Assignment 3

Name:Pratiksha Gaikwad
Gr.No: 21920026
RollNo: 321066
TY-Comp Batch: A3

Aim :Build a Data model in Python for the dataset chosen in Assignment 1 or 2 and apply Linear Regression/Logistic Regression . Infer the result using accuracy score .

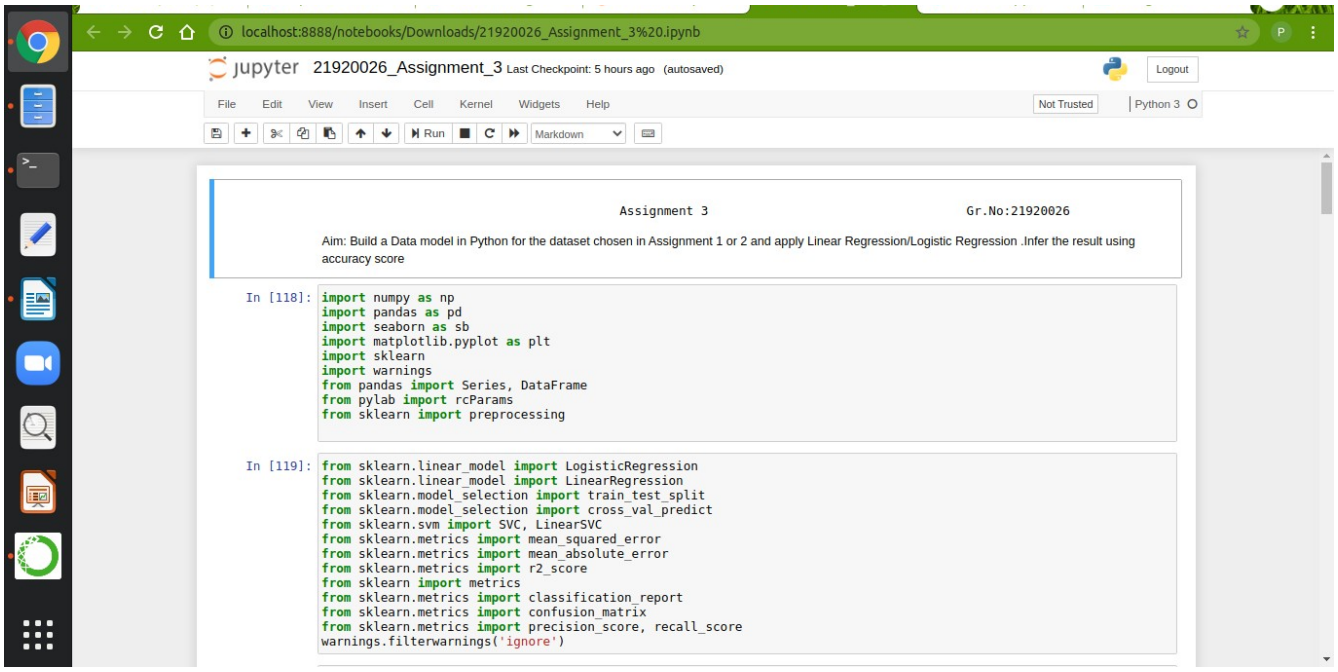
Outcome:

