CLAUDIO STAMILE - 10/12/2021

GRAPH MACHINE LEARNING

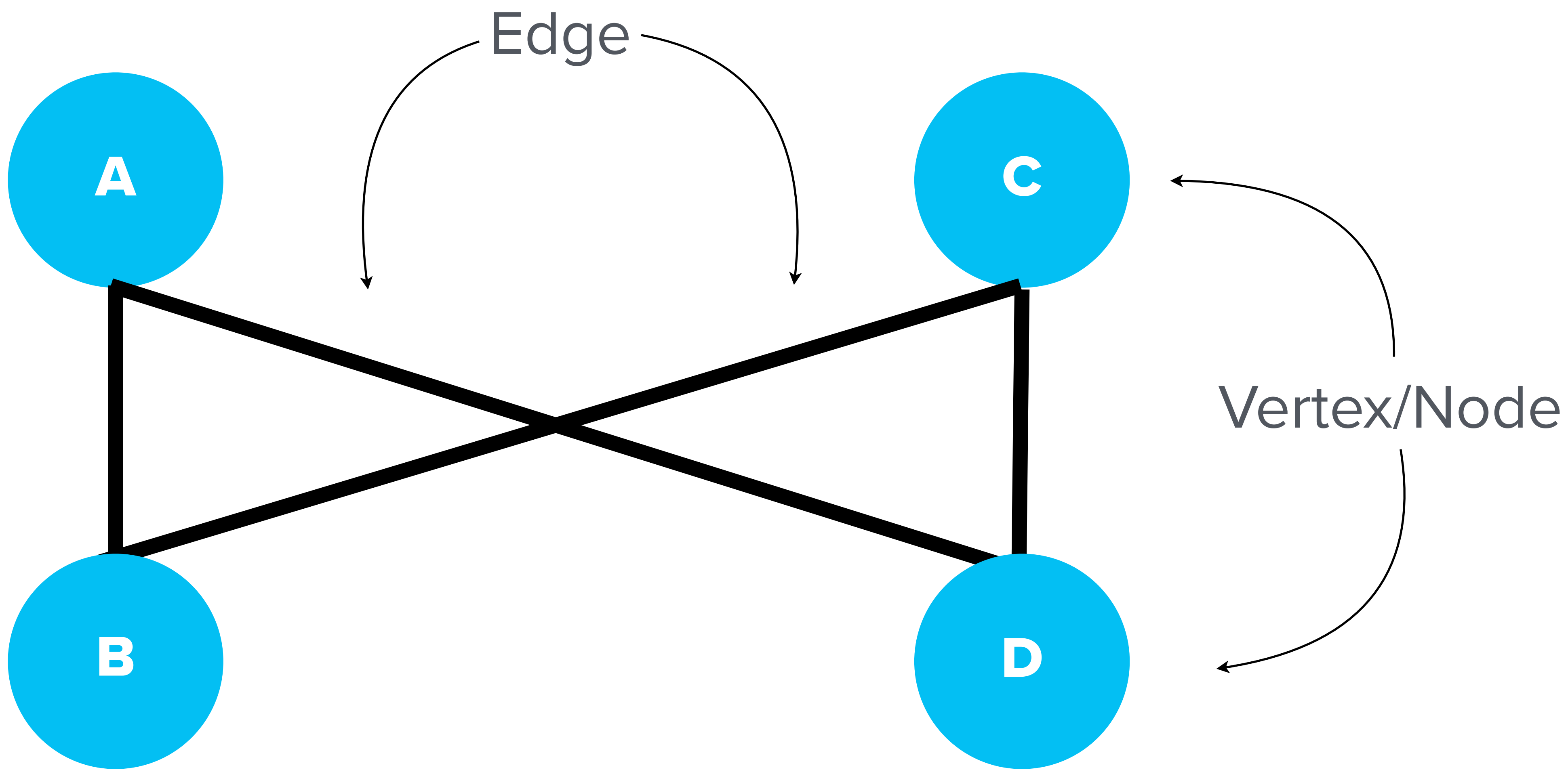
An introduction to graphs and their application in machine learning

WHAT ? GRAPH MACHINE LEARNING ?

