Introduction to Data Science

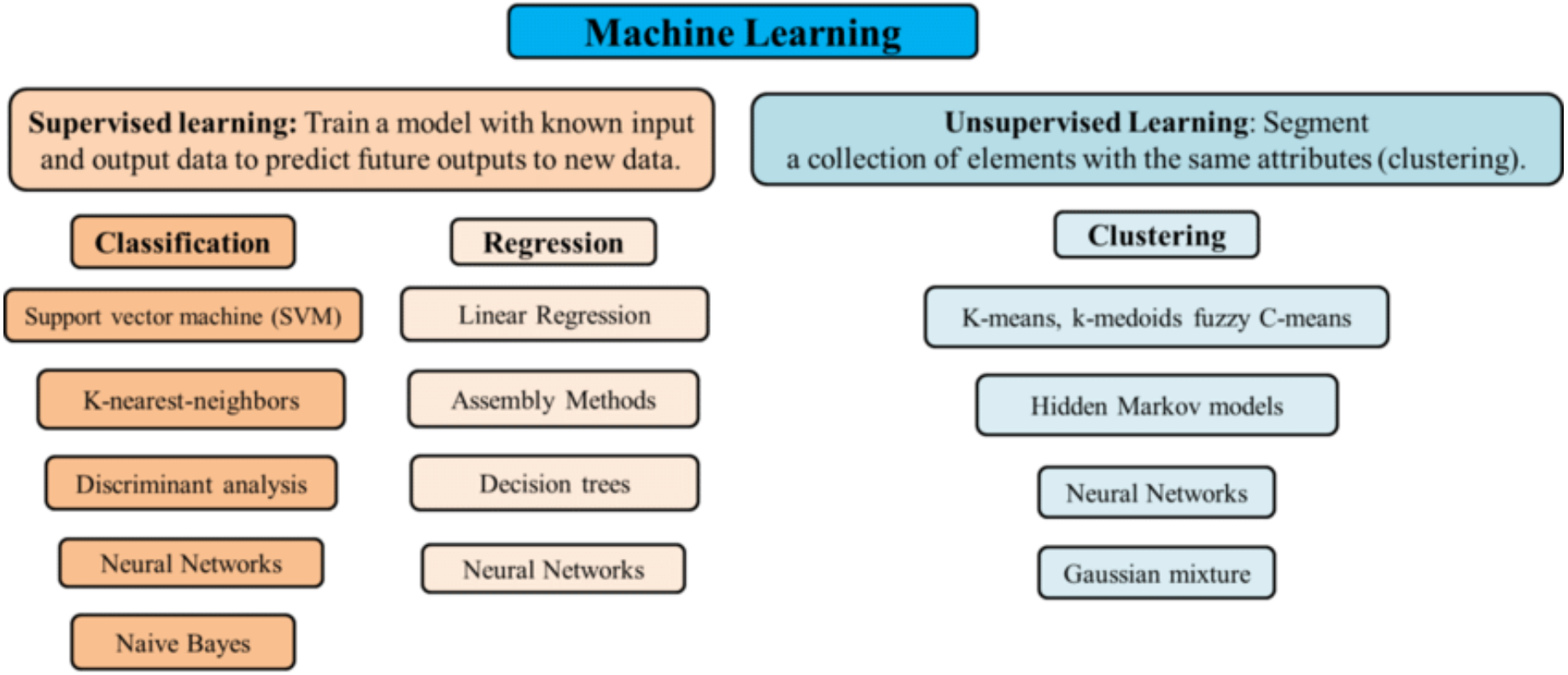
Dr. Irfan Yousuf
Department of Computer Science (New Campus)
UET, Lahore

(Lecture # 16; October 25, 2022)

Outline

- Linear Regression with Gradient Descent

Machine Learning Algorithms



Supervised Machine Learning Algorithms

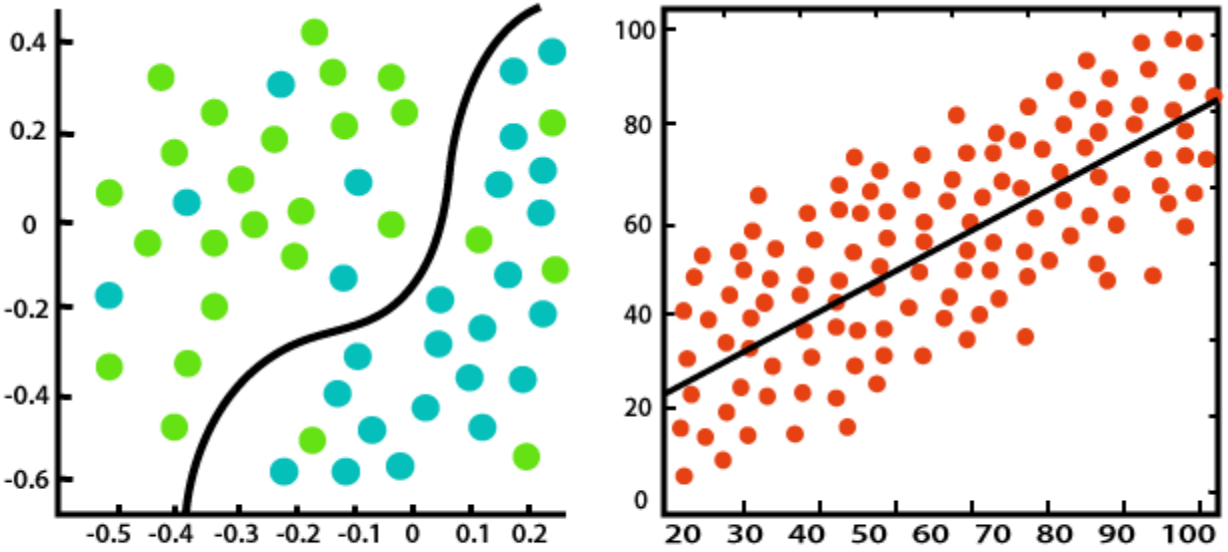
- Supervised learning is where you have input variables (x) and an output variable (Y) and you use an algorithm to learn the mapping function from the input to the output.

$$Y = f(X)$$

- The goal is to approximate the mapping function so well that when you have new input data (x) that you can predict the output variables (Y) for that data.
- The process of an algorithm learning from the training dataset can be thought of as a teacher supervising the learning process. We know the correct answers, the algorithm iteratively makes predictions on the training data and is corrected by the teacher. Learning stops when the algorithm achieves an acceptable level of performance.

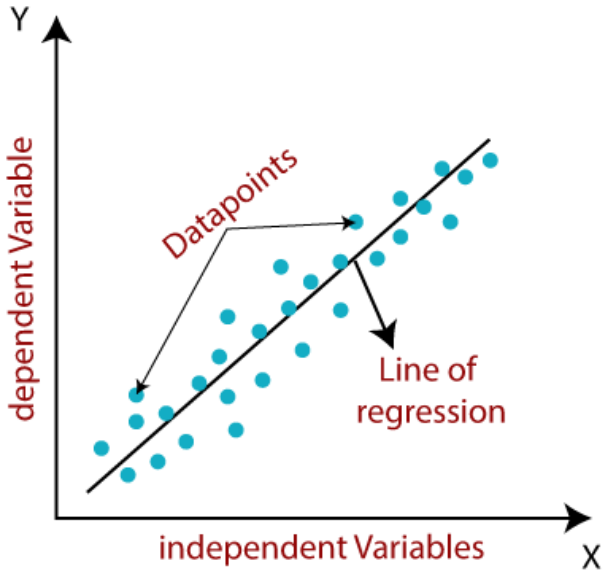
Supervised Machine Learning Algorithms

- **Classification:** A classification problem is when the output variable is a category, such as “red” or “blue” or “disease” and “no disease”.
- **Regression:** A regression problem is when the output variable is a real value, such as “age” or “weight”.



Linear Regression

- It is a statistical method that is used for predictive analysis. Linear regression makes predictions for continuous/real or numeric variables such as sales, salary, age, price, etc.
- Linear regression algorithm shows a linear relationship between a dependent (y) and one or more independent (y) variables, hence called as linear regression.