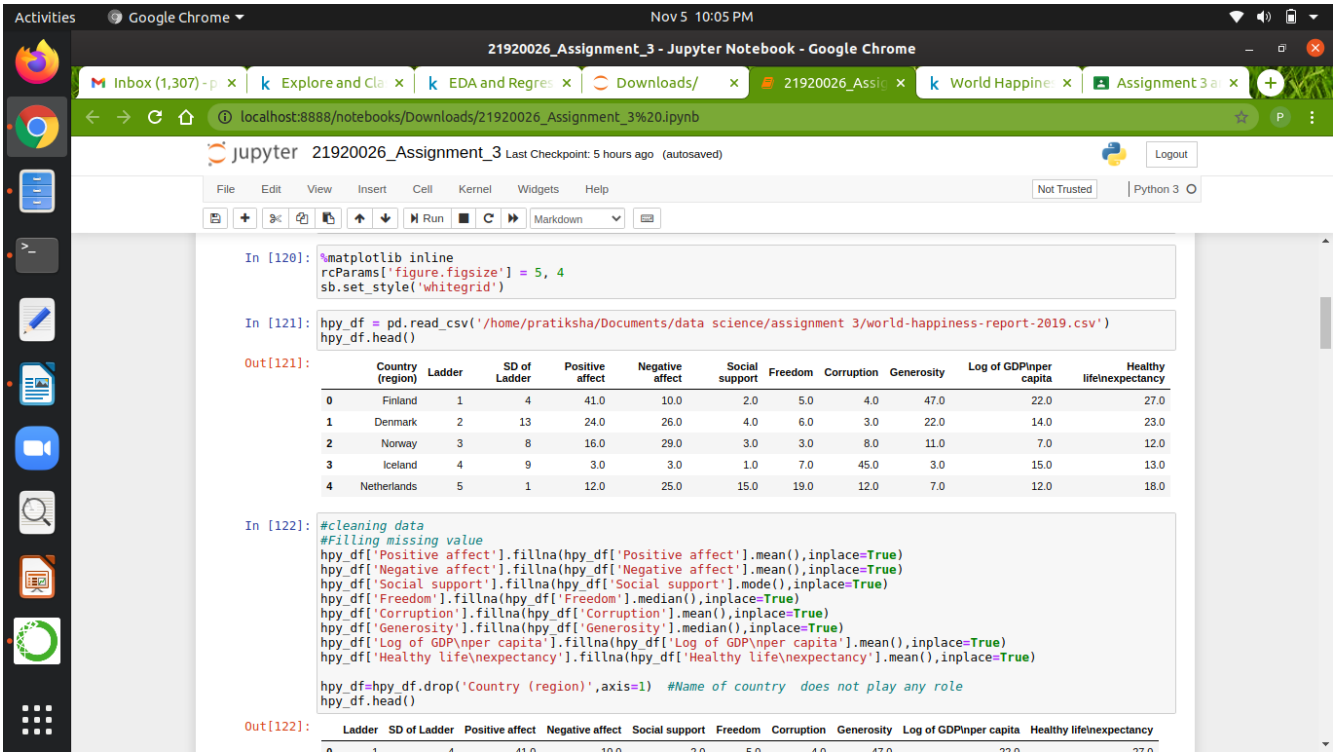
Dataset Name: World Happiness Report 2019

Overview of Dataset attribute:
Each column of data has the next description.

1. **Country (region)** : Name of the country.
2. **Ladder:** is a measure of life satisfaction.
3. **SD of Ladder:** Standard deviation of the ladder.
4. **Positive affect:** Measure of positive emotion.
5. **Negative affect:** Measure of negative emotion.
6. **Social support** :The extent to which Social support contributed to the calculation of the Happiness Score.
7. **Freedom:** The extent to which Freedom contributed to the calculation of the Happiness Score.
8. **Corruption:** The extent to which Perception of Corruption contributes to Happiness Score.
9. **Generosity:** The extent to which Generosity contributed to the calculation of the Happiness Score.
10. **Log of GDP per capita:** The extent to which GDP contributes to the calculation of the Happiness Score.
11. **Healthy life expectancy:** The extent to which Life expectancy contributed to the calculation of the Happiness Score.

