ayebale peter bcs linear regression on wednesday

March 19, 2024

[2]:
```python
#these are my libraries
import numpy as np
import pandas as pd
```

[3]:
```python
#importing my data set from my pc
dset=pd.read_csv('C:\\Users\\hj\\Desktop\\testing.csv')
dset
```

[3]:
```
     slotes  type  gender  years  botters  index  distance
0       713     3  female  47.00        8      2   69.5500
1       714     3  female   6.00        4      2   31.2750
2       715     3    male  14.00        5      2   46.9000
3       716     3  female  45.00        0      1   14.4542
4       717     3    male  25.00        1      0   17.8000
..      …     …       …      …        …      …
177     890     3  female   5.00        4      2   31.3875
178     891     2  female   4.00        1      1   23.0000
179     892     3    male   0.83        0      1    9.3500
180     893     1    male  30.00        1      2  151.5500
181     894     3    male   5.00        4      2   31.3875

[182 rows x 7 columns]
```

[4]:
```python
#am calling the arrays i want
x=np.array(dset["years"]).reshape(-1,1)
y=np.array(dset["distance"])
```

[5]:
```python
#let me check whether some data is missing
dset.isna().sum()
```

[5]:
```
slotes      0
type        0
gender      0
years       0
botters     0
index       0
distance    0
dtype: int64
```

[6]:
```
x
```

[6]:
```
array([[47.  ],
       [ 6.  ],
       [14.  ],
       [45.  ],
       [25.  ],
       [43.  ],
       [45.  ],
       [48.  ],
       [19.  ],
       [15.  ],
       [32.  ],
       [24.  ],
       [30.  ],
       [34.  ],
       [36.5 ],
       [67.  ],
       [47.  ],
       [45.  ],
       [43.  ],
       [34.  ],
       [22.  ],
       [41.  ],
       [22.  ],
       [23.  ],
       [14.  ],
       [30.  ],
       [52.  ],
       [45.  ],
       [26.  ],
       [ 2.  ],
       [20.  ],
       [23.  ],
       [18.  ],
       [18.  ],
       [43.  ],
       [17.  ],
       [18.  ],
       [22.  ],
       [46.  ],
       [24.  ],
       [22.  ],
       [ 2.  ],
       [13.  ],
       [ 9.  ],
       [25.  ],
```

```
[34.  ],
[54.  ],
[32.5 ],
[35.  ],
[33.  ],
[28.  ],
[21.  ],
[23.  ],
[22.  ],
[24.  ],
[60.  ],
[28.  ],
[24.  ],
[39.  ],
[36.  ],
[30.  ],
[23.  ],
[18.  ],
[17.  ],
[18.  ],
[38.  ],
[23.5 ],
[28.  ],
[30.  ],
[70.5 ],
[64.  ],
[75.  ],
[11.  ],
[33.  ],
[42.  ],
[28.5 ],
[18.  ],
[28.5 ],
[33.  ],
[44.  ],
[45.5 ],
[32.  ],
[31.  ],
[23.  ],
[47.  ],
[32.  ],
[24.  ],
[21.  ],
[20.  ],
[ 8.  ],
[30.  ],
[34.  ],
```