1. Getting started with graphs
2. Machine Learning on Graphs
3. Graph Representation Learning
4. Graph Machine Learning for passengers forecasting

AIRPORTS CONNECTIONS LOOKS LIKE GRAPHS

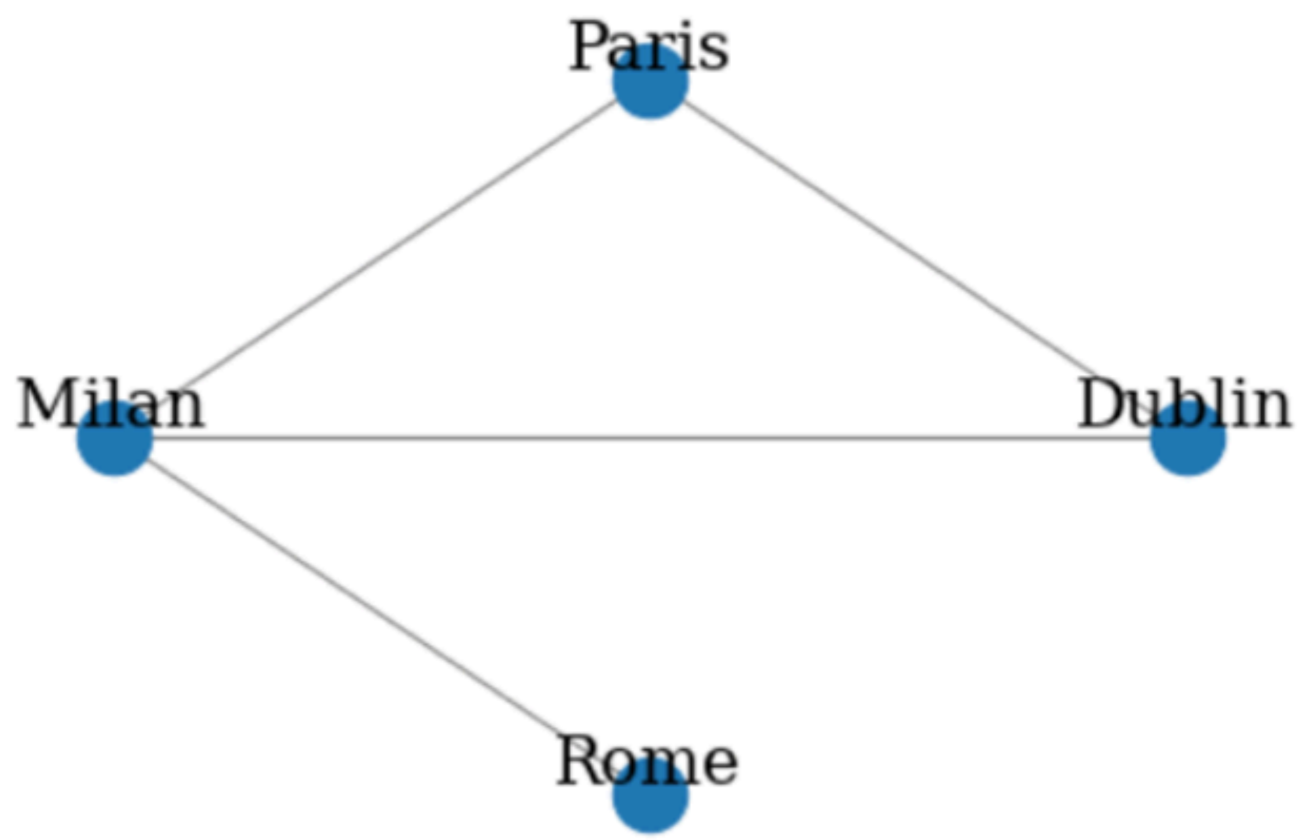
http://www.martingrandjean.ch/connected-world-air-traffic-network/



- Each airport is point (node)
- Each connection between two airports is a line (edge)