Output





	Country (region)	Ladder	SD of Ladder	Positive affect	Negative affect	Social support	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

	Ladder	SD of Ladder	Positive affect	Negative affect	Social support	Freedom	Corruption	Generosity	Log of GDP\nper capita	Healthy life\nexpectancy

localhost:8888/notebooks/Downloads/21920026_Assignment_3%20.ipynb

jupyter 21920026_Assignment_3 Last Checkpoint: 5 hours ago (autosaved)

File Edit View Insert Cell Kernel Widgets Help

Out[122]:

	Ladder	SD of Ladder	Positive affect	Negative affect	Social support	Freedom	Corruption	Generosity	Log of GDP\nper capita	Healthy life\nexpectancy
0	1	4	41.0	10.0	2.0	5.0	4.0	47.0	22.0	27.0
1	2	13	24.0	26.0	4.0	6.0	3.0	22.0	14.0	23.0
2	3	8	16.0	29.0	3.0	3.0	8.0	11.0	7.0	12.0
3	4	9	3.0	3.0	1.0	7.0	45.0	3.0	15.0	13.0
4	5	1	12.0	25.0	15.0	19.0	12.0	7.0	12.0	18.0

Liner Regression

In [123]:
```
X= hpy_df.iloc[:,:-1].values
y= hpy_df.iloc[:,:11].values
```

Splitting the data into test and train

In [124]:
```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.2, random_state=0)
X_train.shape, X_test.shape, y_train.shape, y_test.shape
```

Out[124]: ((124, 9), (32, 9), (124, 10), (32, 10))

In [125]:
```
from sklearn import linear_model
linear_reg = linear_model.LinearRegression()
linear_reg.fit(X_train, y_train)
Y_pred = linear_reg.predict(X_test)
value_log= round(linear_reg.score(X_train, y_train) * 100, 2)
value_log
```

Out[125]: 97.52

localhost:8888/notebooks/Downloads/21920026_Assignment_3%20.ipynb

jupyter 21920026_Assignment_3 Last Checkpoint: 5 hours ago (autosaved)

File Edit View Insert Cell Kernel Widgets Help

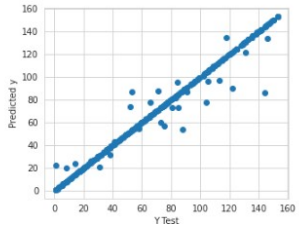
In [125]:
```
from sklearn import linear_model
linear_reg = linear_model.LinearRegression()
linear_reg.fit(X_train, y_train)
Y_pred = linear_reg.predict(X_test)
value_log= round(linear_reg.score(X_train, y_train) * 100, 2)
value_log
```

Out[125]: 97.52

Vizualizing the test and Predicted Results

In [126]:
```
plt.scatter(y_test, Y_pred)
plt.xlabel('Y Test')
plt.ylabel('Predicted y')
```

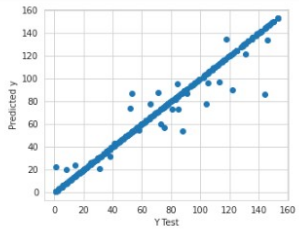
Out[126]: Text(0, 0.5, 'Predicted y')



localhost:8888/notebooks/Downloads/21920026_Assignment_3%20.ipynb

jupyter 21920026_Assignment_3 Last Checkpoint: 5 hours ago (autosaved)

File Edit View Insert Cell Kernel Widgets Help



Evaluating the Results

In [127]:
```
# Evaluating the data
from sklearn import metrics
print("Mean Absolute Error= ", metrics.mean_absolute_error(y_test, Y_pred))
print("Mean Squared Error= ", metrics.mean_squared_error(y_test, Y_pred))
print("Root Mean Squared Error= ", np.sqrt(metrics.mean_squared_error(y_test, Y_pred)))
acc=round(metrics.r2_score(y_test,Y_pred)*100,2)
print("Accuracy= ",acc)
```

```
Mean Absolute Error=  1.3612989669497302
Mean Squared Error=  33.98002936016782
Root Mean Squared Error=  5.8292391750697465
Accuracy=  97.78
```

Logistic Regression for Healthy life\nexpectancy

localhost:8888/notebooks/Downloads/21920026_Assignment_3%20.ipynb

jupyter 21920026_Assignment_3 Last Checkpoint: 5 hours ago (autosaved)

