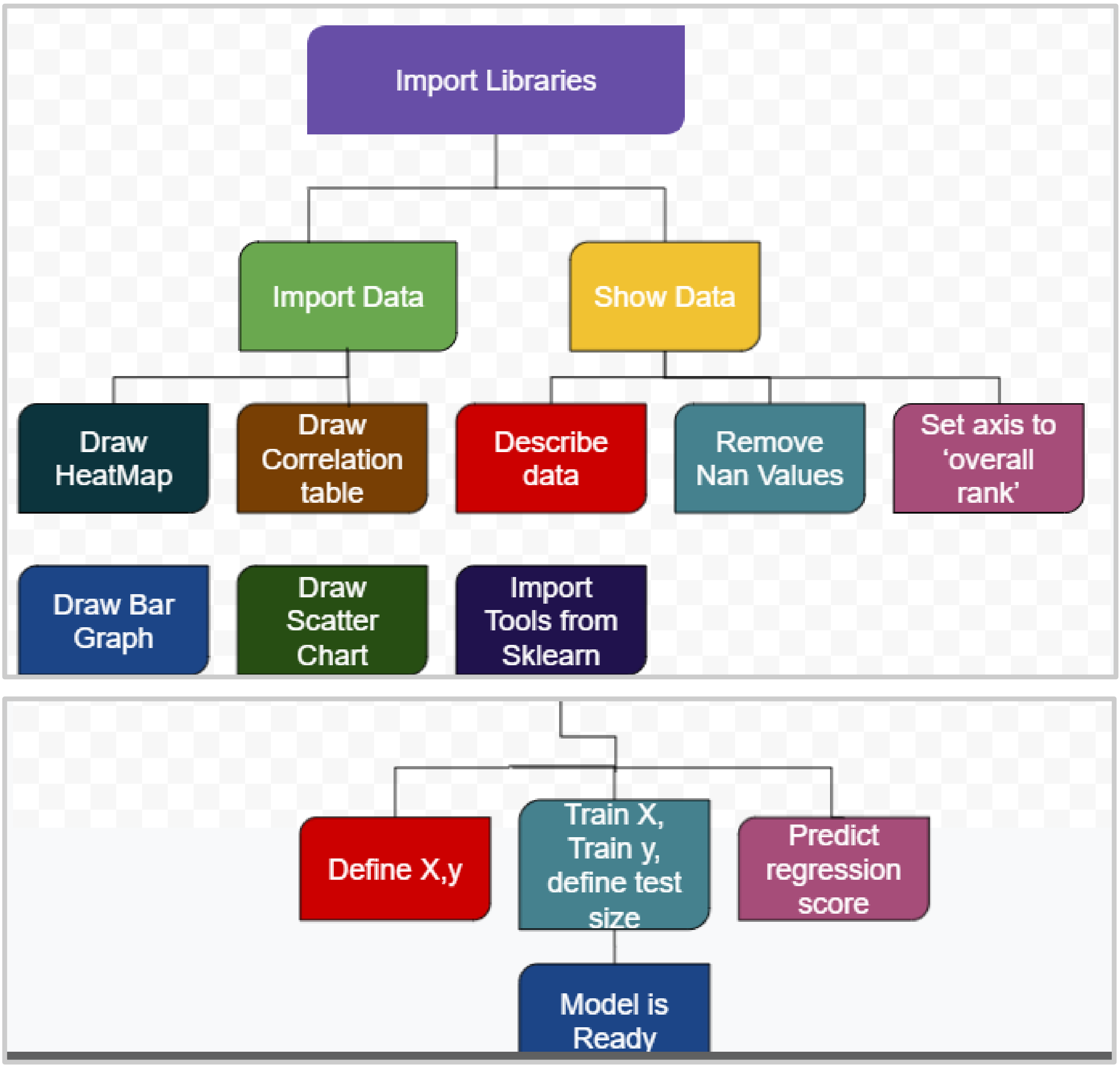
Introduction to the problem -

We are given the World happiness data of the year 2019 which was collected based on the data collected in the year 2016 - 2018. By using the following data, I have to build a training model using Sklearn, NumPy, Pandas, random forest. By using various methods, we will define the relation between the happiness score and the factors it depends on. Factors like GDP per capita, social support, Healthy life expectancy, Generosity and a few more.