GETTING STARTED WITH GRAPHS

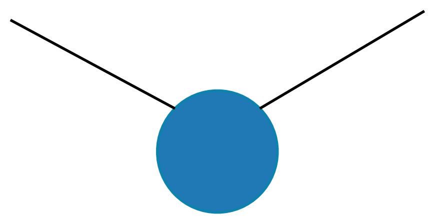
G=(V,E)
V = {Paris, Milan, Dublin, Rome}
E={{Paris, Milan}, {Paris, Dublin}, {Milan, Dublin}, {Milan, Rome}}




```
import networkx as nx
G = nx.Graph()
V = {'Dublin', 'Paris', 'Milan', 'Rome'}
E = [('Milan','Dublin'), ('Milan','Paris'),
('Paris','Dublin'), ('Milan','Rome')]
G.add_nodes_from(V)
G.add_edges_from(E)
```

```
nx.to_pandas_adjacency(G)
```

	Paris	Rome	Dublin	Milan
Paris	0.0	0.0	1.0	1.0
Rome	0.0	0.0	0.0	1.0
Dublin	1.0	0.0	0.0	1.0
Milan	1.0	1.0	1.0	0.0

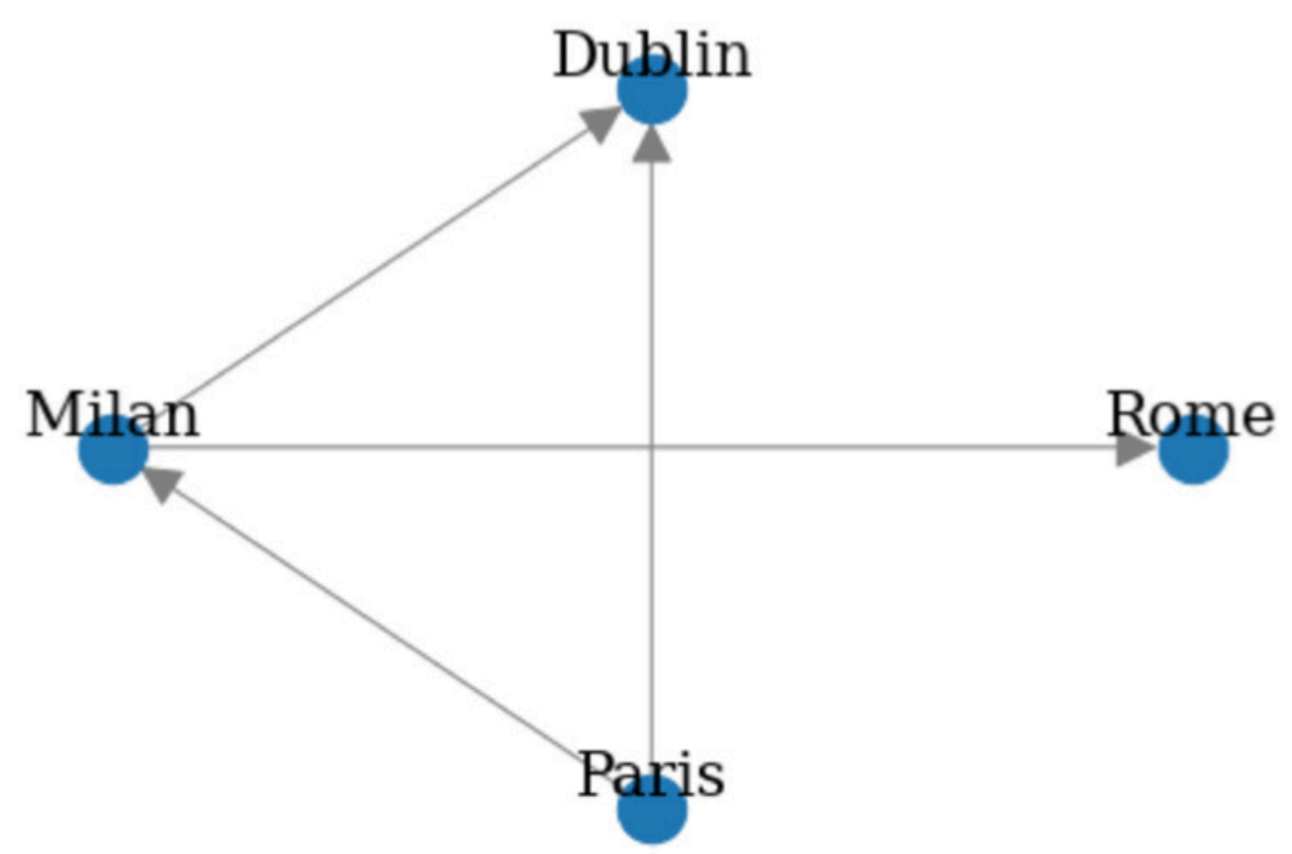


Node Degree: G.degree('Paris')

Graph Order: G.number_of_nodes()
Graph Size: G.number_of_edges()

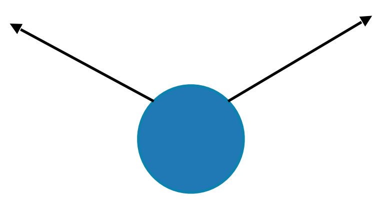
ONE GRAPH, TWO GRAPHS, MORE GRAPHS

- Directed Graph

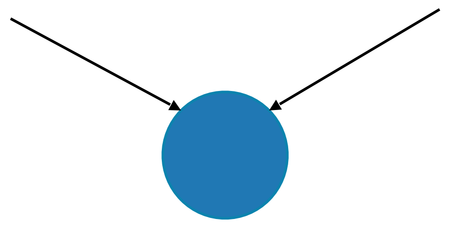



```
import networkx as nx
G = nx.DiGraph()
V = {'Dublin', 'Paris', 'Milan', 'Rome'}
E = [('Milan','Dublin'), ('Milan','Paris'),
('Paris','Dublin'), ('Milan','Rome')]
G.add_nodes_from(V)
G.add_edges_from(E)
```

