Hello this is @Ranjeet_Kumbhar, Enjoy the Notebook

GitHub: https://github.com/RanjeetKumbhar01/TE_IT_ML_ASSIGNMENTS_SPPU Assignment on Regression technique Download temperature data from below link. https://www.kaggle.com/venky73/temperaturesof-india?select=temperatures.csv This data consists of temperatures of INDIA averaging the temperatures of all places month wise. Temperatures values are recorded in CELSIUS

a. Apply Linear Regression using suitable library function and predict the Month-wise temperature.

b. Assessthe performance of regression models using MSE, MAE and R-Square metrics

c. Visualize simple regression model.

```
import numpy as np # linear algebra
import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)
import seaborn as sns

from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
import matplotlib.pyplot as plt
# import matplotlib.pyplot as plt

data = pd.read_csv("../input/temperatures-of-india/temperatures.csv")
df = data

data.describe()
```

```
              YEAR         JAN         FEB         MAR         APR  \
count   117.000000  117.000000  117.000000  117.000000  117.000000
mean   1959.000000   23.687436   25.597863   29.085983   31.975812
std      33.919021    0.834588    1.150757    1.068451    0.889478
min    1901.000000   22.000000   22.830000   26.680000   30.010000
25%    1930.000000   23.100000   24.780000   28.370000   31.460000
50%    1959.000000   23.680000   25.480000   29.040000   31.950000
75%    1988.000000   24.180000   26.310000   29.610000   32.420000
max    2017.000000   26.940000   29.720000   32.620000   35.380000

              MAY         JUN         JUL         AUG         SEP
OCT  \
count  117.000000  117.000000  117.000000  117.000000  117.000000
117.000000
mean    33.565299   32.774274   31.035897   30.507692   30.486752
29.766581
std      0.724905    0.633132    0.468818    0.476312    0.544295
0.705492
min     31.930000   31.100000   29.760000   29.310000   29.070000
27.900000
```

```
25%     33.110000   32.340000   30.740000   30.180000   30.120000
29.380000
50%     33.510000   32.730000   31.000000   30.540000   30.520000
29.780000
75%     34.030000   33.180000   31.330000   30.760000   30.810000
30.170000
max     35.840000   34.480000   32.760000   31.840000   32.220000
32.290000

               NOV         DEC      ANNUAL     JAN-FEB     MAR-MAY
JUN-SEP  \
count  117.000000  117.000000  117.000000  117.000000  117.000000
117.000000
mean    27.285470   24.608291   29.181368   24.629573   31.517607
31.198205
std      0.714518    0.782644    0.555555    0.911239    0.740585
0.420508
min     25.700000   23.020000   28.110000   22.250000   29.920000
30.240000
25%     26.790000   24.040000   28.760000   24.110000   31.040000
30.920000
50%     27.300000   24.660000   29.090000   24.530000   31.470000
31.190000
75%     27.720000   25.110000   29.470000   25.150000   31.890000
31.400000
max     30.110000   28.010000   31.630000   28.330000   34.570000
32.410000

          OCT-DEC
count  117.000000
mean    27.208120
std      0.672003
min     25.740000
25%     26.700000
50%     27.210000
75%     27.610000
max     30.030000

data.head()

   YEAR    JAN    FEB    MAR    APR    MAY    JUN    JUL    AUG    SEP
OCT  \
0  1901  22.40  24.14  29.07  31.91  33.41  33.18  31.21  30.39  30.47
29.97
1  1902  24.93  26.58  29.77  31.78  33.73  32.91  30.92  30.73  29.80
29.12
2  1903  23.44  25.03  27.83  31.39  32.91  33.00  31.34  29.98  29.85
29.04
3  1904  22.50  24.73  28.21  32.02  32.64  32.07  30.36  30.09  30.04
29.20
```

```
4  1905  22.00  22.83  26.68  30.01  33.32  33.25  31.44  30.68  30.12
30.67
```

```
     NOV    DEC  ANNUAL  JAN-FEB  MAR-MAY  JUN-SEP  OCT-DEC
0  27.31  24.49   28.96    23.27    31.46    31.27    27.25
1  26.31  24.04   29.22    25.75    31.76    31.09    26.49
2  26.08  23.65   28.47    24.24    30.71    30.92    26.26
3  26.36  23.63   28.49    23.62    30.95    30.66    26.40
4  27.52  23.82   28.30    22.25    30.00    31.33    26.57
```

```
data.tail()
```

```
     YEAR    JAN    FEB    MAR    APR    MAY    JUN    JUL    AUG
SEP  \
112  2013  24.56  26.59  30.62  32.66  34.46  32.44  31.07  30.76
31.04
113  2014  23.83  25.97  28.95  32.74  33.77  34.15  31.85  31.32
30.68
114  2015  24.58  26.89  29.07  31.87  34.09  32.48  31.88  31.52
31.55
115  2016  26.94  29.72  32.62  35.38  35.72  34.03  31.64  31.79
31.66
116  2017  26.45  29.46  31.60  34.95  35.84  33.82  31.88  31.72
32.22
```

```
       OCT    NOV    DEC  ANNUAL  JAN-FEB  MAR-MAY  JUN-SEP  OCT-DEC
112  30.27  27.83  25.37   29.81    25.58    32.58    31.33    27.83
113  30.29  28.05  25.08   29.72    24.90    31.82    32.00    27.81
114  31.04  28.10  25.67   29.90    25.74    31.68    31.87    28.27
115  31.98  30.11  28.01   31.63    28.33    34.57    32.28    30.03
116  32.29  29.60  27.18   31.42    27.95    34.13    32.41    29.69
```

```
type(data)
```

```
pandas.core.frame.DataFrame
```

```
data.shape
```

```
(117, 18)
```