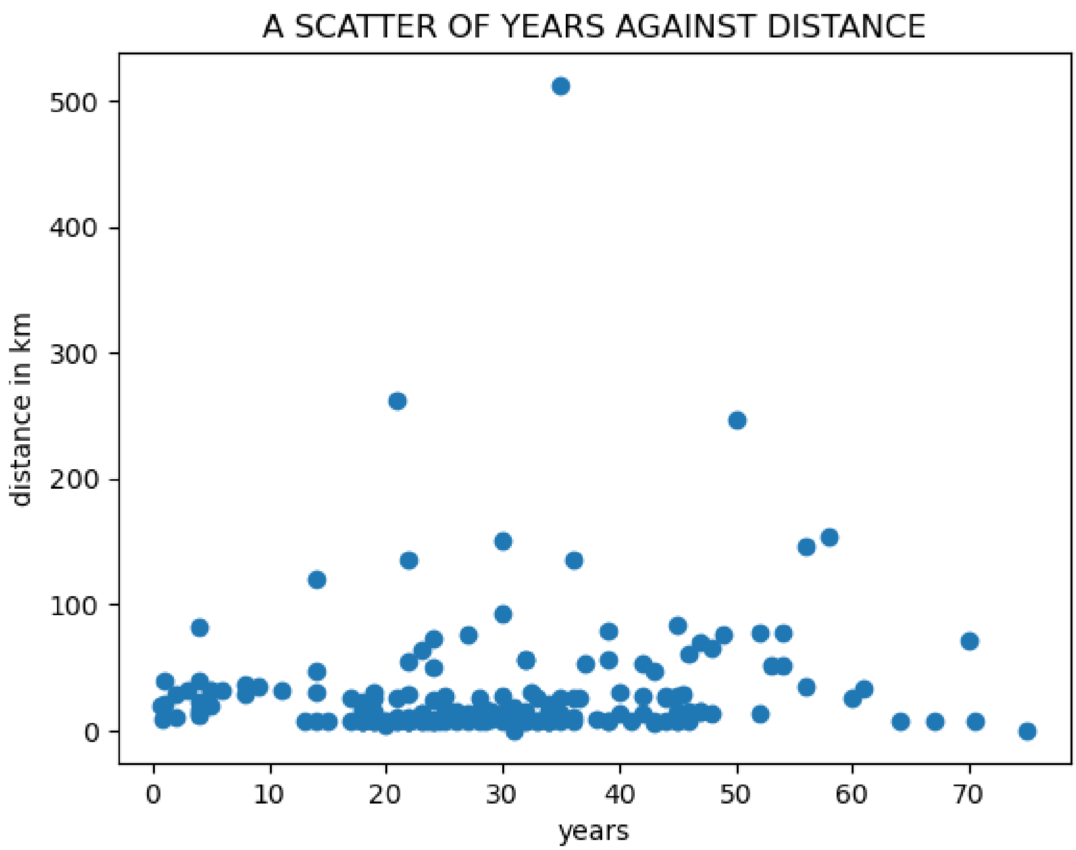
```
[29.  ],
[19.  ],
[39.  ],
[22.  ],
[18.  ],
[45.  ],
[42.  ],
[24.5 ],
[46.  ],
[61.  ],
[22.  ],
[31.  ],
[28.  ],
[70.  ],
[19.  ],
[36.  ],
[35.  ],
[34.  ],
[35.  ],
[29.  ],
[18.  ],
[56.  ],
[46.  ],
[37.  ],
[40.  ],
[ 5.  ],
[23.  ],
[36.  ],
[44.  ],
[21.  ],
[19.  ],
[20.  ],
[42.  ],
[49.  ],
[31.  ],
[ 1.  ],
[14.  ],
[19.  ],
[48.  ],
[24.  ],
[34.  ],
[56.  ],
[32.  ],
[ 4.  ],
[29.  ],
[14.  ],
[ 4.  ],
```

```
       [22.  ],
       [53.  ],
       [28.  ],
       [27.  ],
       [35.  ],
       [ 8.  ],
       [25.  ],
       [54.  ],
       [24.  ],
       [ 1.  ],
       [19.  ],
       [22.  ],
       [22.  ],
       [30.  ],
       [52.  ],
       [19.  ],
       [39.  ],
       [27.  ],
       [ 0.75],
       [40.  ],
       [ 3.  ],
       [32.  ],
       [27.  ],
       [21.  ],
       [ 4.  ],
       [36.  ],
       [58.  ],
       [ 4.  ],
       [35.  ],
       [34.  ],
       [23.  ],
       [44.  ],
       [50.  ],
       [39.  ],
       [21.  ],
       [18.  ],
       [26.  ],
       [ 4.  ],
       [ 5.  ],
       [ 4.  ],
       [ 0.83],
       [30.  ],
       [ 5.  ]])
```