Using *correlation table and heat map*, the relation between all the factors is defined. The rankings of national happiness are based on a Cantril ladder survey. Nationally representative samples of respondents are asked to think of a ladder, with the best possible life for them being a 10, and the worst possible life being a 0. They are then asked to rate their own current lives on that 0 to 10 scale. The report correlates the life evaluation results with various life factors. In the reports, experts in fields including economics, psychology, survey analysis, and national statistics, describe how measurements of well-being can be used effectively to assess the progress of nations, and other topics. Each report is organized by chapters that delve deeper into issues relating to happiness, including mental illness, the objective benefits of happiness, the importance of ethics, policy implications, and links with the Organisation for Economic Co-operation and Development's (OECD) approach to measuring subjective well-being and other international and national efforts.



Proposed Workflow -

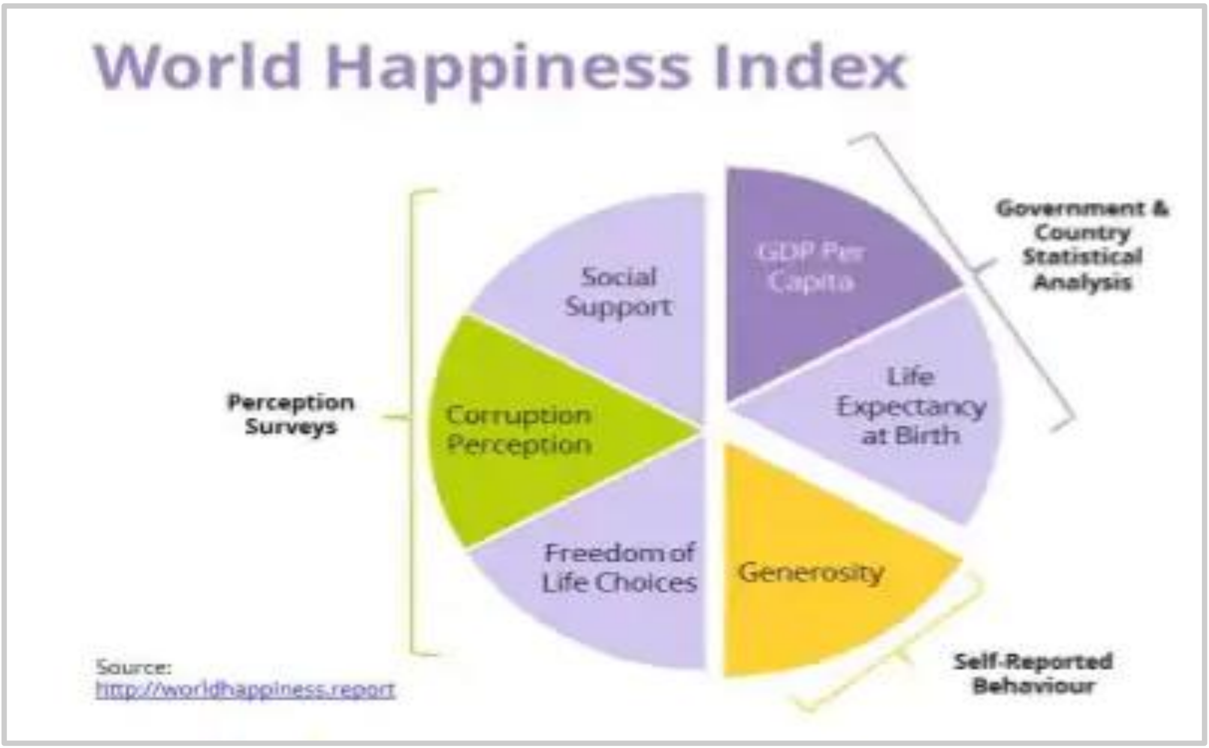


<u>Introduction of dataset used -</u>

The given dataset belongs to the world happiness report of 2019, which comprises data collected from the year 2016 to the year 2018. This given dataset comprises of 8 columns which are -