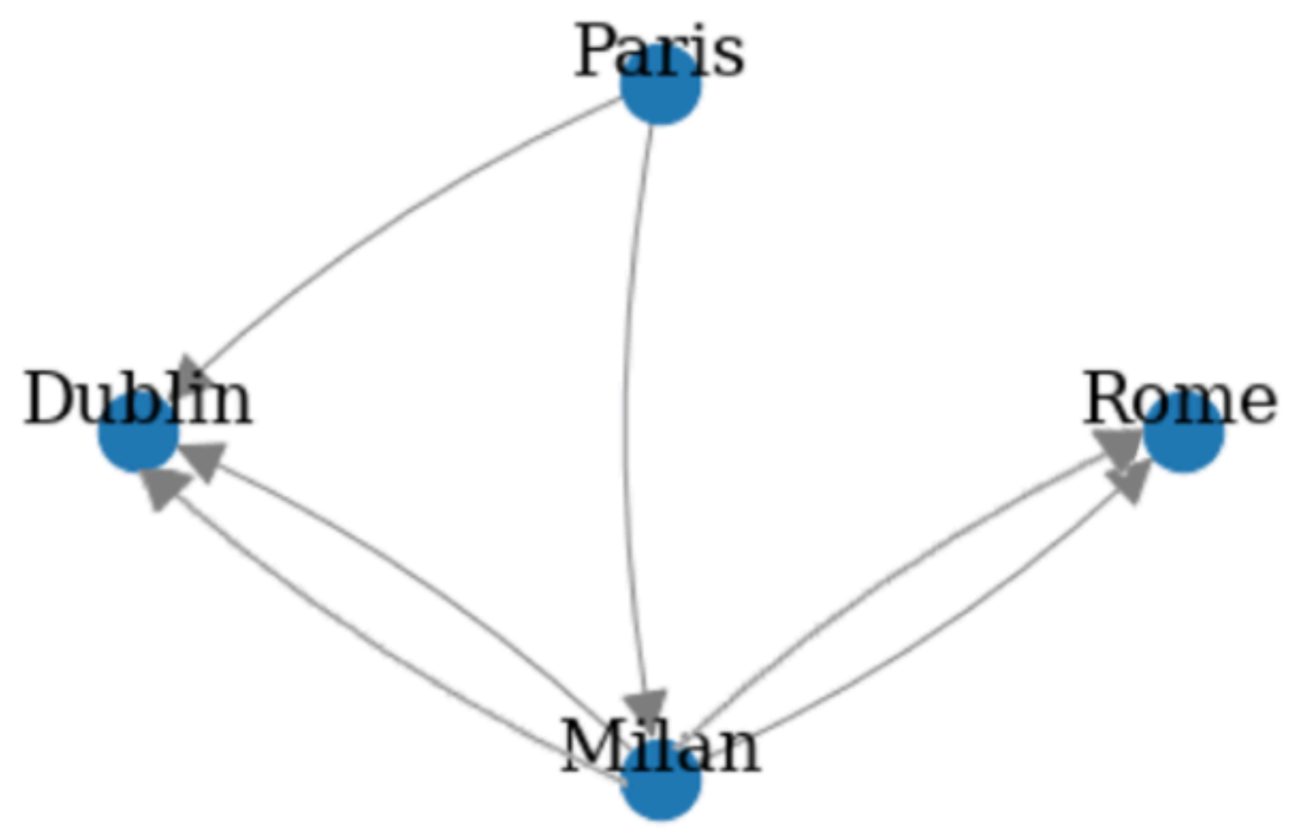
Out Degree



In Degree

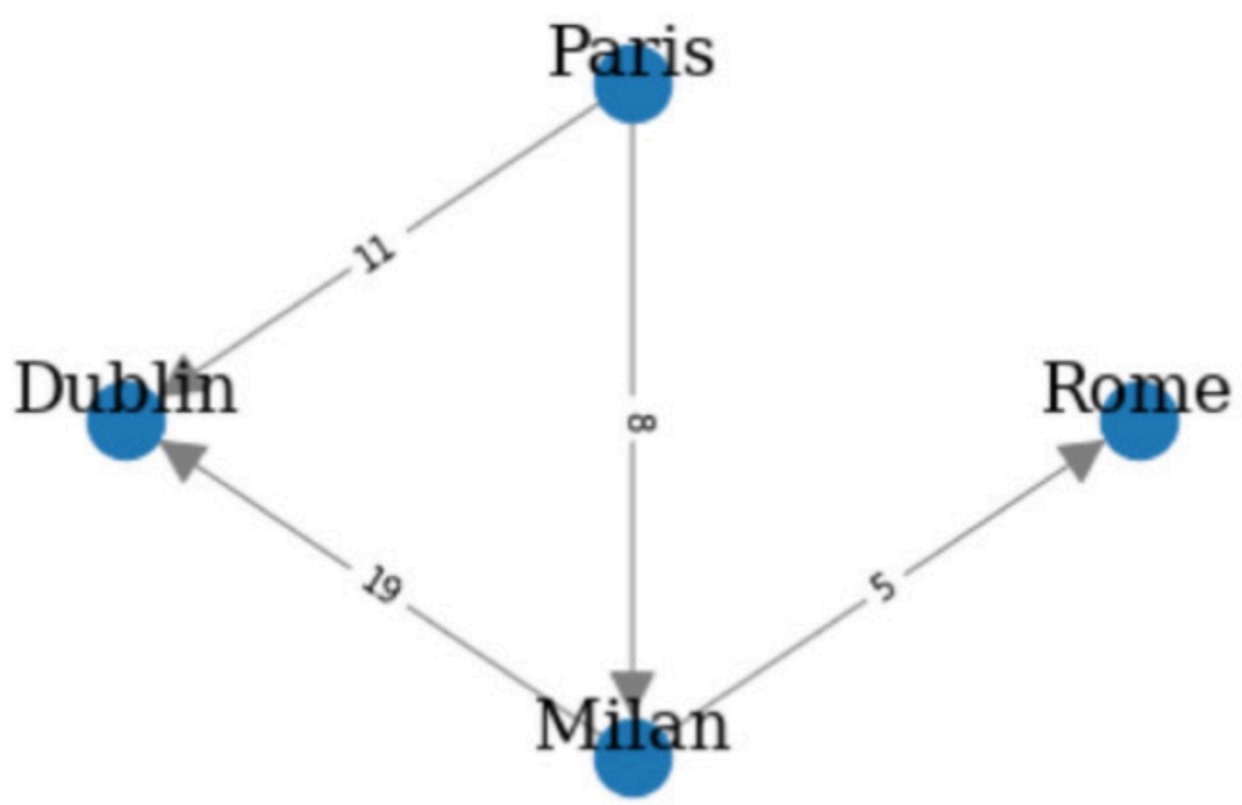


- Multi Graph




```
dmg = nx.MultiDiGraph()
umg = nx.MultiGraph()
V = {'Dublin', 'Paris', 'Milan', 'Rome'}
E = [('Milan','Dublin'), ('Milan','Dublin'), ('Paris','Milan'),
('Paris','Dublin'), ('Milan','Rome'), ('Milan','Rome')]
dmg.add_nodes_from(V)
umg.add_nodes_from(V)
dmg.add_edges_from(E)
umg.add_edges_from(E)
```

- Weighed Graph




```
G = nx.DiGraph()
V = {'Dublin', 'Paris', 'Milan', 'Rome'}
E = [('Milan','Dublin', 19), ('Paris','Milan',
