Task

- To build a Linear Regression model that shows the relationship between `Gross Domestic Product per capita` and `Life Satisfaction` of various countries
- The Better Life Index dataset is downloaded from OECD's Website, and the stats about GDP per capita from IMF's website
- The `Life satisfaction value` can be obtained from BLI dataset and the `GDP per capita` can be obtained from GDP dataset

Importing dependencies

```
!pip install -U -q PyDrive
from pydrive.auth import GoogleAuth
from pydrive.drive import GoogleDrive
from google.colab import auth
from oauth2client.client import GoogleCredentials
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import sklearn
from sklearn.linear_model import LinearRegression
from scipy import stats
import seaborn as sns
from pylab import rcParams

sns.set()
pd.options.display.max_rows = 50
```

Downloading the datasets from Google Drive

The datasets are already saved in Google Drive, we will be downloading the datasets from google drive

```
auth.authenticate_user()
gauth = GoogleAuth()
gauth.credentials = GoogleCredentials.get_application_default()
drive = GoogleDrive(gauth)
```

```
# https://drive.google.com/open?id=1Qffns5fjSa1pPqlRlQQTtP5D8DTgeQOq
file_id = '1Qffns5fjSa1pPqlRlQQTtP5D8DTgeQOq'
downloaded = drive.CreateFile({'id': file_id})
```

```
downloaded.GetContentFile("BLI.csv")
```

```
# https://drive.google.com/open?id=1atvVP-maH-nHXlV8ib6fvCBRJfWAOoP0
file_id = '1atvVP-maH-nHXlV8ib6fvCBRJfWAOoP0'
downloaded = drive.CreateFile({'id': file_id})
```

```
downloaded.GetContentFile("GDP.csv")
```

Loading the datasets

Loading the datasets into the program as pandas dataframe

```
bli = pd.read_csv("BLI.csv", thousands=",")
gdp = pd.read_csv("GDP.csv", thousands=",", delimiter="\t", encoding='Latin1', na_values="
```

Data Cleaning and Preprocessing

Better Life Index dataset

We will be considering the attributes = {`Country`, `Indicator`, `Value`}, from the BLI dataset

```
bli = bli.loc[:,["Country", "Indicator", "Value"]]
```

The set of values of `Indicator` attribute are as follows

```
set(bli["Indicator"])
```

```
{'Air pollution',
 'Dwellings without basic facilities',
 'Educational attainment',
 'Employees working very long hours',
 'Employment rate',
 'Feeling safe walking alone at night',
 'Homicide rate',
 'Household net adjusted disposable income',
 'Household net financial wealth',
 'Housing expenditure',
 'Labour market insecurity',
 'Life expectancy',
 'Life satisfaction',
 'Long-term unemployment rate',
 'Personal earnings',
 'Quality of support network',
 'Rooms per person',
 'Self-reported health',
 'Stakeholder engagement for developing regulations',
 'Student skills',
 'Time devoted to leisure and personal care',
 'Voter turnout',
 'Water quality',
 'Years in education'}
```

Among all those indicators, we will be considering the `Life satisfaction` indicator only Then we can drop the `Indicator` column

```
bli_df = bli[bli["Indicator"] == 'Life satisfaction'].reset_index(drop=True).drop("Indicat
bli_df.head(20)
```

	Country	Value
0	Australia	7.3
1	Austria	7.1
2	Belgium	6.9
3	Canada	7.4
4	Czech Republic	6.6
5	Denmark	7.5
6	Finland	7.4
7	France	6.4
8	Germany	7.0
9	Greece	5.6
10	Hungary	5.3
11	Iceland	7.5
12	Ireland	6.8
13	Italy	5.8
14	Japan	5.9
15	Korea	5.8
16	Luxembourg	6.7
17	Mexico	6.2
18	Netherlands	7.3

```
print("Number of null values in Better Life Index data:")
bli_df.isna().sum()
```

```
Number of null values in Better Life Index data:
Country    0
Value      0
dtype: int64
```

```
print("Number of duplicate entries in the Better Life Index dataset:", bli_df.duplicated()
bli_df[bli_df.duplicated()]
```

```
Number of duplicate entries in the Better Life Index dataset: 35
```