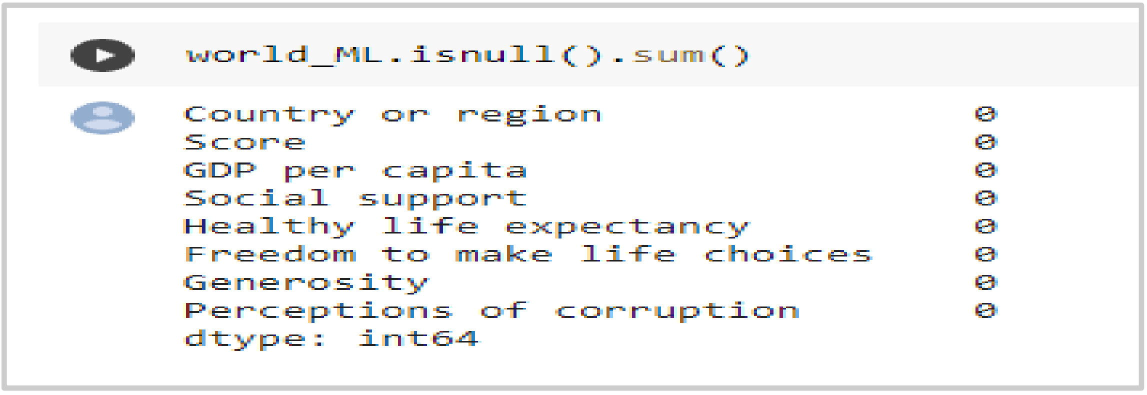
- Overall rank
- Happiness score
- GDP per capita
- Social support
- Healthy life expectancy
- Freedom to make life choices
- Generosity
- Perceptions of corruption

Among these factors except the first two which are overall rank and happiness score, the rest all are responsible for calculating the happiness score of any nation.



Data Preprocessing -

Data preprocessing means analysing the shape of the dataset, values of columns, rows, filtering the dataset for the nan or none values present in the dataset by either deleting the rows or columns with the nan values or by replacing the values with a predictable output based on the data. Now, for doing that we need to first identify the number of missing values in the dataset.

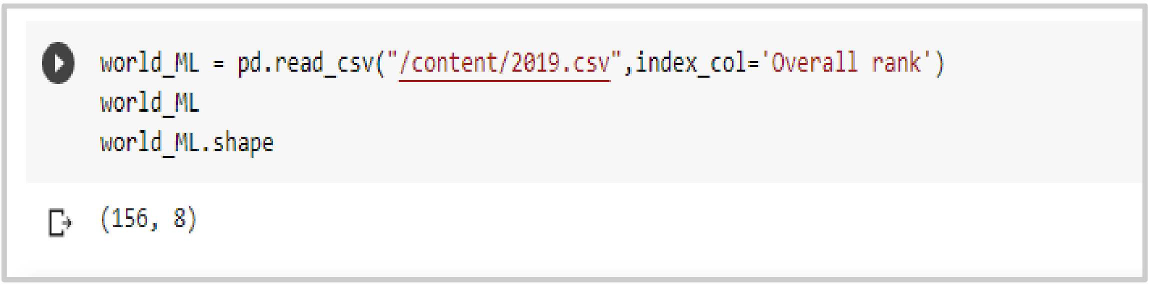


```
world_ML.isnull().sum()
```

```
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
```

->> Fortunately, this data set doesn't have any missing values so this step is avoided.