File Edit View Insert Cell Kernel Widgets Help

Logistic Regression for Healthy life\nexpectancy

In [128]:
```
X = hpy_df.iloc[:, 2:10]
y = hpy_df['Healthy life\nexpectancy']
```

In [129]:
```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3)
X_train.shape, X_test.shape, y_train.shape, y_test.shape
```

Out[129]:
```
((109, 8), (47, 8), (109,), (47,))
```

In [130]:
```
lab_enc = preprocessing.LabelEncoder()
training_scores_encoded = lab_enc.fit_transform(y_train)
```

In [131]:
```
clf = LogisticRegression()
clf.fit(X_train, training_scores_encoded)
print("LogisticRegression")
y_pred = clf.predict(X_test)
y_pred

acc=round(metrics.r2_score(y_test,y_pred)*100,2)
print(acc)
```

```
LogisticRegression
-5.25
```

In [132]:
```
print("Mean Absolute Error= ", metrics.mean_absolute_error(y_test, y_pred))
print("Mean Squared Error= ", metrics.mean_squared_error(y_test, y_pred))
print("Root Mean Squared Error= ", np.sqrt(metrics.mean_squared_error(y_test, y_pred)))
acc1=round(metrics.r2_score(y_test,y_pred)*100,2)
print("Accuracy= ",acc1)
```

```
Mean Absolute Error=  34.51063829787234
```

localhost:8888/notebooks/Downloads/21920026_Assignment_3%20.ipynb

jupyter 21920026_Assignment_3 Last Checkpoint: 5 hours ago (autosaved)

File Edit View Insert Cell Kernel Widgets Help

Logistic Regression for Healthy life\nexpectancy

In [128]:
```
X = hpy_df.iloc[:, 2:10]
y = hpy_df['Healthy life\nexpectancy']
```

In [129]:
```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3)
X_train.shape, X_test.shape, y_train.shape, y_test.shape
```

Out[129]:
```
((109, 8), (47, 8), (109,), (47,))
```

In [130]:
```
lab_enc = preprocessing.LabelEncoder()
training_scores_encoded = lab_enc.fit_transform(y_train)
```

In [131]:
```
clf = LogisticRegression()
clf.fit(X_train, training_scores_encoded)
print("LogisticRegression")
y_pred = clf.predict(X_test)
y_pred

acc=round(metrics.r2_score(y_test,y_pred)*100,2)
print(acc)
```

```
LogisticRegression
-5.25
```

In [132]:
```
print("Mean Absolute Error= ", metrics.mean_absolute_error(y_test, y_pred))
print("Mean Squared Error= ", metrics.mean_squared_error(y_test, y_pred))
print("Root Mean Squared Error= ", np.sqrt(metrics.mean_squared_error(y_test, y_pred)))
acc1=round(metrics.r2_score(y_test,y_pred)*100,2)
print("Accuracy= ",acc1)
```

localhost:8888/notebooks/Downloads/21920026_Assignment_3%20.ipynb

jupyter 21920026_Assignment_3 Last Checkpoint: 5 hours ago (autosaved)

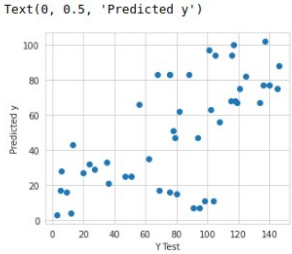
File Edit View Insert Cell Kernel Widgets Help

In [132]:
```
print("Mean Absolute Error= ", metrics.mean_absolute_error(y_test, y_pred))
print("Mean Squared Error= ", metrics.mean_squared_error(y_test, y_pred))
print("Root Mean Squared Error= ", np.sqrt(metrics.mean_squared_error(y_test, y_pred)))
acc1=round(metrics.r2_score(y_test,y_pred)*100,2)
print("Accuracy= ",acc1)
```

```
Mean Absolute Error=  34.51063829787234
Mean Squared Error=  1874.9255319148936
Root Mean Squared Error=  43.30041029730427
Accuracy=  -5.25
```