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 117 entries, 0 to 116
Data columns (total 18 columns):
 #   Column   Non-Null Count  Dtype
---  ------   --------------  -----
 0   YEAR     117 non-null    int64
 1   JAN      117 non-null    float64
 2   FEB      117 non-null    float64
 3   MAR      117 non-null    float64
```

```
 4   APR       117 non-null    float64
 5   MAY       117 non-null    float64
 6   JUN       117 non-null    float64
 7   JUL       117 non-null    float64
 8   AUG       117 non-null    float64
 9   SEP       117 non-null    float64
 10  OCT       117 non-null    float64
 11  NOV       117 non-null    float64
 12  DEC       117 non-null    float64
 13  ANNUAL    117 non-null    float64
 14  JAN-FEB   117 non-null    float64
 15  MAR-MAY   117 non-null    float64
 16  JUN-SEP   117 non-null    float64
 17  OCT-DEC   117 non-null    float64
dtypes: float64(17), int64(1)
memory usage: 16.6 KB
```

```
count = (data["JAN"]==22).sum()
print(count)
```

```
1
```

```
column = data
count = column[column == 0].count()
print(count)
```

```
YEAR       0
JAN        0
FEB        0
MAR        0
APR        0
MAY        0
JUN        0
JUL        0
AUG        0
SEP        0
OCT        0
NOV        0
DEC        0
ANNUAL     0
JAN-FEB    0
MAR-MAY    0
JUN-SEP    0
OCT-DEC    0
dtype: int64
```

```
data.isnull().sum()
```

```
YEAR       0
JAN        0
FEB        0
```

```
MAR         0
APR         0
MAY         0
JUN         0
JUL         0
AUG         0
SEP         0
OCT         0
NOV         0
DEC         0
ANNUAL      0
JAN-FEB     0
MAR-MAY     0
JUN-SEP     0
OCT-DEC     0
dtype: int64
```

```
data.isnull().head()
```

```
    YEAR    JAN    FEB    MAR    APR    MAY    JUN    JUL    AUG
SEP  \
0  False  False  False  False  False  False  False  False  False
False
1  False  False  False  False  False  False  False  False  False
False
2  False  False  False  False  False  False  False  False  False
False
3  False  False  False  False  False  False  False  False  False
False
4  False  False  False  False  False  False  False  False  False
False

     OCT    NOV    DEC  ANNUAL  JAN-FEB  MAR-MAY  JUN-SEP  OCT-DEC
0  False  False  False   False    False    False    False    False
1  False  False  False   False    False    False    False    False
2  False  False  False   False    False    False    False    False
3  False  False  False   False    False    False    False    False
4  False  False  False   False    False    False    False    False
```

```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 117 entries, 0 to 116
Data columns (total 18 columns):
 #   Column   Non-Null Count  Dtype
---  ------   --------------  -----
 0   YEAR     117 non-null    int64
 1   JAN      117 non-null    float64
 2   FEB      117 non-null    float64
 3   MAR      117 non-null    float64
```