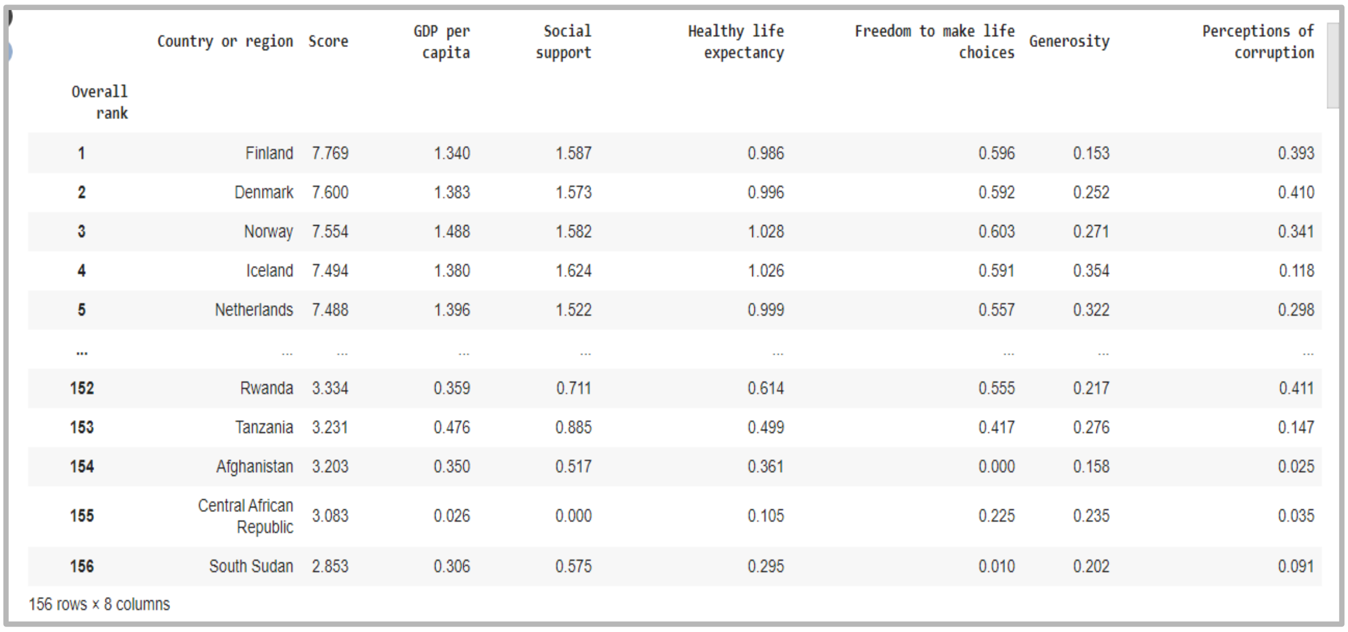
->> Since we are also analysing the dataset, we can comment on the shape of the same too. In this case, the dataset is **(156,8).**



```
world_ML = pd.read_csv("/content/2019.csv",index_col='Overall rank')
world_ML
world_ML.shape
```

```
(156, 8)
```

->> The columns represent various information such as Happiness score, Overall rank, GDP per capita and many more too.



Overall rank	Country or region	Score	GDP per capita	Social support	Healthy life expectancy	Freedom to make life choices	Generosity	Perceptions of 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
...	...	...	...	...	...	...	...	...
152	Rwanda	3.334	0.359	0.711	0.614	0.555	0.217	0.411
153	Tanzania	3.231	0.476	0.885	0.499	0.417	0.276	0.147
154	Afghanistan	3.203	0.350	0.517	0.361	0.000	0.158	0.025
155	Central African Republic	3.083	0.026	0.000	0.105	0.225	0.235	0.035
156	South Sudan	2.853	0.306	0.575	0.295	0.010	0.202	0.091

