LECTURE 18

Clustering

Building models of clustering in sklearn

Data Science, Fall 2024 @ Knowledge Stream

Sana Jabbar

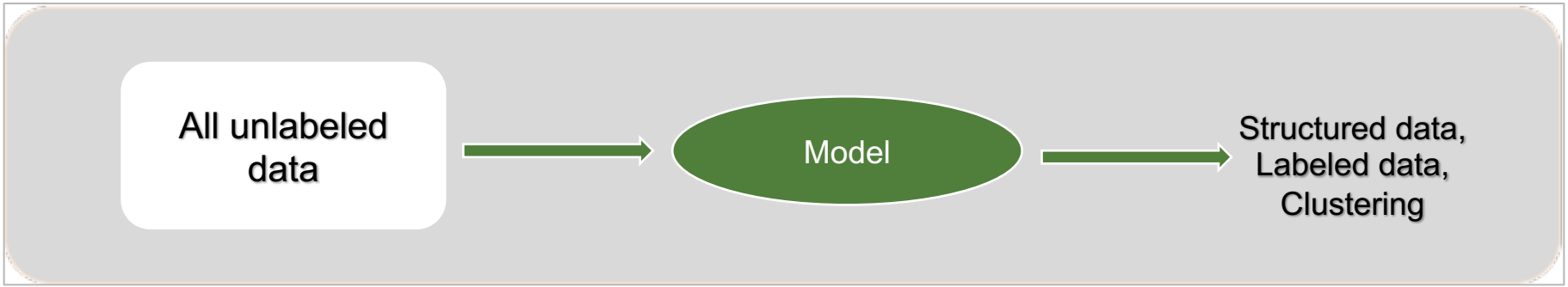
Outline

Lecture 18

- Introduction to Unsupervised Learning
- Clustering Overview
- K-means Clustering

Unsupervised Learning

- The learning algorithm would receive unlabelled raw data to train a model and to find patterns in the data



Clustering, aka unsupervised learning (due to historical reasons), is the most widely used technique.

Overview

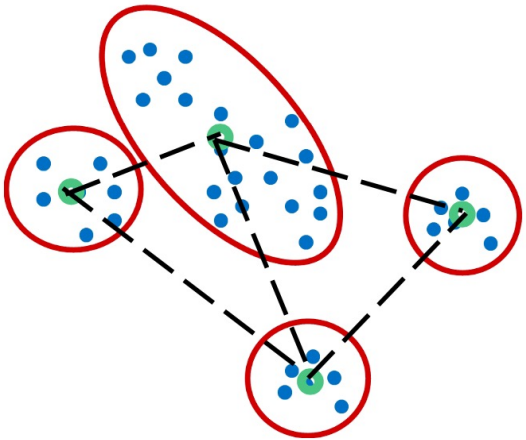
- Given the data, group ‘similar’ points into the form of clusters.

Overview

- Idea: the process of grouping data into similar groups known as **clusters.**
- Formally, organize the unlabelled data into classes such that
  - **Inter-cluster similarity is minimized:** low similarity between data points in different classes
  - **Intra-cluster similarity is maximized:** High similarity between data points of each class
- In contrast to classification, we learn the number of classes and class labels directly from the data.

Evaluation

- **Evaluation of Clustering using Internal Data:**
- **Inter-cluster separability**
  - measure of the isolation of the cluster - E.g., measured as the distance between
- **Intra-cluster cohesion**
  - measure of the compactness of the cluster
  E.g., measured by the sum of squared error
  that quantifies the spread of the points around
  the centroid.



Clustering Algorithms:

- The K-mean algorithm partitions the data D points into K clusters.
- We assume that the user gives K
- Each cluster is a group of points
- Each cluster is denoted by $C_1, C_2, C_3 \ldots C_k$
- Each cluster is defined by it's center, known as centroid.
- Mathmatically the centroid is the mean of all points in cluster.

Algorithm

In K-means clustering algorithm, we carry out the following steps:

- - **Input:** K and Data D
- - **Randomly choose K cluster centres (centroids)**
- - **Repeat until convergence:**
  - Each data point is assigned membership of the cluster of the closest centroid
  - Compute the centroids again for each cluster using the current cluster memberships

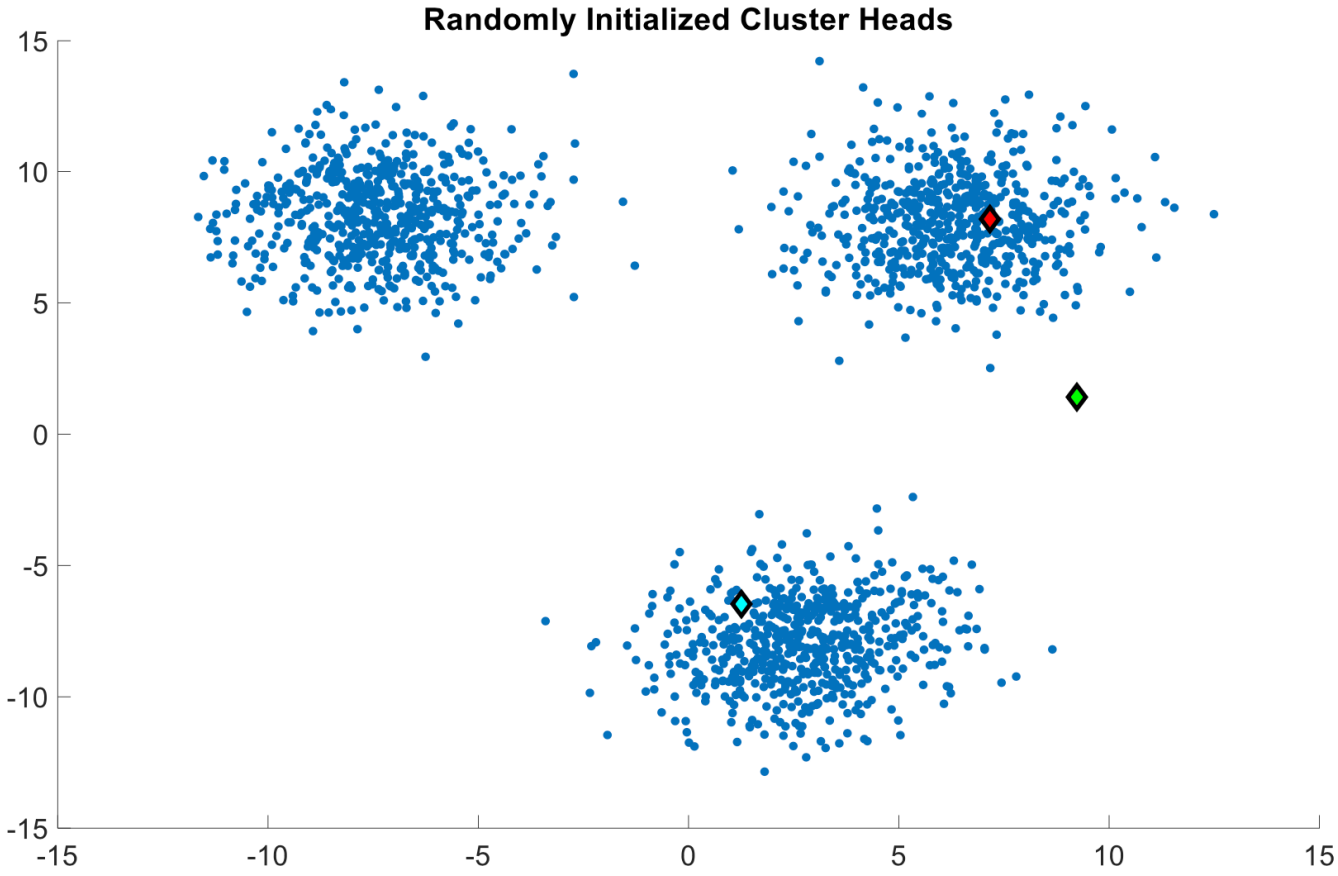
Process:

