# CS 5565: INTRODUCTION TO STATISTICAL LEARNING

# World Happiness Report : Project Report

**Team Members**

Pradeepika Kolluru(16283597)

Anusha Muppalla(16286311)

**Description :** SDSN released the data and extracted it from the custom web crawling solution from PromptCloud. The World Happiness Report is a groundbreaking study of the global happiness condition which rates 156 countries by how happy they think their people are. This year's World Happiness Report focuses on happiness and the community: how happiness has evolved over the past twelve years, with a focus on innovations, social norms, disputes and government policies that have guided those changes.

**Dataset Description :** This dataset provides the happiness report data for 156 countries. It contains 11 columns. They are Country, Ladder, SD of Ladder, Positive affect, Negative affect, Social support, Freedom, Corruption, Generosity, Log of GDP per capita, Health life expectancy. **Country** : Name of the Country, **Ladder** : It is a measure of life satisfaction, **SD of Ladder :** Standard deviation of the Ladder, **Positive affect :** Measure of positive emotion, **Negative affect :** Measure of negative emotion, **Social support :** The degree to which social support has contributed to the Happiness Score measurement, **Freedom :** To what extent freedom contributed to Happiness Score measurement, **Corruption :** To what extent the Perception of Corruption contributes to the Score of Happiness, **Generosity :** The degree to which Generosity contributed to Happiness Score Measurement, **Log of GDP per capita :** The degree to which Generosity contributed to the Happiness Score measurement, **Health life expectancy :** To what degree Life expectancy applied to the Happiness Rating measurement.

Downloaded the dataset from Kaggle using the following link.

https://www.kaggle.com/PromptCloudHQ/world-happiness-report-2019

Loading the data using pandas

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import os
df = pd.read_csv('world-happiness-report-2019.csv')
```

Renaming columns in the dataset

```
df = df.rename(columns = {'Country (region)':'Country','SD of Ladder':'SD',
                          'Positive affect':'Positive','Negative affect':'Negative','Social support':'Social',
                          })
print(df.head())
```

```
E:\ISL\Happiness_Report>Main.py
      Country  Ladder  SD  Positive  Negative  Social  Freedom  Corruption  Generosity  Log of GDP\nper capita  Healthy life\nexpectancy
0     Finland       1   4      41.0      10.0     2.0      5.0         4.0        47.0                  22.0                    27.0
1     Denmark       2  13      24.0      26.0     4.0      6.0         3.0        22.0                  14.0                    23.0
2      Norway       3   8      16.0      29.0     3.0      3.0         8.0        11.0                   7.0                    12.0
3     Iceland       4   9       3.0       3.0     1.0      7.0        45.0         3.0                  15.0                    13.0
4  Netherlands      5   1      12.0      25.0    15.0     19.0        12.0         7.0                  12.0                    18.0
```

Displaying the summary of the data frame using info()

```
print(df.info())
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 156 entries, 0 to 155
Data columns (total 11 columns):
Country                           156 non-null object
Ladder                            156 non-null int64
SD                                156 non-null int64
Positive                          155 non-null float64
Negative                          155 non-null float64
Social                            155 non-null float64
Freedom                           155 non-null float64
Corruption                        148 non-null float64
Generosity                        155 non-null float64
Log of GDP
per capita      152 non-null float64
Healthy life
expectancy    150 non-null float64
dtypes: float64(8), int64(2), object(1)
memory usage: 13.5+ KB
None
```

Finding and printing the count of null values in each column of the data set.

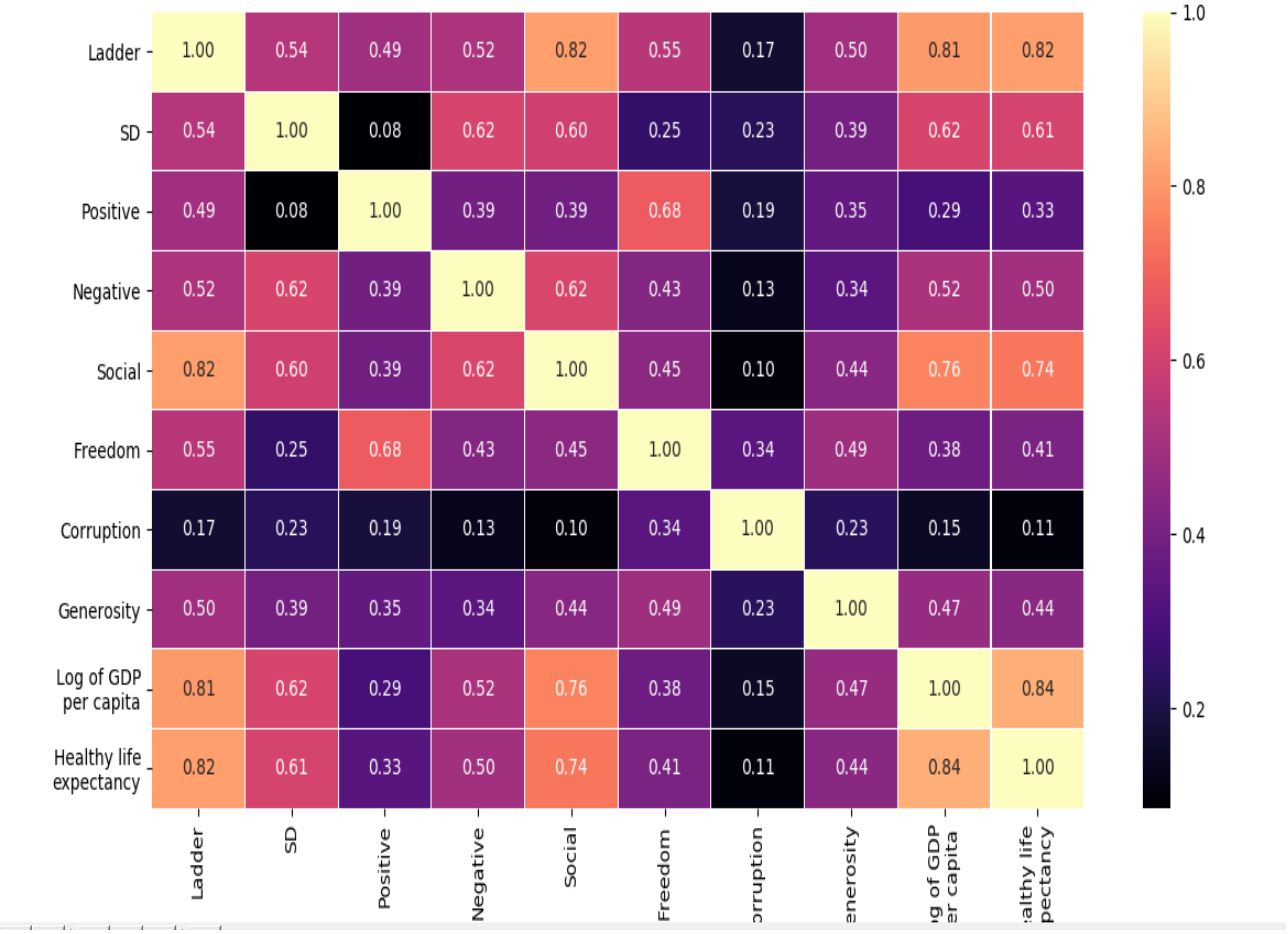
```
print(df.isnull().sum())
```

Filling all the null values