8), ('Paris','Dublin', 11), ('Milan','Rome', 5)]
G.add_nodes_from(V)
G.add_weighted_edges_from(E)
```

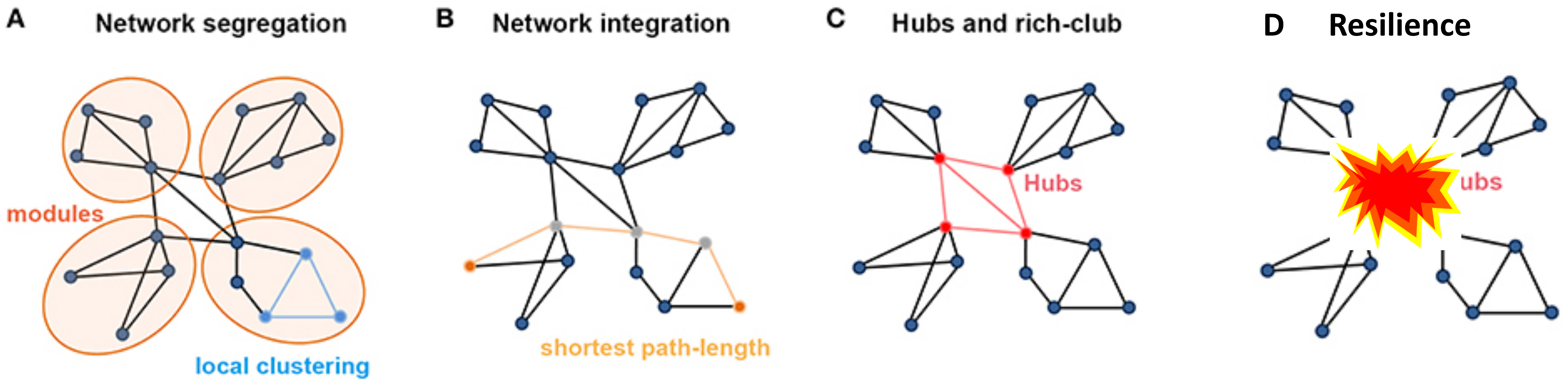
GRAPH DERIVED METRICS

A. Quantify the presence of groups of interconnected nodes, known as communities or modules, within a network

B. Measure how nodes tend to be interconnected with each other.

C. Assess the importance of individual nodes inside a network.

D. Measure of how much a network is able to maintain and adapt its operational performance when facing failures or other adverse conditions.