```
 4   APR      117 non-null    float64
 5   MAY      117 non-null    float64
 6   JUN      117 non-null    float64
 7   JUL      117 non-null    float64
 8   AUG      117 non-null    float64
 9   SEP      117 non-null    float64
 10  OCT      117 non-null    float64
 11  NOV      117 non-null    float64
 12  DEC      117 non-null    float64
 13  ANNUAL   117 non-null    float64
 14  JAN-FEB  117 non-null    float64
 15  MAR-MAY  117 non-null    float64
 16  JUN-SEP  117 non-null    float64
 17  OCT-DEC  117 non-null    float64
dtypes: float64(17), int64(1)
memory usage: 16.6 KB
```

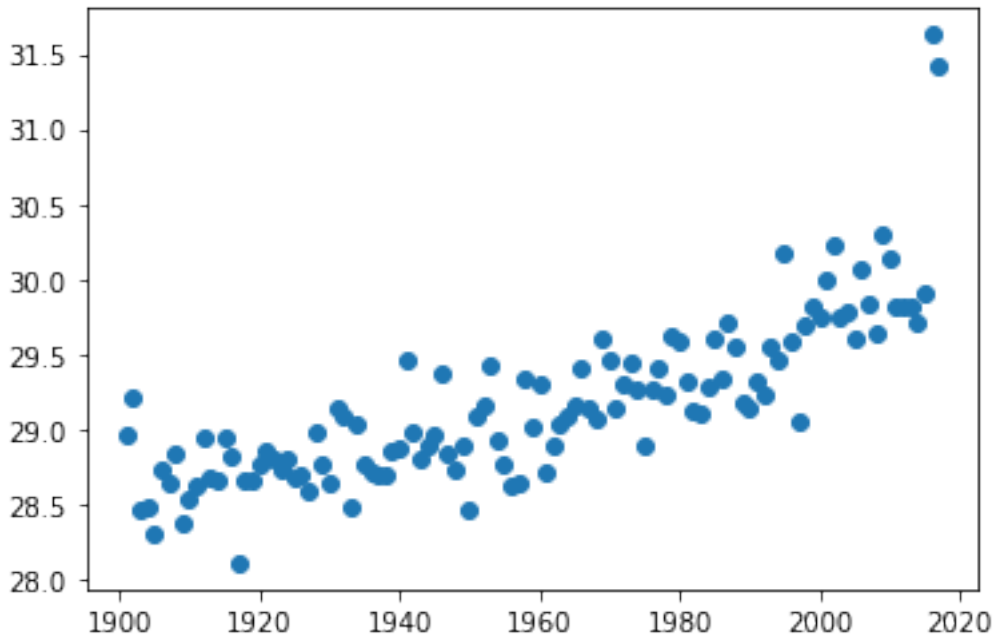
```
data.head()
```

| | YEAR | JAN | FEB | MAR | APR | MAY | JUN | JUL | AUG | SEP | OCT | \ |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 0 | 1901 | 22.40 | 24.14 | 29.07 | 31.91 | 33.41 | 33.18 | 31.21 | 30.39 | 30.47 | 29.97 | |
