Linear Regression

By Saurav Poudel

Types of Linear Regression

1. Simple Linear Regression
2. Multiple Linear Regression

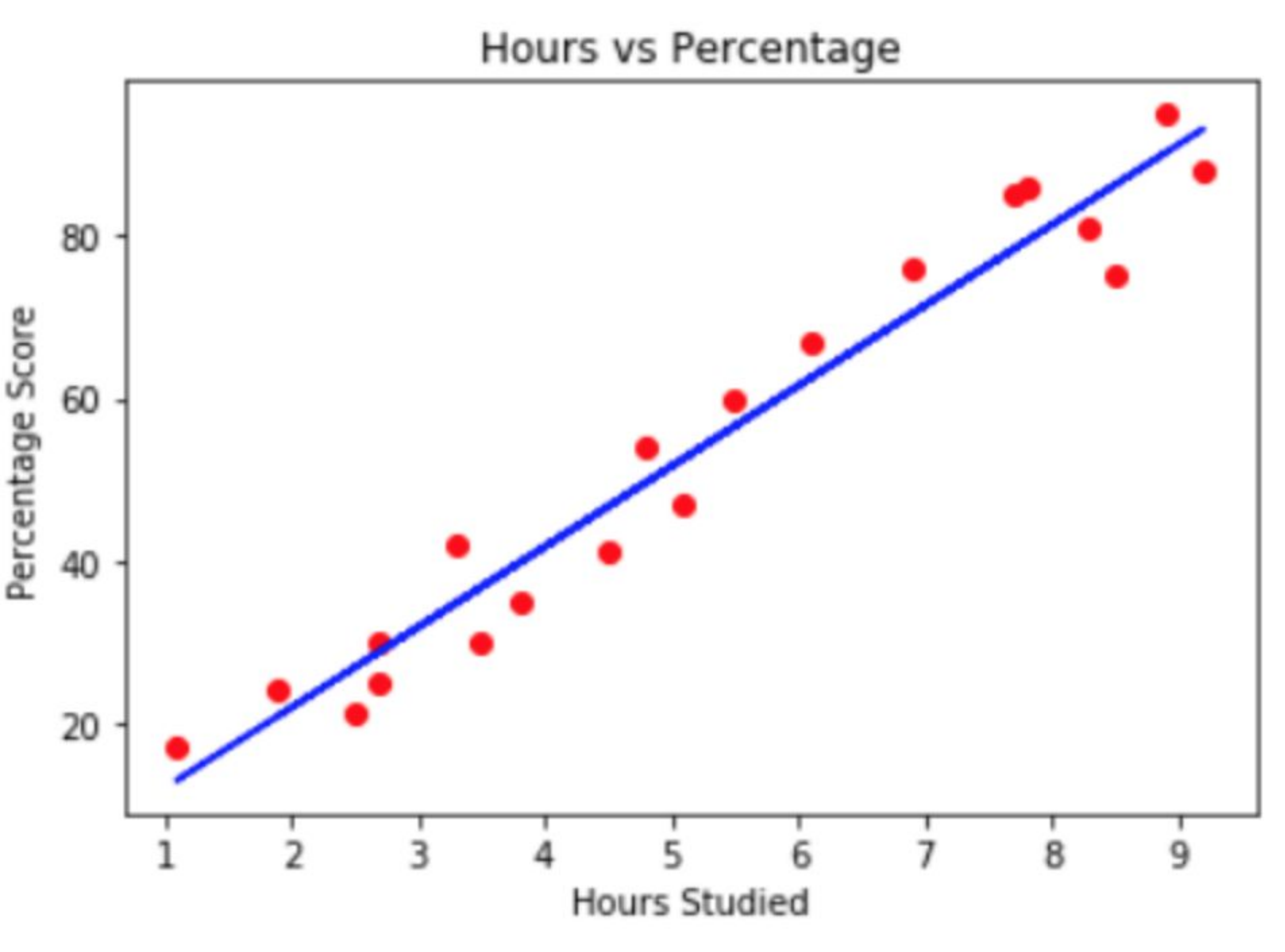
Simple Linear Regression!

