Overfitting to evaluate Linear Regression Model and Non-linear Regression

NAGORI_FATEMA_19635

TABLE OF CONTENT

INTRODUCTION

Regression Definition:

- A regression is a statistical analysis assessing the association between two variables. It is used to find the relationship between two variables.

- **Regression Formula: (another formla produces the same result)**

  Regression Equation(y) = a + bx

  Slope(b) = (NΣXY - (ΣX)(ΣY)) / (NΣX^2 - (ΣX)^2)

  Intercept(a) = (ΣY - b(ΣX)) / N

Where:

x and y are the variables.
b = The slope of the regression line
a = The intercept point of the regression line and the y axis.
N = Number of values or elements
X = First Score
Y = Second Score
ΣXY = Sum of the product of first and Second Scores
ΣX = Sum of First Scores
ΣY = Sum of Second Scores
ΣX^2 = Sum of square First Scores

DESIGN

Linear Regression using Least Square Method

- This formula is called The Normal Equation which is based on Linear Regression using Least Square Method.
- To find the Simple/Linear Regression of given data X and Y
- First find slope, intercept and use it to form regression equation.
  1. Step 1:
     Count the number of values. N = 10
  2. Step 2:
     Find X * Y, X^2

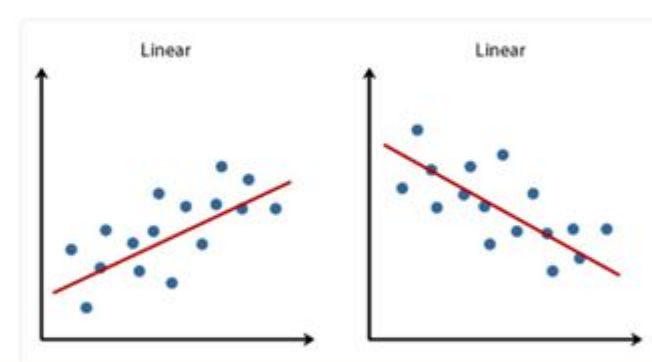
X Value	Y Value	X*Y	X*X
1	1.8	1*1.8=1.8	1*1=1
2	2.4	2*2.4=4.8	2*2=4
3.3	2.3	3.3*2.3=7.59	3.3*3.3=10.89
4.3	3.8	4.3*3.8=16.34	4.3*4.3=18.49
5.3	5.3	5.3*5.3=28.09	5.3*5.3=28.09
1.4	1.5	1.4*1.5=2.1	1.4*1.4=1.96
2.5	2.2	2.5*2.2=5.5	2.5*2.5=6.25
2.8	3.8	2.8*3.8=10.64	2.8*2.8=7.84
4.1	4.0	4.1*4.0=16.4	4.1*4.1=16.81
5.1	5.4	5.1*5.4=27.54	5.1*5.1=26.01

Linear Regression using Least Square Method

3 Step 3:

Find ΣX, ΣY, ΣXY, ΣX^2.

- $\Sigma X = 31.8$
- $\Sigma Y = 32.5$
- $\Sigma XY = 120.8$
- $\Sigma X^2 = 121.34$



Step 4:

Substitute in the above slope formula given.

Slope(b) = $(N\Sigma XY - (\Sigma X)(\Sigma Y)) / (N\Sigma X^2 - (\Sigma X)^2)$
$= ((10)*(120.8)-(31.8)*(32.5))/((10)*(121.34)-(31.8)^2)$
$= (1208 - 1033.5)/(1213.4 - 1011.24)$
$= 174.5/202.16$
$= 0.86$

Linear Regression using Least Square Method

Step 5:

Now, again substitute in the above intercept formula given.

Intercept(a) = (ΣY - b(ΣX)) / N

= (32.5 - 0.86(31.8))/10

= (32.5 – 27.34)/10

= 51.52/10

= 0.5152

Step 6:

Then substitute Intercept(a) and Slope(b) in regression equation formula

Linear Regression Equation(y) = a + bx

= 0.51 + 0.86x.

Linear Regression using Least Square Method

Step 7:

Suppose if we want to know the approximate y value for the variable $x = 64$. Then we can substitute the value in the above equation.

Regression Equation(y) $= a + bx$

$= -8.098 + 0.19(64)$.

$= -8.098 + 12.16$

$= 4.06$

Non-linear Regression:

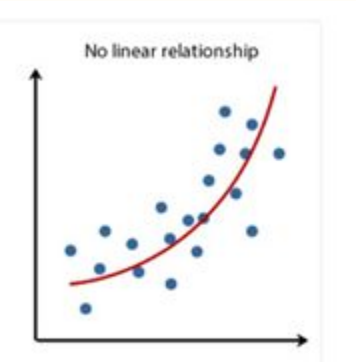
Regression Equation(y) = $a + b^2$

We can still use Linear Regression formula

Slope(b) = $(N\Sigma\underline{P}Y - (\Sigma\underline{P})(\Sigma Y)) / (N\Sigma\underline{P}^2 - (\Sigma\underline{P})^2)$

Intercept(a) = $(\Sigma Y - b(\Sigma\underline{P})) / N$

Where $\underline{P} = X * X$



Non-linear Regression:

To find the Simple Non-Linear Regression of given data X and Y
We can simply create $\underline{X}$ from X

To find regression equation, we will first find slope, intercept and use it to form regression equation.