- Randomly choose $\mu(c_i)$ for $i = 1, 2, \ldots, K$

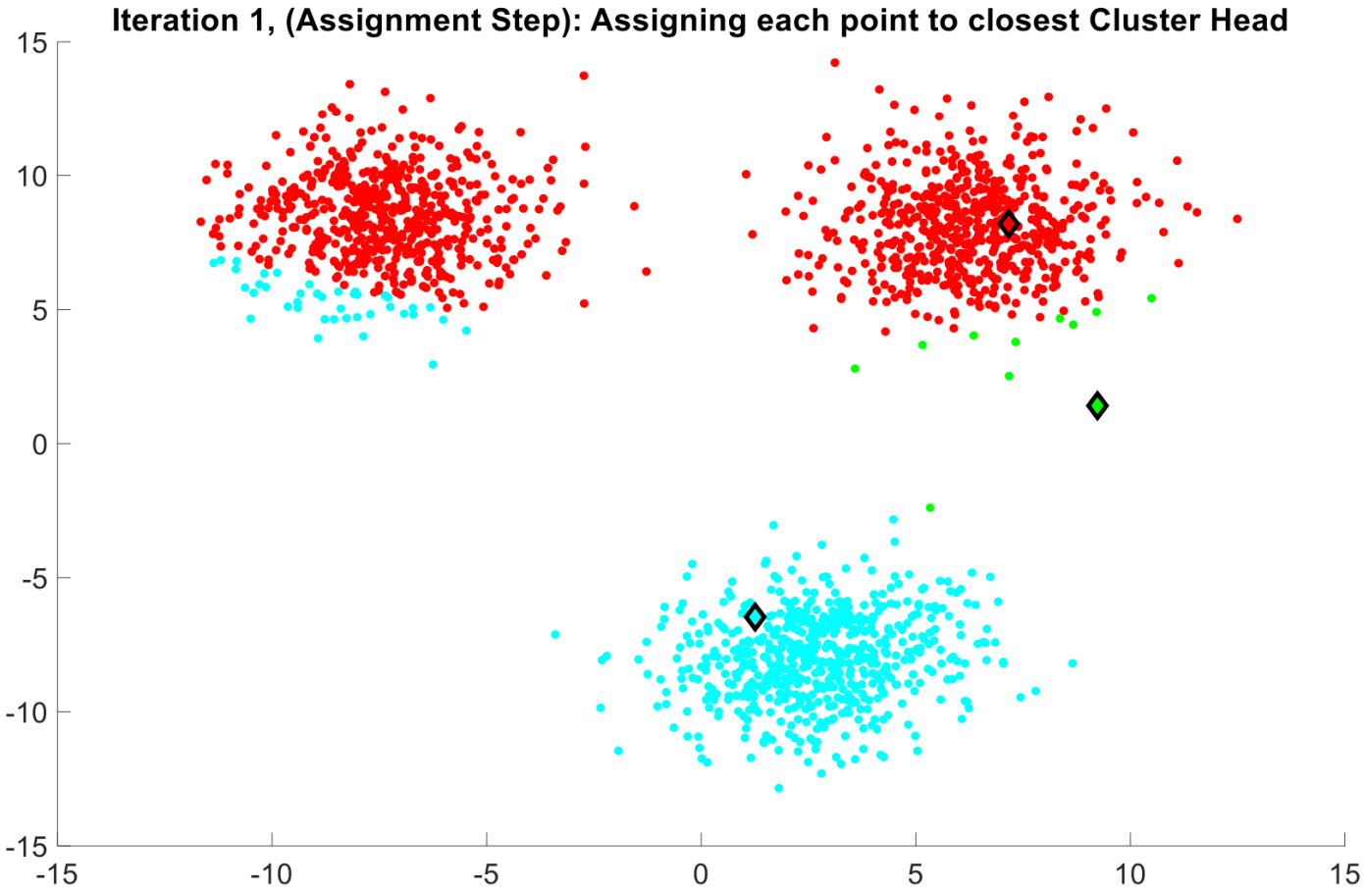
Repeat until convergence

- For each $\mathbf{x}_j$, assign the cluster c_i such that $\text{dist}(\mathbf{x}_j, \mu(c_i))$ is minimal.
- Recompute the centroids as follows