```
df = df.fillna(method = 'ffill')
```

Overall correlation matrix

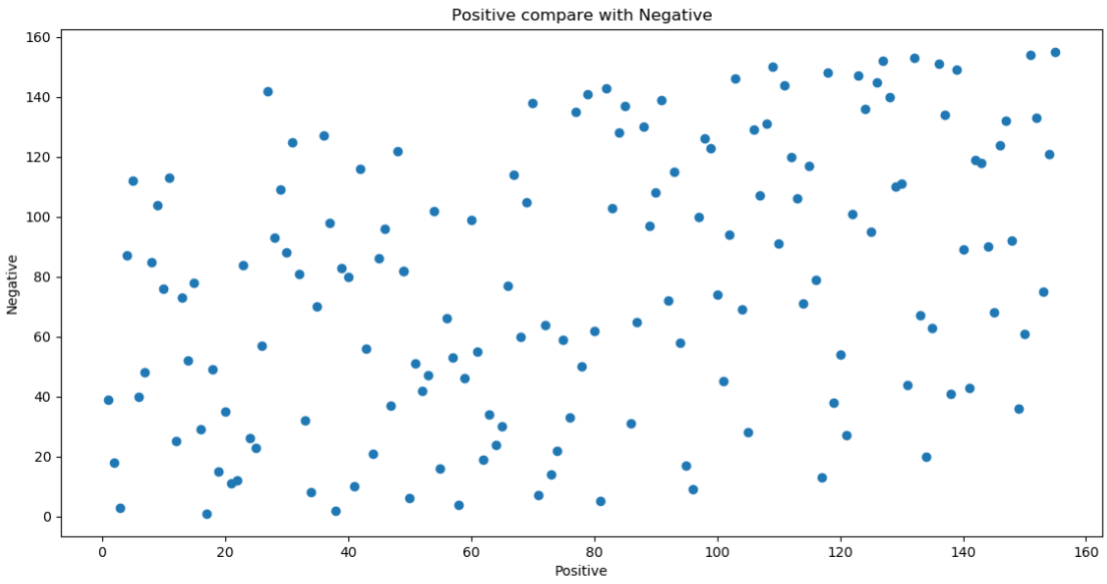
```
fig, ax = plt.subplots(figsize=(10,10))
sns.heatmap(df.corr(),ax=ax,annot=True,linewidth=0.05,fmt='.2f',cmap='magma')
plt.show()
```




This matrix displays the correlation between different attributes in the data set. Dark blue indicates the attributes that are lightly correlated and pale pink indicates the attributes that are highly correlated.

Comparing positive effect with the negative effect

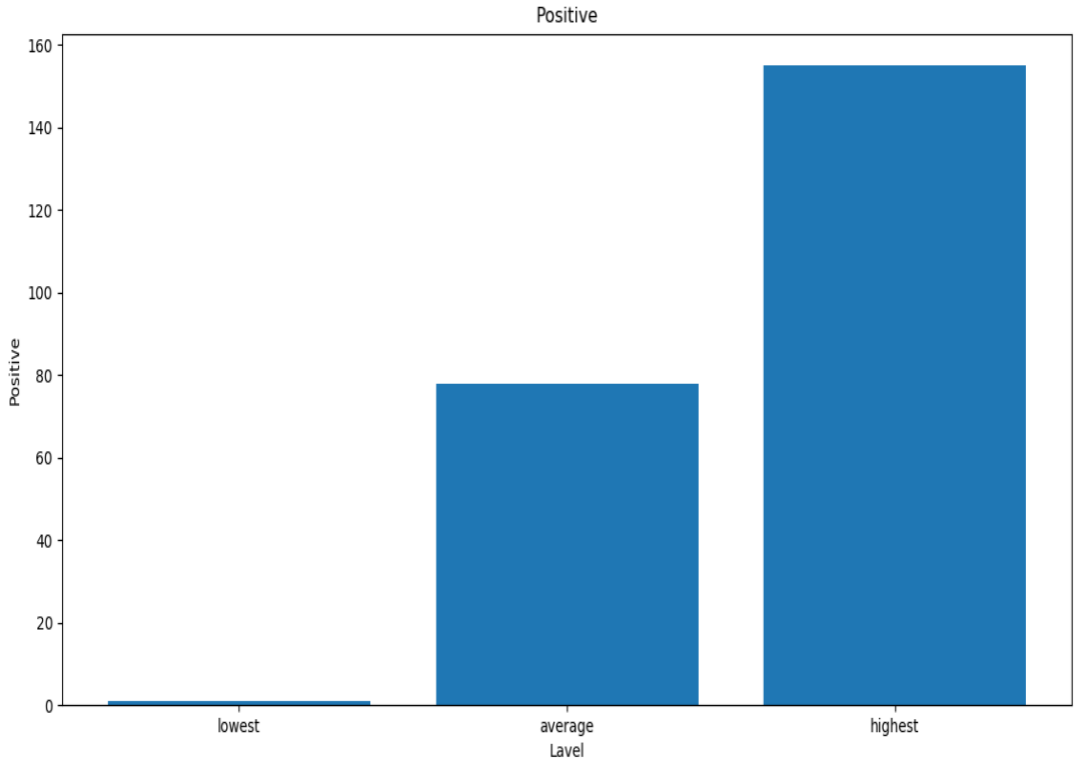
```
plt.scatter(df['Positive'],df.Negative)
plt.title('Positive compare with Negative')
plt.xlabel('Positive')
plt.ylabel('Negative')
plt.show()
```




In the above scatter plot, we have taken positive effect on the x-axis and negative effect on the y-axis. We can see that there are no outliers in the data.

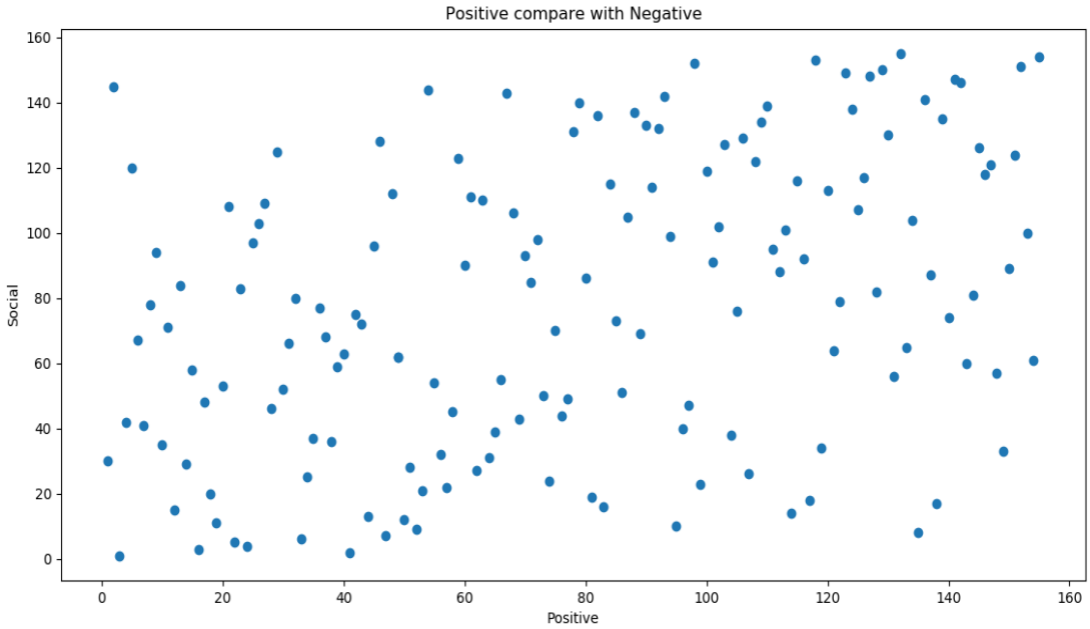
**Comparing "Positive effect" with every positive thing**