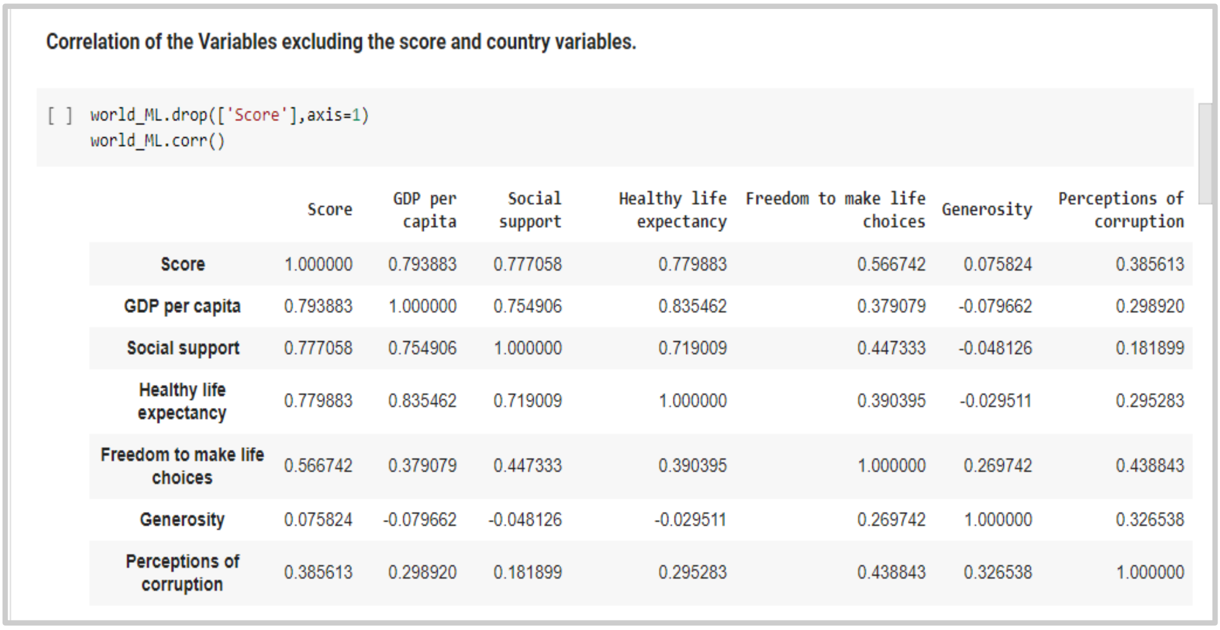
156 rows × 8 columns

Results -

This section includes various steps -

1. Correlation table
2. Heat map
3. Bar graph
4. Scatter plot
5. Training the model
6. Printing the regression score

1.) <u>Correlation -</u> This table represents all the variables present in a matrix format so that they are linked with each other. One in this table represents a completely linear relation between two variables which can be only possible if both the variables are same only. Any number less than and close to one between two variables represents how linearly they both are connected. In this case, GDP per capita has a score of 0.79388 which is closest to one.



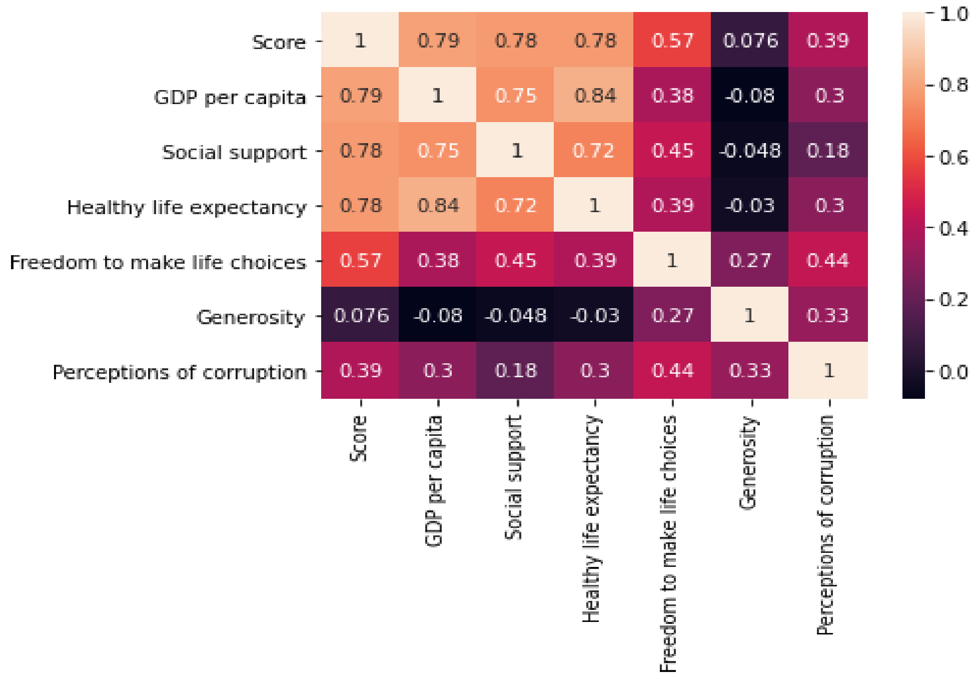
Correlation of the Variables excluding the score and country variables.

```
[ ] world_ML.drop(['Score'],axis=1)
    world_ML.corr()
```

