Simple Linear Regression

```
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd
```

```
dataset = pd.read_csv('/content/energy_statistics1 MALAK.csv')
X = dataset.iloc[:, :-1].values
y = dataset.iloc[:, -1].values
```

```
print(X)
```

```
[['Germany' 2014]
 ['Germany' 2013]
 ['Germany' 2012]
 ['Germany' 2011]
 ['Germany' 2010]
 ['Germany' 2009]
 ['Germany' 2008]
 ['Germany' 2007]
 ['Germany' 2006]
 ['Germany' 2005]
 ['Germany' 2004]
 ['Germany' 2003]
 ['Germany' 2002]
 ['Germany' 2001]
 ['Germany' 2000]
 ['Germany' 1999]
 ['Germany' 1998]
 ['Germany' 1997]
 ['Germany' 1996]
 ['Germany' 1995]
 ['Germany' 1994]
 ['Germany' 1993]
 ['Germany' 1992]
 ['Germany' 1991]
 ['Turkey' 2014]]
```

```
print(y)
```

```
[36056 31010 26380 19599 11729  6584  4420  3075  2220  1282   557   313
   188   116    60    30    35    18    12     7     7     3     4     1
    17]
```

```
from sklearn.preprocessing import LabelEncoder, OneHotEncoder
from sklearn.compose import ColumnTransformer
ct = ColumnTransformer([("country_or_area", OneHotEncoder(), [0])], remainder = 'passthrou
X = np.array(ct.fit_transform(X))
```

```
print(X)
```

```
[[1.0 0.0 2014]
 [1.0 0.0 2013]
 [1.0 0.0 2012]
 [1.0 0.0 2011]
 [1.0 0.0 2010]
 [1.0 0.0 2009]
 [1.0 0.0 2008]
 [1.0 0.0 2007]
 [1.0 0.0 2006]
 [1.0 0.0 2005]
 [1.0 0.0 2004]
 [1.0 0.0 2003]
 [1.0 0.0 2002]
 [1.0 0.0 2001]
 [1.0 0.0 2000]
 [1.0 0.0 1999]
 [1.0 0.0 1998]
 [1.0 0.0 1997]
 [1.0 0.0 1996]
 [1.0 0.0 1995]
 [1.0 0.0 1994]
 [1.0 0.0 1993]
 [1.0 0.0 1992]
 [1.0 0.0 1991]
 [0.0 1.0 2014]]
```

```
from sklearn.preprocessing import LabelEncoder
le = LabelEncoder()
y = le.fit_transform(y)
```

```
print(y)
```

```
[23 22 21 20 19 18 17 16 15 14 13 12 11 10  9  7  8  6  4  3  3  1  2  0
  5]
```

```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.25, random_state =
```

```
print(X_train)
```

```
[[0.0 1.0 2014]
 [1.0 0.0 1991]
 [1.0 0.0 2000]
 [1.0 0.0 2013]
 [1.0 0.0 2004]
 [1.0 0.0 2001]
 [1.0 0.0 2006]
 [1.0 0.0 2008]
 [1.0 0.0 1996]
 [1.0 0.0 2010]
 [1.0 0.0 2005]
 [1.0 0.0 2007]
 [1.0 0.0 1994]
 [1.0 0.0 2011]
 [1.0 0.0 2014]
```

```
       [1.0 0.0 1993]
       [1.0 0.0 1999]
       [1.0 0.0 2002]]
```

```
print(X_test)
```

```
     [[1.0 0.0 2009]
      [1.0 0.0 2012]
      [1.0 0.0 1995]
      [1.0 0.0 1998]
      [1.0 0.0 2003]
      [1.0 0.0 1992]
      [1.0 0.0 1997]]
```

```
print(y_train)
```

```
     [ 5  0  9 22 13 10 15 17  4 19 14 16  3 20 23  1  7 11]
```

```
print(y_test)
```

```
     [18 21  3  8 12  2  6]
```

```
from sklearn.preprocessing import StandardScaler
sc = StandardScaler()
X_train[:, 2:] = sc.fit_transform(X_train[:, 2:])
X_test[:, 2:] = sc.fit_transform(X_test[:, 2:])
```

```
print(X_train)
```

```
     [[0.0 1.0 1.4431483518177577]
      [1.0 0.0 -1.8039354397722132]
      [1.0 0.0 -0.533337434367442]
      [1.0 0.0 1.301970795661672]
      [1.0 0.0 0.0313727902569008]
      [1.0 0.0 -0.3921598782113563]
      [1.0 0.0 0.3137279025690722]
      [1.0 0.0 0.5960830148812436]
      [1.0 0.0 -1.0980476589917847]
      [1.0 0.0 0.878438127193415]
      [1.0 0.0 0.17255034641298647]
      [1.0 0.0 0.45490545872515786]
      [1.0 0.0 -1.3804027713039562]
      [1.0 0.0 1.0196156833495007]
      [1.0 0.0 1.4431483518177577]
      [1.0 0.0 -1.5215803274600417]
      [1.0 0.0 -0.6745149905235276]
      [1.0 0.0 -0.2509823220552706]]
```

```
print(X_test)
```

```
     [[1.0 0.0 1.1844195797658157]
      [1.0 0.0 1.6207846881005914]
      [1.0 0.0 -0.851950925796472]
```

```
       [1.0 0.0 -0.41558581746169604]
       [1.0 0.0 0.31168936309626377]
       [1.0 0.0 -1.2883160341312478]
       [1.0 0.0 -0.561040853573288]]
```

```
from sklearn.linear_model import LinearRegression
regressor = LinearRegression()
regressor.fit(X_train, y_train)
```

```
     LinearRegression()
```

```
y_pred = regressor.predict(X_test)
print(y_pred)
```

```
     [21.23514076 24.4100083   6.41909228  9.59395981 14.8854057   3.24422475
       8.53567064]
```

```
y_pred_vertiacl = y_pred.reshape(len(y_pred), 1)
y_pred_vertiacl = np.round(y_pred_vertiacl)
print(y_pred_vertiacl)
```

```
     [[21.]