```
y = np.array([df['Positive'].min(),df['Positive'].mean(),df['Positive'].max()])
x = ['lowest','average','highest']
plt.bar(x,y)
plt.xlabel('Lavel')
plt.ylabel('Positive')
plt.title('Positive')
plt.show()
```




Comparing positive effect with Social support

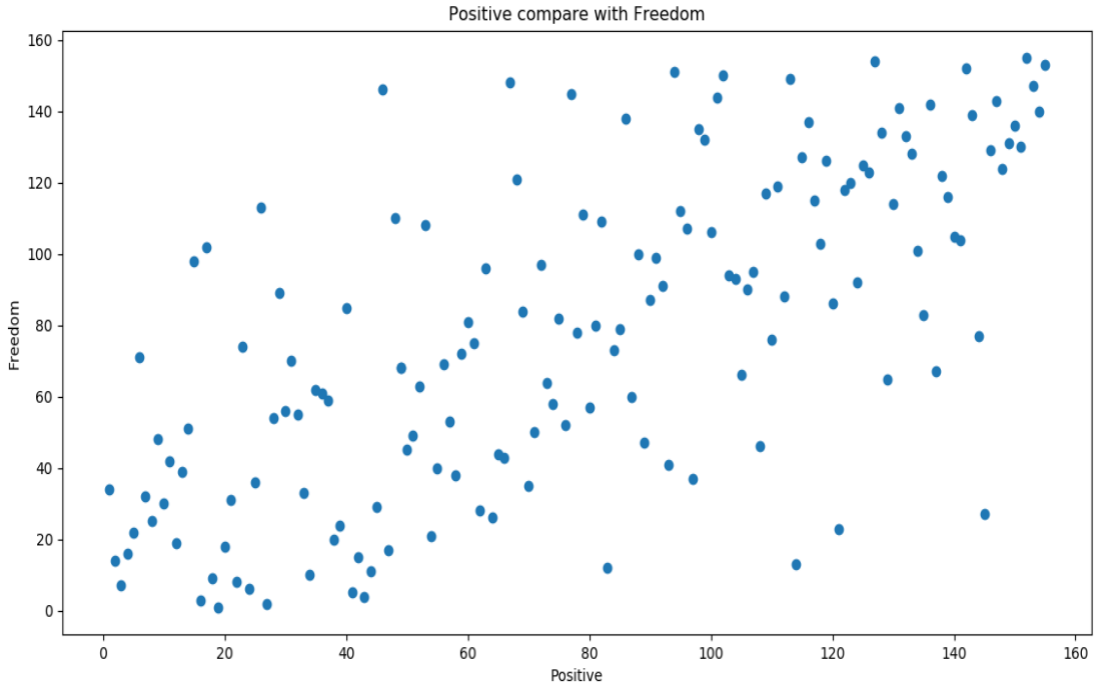
```
plt.scatter(df['Positive'],df.Social)
plt.xlabel('Positive')
plt.ylabel('Social')
plt.title('Positive compare with Negative')
plt.show()
```




In the above scatter plot we compared we compared positive effect by taking it on X-axis with Social support by taking it on y-axis.

Comparing positive effect with Freedom

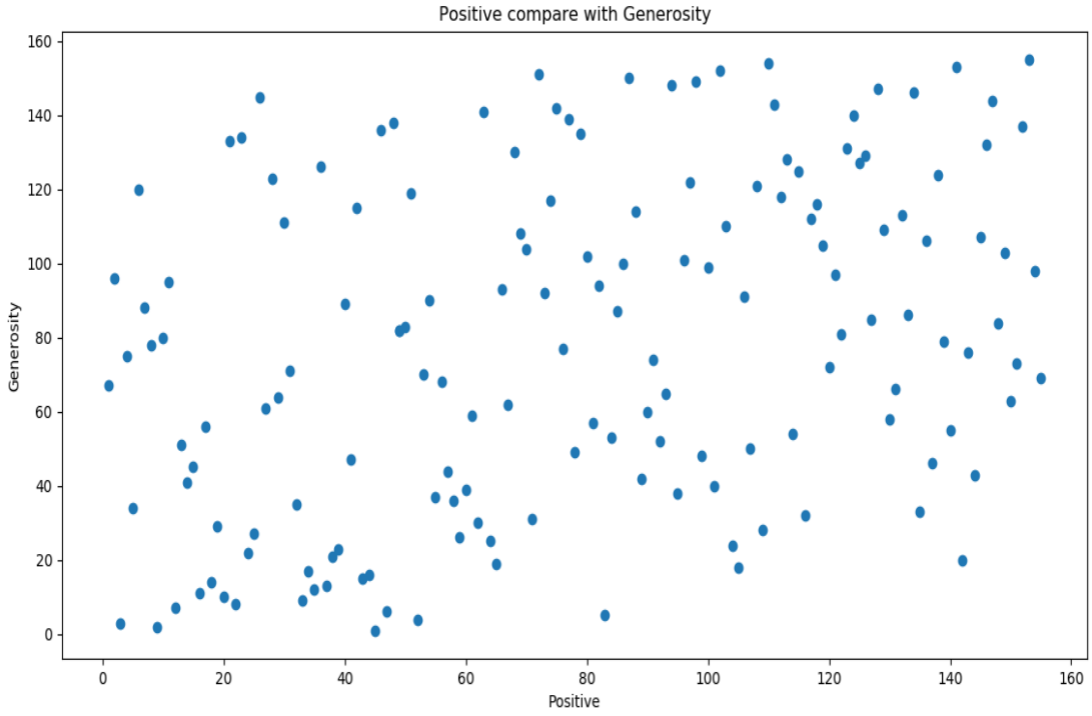
```
plt.scatter(df['Positive'],df.Freedom)
plt.xlabel('Positive')
plt.ylabel('Freedom')
plt.title('Positive compare with Freedom')
plt.show()
```




In the above scatter plot we compared positive effect by taking it on x-axis with Freedom on Y-axis.

Comparing positive effect with Generosity

