Fifth Session

Alireza Moradi

Machine learning

Machine learning (ML)

- **Machine Learning** is the learning in which machine can learn by its own without being explicitly programmed.

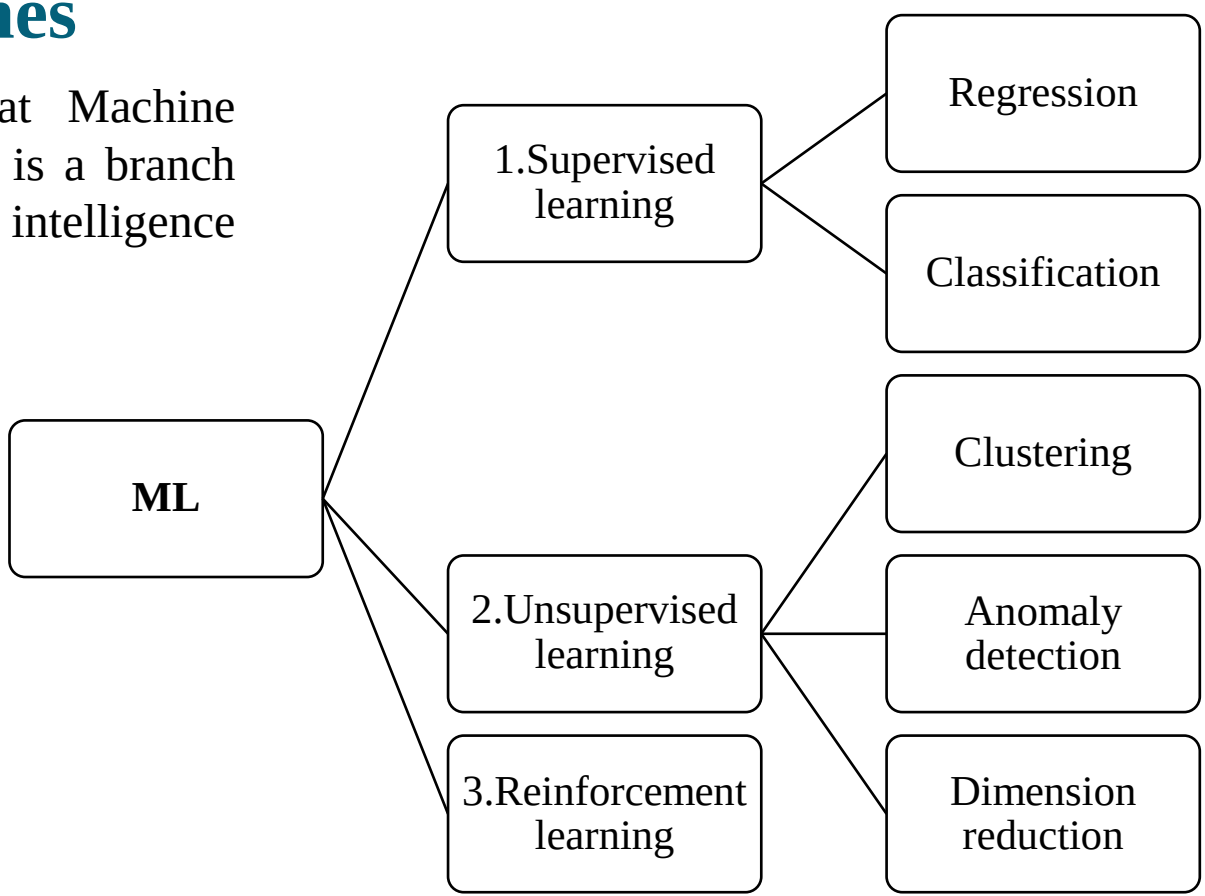
- Humans are also capable of learning. More precisely, when the agent is a computer, we call it machine learning.

- An agent is **learning** if it improves its performance after making observations about the world. In some AI applications, we need to develop a system that can learn from experience.

- Why Machine learning?
  1. The designers cannot anticipate all possible future situations. (E.g., maze)
  2. The designers have no idea how to program a solution themselves. (E.g., stock market)

ML branches

- Remember that Machine learning (ML) is a branch of artificial intelligence (AI).



ML branches

1. Supervised learning:

Someone gives us examples and the right answer for those examples.
We must predict the right answer for unseen examples.

2. Unsupervised learning:

We see examples but get no feedback.
We need to find patterns in the data.

3. Reinforcement learning:

We take actions and get rewards.
Must learn how to get high rewards.

- Among supervised, unsupervised, and reinforcement learning, supervised learning has seen the most advancements.

Supervised learning

- We have two kinds of variables:

1. Outcome measurement (Y)
2. Vector of p predictor measurements (X)

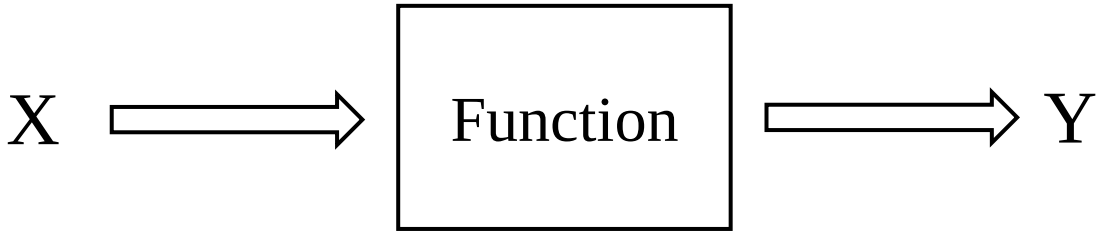
$\mathbf{Y}$ dependent variable, response, target, outcome, label