In [133]:
```
plt.scatter(y_test, y_pred)
plt.xlabel('Y Test')
plt.ylabel('Predicted y')
```

Out[133]:
```
Text(0, 0.5, 'Predicted y')
```



using random values of attribute of dataset and chacking accuracy with SD of ladder which will rename as ranking of

localhost:8888/notebooks/Downloads/21920026_Assignment_3%20.ipynb

jupyter 21920026_Assignment_3 Last Checkpoint: 5 hours ago (autosaved)

File Edit View Insert Cell Kernel Widgets Help

using random values of attribute of dataset and checking accuracy with SD of ladder which will rename as ranking of happiness

In [156]:
```
hpy_df = pd.read_csv('/home/pratiksha/Documents/data science/assignment 4/world-happiness-report-2019.csv')
hpy_df.head()
```

Out[156]:

	Country (region)	Ladder	SD of Ladder	Positive affect	Negative affect	Social support	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

In [157]:
```
hpy_df.rename(columns={'SD of Ladder':'Ranking'}, inplace = True)
hpy_df.fillna(0, inplace = True)
hpy_df.head()
```

Out[157]:

	Country (region)	Ladder	Ranking	Positive affect	Negative affect	Social support	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

In [148]:
```
attributes = ['Negative affect','Social support','Log of GDP\nper capita','Healthy life\nexpectancy','Positive affec
```

Activities Google Chrome Nov 5 10:06 PM

21920026_Assignment_3 - Jupyter Notebook - Google Chrome

Inbox (1,307) - p | Explore and Cla | EDA and Regres | Downloads/ | 21920026_Assig | World Happine | Assignment 3 a

localhost:8888/notebooks/Downloads/21920026_Assignment_3%20.ipynb

jupyter 21920026_Assignment_3 Last Checkpoint: 5 hours ago (autosaved)

File Edit View Insert Cell Kernel Widgets Help

In [148]:
```
attributes = ['Negative affect','Social support','Log of GDP\nper capita','Healthy life\nexpectancy','Positive affec
              'Freedom','Corruption','Generosity']
X = hpy_df[attributes]
y = hpy_df.Ranking
```

In [149]:
```
X_train, X_test, y_train, y_test = train_test_split(X,y,test_size = 0.2, random_state = 1)
print(X_train.shape,y_train.shape)
print(X_test.shape, y_test.shape)
```

```
(124, 8) (124,)
(32, 8) (32,)
```

In [150]:
```
lr =linear_model.LinearRegression()
lr= lr.fit(X_train,y_train)
predicted = cross_val_predict(lr, X_train, y_train, cv=10)
value_log= round(lr.score(X_train, y_train) * 100, 2)
value_log
```

Out[150]:
```
59.8
```

In [151]:
```
ranking_predict = lr.predict(X_test)
print('Root Squared:',r2_score(y_test,ranking_predict))
print('Root Mean Squared Error:', np.sqrt(mean_squared_error(y_test,ranking_predict)))
print('score:', lr.score(X_train, y_train)*100)
```

```
Root Squared: 0.5332873046688127
Root Mean Squared Error: 30.832605311590594
score: 59.7986864065365
```

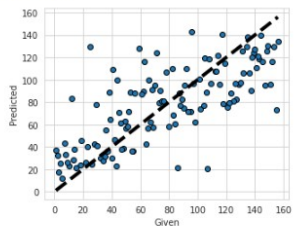
In [152]:
```
fig, ax = plt.subplots()
ax.scatter(y_train, predicted, edgecolors=(0, 0, 0))
ax.plot([y.min(), y.max()], [y.min(), y.max()], 'k--', lw=4)
```

localhost:8888/notebooks/Downloads/21920026_Assignment_3%20.ipynb

jupyter 21920026_Assignment_3 Last Checkpoint: 5 hours ago (autosaved)