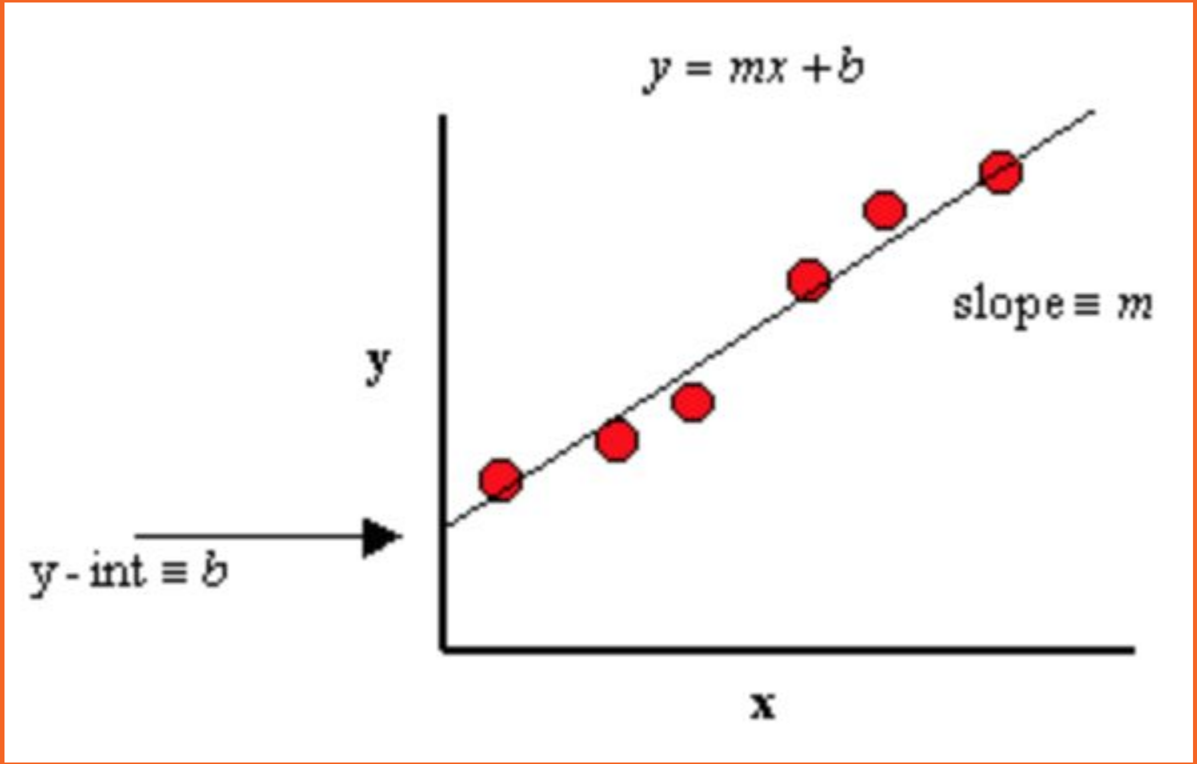
Building block of Machine Learning

Tip

One Input Variable.

One Output Variable


Hours vs Percentage
Percentage Score
20
40
60
80
1
2
3
4
5
6
7
8
9
Hours Studied

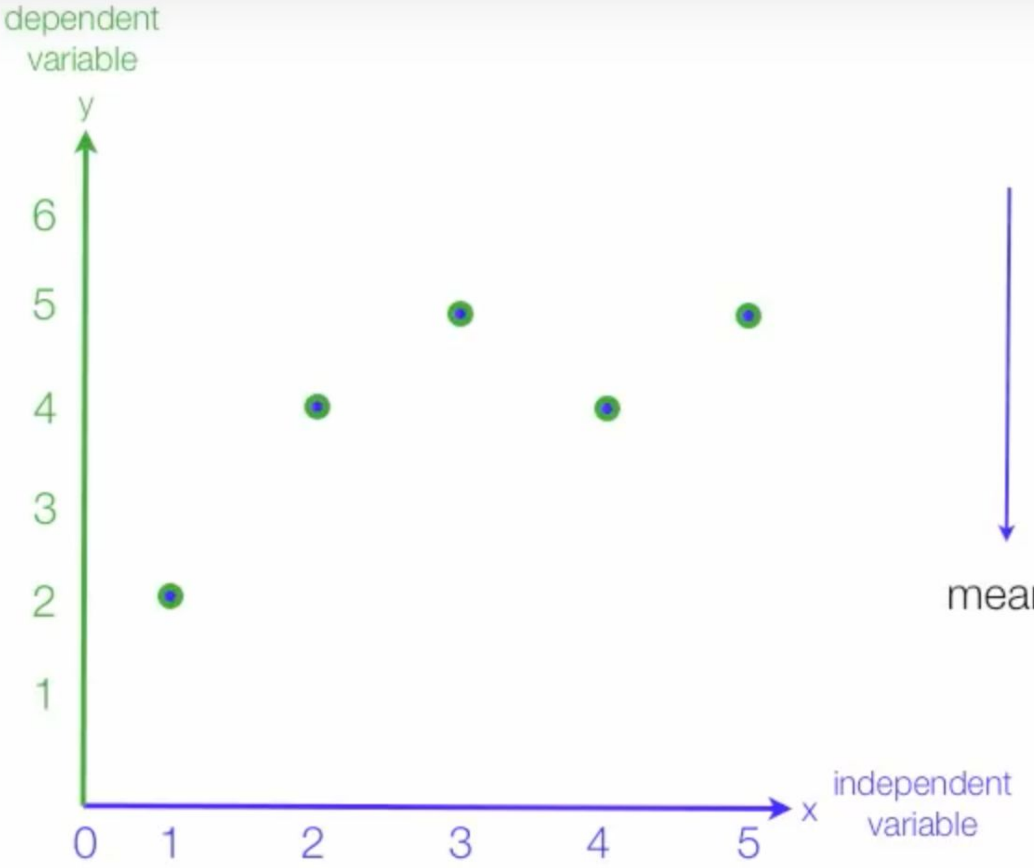

$y = mx + b$
slope ≡ m
y
y - int ≡ b
x

How to calculate the coefficients of Simple Linear Regression?

