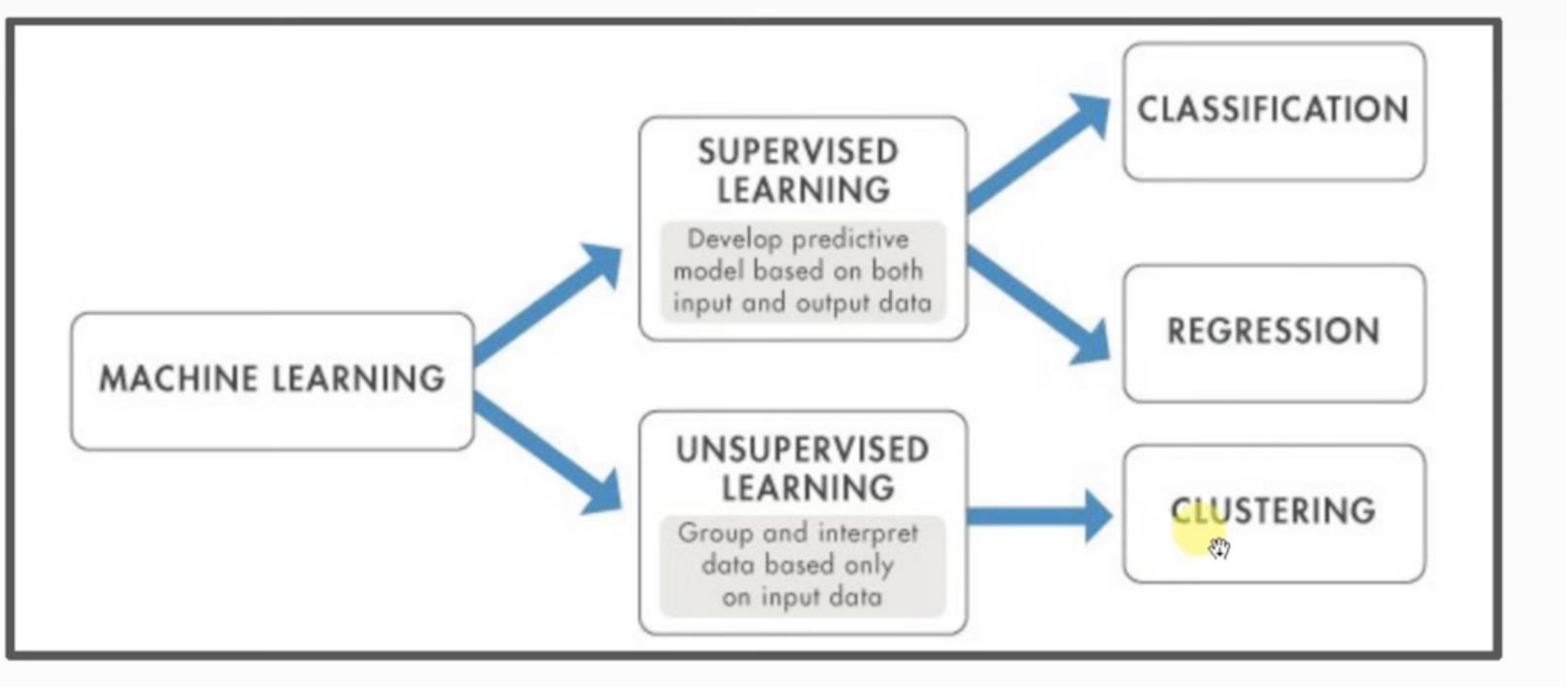
Machine Learning

- 1st Term, 2025/2026
- September 2025

- **Prof. Mohammed A. Al Ghamdi**

Regression:

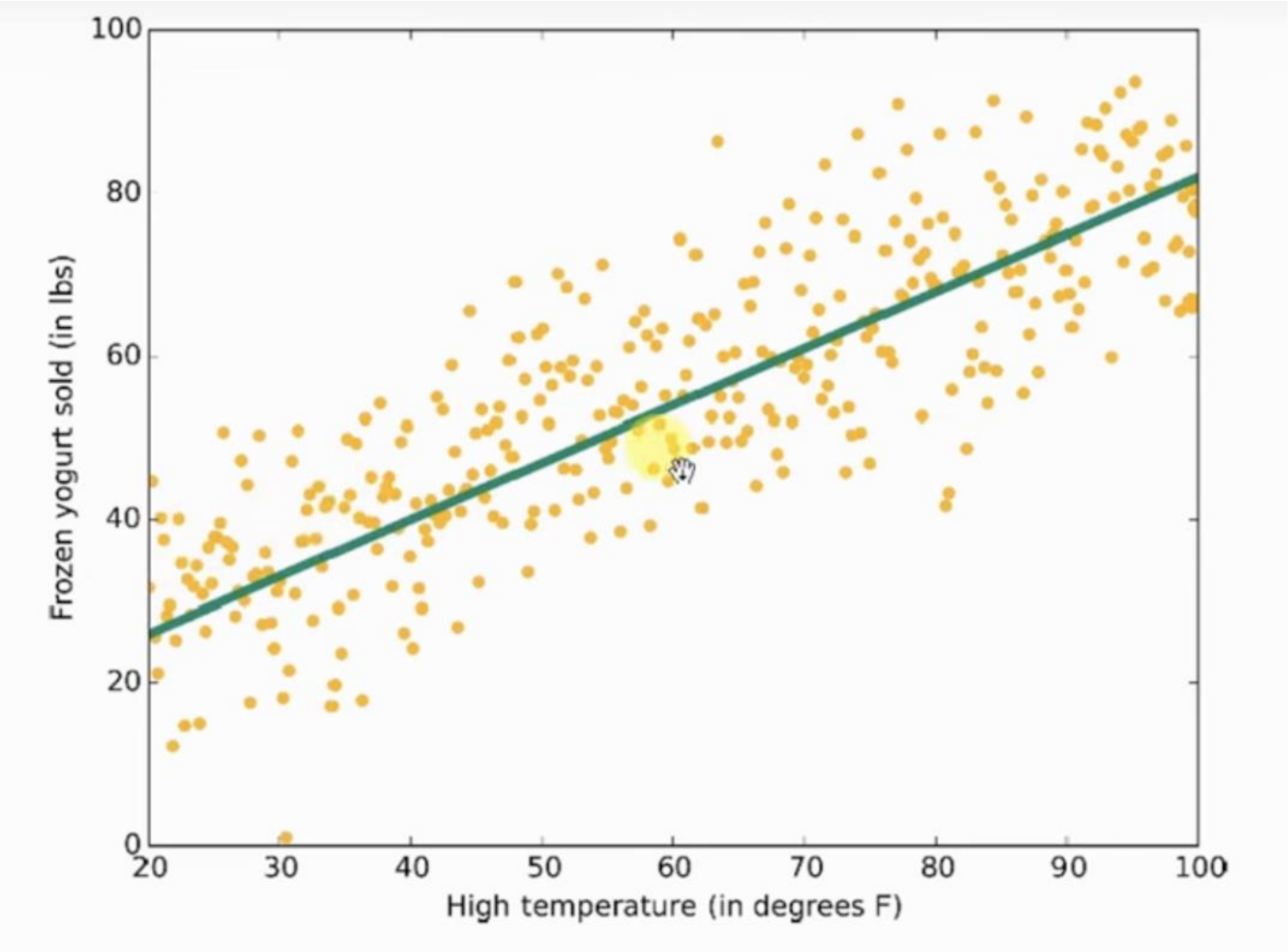


MACHINE LEARNING
SUPERVISED LEARNING
Develop predictive model based on both input and output data
UNSUPERVISED LEARNING
Group and interpret data based only on input data
CLASSIFICATION
REGRESSION
CLUSTERING