Linear Regression

The formula for a simple linear regression is:

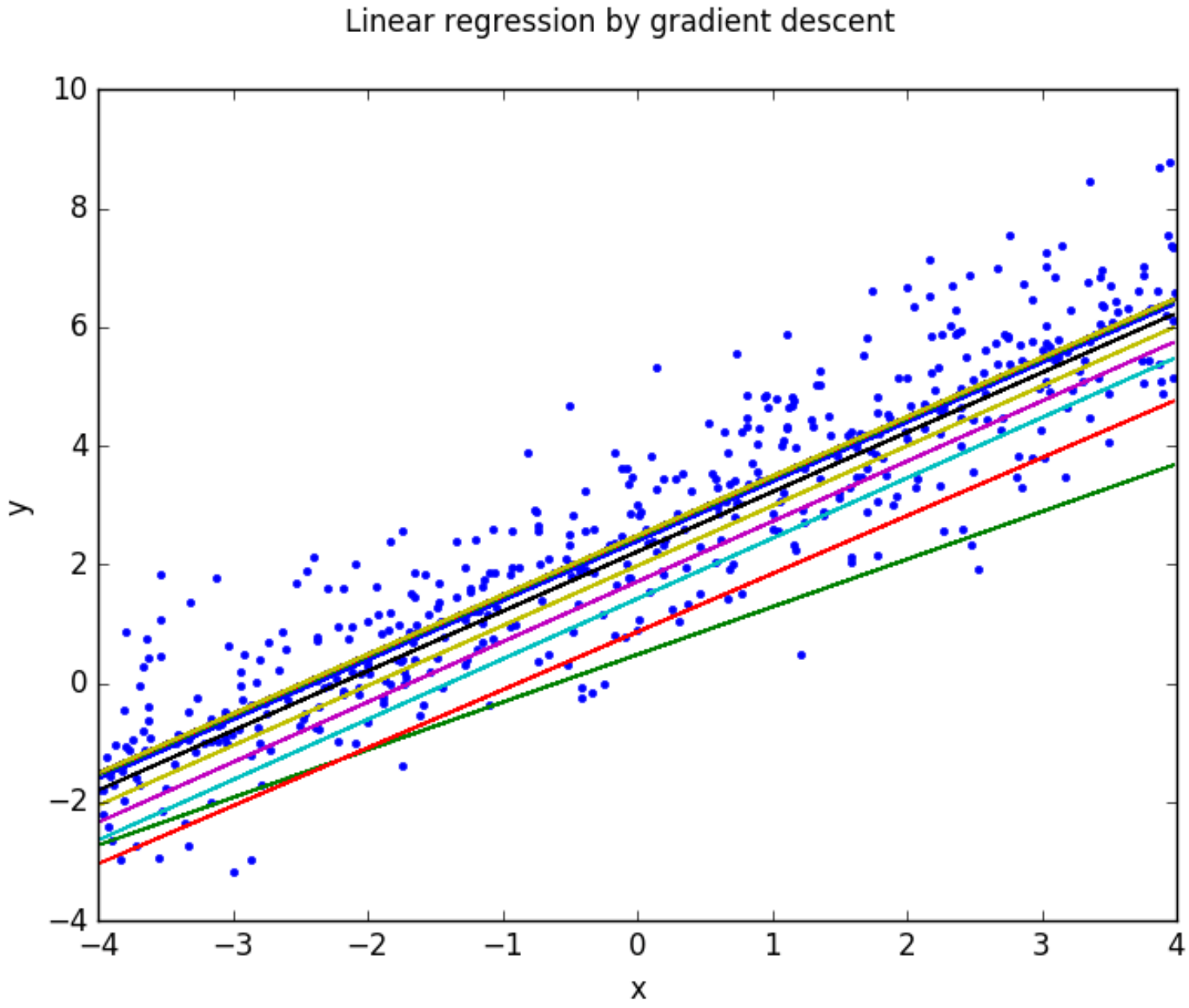
$$y = \beta_0 + \beta_1 X + \varepsilon$$

- **y** is the predicted value of the dependent variable (**y**) for any given value of the independent variable (**x**).
- B_0 is the **intercept**, the predicted value of **y** when the **x** is 0.
- B_1 is the regression coefficient – how much we expect **y** to change as **x** increases.
- **x** is the independent variable (the variable we expect is influencing **y**).
- **e** is the **error** of the estimate, or how much variation there is in our estimate of the regression coefficient.

Linear Regression

- When working with linear regression, our main goal is to find the best fit line that means the error between predicted values and actual values should be minimized. The best fit line will have the least error.

- The different values for weights or the coefficient of lines (β_0, β_1) gives a different line of regression, so we need to calculate the best values for β_0 and β_1 to find the best fit line, so to calculate this we use **cost function**.


Linear regression by gradient descent
10
8
6
4
2
0
−2
−4
y
−4
−3
−2
−1
0
1
2
3
4
x

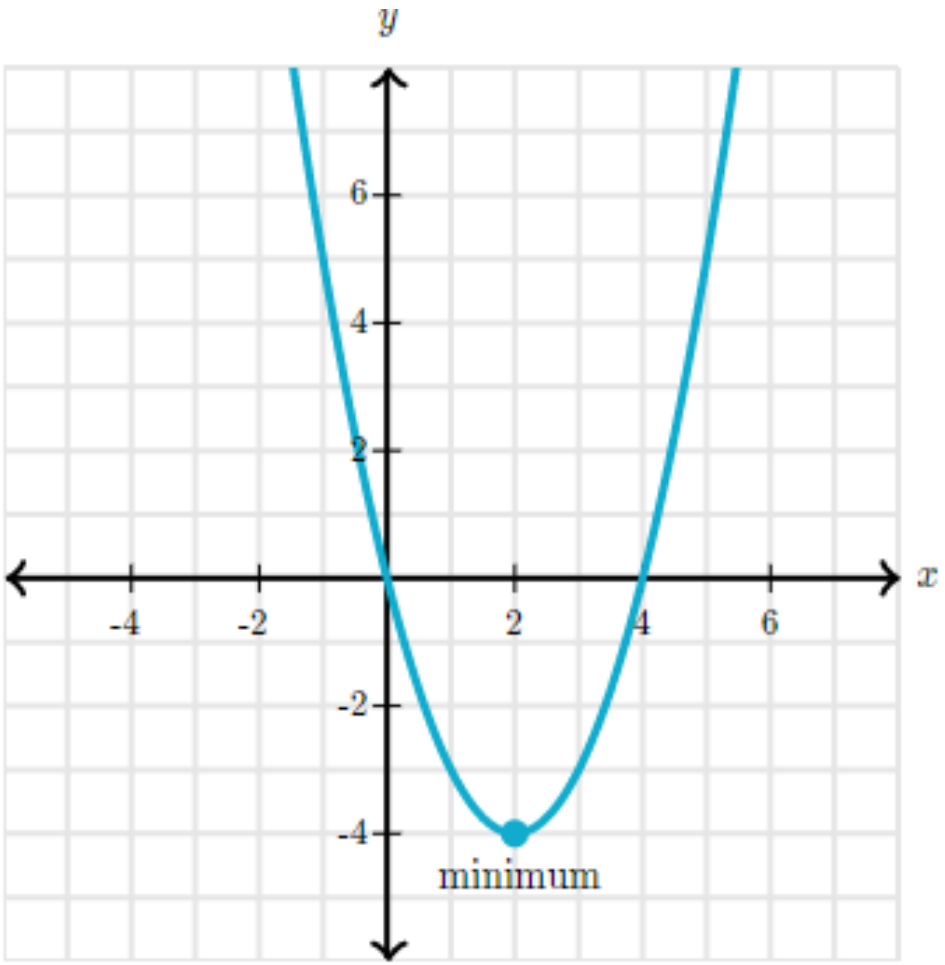
Gradient Descent

- Gradient descent is an algorithm that **numerically estimates** where a function outputs its lowest values.

- That means it finds local minima, but not by setting f=0. Instead of finding minima by manipulating symbols, gradient descent approximates the solution with numbers.