$\mathbf{X}$ independent variables, inputs, regressors, predictors

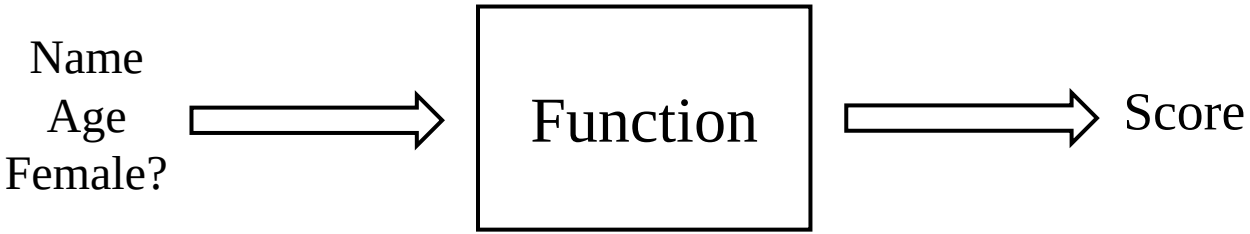
	Name	Age	Female?	Score
Student 01	Ali	21	No	17
Student 02	Ahmad	21	No	19
Student 03	Zahra	22	Yes	15
Student 04	Sorosh	19	Yes	20
Student 05	Mehdi	20	No	19
Student 06	Negin	22	Yes	20

Observation

Supervised learning



- In our example:



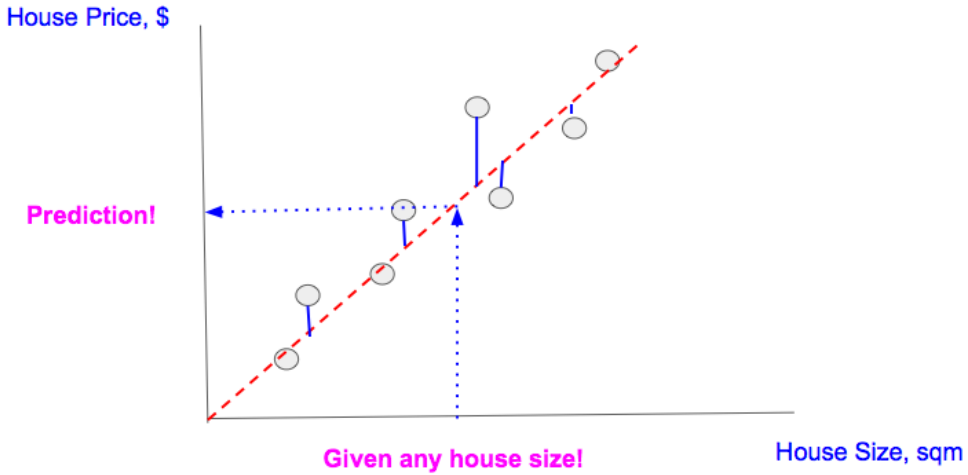
Supervised learning

Some applications of supervised learning:

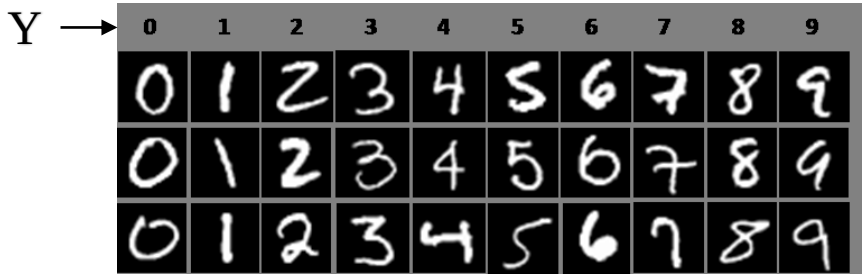
Input(X)	Output(Y)	Application
email	Spam? (yes/no)	Spam filtering
audio	Text transcripts	Speech recognition
English text	Persian text	Machine translation
Ad, user info	Click? (yes/no)	Online advertising
Image, radar info	Position of other cars	Self-driving cars
Image of phone	Defect? (yes/no)	Visual inspection

Supervised learning

- In the **regression** problem, Y is continuous (E.g., price, blood pressure).
- infinitely many possible outputs.



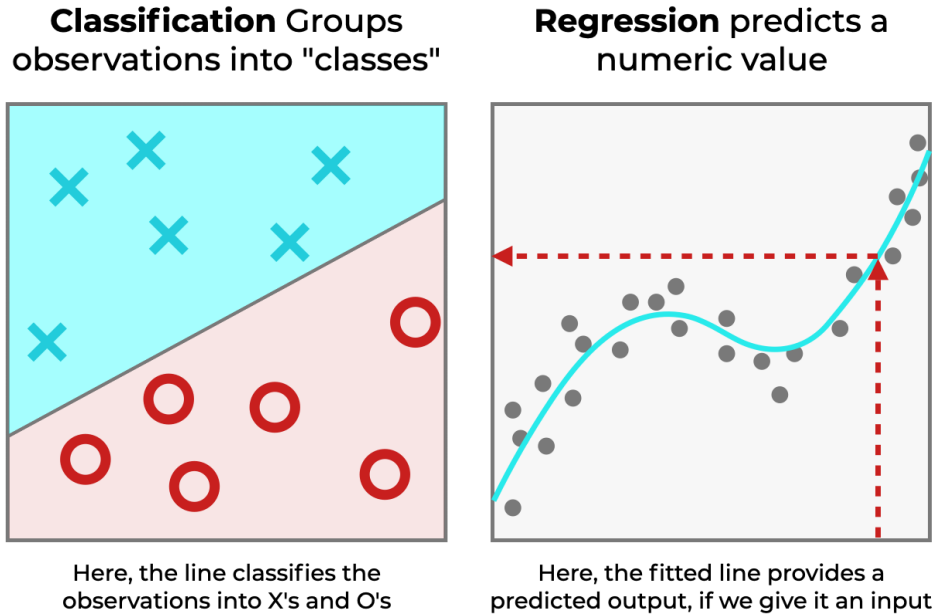
- In the **classification** problem, Y takes values in a finite, unordered set (E.g., survived/died, digit 0-9).



Supervised learning

Difference:

Regression	Classification
Predict a number	Predict categories
infinitely many possible outputs	small number of possible outputs



Supervised learning

Specify the type; Regression or Classification problem?