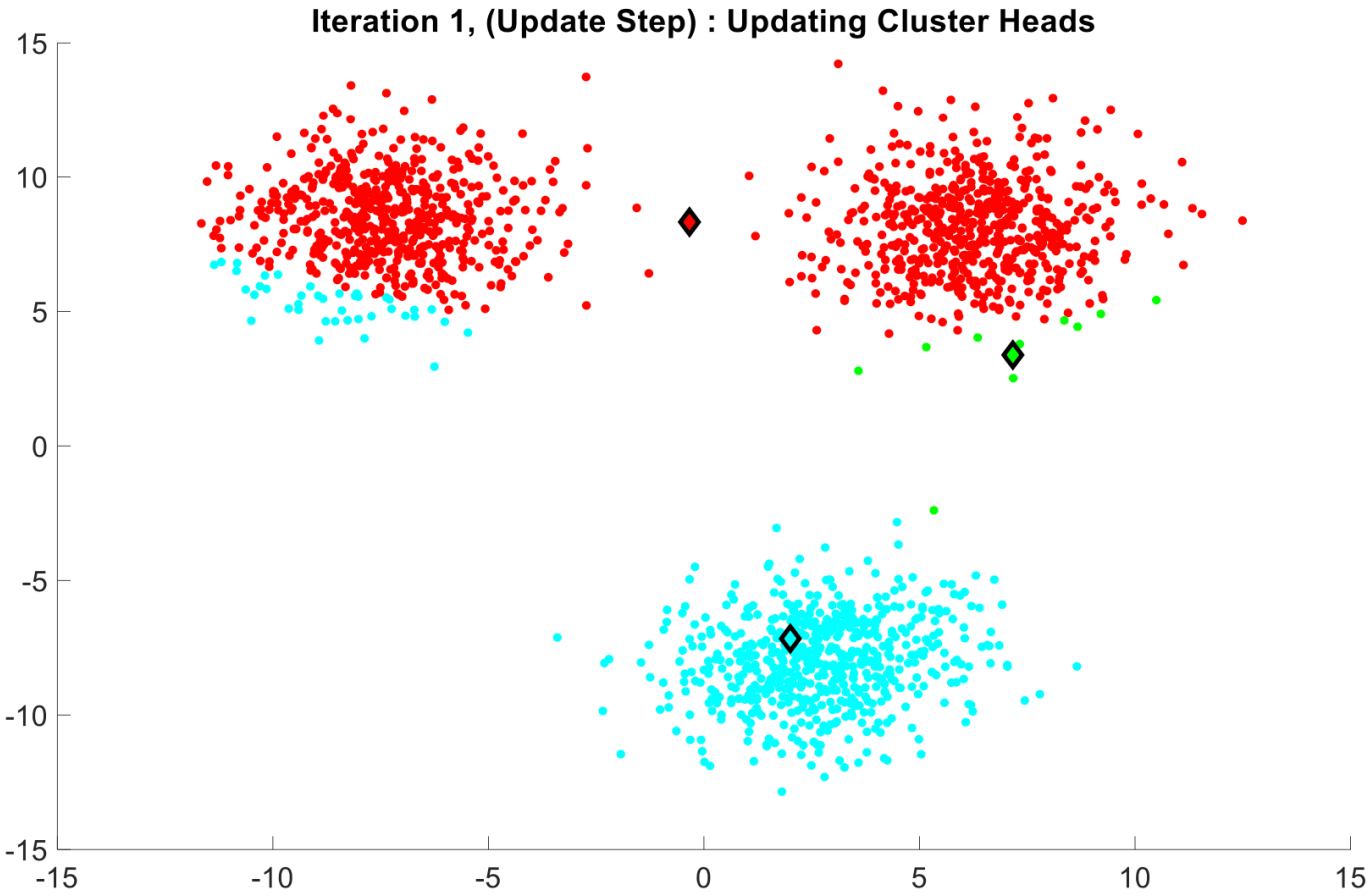
Illustration



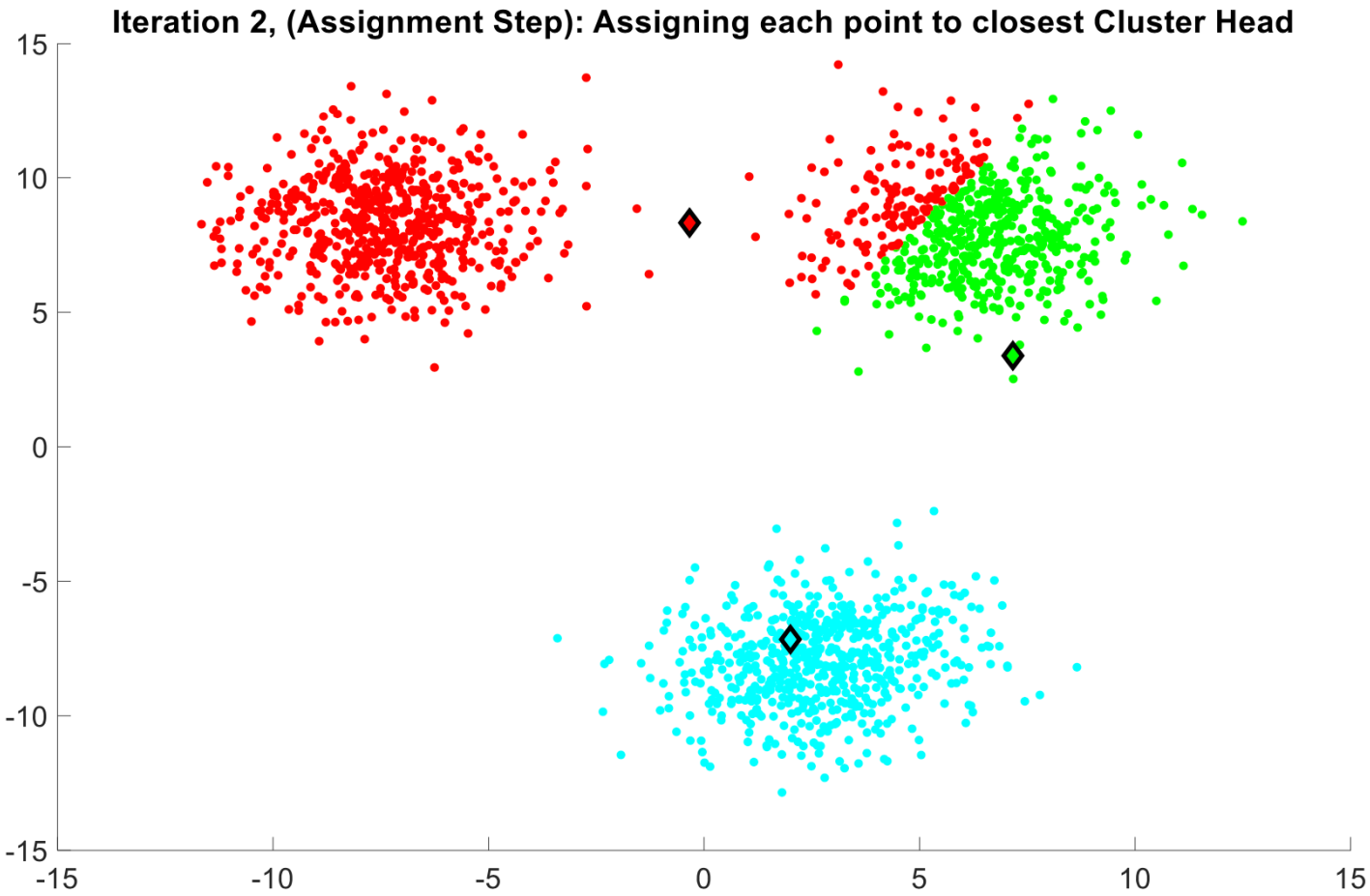
Illustration



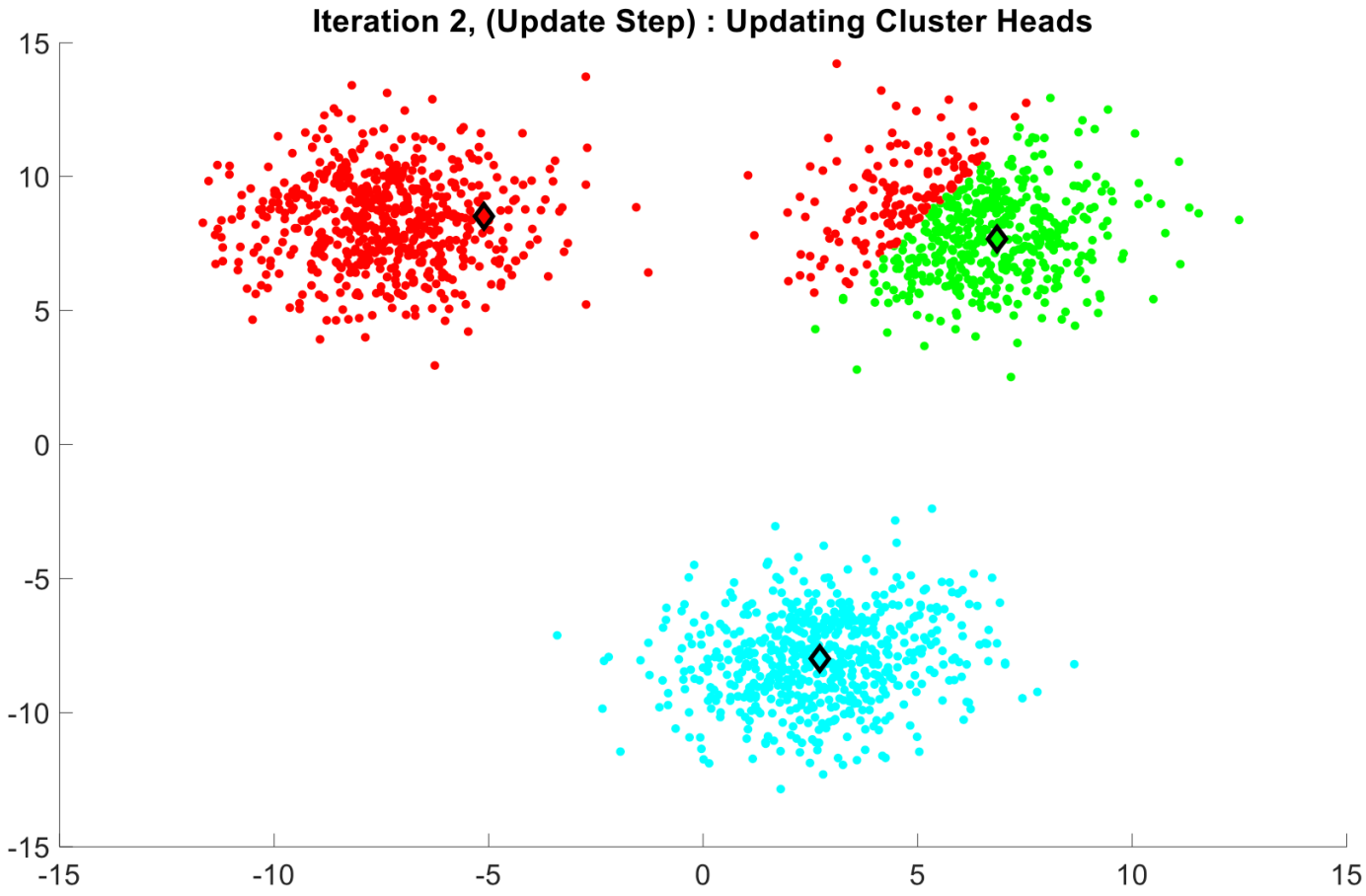
Illustration



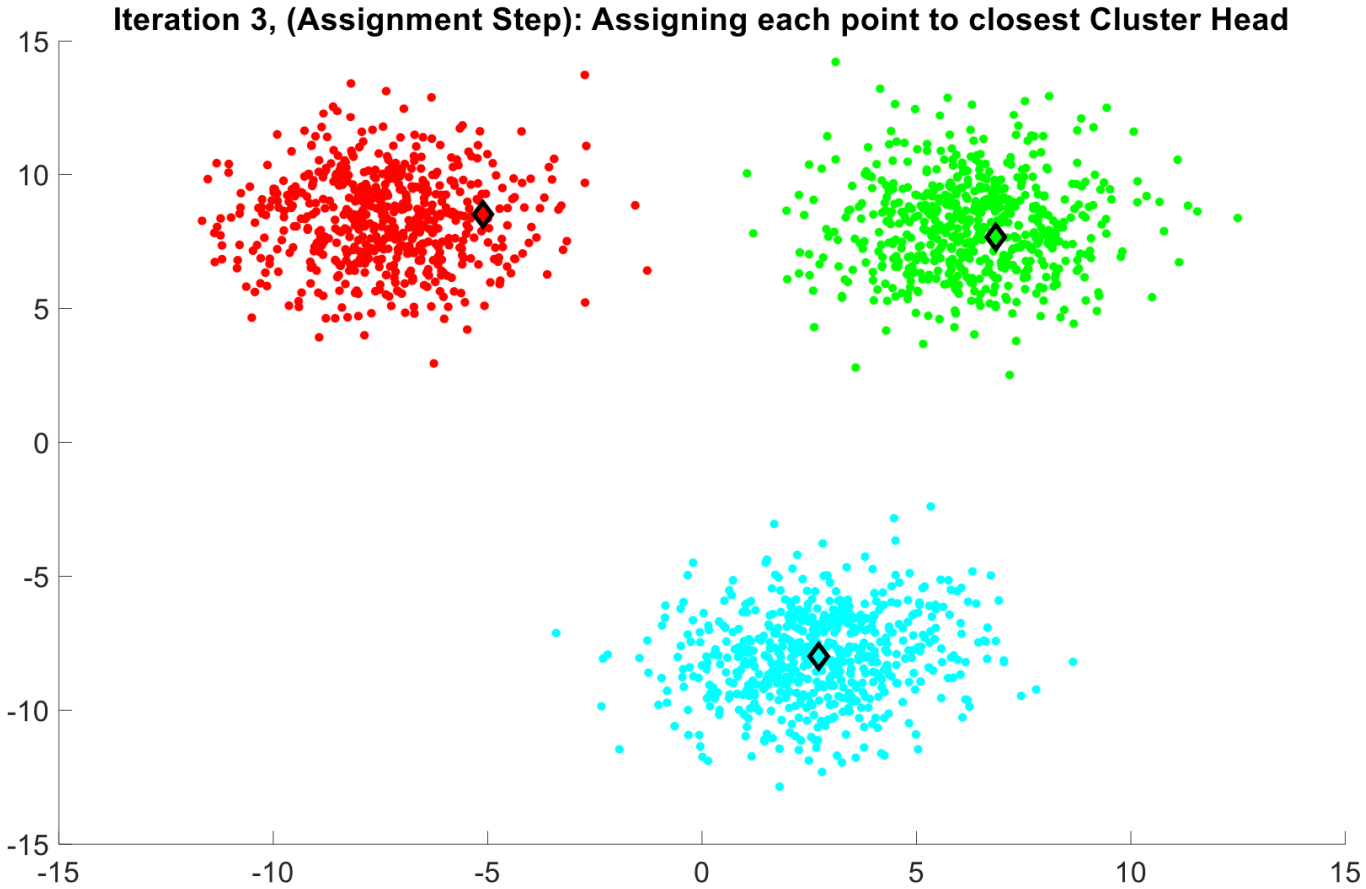
Illustration



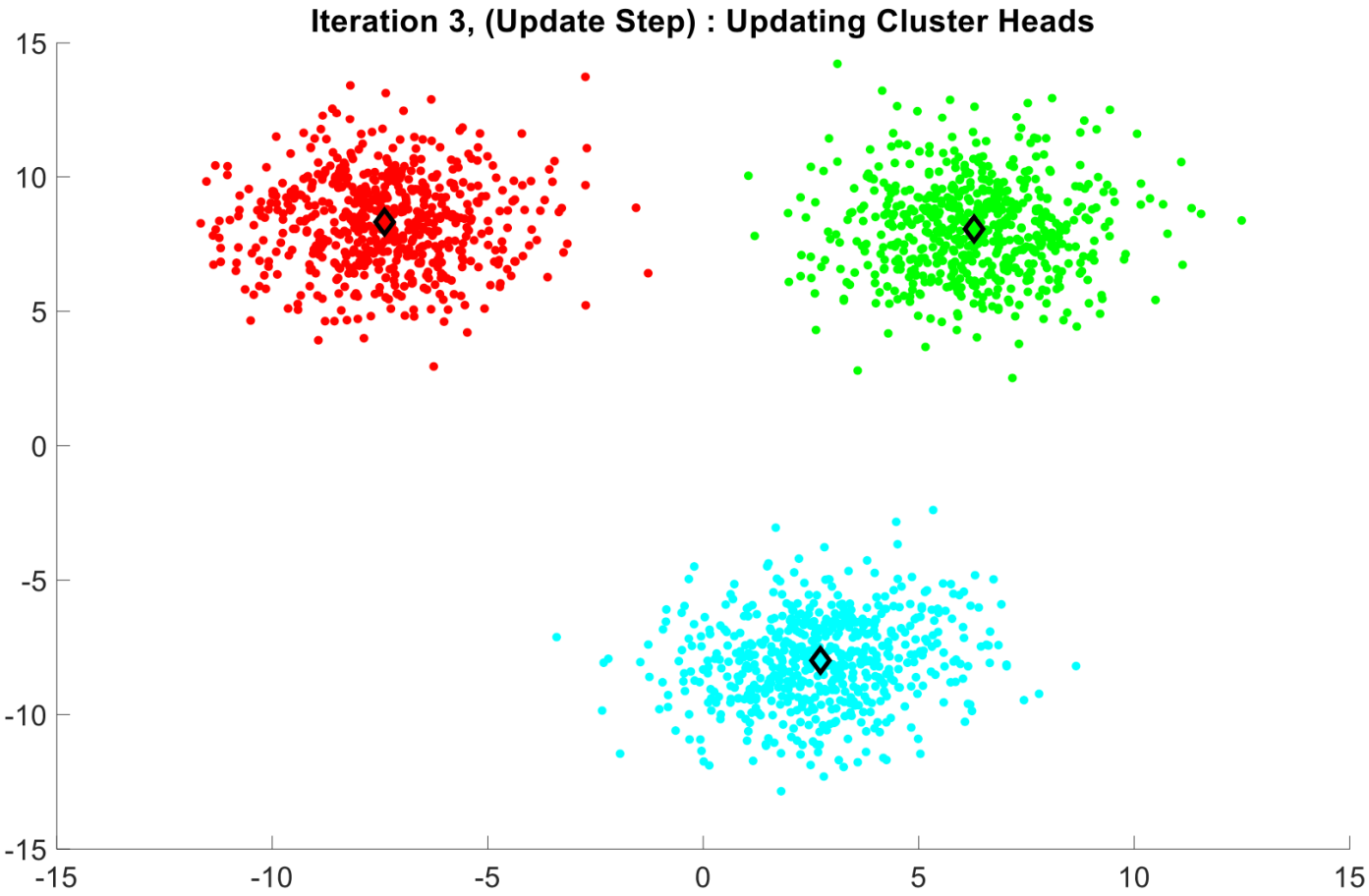
Illustration



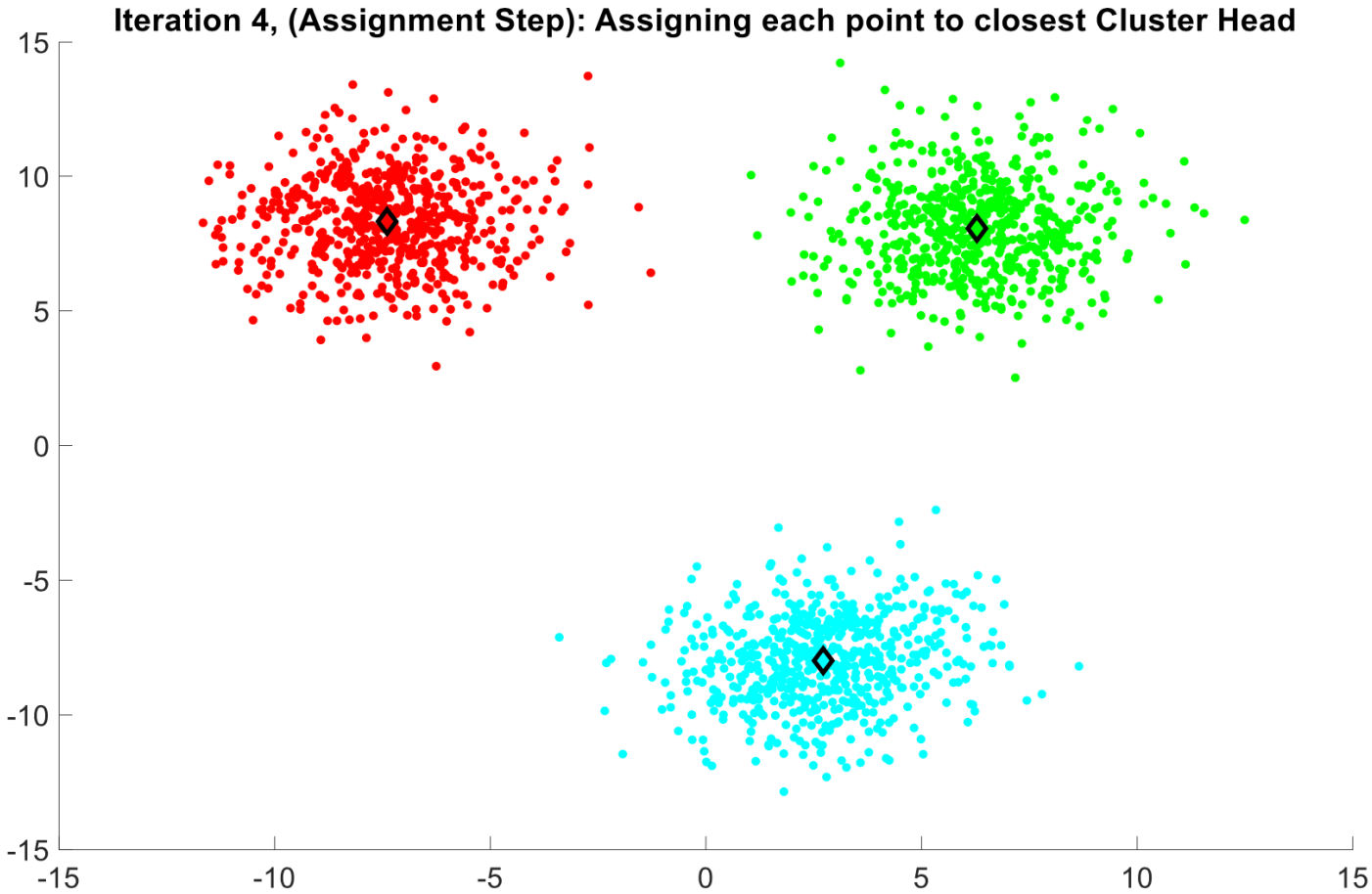
Illustration



Illustration



Illustration



Convergence

- Convergence of the re-assignment of data points to different clusters: re-assignment is stopped or minimized
- Convergence of location of centroids: minimum change in centroid position
- **Interpretation:** clusters are no more changing

Elbow Method

Step 1: Execute the K-means clustering on a given dataset for different K values (ranging from 1-10).