Gradient Descent

- $f(x) = x^2-4x$



Gradient Descent

$$x_{n+1} = x_n - \alpha \nabla F(x_n)$$

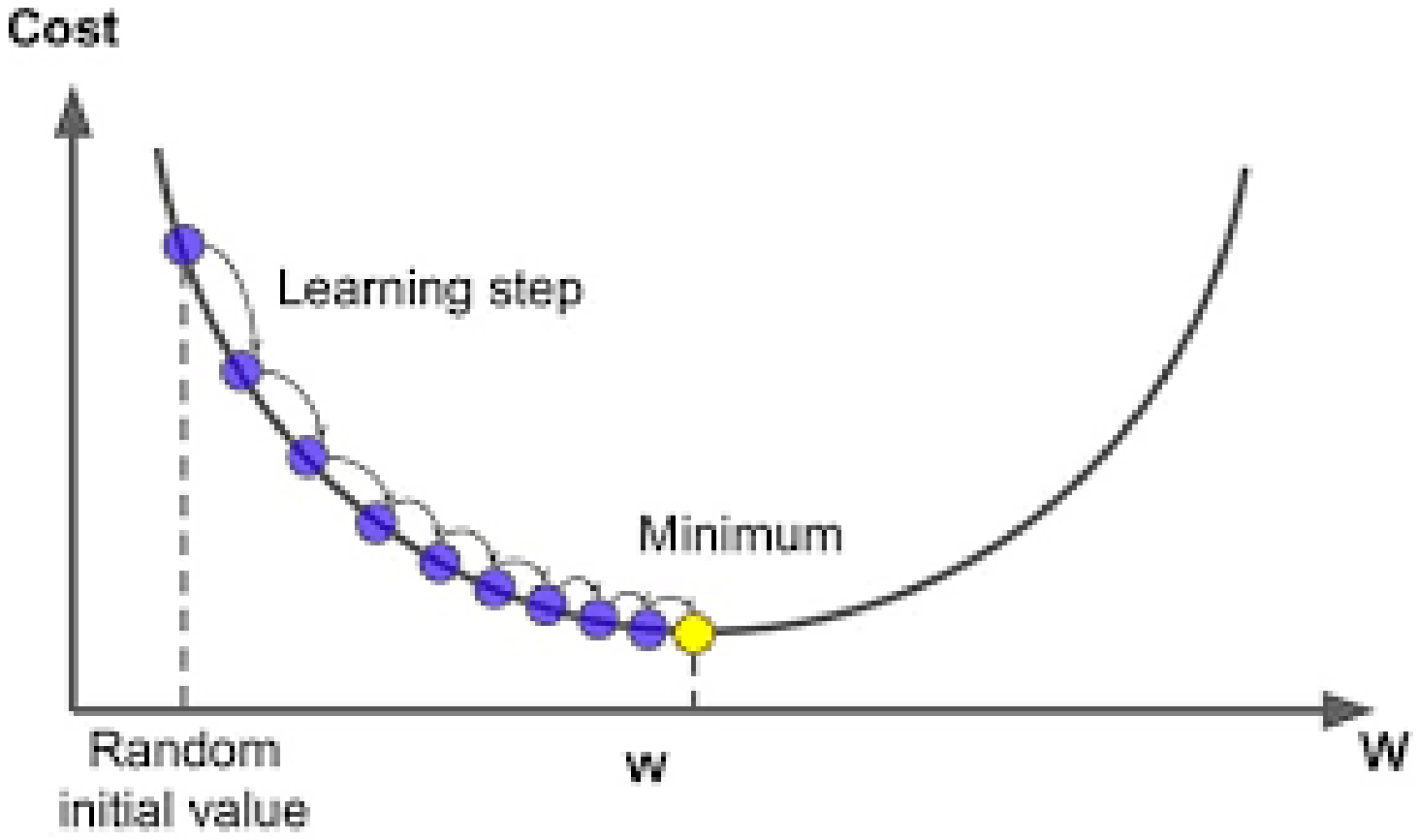
$$y = x^2$$

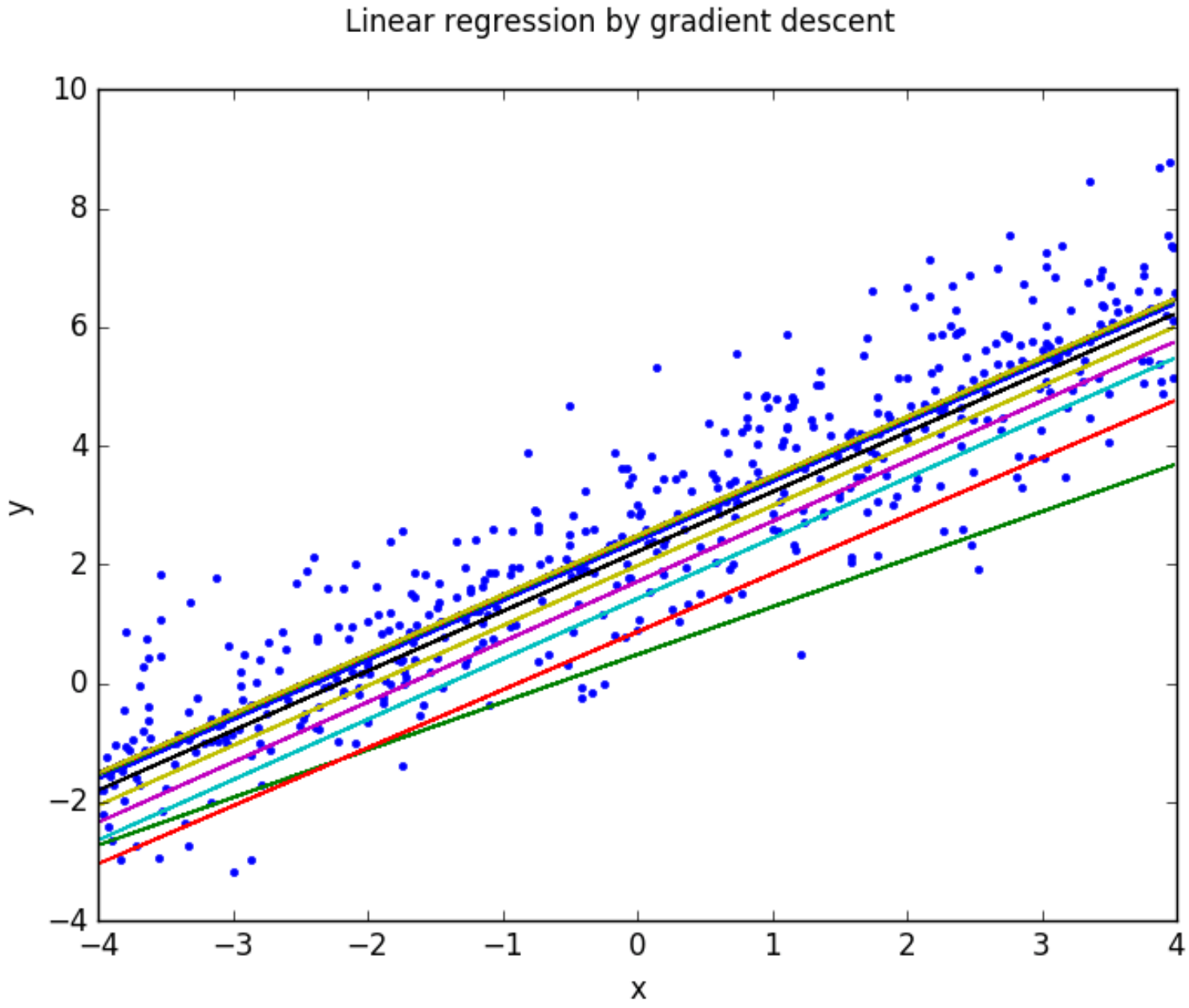
We calculate $\nabla F(x_n)$:

$$\nabla F(x_n) = \frac{\partial F(x_n)}{\partial x_n} = 2x_n$$

$$\Rightarrow x_{n+1} = x_n - \alpha \cdot 2x_n$$

Gradient Descent




Linear regression by gradient descent
10
8
6
4
2
0
−2
−4
y
−4
−3
−2
−1
0
1
2
3
4
x

Linear Regression: Cost Function

- The different values for weights or coefficient of lines (β_0, β_1) gives the different line of regression, and the cost function is used to estimate the values of the coefficient for the best fit line.

- Cost function optimizes the regression coefficients or weights. It measures how a linear regression model is performing.