Input(X)	Output(Y)	Application	Type?
email	Spam? (yes/no)	Spam filtering	
audio	Text transcripts	Speech recognition	
English text	Persian text	Machine translation	
Ad, user info	Click? (yes/no)	Online advertising	
Image, radar info	Position of other cars	Self-driving cars	
Image of phone	Defect? (yes/no)	Visual inspection	

Supervised learning

- Objectives are:
  1. Accurately predict unseen cases.
  2. Understand which inputs affect the outcome, and how.

- It is important to understand the ideas behind the various techniques, to know **how** and **when** to use them.

- One must understand the simpler methods first, to grasp the more sophisticated ones.

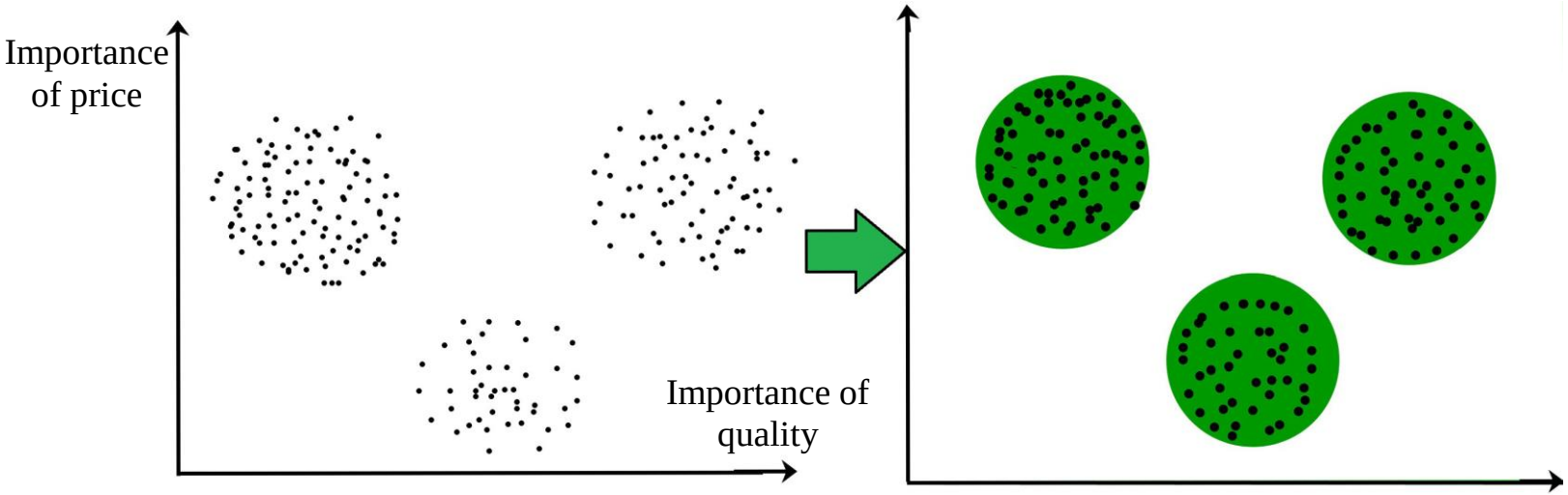
- It is important to accurately assess the performance of a method, to know how well or how badly it is working. (**performance measures**)

- Simpler methods often perform as well as fancier ones!

Unsupervised learning

- No outcome variable, just a set of predictors (features) measured on a set of samples. In other words, Data only comes with inputs x, but not output labels y. Algorithm must find structure in the data.

- For example: market segmentation



Regression

Correlation

- The **correlation coefficient** is a statistical measure of the strength of a linear relationship between two variables.

- Pearson correlation = linear correlation

✓ Pearson correlation for population:

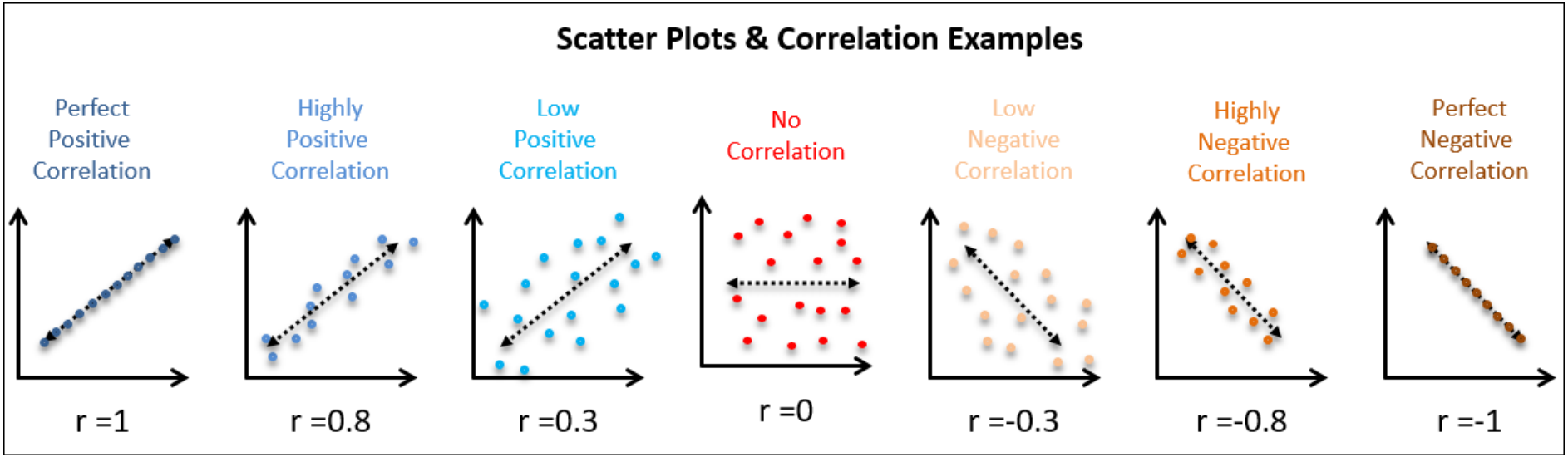
$$\rho_{XY} = \frac{\operatorname{cov}(X, Y)}{\sqrt{\operatorname{var}(X) \cdot \operatorname{var}(Y)}}$$