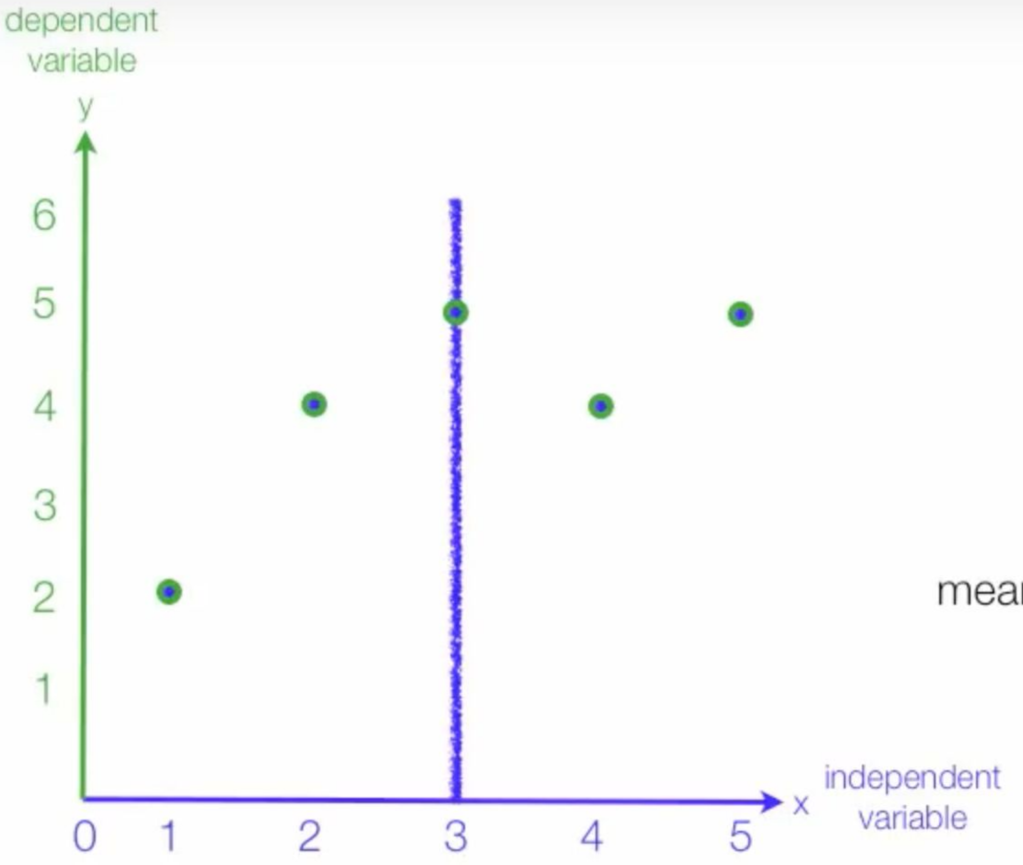
Tip

We will go from High School slope calculation to some basic statistics now!