	Country	Value
37	Australia	7.3
39	Belgium	6.9
41	Czech Republic	6.6
44	France	6.4
45	Germany	7.0
46	Greece	5.6
54	Mexico	6.2
57	Norway	7.6
60	Slovak Republic	6.2
64	Turkey	5.5
65	United Kingdom	6.5
67	Brazil	6.5
70	Israel	7.1
74	Australia	7.3
75	Austria	7.1
76	Belgium	6.9
78	Czech Republic	6.6
83	Greece	5.6
84	Hungary	5.3
86	Ireland	6.8
87	Italy	5.8
94	Norway	7.6
99	Sweden	7.3
101	Turkey	5.5
102	United Kingdom	6.5
103	United States	6.9
105	Chile	6.5
107	Israel	7.1
109	Slovenia	5.7
110	OECD - Total	6.5

Removing the duplicates from BLI dataset and resetting its index values

157	Spain	6.2

```
bli_df = bli_df.drop_duplicates(subset="Country").reset_index(drop=True)
```

```
bli_df.head(10)
```

	Country	Value
0	Australia	7.3
1	Austria	7.1
2	Belgium	6.9
3	Canada	7.4
4	Czech Republic	6.6
5	Denmark	7.5
6	Finland	7.4
7	France	6.4
8	Germany	7.0
9	Greece	5.6

Gross Domestic Product dataset

From the GDP dataset, we want the `Country` and `2015` attributes only

```
gdp_df = gdp.loc[:, ["Country", "2015"]]
```

Checking for any null values in the GDP dataset

```
gdp_df.isna().sum()
```

```
Country    0
2015       3
dtype: int64
```

Dropping the rows with null values and resetting the index

```
gdp_df = gdp_df.dropna().reset_index(drop=True)
```

Checking for duplicate values in GDP dataset

```
gdp_df.duplicated().sum()
```

```
0
```

```
gdp_df.head(20)
```

	Country	2015
0	Afghanistan	599.994
1	Albania	3995.383
2	Algeria	4318.135
3	Angola	4100.315
4	Antigua and Barbuda	14414.302
5	Argentina	13588.846
6	Armenia	3534.860
7	Australia	50961.865
8	Austria	43724.031
9	Azerbaijan	5739.433
10	The Bahamas	23902.805
11	Bahrain	23509.981
12	Bangladesh	1286.868
13	Barbados	15773.555
14	Belarus	5749.119
15	Belgium	40106.632
16	Belize	4841.735
17	Benin	780.063
18	Bhutan	2843.402

Data Integration

Joining both the datasets and renaming the attribute names

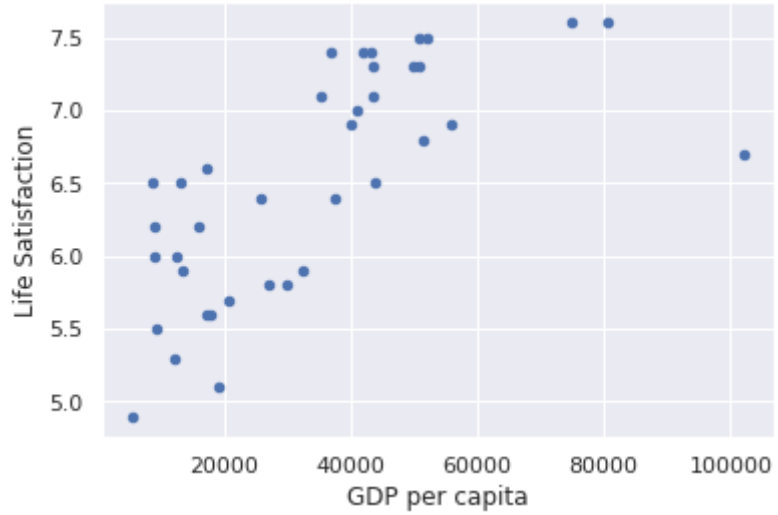
```
dataset = bli_df.join(gdp_df.set_index("Country"), on="Country", how="inner", sort=True).r
dataset.rename(columns={"Value":"Life Satisfaction", "2015":"GDP per capita"}, inplace=Tru
dataset.head(20)
```