✓ Pearson correlation for sample:

$$r_{xy} = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sqrt{\sum_{i=1}^{n}(x_i - \bar{x})^2 \sum_{i=1}^{n}(y_i - \bar{y})^2}}$$

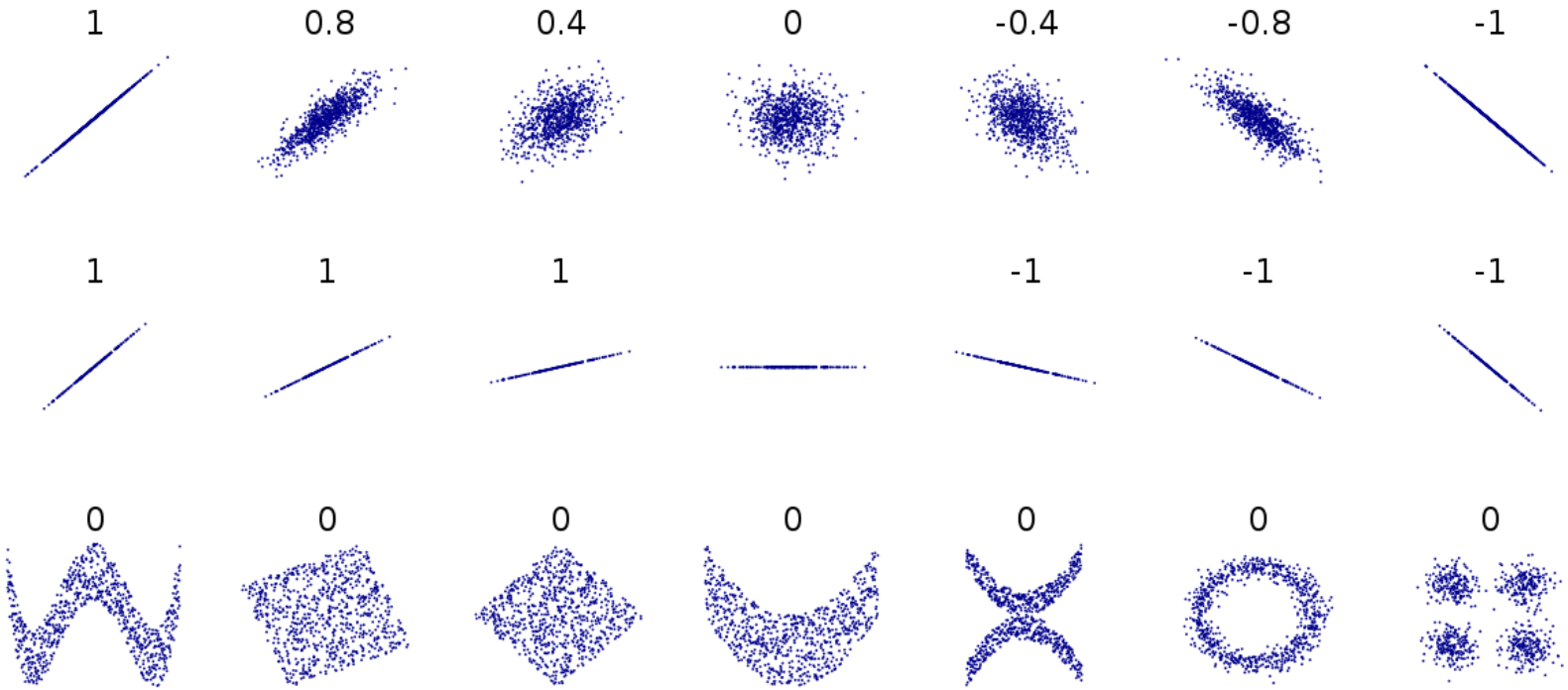
Correlation

- Correlation values can range from -1 to 1.



- Correlation is the main idea of **linear regression**.

Correlation



- A correlation coefficient of 0 means there is no linear relationship.

Linear regression

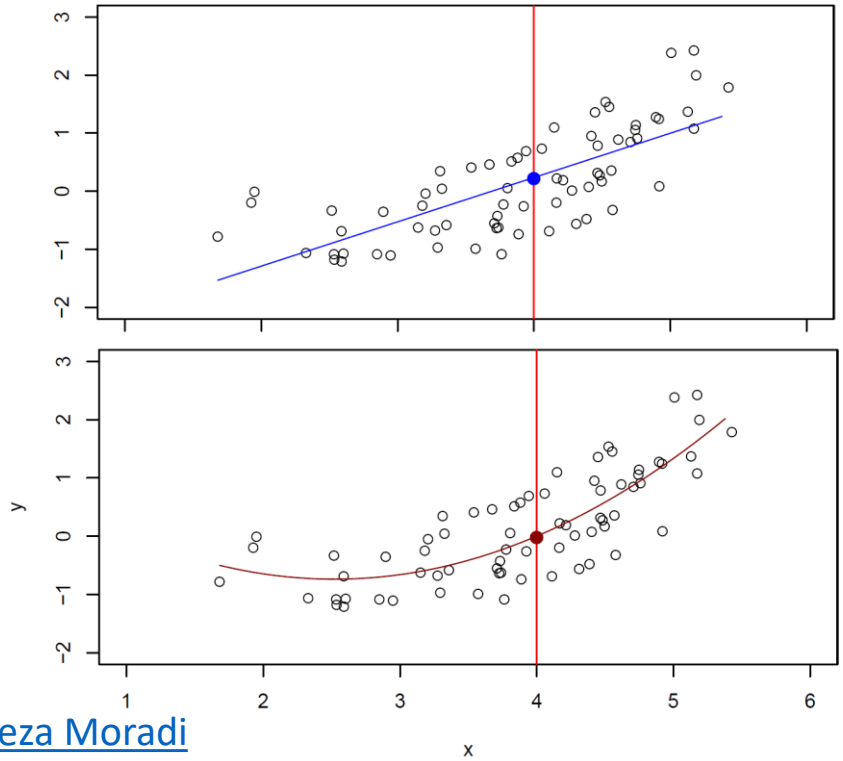
- **Linear regression** is an algorithm that provides a linear relationship between independent variables and a dependent variable.