[7]:
```
y
```

```
[7]: array([ 69.55  ,  31.275 ,  46.9   ,  14.4542,  17.8   ,  46.9   ,
          83.475 ,  13.    ,  14.5   ,   7.2292,   7.925 ,   8.05  ,
           8.6625,   8.05  ,  26.    ,   7.3125,  15.    ,   6.975 ,
           7.7292,  14.4   ,   7.8958,   7.125 ,   7.25  ,  10.5   ,
           7.8542,  13.    ,  13.    ,   7.225 ,  14.4542,  10.4625,
           7.925 ,   6.95  ,  14.4542,   7.7958,   6.45  ,   7.0542,
           7.75  ,   7.2292,  61.175 ,  24.15  ,   9.    ,  29.125 ,
           7.8958,  34.375 ,  27.7208,   7.725 ,  51.8625,  30.0708,
          10.5   ,   7.8958,   7.8542,  25.4667,  13.    ,  10.5167,
          73.5   ,  26.55  ,  14.4   ,   9.5   ,   7.225 ,   7.8958,
          27.75  ,   7.8958,  11.5   ,  25.4667,   7.775 ,   8.6625,
           7.2292,  13.    ,   8.6625,   7.75  ,   7.8958,   0.    ,
          31.275 ,   7.775 ,  13.    ,  16.1   ,  11.5   ,   7.2292,
          25.925 ,  26.    ,  28.5   ,   7.75  ,   0.    ,   9.225 ,
          14.5   ,  15.85  ,   7.6292,   7.8542,   4.0125,  29.125 ,
           7.8958,  21.    ,  10.5   ,   7.65  ,   7.925 ,   7.25  ,
          17.8   ,  27.9   ,  27.    ,   8.05  ,  15.2458,  33.5   ,
         135.6333,  18.    ,  10.5   ,  71.    ,   7.8958,  10.5   ,
           8.05  ,   7.8958,  13.    ,   9.5   ,   8.05  , 146.5208,
           7.8792,  52.5542,  13.    ,  19.2583,  63.3583,  26.    ,
           7.925 , 262.375 ,  26.    ,   7.925 ,  52.5542,  76.7292,
          13.    ,  39.    ,  30.0708,  10.5   ,  65.    ,  49.5042,
           7.75  ,  35.5   ,  56.4958,  39.    ,  15.2458, 120.    ,
          16.7   ,  29.    ,  51.4792,  26.55  ,  76.7292,  26.2875,
          36.75  ,   7.775 ,  78.2667,   7.8292,  20.575 ,   7.8792,
           7.25  ,  55.    ,  93.5   ,  78.2667,  30.    ,  55.9   ,
           7.7958,  19.2583,  31.    ,  31.3875,  56.4958,  13.8583,
          10.5   ,  12.35  , 135.6333, 153.4625,  13.4167, 512.3292,
          15.5   ,  13.7917,  27.7208, 247.5208,  79.65  ,   7.65  ,
          23.    ,   7.8542,  81.8583,  31.3875,  23.    ,   9.35  ,
         151.55  ,  31.3875])
```

```
[8]: #let me try to draw a graph
import matplotlib.pyplot as plt
plt.scatter(x,y)
plt.xlabel("years")
plt.ylabel("distance in km")
plt.title("A SCATTER OF YEARS AGAINST DISTANCE")
plt.show()
```



[9]:
```python
#data splitting
from sklearn.model_selection import train_test_split
```

[10]:
```python
x_train,x_test,y_train,y_test=train_test_split(x,y,test_size=0.
 ↪2,random_state=39)
```

[11]:
```python
#creating a linear regression model plus fitting it
from sklearn.linear_model import LinearRegression
model=LinearRegression().fit(x_train,y_train)
model
```

[11]:
```
LinearRegression()
```

[12]:
```python
#lets make prediction
ypredi=model.predict(x_test)
ypredi
```

[12]:
```
array([42.48374868, 25.57297196, 42.89620665, 29.28509368, 28.87263571,
       34.64704727, 32.17229946, 35.47196321, 35.88442118, 26.81034587,
       24.74805602, 21.03593431, 27.22280384, 31.75984149, 30.11000962,
```

```
       18.14872852, 30.93492555, 30.52246759, 26.81034587, 19.38610243,
       35.47196321, 25.98542993, 31.75984149, 32.17229946, 36.29687915,
       36.29687915, 28.04771977, 27.22280384, 30.93492555, 36.70933712,
       32.17229946, 27.6352618 , 32.58475743, 26.81034587, 30.52246759,
       18.56118649, 30.11000962])
```

[13]:
```
#lets get the coefficients
model.coef_
```

[13]: array([0.41245797])

[14]:
```
#lets try the intercept
model.intercept_
```

[14]: 17.73627055567092

[15]:
```
#lets try the accuracy of the model on the test values
model.score(x_train,y_train)
```

[15]: 0.024965104682953987

[16]:
```
#now lets get the mse,mae and rsquared
from sklearn.metrics import mean_absolute_error,r2_score,mean_squared_error
```

[17]:
```
mae=mean_absolute_error(y_test,ypredi)
mae
```

[17]: 31.889356499778604

[18]:
```
mse=mean_squared_error(y_test,ypredi)
mse
```

[18]: 6751.032110787296

[19]:
```
r2=r2_score(y_test,ypredi)
r2
```

[19]: 0.001168980981393064

[20]:
```
#now its time for me to optimize my model
from sklearn.model_selection import GridSearchCV
model=LinearRegression()
model
```

[20]: LinearRegression()

[21]:
```
#let me define some of the parameters
param_grid={
```

```
'n_jobs':[True,False],
'copy_X':[True,False],
'fit_intercept':[True,False],
'positive':[True,False],}
param_grid
```

[21]:

```
{'n_jobs': [True, False],
 'copy_X': [True, False],
 'fit_intercept': [True, False],
 'positive': [True, False]}
```

[22]:

```
#its time to perform a gridsearchcv
gridsearch=GridSearchCV(model,param_grid,cv=5)
gridsearch
```

[22]:

```
GridSearchCV(cv=5, estimator=LinearRegression(),
             param_grid={'copy_X': [True, False],
                         'fit_intercept': [True, False],
                         'n_jobs': [True, False], 'positive': [True, False]})
```

[23]:

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

[23]:

```
GridSearchCV(cv=5, estimator=LinearRegression(),
             param_grid={'copy_X': [True, False],
                         'fit_intercept': [True, False],
                         'n_jobs': [True, False], 'positive': [True, False]})
```

[24]:

```
#getting the best parameters found in the gridsearch
best_params=gridsearch.best_params_
```

[25]:

```
best_params
```

[25]:

```
{'copy_X': True, 'fit_intercept': True, 'n_jobs': True, 'positive': False}
```

[26]:

```
#let me train my linear regression model with the best parameters
best_model=LinearRegression(**best_params).fit(x_train,y_train)
best_model
```

[26]:

```
LinearRegression(n_jobs=True)
```

[27]:

```
#lets make predictions
ypre=best_model.predict(x_test)
ypre
```

[27]:

```
array([42.48374868, 25.57297196, 42.89620665, 29.28509368, 28.87263571,
       34.64704727, 32.17229946, 35.47196321, 35.88442118, 26.81034587,
```

```
24.74805602, 21.03593431, 27.22280384, 31.75984149, 30.11000962,
18.14872852, 30.93492555, 30.52246759, 26.81034587, 19.38610243,
35.47196321, 25.98542993, 31.75984149, 32.17229946, 36.29687915,
36.29687915, 28.04771977, 27.22280384, 30.93492555, 36.70933712,
32.17229946, 27.6352618 , 32.58475743, 26.81034587, 30.52246759,
18.56118649, 30.11000962])
```

[28]:
```
#let finalise by evaluating the model
r2=r2_score(y_test,ypredi)
mae=mean_absolute_error(y_test,ypredi)
mae=mean_absolute_error(y_test,ypredi)
```

[29]:
```
r2
```

[29]:
```
0.001168980981393064
```

[30]:
```
mae
```

[30]:
```
31.889356499778604
```

[31]:
```
mse
```

[31]:
```
6751.032110787296
```

[ ]:
```

```

[ ]:
```

```