- Step 1:

  Count the number of values. N = 10

- Step 2:

  Find $\underline{X} * Y$, $\underline{X}^2$

X Value	Y Value	$\underline{X}$=X*X
1	1.8	1*1=1
2	2.4	2*2=4
3.3	2.3	3.3*3.3=10.89
4.3	3.8	4.3*4.3=18.49
5.3	5.3	5.3*5.3=28.09
1.4	1.5	1.4*1.4=1.96
2.5	2.2	2.5*2.5=6.25
2.8	3.8	2.8*2.8=7.84
4.1	4.0	4.1*4.1=16.81
5.1	5.4	5.1*5.1=26.01

Non-linear Regression:

Step 3:

Find ΣX, ΣY, ΣXY, ΣX^2.

$\Sigma X = 121.34$

$\Sigma Y = 32.5$

$\Sigma XY = 509.5$

$\Sigma X^2 = 2329.92$

Step 4:

Substitute in the above slope formula given.

X Value	Y Value	X*Y	X*X
1	1.8	1*1.8=1.8	1*1=1
4	2.4	4*2.4=9.6	4*4=16
10.89	2.3	10.89*2.3=25.04	10.89*10.89=118.59
18.49	3.8	18.49*3.8=70.02	18.49*18.49=341.88
28.09	5.3	28.09*5.3=148.87	28.09*28.09=789
1.96	1.5	1.96*1.5=2.94	1.96*1.96=3.84
6.25	2.2	6.25*2.2=13.75	6.25*6.25=39.06
7.84	3.8	7.84*3.8=29.79	7.84*7.84=61.46
16.81	4.0	16.81*4.0=67.24	16.81*16.81=282.57
26.01	5.4	26.01*5.4=140.45	26.01*26.01=676.52

Slope(b) $= (N\Sigma XY - (\Sigma X)(\Sigma Y)) / (N\Sigma X^2 - (\Sigma X)^2)$

$= ((10)*(509.5)-(121.34)*(32.5))/((10)*(2329.92)-(121.34)^2)$

$= (5095 - 3943.55)/(23299.2 - 14723.39)$

$= 1151.45/8575.80$

$= 0.1346$

- Step 5:

Now, again substitute in the above intercept formula given.

```
Intercept(a) = (ΣY - b(ΣX)) / N
= (32.5 - 0.00021(121.34))/10
= (32.5 – 0.0254)/10
= 32.474/10
= 1.6168
```

Step 6:
Then substitute these values in regression equation formula
Regression Equation(y) = a + bx^2

= 1.6168 + 0.1346x^2

Step 7:
Suppose if we want to know the approximate y value for the variable x = 64. Then we can substitute the value in the above equation.
Regression Equation(y) = a + bx^2

= 1.6168 + 0.1346 (64 * 64)

=1.6168 + 551.32

= 552.93

Mean Square Error:

Mean Squared Error (MSE)

- The Mean Squared Error (MSE) is a measure of how close a fitted line is to data points.
  - The smaller the MSE, the closer the fit is to the data.
- If $\hat{Y}$ is a vector of n predictions, and Y is the vector of the true values, then the (estimated) MSE of the predictor is:

$$\mathrm{MSE} = \frac{1}{n}\sum_{i=1}^{n}(\hat{Y}_i - Y_i)^2.$$

Training Set						Validation Set					
Real Data		Model 1		Model 2		Real Data		Model 1		Model 2	
x	y	x	ŷ	x	ŷ	x	y	x	ŷ	x	ŷ
1	1.8	1	1.38	1	1.75	1	1.7	1	1.81	1	1.91
2	2.4	2	2.24	2	2.15	2	2.7	2	3.014	2	2.74
3.3	2.3	3.3	3.358	3.3	3.08	3.3	2.5	3.3	3.702	3.3	3.45
4.3	3.8	4.3	4.218	4.3	4.10	4.3	2.8	4.3	4.562	4.3	4.59
5.3	5.3	5.3	5.078	5.3	5.39	5.3	5.5	5.3	4.902	5.3	5.11

Training Set

- Model 1
  - MSE

    = [(1.38-1.8)^2 + (2.24-2.4)^2 + (3.358-2.3)^2 + (4.218-3.8)^2 + (5.078-5.3)^2 +(1.74-1.5)^2 +(2.67-2.2)^2 +(2.92-3.8)^2 + (4.04-4.0)^2 + (4.90-5.4)^2]/10

    = 0.28

- Model 2
  - MSE

    [(1.75-1.8)^2 + (2.15-2.4)^2 + (3.08-2.3)^2 + (4.10-3.8)^2 + (5.39-5.3)^2 + (1.88-1.5)^2 +(2.45-2.2)^2 +(2.67-3.8)^2 + (3.87-4.0)^2 + (5.11-5.4)^2]/10

    = 0.24

Validation Set

- Model 1
  - MSE

    = [(1.81-1.7)^2 + (3.014-2.7)^2 + (3.70-2.5)^2 + (4.56-2.8)^2 + (4.90-5.5)^2]/5

    = 1.0035

- Model 2
  - MSE

    = [(1.91-1.7)^2 + (2.74-2.7)^2 + (3.45-2.5)^2 + (4.59-2.8)^2 + (5.11-5.5)^2]/5

    = 0.86

```
max(Training_Set_MSE, Validation_Set_MSE) / min(Training_Set_MSE, Validation_Set_MSE)
```