✓ A linear model

Equation :

✓ A quadratic model

Equation:



Linear regression

- Now the question is how to choose the best line for a linear regression problem?

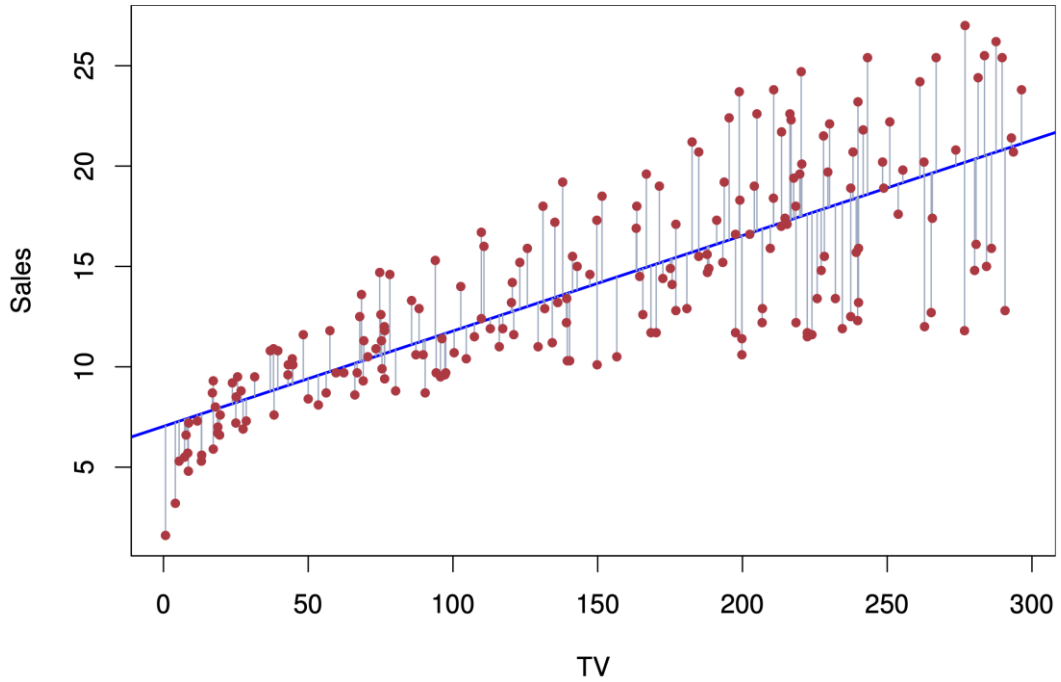
- Hypothesis:

$$y_i = \beta_0 + \beta_1 x_i + \epsilon_i$$

- Here :

$$Sales_i = \beta_0 + \beta_1 TV_i + \epsilon_i$$

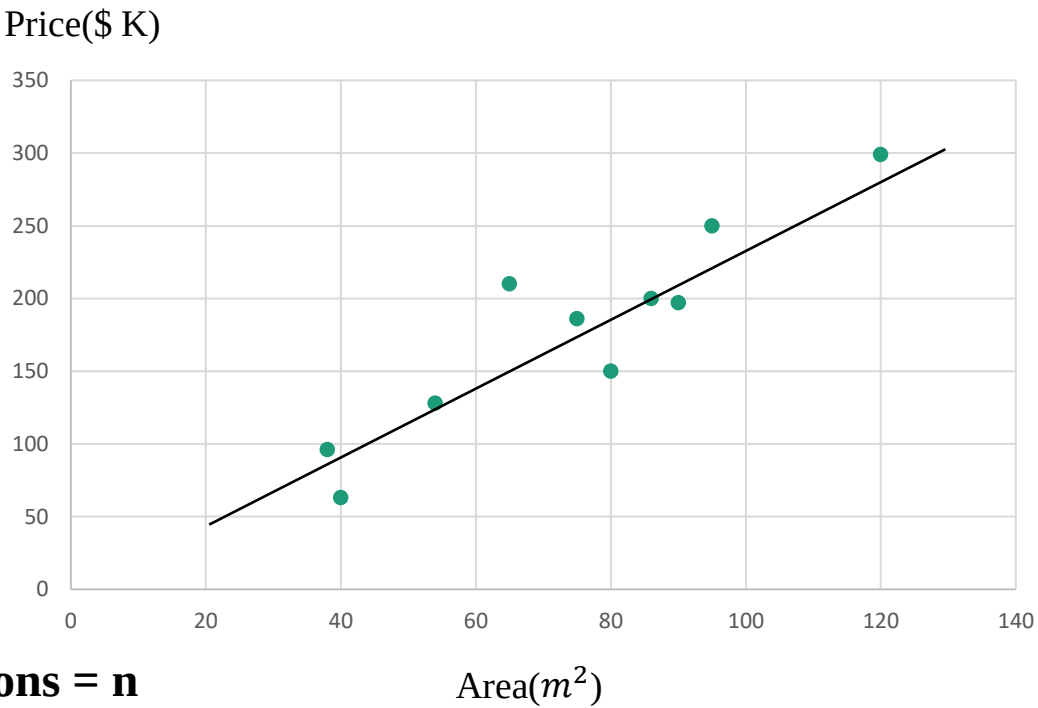
✓ $\beta_0 \Rightarrow$ intercept
✓ $\beta_1 \Rightarrow$ slope
✓ $\epsilon \Rightarrow$ error term



Linear regression

- Let's see an example:

Observation	Area(m^2)	Total Price($ K)
1	54	128
2	75	186
3	80	150
4	40	63
5	38	96
6	120	299
7	90	197
8	95	250
9	86	200
10	65	210



- **Number of training observations = n**
- Here n = 10

Linear regression

- Single training example (observation) :

$$(x_i, y_i) \qquad \forall i = 1,2,\cdots,m$$

- For example:

- Learning is done on these training examples. In other words, our model gain experience on these observations.

