Machine Learning Tasks on Graphs

ACMS 80770: Deep Learning with Graphs

Instructor: Navid Shervani-Tabar

Department of Applied and Comp Math and Stats

University of Notre Dame

Learning

❖ In data science, we often deal with problems that predict a target variable $\boldsymbol{y}$ given an input variable $\boldsymbol{x}$.

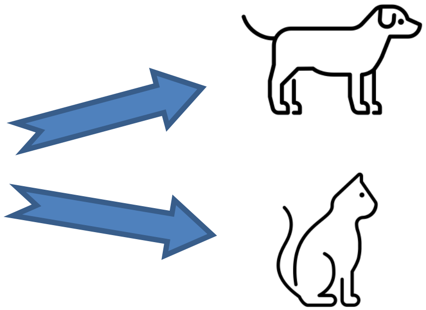
❖ Traditionally, machine learning models address this by learning a map from input data to target labels.

➢ Regression

- Advertising spending and revenue.

➢ Classification

- Predict image categories.

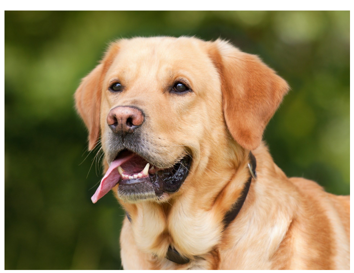


Learning

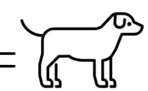
- You use a dataset $D = \{x^{(i)}, y^{(i)}\}_{i \leq \mathrm{N}}$ of N data points $x^{(i)}$ and their corresponding labels $y^{(i)}$ to train your model.
  - Regression

$\rightarrow \boxed{f} \rightarrow$

  - Classification



$\rightarrow \boxed{f} \rightarrow$

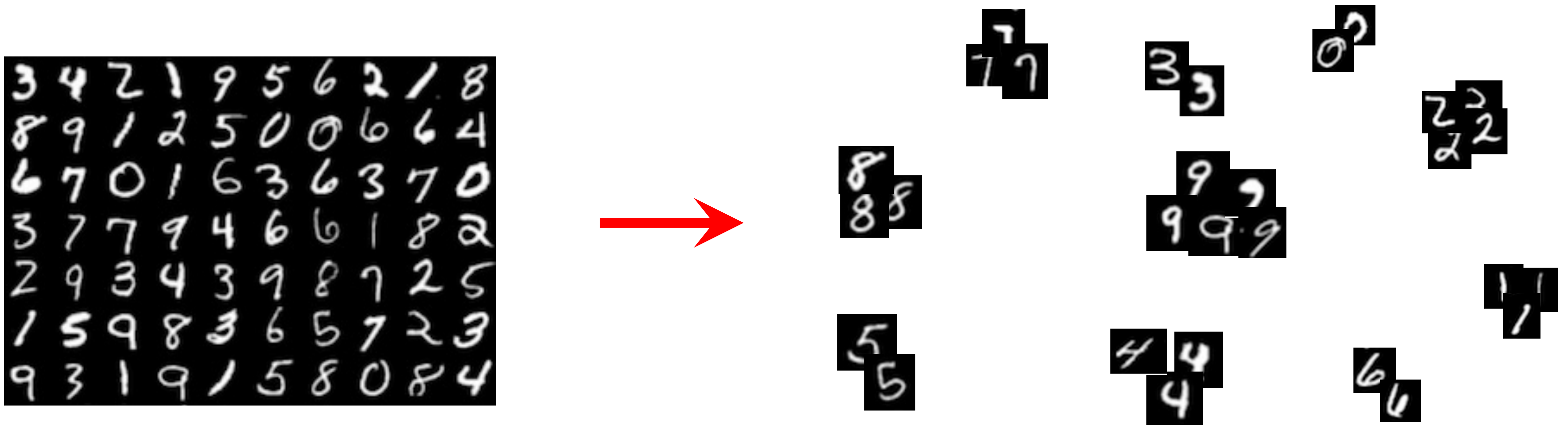
$P(X =$  $)$

$P(X =$  $)$

- This approach is called supervised learning.

Learning

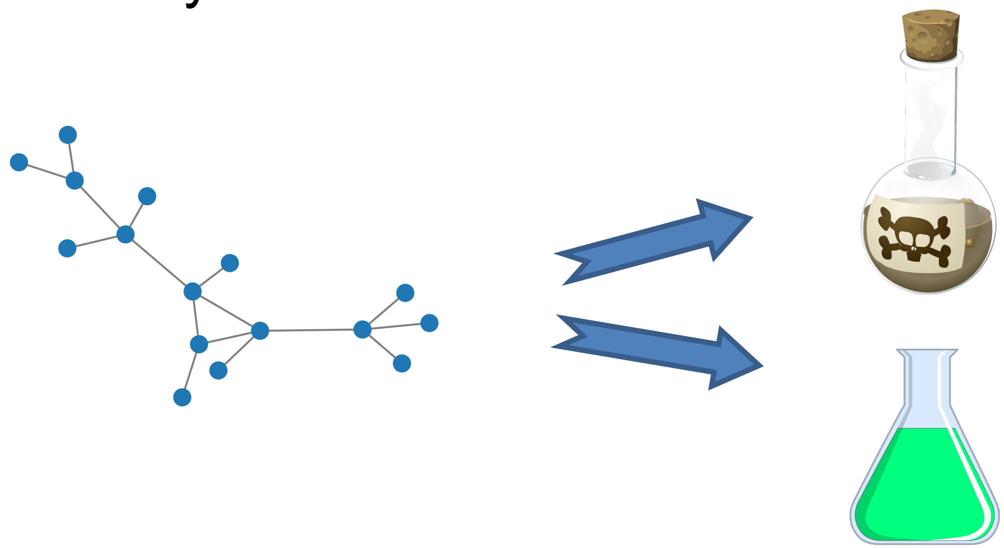
- In some problems, we don't have access to labels.
- Cluster
  - Classify MNIST digits without labels.



- The model is trained using a dataset $D = \{x^{(i)}\}_{i \leq \mathrm{N}}$ of N data points $x^{(i)}$ without any labels.
- This approach is called unsupervised learning.

Learning on Graphs

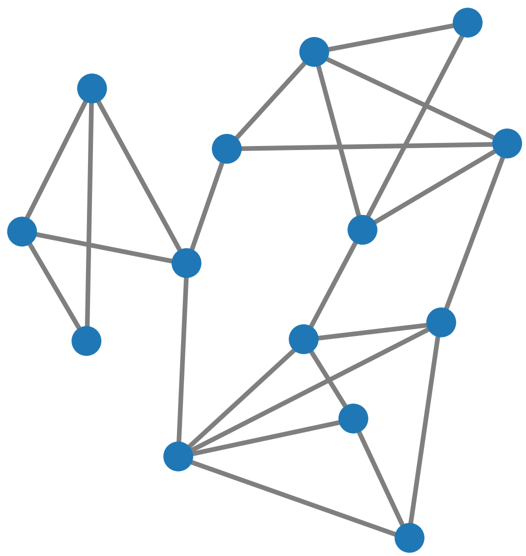
- Similar ML approaches can learn over graphs.
- Classification
  - Predict toxicity of chemicals.



- In this example, the model is trained using a dataset $D = \{G^{(i)}, y^{(i)}\}_{i \leq \mathrm{N}}$ of N graphs $G^{(i)}$, which are independent, and the corresponding labels $y^{(i)}$.