| 1 | 1902 | 24.93 | 26.58 | 29.77 | 31.78 | 33.73 | 32.91 | 30.92 | 30.73 | 29.80 | 29.12 | |
| 2 | 1903 | 23.44 | 25.03 | 27.83 | 31.39 | 32.91 | 33.00 | 31.34 | 29.98 | 29.85 | 29.04 | |
| 3 | 1904 | 22.50 | 24.73 | 28.21 | 32.02 | 32.64 | 32.07 | 30.36 | 30.09 | 30.04 | 29.20 | |
| 4 | 1905 | 22.00 | 22.83 | 26.68 | 30.01 | 33.32 | 33.25 | 31.44 | 30.68 | 30.12 | 30.67 | |

| | NOV | DEC | ANNUAL | JAN-FEB | MAR-MAY | JUN-SEP | OCT-DEC |
|---|---|---|---|---|---|---|---|
| 0 | 27.31 | 24.49 | 28.96 | 23.27 | 31.46 | 31.27 | 27.25 |
| 1 | 26.31 | 24.04 | 29.22 | 25.75 | 31.76 | 31.09 | 26.49 |
| 2 | 26.08 | 23.65 | 28.47 | 24.24 | 30.71 | 30.92 | 26.26 |
| 3 | 26.36 | 23.63 | 28.49 | 23.62 | 30.95 | 30.66 | 26.40 |
| 4 | 27.52 | 23.82 | 28.30 | 22.25 | 30.00 | 31.33 | 26.57 |

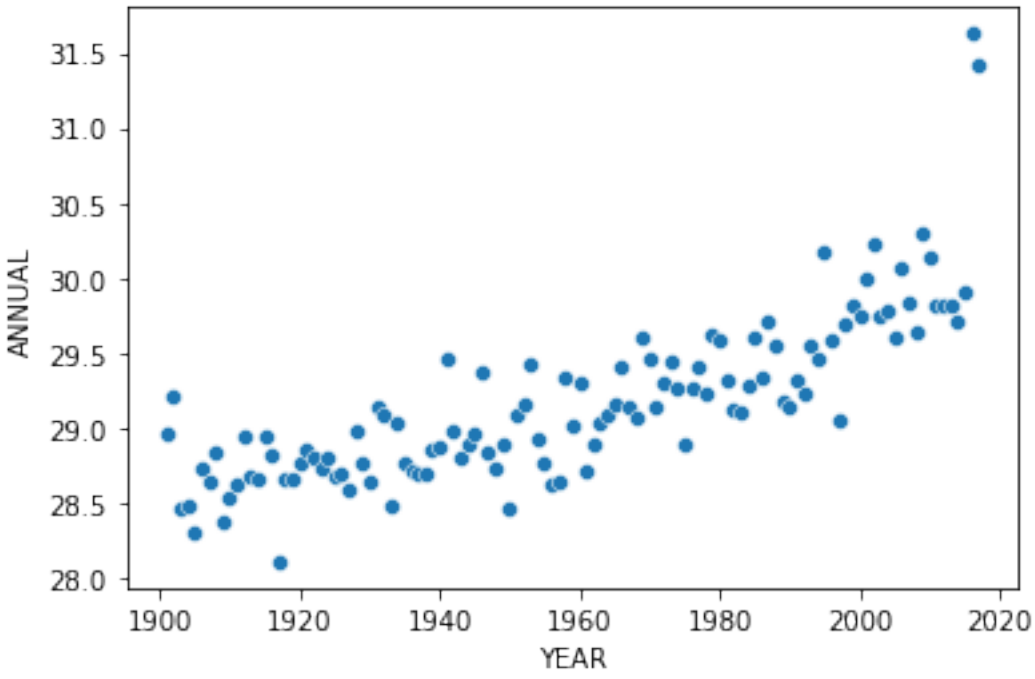
```
# x = data.iloc[:,1:6]
# y = data.iloc[:,-1:]
x = data["YEAR"]
y = data["ANNUAL"]