Observation	Area(m^2)	Total Price($ K)
1	54	128
2	75	186
3	80	150
4	40	63
5	38	96
6	120	299
7	90	197
8	95	250
9	86	200
10	65	210

✓ We want to **estimate the best linear line** => **estimate the best parameters**
✓ We also predict future prices using this best linear line:

$$\hat{Y}_i = \hat{\beta}_0 + \hat{\beta}_1 \cdot X_i$$

Cost function

- β_0 and β_1 are **parameters**.
- They are also called **coefficients** or **weights**.
- We use a cost function for estimating our parameters:

sum of squared errors (SSE)

$$J(\beta_0, \beta_1) = \frac{1}{2n} \sum_{i=1}^{n} (\hat{y}_i - y_i)^2$$

- This related to the AI paradigm. (do the best = an optimization problem)

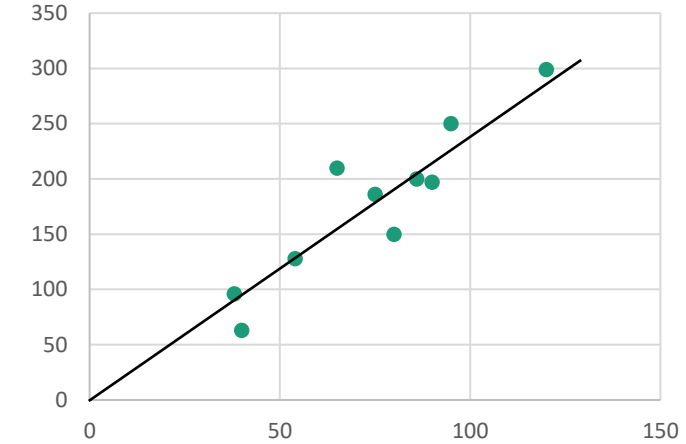
$$Min\ J(\beta_0, \beta_1)$$

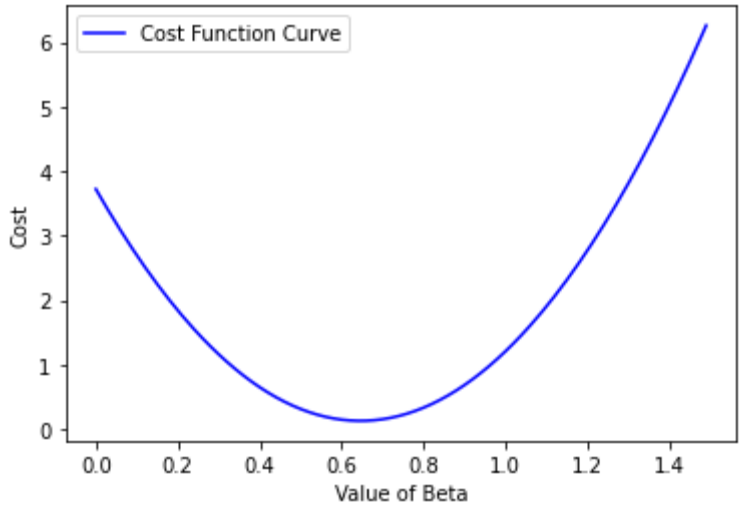
Cost function

- Assume we have a simple model:

$$\hat{Y}_i = \hat{\beta}_1 \cdot X_i$$

- Then the cost function for regression problem will be like this:



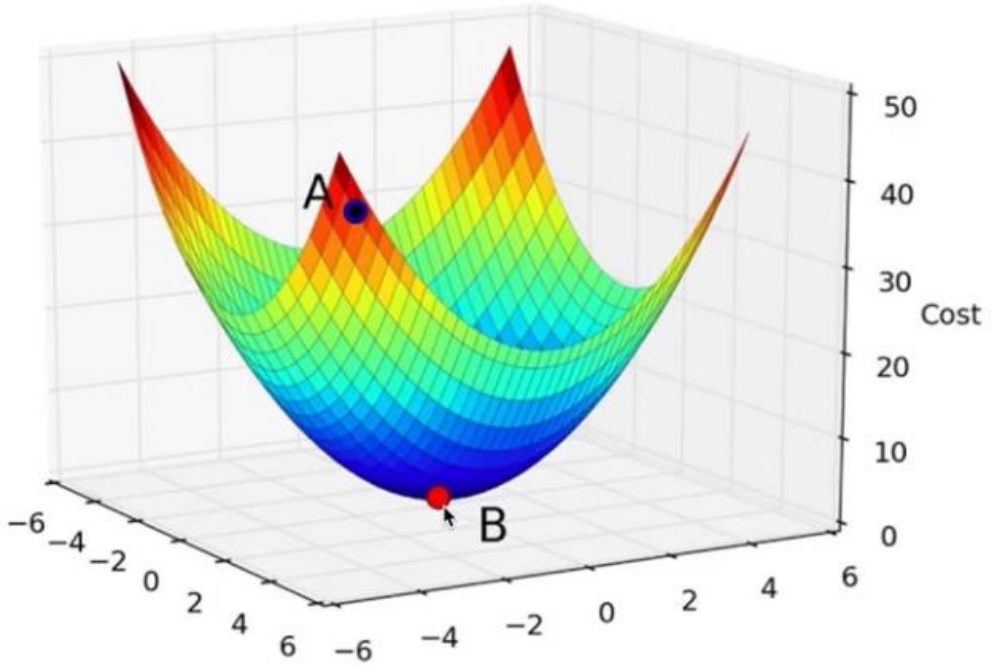


Cost function

- For a more complex model like:

$$\hat{Y}_i = \hat{\beta}_0 + \hat{\beta}_1 \cdot X_i$$

- Then the cost function for regression problem will be like this:



Cost function optimization

- What are our options for solving this problem?