	Score	GDP per capita	Social support	Healthy life expectancy	Freedom to make life choices	Generosity	Perceptions of corruption
Score	1.000000	0.793883	0.777058	0.779883	0.566742	0.075824	0.385613
GDP per capita	0.793883	1.000000	0.754906	0.835462	0.379079	-0.079662	0.298920
Social support	0.777058	0.754906	1.000000	0.719009	0.447333	-0.048126	0.181899
Healthy life expectancy	0.779883	0.835462	0.719009	1.000000	0.390395	-0.029511	0.295283
Freedom to make life choices	0.566742	0.379079	0.447333	0.390395	1.000000	0.269742	0.438843
Generosity	0.075824	-0.079662	-0.048126	-0.029511	0.269742	1.000000	0.326538
Perceptions of corruption	0.385613	0.298920	0.181899	0.295283	0.438843	0.326538	1.000000

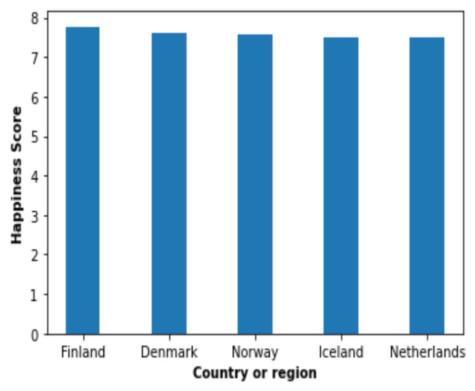
2.) <u>HeatMap -</u> The above correlation was plotted to draw the following heat map which shows the similar results as a correlation matrix but with colors and numbers for better understanding instead of only numbers in case of correlation.

```
[4]  heat_map = sns.heatmap(world_ML.corr(),annot=True)
```