plt.plot(x,y,'o')
```

```
[<matplotlib.lines.Line2D at 0x7fe03a5342d0>]
```




```
sns.scatterplot(x=x,y=y,data=df)
```

```
<AxesSubplot:xlabel='YEAR', ylabel='ANNUAL'>
```




```
type(x)
```

```
pandas.core.series.Series
```

```
x.shape
```

```
(117,)
```

```
x = x.values
```

```
x = x.reshape(117,1)
```

```
x.shape
```

```
(117, 1)
```

```
type(x)
```

```
numpy.ndarray
```

```
x_train, x_test, y_train, y_test = train_test_split(x,
y,test_size=0.25)
```

```
print(f"x Training dataset: {x_train.shape}")
print(f"y Training dataset: {y_train.shape}")
print(f"x test dataset: {x_test.shape}")
print(f"y test dataset: {y_test.shape}")
```

```
x Training dataset: (87, 1)
y Training dataset: (87,)
x test dataset: (30, 1)
y test dataset: (30,)
```

```
model = LinearRegression()
```

```
model.fit(x_train,y_train)
```

```
LinearRegression()
```

# y = w.x + b

```
model.coef_ #w
```

```
array([0.01181101])
```

```
model.intercept_ #b
```

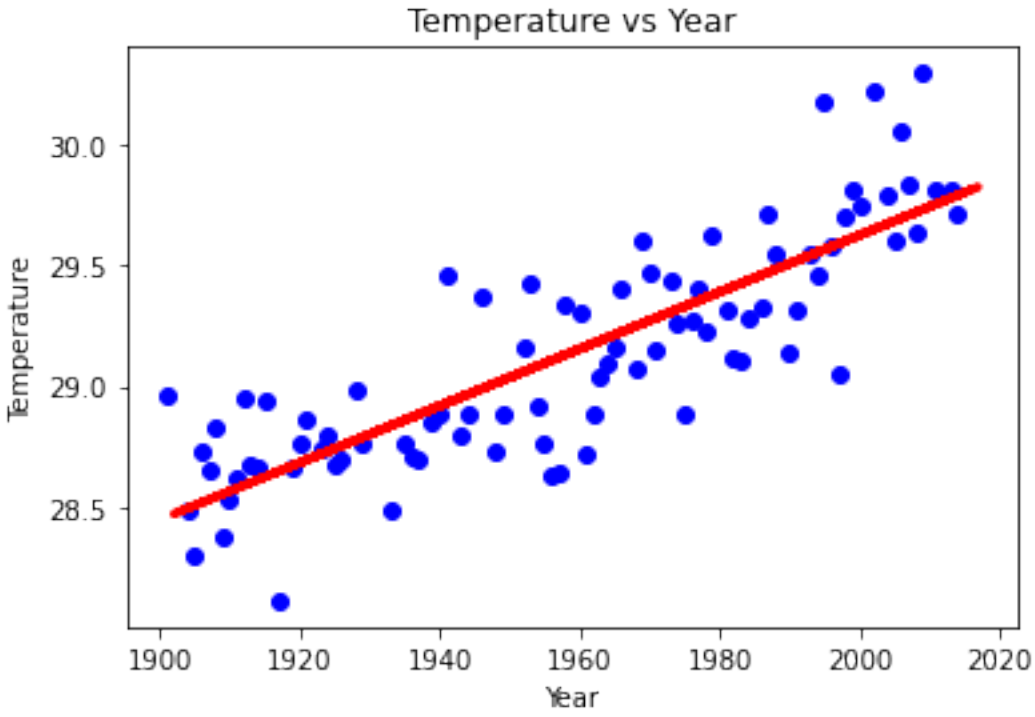
```
6.004942319521042
```

```
y_pred = model.predict(x_test)
```

```
y_pred.shape
```

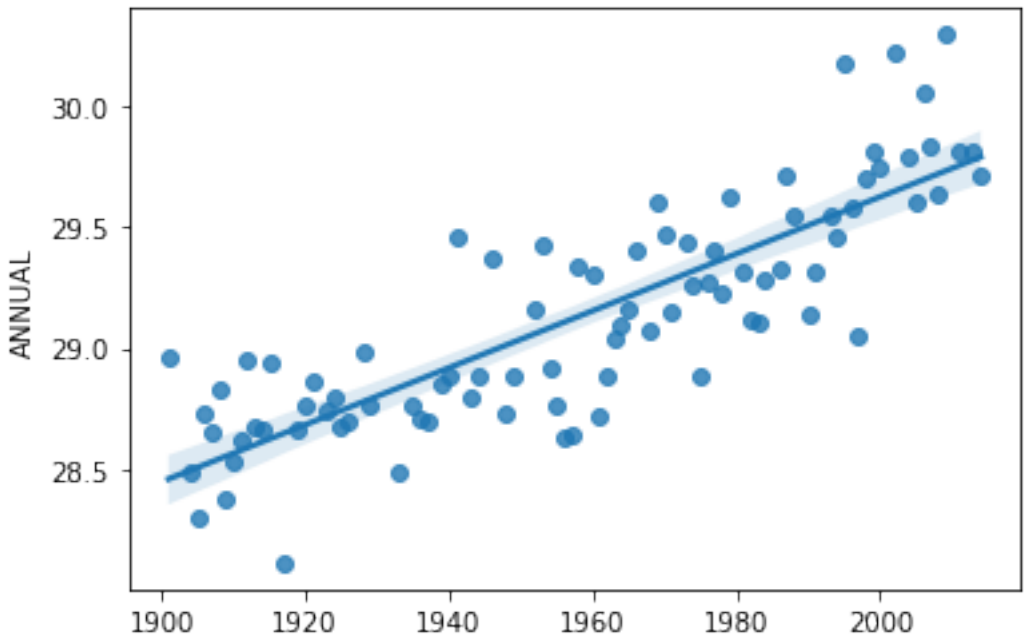
```
(30,)
```

```
plt.scatter(x_train, y_train, color='blue')
plt.plot(x_test, y_pred, color='red', linewidth=3)
plt.title("Temperature vs Year")
plt.xlabel("Year")
plt.ylabel("Temperature")
plt.show()
```




```
sns.regplot(data=df,x=x_train,y=y_train,)
```

```
<AxesSubplot:ylabel='ANNUAL'>
```




```
from sklearn.metrics import
mean_absolute_error,mean_squared_error,r2_score
print(f"MSE:  {mean_squared_error(y_test,y_pred)}")
print(f"MAE:  {mean_absolute_error(y_test,y_pred)}")
print(f"R-Sqaure :  {r2_score(y_test,y_pred)}")
```

```
MSE:  0.25534763392126797
MAE:  0.2911243760198463
R-Sqaure :  0.5321685895690408
```