  1. Solve the optimization problem for exact answer.
  2. Use Cost function optimization algorithms.

- Recall that solving unconstrained optimization problems involves finding stationary points using **derivatives**. In this case, we aim to minimize the cost function $J(\beta_0, \beta_1)$.

$$\begin{cases} \dfrac{\partial J(\beta_0, \beta_1)}{\partial \beta_0} = 0 \\ \dfrac{\partial J(\beta_0, \beta_1)}{\partial \beta_1} = 0 \end{cases} \Longrightarrow \begin{cases} \widehat{\beta_1} = \dfrac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^{n}(x_i - \bar{x})^2} \\ \widehat{\beta_0} = \bar{y} - \widehat{\beta_1}.\bar{x} \end{cases}$$

OLS

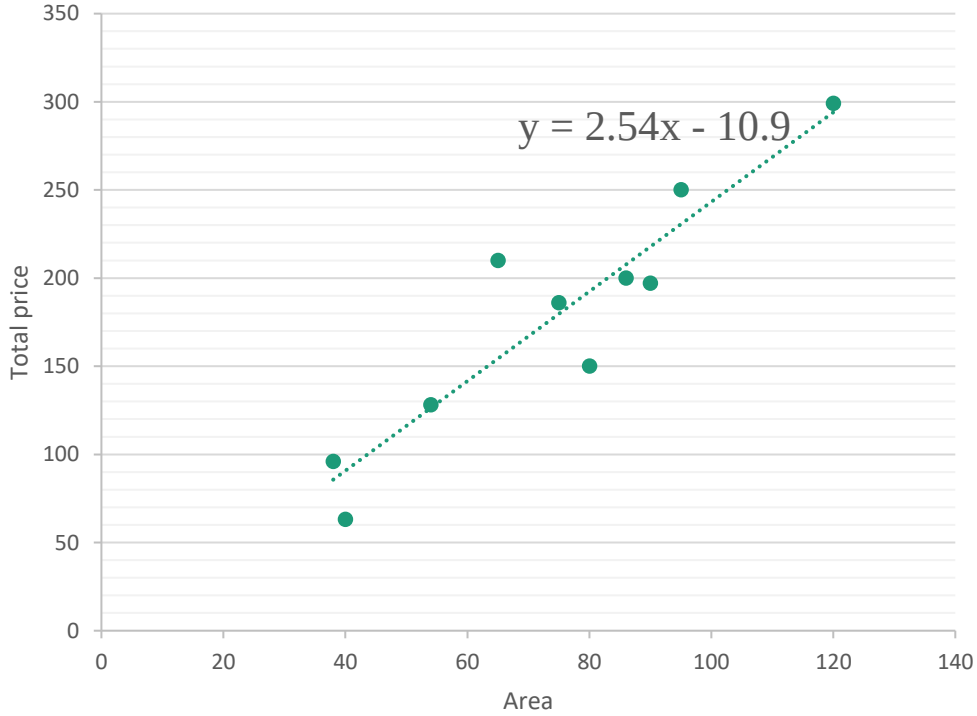
Solution for Cost function optimization

- For our example:

$$\widehat{\beta_1} = \frac{\sum_{i=1}^{n}(x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^{n}(x_i - \bar{x})^2} = 2.54$$

$$\widehat{\beta_0} = \bar{y} - \widehat{\beta_1}.\bar{x} = -10.9$$

- Exact answers may not always be computationally feasible, especially for complex problems or large datasets.

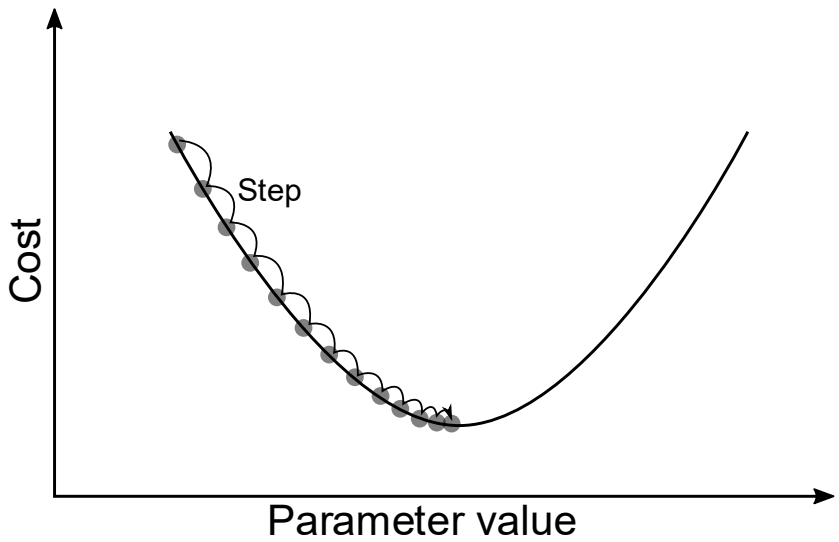


Gradient Descent

- We can estimate solutions with sufficient accuracy using approximation methods. **Numerical methods** like **Gradient Descent** finds approximate solutions for a wide range of problems.

- How does gradient descent work?

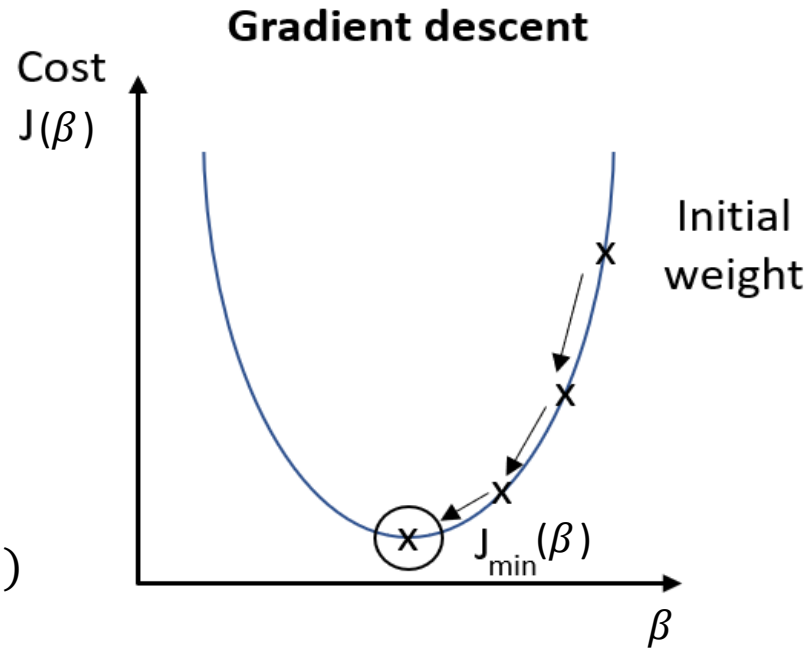
- Gradient descent is an iterative algorithm, which means we apply an update repeatedly until some criterion is met.