	Country	Life Satisfaction	GDP per capita
0	Australia	7.3	50961.865
1	Austria	7.1	43724.031
2	Belgium	6.9	40106.632
3	Brazil	6.5	8669.998
4	Canada	7.4	43331.961
5	Chile	6.5	13340.905
6	Czech Republic	6.6	17256.918
7	Denmark	7.5	52114.165
8	Estonia	5.6	17288.083
9	Finland	7.4	41973.988
10	France	6.4	37675.006
11	Germany	7.0	40996.511
12	Greece	5.6	18064.288
13	Hungary	5.3	12239.894

Outlier analysis

```
dataset.plot.scatter(x="GDP per capita", y="Life Satisfaction")
plt.show()
```

```
'c' argument looks like a single numeric RGB or RGBA sequence, which should be
```

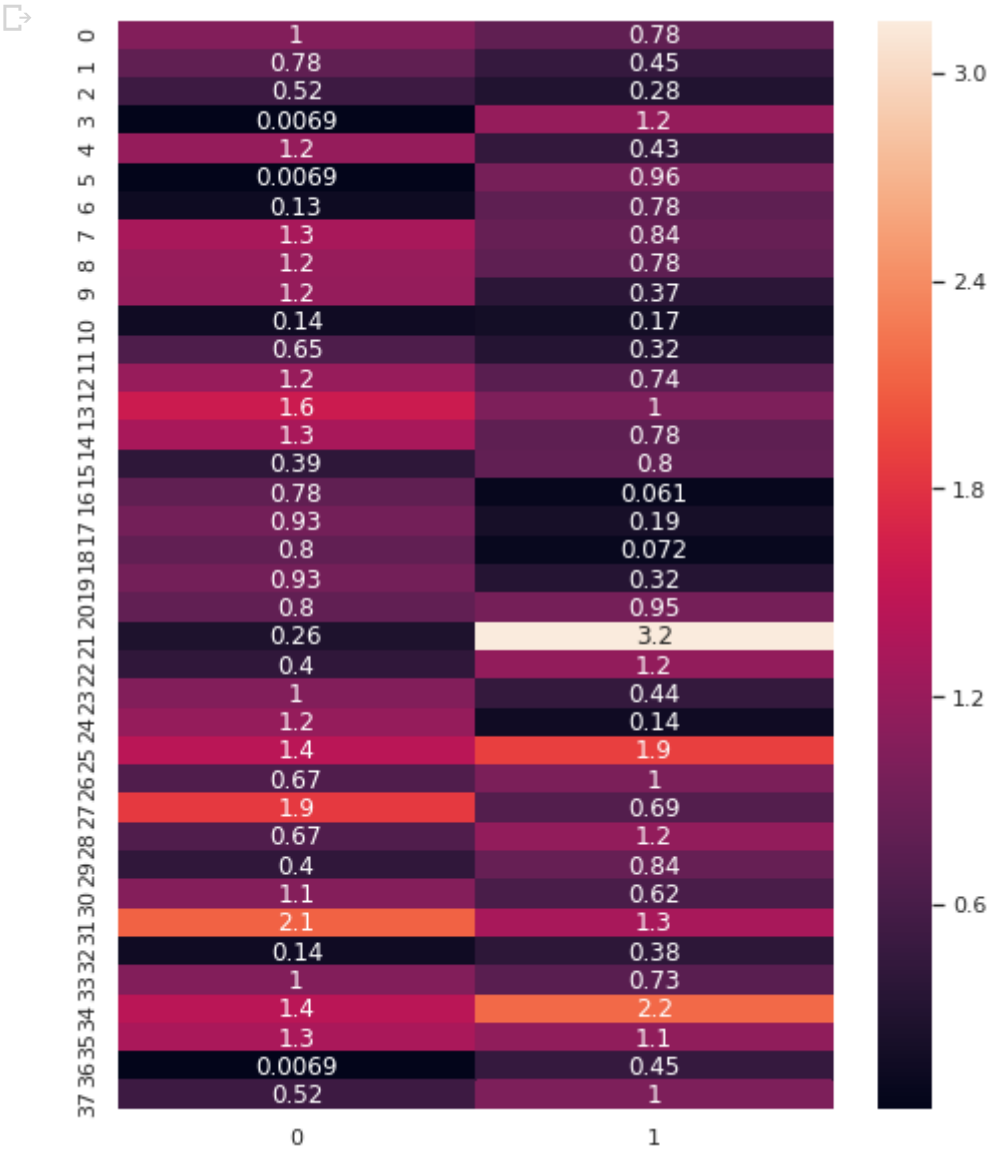


Z-Score

```
Z-score threshold = 3.0
```

Calculating and plotting the Z-score values on a heat map

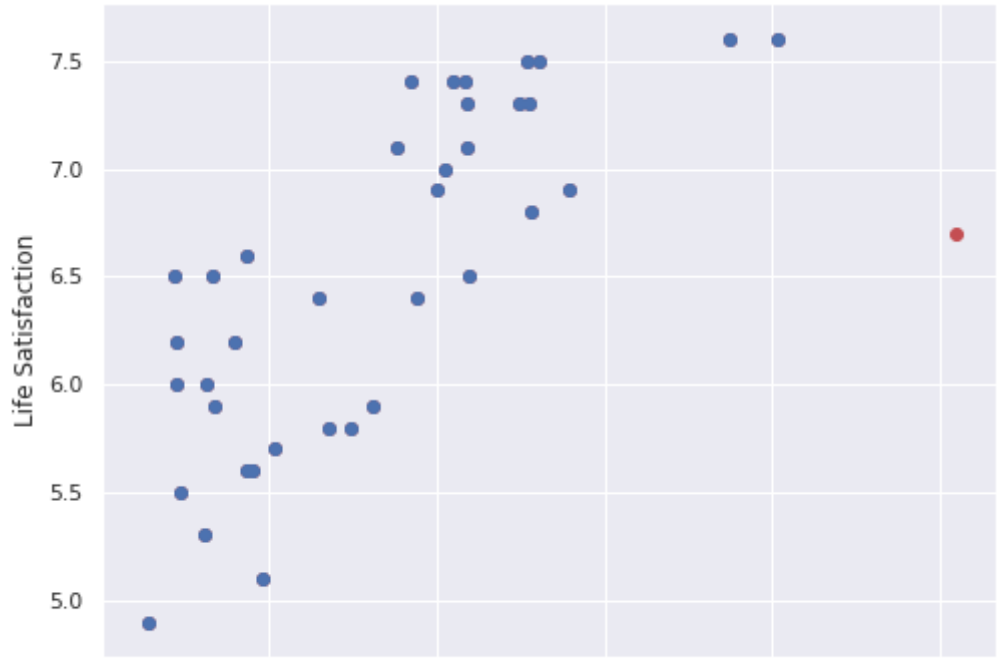
```
z = np.abs(stats.zscore(dataset.loc[:, ["Life Satisfaction", "GDP per capita"]]))

rcParams['figure.figsize'] = 8, 10
sns.heatmap(z, annot=True)
plt.show()

rcParams['figure.figsize'] = 8, 6
```



Dropping the data items with Z-score greater than the threshold

```
dataset_Zscore = dataset.drop(index=np.where(z>3.0)[0])
```

```
plt.scatter(dataset["GDP per capita"], dataset["Life Satisfaction"], c="r")
plt.scatter(dataset_Zscore["GDP per capita"], dataset_Zscore["Life Satisfaction"], c="b")
plt.xlabel("GDP per capita")
plt.ylabel("Life Satisfaction")
plt.show()
```



Model building

Linear Regression

```
X = np.c_[dataset_Zscore["GDP per capita"]]
Y = np.c_[dataset_Zscore["Life Satisfaction"]]
```

Building and Training a Linear Regression model

```
model = sklearn.linear_model.LinearRegression()
model.fit(X, Y)
```

```
LinearRegression(copy_X=True, fit_intercept=True, n_jobs=None,
         normalize=False)
```

Intercept and Coefficient values of the model

```
print("Model coefficient", model.coef_.item())
print("Model intercept", model.intercept_.item())
```

```
Model coefficient 3.2659628436612834e-05
Model intercept 5.448317258787462
```

Making predictions

```
Y_pred = model.predict(X)
```

```
plt.scatter(X, Y, c="b")
plt.plot(X, Y_pred, c='r')
```

```
    plt.xlabel("GDP per capita")
    plt.ylabel("Life Satisfaction")
    plt.show()
```