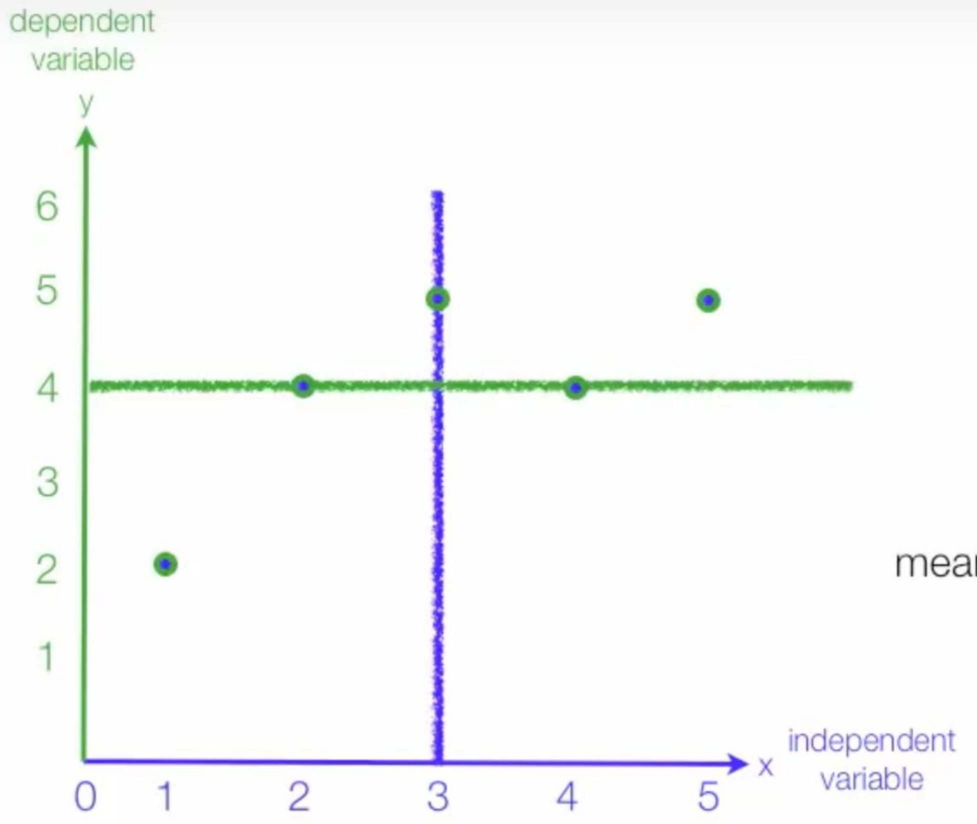
Using Statistical Method



	independent variable	dependent variable
	x	y
	1	2
	2	4
	3	5
	4	4
	5	5
mean	3	

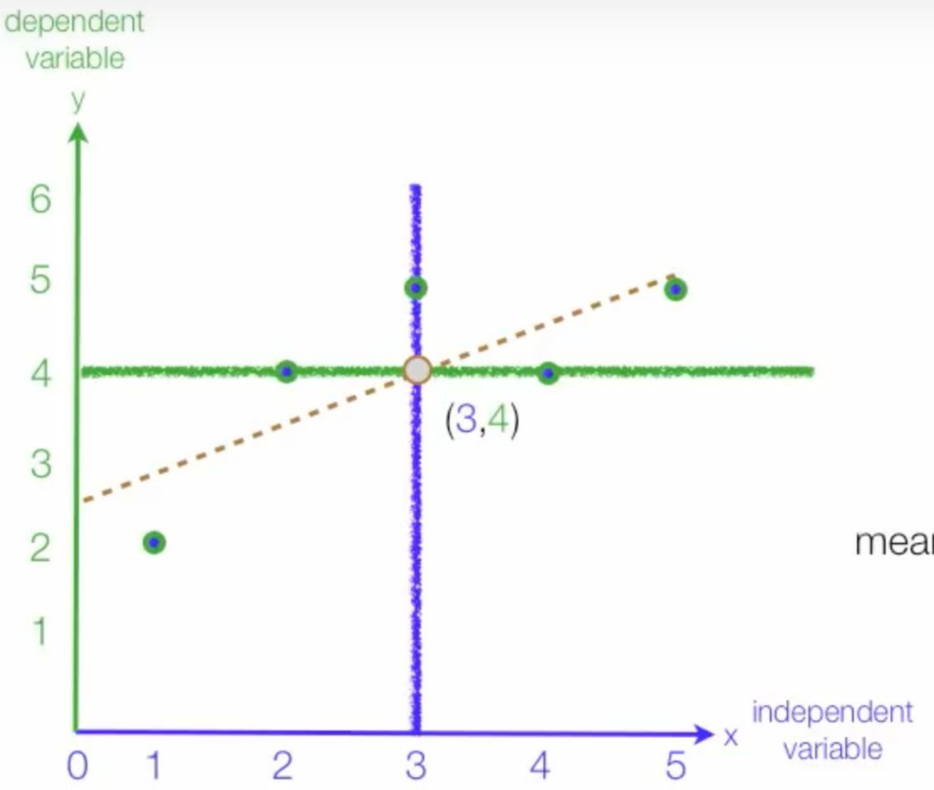


independent variable	dependent variable
x	y
1	2
2	4
3	5
4	4
5	5
mean 3	



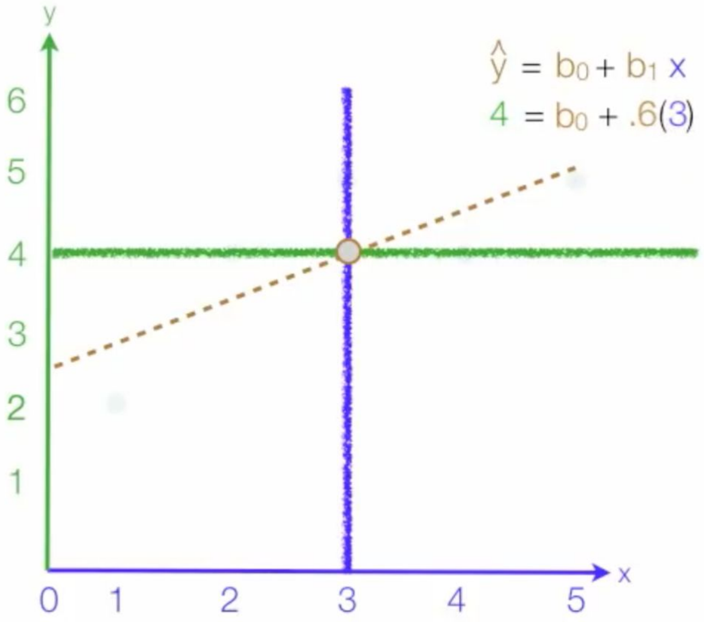
independent variable	dependent variable
x	y
1	2
2	4
3	5
4	4
5	5