File Edit View Insert Cell Kernel Widgets Help

In [152]:
```
fig, ax = plt.subplots()
ax.scatter(y_train, predicted, edgecolors=(0, 0, 0))
ax.plot([y.min(), y.max()], [y.min(), y.max()], 'k--', lw=4)
ax.set_xlabel('Given')
ax.set_ylabel('Predicted')
plt.show()
```



Logistic Regression

In [153]:
```
attributes = ['Negative affect','Social support','Log of GDP\nper capita','Healthy life\nexpectancy',
              'Positive affect', 'Freedom','Corruption','Generosity']
X = hpy_df[attributes]
y = hpy_df.Ranking

X_train, X_test, y_train, y_test = train_test_split(X,y,test_size = 0.2)
print(X_train.shape,y_train.shape)
```

localhost:8888/notebooks/Downloads/21920026_Assignment_3%20.ipynb

jupyter 21920026_Assignment_3 Last Checkpoint: 5 hours ago (autosaved)

File Edit View Insert Cell Kernel Widgets Help

Logistic Regression

In [153]:
```
attributes = ['Negative affect','Social support','Log of GDP\nper capita','Healthy life\nexpectancy',
              'Positive affect', 'Freedom','Corruption','Generosity']
X = hpy_df[attributes]
y = hpy_df.Ranking

X_train, X_test, y_train, y_test = train_test_split(X,y,test_size = 0.2)
print(X_train.shape,y_train.shape)
print(X_test.shape, y_test.shape)
```


```
(124, 8) (124,)
(32, 8) (32,)
```

In [154]:
```
logistic_regression = LogisticRegression()
logistic_regression.fit(X_train, y_train)
Y_predict = logistic_regression.predict(X_test)
value_log= round(logistic_regression.score(X_train, y_train) * 100, 2)
value_log
```

Out[154]: 100.0

In [155]:
```
ranking_predict = logistic_regression.predict(X_test)
print('Root Squared:',r2_score(y_test,ranking_predict))
print('Root Mean Squared Error:', np.sqrt(mean_squared_error(y_test,ranking_predict)))
print('score:', logistic_regression.score(X_train, y_train)*100)
```

```
Root Squared: 0.18335088694094526
Root Mean Squared Error: 39.22570904394209
score: 100.0
```

In [ ]:

Description:

- **Regression** : The term regression is used when you try to find the relationship between a dependent (target) and independent variable (s) (predictor).
- **Linear regression** is used to predict the continuous dependent variable using a given set of independent variables. There is 97.7 % accuracy found in liner regression for overall dataset
- **Logistic Regression** is used to predict the categorical dependent variable using a given set of independent variables.
- After implementing a machine learning algorithm, the next step we move towards is to find how effective our model is based on some metrics. This is the most essential part of any project as different performance metrics are used to evaluate different Machine Learning algorithms. Following 3 main metrics for model evaluation in regression:
- **Mean Absolute Error** – Mean Absolute Error is the average of the absolute difference between the Original Values and the Predicted Values of data. For linear regression it is 1.36% and logistic regrssion it is 34.5% found.
- **Mean Squared Error** – Mean Squared Error is much like Mean Absolute Error except that It finds the average squared error between the predicted and actual values . For linear regression it is 33.98% and logistic regrssion it is 1874.9% found.
- **Root Mean Squared Error** – Root Mean Squared Error (RMSE) measures the average magnitude of the error by taking the square root of the average of squared differences between prediction and actual observation. For linear regression it is 5.82% and logistic regrssion it is 43.3% found.
- In datset 'SD of Ladder' is rename as '**Ranking**' and by using it with other attribute for traning and testing data in 2nd linear regression. It is found that 59.8% ranking attribute impacting on other attribute.

Interpretation:

- In linear regression got very good accuracy i.e. 97.7% .
- SD of life satisfaction (Ranking) is 59% impacting on happiness level.