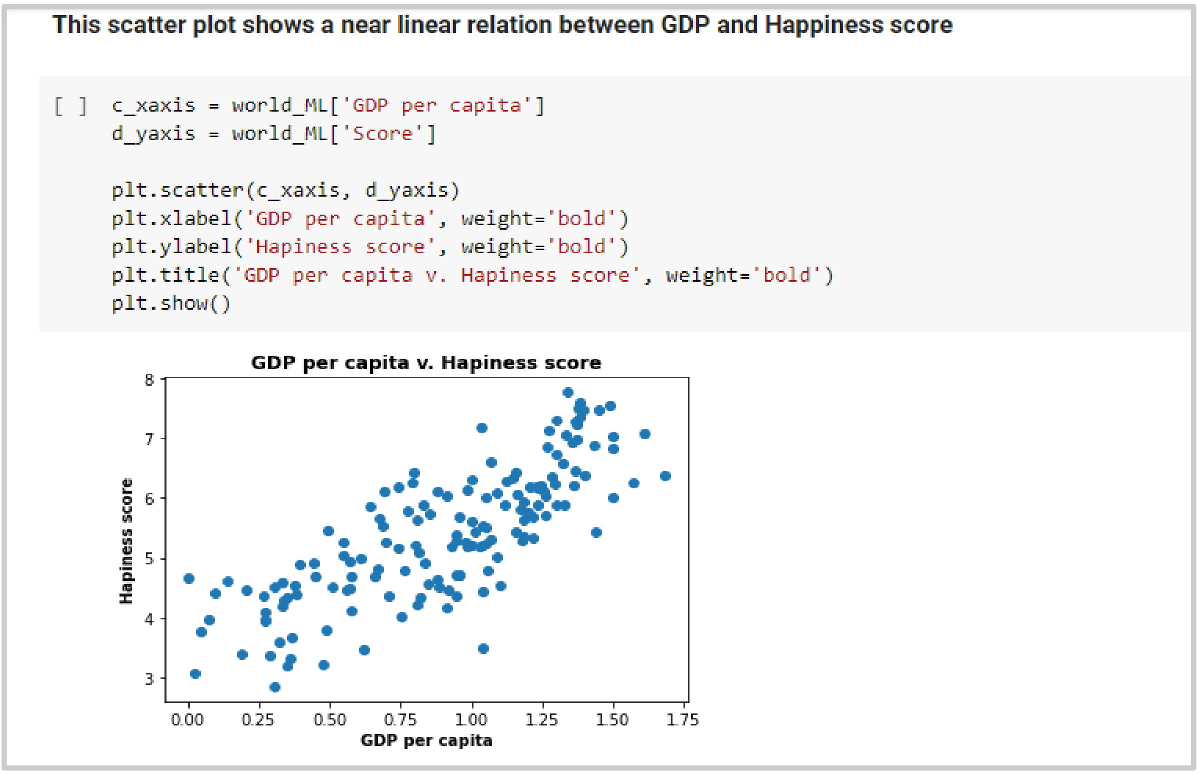


3.) **<u>Bar graph -</u>** Bar graph will represent the top five countries' happiness scores.

```
[ ]  a_xaxis = world_ML['Country or region'].iloc[:5]
     b_yaxis = world_ML['Score'].iloc[:5]
     #plt.ylim(0,8)

     ticks = [0,1,2,3,4,5,6,7,8]

     plt.xlabel('Country or region', weight='bold')
     plt.ylabel('Happiness Score', weight='bold' )

     plt.yticks(ticks)
     plt.bar(a_xaxis,b_yaxis,width=0.4)
     plt.show()
```



4.) **<u>Scatter plot -</u>** Scatter plot will help in better understanding how GDP per capita and happiness score are closely related with each other.

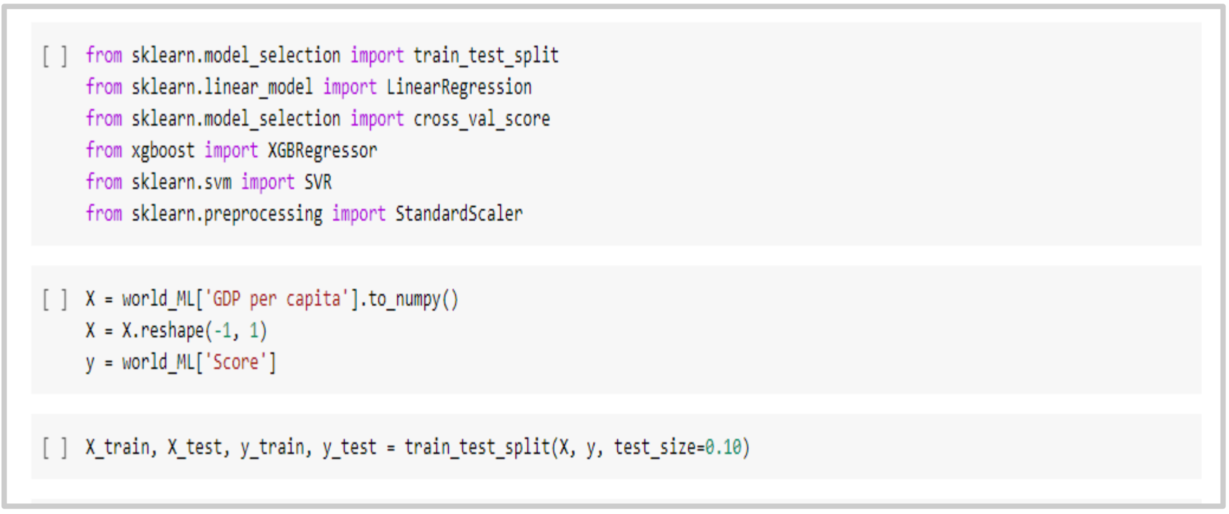


This scatter plot shows a near linear relation between GDP and Happiness score

```
[ ]  c_xaxis = world_ML['GDP per capita']
     d_yaxis = world_ML['Score']

     plt.scatter(c_xaxis, d_yaxis)
     plt.xlabel('GDP per capita', weight='bold')
     plt.ylabel('Hapiness score', weight='bold')
     plt.title('GDP per capita v. Hapiness score', weight='bold')
     plt.show()
```

5.) **Train the model -** To perform this step we have to import necessary functions from the Sklearn library like train_test_split, LinearRegression.