mean 3 4



independent variable	dependent variable
x	y
1	2
2	4
3	5
4	4
5	5

mean



	x	y	$x - \bar{x}$	$y - \bar{y}$	$(x - \bar{x})^2$	$(x - \bar{x})(y - \bar{y})$
	1	2	-2	-2	4	4
	2	4	-1	0	1	0
	3	5	0	1	0	0
	4	4	1	0	1	0
	5	5	2	1	4	2
					10	6
mean	3	4				

$$4 = b_0 + .6(3)$$

$$b_1 = \frac{6}{10} = .6 = \frac{\Sigma (x - \bar{x})(y - \bar{y})}{\Sigma (x - \bar{x})^2}$$

Using Statistical Method

$$m = \frac{\sum_{i=1}^{n}(x_i - \overline{X})(y_i - \overline{Y})}{\sum_{i=1}^{n}(x_i - \overline{X})^2}$$

$$b = \overline{Y} - m\overline{X}$$

X^ is mean of X values , Y^ mean of y values

Multiple Linear Regression!

Complex and more applicable in real.

Tip

Multiple Input Variables.

One Output Variable.

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \ldots + \beta_n x_n$$

Where:

- y is the response
- β values are called the **model coefficients**. These values are "learned" during the model fitting/training step.
- $\beta 0$ is the intercept
- $\beta 1$ is the coefficient for $X1$ (the first feature)
- βn is the coefficient for Xn (the nth feature)

How to calculate the coefficients of Multiple Linear Regression?

Using Ordinary Least Square (OLS Method)

(With the help of Matrix and Linear Algebra)

Quick Revision of Matrix and Linear Algebra!

$$\mathbf{y} = \mathbf{X}\boldsymbol{\beta} + \boldsymbol{\epsilon}$$

where

$$\mathbf{y} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix} \quad \mathbf{X} = \begin{bmatrix} 1 & x_{11} & x_{12} & \cdots & x_{1k} \\ 1 & x_{21} & x_{22} & \cdots & x_{2k} \\ \vdots & \vdots & \vdots & & \vdots \\ 1 & x_{n1} & x_{n2} & \cdots & x_{nk} \end{bmatrix} \quad \boldsymbol{\beta} = \begin{bmatrix} \beta_0 \\ \beta_1 \\ \vdots \\ \beta_k \end{bmatrix} \quad \text{and} \quad \boldsymbol{\epsilon} = \begin{bmatrix} \epsilon_1 \\ \epsilon_2 \\ \vdots \\ \epsilon_n \end{bmatrix}$$

With squared-error loss the solution has a closed-form

$$
\begin{aligned}
\hat{\beta} &= (X^T X)^{-1} X^T Y \\
\hat{Y} &= X\hat{\beta} = X(X^T X)^{-1} X^T Y \\
&= \underbrace{H}_{\text{"Hat matrix"}} Y
\end{aligned}
$$

But how did we arrive at that equation!?

(And what on earth is a closed form solution?)

Since $$L = \sum_{i=1}^{n} \varepsilon_i^2 = \varepsilon' \varepsilon = (y - X\beta)'(y - X\beta)$$

$$\frac{\partial L}{\partial \beta} = 0 \Rightarrow X' X \hat{\beta} = X' y$$

Therefore $$\boxed{\hat{\boldsymbol{\beta}} = (\mathbf{X}'\mathbf{X})^{-1} \mathbf{X}'\mathbf{y}}$$ and $$\hat{y} = X \hat{\beta}$$

$$e = y - \hat{y}$$

Hopefully this scary looking method will make sense after we cover other topics first!

Residual and Cost Function (Loss Function vs Cost Function)

Residual = Predicted Value - Original Value

That is , Y’ - Y

And we want to make that less. Over all the data points.

Choice of the Cost Function (Loss Function)

So everything boils down to

...

Find the best Parameters that minimise the Loss Function

$$minimize \frac{1}{n} \sum_{i=1}^{n} (pred_i - y_i)^2$$

$$J = \frac{1}{n} \sum_{i=1}^{n} (pred_i - y_i)^2$$

Minimization and Cost Function

And why use the square of the error in the loss function?

(Why not absolute value for example!)