Regression:

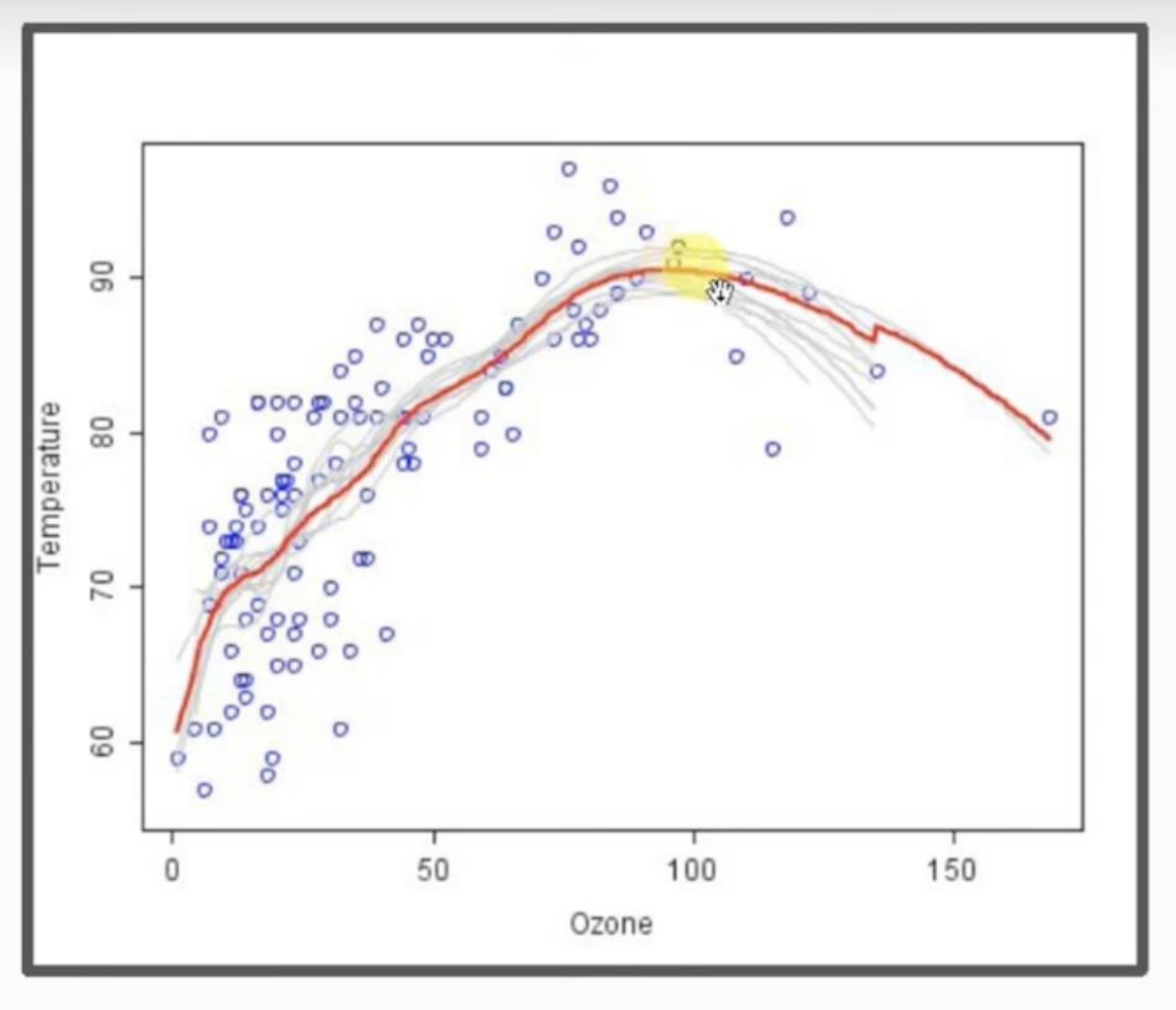
- Data.
- Study the Data.
- Connected Data.
- Find New Data

Regression:

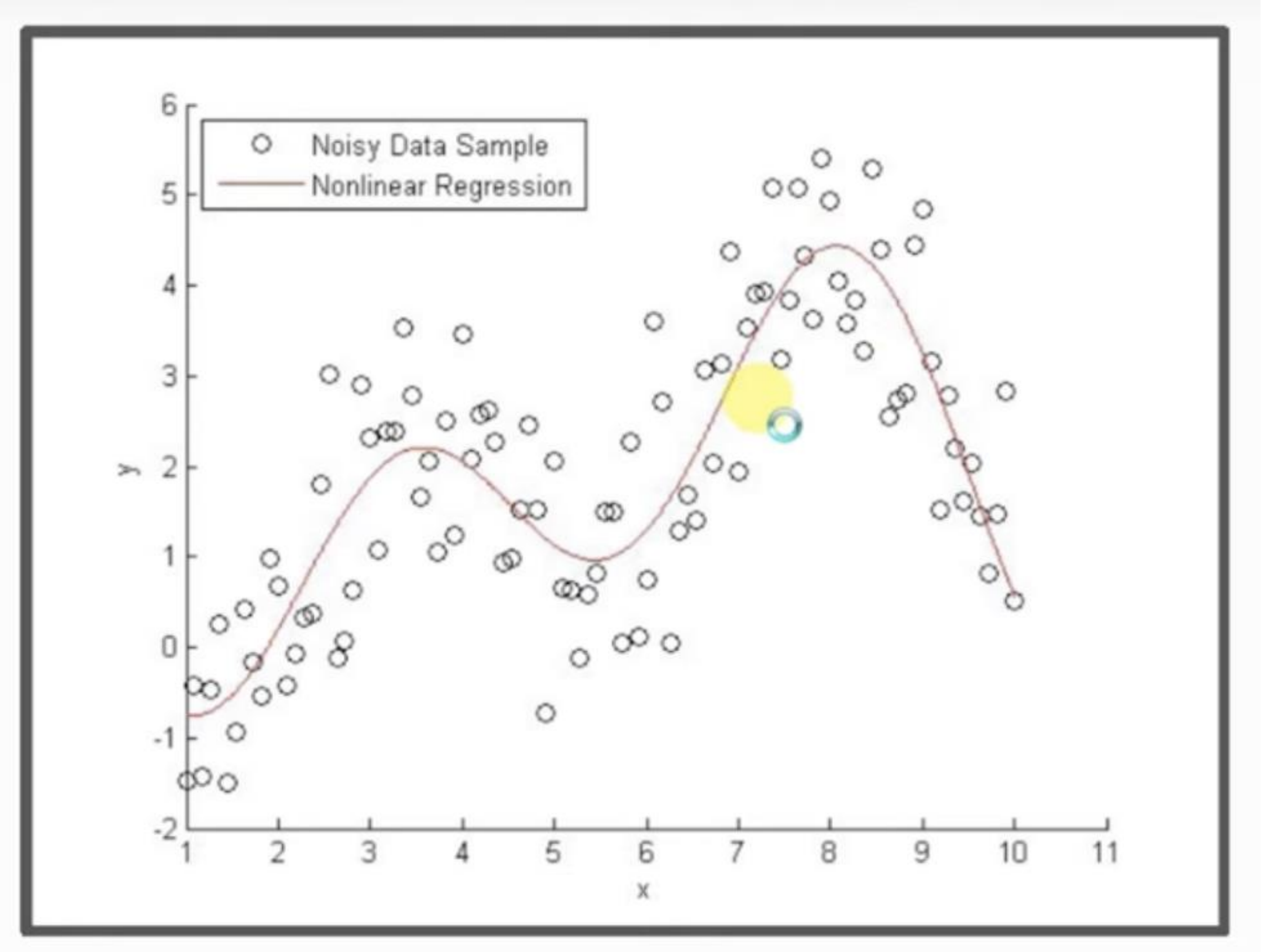


Frozen yogurt sold (in lbs)
High temperature (in degrees F)
0
20
40
60
80
100
20
30
40
50
60
70
80
90
100

Regression:



Regression:



Noisy Data Sample
Nonlinear Regression
y
x
6
5
4
3
2
1
0
-1
-2
1
2
3
4
5
6
7
8
9
10
11

Regression Applications:

- Temperatures.
- House Prices.
- Car Prices.
- Match Results.
- Stock Market.
- ... etc.

Linear Regression:

- **It is sometimes called:**
- One Variable Regression or Univariate Regression.

# of Favourites (X)	# of Posts (Y)
36	14
21	12
47	22
11	11
72	33
95	46
58	25
81	34
9	3
18	12
2	0
15	4
29	10
66	19
31	20

Linear regression equation

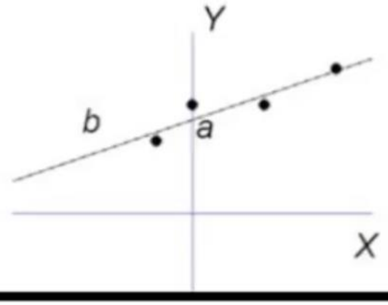
(without error)

$$\hat{Y} = bX + a$$

predicted values of Y

b = slope = rate of predicted ↑/↓ for Y scores for each unit increase in X

Y-intercept = level of Y when X is 0



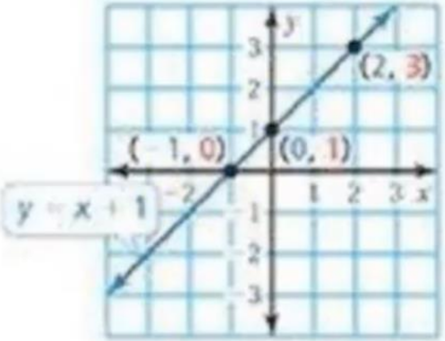
Linear Regression:

Relationship between Two Variables.

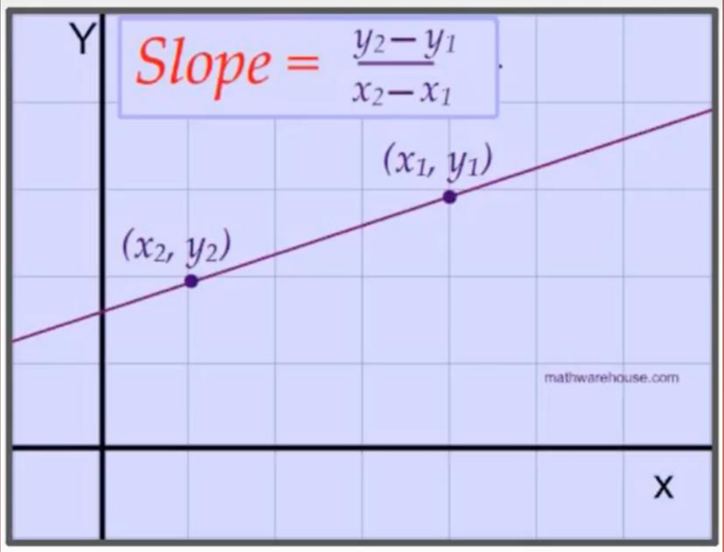
Linear Equations

- A **linear equation** is an equation whose graph is a line.
- The points on the line are **solutions** of the equation.

x	y	(x, y)
−1	0	(−1, 0)
0	1	(0, 1)
2	3	(2, 3)



Linear Regression:

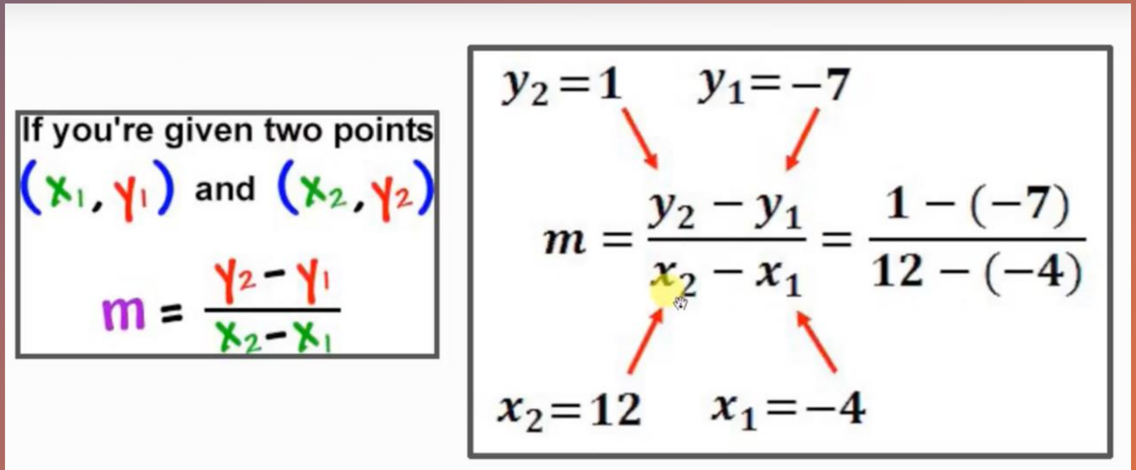


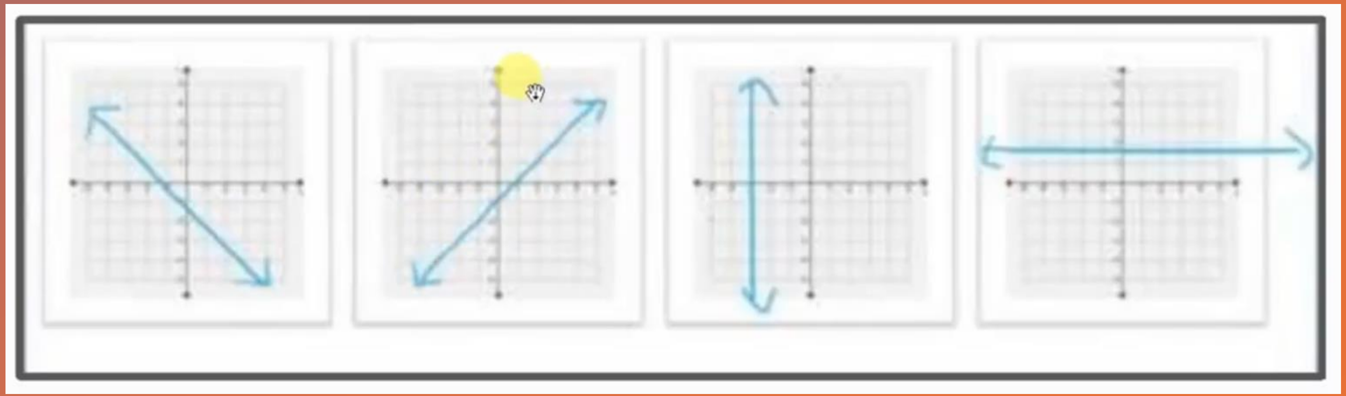
$y_2 = 1 \quad y_1 = -7$

$$m = \frac{y_2 - y_1}{x_2 - x_1} = \frac{1 - (-7)}{12 - (-4)}$$

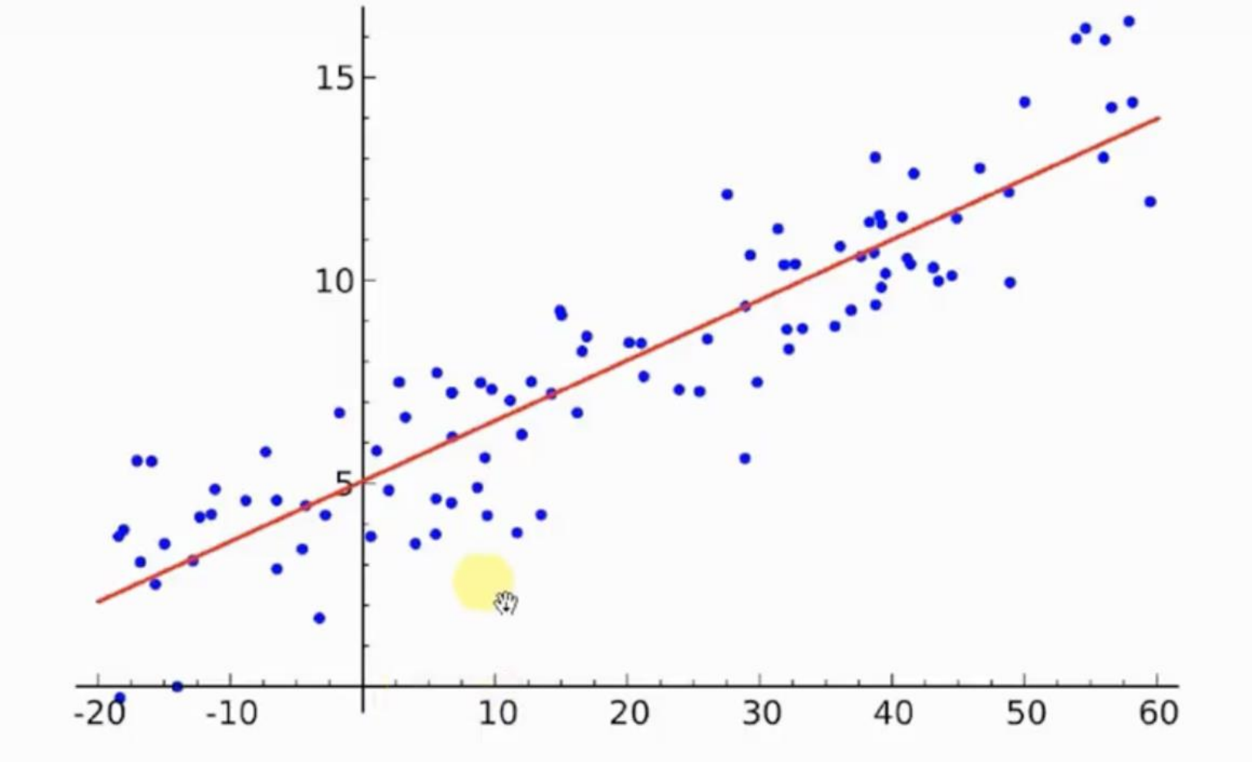
$x_2 = 12 \quad x_1 = -4$

Linear Regression:





Linear Regression:



Linear Regression:

X = Input.
Y = Output.
m = Rows.
n = Features.
H(x) = Predicted Value.
Cost J = Mistake Value.
Theta θ = Theta of X.

# of Favourites (X)	# of Posts (Y)
36	14
21	12
47	22
11	11
72	33
95	46
58	25
81	34
9	3
18	12
2	0
15	4
29	10
66	19
31	20

Linear Regression Equation:

Hypothesis: $h_\theta(x) = \theta_0 + \theta_1 x$

Parameters: θ_0, θ_1

Cost Function: $J(\theta_0, \theta_1) = \frac{1}{2m} \sum_{i=1}^{m} \left(h_\theta(x^{(i)}) - y^{(i)}\right)^2$

Goal: $\underset{\theta_0, \theta_1}{\text{minimize}}\ J(\theta_0, \theta_1)$

- The aim is to reduce the difference between the predicted value "h(x)" and the real value "y".
- Also, the target is to find out the value of θ_0 and θ_1 to reduce the cost value as much as we can.
- Sometimes it is called Cost Error Function.

Linear Regression Equation:

Theta0 = 5 , theta 1 = 2 Equation $h(x) = 5 + 2x$

X	Y	h(x)	h(x) - y	$(h(x) - y)^2$
1	7			
2	8			
2	7			
3	9			
4	11			
5	10			
5	12			

Linear Regression Equation:

Theta0 = 5 , theta 1 = 2 Equation $h(x) = 5 + 2x$

X	Y	h(x)	h(x) - y	$(h(x) - y)^2$
1	7	7	0	0
2	8	9	1	1
2	7	9	2	4
3	9	11	2	4
4	11	13	2	4
5	10	15	5	25
5	12	15	3	9

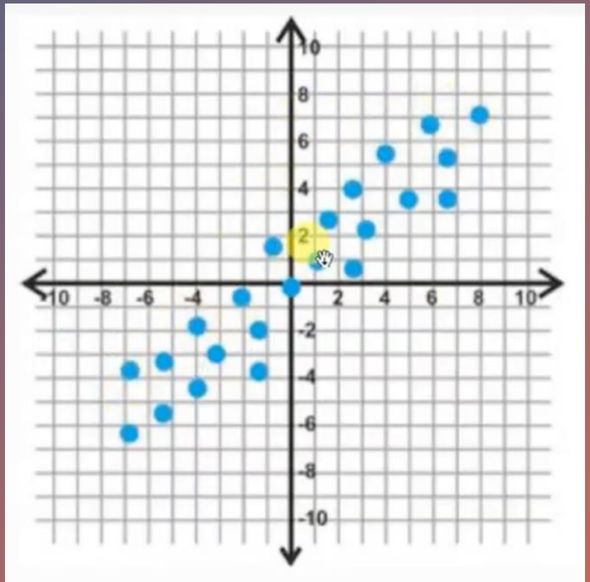
$$\boxed{J(\theta_0, \theta_1) = \frac{1}{2m} \sum_{i=1}^{m} \left(h_\theta(x^{(i)}) - y^{(i)}\right)^2}$$

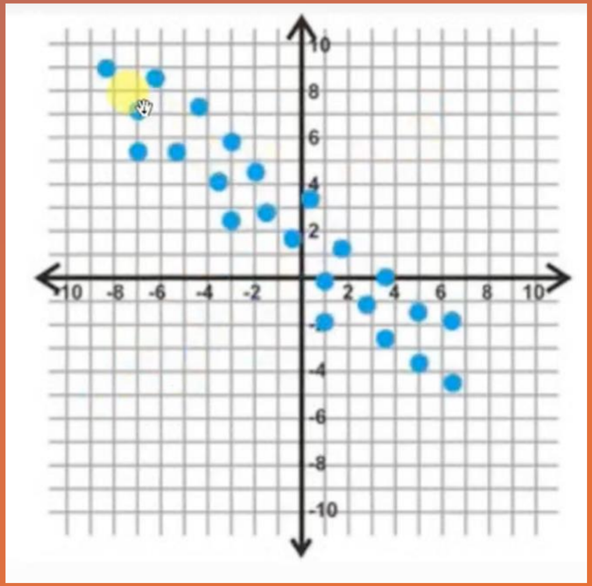
J = 1 / 14 (0+1+4+4+4+25+9)

J = 47/14 = 3.3

Linear Regression Equation:

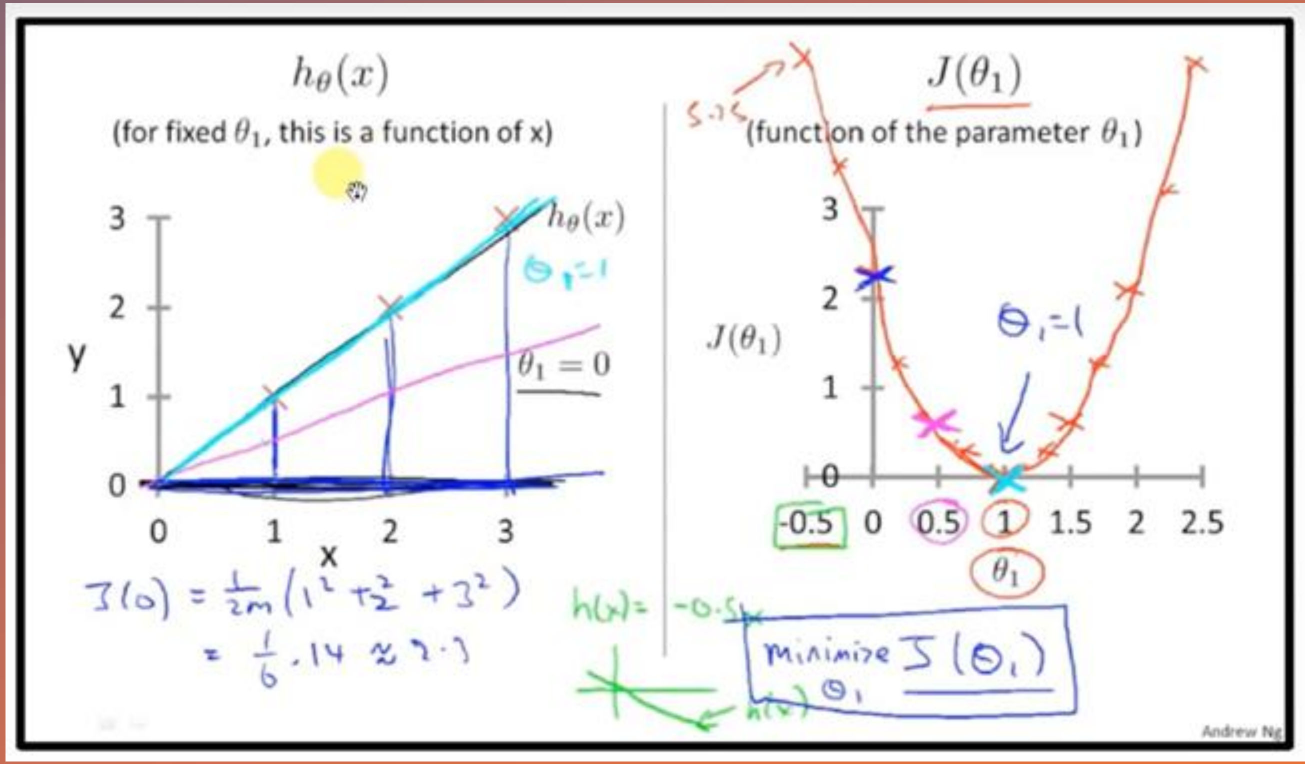
Best Fit Line





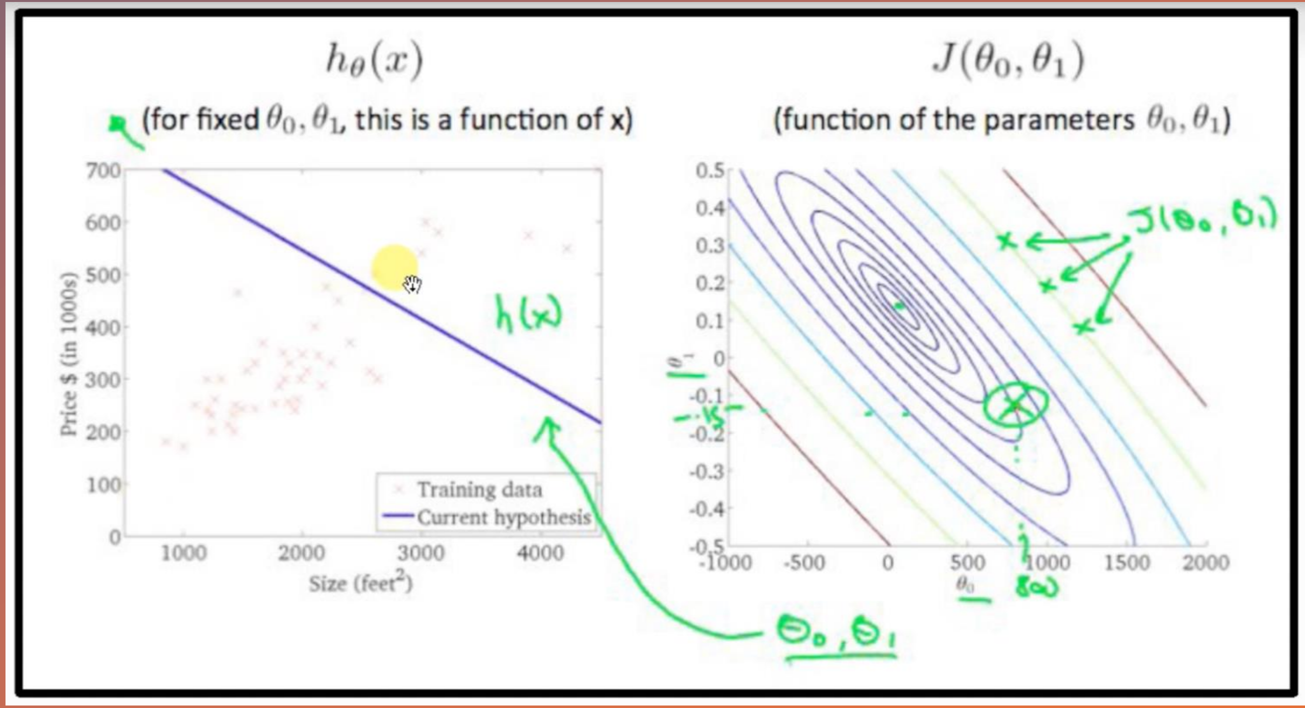
Linear Regression Equation:

Best Fit Line



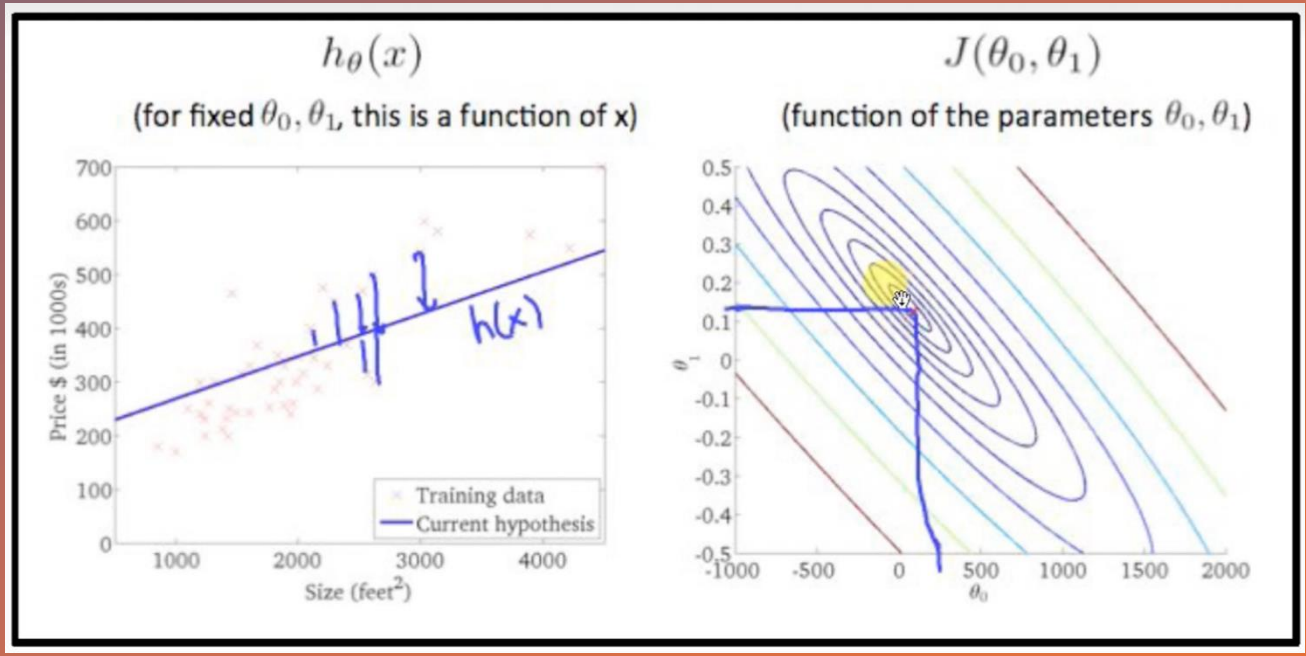
Linear Regression Equation:

Best Fit Line

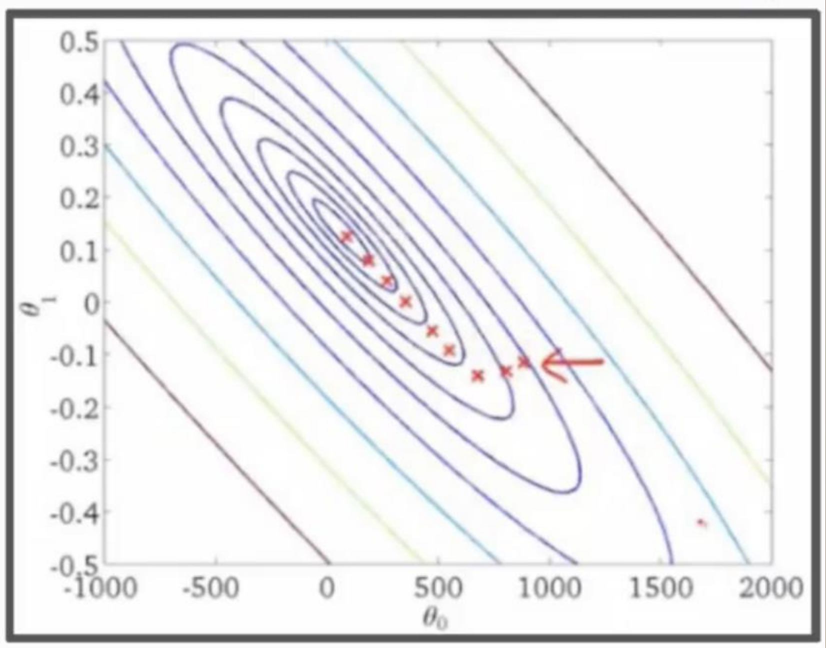


Linear Regression Equation:

Best Fit Line



Gradient Descent:



- We keep looking for minimizing the values of θ_0 and θ_1 for the best cost value.

Gradient Descent:

repeat until convergence:

$$\theta_j := \theta_j - \alpha \frac{\partial}{\partial \theta_j} J(\theta_0, \theta_1)$$

- := mean overwriting.
- α is a factor.
- If the value of α is large, it means the procedures will be conducted faster but with less accuracy.
- and vice versa.
- Equation is repeated for θ_0 and θ_1 in parallel.

Gradient Descent:

$$\theta_0 := \theta_0 - \alpha \frac{1}{m} \sum_{i=1}^{m} (h_\theta(x_i) - y_i)$$

$$\theta_1 := \theta_1 - \alpha \frac{1}{m} \sum_{i=1}^{m} ((h_\theta(x_i) - y_i) x_i)$$

Example:

House Size (X)	Price $$$ (Y)
100	300
95	285
90	270
80	240
80	235
70	200
70	205
60	180

- For obtaining the best fit line, we suppose that:
  - ❖ $\theta_0 = 1$ and $\theta_1 = 3$.
- So, the equation will be:
  - ❖ $h(x) = 1 + 3x$.

Example:

House Size (X)	Price $$$ (Y)	h(x)	h(x) - y
100	300	301	1
95	285	286	1
90	270	271	1
80	240	241	1
80	235	241	6
70	200	211	11
70	205	211	6
60	180	181	1
sum			28

$$\theta_0 := \theta_0 - \alpha \frac{1}{m} \sum_{i=1}^{m} (h_\theta(x_i) - y_i)$$

- **Suppose the value of α is equal to 0.002. So,**
- **θ_0 = 1- ((0.002/8) * 28))**
- **θ_0 = 1- 0.007**
- **θ_0 = 0.993**

Example:

House Size (X)	Price $$$ (Y)	h(x)	h(x) - y	(h(x) – y) * x
100	300	301	1	100
95	285	286	1	95
90	270	271	1	90
80	240	241	1	80
80	235	241	6	480
70	200	211	11	770
70	205	211	6	420
60	180	181	1	60
sum			28	2095

$$\theta_1 := \theta_1 - \alpha \frac{1}{m} \sum_{i=1}^{m} ((h_\theta(x_i) - y_i)x_i)$$

- **Suppose the value of α is equal to 0.002. So,**
- **θ_1 = 3 - ((0.002/8) * 2095))**
- **θ_1 = 3- 0.52**
- **θ_1 = 2.48**

Example:

House Size (X)	Price $$$ (Y)	h(x)	h(x) - y	(h(x) – y) * x
100	300	301	1	100
95	285	286	1	95
90	270	271	1	90
80	240	241	1	80
80	235	241	6	480
70	200	211	11	770
70	205	211	6	420
60	180	181	1	60
sum			28	2095

- **Iteration 0 => $\theta_0 = 1$, $\theta_1 = 3$**
- **Iteration 1 => $\theta_0 = 0.993$, $\theta_1 = 2.48$**
- **……**
- **……**
- **……**
- **Iteration z-1=> θ_0=0.824, $\theta_1 = 1.773$**
- **Iteration z => $\theta_0 = 0.825$, $\theta_1 = 1.772$**

Linear Regression in Python:

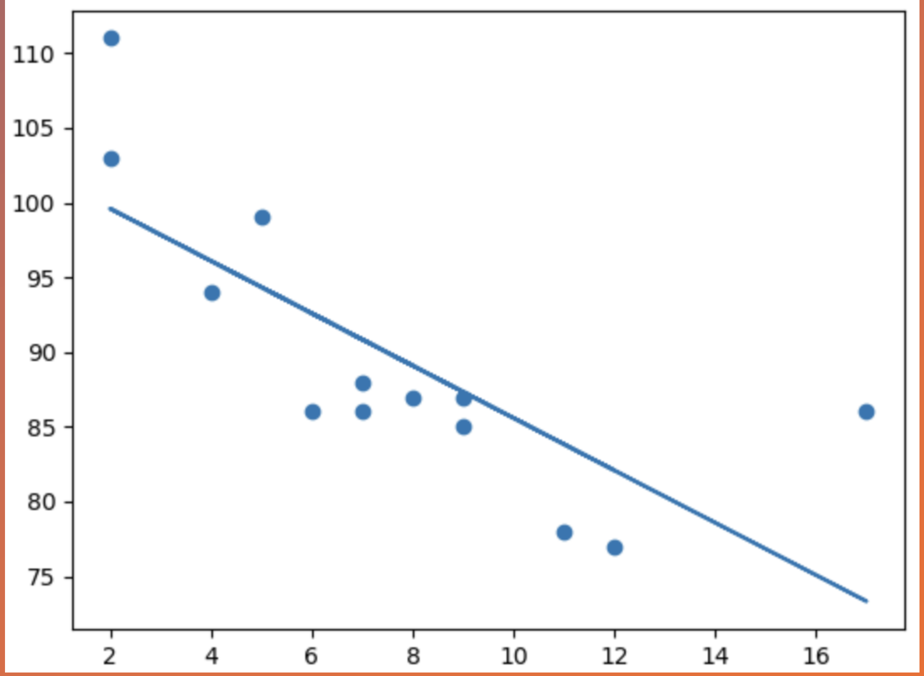
- We check weather the there is a relationship between the provided data x-axis and y-axis or not, R.
- The r value ranges from -1 to 1, where 0 means no relationship, and 1 (and -1) means 100% related.

```
from scipy import stats #Scientific Python Library

x = [5,7,8,7,2,17,2,9,4,11,12,9,6]
y = [99,86,87,88,111,86,103,87,94,78,77,85,86]

slope, intercept, r, p, std_err = stats.linregress(x, y)

print(r)
```

Linear Regression in Python:

```
import matplotlib.pyplot as plt
from scipy import stats

x = [5,7,8,7,2,17,2,9,4,11,12,9,6]
y = [99,86,87,88,111,86,103,87,94,78,77,85,86]

slope, intercept, r, p, std_err = stats.linregress(x, y)

def myfunc(x):
  return slope * x + intercept

mymodel = list(map(myfunc, x))

plt.scatter(x, y) #Draw the original scatter plot.
plt.plot(x, mymodel) #Draw the line of linear regression.
plt.show() #Display the diagram.
```

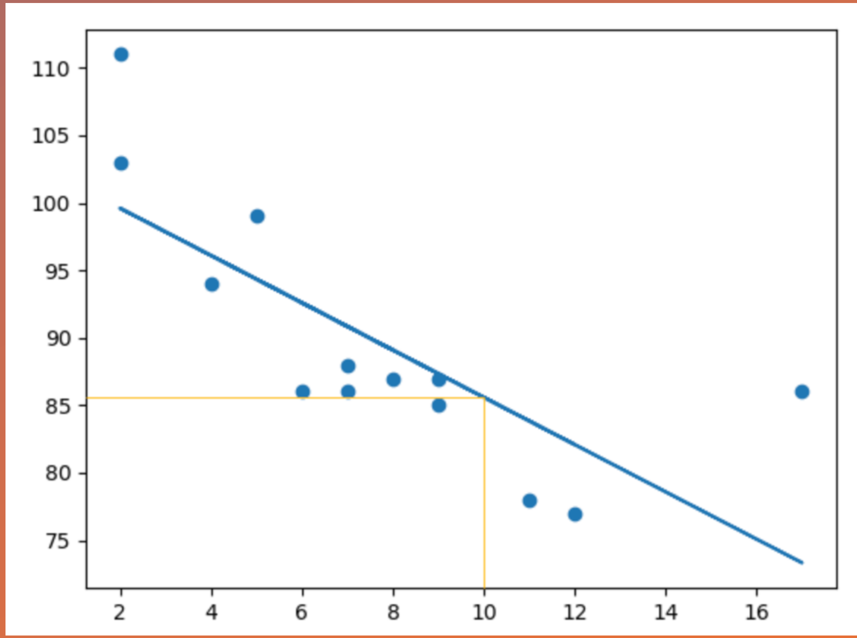
- **Remember the key values of Linear Regression (Slope, intercept, ...).**



Linear Regression in Python:

- **Predict the speed of a 10 years old car.**

```python
#Predict Future Values
from scipy import stats

x = [5,7,8,7,2,17,2,9,4,11,12,9,6]
y = [99,86,87,88,111,86,103,87,94,78,77,85,86]

slope, intercept, r, p, std_err = stats.linregress(x, y)

def myfunc(x):
  return slope * x + intercept

speed = myfunc(10)

print(speed)
```



Multiple Linear Regression in Python:

- The data is named data.csv
- It contain information about group of cars.
- It is required to predict the CO_2 of a car with specific info.

Car	Model	Volume	Weight	CO2
Toyoty	Aygo	1000	790	99
Mitsubishi	Space Star	1200	1160	95
Skoda	Citigo	1000	929	95
Fiat	500	900	865	90
Mini	Cooper	1500	1140	105
VW	Up!	1000	929	105
Skoda	Fabia	1400	1109	90
Mercedes	A-Class	1500	1365	92
Ford	Fiesta	1500	1112	98
Audi	A1	1600	1150	99
Hyundai	I20	1100	980	99
Suzuki	Swift	1300	990	101
Ford	Fiesta	1000	1112	99
Honda	Civic	1600	1252	94
Hundai	I30	1600	1326	97
Opel	Astra	1600	1330	97
BMW	1	1600	1365	99

Multiple Linear Regression in Python:

- The output is: [107.2087328]

```
import pandas #The Pandas module allows us to read csv files and return a DataFrame object.
from sklearn import linear_model #Scikit Learn Library

df = pandas.read_csv("data.csv")

x = df[['Weight', 'Volume’]] #Make a list of the independent values and call it x.
y = df['CO2’] #Put the dependent values in a variable called it y.

regr = linear_model.LinearRegression() #From the sklearn module we will use the LinearRegression() method to create a linear regression object.
regr.fit(x.values, y) #This object has a method called fit() that takes the independent and dependent values as parameters and fills the regression object with data that describes the relationship

#predict the CO2 emission of a car where the weight is 2300kg, and the volume is 1300cm3:
predictedCO2 = regr.predict([[2300, 1300]])

print(predictedCO2)
```

Regression Evaluation Metrics

- **Mean Absolute Error (MAE):**

It calculates the absolute difference between actual and predicted values.

$$MAE = \frac{1}{n} \sum_{i=1}^{n} |xi - yi|$$

where:

- x_i represents the actual or observed value for the *i-th* data point.
- y_i represents the predicted value for the *i-th* data point.

Mean Absolute Error (MAE)

$$MAE = \frac{1}{n} \sum_{i=1}^{n} |xi - yi|$$

Actual Values	10	15	12	18	20
Predicted Values	12	15	10	20	18

$$= \frac{|10-12|+|15-15|+|12-10|+|18-20|+|20-18|}{5}$$

$$= \frac{2+0+2+2+2}{5} = \frac{8}{5}$$

$$= 1.6$$

It means that, on average, the model's predictions are approximately 1.6 away from the true values.

Regression Evaluation Metrics

- **Mean Squared Error (MSE):**

The average squared difference between the predicted and actual values of the target variable.

$$MSE = \frac{1}{n} \sum_{i=1}^{n} (x_i - yi)^2$$

where:

- x_i represents the actual or observed value for the *i-th* data point.
- y_i represents the predicted value for the *i-th* data point.

Mean Squared Error (MSE)

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(x_i - yi)^2$$

Actual Values	10	15	12	18	20
Predicted Values	12	15	10	20	18

$$= \frac{(10-12)^2+(15-15)^2+(12-10)^2+(18-20)^2+(20-18)^2}{5}$$

$$= \frac{4+0+4+4+4}{5}$$

$$= \frac{16}{5} = 3.2$$

It means that, on average, the squared prediction errors are approximately 3.2%.

Regression Evaluation Metrics

- **Root Mean Squared Error (RMSE):**

It is the square root of the mean squared error.

$$RMSE = \sqrt{MSE}$$
$$= \sqrt{\frac{1}{n} \sum_{i=1}^{n} (x_i - yi)^2}$$

where:

- x_i represents the actual or observed value for the *i-th* data point.
- y_i represents the predicted value for the *i-th* data point.

Root Mean Squared Error (RMSE)

$$RMSE = \sqrt{MSE}$$

Actual Values	10	15	12	18	20
Predicted Values	12	15	10	20	18

$$\text{RMSE} = \sqrt{3.2}$$
$$= 1.8$$

It indicates that, on average, the model's predictions have an error of approximately 1.8 in the same units as the actual value.

Regression Evaluation Metrics

- **R^2 – Score:**

It determines the proportion of variance in the dependent variable that can be explained by the independent variable. In other words, r-squared shows how well the data fit the regression model (the goodness of fit).

$$R^2 = 1 - \frac{sum\ squared\ regression\ (SSR)}{total\ sum\ of\ squares\ (SST)}$$

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(x_i - yi)^2/n}{\sum_{i=1}^{n}(xi - zi)^2/n}$$

where:

- x_i represents the actual or observed value for the *i-th* data point.
- y_i represents the predicted value for the *i-th* data point.
- z_i represents the mean value of the actual values.

R^2 – Score

$$R^2 = 1 - \frac{sum\ squared\ regression\ (SSR)}{total\ sum\ of\ squares\ (SST)}$$

$$R^2 = 1 - \frac{\sum_{i=1}^{n}(x_i - yi)^2/n}{\sum_{i=1}^{n}(xi - zi)^2/n}$$

Actual Values	10	15	12	18	20
Predicted Values	12	15	10	20	18

$$SSR = \frac{(10-12)^2+(15-15)^2+(12-10)^2+(18-20)^2+(20-18)^2}{5}$$

$= 3.2$

$$SST = \frac{10+15+12+18+20}{5} = 15$$

$$= \frac{(10-15)^2+(15-15)^2+(12-15)^2+(18-15)^2+(20-15)^2}{5}$$

$= \frac{68}{5}$

$= 13.6$

$$R^2 = 1 - \frac{3.2}{13.6} = 1 - 0.235$$

$$R^2 = 0.765$$

This means that the actual number account for 76.5 % of the variation.

Thanks