This step includes functions like train, split, and providing the model with the test data which is 10% of the total data in this particular ML model.

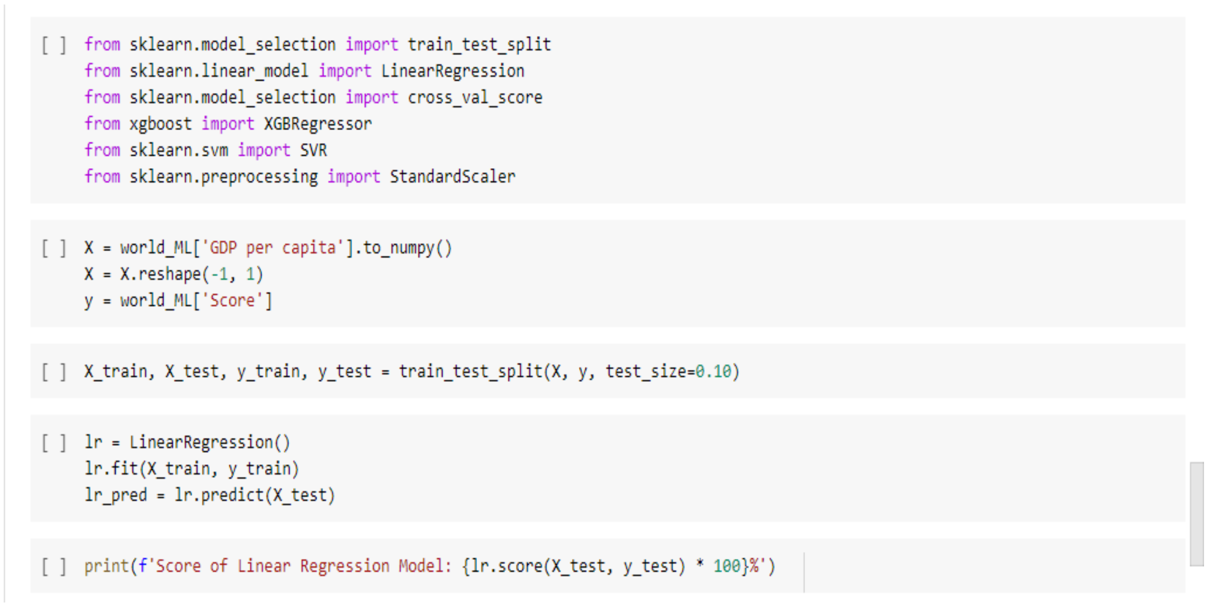


```
[ ]  from sklearn.model_selection import train_test_split
     from sklearn.linear_model import LinearRegression
     from sklearn.model_selection import cross_val_score
     from xgboost import XGBRegressor
     from sklearn.svm import SVR
     from sklearn.preprocessing import StandardScaler
```

```
[ ]  X = world_ML['GDP per capita'].to_numpy()
     X = X.reshape(-1, 1)
     y = world_ML['Score']
```

```
[ ]  X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.10)
```

6.) **Printing the Regression Score -** This is the final step of the ML model where the regression score is printed after the model is said to predict the data.



```
[ ] from sklearn.model_selection import train_test_split
    from sklearn.linear_model import LinearRegression
    from sklearn.model_selection import cross_val_score
    from xgboost import XGBRegressor
    from sklearn.svm import SVR
    from sklearn.preprocessing import StandardScaler
```

```
[ ] X = world_ML['GDP per capita'].to_numpy()
    X = X.reshape(-1, 1)
    y = world_ML['Score']
```

```
[ ] X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.10)
```

```
[ ] lr = LinearRegression()
    lr.fit(X_train, y_train)
    lr_pred = lr.predict(X_test)
```

```
[ ] print(f'Score of Linear Regression Model: {lr.score(X_test, y_test) * 100}%')
```

In this particular model, the score was 56%.