Quick Revision of Derivative and Calculus!

With squared-error loss the solution has a closed-form

$$\begin{aligned}
\hat{\beta} &= (X^T X)^{-1} X^T Y \\
\hat{Y} &= X\hat{\beta} = X(X^T X)^{-1} X^T Y \\
&= HY
\end{aligned}$$

"Hat matrix"

Problem with Matrix Method!

(They are not much used in Machine Learning Algorithms!)

Cons of Matrix Method

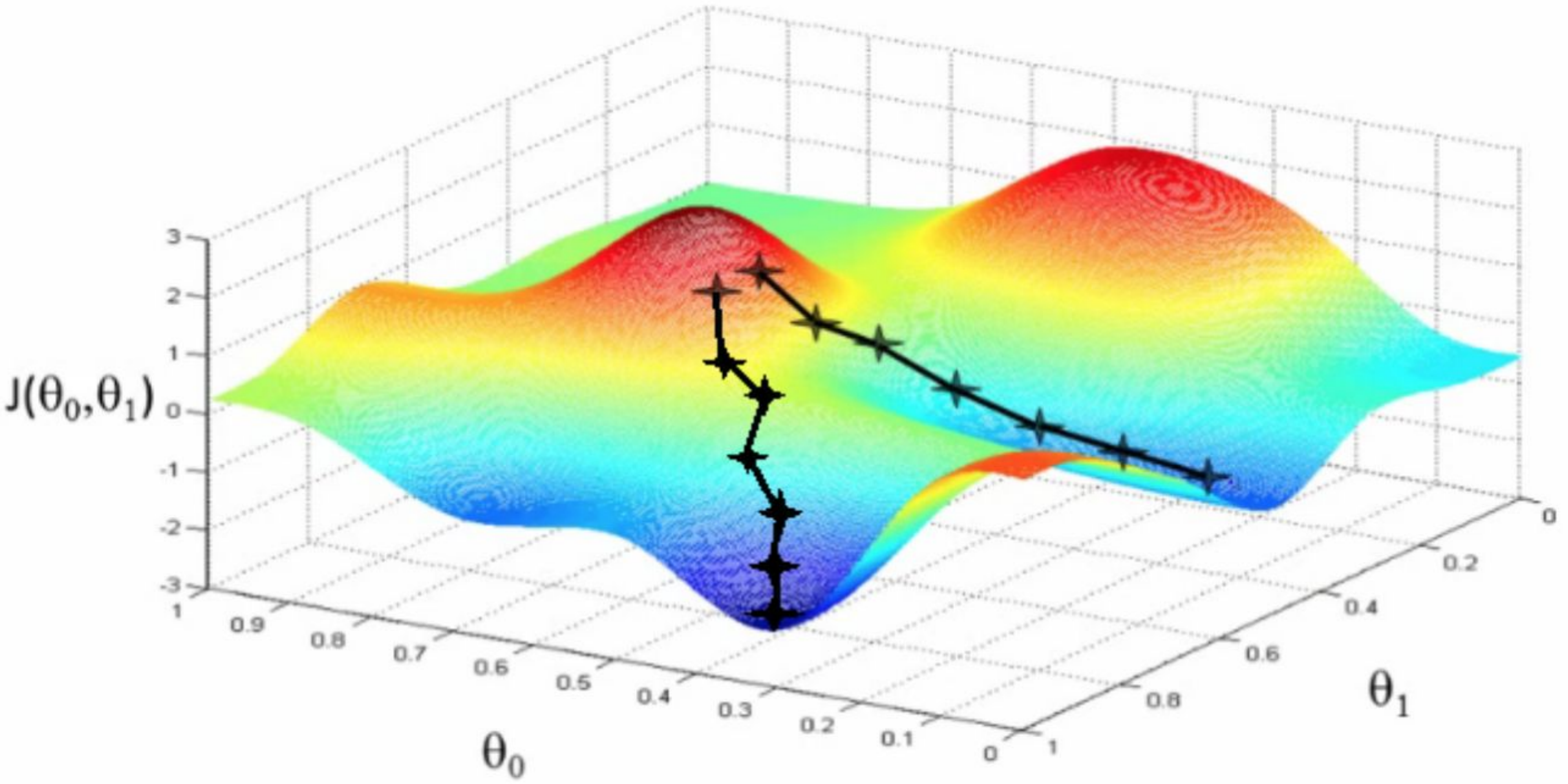
1. Memory issue to store the whole Data Matrix for huge data.
2. Inverse of a big matrix is computationally expensive.