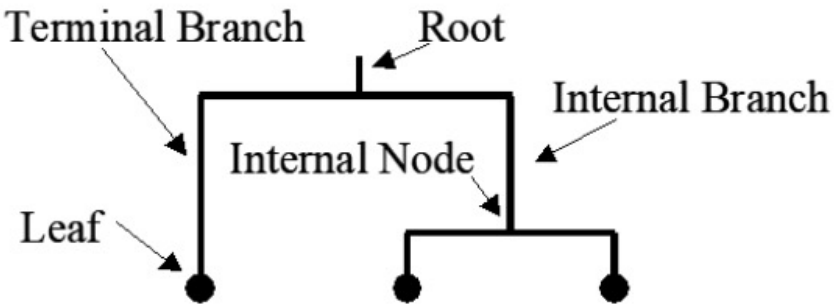
Step 2: For each value of K, calculate the WCSS value.

Step 3: Plot a graph/curve between WCSS values and the respective number of clusters K.

Step 4: The sharp point of bend or a point (looking like an elbow joint) of the plot, like an arm, will be considered as the best/optimal value of K.

Agglomerative Clustering

- In hierarchical clustering, we carry out a hierarchical decomposition of the data points using some criterion.
- Use distance or similarity metric to carry out hierarchical decomposition. We do not need to define the number of clusters as an input.
- A nested sequence of clusters is created in this decomposition process.
- This nested sequence of clusters, a tree, is also called **Dendrogram**.