- Compare Model 1 and Model 2
  - Mode1

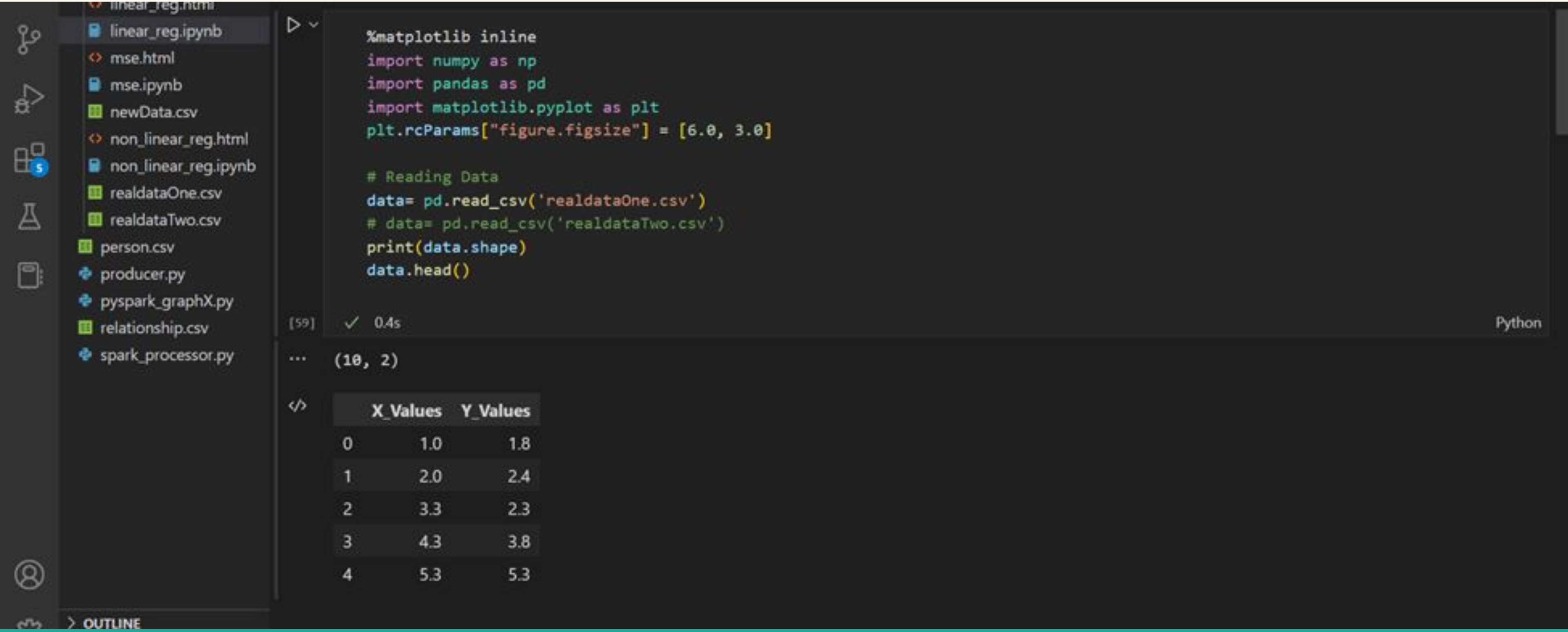
    1.003 / 0.297 = 3.38

  - Model 2

    0.860 / 0.247 = 3.48

- **Conclusion**
  - **Model 1 is a better model**

IMPLEMENTATION: Linear Regression



```python
#Collecting X and Y
X= data['X_Values'].values
Y= data['Y_Values'].values
```

[139] Python

```python
x_values=np.array(X)
print("Sum of all the X_Values: ", round(x_values.sum(),2))

y_values=np.array(Y)
print("Sum of all the Y_Values: ", y_values.sum())

xy = [x_values * y_values for x_values, y_values in zip(X, Y)]
xy_values=np.array(xy)
print("Sum of all the XY_Values: ", round(xy_values.sum(),2))

xx = [x_values * x_values for x_values, x_values in zip(X, X)]
xx_values=np.array(xx)
print("Sum of all the XX_Values: ", xx_values.sum())
```

[145] Python

```
Sum of all the X_Values:  31.8
Sum of all the Y_Values:  32.5
Sum of all the XY_Values:  120.8
Sum of all the XX_Values:  121.34
```

```python
# Using the formula to calculate slope(b)
n= len(X)
b = round(((n* xy_values.sum())-(x_values.sum()*y_values.sum()))/ ((n*xx_values.sum())-(x_values.sum()**2)),2)
print("slope(b) is: " , b)
```