One Small Twist about Matrix Method

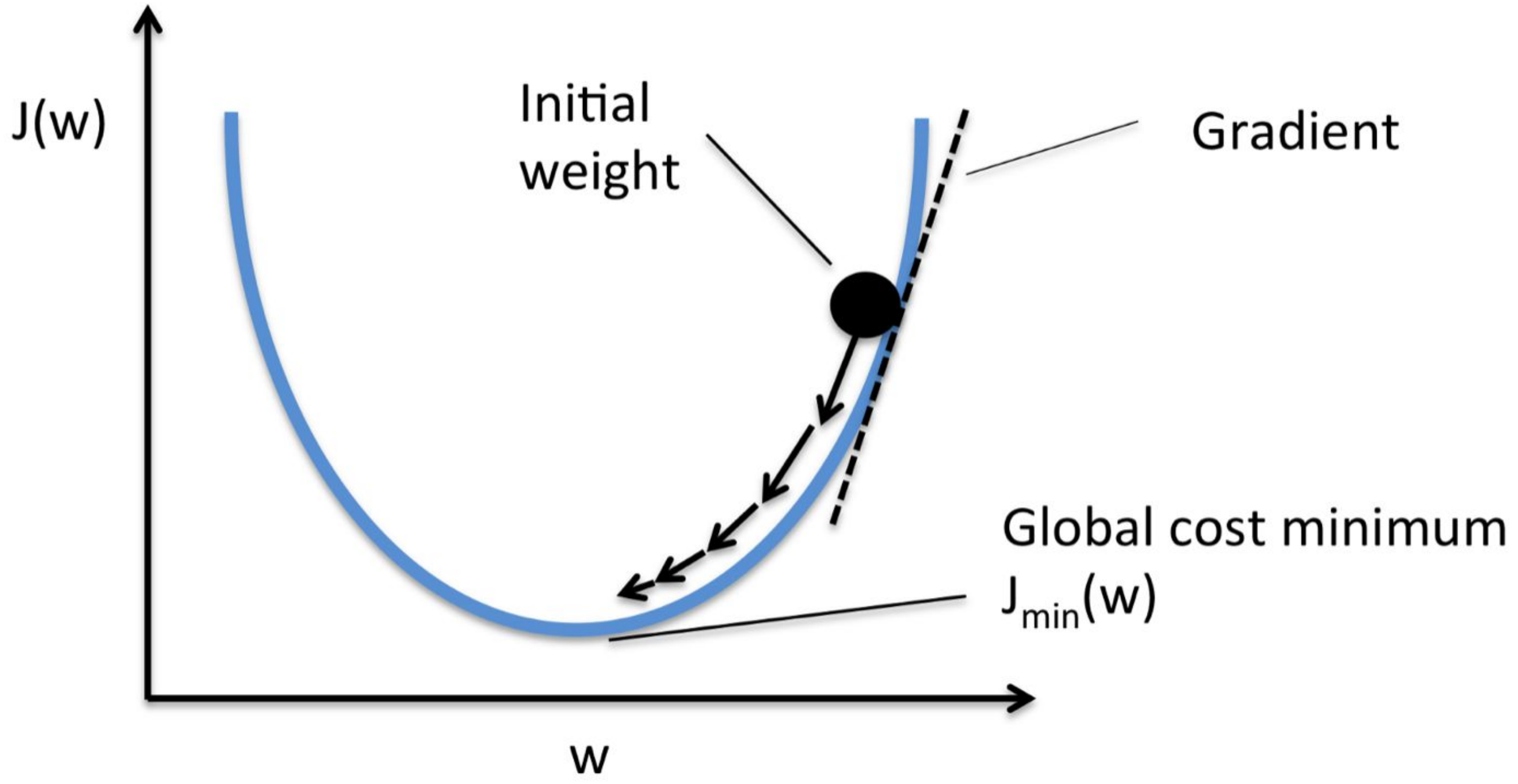
1. Inverse is not calculated internally to calculate the coefficients.
2. QR Decomposition or Singular Value Decomposition (SVD) is used.

Hence comes in our saviour!

Gradient Descent Algorithm



$J(\theta_0, \theta_1)$
3
2
1
0
-1
-2
-3
1
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
θ_0
0
0.2
0.4
0.6
0.8
1
θ_1



J(w)
Initial weight
Gradient
Global cost minimum
$J_{min}(w)$
w

Two important deciding factors

1. Your initial point (Where you start)
2. The speed at which you do down (Speed of Steps)

Learning Rate

$$\Theta_{n+1} = \Theta_n - \alpha \boxed{\frac{\partial}{\partial \Theta_n} J(\Theta_n)}$$