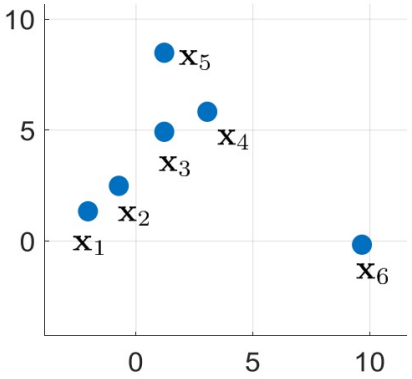
Overview

- A tree data structure, that records the sequences of splits or merges, used for the visualization of hierarchical clustering techniques.
- We represent the similarity between two data-points in the dendrogram as the height of the lowest internal node they share.
- Root corresponds to one cluster and leaf represents individual clusters.
- Each level of the tree shows clusters for that level.

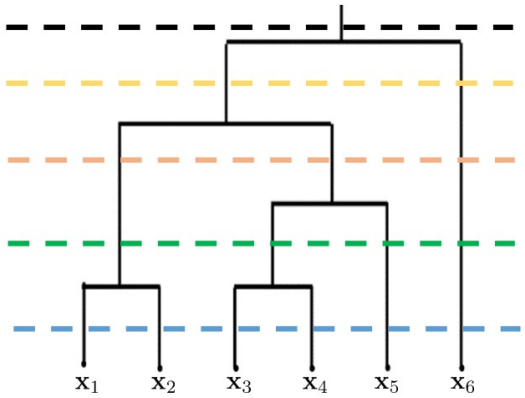
Dendrogram

- Dendrogram Nested Clusters: Dendrogram is the representation of nested clusters.
- We can cut the dendrogram at a desired level to carry out clustering; the connected data-points below the desired level form a cluster.

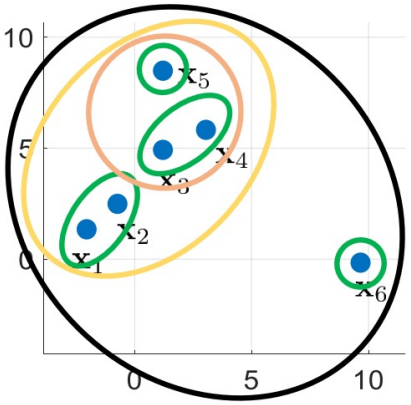
Data



Dendrogram



Nested Clusters



Applications

- **Marketing:** Clustering is used for segmentation of the customers/markets to do targeted marketing.
- **Sales Analysis:** It's used to understand where and when products sell the most based on factors like location and time.
- **Text Analysis:** Grouping of a collection of text documents with respect to similarity in their content. – Grouping of news items when you search for an item
- **Anomaly Detection:** Given data from the sensors, grouping of sensor readings for machines operating in different states and detect anomaly as an outlier.

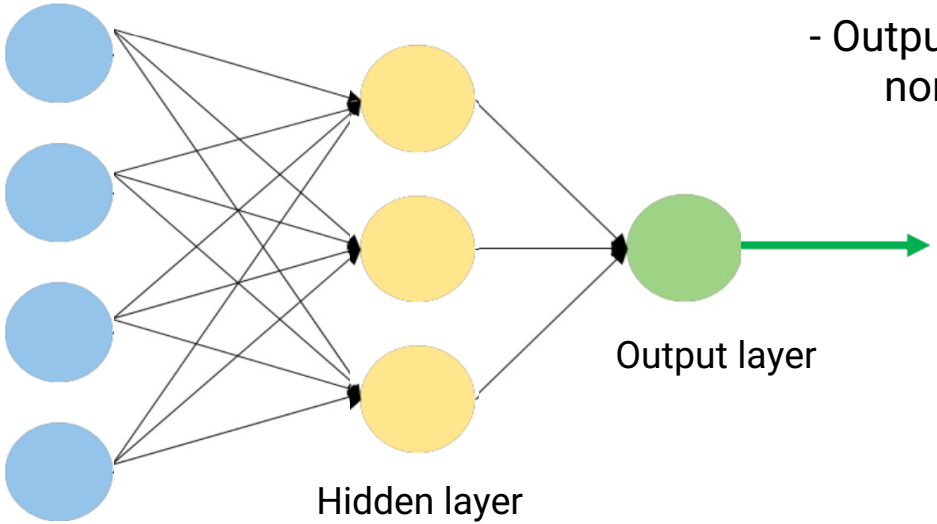
Neural Networks

Lecture 18

- Introduction to Neural Networks
- Neural network forward pass
- Activation functions

Neural Networks

A neural network is a set of neurons organized in layers

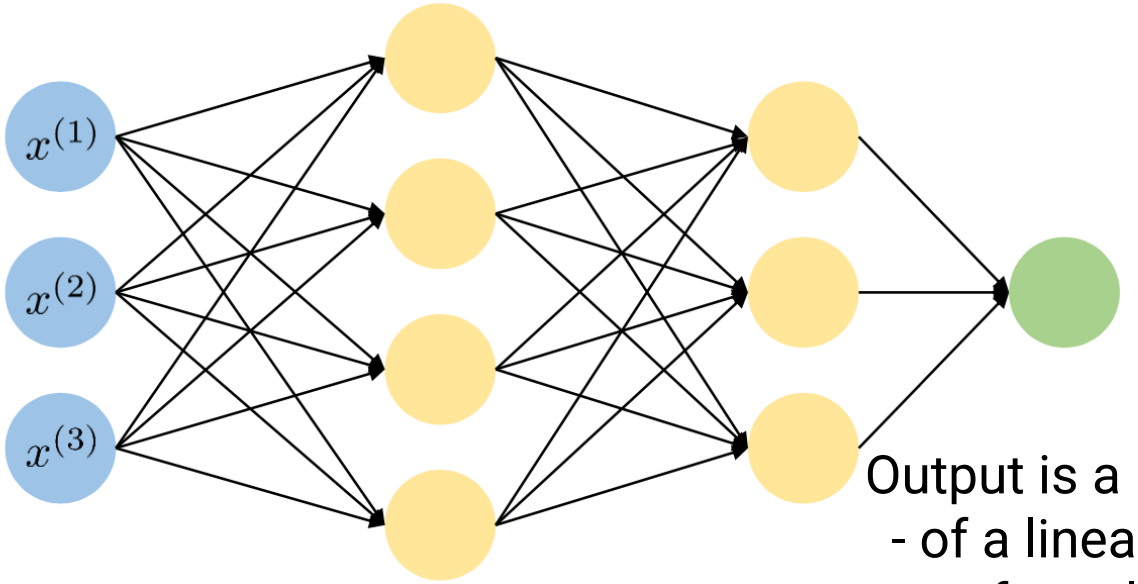


- Output is a
  non-linear function
  of a linear combination
  of non-linear functions
  of a linear combination of inputs

Each unit in the network is called a neuron and the network of neurons is called **Neural Network.**

- Given the input and parameters of the neurons, we can determine the output by traversing layers from input to output. This is referred to as **Forward Pass**.

- **Example: 3-layer network, 2 hidden layers**



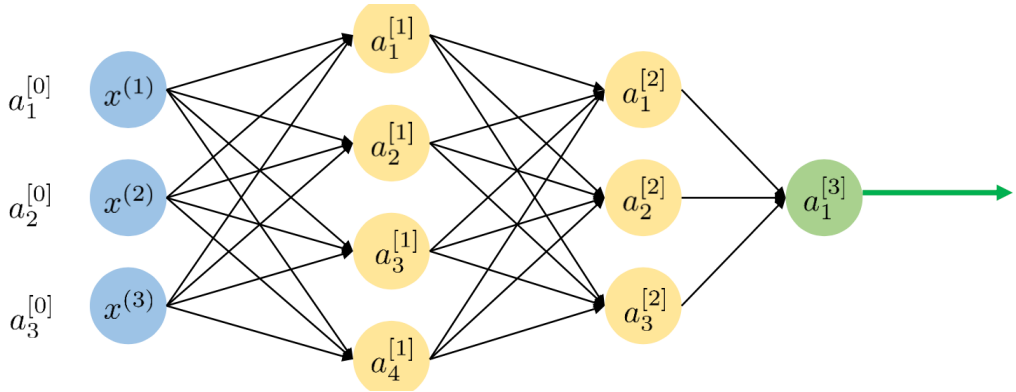
Output is a non-linear function
- of a linear combination
  - of non-linear functions
    - of linear combinations
      - of non-linear functions
        - of linear combinations of inputs

Feedforward Neural Network: Output from one layer is an input to the next layer.

- L - number of layers.
- $\mathbf{a}^{[\ell]} = \mathbf{x}$ input layer.
- $\mathbf{a}^{[L]} = y$ output layer.
- Number of nodes in the ℓ-th layer, $m^{[\ell]}$
- $\mathbf{a}^{[\ell]}$ - vector of outputs of ℓ-th node.
- $a_i^{[\ell]}$ denotes the output of i-th node in the ℓ-th layer.

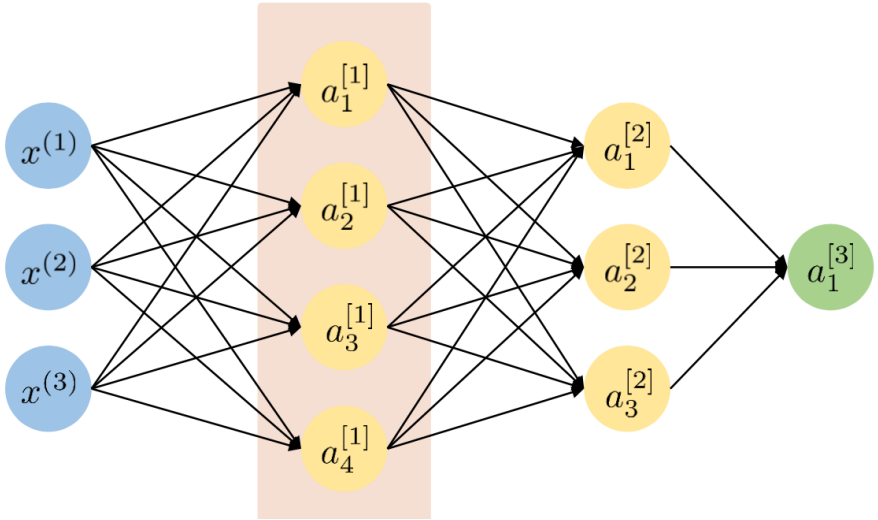
Example: 3-layer network, 2 hidden layers

- $L = 3$
- $m^{[1]} = 4,\ m^{[2]} = 3,\ m^{[3]} = 1$



- $\mathbf{w}_i^{[\ell]}$ and $b_i^{[\ell]}$ denote the weight and bias associated with the i-th node in the ℓ-th layer respectively.
- $w_{i,j}^{[\ell]}$ denote the weight and bias associated with the $j-th$ input of the i-th node in the ℓ-th layer respectively.

Example: 3-layer network, 2 hidden layers



Layer 1 output

$$a_1^{[1]} = g(z_1^{[1]}), \quad z_1^{[1]} = \mathbf{w}_1^{[1]T}\mathbf{x} + b_1^{[1]}$$

$$a_2^{[1]} = g(z_2^{[1]}), \quad z_2^{[1]} = \mathbf{w}_2^{[1]T}\mathbf{x} + b_2^{[1]}$$

$$a_3^{[1]} = g(z_3^{[1]}), \quad z_3^{[1]} = \mathbf{w}_3^{[1]T}\mathbf{x} + b_3^{[1]}$$

$$a_4^{[1]} = g(z_4^{[1]}), \quad z_4^{[1]} = \mathbf{w}_4^{[1]T}\mathbf{x} + b_4^{[1]}$$

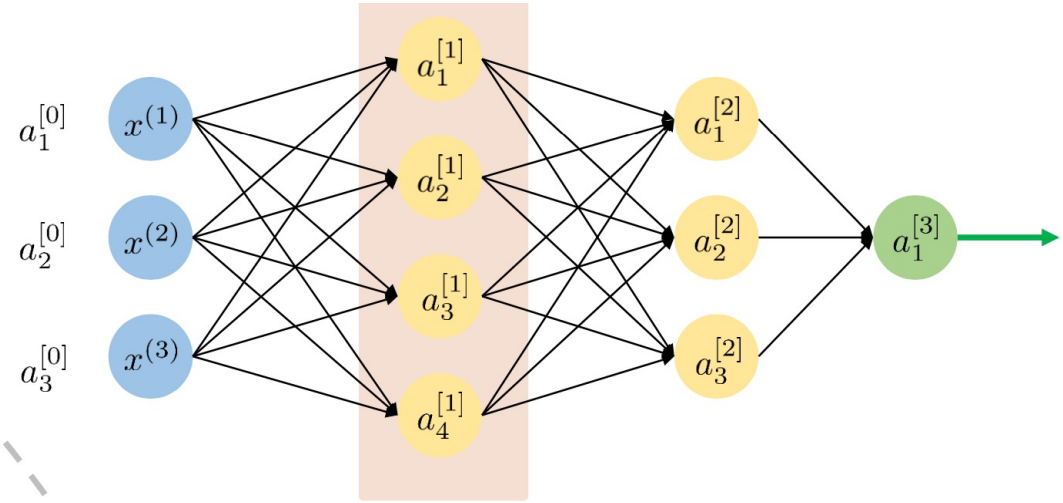
Layer 1 output

$$a_1^{[1]} = g(z_1^{[1]}), \quad z_1^{[1]} = {\mathbf{w}_1^{[1]}}^T \mathbf{x} + b_1^{[1]}$$

$$a_2^{[1]} = g(z_2^{[1]}), \quad z_2^{[1]} = {\mathbf{w}_2^{[1]}}^T \mathbf{x} + b_2^{[1]}$$

$$a_3^{[1]} = g(z_3^{[1]}), \quad z_3^{[1]} = {\mathbf{w}_3^{[1]}}^T \mathbf{x} + b_3^{[1]}$$

$$a_4^{[1]} = g(z_4^{[1]}), \quad z_4^{[1]} = {\mathbf{w}_4^{[1]}}^T \mathbf{x} + b_4^{[1]}$$



$$\mathbf{W}^{[1]} = \begin{bmatrix} {\mathbf{w}_1^{[1]}}^T \\ {\mathbf{w}_2^{[1]}}^T \\ {\mathbf{w}_3^{[1]}}^T \\ {\mathbf{w}_4^{[1]}}^T \end{bmatrix} \quad \mathbf{b}^{[1]} = \begin{bmatrix} {b_1^{[1]}}^T \\ {b_2^{[1]}}^T \\ {b_3^{[1]}}^T \\ {b_4^{[1]}}^T \end{bmatrix} \quad \mathbf{z}^{[1]} = \begin{bmatrix} {z_1^{[1]}}^T \\ {z_2^{[1]}}^T \\ {z_3^{[1]}}^T \\ {z_4^{[1]}}^T \end{bmatrix}$$

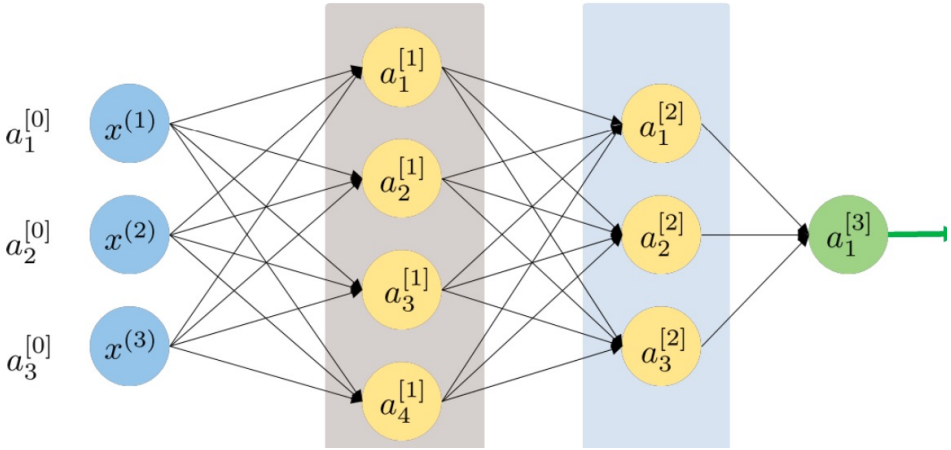
$$\mathbf{a}^{[1]} = g(\mathbf{z}^{[1]}), \quad \mathbf{z}^{[1]} = \mathbf{W}^{[1]}\mathbf{x} + \mathbf{b}^{[1]}$$

$$\mathbf{z}^{[1]} = \mathbf{W}^{[1]}\mathbf{a}^{[0]} + \mathbf{b}^{[1]}$$

- $\mathbf{W}^{[1]}$ and $\mathbf{b}^{[1]}$ are the parameters of the first layer.

$$\mathbf{a}^{[1]} = g(\mathbf{z}^{[1]}), \quad \mathbf{z}^{[1]} = \mathbf{W}^{[1]}\mathbf{x} + \mathbf{b}^{[1]}$$

$$\mathbf{z}^{[1]} = \mathbf{W}^{[1]}\mathbf{a}^{[0]} + \mathbf{b}^{[1]}$$



- Q. What is the size of $\mathbf{W}^{[1]}$?

A. No. of nodes $\times$ No. of inputs. 4×3

No. of nodes $\times$ No. nodes in the previous layer.