- We can use the cost function to find the accuracy of the mapping function, which maps the input variable to the output variable.

Linear Regression: Cost Function

- For Linear Regression, we use the Mean Squared Error (MSE) cost function, which is the average of squared error occurred between the predicted values and actual values.

$$error = (guess - actual)$$

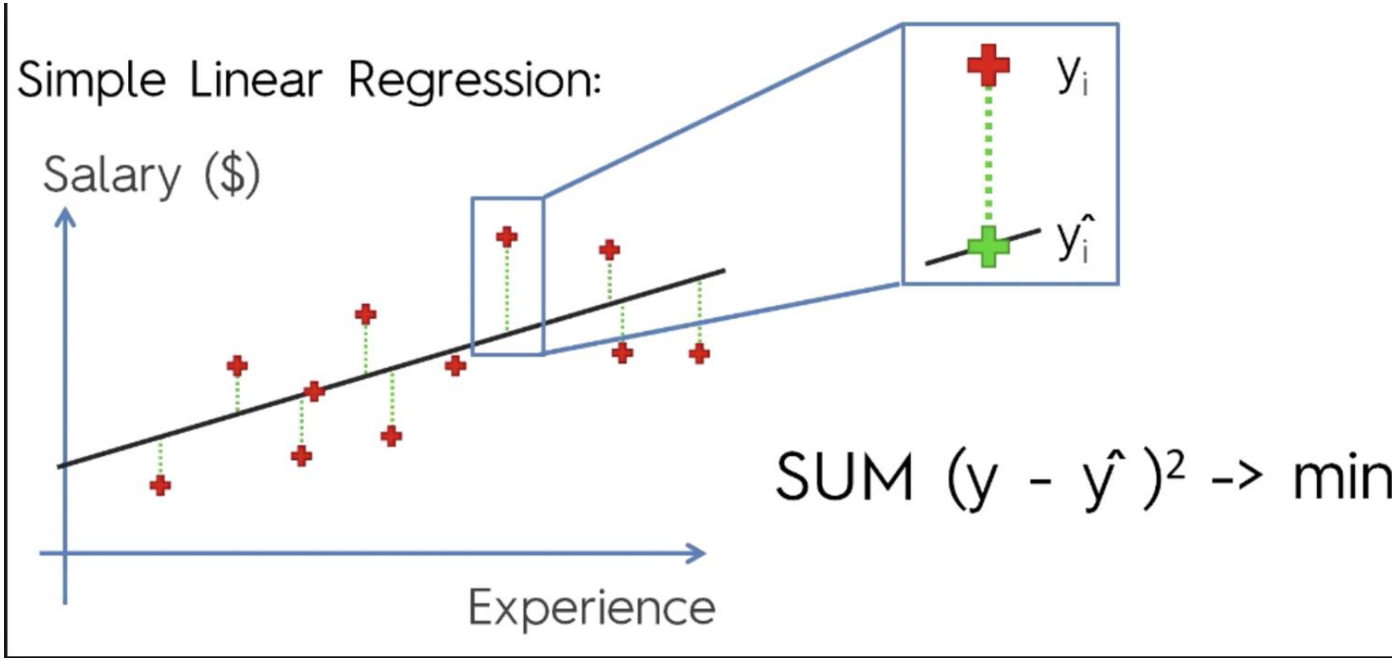
$$MSE = \frac{1}{n}\Sigma\underbrace{\left(y - \widehat{y}\right)}^{2}$$

The square of the difference between actual and predicted

Linear Regression: Cost Function

$$MSE = \frac{1}{n} \Sigma \underbrace{\left(y - \hat{y} \right)^2}$$

The square of the difference between actual and predicted



Linear Regression: Gradient Descent:

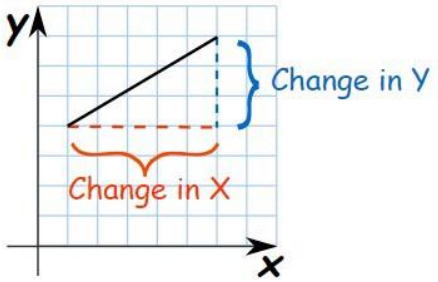
- Gradient descent is used to minimize the MSE by calculating the gradient of the cost function.
- A regression model uses gradient descent to update the coefficients of the line by reducing the cost function.
- It is done by a random selection of values of coefficient and then iteratively update the values to reach the minimum cost function.

- Gradient:
  - An inclined part of a road or railway; a slope.
  - An increase or decrease in the magnitude of a property (e.g. temperature, pressure, or concentration) observed in passing from one point or moment to another.

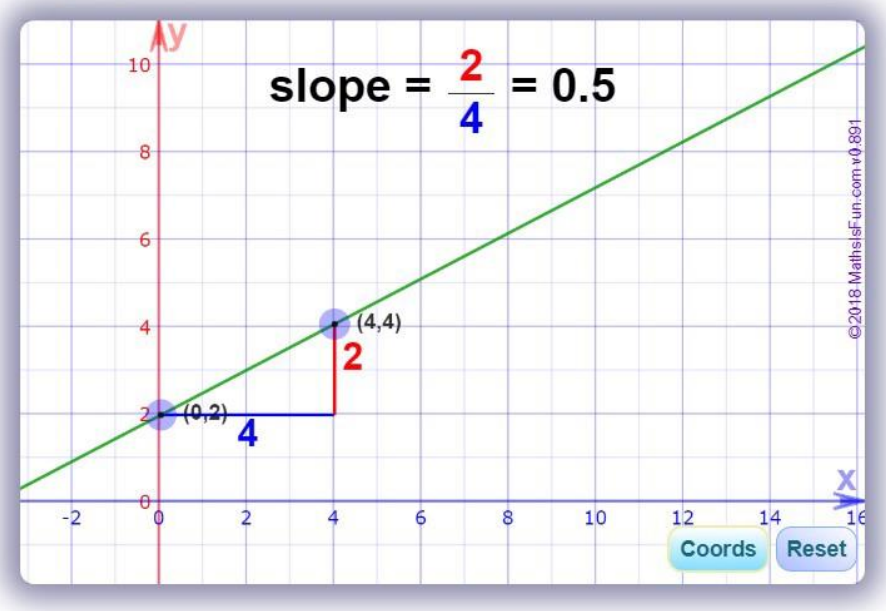
Linear Regression: Gradient Descent:

- Optimization algorithm that tweaks its parameters iteratively.

$$\text{Gradient} = \frac{\text{Change in Y}}{\text{Change in X}}$$

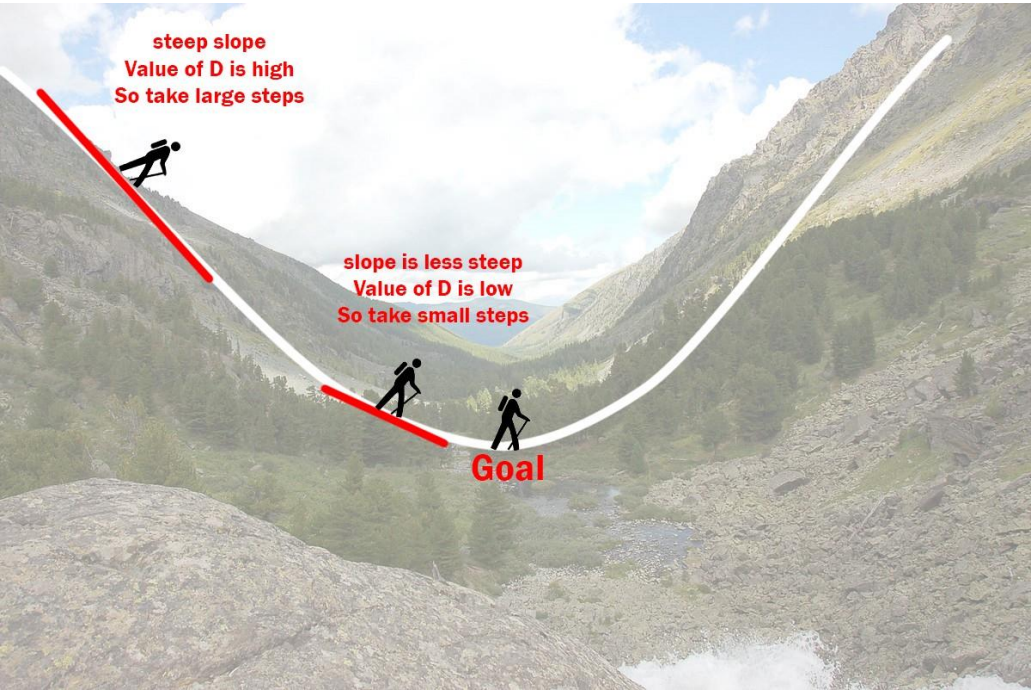


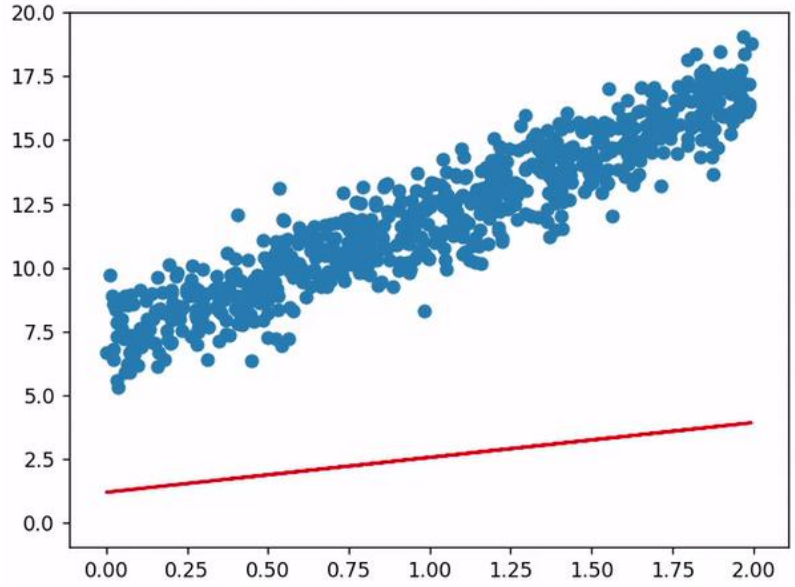
Have a play (drag the points):



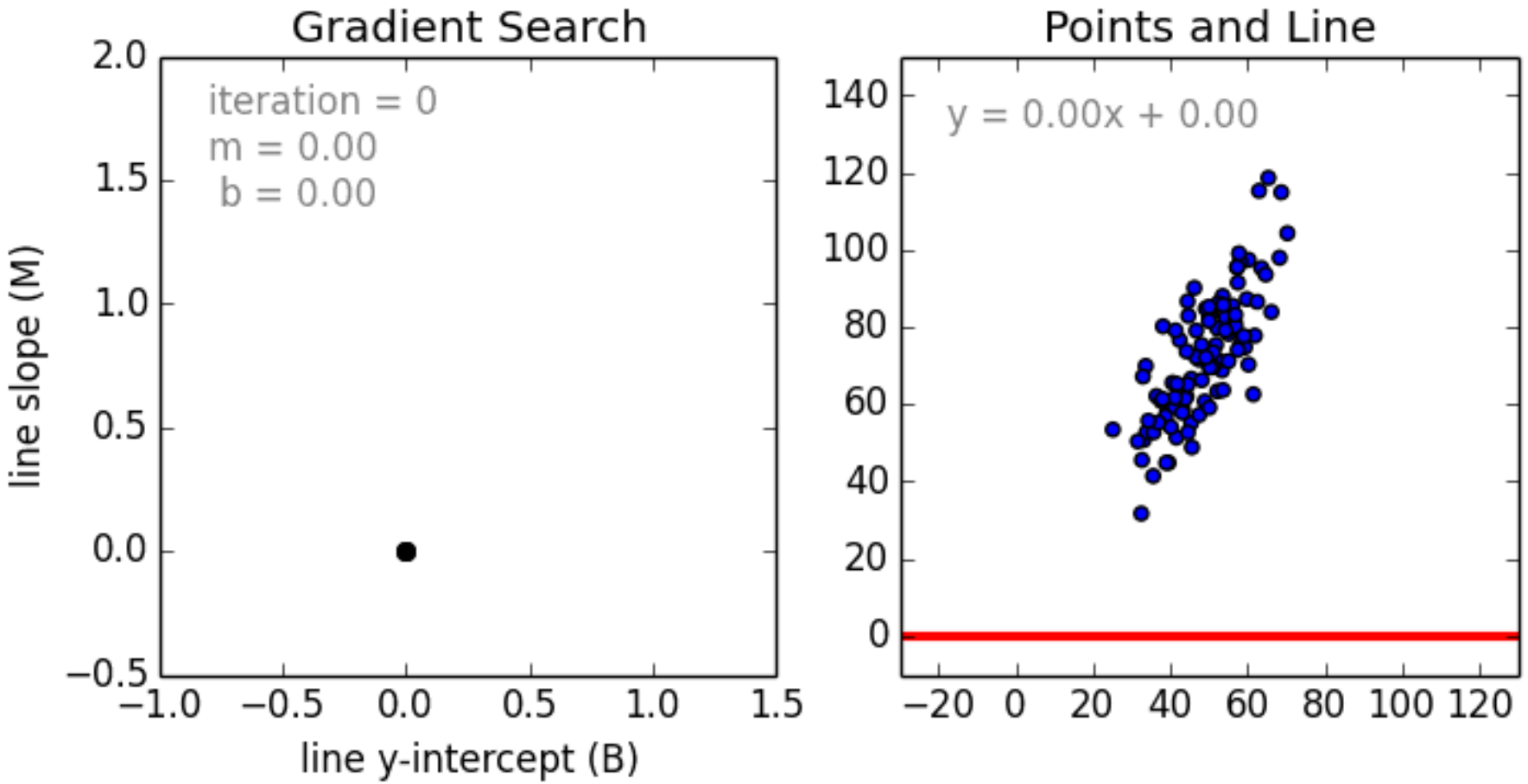
Linear Regression: Gradient Descent:

- Optimization algorithm that tweaks its parameters iteratively.





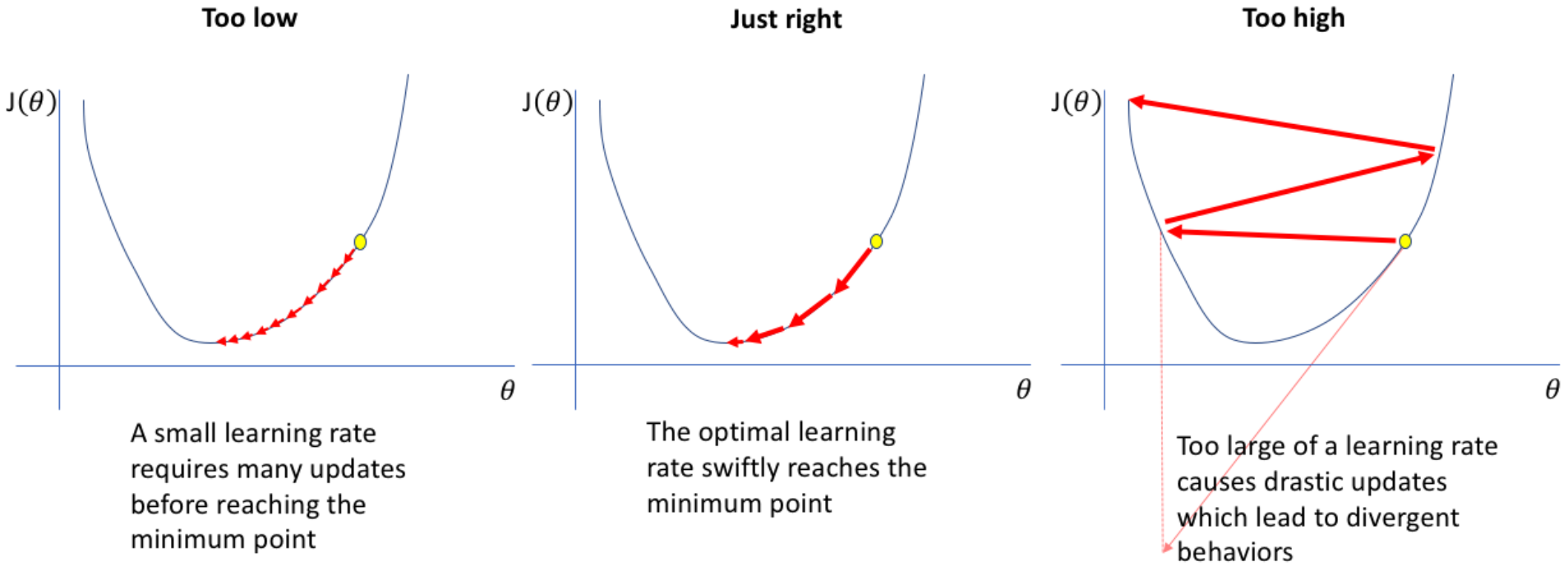
Linear Regression: Finding Gradient Descent:



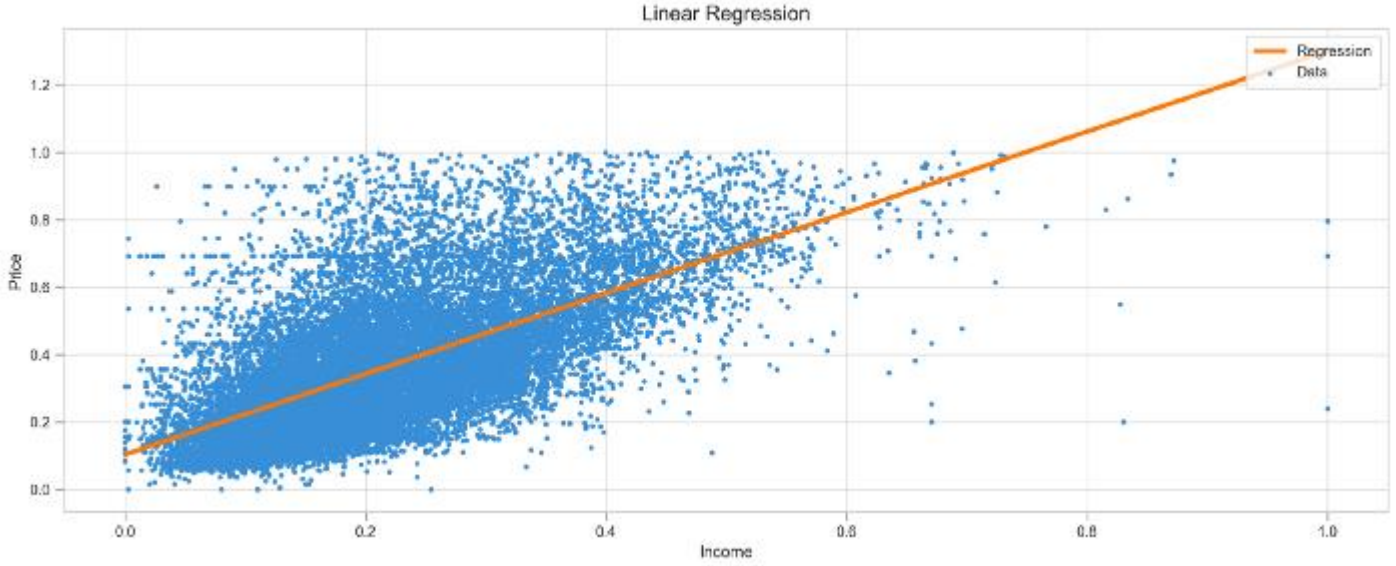
Source: https://raw.githubusercontent.com/mattnedrich/GradientDescentExample/master/gradient_descent_example.gif?source=post_page----------

Linear Regression: Gradient Descent: Learning Rate

- Optimization algorithm that tweaks its parameters iteratively.



Linear Regression: Finding Gradient Descent:



$$\hat{y} = mx + b$$

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 \quad \text{where} \quad \hat{y}_i = mx_i + b$$

Linear Regression: Finding Gradient Descent:

We use partial derivatives to find how each individual parameter affects MSE, so that's where word partial comes from. We take these derivatives with respect to m and b separately.

$$f(m, b) = \frac{1}{n} \sum_{i=1}^{n} (y_i - (mx_i + b))^2$$

$$\frac{\partial f}{\partial m} = \frac{1}{n} \sum_{i=1}^{n} -2x_i(y_i - (mx_i + b))$$

$$\frac{\partial f}{\partial b} = \frac{1}{n} \sum_{i=1}^{n} -2(y_i - (mx_i + b))$$

Linear Regression: Finding Gradient Descent:

$$D_m = \frac{\partial(Cost\ Function)}{\partial m} = \frac{\partial}{\partial m}\left(\frac{1}{n}\sum_{i=0}^{n}\left(y_i - y_{i\,pred}\right)^2\right)$$

$$= \frac{1}{n}\frac{\partial}{\partial m}\left(\sum_{i=0}^{n}\left(y_i - (mx_i + c)\right)^2\right)$$

$$= \frac{1}{n}\frac{\partial}{\partial m}\left(\sum_{i=0}^{n}\left(y_i^2 + m^2x_i^2 + c^2 + 2mx_ic - 2y_imx_i - 2y_ic\right)\right)$$

$$= \frac{-2}{n}\sum_{i=0}^{n} x_i\left(y_i - (mx_i + c)\right)$$

$$= \frac{-2}{n}\sum_{i=0}^{n} x_i\left(y_i - y_{i\,pred}\right)$$

Linear Regression: Finding Gradient Descent:

$$D_c = \frac{\partial(Cost\ Function)}{\partial c} = \frac{\partial}{\partial c}\left(\frac{1}{n}\sum_{i=0}^{n}\left(y_i - y_{i\,pred}\right)^2\right)$$

$$= \frac{1}{n}\frac{\partial}{\partial c}\left(\sum_{i=0}^{n}\left(y_i - (mx_i + c)\right)^2\right)$$

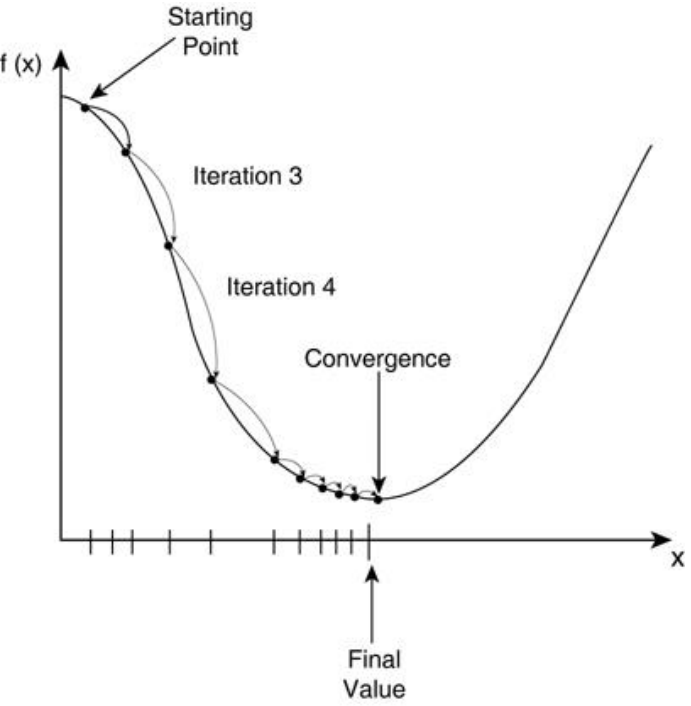
$$= \frac{1}{n}\frac{\partial}{\partial c}\left(\sum_{i=0}^{n}\left(y_i^2 + m^2x_i^2 + c^2 + 2mx_ic - 2y_imx_i - 2y_ic\right)\right)$$

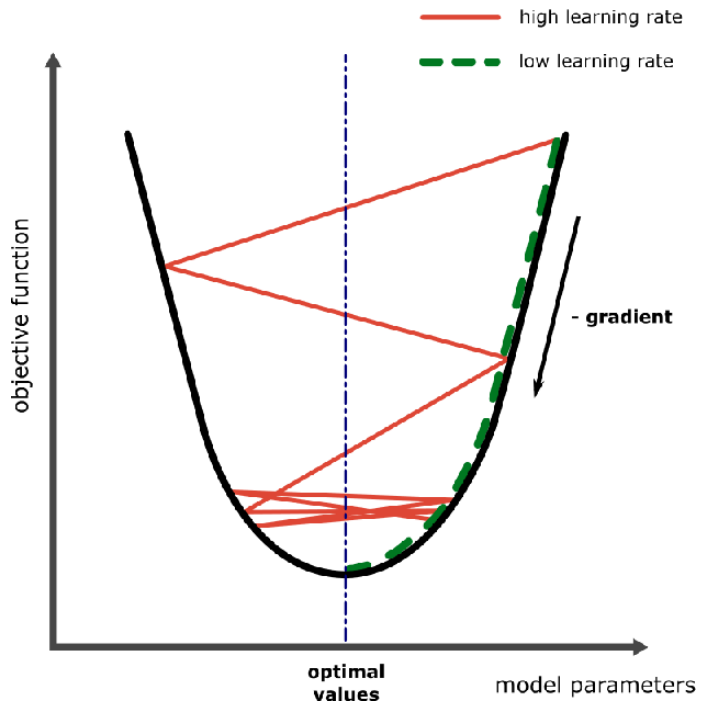
$$= \frac{-2}{n}\sum_{i=0}^{n}\left(y_i - (mx_i + c)\right)$$

$$\frac{-2}{n}\sum_{i=0}^{n}\left(y_i - y_{i\,pred}\right)$$

Linear Regression: Finding Gradient Descent:

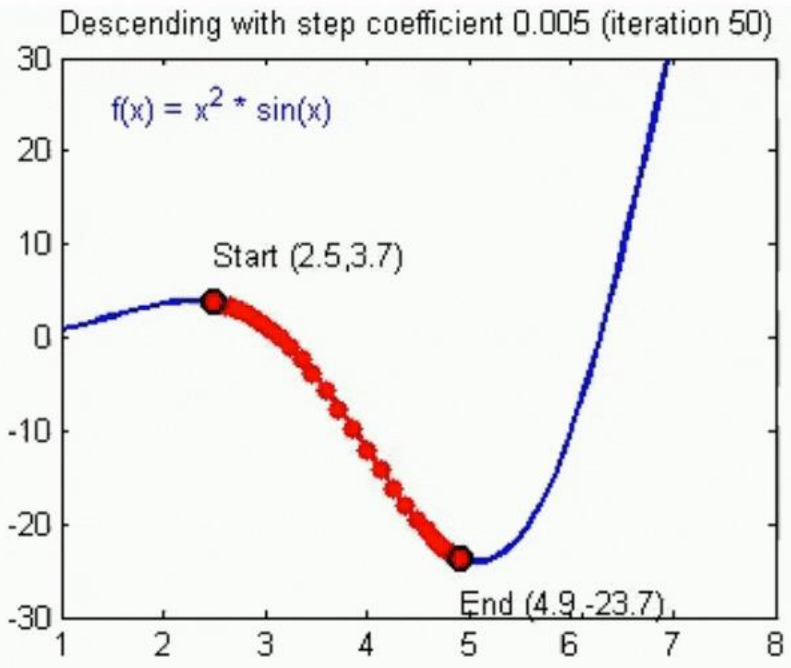
Convergence: Property of approaching a limit more and more closely as an argument (variable) of the function increases or decreases. For example, the function y = 1/x converges to zero as x increases.



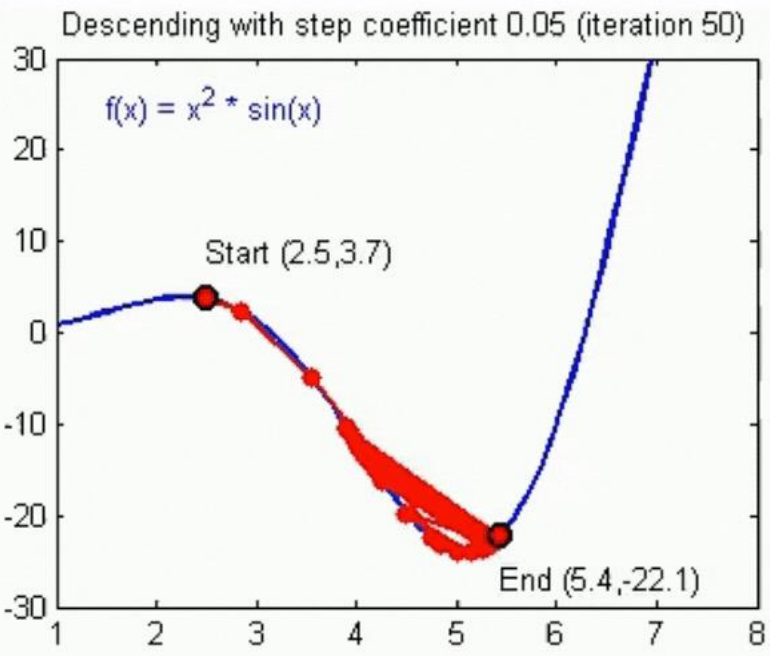


Linear Regression: Finding Gradient Descent:

Convergence



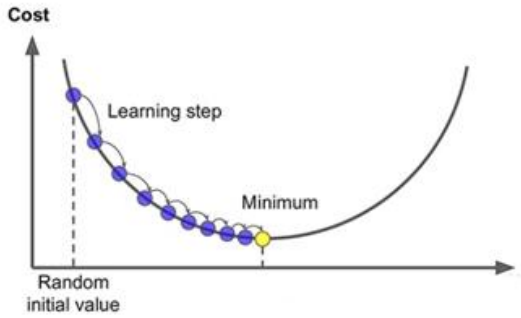
Divergence

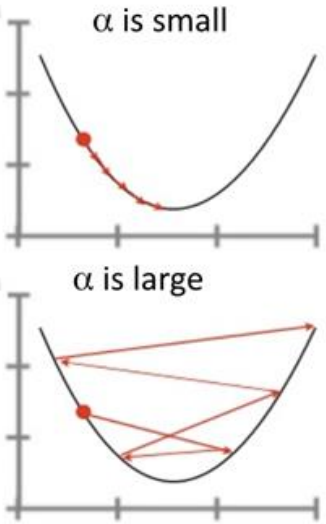


Linear Regression: Finding Gradient Descent:

$$\text{Repeat until converge}\{x := x - \alpha \nabla F(x)\}$$

- The notation := stands for overwriting the value on the left of := with values on the right of := .
- ∇ stands for the gradient of a function, which is the collection of all its partial derivatives into a vector (taking the first derivative of the function with respect to all possible x).
- α stands for the learning rate which is set manually.