Gradient Descent

- We **initialize** the weights (parameters) to something reasonable (e.g., all zeros) and **repeatedly adjust them** in the direction of steepest descent.

- Outline:
  1. Start with some β_0, β_1
  2. Keep changing β_0, β_1 to reduce $J(\beta_0, \beta_1)$
  3. Until we settle at or near a minimum



Gradient Descent; Algorithm

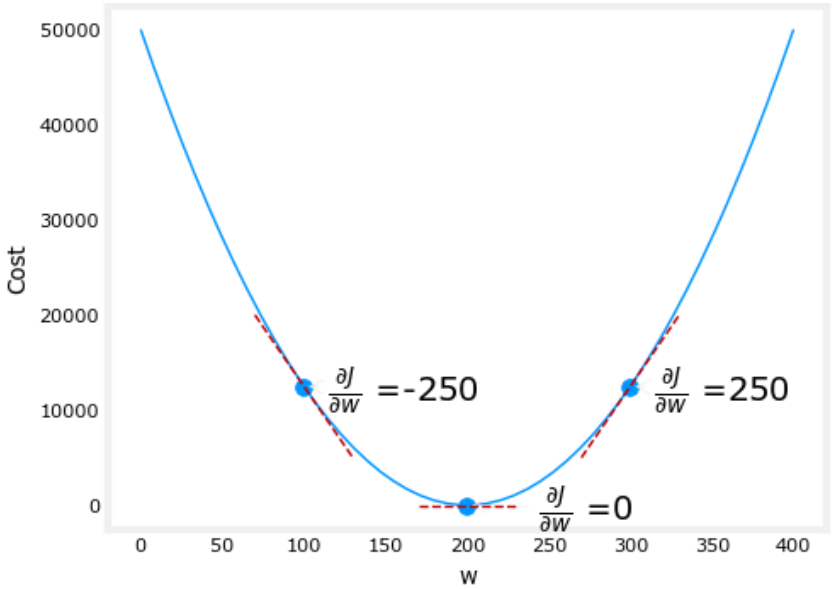
- We initialize the weights (parameters) to something reasonable and then we observe:
  - If $\frac{\partial J}{\partial \beta_0} > 0$, then increasing β_0 increases J.
  - If $\frac{\partial J}{\partial \beta_0} < 0$, then increasing β_0 decreases J.

- So, the following update **decreases the cost function**:

$$\beta_{1,new} = \beta_{1,old} - \alpha \frac{\partial J}{\partial \beta_1}$$

- We can update β_0 the same way:

$$\beta_{0,new} = \beta_{0,old} - \alpha \frac{\partial J}{\partial \beta_0}$$



Gradient Descent; Algorithm

- Now we must calculate the gradient of cost function with respect to B_1 and B_0:
  - Gradient of cost function with respect to β_1:

$$\frac{\partial J}{\partial \beta_1} = \frac{1}{n}\sum_{i=1}^{n}(\hat{y}_i - y_i)x_i$$

  - Gradient of cost function with respect to β_0:

$$\frac{\partial J}{\partial \beta_0} = \frac{1}{n}\sum_{i=1}^{n}(\hat{y}_i - y_i)$$

Gradient Descent; Algorithm

- Finally, we can use the following updates to decreases the cost function:
  - Updates for β_1:

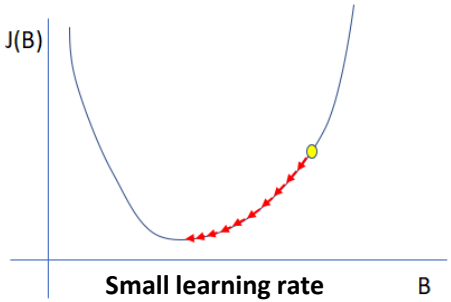
$$\beta_{1,new} = \beta_1 - \alpha \frac{\partial J}{\partial \beta_1} = \beta_1 - \alpha \frac{1}{n} \sum_{i=1}^{n} (\hat{y}_i - y_i) x_i$$

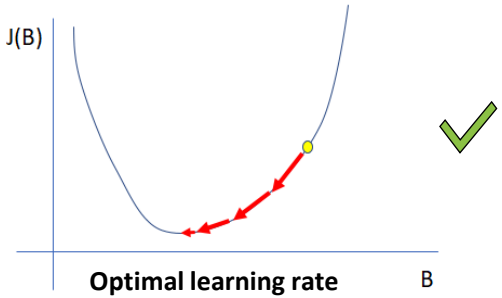
  - Updates for β_0:

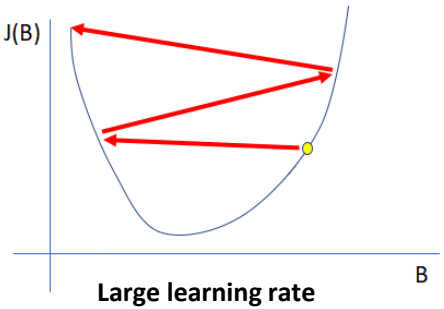
$$\beta_{0,new} = \beta_0 - \alpha \frac{\partial J}{\partial \beta_0} = \beta_0 - \alpha \frac{1}{n} \sum_{i=1}^{n} (\hat{y}_i - y_i)$$

Gradient Descent; Learning Rate

- Alpha (α) is the **learning rate**. The larger it is, the faster parameters change.
    - If α is too small the gradient decent may be very slow.
    - If α is too large gradient descent may never reach minimum or fail to converge.







Gradient Descent; Visualization

- A non-convex cost function: