Machine Learning
ICT-4261

By-
Dr. Jesmin Akhter
Professor
Institute of Information Technology
Jahangirnagar University

Contents

The course will mainly cover the following topics:

- ✔ A Gentle Introduction to Machine Learning
- ✔ Linear Regression
- ✔ Logistic Regression
- ✔ Naive Bayes
- ✔ Support Vector Machines
- ✔ Decision Trees and Ensemble Learning
- ✔ Clustering Fundamentals
- ✔ Hierarchical Clustering
- ✔ Neural Networks and Deep Learning
- ✔ Unsupervised Learning……

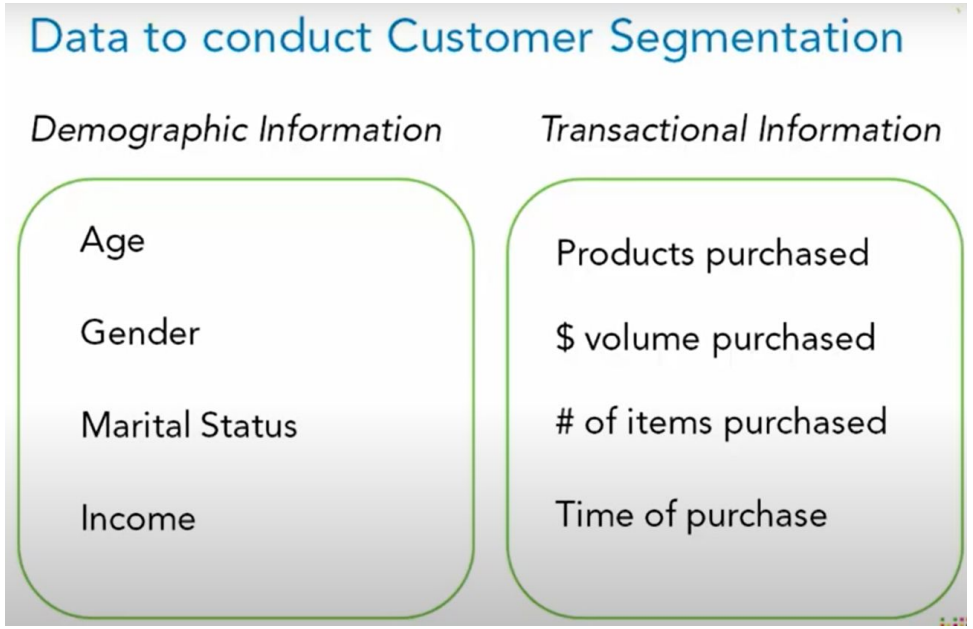
Outline

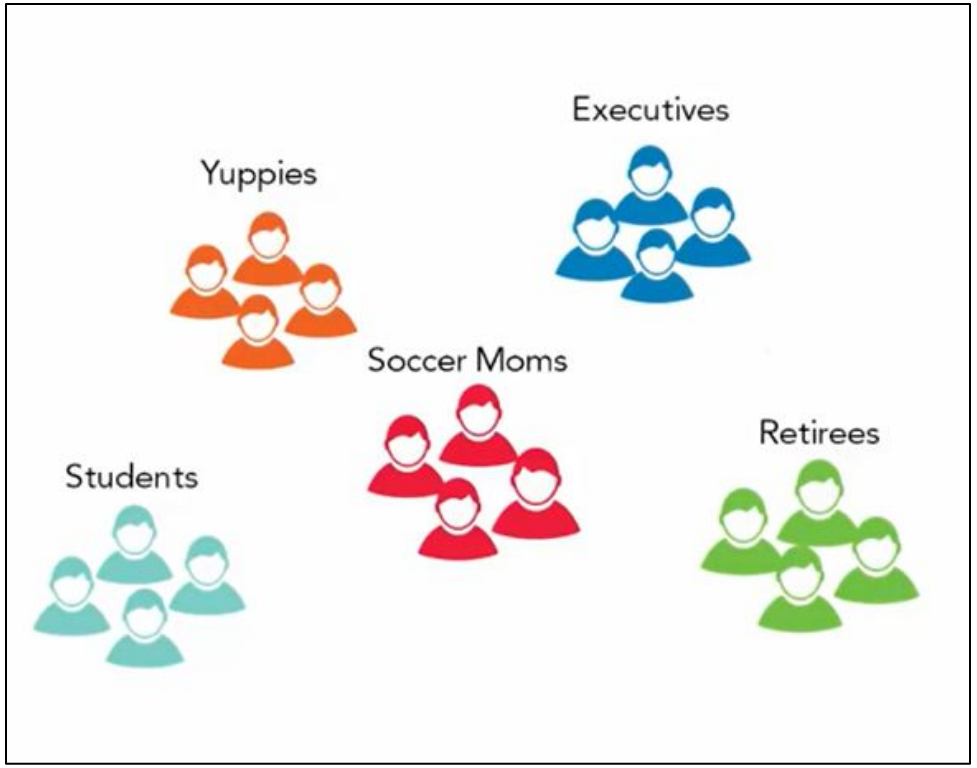
- ✔ A Gentle Introduction to Machine Learning
  - Unsupervised learning
  - Reinforcement learning
- ✔ Linear Regression
  - Linear models
  - **Hypothesis function for Linear Regression**
  - **Cost function**
  - Gradient descent algorithm
  - Polynomial regression

Unsupervised machine learning

- ✔ **Unsupervised learning** is a type of machine learning where the algorithm learns from **unlabeled data** without any predefined outputs or target variables.
- ✔ It finds **hidden patterns, similarities, or groupings** within the data to get insights and make data-driven decisions without the need for human intervention..
- ✔ It is particularly **useful when dealing with large datasets** where **manual labeling would be impractical or costly**.
- ✔ This method's ability to discover similarities and differences in information make it ideal for **exploratory data analysis, cross-selling strategies, customer segmentation, and image and pattern recognition**.
  - Cross-selling is the process of offering a customer products that are compatible with the ones they're purchasing

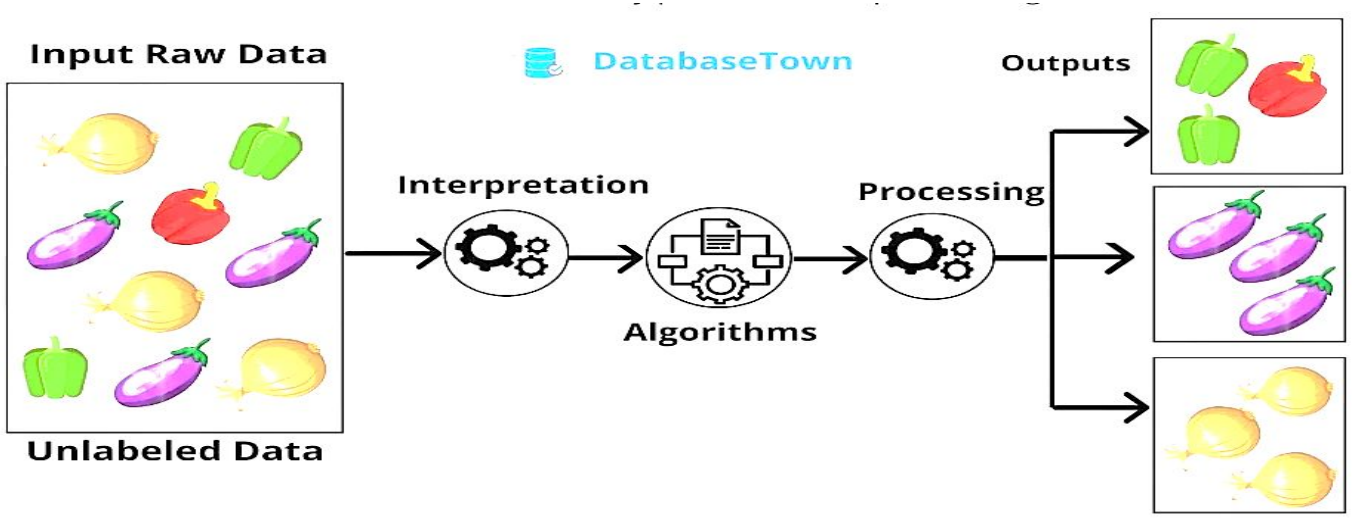
✔ Customer segmentation is the process by which you divide your customers based on common characteristics – such as demographics or behaviours, so your marketing team or sales team can reach out to those customers more effectively.





How does Unsupervised Learning Work?

- ✔ Suppose an input dataset containing images of different types of cats and dogs. Here, we have taken an unlabeled input data, which means it is not categorized and corresponding outputs are not given.
- ✔ Now, this unlabeled input data is fed to the machine learning model in order to train it.
- ✔ Firstly, it will interpret the raw data to find the hidden patterns from the data and then will apply suitable algorithms such as K-means clustering algorithms.
- ✔ Once it applies the suitable algorithm, the algorithm divides the data objects into groups according to the similarities and difference between the objects.



Types of Unsupervised Learning

- **Unsupervised** learning can be broken down into three main tasks:
  - Clustering
  - Association rules
  - Dimensionality reduction.

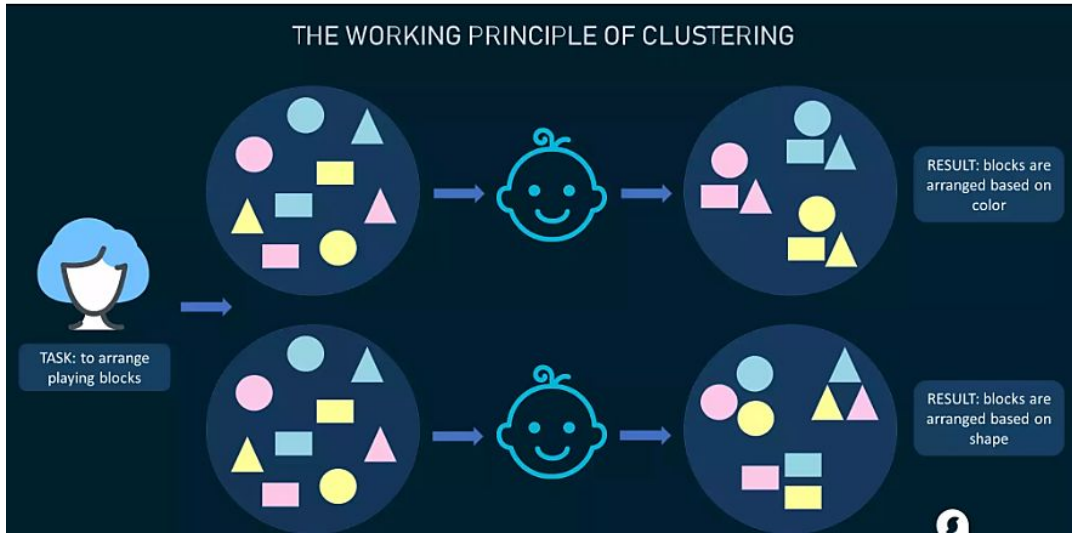
Unsupervised Algorithm:

1. K-means clustering
2. KNN (k-nearest neighbours)
3. Hierarchal clustering
4. Anomaly detection
5. Neural Networks
6. Principle Component Analysis
7. Apriori algorithm

Types of Unsupervised Learning

Clustering Algorithms

✔ Clustering algorithms only interpret the input data and find natural groups or clusters in feature space

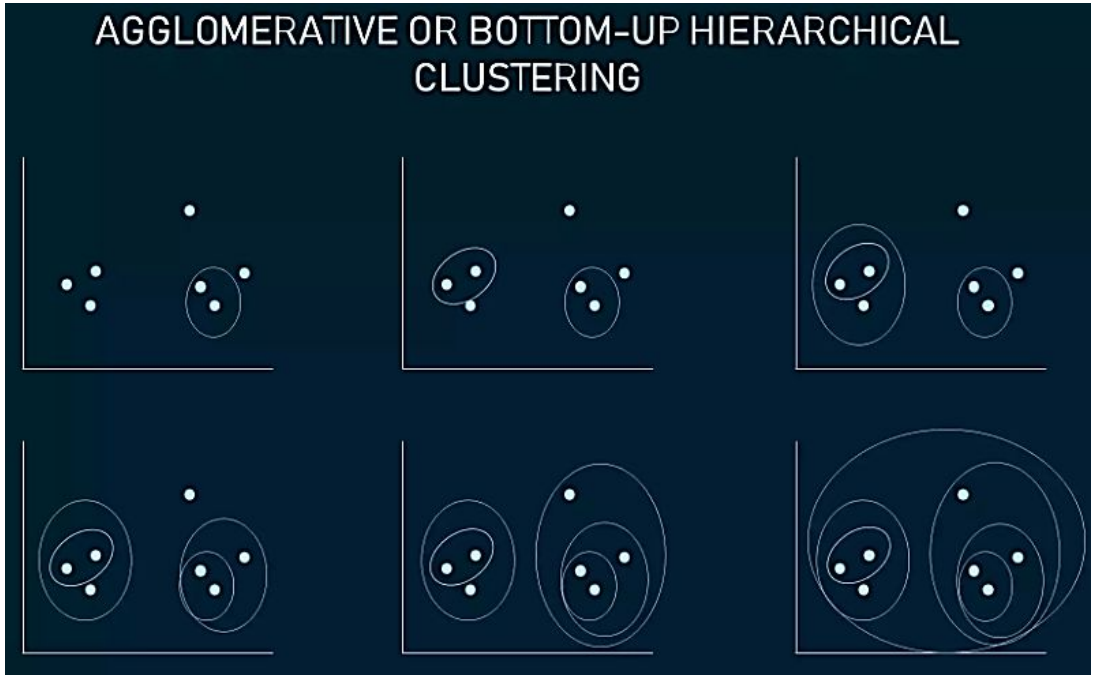


✔ Clustering is a popular type of unsupervised learning approach. You can even break it down further into different types of clustering; for example:

- **Exlcusive clustering**: or "hard" clustering
  - It is the kind of grouping in which one piece of data can belong only to one cluster.
- **Overlapping clustering**:
  - A soft cluster in which a single data point may belong to multiple clusters with varying degrees of membership.

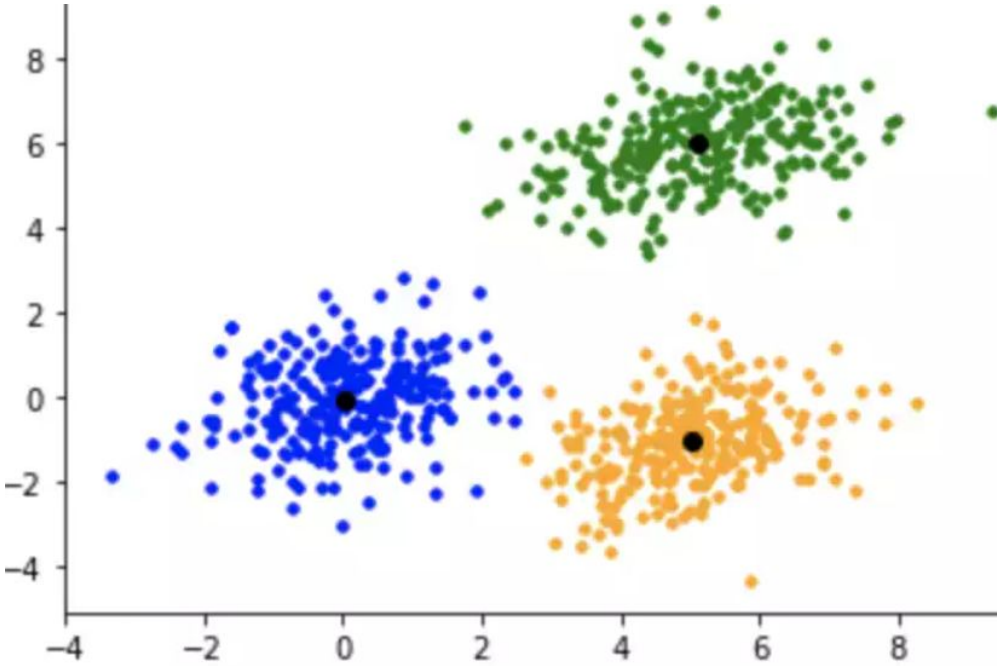
Types of Unsupervised Learning

- ✔ **Clustering Algorithms**
  - **Hierarchical Clustering:**
    - Hierarchical clustering develops a hierarchy of clusters by merging or splitting them depending on their similarity. Here, two close cluster are going to be in the same cluster.
    - In case you start with all data items attached to the same cluster and then **perform splits** until each data item is set as a separate cluster, the approach will be called **top-down or *divisive* hierarchical clustering.**
    - Two clusters that are closest to one another are then **merged** into a single cluster. The merging goes on iteratively till there's only one cluster left at the top. Such an approach is known **as bottom-up or *agglomerative*.**
    - The example shows how seven different clusters (data points) are merged step by step based on distance until they all create one large cluster.



Types of Unsupervised Learning

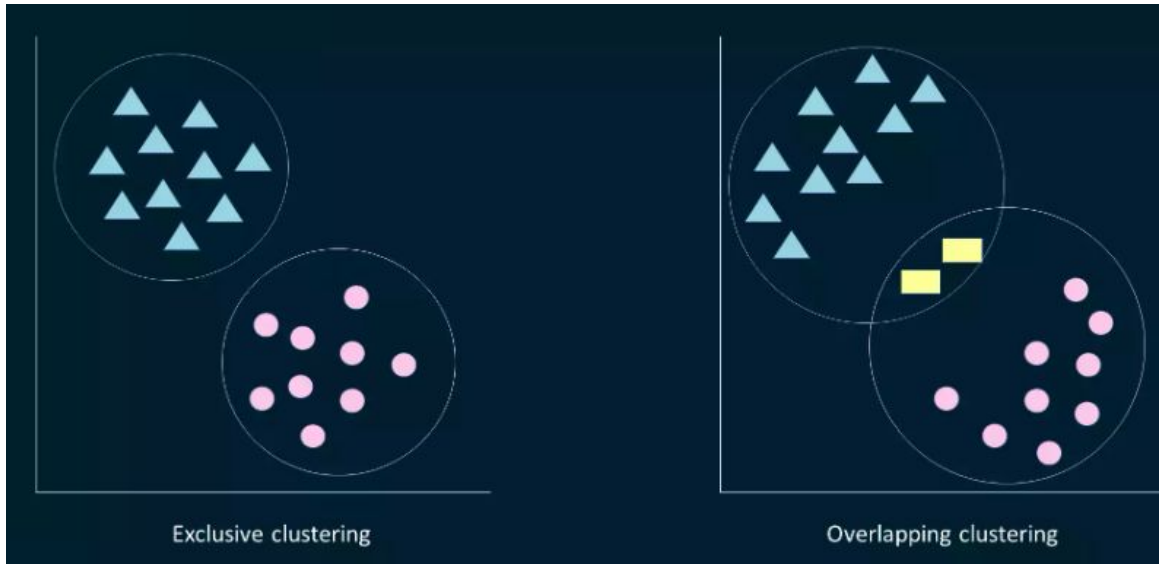
- **Clustering Algorithms**
  - **K-Means Clustering**
    - In K-means clustering, data is grouped in terms of characteristics and similarities.
    - K is a letter that represents the number of clusters. For example, if K=3, then the number of desired clusters is 3. If K=10, then the number of desired clusters is 10.
    - It puts the data points into the predefined number of clusters *K*.
    - Each data item then gets assigned to the nearest cluster center, called *centroids* (black dots in the picture). The latter act as data accumulation areas.
    - The procedure of clustering may be repeated several times until the clusters are well-defined.



Ideal clustering with a single centroid in each cluster. Source: GeeksforGeeks

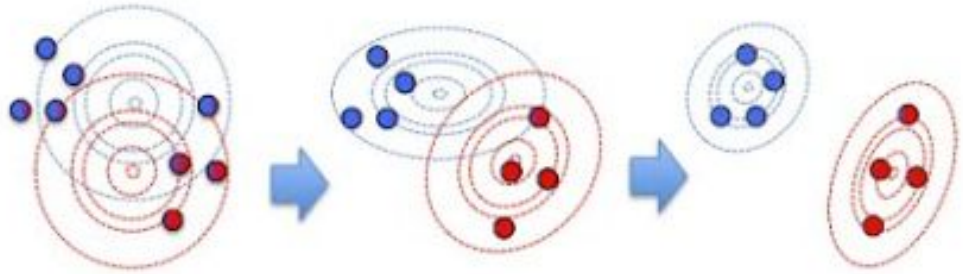
Types of Unsupervised Learning

- ✔ **Clustering Algorithms**
  - **Fuzzy K-means**
    - It is an extension of the K-means algorithm used to perform overlapping clustering. Unlike the K-means algorithm, fuzzy K-means implies that data points can belong to more than one cluster with a certain level of closeness towards each.
    - The closeness is measured by the distance from a data point to the centroid of the cluster. So, sometimes there may be an overlap between different clusters.
  - –

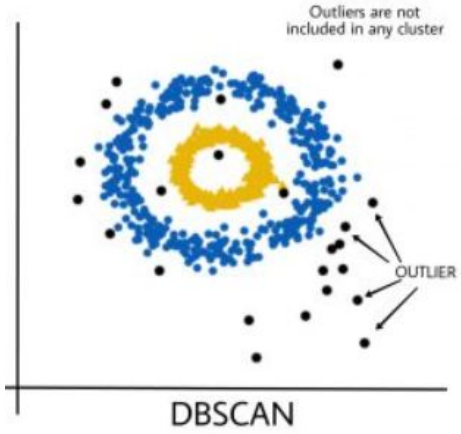


Types of Unsupervised Learning

- ✔ **Clustering Algorithms**
  - **Gaussian Mixture Models** (GMMs) is an algorithm used in probabilistic clustering.
    - Models assume that there is a certain number of Gaussian distributions, each representing a separate cluster.
    - The algorithm is basically utilized to decide which cluster a particular data point belongs to.



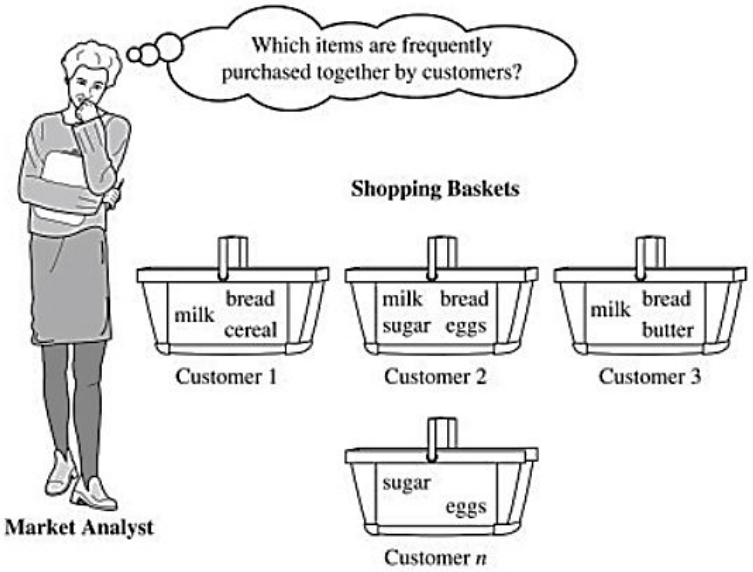
  - **DBSCAN (Density-Based Spatial Clustering of Applications with Noise):** DBSCAN groups data points based on their density, identifying clusters of high-density regions and classifying outliers as noise.



Types of Unsupervised Learning

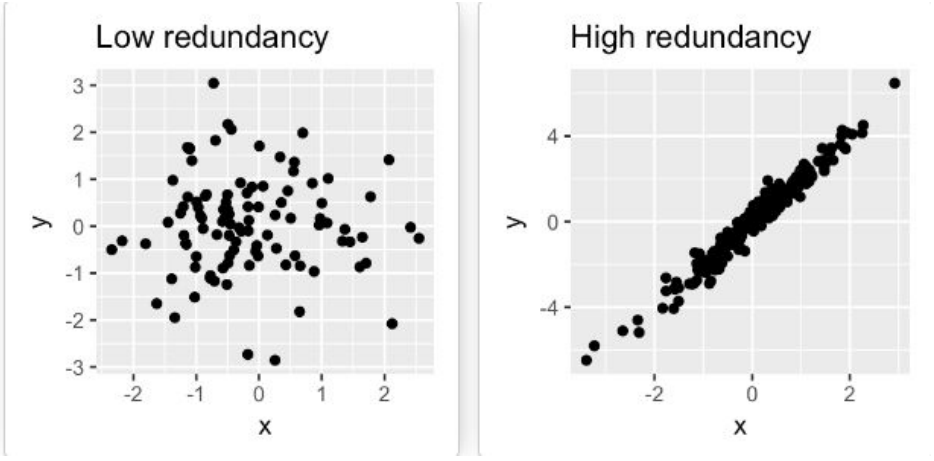
- **Association Rule Mining**
  - Association rule mining focuses on discovering interesting relationships or patterns between variables in the large database. It determines the **set of items that occurs together in the dataset**. Association rule makes marketing strategy more effective. Such as people who buy X (suppose a bread) also tend purchase Y (Butter/Jam) item. The widely used algorithm for association rule mining is the **Apriori** algorithm.

    - A real-life example of this is market basket analysis, where retailers analyze customer purchase data to identify relationships between products frequently bought together. For instance, this analysis might reveal that customers who purchase diapers also tend to buy baby wipes



Types of Unsupervised Learning

- ✔ **Dimensionality Reduction Algorithms** Dimensionality reduction techniques are used to reduce the number of input variables or features while retaining meaningful information. Some popular dimensionality reduction algorithms include:
  - **Principal Component Analysis (PCA):** PCA **transforms** the original features into a **lower-dimensional space** while **preserving** the maximum amount of **information**.
  - The PCA method is particularly useful when the variables within the data set are highly correlated. Correlation indicates that there is redundancy in the data. Due to this redundancy, PCA can be used to reduce the original variables into a smaller number of new variables (**principal components**) explaining most of the variance in the original variables.

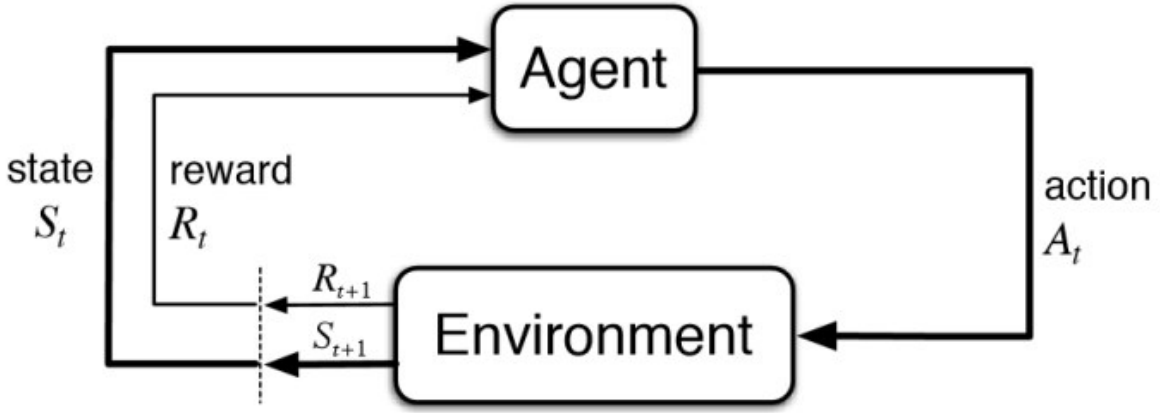


What is the difference between supervised and unsupervised learning?

- ✔ Supervised learning requires **labeled data** with **input features and corresponding output labels**, while unsupervised learning aims to **discover patterns or structures in unlabeled data** without predefined output labels.

Reinforcement machine learning

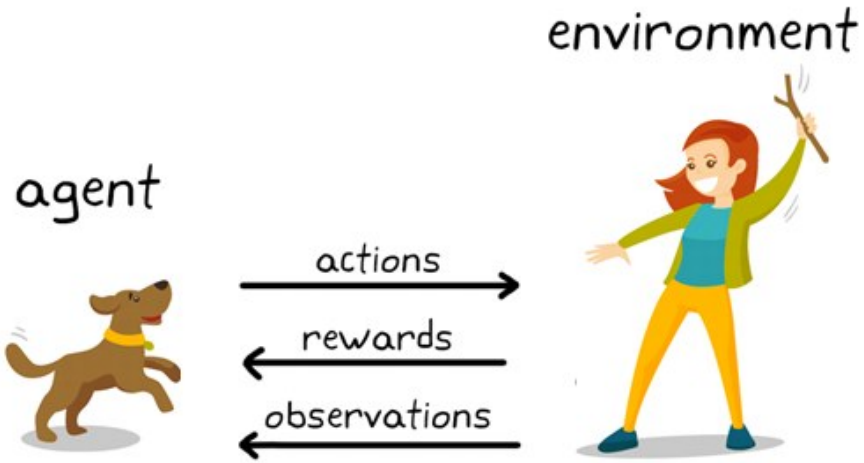
- ✔ Reinforcement Learning(RL) enables an agent to learn in an interactive environment by trial and error using feedback from its own actions and experiences.
- ✔ Here, agents are self-trained on reward and punishment mechanisms.
- ✔ It can take actions and interact with it.
- ✔ Reinforcement machine learning algorithm isn't trained using sample data.
- ✔ A sequence of successful outcomes will be reinforced to develop the best recommendation or policy for a given problem.



Basic Diagram of Reinforcement Learning

Reinforcement machine learning

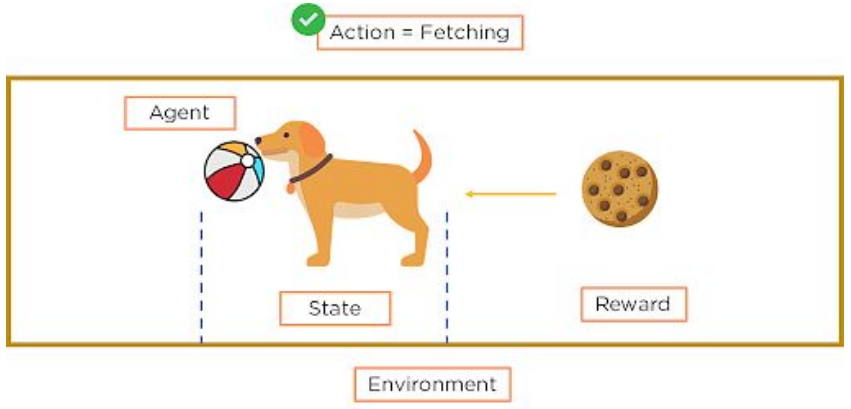
- ✔ Through a series of Trial and Error methods, an agent keeps learning continuously in an interactive environment from its own actions and experiences.
- ✔ The only goal of it is to find a suitable action model which would increase the total cumulative reward of the agent.
- ✔ It learns via interaction and feedback.
- ✔ You can see a dog and a master. Let's imagine you are training your dog to get the stick. Each time the dog gets a stick successfully, you offered him a feast (a bone).
- ✔ Eventually, the dog understands the pattern, that whenever the master throws a stick, it should get it as early as it can gain a reward (a bone) from a master in a lesser time.



Reinforcement Learning Example

Important Terms in Reinforcement Learning

- ✔ Agent: Agent is the model that is being trained via reinforcement learning. It is the sole decision-maker and learner
- ✔ Environment: a physical world where an agent learns and decides the actions to be performed
- ✔ Action: All possible steps that can be taken by the model/agent
- ✔ State: The current position/ condition returned by the model/the current situation of the agent in the environment
- ✔ Reward: To help the model move in the right direction, it is rewarded/points are given to it to appraise some action. It's usually a scalar value and nothing but feedback from the environment
- ✔ Policy: Policy determines how an agent will behave at any time. It acts as a mapping between Action and present State. The agent prepares strategy(decision-making) to map situations to actions.
- ✔ **Value —** Future reward that an agent would receive by taking an action in a particular state



SUPERVISED
OR
UNSUPERVISED?

SCENARIO - 1

Facebook
Face Recognition

SCENARIO - 2

Netflix Movie
Recommendation

SCENARIO - 3

Fraud
Detection

Explanation

- ✔ Scenario 1: Facebook recognizes your friend in a picture from an album of tagged photographs
  - Explanation: It is supervised learning. Here Facebook is using tagged photos to recognize the person. Therefore, the tagged photos become the labels of the pictures and we know that when the machine is learning from labeled data, it is supervised learning.
- ✔ Scenario 2: Recommending new songs based on someone's past music choices
  - Explanation: It is supervised learning. The model is training a classifier on pre-existing labels (genres of songs). This is what Netflix, Pandora, and Spotify do all the time, they collect the songs/movies that you like already, evaluate the features based on your likes/dislikes and then recommend new movies/songs based on similar features.
- ✔ Scenario 3: Analyze bank data for suspicious looking transactions and flag the fraud transactions
  - Explanation: It is unsupervised learning. In this case, the suspicious transactions are not defined, hence there are no labels of "fraud" and "not fraud". The model tries to identify outliers by looking at anomalous transactions and flags them as 'fraud'.

Outline

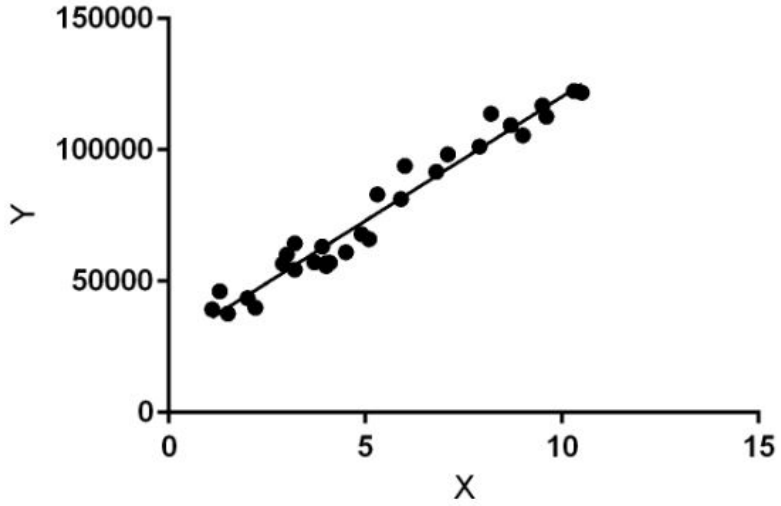
- ✔ Linear Regression
  - Linear models
  - **Hypothesis function for Linear Regression**
  - **Cost function**

Linear Regression

- ✔ Linear regression is a type of supervised machine learning algorithm that computes the **linear relationship** between **a dependent variable and one or more independent features**.
- ✔ When the number of the independent feature, is **1** then it is known as **Univariate** Linear regression, and in the case of **more than one feature**, it is known as **multivariate** linear regression.
- ✔ It predicts the **continuous output variables** based on the independent input variable like the prediction of **house prices** based on different parameters like **house age, distance from the main road, location, area, size** etc.
- ✔ The **goal** of the algorithm is to find the **best linear equation** that can predict the value of the **dependent variable** based on the independent variables.

Linear Regression

- ✔ Y is a dependent or target variable and X is an independent variable also known as the predictor of Y.
- ✔ There are many types of functions or modules that can be used for regression.
  - A **linear function** is the simplest type of function.
- ✔ X may be a single feature or multiple features representing the problem.
- ✔ **Linear regression performs the task to predict a dependent variable value (y) based on a given independent variable (x)). Hence, the name is Linear Regression**.
- ✔ In the figure, X (input) is the work experience and Y (output) is the salary of a person.



Hypothesis function for Linear Regression

- ✔ salary of a person
  - m = number of training examples
  - x = input variables / features
  - y = output variable "target" variables
    - (x,y) - single training example
    - (x_i, y_j) - specific example (ith training example)
- ✓ We have assumed that our independent feature is the experience x and the respective salary y is the dependent variable.

x(Work experience in years)	y(salary)
2	3000
3	4000
4	5000
5	6000
10	11000
12	13000

Hypothesis function for Linear Regression

✔ Let's assume there is a linear relationship between X and Y then the salary can be predicted using:

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

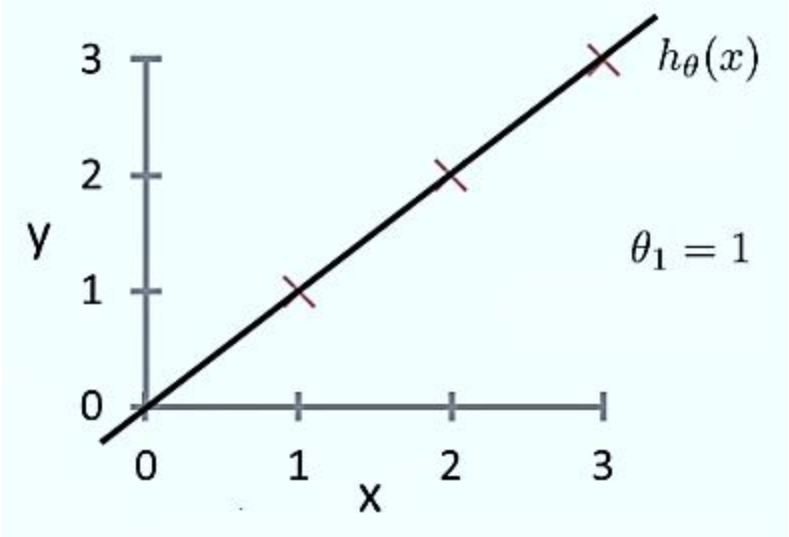
- ✓ The model gets the best regression fit line by finding the best θ_0 and θ_1 values.
- ✓ **θ_0:** intercept
- ✓ **θ_1:** coefficient of x or gradient
- ✓ Chosen these parameters so $h_\theta(x)$ is close to y for our training examples
- ✓ Once we find the best θ_0 and θ_1 values, we get the best-fit line. So when we are finally using our model for prediction, it will predict the value of $h_\theta(x)$ for the input value of x.
  - Different values give you different functions
    - If θ_0 is 1.5 and θ_1 is 0 then we get straight line parallel with X along 1.5
    - If θ_1 is > 0 then we get a positive slope
  - Think of $h_\theta(x)$ as a "y imitator" - it tries to convert the x into y, and considering we already have y we can evaluate how well $h_\theta(x)$ does this

Linear models

- ✔ We can learn with a larger number of features
  - So may have other parameters which contribute towards a price of houses
    - Size
    - Age
    - Number bedrooms
    - Number floors
  - x_1, x_2, x_3, x_4
  - With **multiple features becomes hard to plot**
    - Notation becomes more complicated too
      - Best way to get around with this is the notation of **linear algebra**
      - Gives notation and set of things you can do with **matrices and vectors**
- ✓ Now we have multiple features. A linear model is based on the assumption that it's possible to approximate the output values through a regression process based on the rule. Hypothesis can be written
  - $h_\theta(x) = \theta_0 + \theta_1 x_1 + \theta_2 x_2 + \theta_3 x_3 + \theta_4 x_4 = \theta_0 + \sum_{i=1}^{m} \theta_i x_i$

Implementation: cost function(Cont.)

- ✔✔ **Hypothesis -** is like your prediction machine, throw in an x value, get a putative y value
  - ✓ The **cost function or the loss function** is nothing but the **error** or **difference between the predicted value $h_\theta(x)$ and the true value Y**. It is the **Mean Squared Error (MSE)** between the predicted value and the true value.
  - ✓ **Cost function -** is a way to, use your training data, determine values for your θ values which make the hypothesis as accurate as possible



Cost function - a deeper look

- ✔ Lets consider some intuition about the cost function and why we want to use it
  - The cost function determines parameters
  - The value associated with the parameters determines how your hypothesis behaves, with different values
  - To achieve the best-fit regression line, the model aims to predict the target value. So the **Cost function** updates the θ_0 and θ_1 values, to reach the best value that minimizes the error between the predicted $h_\theta(x)$ value and the true y value.
- ✓ Simplified hypothesis
  - Assumes $\theta_0 = 0$
- ✓ Cost function and goal here are very similar to when we have θ_0, but with a simpler parameter
  - Simplified hypothesis makes visualizing cost function $J(\theta_1)$ a bit easier
- ✓ So hypothesis pass through 0,0

$$h_\theta(x) = \theta_1 x$$

$\theta_0 = 0$

θ_1

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

$$\underset{\theta_1}{\text{minimize}}\ J(\theta_1)$$

Exploring the Cost Function

Suppose we have a training set with three points (1, 1), (2, 2), and (3, 3). We plot the function
$h_\theta(x) = \theta_1 * x$ for different values of θ_1 and calculate the corresponding cost function $J(\theta_1)$.

- ✔ When $\theta_1 = 1$: $h_\theta(x) = x$, the line passes through the origin, perfectly fitting the data. The cost function $J(\theta_1)$ is 0 since $h_\theta(x)$ equals y for all training examples.

- ✔ Setting $\theta_1 = 0.5$: $h_\theta(x) = 0.5 * x$, the line has a smaller slope. The cost function $J(\theta_1)$ now measures the squared errors between $h_\theta(x)$ and y for each example. It provides a measure of how well the line fits the data.

Cost function - a deeper look (Cont.)

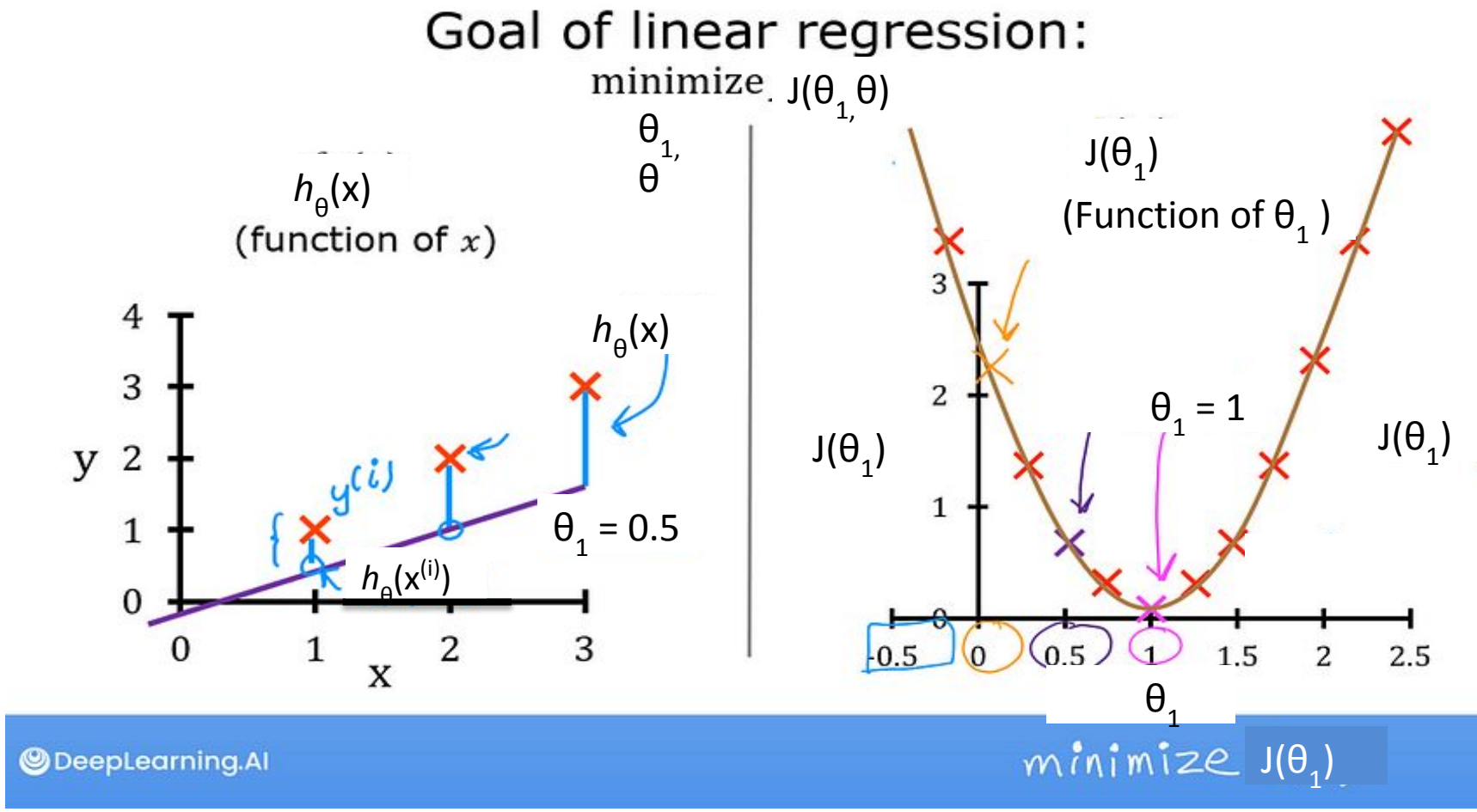
- Two key functions we want to understand
  - $h_\theta(x)$
    - Hypothesis is a function of x - function of what the size of the house is
  - $J(\theta_1)$
    - Is a function of the parameter of θ_1
  - So for example
  - $\theta_1 = 1$
  - $J(\theta_1) = 0$

  - Plot
    - θ_1 vs $J(\theta_1)$
    - Data
      - 1)
        - » $\theta_1 = 1$
        - » $J(\theta_1) = 0$
      - 2)
        - » $\theta_1 = 0.5$
        - » $J(\theta_1) = \sim 0.58$
      - 3)
        - » $\theta_1 = 0$
        - » $J(\theta_1) = \sim 2.3$

x	y
1	1
2	2
3	3

x	y
1	1
2	2
3	3

- ✔ If we compute a range of values plot
  - $J(\theta_1)$ vs θ_1 we get a polynomial (looks like a quadratic)
- ✔ The optimization objective for the learning algorithm is find the value of θ_1 which minimizes $J(\theta_1)$
  - So, here $\theta_1 = 1$ is the best value for θ_1



Outline

- ✔ Linear Regression
  - Gradient descent algorithm
  - Polynomial regression

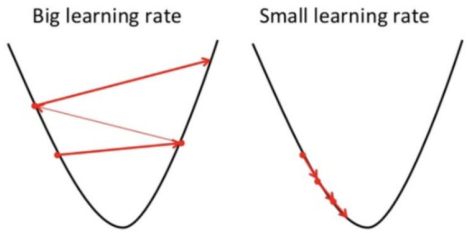
Gradient Descent Algorithm

- ✔ A linear regression model can be trained using the optimization algorithm gradient descent by iteratively modifying the model's parameters to reduce the mean squared error (MSE) of the model on a training dataset.
- ✔ To update θ_0 and θ_1 values in order to reduce the Cost function (minimizing RMSE value) and achieve the best-fit line the model uses Gradient Descent.
- ✔ The idea is to start with random θ_0 and θ_1 values and then iteratively update the values, reaching minimum cost.
- ✔ A gradient is nothing but a **derivative** that defines the **effects on outputs** of the function with a little bit of **variation in inputs**.

Gradient Descent Algorithm

- ✔ Let's differentiate the cost function(J) with respect to θ_0 and θ_1

Gradient Descent Algorithm



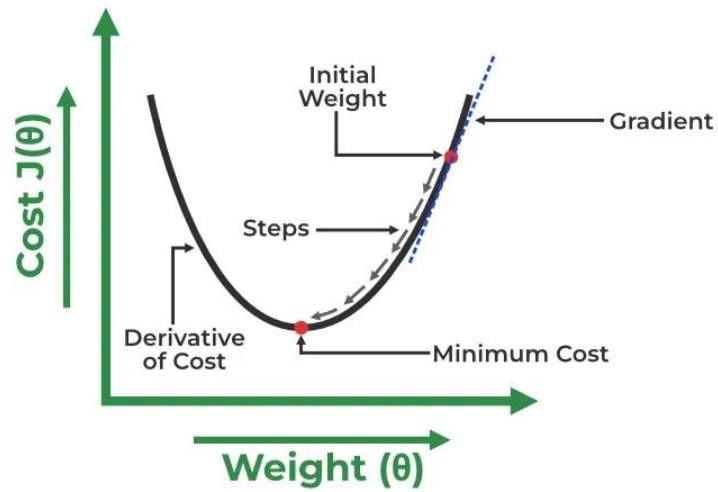
- ✔ Finding the **coefficients of a linear** equation that **best fits the training data** is the objective of linear regression.
  - ✓ By moving in the direction of the Mean Squared Error negative gradient with respect to the coefficients, the coefficients can be changed.
  - ✓ Here we choose a **hyperparameter learning rate** which is denoted by alpha α. The learning rate determines the size of the steps the algorithm takes towards the minimum. For example, if the gradient at a point is 4 and the learning rate is 0.1, the descent value would be $-0.1 \times 4 = -0.4$.
  - ✓ **Repeat until convergence**. Continuing with the example above, if the current value of a parameter is 5, it would be updated to $5-(-0.4)=5.4$. This updating moves the parameter towards the function's minimum.

$$\{$$

$$\theta_0 := \theta_0 - \alpha \underbrace{\frac{1}{m}\sum_{i=1}^{m}(h_\theta(x^{(i)}) - y^{(i)})}_{\frac{\partial}{\partial\theta_0}J(\theta)}$$

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

(simultaneously update θ_0, θ_1) $\}$



Gradient Descent Algorithm

Step 1 we first initialize the parameters of the model randomly

- Start at *0,0* (or any other value)
- Keeping changing θ_0 and θ_1 a little bit to try and reduce $J(\theta_0, \theta_1)$

Step 2 Compute the gradient of the cost function with respect to each parameter. It involves making partial differentiation of cost function with respect to the parameters.

Step 3 Update the parameters of the model by taking steps in the opposite direction of the model. Here we choose a **hyperparameter learning rate** which is denoted by alpha. It helps in deciding the step size of the gradient.

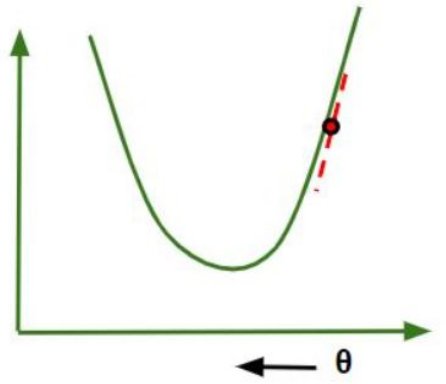
- Each time you change the parameters, you select the gradient which reduces $J(\theta_0, \theta_1)$, the most possible.

Step 4 Repeat steps 2 and 3 iteratively to get the best parameter for the defined model.

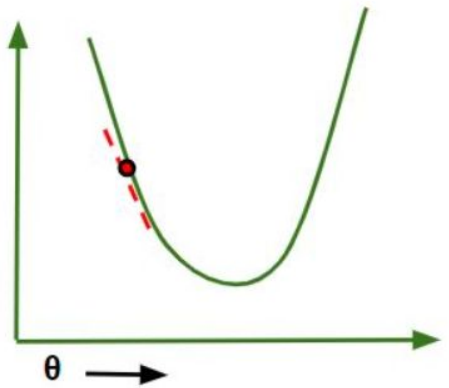
- Do so until you converge to a local minimum

How Does Gradient Descent Work

- ✔ Gradient descent works by moving downward toward the pits or valleys in the graph to find the minimum value. This is achieved by taking the derivative of the cost function, as illustrated in the figure below.
- ✔ During each iteration, gradient descent step-downs the **cost function** in the direction of the steepest descent.
- ✔ By adjusting the parameters in this direction, it seeks to reach the minimum of the cost function and find the best-fit values for the parameters.
- ✔ The size of each step is determined by parameter **α** known as **Learning Rate**.
- ✔ In the Gradient Descent algorithm, one can infer two points :
- ✔ **If slope is +ve** : $\theta_j = \theta_j$ – (+ve value). Hence the value of θ_j decreases.
- ✔ **If slope is -ve** : $\theta_j = \theta_j$ – (-ve value). Hence the value of θ_j increases.



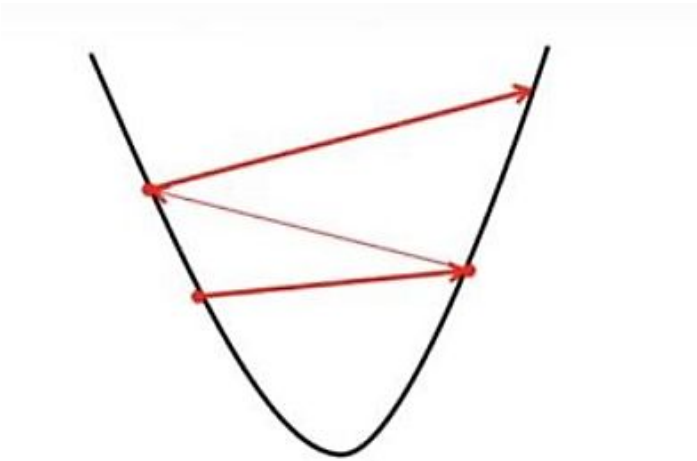
If slope is +ve in Gradient Descent



If slope is -ve in Gradient Descent

How To Choose Learning Rate

- ✔ The choice of correct learning rate is very important as it ensures that Gradient Descent converges in a reasonable time.
- ✔ If we choose **α to be very large**, Gradient Descent can overshoot the minimum. It may fail to converge or even diverge.
- ✔ If we choose **α to be very small**, Gradient Descent will take small steps to reach local minima and will take a longer time to reach minima.

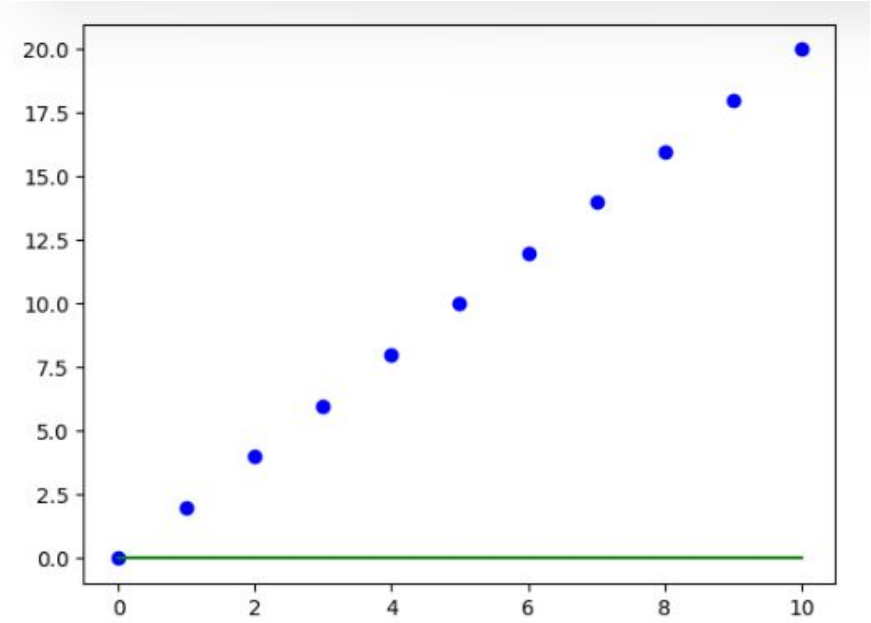


Effect of large alpha value on Gradient Descent

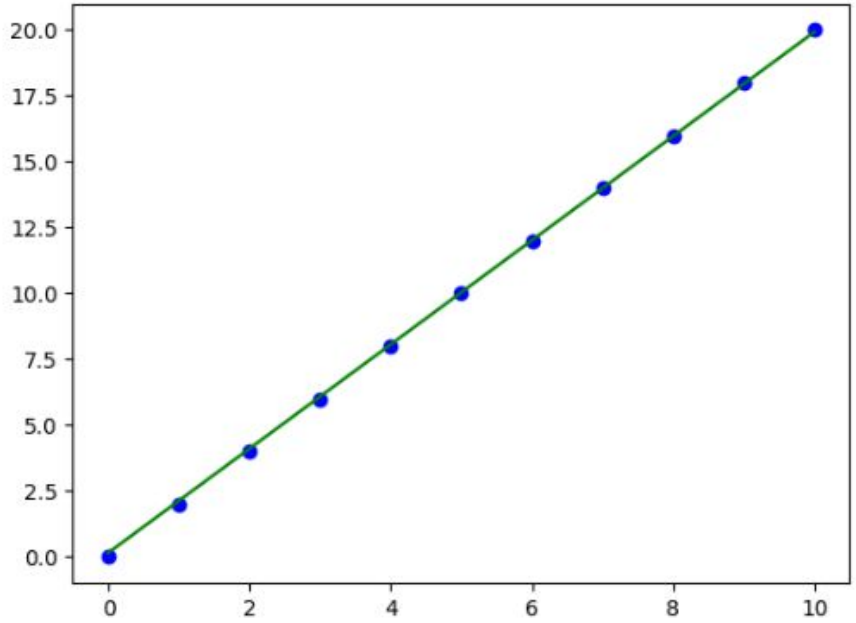
Effect of small alpha value on Gradient Descent

Example in a Real-World Scenario

- ✔ Consider a machine learning model where you're trying to minimize the error between predicted and actual values (the loss function). The parameters could be the weights in a linear regression model_predicting house prices. The gradient descent algorithm would iteratively adjust these weights to minimize the difference (loss) between the predicted and actual house prices.



Regression line before gradient descent iteration



Regression line after gradient descent iteration

Numerical example

- ✔✔ In the previous example, the data set are given in the right side
  - ✓ Assume θ_0 =0; so that the hypothesis is $h(x) = \theta_1 x$ and
  - ✓ Using gradient descent, updating $\theta_1 = \theta_1 - \frac{\alpha}{m}\sum\left(h\left(x^{(i)}\right) - y^{(i)}\right) \cdot \left(x^{(i)}\right)$

x	y
1	1
2	2
3	3

.9966

=.996 (.996-1)x1+ (1.98-2)x2+ (2.98-3)x3 **=1.002=1** **(Converged)**

Thank You