      [24.]
      [ 6.]
      [10.]
      [15.]
      [ 3.]
      [ 9.]]
```

```
y_true_vertiacl = y_test.reshape(len(y_test), 1)
y_true_vertiacl = np.round(y_true_vertiacl)
print(y_true_vertiacl)
```

```
     [[18]
      [21]
      [ 3]
      [ 8]
      [12]
      [ 2]
      [ 6]]
```

```
true_pred = np.concatenate((y_true_vertiacl, y_pred_vertiacl), axis = 1)
print(true_pred)
```

```
     [[18. 21.]
      [21. 24.]
      [ 3.  6.]
      [ 8. 10.]
      [12. 15.]
      [ 2.  3.]
      [ 6.  9.]]
```

```
from sklearn.metrics import mean_absolute_error
```

```
mae = mean_absolute_error(y_test, y_pred)
print(mae)
```

```
2.617643177619442
```

```
from sklearn.metrics import r2_score
score = r2_score(y_test, y_pred)
print(score)
```

```
0.8365575182377639
```

```
prediction = regressor.predict([[2014,2020,2025]])
print(prediction)
```

```
[14682.52606391]
```

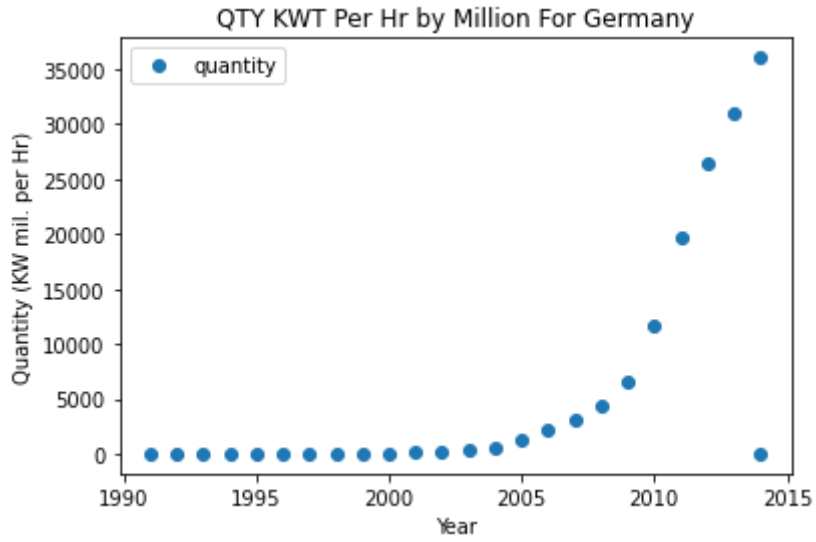
```
print(regressor.coef_)
print(regressor.intercept_)
```

```
[ 9.0587886  -9.0587886   7.27571355]
3.5588545790446577
```

```
dataset.plot(x='year', y='quantity', style='o')
plt.title('QTY KWT Per Hr by Million For Turkey')
plt.xlabel('Year')
plt.ylabel('Quantity (KW mil. per Hr)')
ypoints = np.array(y_pred)
plt.plot(X,ypoints)
plt.show()
x1 = np.array([2015,2016,2017,2018,2019,2020,2021,2022,2023,2024,2025])
predy=2014
for i in range(11):
    predy=predy+1
    print(x1[i],' ',y_pred1)
```


```
#y_pred1 = model.intercept_ + model.coef_ * X
#print('predicted response1:', y_pred1)
dataset.plot(x='year', y='quantity', style='o')

plt.title('QTY KWT Per Hr by Million For Germany')
plt.xlabel('Year')
plt.ylabel('Quantity (KW mil. per Hr)')
ypoints = np.array(y_pred)
plt.show()
x1 = np.array([2015,2016,2017,2018,2019,2020,2021,2022,2023,2024,2025])
predy=2014
for i in range(11):
    predy=predy+1
    print(x1[i],' ',y_pred1)
```




```
2015   [ 18347.60576669 -18340.48805753  14736.8787955 ]
2016   [ 18347.60576669 -18340.48805753  14736.8787955 ]
2017   [ 18347.60576669 -18340.48805753  14736.8787955 ]
2018   [ 18347.60576669 -18340.48805753  14736.8787955 ]
2019   [ 18347.60576669 -18340.48805753  14736.8787955 ]
2020   [ 18347.60576669 -18340.48805753  14736.8787955 ]
2021   [ 18347.60576669 -18340.48805753  14736.8787955 ]
2022   [ 18347.60576669 -18340.48805753  14736.8787955 ]
2023   [ 18347.60576669 -18340.48805753  14736.8787955 ]
2024   [ 18347.60576669 -18340.48805753  14736.8787955 ]
2025   [ 18347.60576669 -18340.48805753  14736.8787955 ]
```