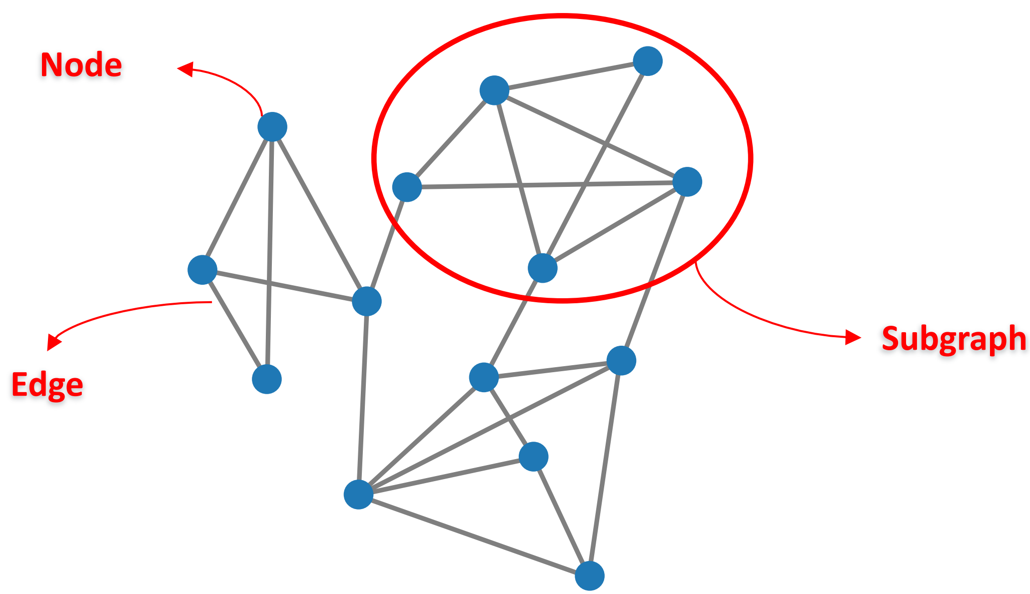
Learning on Graphs

- Extension of these ideas to graph domain considers each graph as an independent and identically distributed sampled datapoint.
- Graph ML consists of:
- Learning over a dataset of graphs.
- Learning within a single graph.



Learning on Graphs

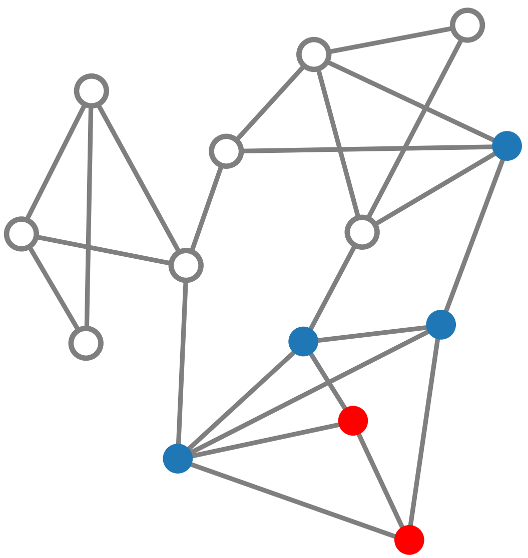
- Graph ML tasks are in the level of nodes, edges, subgraphs, or graphs.
- Within-graph tasks are based on data that are not independently and identically distributed.
- Traditional machine learning is equivalent to graph-level tasks.



Node-Level Tasks

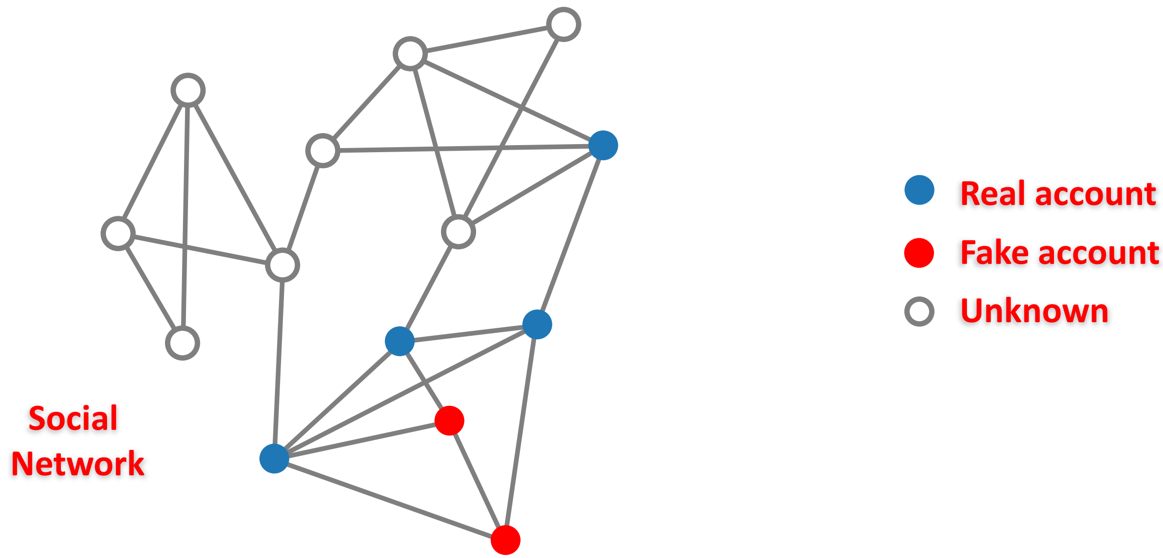
- Node Classification:
- In this problem, we learn to predict labels associated with each node in the graph.
- Such model is trained on a graph and its partial node attributes, i.e. training dataset is defined as

$$D = \{V, E, \{f_u | u \in \mathcal{A}, \mathcal{A} \subset V\}\}$$



Node-Level Tasks

- ➢ Fake accounts problem:
- ❖ Classify users of a social network website to fake or real users given labels for a small portion of users.



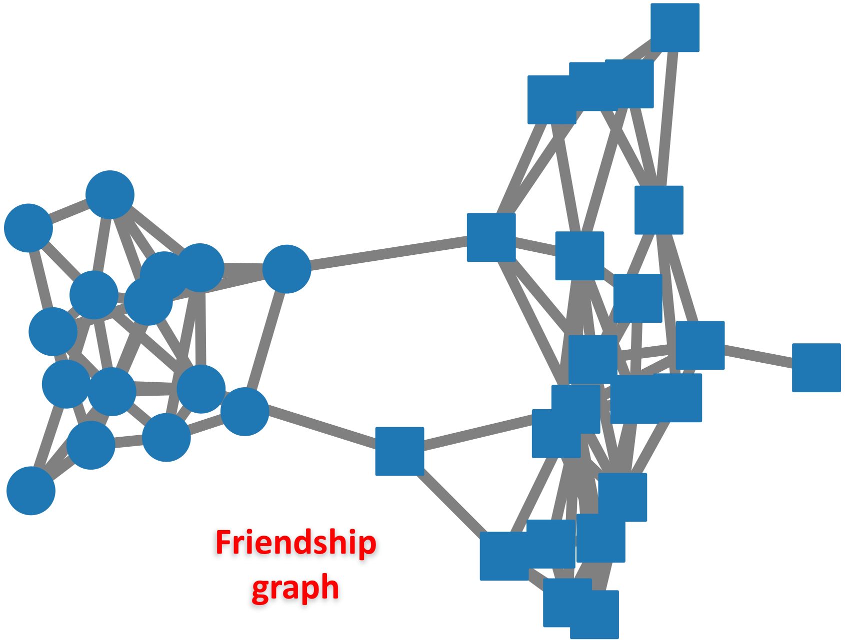
Node-Level Tasks

- Instead of node attributes, their connections are leveraged to yield predictions.
- General strategies to tackle node classification include:
- **Homophily**
- **Equivalence**
- **Heterophily**