- Q. Can we write output of second layer using the notation we have defined?

$$\mathbf{a}^{[2]} = g(\mathbf{z}^{[2]}), \quad \mathbf{z}^{[2]} = \mathbf{W}^{[2]}\mathbf{a}^{[1]} + \mathbf{b}^{[2]}$$

- Q. What is the size of $\mathbf{W}^{[2]}$? 3×4
- Q. What is the size of $\mathbf{W}^{[3]}$? 1×3

$$\mathbf{a}^{[3]} = g(\mathbf{z}^{[3]}), \quad \mathbf{z}^{[3]} = \mathbf{W}^{[3]}\mathbf{a}^{[2]} + \mathbf{b}^{[3]}$$

$$\mathbf{a}^{[1]} = g(\mathbf{z}^{[1]}), \quad \mathbf{z}^{[1]} = \mathbf{W}^{[1]}\mathbf{x} + \mathbf{b}^{[1]}$$

$$\mathbf{z}^{[1]} = \mathbf{W}^{[1]}\mathbf{a}^{[0]} + \mathbf{b}^{[1]}$$

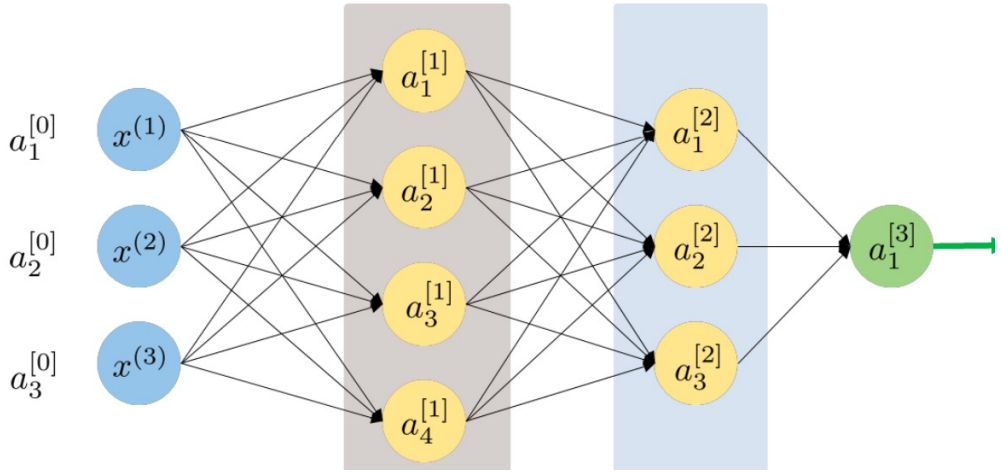
$$(4 \times 1) = (4 \times 3)(3 \times 1) + (4 \times 1)$$

$$\mathbf{a}^{[2]} = g(\mathbf{z}^{[2]}), \quad \mathbf{z}^{[2]} = \mathbf{W}^{[2]}\mathbf{a}^{[1]} + \mathbf{b}^{[2]}$$

$$(3 \times 1) = (3 \times 4)(4 \times 1) + (3 \times 1)$$

$$\mathbf{a}^{[3]} = g(\mathbf{z}^{[3]}), \quad \mathbf{z}^{[3]} = \mathbf{W}^{[3]}\mathbf{a}^{[2]} + \mathbf{b}^{[3]}$$

$$(1 \times 1) = (1 \times 3)(3 \times 1) + (1 \times 1)$$



- How many parameters do we have by the way?

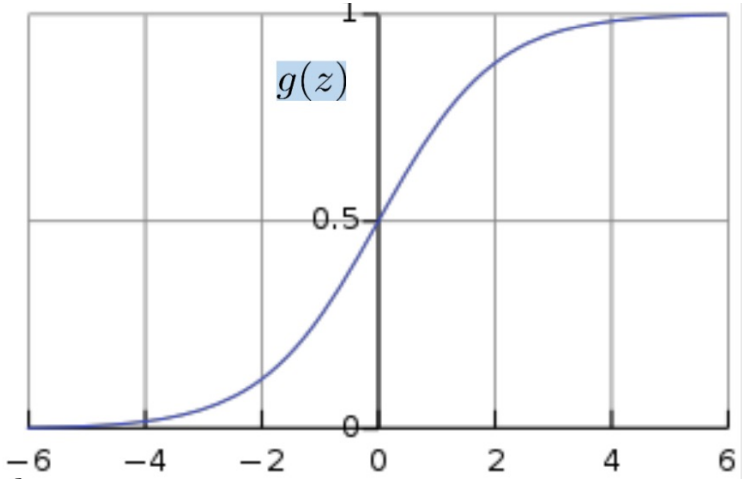
Neural Networks

Lecture 18

- Introduction to Neural Networks
- Neural network forward pass
- **Activation functions**

Sigmoid

$$g(z) = \sigma(z) = \frac{1}{1 + e^{-z}}$$ (Sigmoid: because of S shaped curve)



- Squishes the input between (0,1)
- Suitable for output neuron when classification is applied
- We have already used this function: Logistic regression

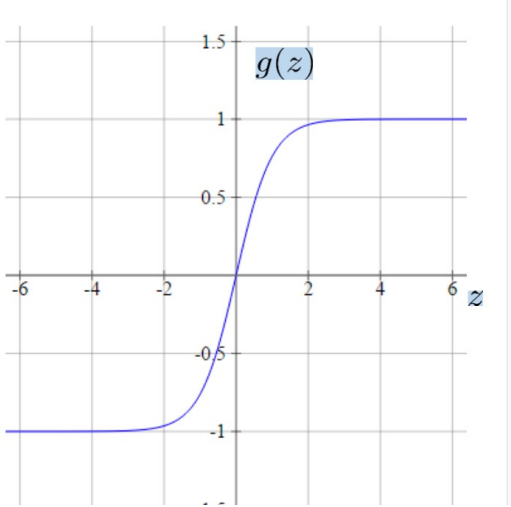
Issues

- **Vanishing Gradient:** As z becomes very large or very small, the sigmoid function approaches 1 or 0
- **Output Not Zero-Centric**: it can slow down training. Neural networks typically learn better when the outputs are centered around zero because this facilitates faster convergence during training.

tanh

$$g(z) = \tanh(z) = 2\sigma(2z) - 1 = \frac{2}{1 + e^{-2z}} - 1 = \frac{e^{z} - e^{-z}}{e^{z} + e^{-z}}$$

- Squishes the input between (-1,1)
- Suitable for output neurons when classification is applied

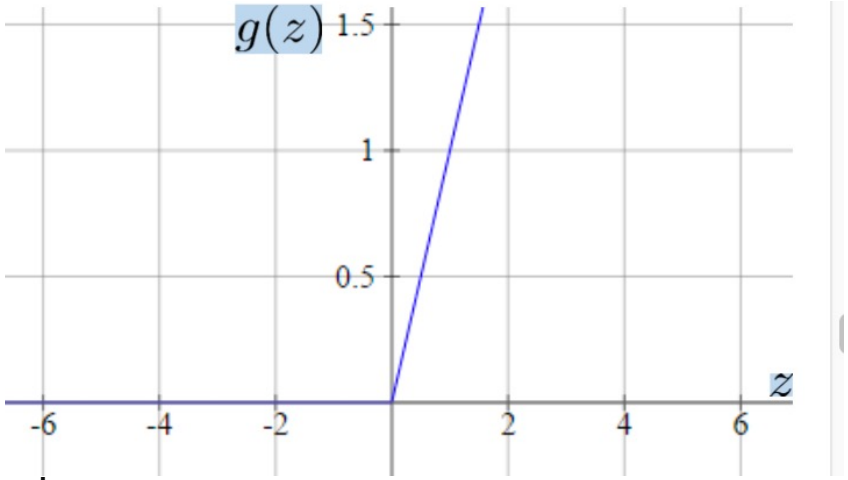


Issues

- **Vanishing Gradient:** As z becomes very large or very small, the sigmoid function approaches 1 or 0
- **Output Zero-Centric**: The Network will learn better because the output is zero-centric, it can solve the problem of sigmoid.

ReLU

$$g(z) = \max(0, z) = \begin{cases} z & z \geq 0 \\ 0 & z < 0 \end{cases}$$



The most used activation function

Why?

- Unlike Sigmoid and tanh that required the computation of exponents, easier to compute
- It only activates the output is non-negative.
- It deactivates the neurons with values lass than 0.