```
plt.scatter(df['Positive'],df.Generosity)
plt.xlabel('Positive')
plt.ylabel('Generosity')
plt.title('Positive compare with Generosity')
plt.show()
```

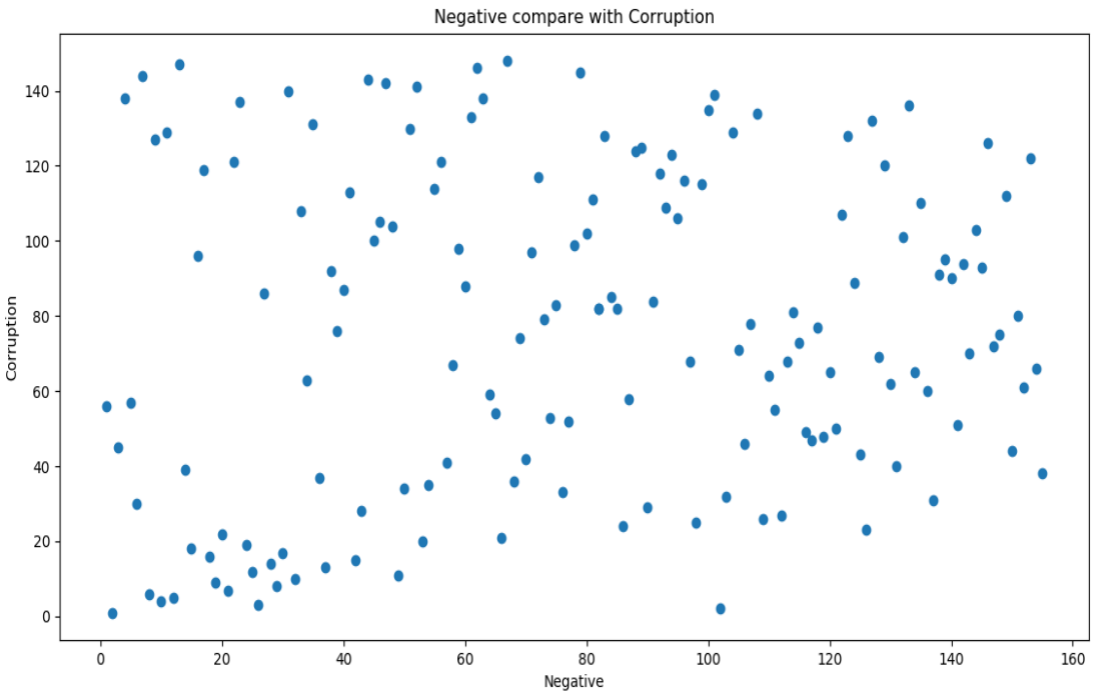



In the above scatter plot, we compared positive effect by taking it on X-axis with Generosity by taking it on Y-axis.

**Using “Negative effect” compared with every negative thing**

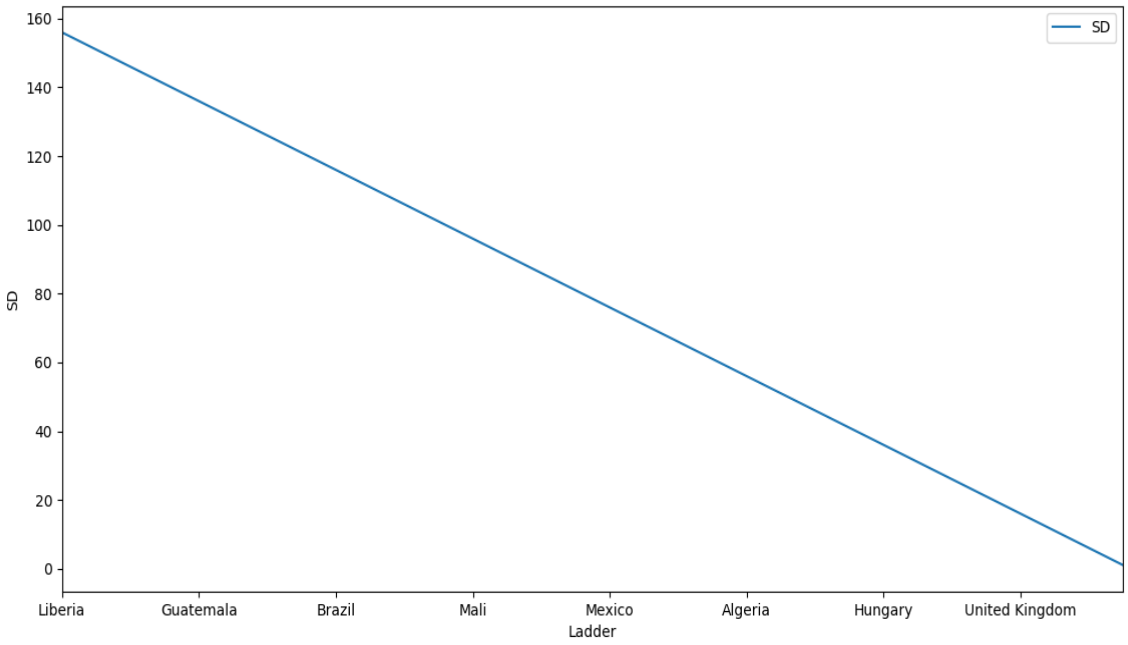
Comparing Negative effect with Corruption

```
plt.scatter(df['Negative'],df.Corruption)
plt.xlabel('Negative')
plt.ylabel('Corruption')
plt.title('Negative compare with Corruption')
plt.show()
```




Comparing SD with Ladder

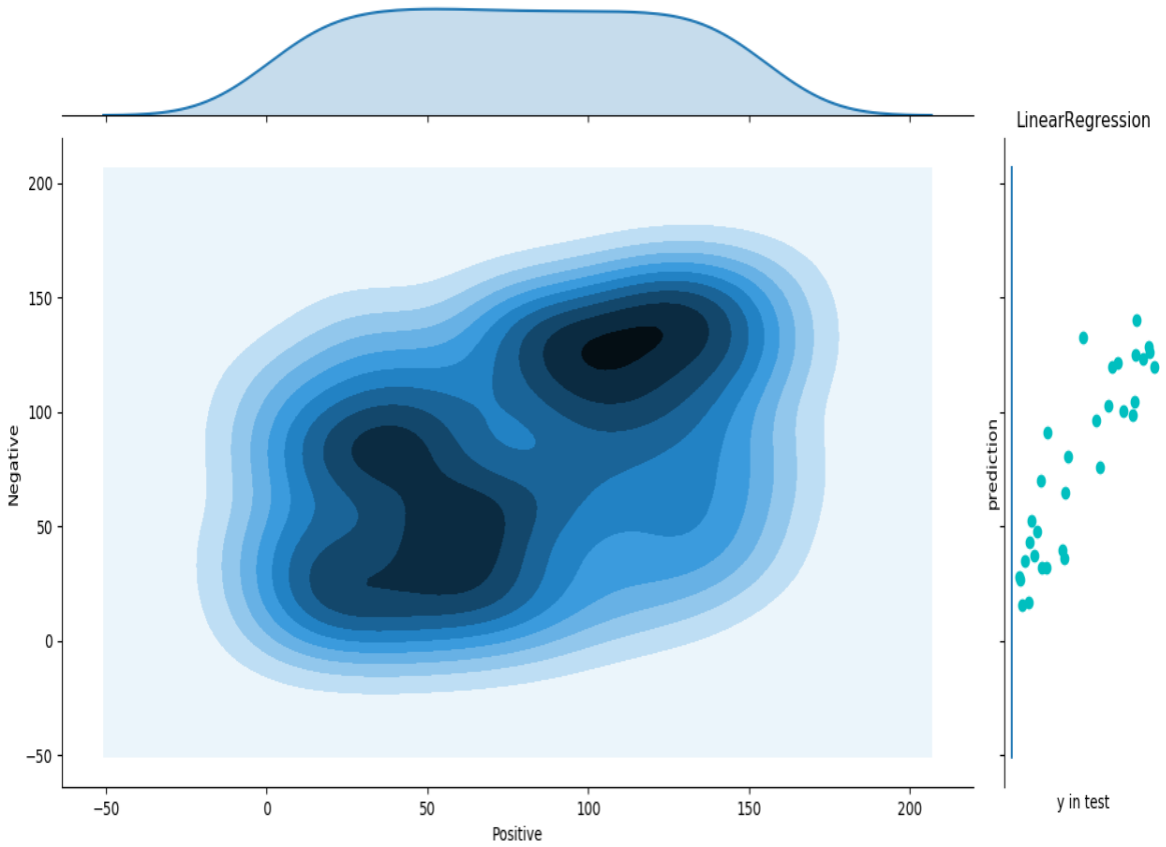
```
df.sort_values(by = ['SD'],ascending=False).plot(x='Country',y='SD')
plt.xlabel('Ladder')
plt.ylabel('SD')
plt.show()
```




Comparing all Positive effects with Negative effects using jointplot

Joint plot is used to find relation between two numeric variables. Here we are displaying the correlation between Negative and positive effects

```
sns.jointplot('Positive','Negative',data=df,kind='kde')
```




Splitting the data into training and testing datasets

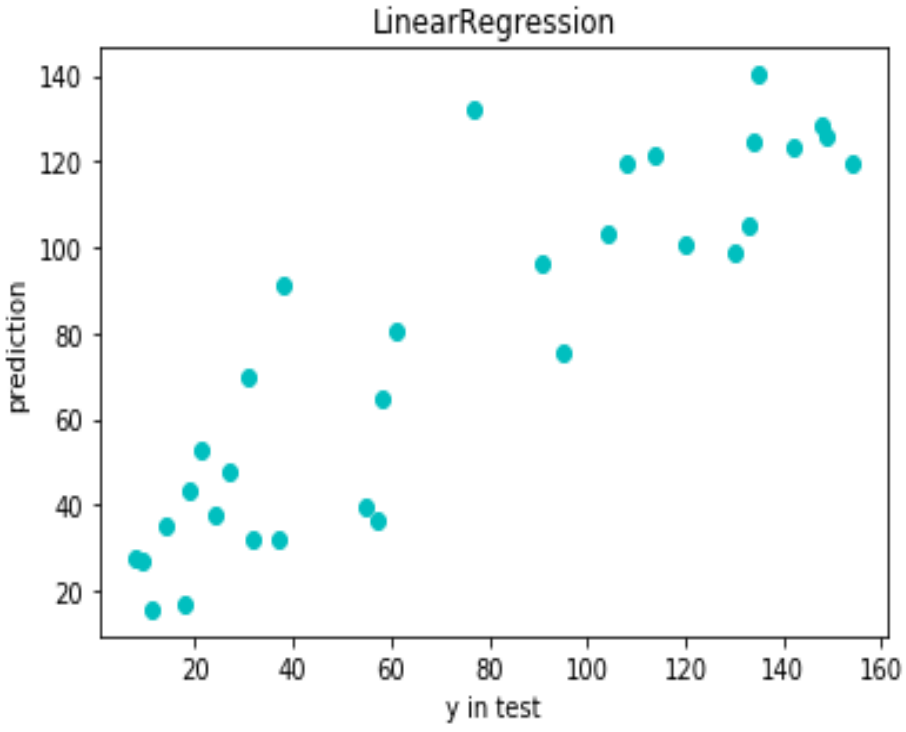
We are taking all attributes except the SD and country in X and SD in Y. Using train_test_split the dataset is divided into training and testing data sets.

```
from sklearn.model_selection import train_test_split
x = df.drop(['SD','Country'],axis=1)
y = df['SD']
x_train,x_test,y_train,y_test = train_test_split(x,y,test_size=0.2,random_state=2)
```

Applying Linear regression for the data set

Linear regression is a linear approach of modelling the relationship between a dependent variable and one or more explanatory variables.

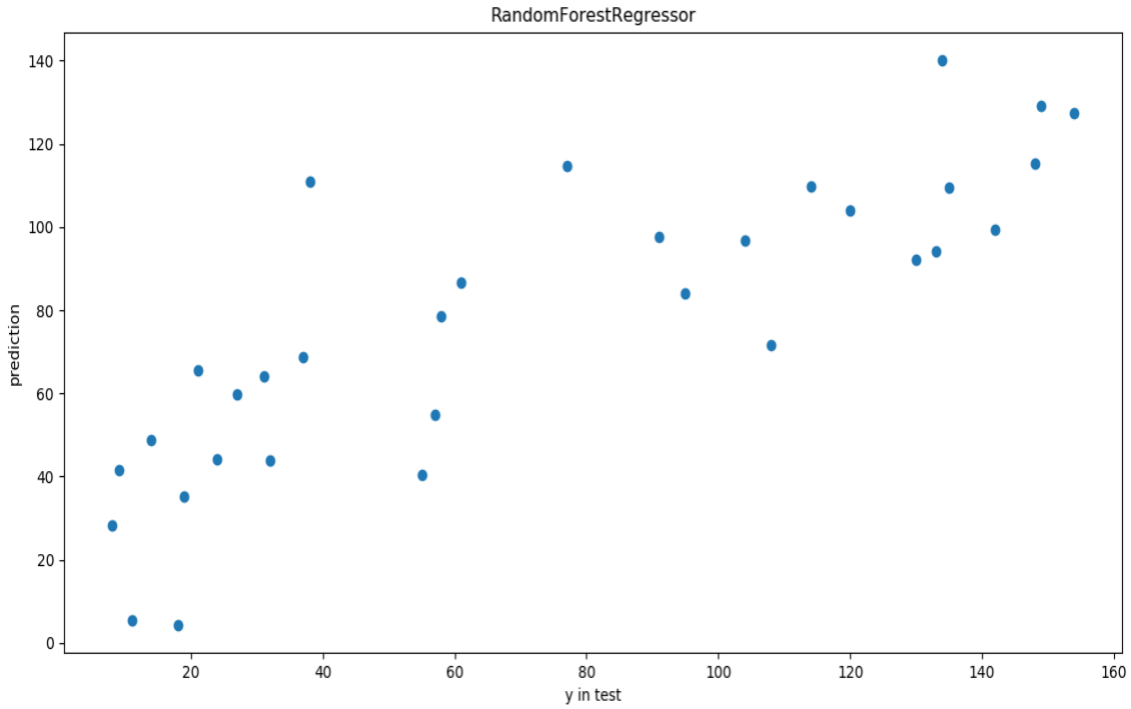
```
from sklearn.linear_model import LinearRegression
lr = LinearRegression()
lr.fit(x_train,y_train)
predict_lr = lr.predict(x_test)
print('r2 score using LinearRegression:' , r2_score(y_test,predict_lr))
plt.scatter(y_test,predict_lr,color='c')
plt.xlabel('y in test')
plt.ylabel('prediction')
plt.title('LinearRegression')
plt.show()
```



Applying Random forest regressor for the data set

Random forest fits a number of classifying decision trees on various sub samples of the dataset and uses averaging to improve the predictive accuracy and control over fitting.

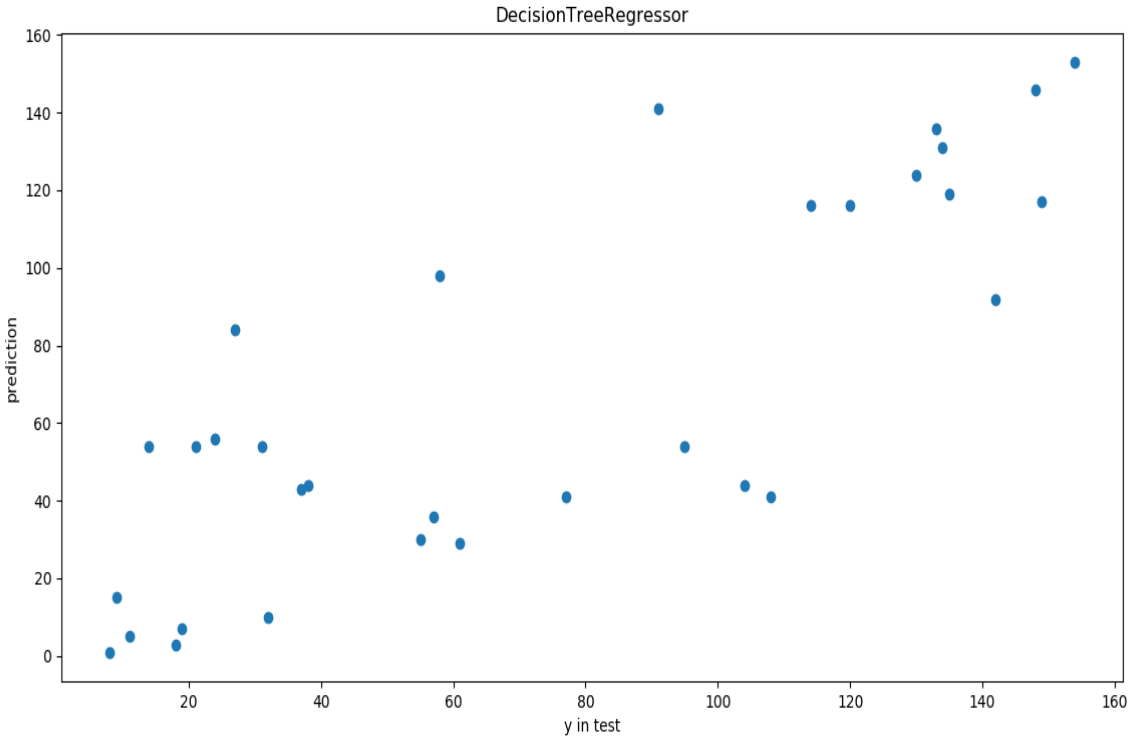
```
from sklearn.ensemble import RandomForestRegressor
rfg = RandomForestRegressor()
rfg.fit(x_train,y_train)
predict_rfg = rfg.predict(x_test)
print('r2 score:' , r2_score(y_test,predict_rfg))
plt.scatter(y_test,predict_rfg)
plt.xlabel('y in test')
plt.ylabel('prediction')
plt.title('RandomForestRegressor')
plt.show()
```



Applying Decision tree regressor for the data set

Decision tree builds regression or classification models in the form of tree structure. It breaks down a dataset into smaller and smaller subsets while at the same time an associated decision tree is incrementally developed.

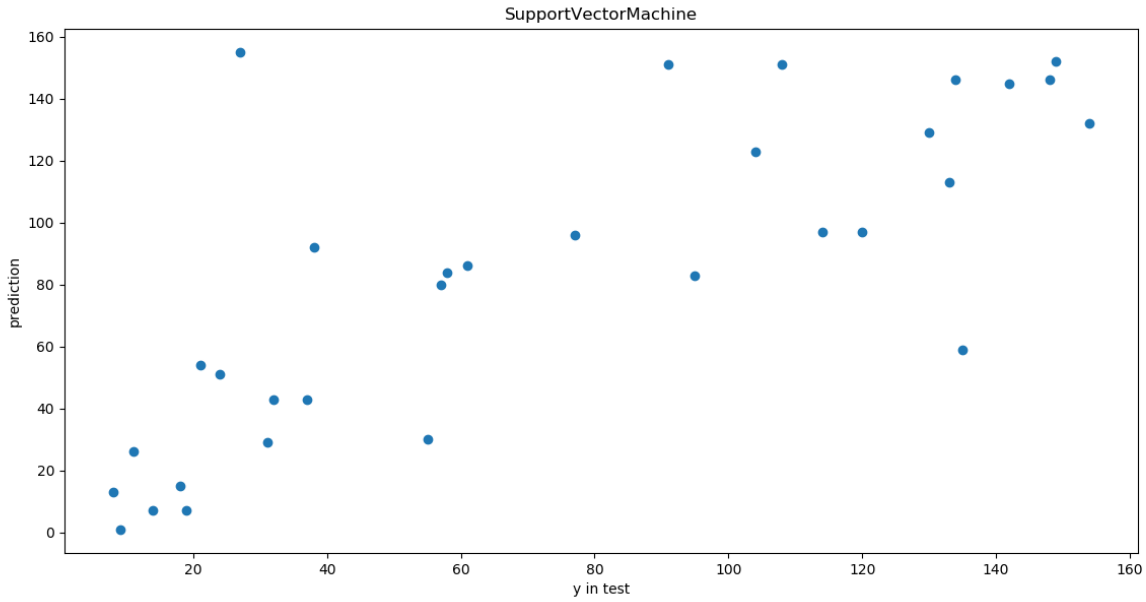
```
from sklearn.tree import DecisionTreeRegressor
dtr = DecisionTreeRegressor()
dtr.fit(x_train,y_train)
predict_dtr = dtr.predict(x_test)
print('r2 score:' , r2_score(y_test,predict_dtr))
plt.scatter(y_test,predict_dtr)
plt.xlabel('y in test')
plt.ylabel('prediction')
plt.title('DecisionTreeRegressor')
plt.show()
```



Applying SVM for the data set

SVM is a supervised machine learning algorithm, which can be used for classification or regression problems. SVM finds the optimal boundary between the possible inputs.

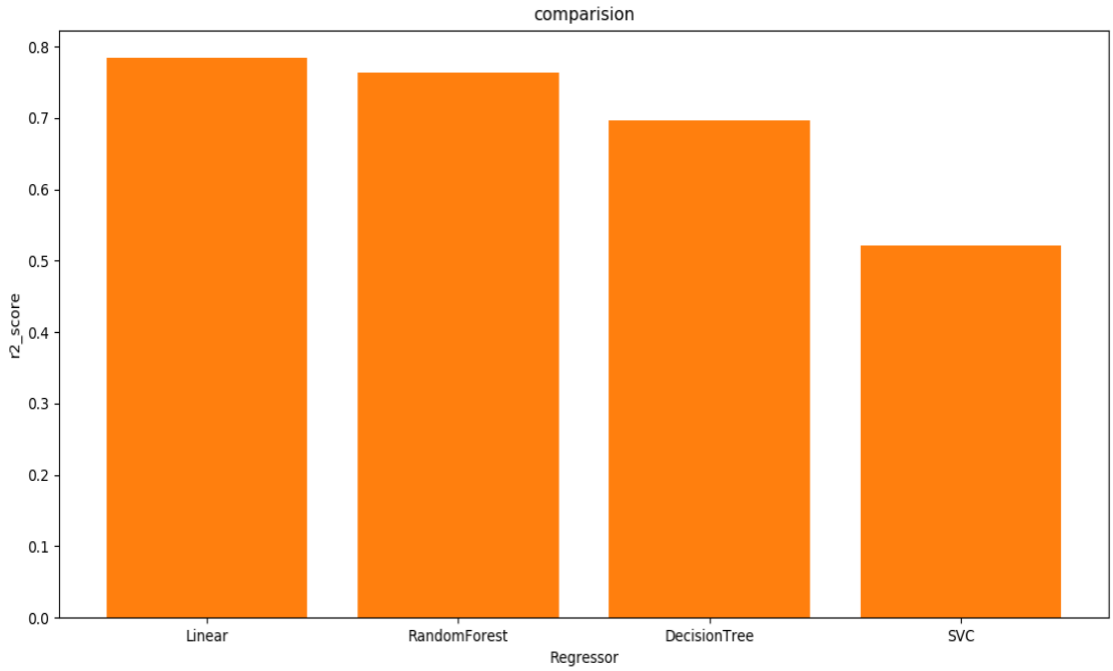
```
from sklearn import svm
svclassifier = svm.SVC(kernel='linear')
svclassifier.fit(x_train, y_train)
predict_svm = svclassifier.predict(x_test)
print('r2 score for svm:' , r2_score(y_test,predict_svm))
plt.scatter(y_test,predict_svm)
plt.xlabel('y in test')
plt.ylabel('prediction')
plt.title('SupportVectorMachine')
```



## Comparing Results:

```
y = np.array([r2_score(y_test,predict_lr),r2_score(y_test,predict_rfg),r2_score(y_test,predict_dtr),
r2_score(y_test,predict_svm)])
x = ['Linear','RandomForest','DecisionTree','SVC']
plt.bar(x,y)
plt.title('comparision')
plt.xlabel('Regressor')
plt.ylabel('r2_score')
plt.show()
```

```
E:\ISL\Happiness_Report>Main.py
r2 score using LinearRegression: 0.7839583862998767
E:\Python\Python37\lib\site-packages\sklearn\ensemble\forest.py:246: FutureWarning: The default value of n_estimators wi
ll change from 10 in version 0.20 to 100 in 0.22.
  "10 in version 0.20 to 100 in 0.22.", FutureWarning)
r2 score using Random Forest: 0.7627783115647516
r2 score using Decision tree: 0.6959117490779766
r2 score for svm: 0.5208769919702069
```



Conclusion:

In this, We had considered only correlated features other than SD of ladder and Country columns. Moreover, We have concentrated on supervised learning for that we have applied regression method i,e linear regression and classification methods as Random Forest, SupportVectorMachines and Decision Trees ,by comparing R^2 score is better for regression than classification methods and out of them the data better fits for Random Forest.