[141] Python

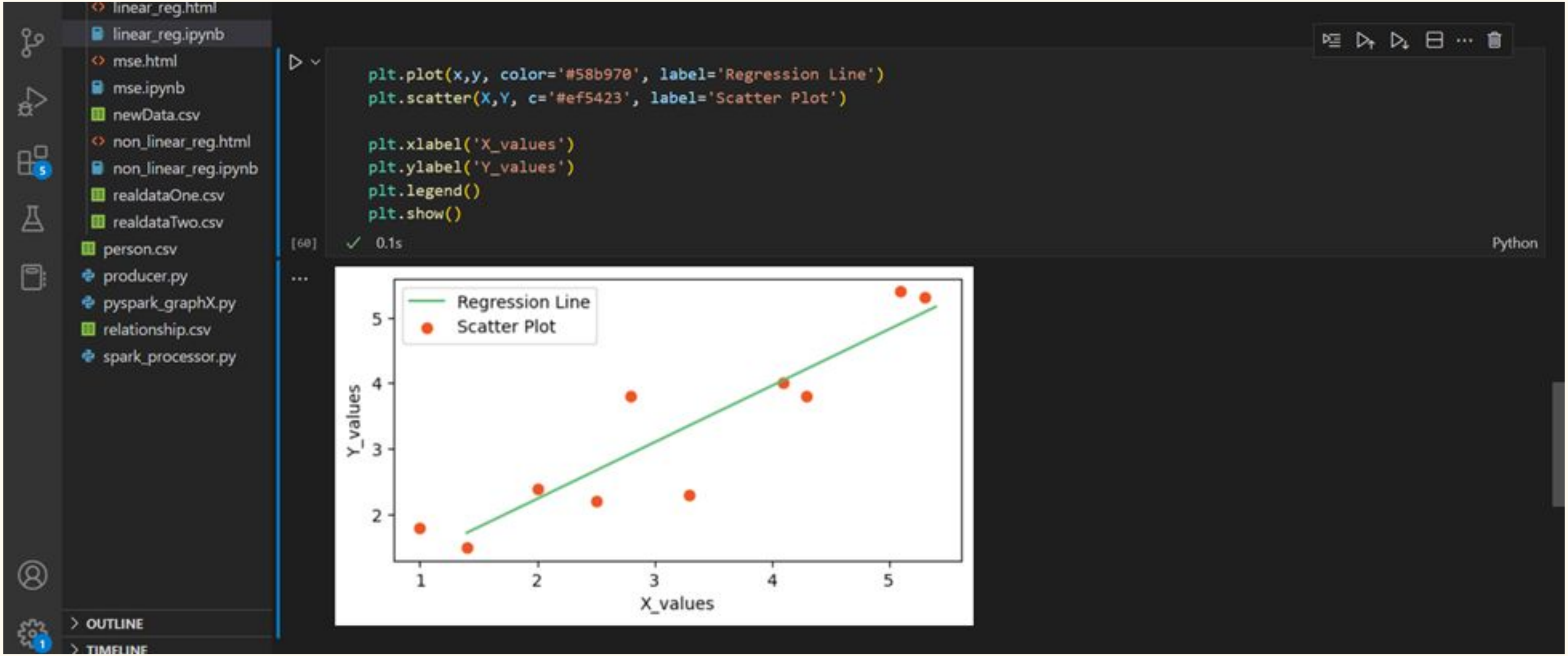
```
slope(b) is:  0.86
```

```python
# Using the formula to calculate intercept(a)
a = round((y_values.sum() - (b*x_values.sum()))/n ,2)
print("intercept(a): ", a)
```

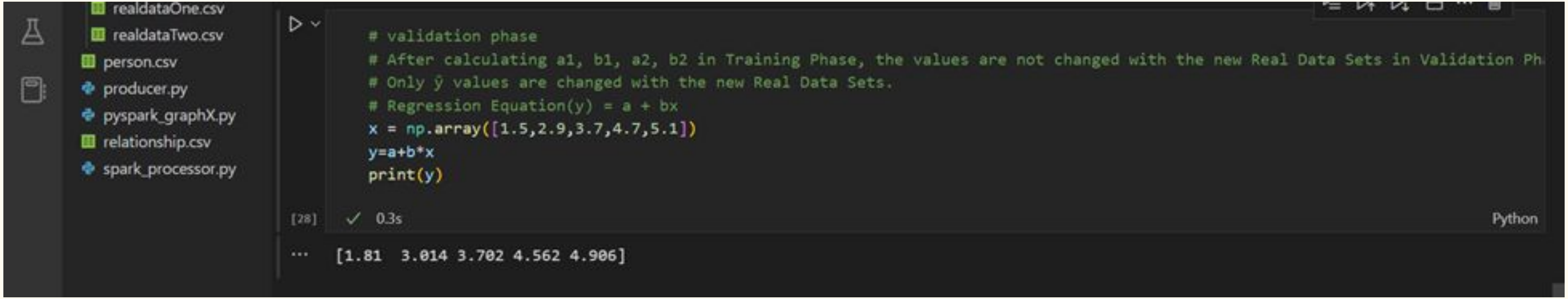
[146] Python

```
intercept(a):  0.52
```

