# Algorithms and Data Structures with Applications in Machine Learning

Graph Representation Learning

December 30, 2024

# Outline

# Outline

## Definition

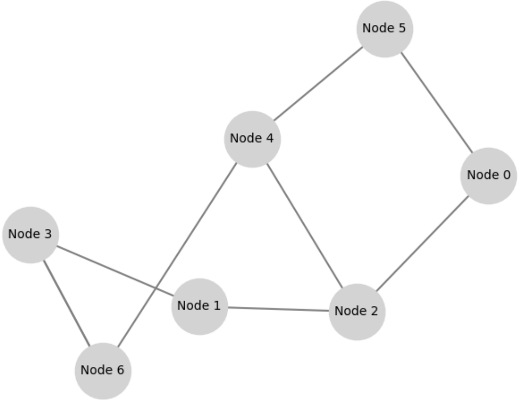
A graph is defined as:

$$G = (V, E, u)$$

- **Nodes (Vertices):** The set $V$ represents the nodes in the graph.
- **Edges:** The set $E \subseteq V \times V$ represents the connections (relationships) between the nodes.
- **Features:** Each node can have a feature vector $u(v)$ representing its attributes.
- **Labels:** Nodes (or edges) can also have labels, which are used for tasks like classification.

# Example Graph

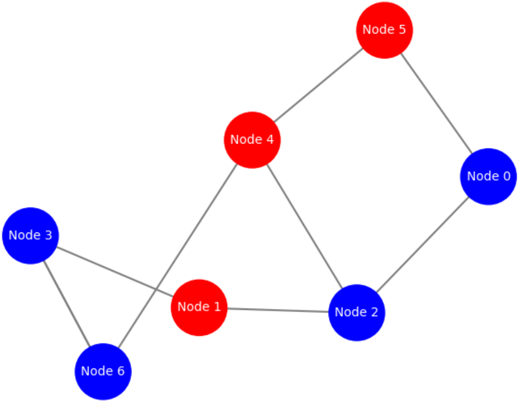
**Example:** The graph below has 7 connected nodes ($V = \{0, 1, 2, 3, 4, 5, 6\}$) and their edges ($E$).



# Example Graph: Node Labels

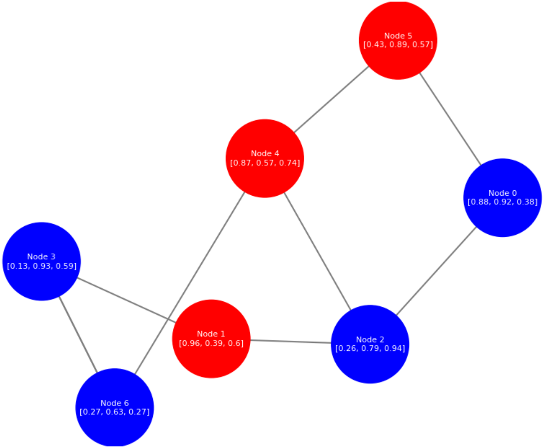
**Example:** Nodes in a graph can be associated with labels.

**Blue nodes**: Label 0 **Red nodes**: Label 1



# Example Graph: Node Features

**Example:** Each node in the graph can have associated features. In this case: Each node has a feature vector of dimension 3.

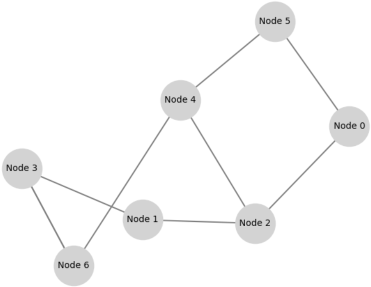


**Definition**

The adjacency matrix $A$ of a graph $G = (V, E)$ is a matrix of size $|V| \times |V|$, where:

- $A[i][j] = 1$ if there is an edge between node $i$ and node $j$.
- $A[i][j] = 0$ if there is no edge between node $i$ and node $j$.

**Example:** A graph and its corresponding adjacency matrix:




**Adjacency Matrix:**

$$A = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 1 & 1 \\ 1 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 \end{bmatrix}$$

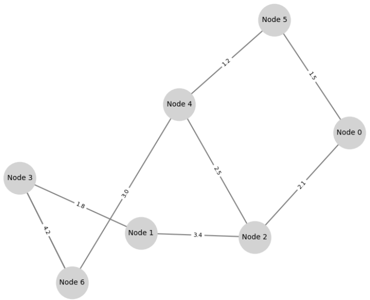
# Weighted Adjacency Matrix

**Definition**

The adjacency matrix $A$ can be extended to a weighted matrix $W$, where:

- $W[i][j]$ represents the weight of the edge between node $i$ and node $j$.

**Example:** A graph and its a weighted adjacency matrix:




**Weighted Matrix:**

$$W = \begin{bmatrix} 0 & 0 & 2.1 & 0 & 0 & 1.5 & 0 \\ 0 & 0 & 3.4 & 1.8 & 0 & 0 & 0 \\ 2.1 & 3.4 & 0 & 0 & 2.5 & 0 & 0 \\ 0 & 1.8 & 0 & 0 & 0 & 0 & 4.2 \\ 0 & 0 & 2.5 & 0 & 0 & 1.2 & 3.0 \\ 1.5 & 0 & 0 & 0 & 1.2 & 0 & 0 \\ 0 & 0 & 0 & 4.2 & 3.0 & 0 & 0 \end{bmatrix}$$

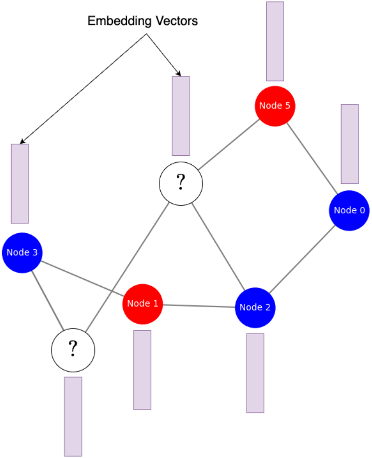
**Applications:** Machine Learning on graphs enables a variety of tasks, including:

- **Node Prediction:** Predict properties or labels of nodes in a graph (e.g., user classification in social networks).
- **Link Prediction:** Predict the existence or strength of a connection between two nodes (e.g., recommendation systems).
- **Graph Classification:** Assign labels to entire graphs (e.g., chemical compound classification).
- **Clustering:** Group nodes into communities or clusters based on their properties or structure.

# Objective: Node Classification

**Objective:** The objective of this course is two-fold:

1. **Learning a *D*-dimensional representation:**
   Create embedding vectors for nodes that capture the structure of the graph.
2. **Node Classification:**
   Use the learned embeddings to predict the labels of the nodes.



# Outline

# Outline

- Graph Terminology and Representation
- Graph Representation Learning: DeepWalk and Node2Vec
- Graph Neural Networks
- Application: Node Classification on Cora Dataset

# Outline

Graph Terminology and Representation

Graph Representation Learning: DeepWalk and Node2Vec

Graph Neural Networks

Application: Node Classification on Cora Dataset

Thank you for your attention