Node-Level Tasks

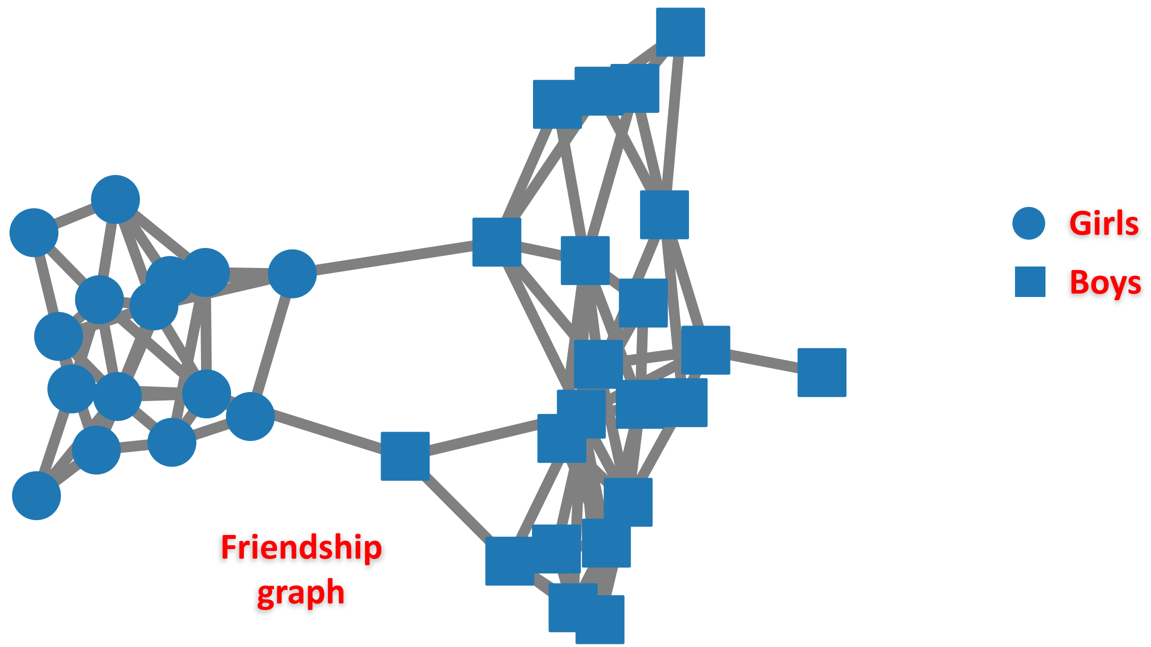
- **Homophily:**

- Elementary school friendship network:



Node-Level Tasks

- **Homophily:** Tendency to associate with similar others.
- Birds of a feather flock together.
- Citation network: same field.
- Elementary school friendship network: gender.



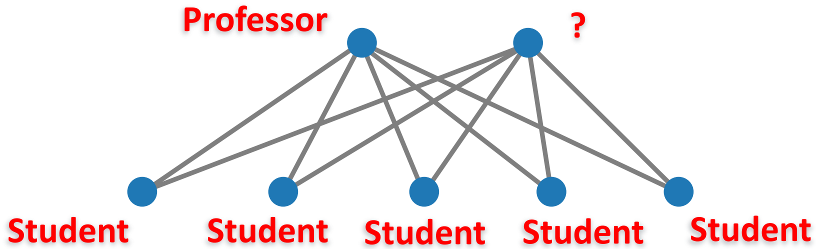
Node-Level Tasks

- **Equivalence:**
- Nodes with similar neighborhood structure tend to have similar features.
  - Structural equivalence
  - Regular equivalence

Node-Level Tasks

- **Equivalence:**
- Nodes with similar neighborhood structure tend to have similar features.

➤ Structural equivalence

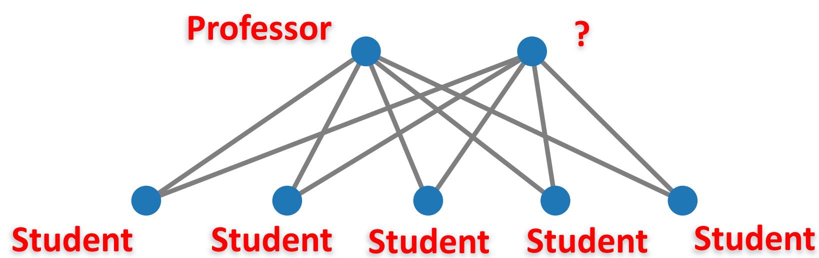


➤ Regular equivalence

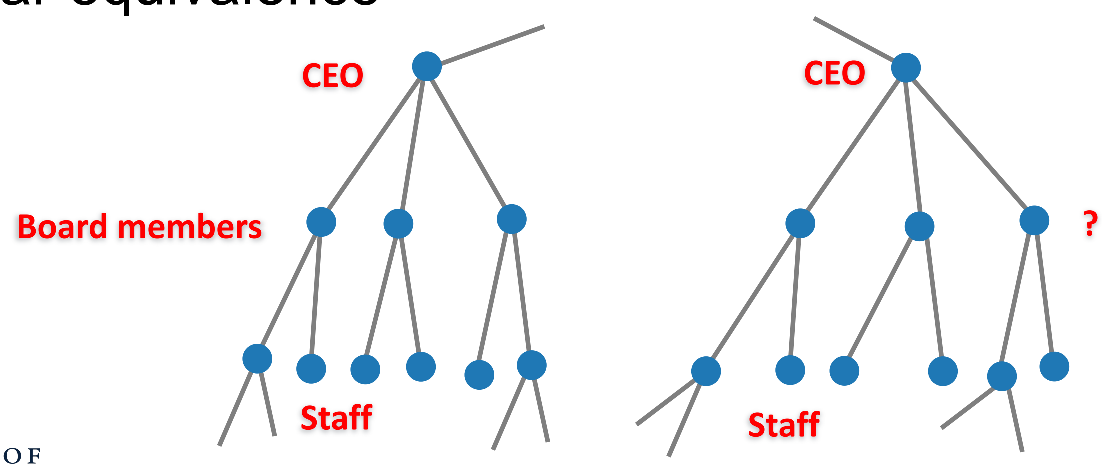
Node-Level Tasks

- **Equivalence:**
- Nodes with similar neighborhood structure tend to have similar features.

> Structural equivalence



> Regular equivalence



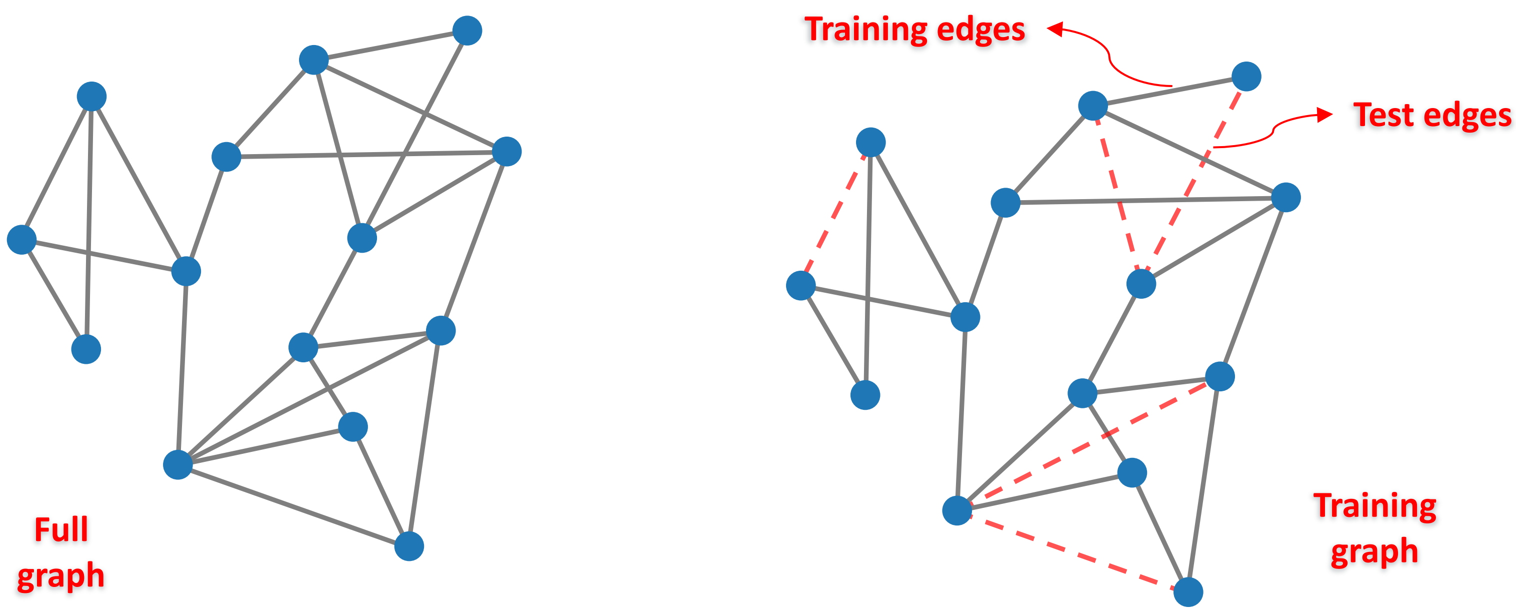
Node-Level Tasks

- **Heterophily:**
- Nodes that interact with each other tend to have opposite attributes.
- Also called disassortative mixing.
- Rarely observed in graphs.
  - Sexual contact network.

Edge-Level Tasks

- Link prediction:
- In this problem, we predict relationship between two nodes by observing a portion of the edges in a graph.
- The model is trained using a dataset that includes set of all nodes and a subset of the edges,

$$D = \{V, \{(u, v) | u, v \in \mathrm{V}, (u, v) \in B, B \subset E\}\}$$



Heterogeneous Graphs

- Heterogeneous graphs are a type of multi-relational graphs where both nodes and edges have types:

$$V = V_1 \cup \ldots \cup V_k, V_i \cap V_j = \emptyset, \forall i \neq j$$

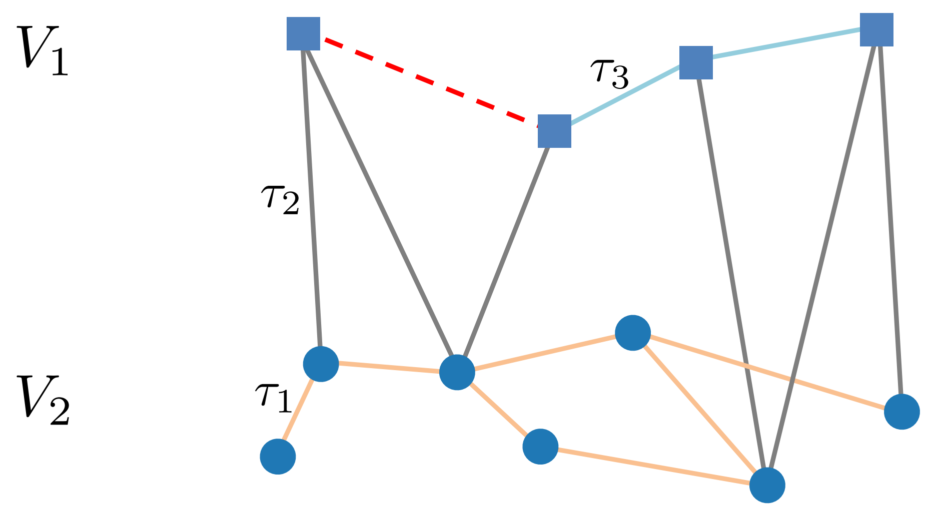
  Here, V_j is set of nodes with type j.

- Edges of specific type τ_i usually only connect nodes of certain types.

$$(u, \tau_i, v) \in \varepsilon \rightarrow u \in V_j, v \in V_k$$

Link Prediction

- ➢ Polypharmacy side-effect prediction
- ❖ Taking multiple drugs results in new side-effects
- ❖ Not all combinations of drugs have been studied.



τ_1 : Protein-protein interaction

τ_2 : Protein-drug interaction

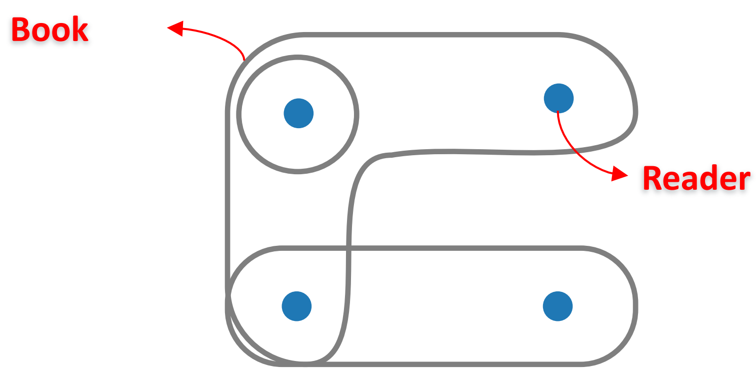
τ_3 : Polypharmacy side-effect

V_1 : Drugs

V_2 : Proteins

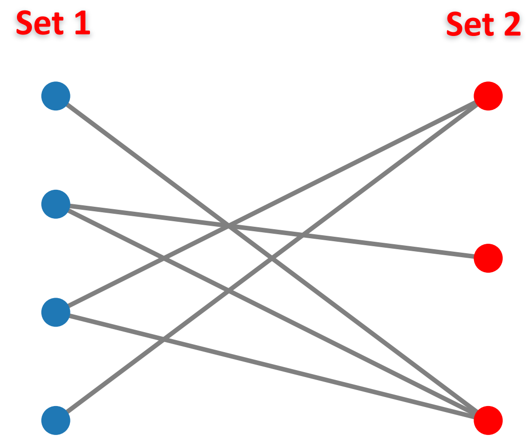
Hypergraph

- An edge that connects more than two nodes is called a hyperedge.
- A graph with hyperedges is called a hypergraph.
- Hyperedges are used to show membership in a group.
- Recommender system:
- Connects book readers and the books they read.



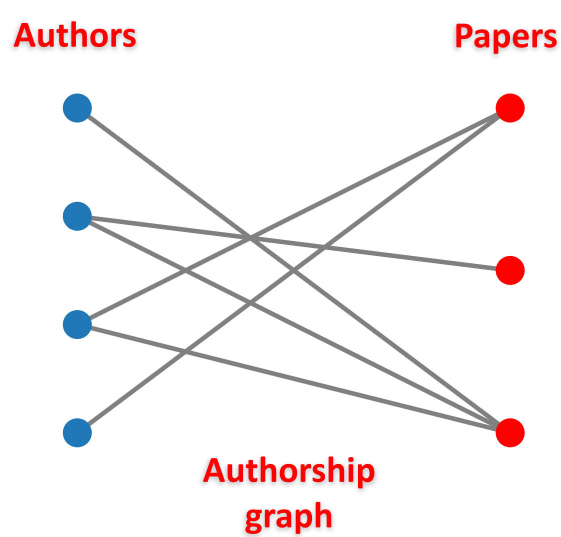
Bipartite Graphs

- Information on a hypergraph can equivalently be represented by a bipartite graph.
- Bipartite graphs, bigraphs, or two mode graphs consist of two disjoint sets of nodes.
- It is a special case of heterogenous graphs where edges only exist between nodes of different sets.



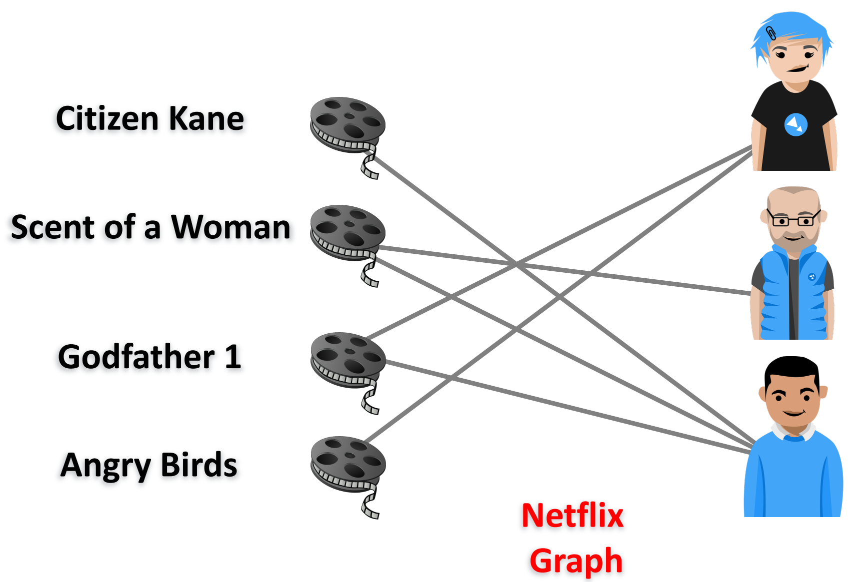
Bipartite Graphs

- ❖ This type of graph is often used to represent interactions between two separate type of components or nodes.
- ➢ The authorship graph:
- ❖ connects authors with their papers.



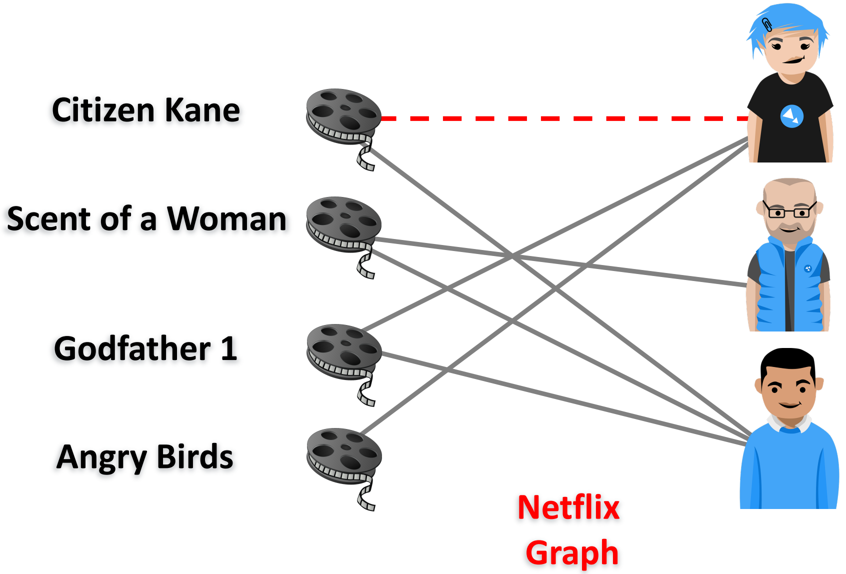
Link Prediction

- ➢ Recommending content:
- ❖ We use a bipartite graph to represent interactions between users and content, e.g. Netflix users and movies, Amazon shoppers and items.



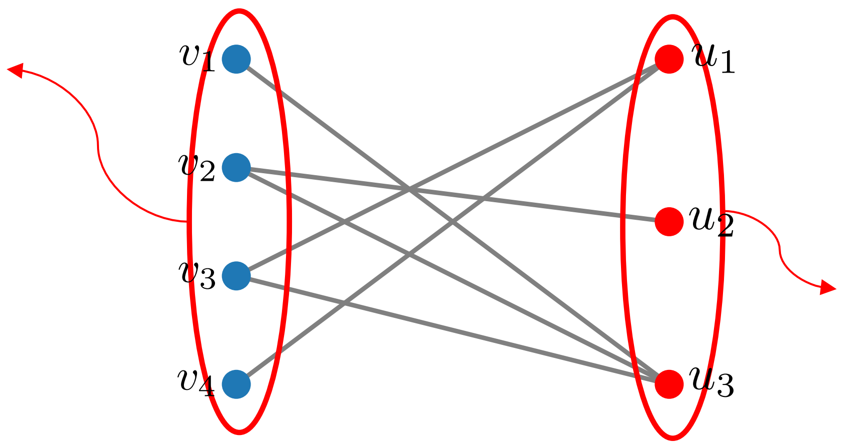
Link Prediction

- Recommending content:
  - We use a bipartite graph to represent interactions between users and content, e.g. Netflix users and movies, Amazon shoppers and items.
  - The predicted link is presented in the form of a recommendation to watch a movie or purchase an item



Bipartite Graphs

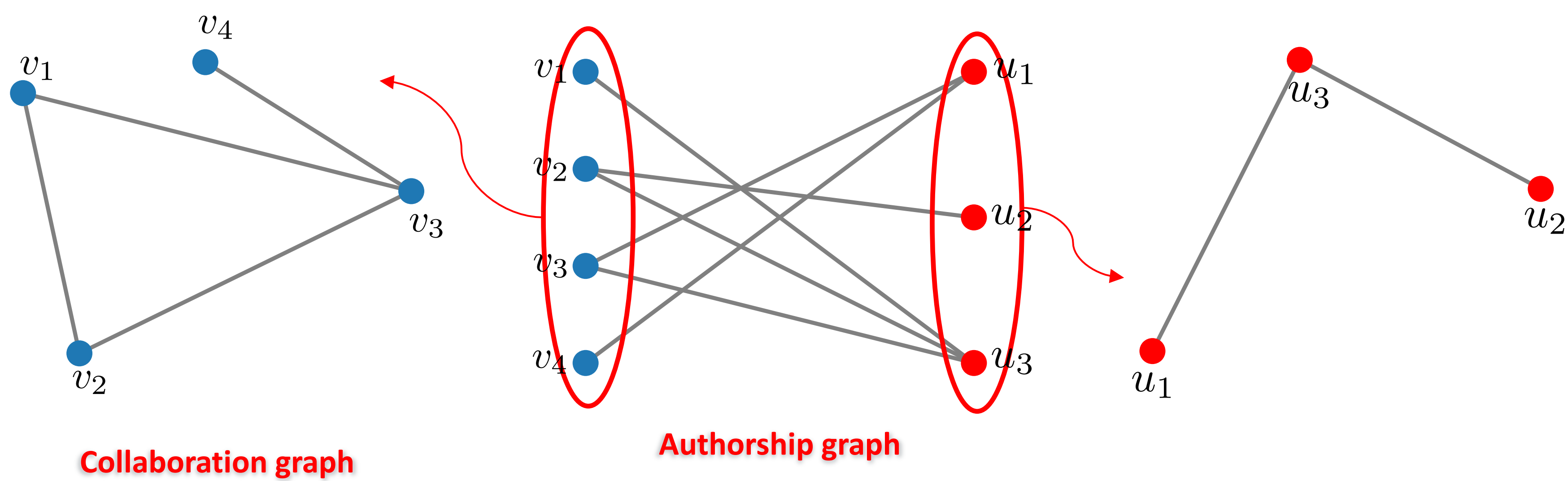
- One interesting property of the bipartite graphs is that each set of nodes can be projected onto a projection graph.
- Each projected graph includes nodes of one set and edges that connect nodes who share a neighbor.



Authorship graph

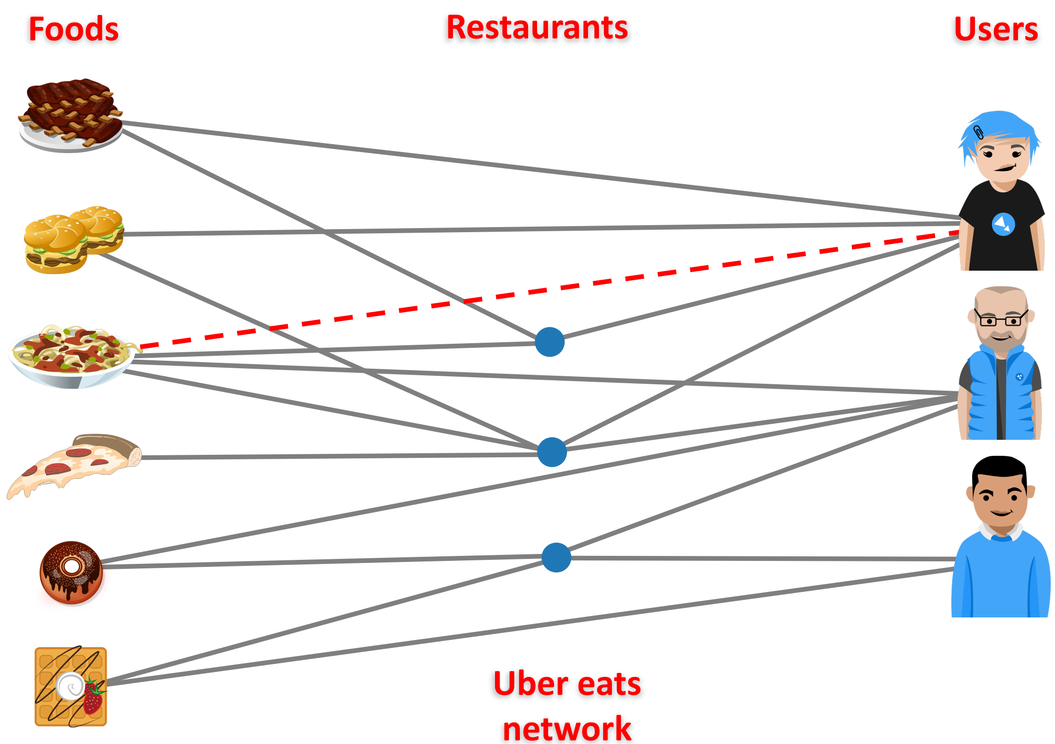
Bipartite Graphs

- One interesting property of the bipartite graphs is that each set of nodes can be projected onto a projection graph.
- Each projected graph includes nodes of one set and edges that connect nodes who share a neighbor.



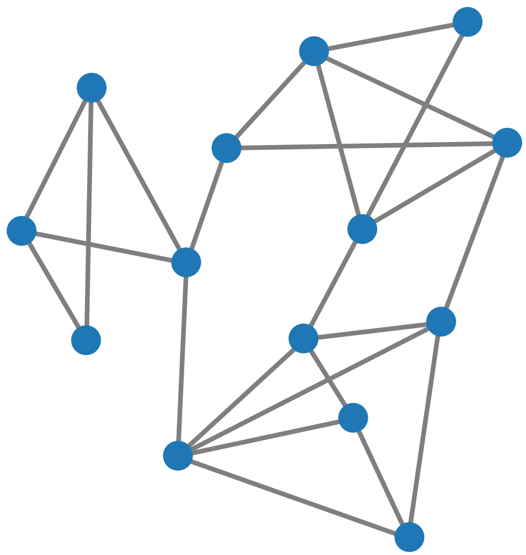
Multi-partite Graph

- K-partite graphs consist of k disjoint set of nodes, where nodes of the same set are not connected.
- UberEats network.



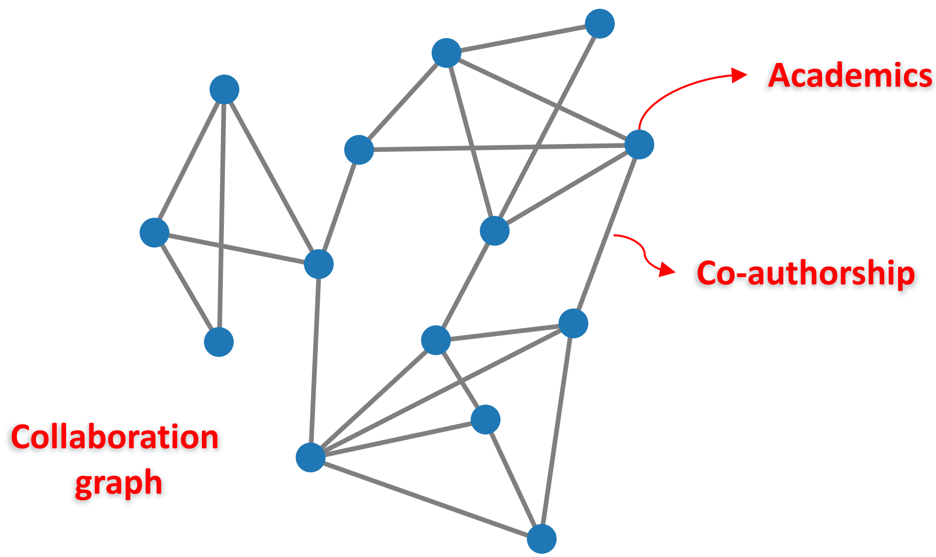
Clustering

- Clustering problem within a graph partitions it into subgraphs with similar underlying structure.
- Depending on the problem, these subgraphs may or may not overlap.
- The model is trained on the set of nodes and edges in the graph $D = \{V, E\}$.



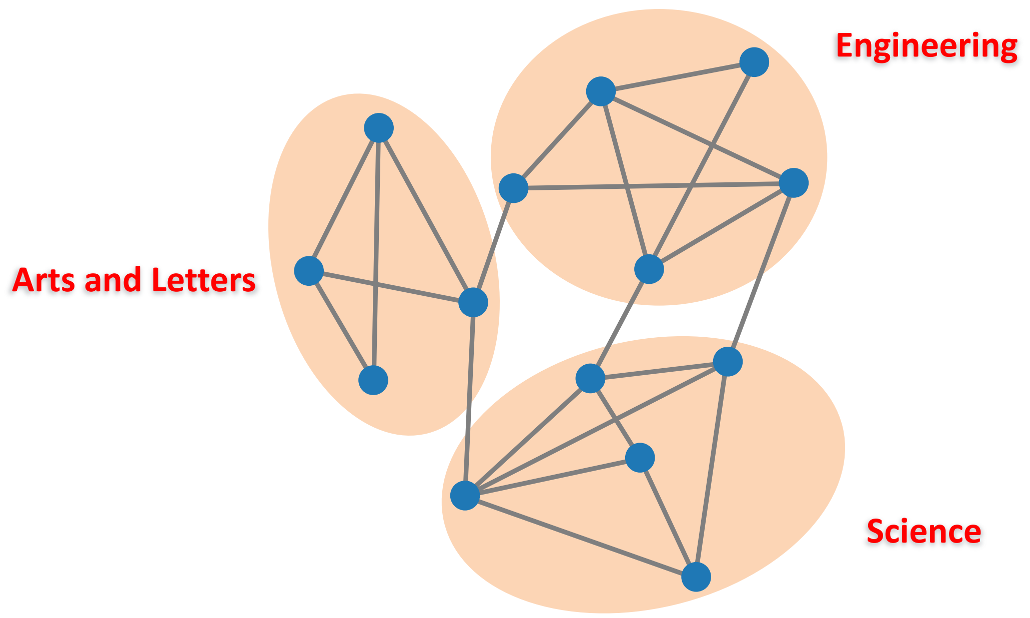
Clustering

- Detect latent community structures (Community detection).



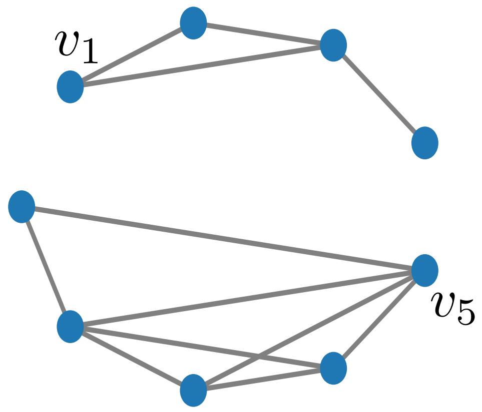
Clustering

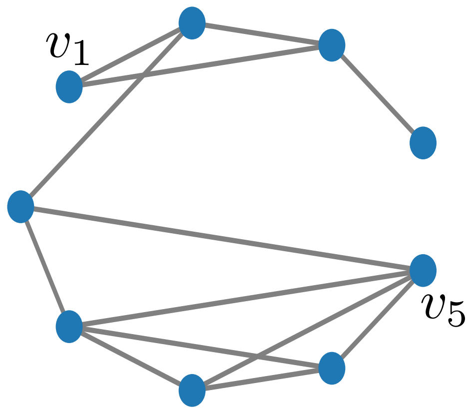
- ➢ Detect latent community structures (Community detection).
- ❖ The underlying structures learned from a collaboration graph may yield communities based on department, demographic, or research interests.



Connectedness

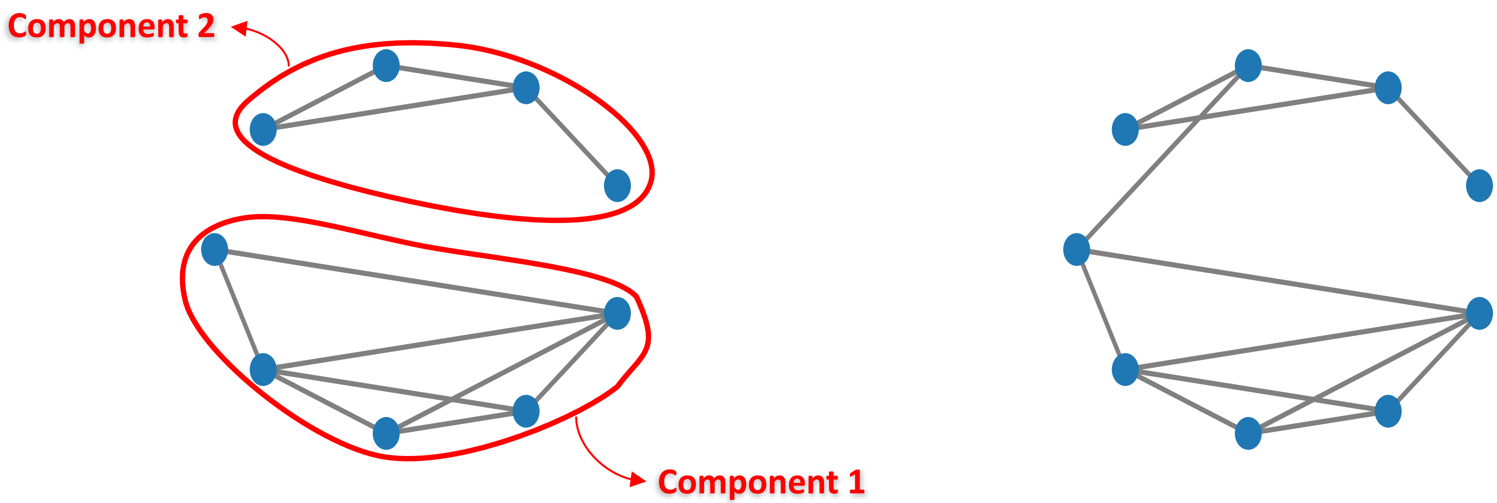
- A graph is connected if there is a path between any given pair of nodes in the graph.





Connectedness

- A graph is connected if there is a path between any given pair of nodes in the graph.

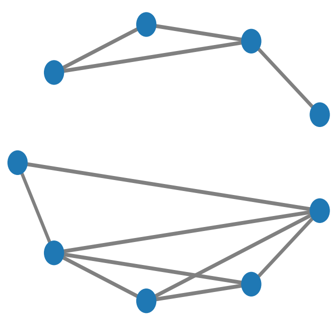


- If graph is not connected, each connected subgraph is called a component.

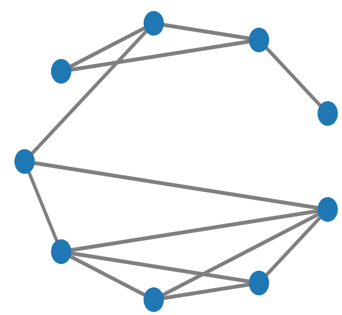
Connectedness

- Disconnected graphs have an adjacency matrix with nonzero square blocks on the diagonal and zeros outside of these blocks.
- Using this principle, linear algebraic methods can inspect the connectedness of the graph.

Block diagonal matrix



```
[[0 1 0 0 1 0 0 0 0]
 [1 0 1 1 1 0 0 0 0]
 [0 1 0 1 1 0 0 0 0]
 [0 1 1 0 1 0 0 0 0]
 [1 1 1 1 0 0 0 0 0]
 [0 0 0 0 0 0 1 0 0]
 [0 0 0 0 0 1 0 1 1]
 [0 0 0 0 0 0 1 0 1]
 [0 0 0 0 0 0 1 1 0]]
```

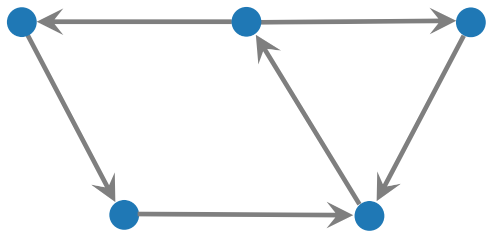


```
[[0 1 0 0 1 0 0 1 0]
 [1 0 1 1 1 0 0 0 0]
 [0 1 0 1 1 0 0 0 0]
 [0 1 1 0 1 0 0 0 0]
 [1 1 1 1 0 0 0 0 0]
 [0 0 0 0 0 0 1 0 0]
 [0 0 0 0 0 1 0 1 1]
 [1 0 0 0 0 0 1 0 1]
 [0 0 0 0 0 0 1 1 0]]
```

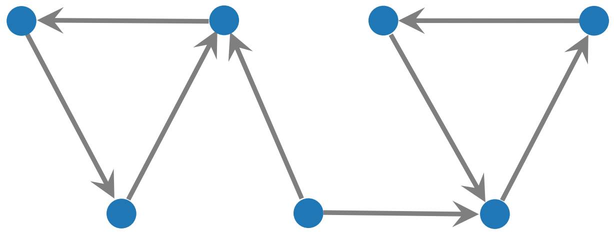
Bridge

Connectedness

- There are two types of connectivity in directed graphs:
- **Strong connectivity:** If there is a path from every node $u \in \mathrm{V}$ to every node $v \in \mathrm{V}$, the graph is strongly connected.

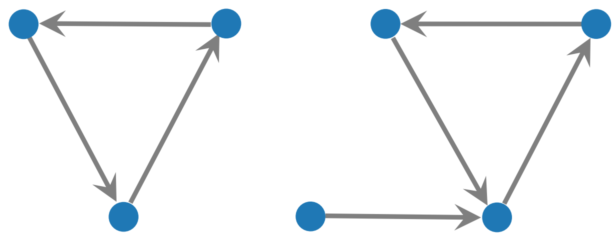


- **Weak connectivity:** If the graph is not strongly connected, but removing directions yields a connected undirected graph.
- Example: Web.



Connectedness

- There are two types of components in directed graphs:
- A **weakly connected component** consists of nodes that are connected through one or more paths that can go either direction.