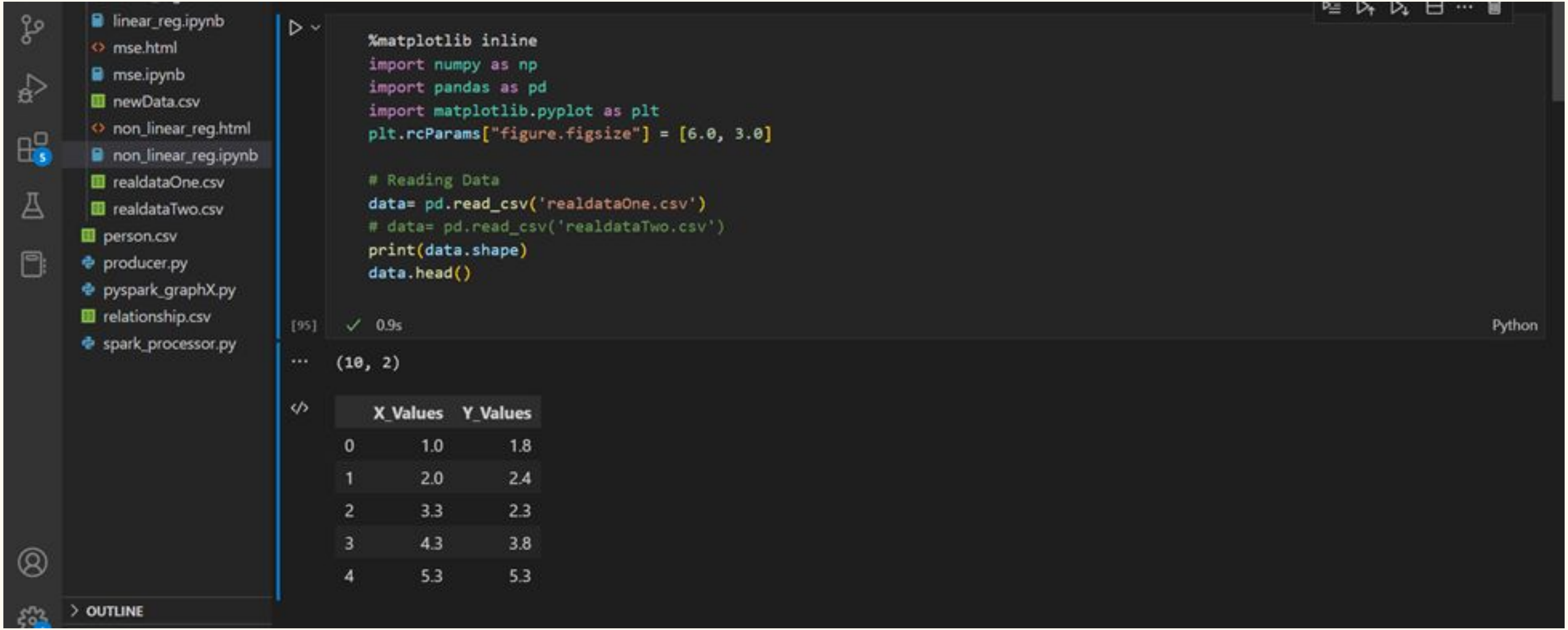
Linear Graph



Validation phase:



IMPLEMENTATION: NonLinear Regression



linear_reg.ipynb
newData.csv
non_linear_reg.ipynb
values.csv
person.csv
producer.py
pyspark_graphX.py
relationship.csv
spark_processor.py

```python
#Collecting X and Y
X= data['X_Values'].values
Y= data['Y_Values'].values
```

[65] ✓ 0.9s Python

```python
y_values=np.array(Y)
print("Sum of all the Y_Values: ", y_values.sum())

xx = [x_values * x_values for x_values, x_values in zip(X, X)]
xx_values=np.array(xx)
print("Sum of all the XX_Values: ", xx_values.sum())
```

[66] ✓ 0.5s Python

```
Sum of all the Y_Values:  32.5
Sum of all the XX_Values:  121.34
```

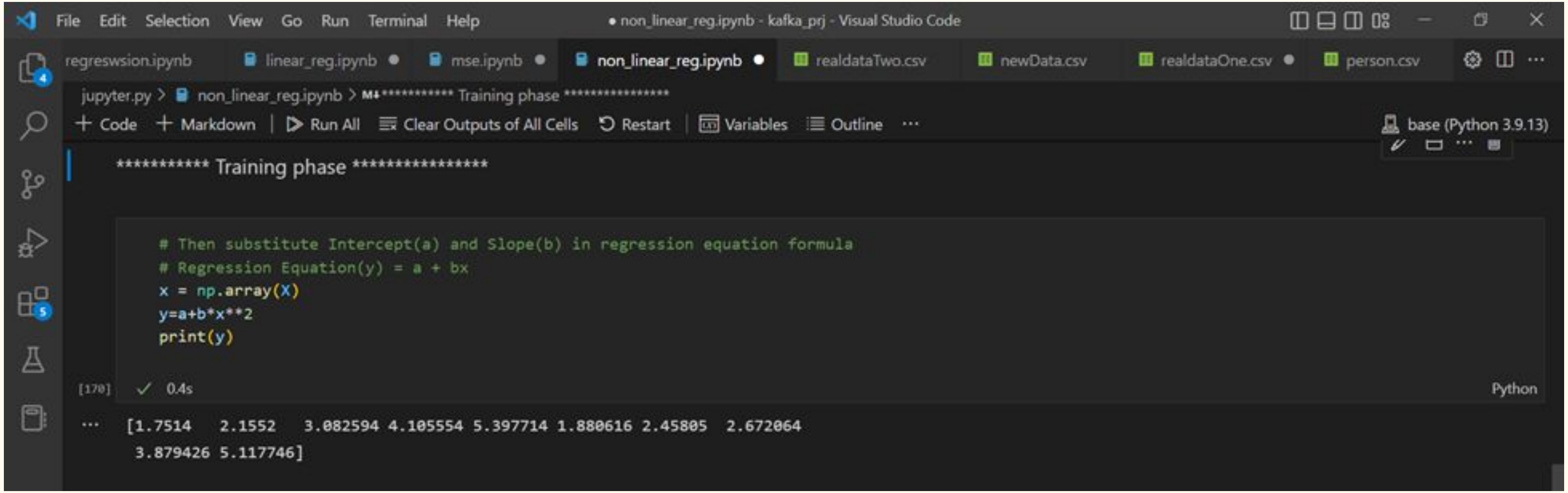
```python
xy = [xx_values * y_values for xx_values, y_values in zip(xx, Y)]
xy_values=np.array(xy)
print("Sum of all the XY_Values: ", round(xy_values.sum(),2))

xxx = [xx_values * xx_values for xx_values, xx_values in zip(xx, xx)]
xxx_values=np.array(xxx)
print("Sum of all the XXX_Values: ", xxx_values.sum())
```

[167] ✓ 0.4s Python

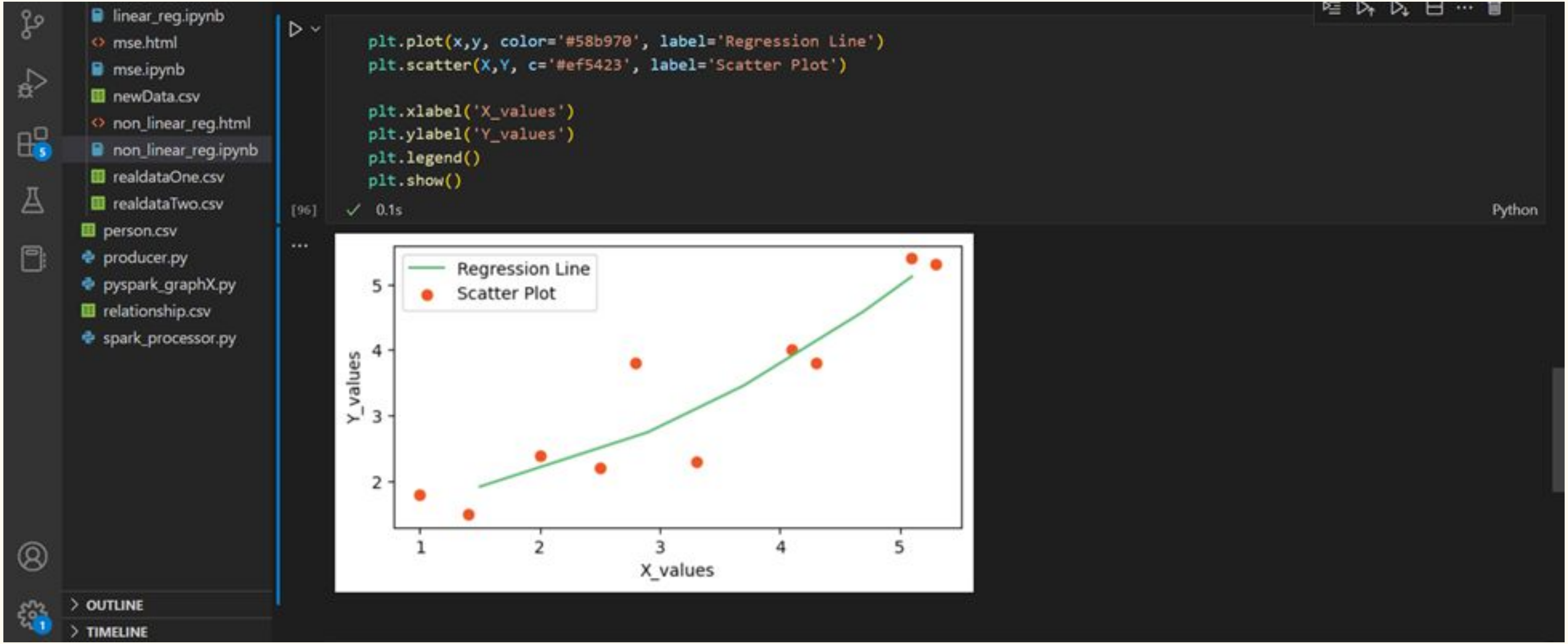
```
Sum of all the XY_Values:  509.76
Sum of all the XXX_Values:  2329.9862
```

# Using the formula to calculate slope(b)
n= len(X)
b = round(((n* xy_values.sum())-(xx_values.sum()*y_values.sum()))/ ((n*xxx_values.sum())-(xx_values.sum()**2)), 4)
print("slope(b) is: " , b)
[168] 0.3s
Python
slope(b) is: 0.1346
# # Using the formula to calculate intercept(a)
a = round((y_values.sum() - (b*xx_values.sum()))/n ,4)
print("intercept(a): ", a)
[169] 0.4s
Python
intercept(a): 1.6168



File Edit Selection View Go Run Terminal Help
● non_linear_reg.ipynb - kafka_prj - Visual Studio Code
regreswsion.ipynb
linear_reg.ipynb
mse.ipynb
non_linear_reg.ipynb
realdataTwo.csv
newData.csv
realdataOne.csv
person.csv
jupyter.py > non_linear_reg.ipynb > M↓ *********** Training phase ****************
+ Code + Markdown Run All Clear Outputs of All Cells Restart Variables Outline ···
base (Python 3.9.13)
*********** Training phase ****************
# Then substitute Intercept(a) and Slope(b) in regression equation formula
# Regression Equation(y) = a + bx
x = np.array(X)
y=a+b*x**2
print(y)
[170] 0.4s
Python
[1.7514 2.1552 3.082594 4.105554 5.397714 1.880616 2.45805 2.672064
3.879426 5.117746]

NonLinear Graph:



Validation Phase

realdataOne.csv
realdataTwo.csv
person.csv
producer.py
pyspark_graphX.py
relationship.csv
spark_processor.py

```python
# validation phase
# After calculating a1, b1, a2, b2 in Training Phase, the values are not changed with the new Real Data Sets in Validation Ph
# Only ŷ values are changed with the new Real Data Sets.
# Regression Equation(y) = a + bx
x = np.array([1.5,2.9,3.7,4.7,5.1])
y=a+b*x**2
print(y)
```

[93] ✓ 0.6s Python

```
[1.91965  2.748786 3.459474 4.590114 5.117746]
```

Mean Square Error(MSE):

*************** Calcuate MSE *************

```python
import numpy as np
#MSE for Training set Model 1
Y_true = [1.8,2.4,2.23,3.8,5.3,1.5,2.2,3.8,4.0,5.4]  # Y_true = Y (original values)

# Calculated values
Y_pred = [1.38,2.24,3.35,4.21,5.078,1.724,2.67,2.92,4.04,4.90] # Y_pred = Y'

# Mean Squared Error
MSE = np.square(np.subtract(Y_true,Y_pred)).mean()
print(MSE)
```

[13] ✓ 0.5s Python

```
0.29708599999999996
```

```python
#  MSE for Training set Model 2
Y_true = [1.8,2.4,2.23,3.8,5.3,1.5,2.2,3.8,4.0,5.4] # Y_true = Y (original values)

# Calculated values
Y_pred = [1.75,2.15,3.08,4.10,5.39,1.88,2.45,2.67,3.87,5.11]

  # Y_pred = Y'

# Mean Squared Error
MSE = np.square(np.subtract(Y_true,Y_pred)).mean()
print(MSE)
```

[14] ✓ 0.7s Python

```
0.24703999999999998
```

- linear_reg.html
- linear_reg.ipynb
- mse.html
- mse.ipynb
- newData.csv
- non_linear_reg.html
- non_linear_reg.ipynb
- realdataOne.csv
- realdataTwo.csv
- person.csv
- producer.py
- pyspark_graphX.py
- relationship.csv
- spark_processor.py

```python
#  MSE for validation set Model 1
Y_true = [1.7,2.7,2.5,2.8,5.5]
 # Y_true = Y (original values)

# Calculated values
Y_pred = [1.81,3.014,3.702,4.562,4.902]
 # Y_pred = Y'

# Mean Squared Error
MSE = np.square(np.subtract(Y_true,Y_pred)).mean()
print(MSE)
```

[3] ✓ 0.4s Python

```
1.0035496000000002
```

```python
# MSE for validation set Model 2
Y_true = [1.7,2.7,2.5,2.8,5.5]  # Y_true = Y (original values)

# Calculated values
Y_pred = [1.91,2.74,3.45,4.59,5.11]
  # Y_pred = Y'

# Mean Squared Error
MSE = np.square(np.subtract(Y_true,Y_pred)).mean()
print(MSE)
```

```
0.8608800000000001
```

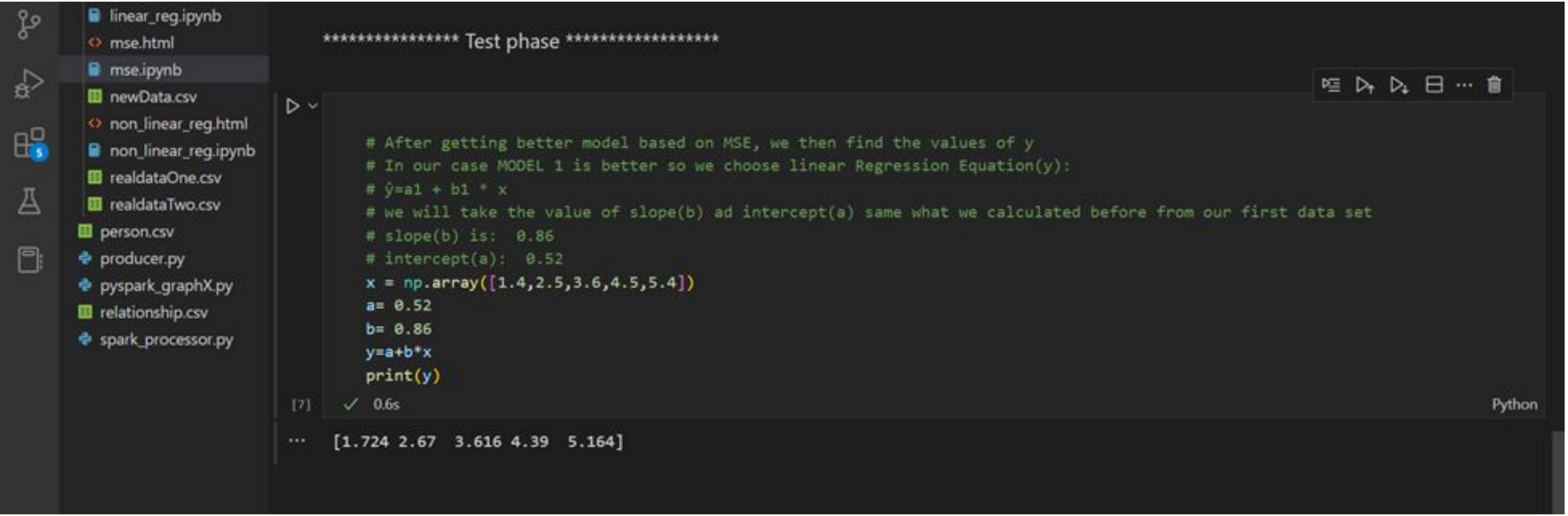
```python
# max(Training_Set_MSE, Validation_Set_MSE) / min(Training_Set_MSE, Validation_Set_MSE)
# Compare Model 1 and Model 2
# Model1
Training_Set_MSE = 0.2970859999999996
Validation_Set_MSE =1.0035496000000002
ModelOne =Validation_Set_MSE/Training_Set_MSE
print("Model 1 MSE :", round(ModelOne,2))


# Model 2
Training_Set_MSE =0.24703999999999998
Validation_Set_MSE = 0.8608800000000001
ModelTwo =Validation_Set_MSE/Training_Set_MSE
print("Model 2 MSE :", round(ModelTwo,2))

# Compare to find better model
if ( ModelTwo < ModelOne ):
    print("Model 2 is a better model")
else:
    print("Model 1 is a better model")
```

```
Model 1 MSE : 3.38
Model 2 MSE : 3.48
Model 1 is a better model
```