New Position

Old Position

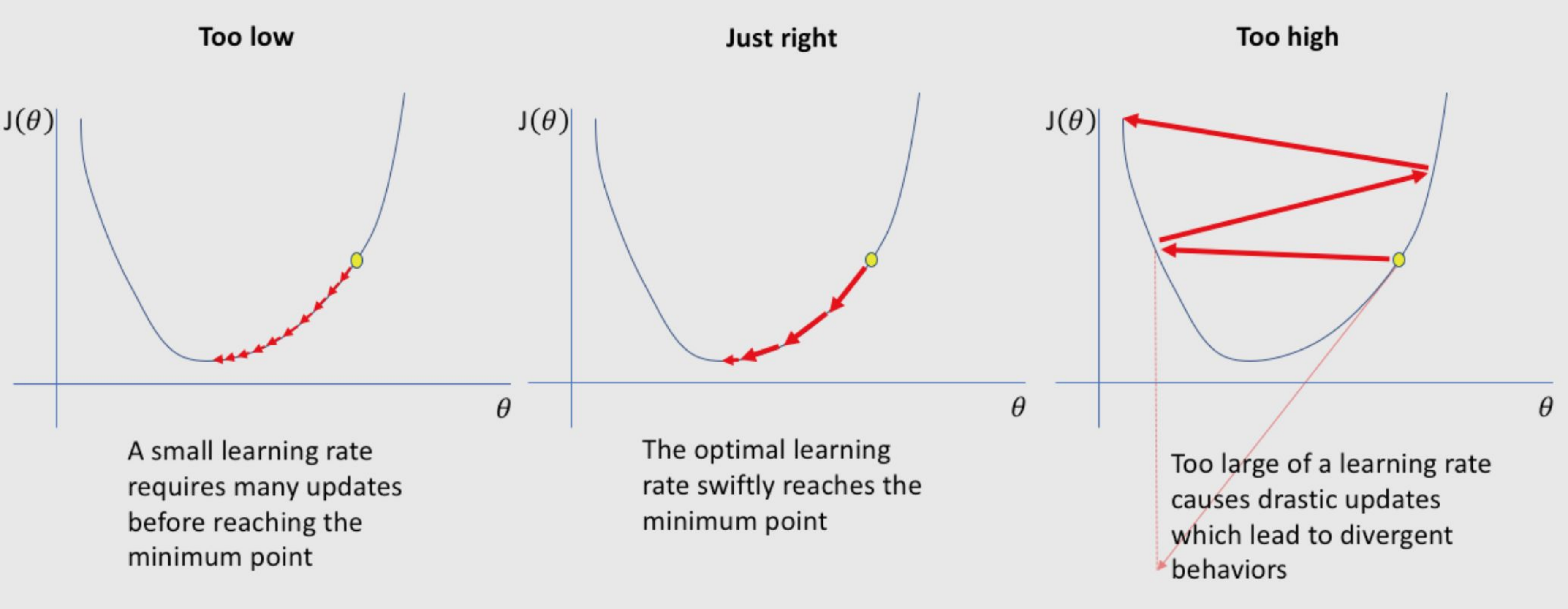
Gradient of function J for Θ_n

Repeat until convergence {

$$\theta_1 \leftarrow \theta_1 - \alpha \frac{\partial}{\partial \theta_1} \left(\frac{1}{2m} \sum_{i=1}^{m} (h(x_i) - y_i)^2 \right)$$

$$\theta_2 \leftarrow \theta_2 - \alpha \frac{\partial}{\partial \theta_2} \left(\frac{1}{2m} \sum_{i=1}^{m} (h(x_i) - y_i)^2 \right)$$

}


Too low
Just right
Too high
J(θ)
θ
J(θ)
θ
J(θ)
θ
A small learning rate requires many updates before reaching the minimum point
The optimal learning rate swiftly reaches the minimum point
Too large of a learning rate causes drastic updates which lead to divergent behaviors

Memo for Linear Regression!

Loss function is always convex.

(Means minimum point is at 0 derivative.)

Types of Gradient Descent

1. Batch Gradient Descent
2. Stochastic Gradient Descent
3. Mini Batch Gradient Descent

Batch vs Stochastic Gradient Descent

Whole dataset used

Whole data used for calculating cost function.

Single data used.

Randomly selected single data used for calculating cost function

Batch Gradient Descent

Stochastic Gradient Descent

Longer Conversion Rate

Not suitable for huge dataset.

Faster Conversion Rate

But also little noisy compared with batch.

Quick Rewind of Linear Regression

Simple Linear Regression

- One Input Variable
- Using Statistical Method

OLS Multiple Regression

- Multiple Input Variables
- Using Matrix Method

Gradient Descent Algorithm

- Multiple Input Variables
- Using Derivative Method

Maybe that scary looking method makes more sense now.

Since $$L = \sum_{i=1}^{n} \varepsilon_i^2 = \varepsilon' \varepsilon = (y - X\beta)'(y - X\beta)$$

$$\frac{\partial L}{\partial \beta} = 0 \Rightarrow X' X \hat{\beta} = X' y$$

Therefore $$\boxed{\hat{\boldsymbol{\beta}} = (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{y}}$$ and $$\hat{y} = X \hat{\beta}$$

$$e = y - \hat{y}$$

End Note!

Linear Regression is the most used machine learning method of all.

- ➔ **Explainable Model**
- ➔ **Building block of complex models.**
- ➔ **Building block of machine learning and deep learning models.**
- ➔ **Used in social science, research and other fields also equally.**