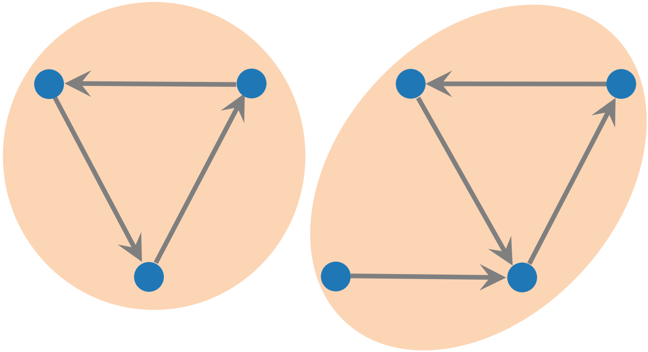


- A **strongly connected component** consists of maximal subset of nodes such that every pair of nodes are connected in both directions.

Connectedness

❖ There are two types of components in directed graphs:

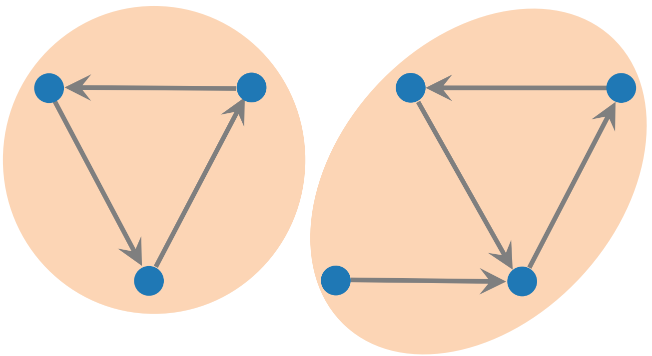
➢ A **weakly connected component** consists of nodes that are connected through one or more paths that can go either direction.



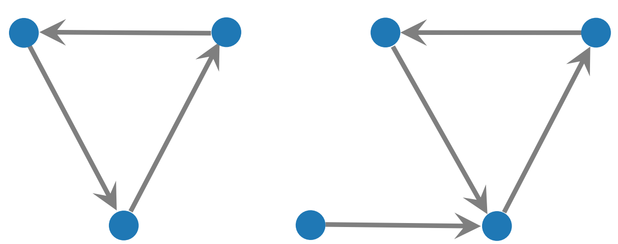
➢ A **strongly connected component** consists of maximal subset of nodes such that every pair of nodes are connected in both directions.

Connectedness

- There are two types of components in directed graphs:
- A **weakly connected component** consists of nodes that are connected through one or more paths that can go either direction.

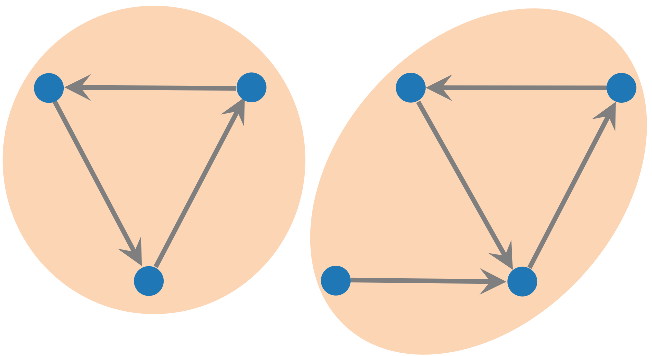


- A **strongly connected component** consists of maximal subset of nodes such that every pair of nodes are connected in both directions.

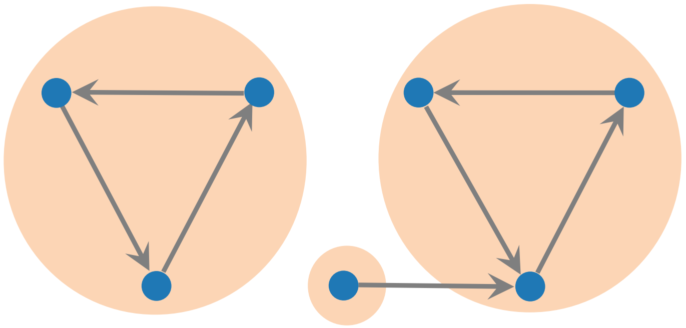


Connectedness

- There are two types of components in directed graphs:
- A **weakly connected component** consists of nodes that are connected through one or more paths that can go either direction.

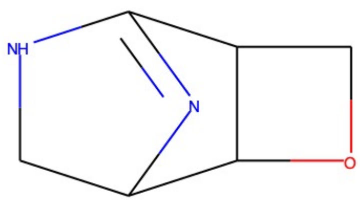


- A **strongly connected component** consists of maximal subset of nodes such that every pair of nodes are connected in both directions.



Graph-Level Tasks

- Make prediction based on the whole graph. Closer to conventional ML tasks.
- Accounts for the inherent topology of the datapoints in addition to the data that it presents.
- **Graph regression:** Predict real value graph-level attributes given the graph structure, e.g. property prediction.
- The training dataset $D = \{G^{(i)}, y^{(i)}\}_{i \leq \mathrm{N}}$ includes a set of N iid graphs $G^{(i)}$ and their corresponding properties $y^{(i)}$.

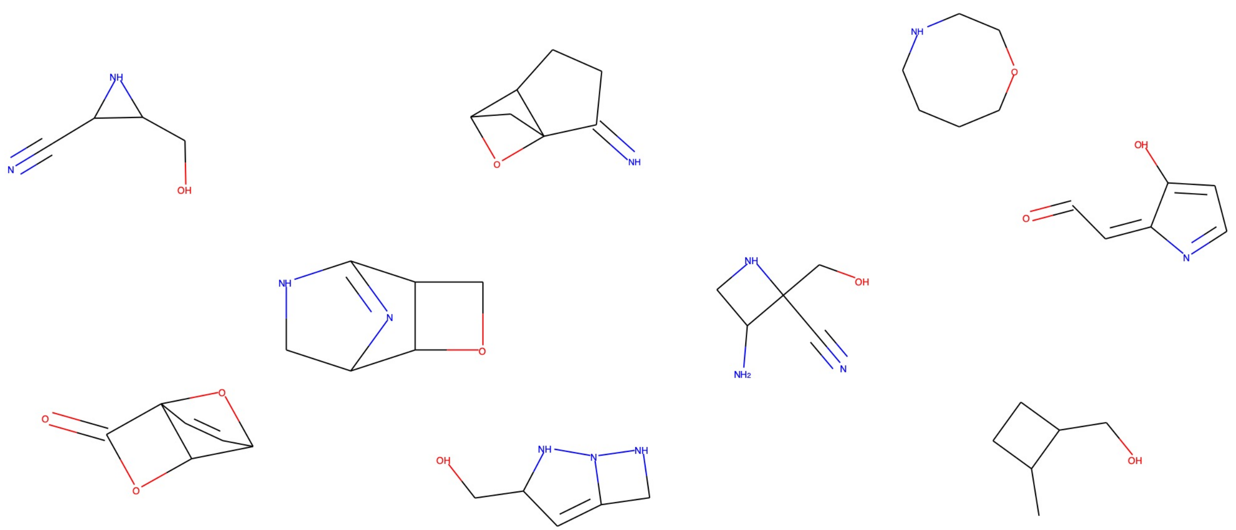


$$\longrightarrow \quad f(G)$$

- **Graph classification:** Predict the graph-level target label for the input graph (e.g., solubility and toxicity prediction)

Graph-Level Tasks

- **Graph clustering:** Group a set of graphs into different clusters.
  - Data: iid data points $G = \{G^{(i)}\}$.
  - This task is equivalent to unsupervised learning in traditional ML.
- Ring size, number of hydrogens, functional groups.



Summary

- Learning Tasks on graphs
  - Node-level tasks: Node classification
    - Strategies: Homophily, Equivalence, Heterophily.
  - Edge-level tasks: Link prediction
    - Heterogenous graphs
    - Hypergraphs
    - Bipartite and multi-partite graphs
  - Subgraph-level tasks: Clustering
    - Connectedness
    - Components
  - Graph-level tasks: Regression, classification, and clustering.