$$X = X - lr * \frac{d}{dX} f(X)$$

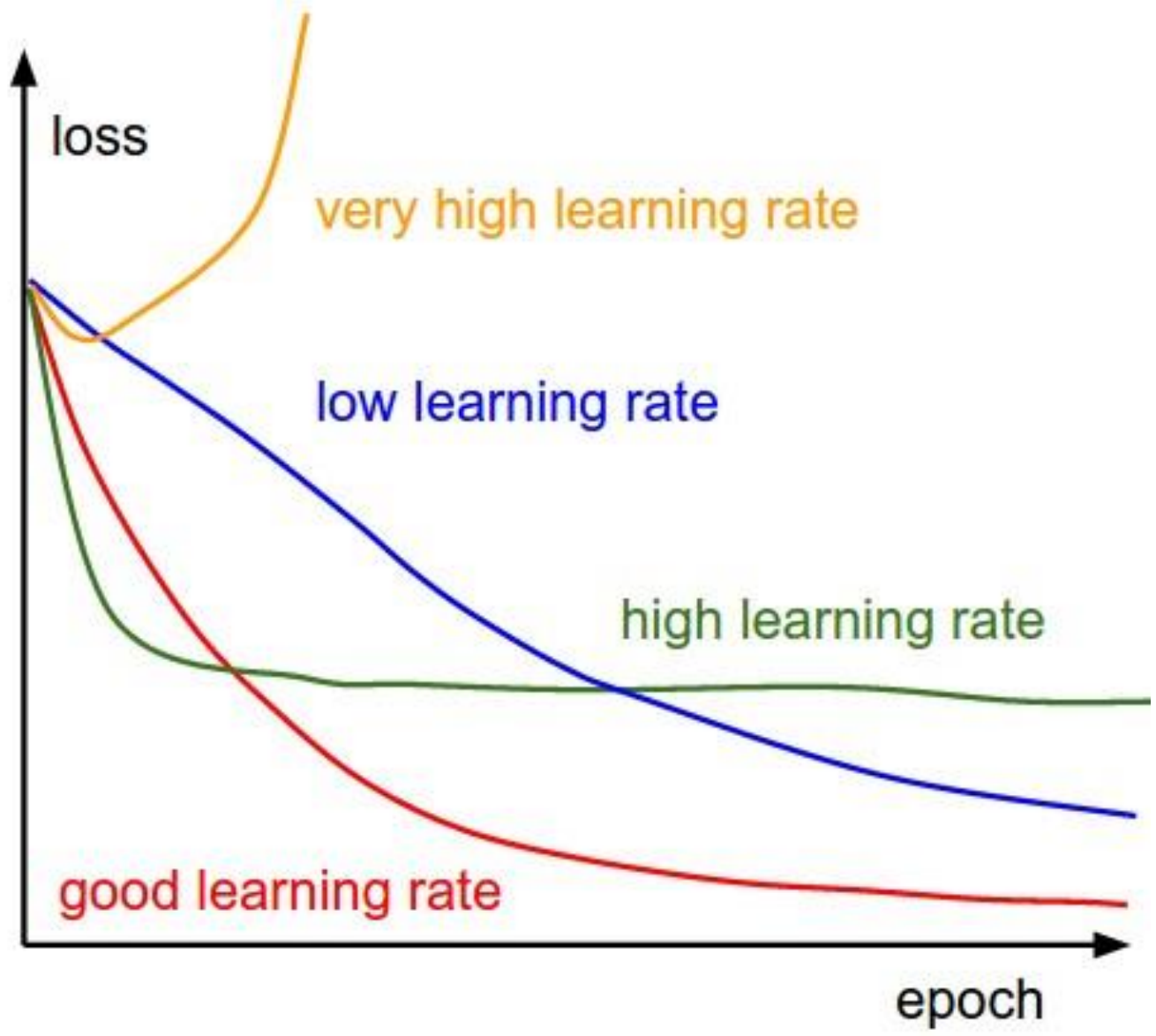
Where,

X = *input*

$F(X)$ = *output based on X*

lr = *learning rate*

Linear Regression: Learning Rate



Gradient Descent Algorithm

STEP 1: Take some random values for the coefficients m and b and calculate the MSE (cost function).

STEP 2: Calculate the partial derivatives of MSE with respect to m and b.

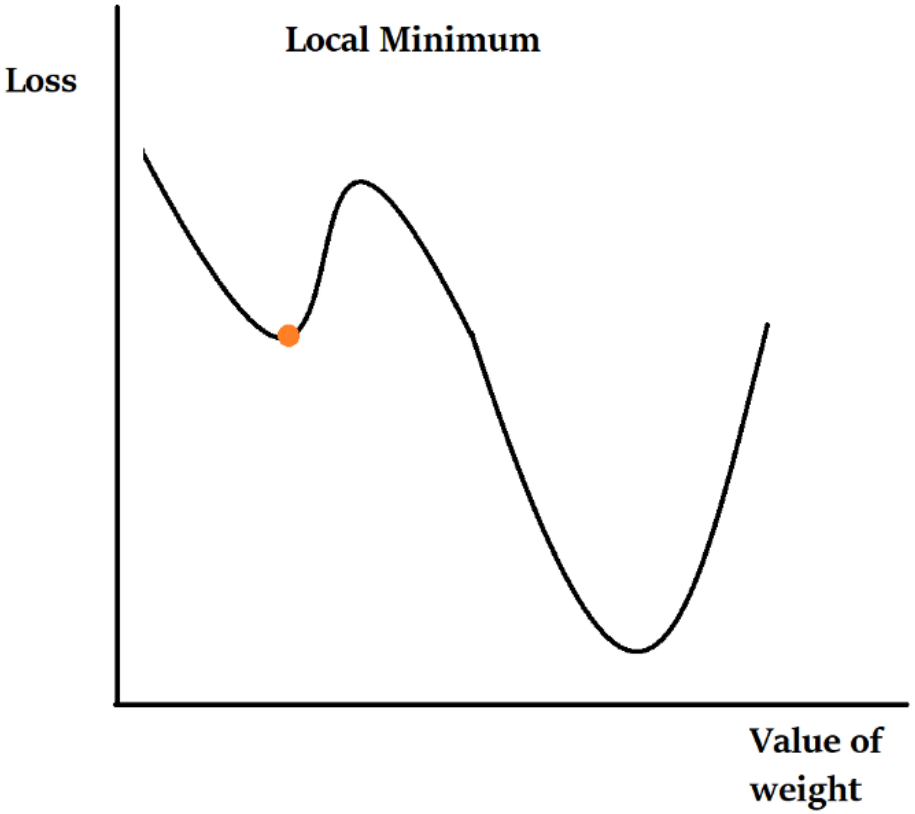
STEP 3: Set a value for the learning rate. And calculate the change in m and b using the following formula.

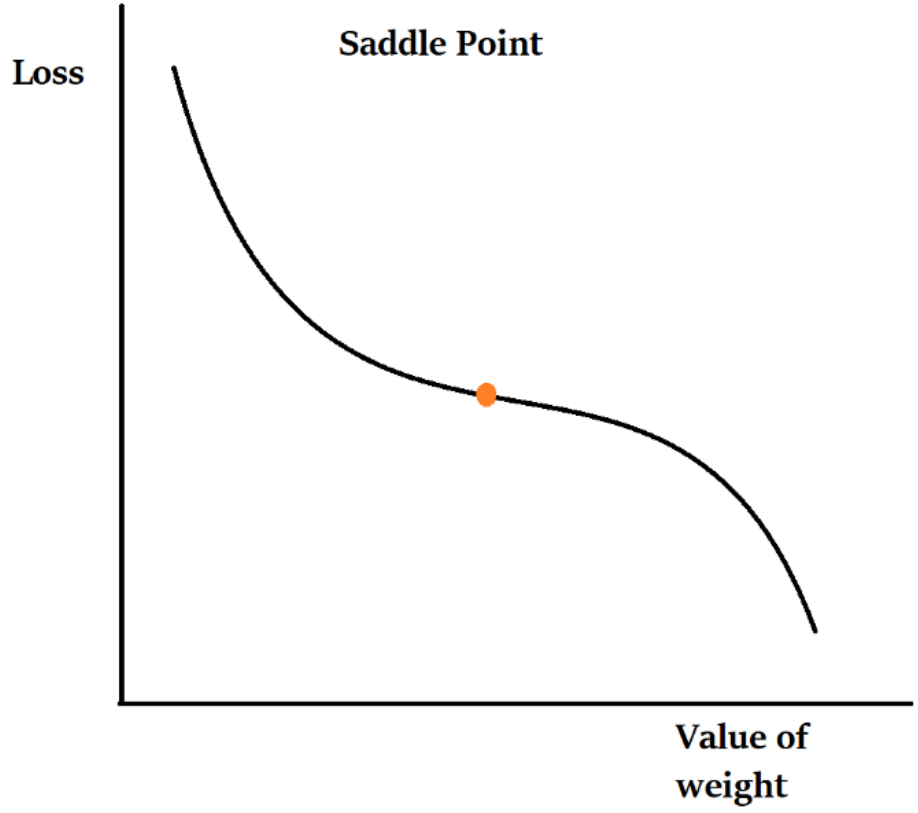
- $m = m - \text{learning rate} * \partial/\partial m = m - \alpha * \partial/\partial m$
- $b = b - \text{learning rate} * \partial/\partial b = b - \alpha * \partial/\partial b$

STEP 4: Use these values of m and b to calculate the new MSE.

STEP 5: Repeat steps 2, 3 and 4 until the changes in m and b do not significantly reduce the MSE (cost).

Linear Regression: Finding Gradient Descent:





Linear Regression: Gradient Descent:

Implementation in Python

Summary

- Linear Regression with Gradient Descent