Test Phase:


linear_reg.ipynb
mse.html
mse.ipynb
newData.csv
non_linear_reg.html
non_linear_reg.ipynb
realdataOne.csv
realdataTwo.csv
person.csv
producer.py
pyspark_graphX.py
relationship.csv
spark_processor.py
**************** Test phase ******************
# After getting better model based on MSE, we then find the values of y
# In our case MODEL 1 is better so we choose linear Regression Equation(y):
# ŷ=a1 + b1 * x
# we will take the value of slope(b) ad intercept(a) same what we calculated before from our first data set
# slope(b) is: 0.86
# intercept(a): 0.52
x = np.array([1.4,2.5,3.6,4.5,5.4])
a= 0.52
b= 0.86
y=a+b*x
print(y)
[7]
0.6s
Python
[1.724 2.67 3.616 4.39 5.164]

Training Phase				Validation Phase				Test Phase	
Real Data Set 1 50% of the collcted data		Model 1: Linear Regression	Model 2: Non-Linear Regression	Real Data Set 2 25% of the collcted data		Model 1: Linear Regression	Model 2: Non-Linear Regression	Real Data Set 3 25% of the collcted data	The better model (Model 1 or Model 2) selected from the Validation Phase based on the analysis of overfitting will be used to calculate ŷ

- **After calculating a1, b1, a2, b2 in Training Phase, the values are not changed with the new Real Data Sets in Validation Phase and Test Phase.**
- **Only ŷ values are changed with the new Real Data Sets.**

x	y	ŷ=a1 + b1 * x	ŷ=a2 + b2 * x^2	x	y	ŷ=a1 + b1 * x	ŷ=a2 + b2 * x^2	x	ŷ=a1 + b1 * x or ŷ=a2 + b2 * x^2
1	1.8	1.38	1.75	1.5	1.7	1.81	1.91	1.4	1.724
2	2.4	2.24	2.15	2.9	2.7	3.014	2.74	2.5	2.67
3.3	2.3	3.358	3.08	3.7	2.5	3.702	3.45	3.6	3.616
4.3	3.8	4.218	4.10	4.7	2.8	4.562	4.59	4.5	4.39
5.3	5.3	5.078	5.39	5.1	5.5	4.902	5.11	5.4	5.164
1.4	1.5	1.724	1.88	X	X	X	X	X	X
2.5	2.2	2.67	2.45	X	X	X	X	X	X
2.8	3.8	2.928	2.67	X	X	X	X	X	X
4.1	4.0	4.046	3.87	X	X	X	X	X	X
5.1	5.4	4.906	5.11	X	X	X	X	X	X

Enhancement Ideas:

- This data set is used to minimize overfitting.
    - To verify that any increase in accuracy over the training data set actually yields an increase in accuracy over a data set that has not been shown to the model before, or at least the model hasn't trained on it (i.e. validation data set).
        - If the accuracy over the training data set increases, but the accuracy over then validation data set stays the same or decreases, then you're overfitting your model and you should stop training.
    - A test set (i.e., validation set) is a set of data that is independent of the training data, but that follows the same probability distribution as the training data.
        - If a model fit to the training set also fits the test set well, minimal overfitting has taken place.
        - If the model fits the training set much better than it fits the test set, overfitting is likely the cause.

- In order to avoid overfitting, when any classification parameter needs to be adjusted, it is necessary to have a validation set in addition to the training and test sets.
    - For example if the most suitable classifier for the problem is sought,
        - Training Data is used to train the candidate algorithms.
        - Validation Data is used to compare their performances and decide which one to take.
        - Test Data is used to obtain the performance characteristics such as accuracy, sensitivity, specificity.

Conclusion:

- If the accuracy over the training data set increases, but the accuracy over the validation data set stays the same or decreases, then you're overfitting your model and you should stop training.

- If the model fits the training set much better than it fits the test set (i.e., validation set), overfitting is likely the cause.

References:

Prof Chang's class material

Linear regression Algorithm

Non Linear Regression Algorithm

Thank you