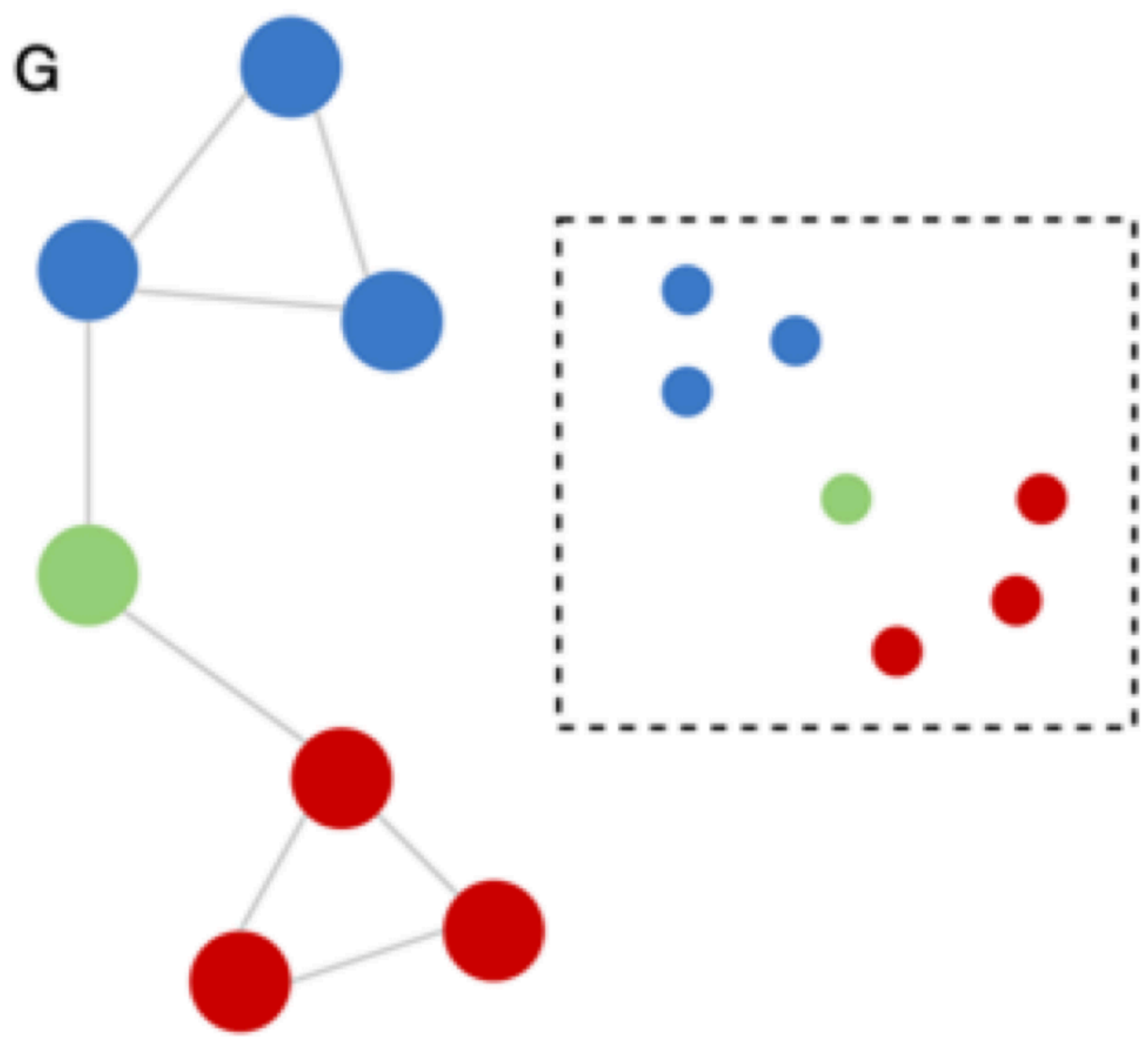


GRAPH DERIVED METRICS ARE A LOT

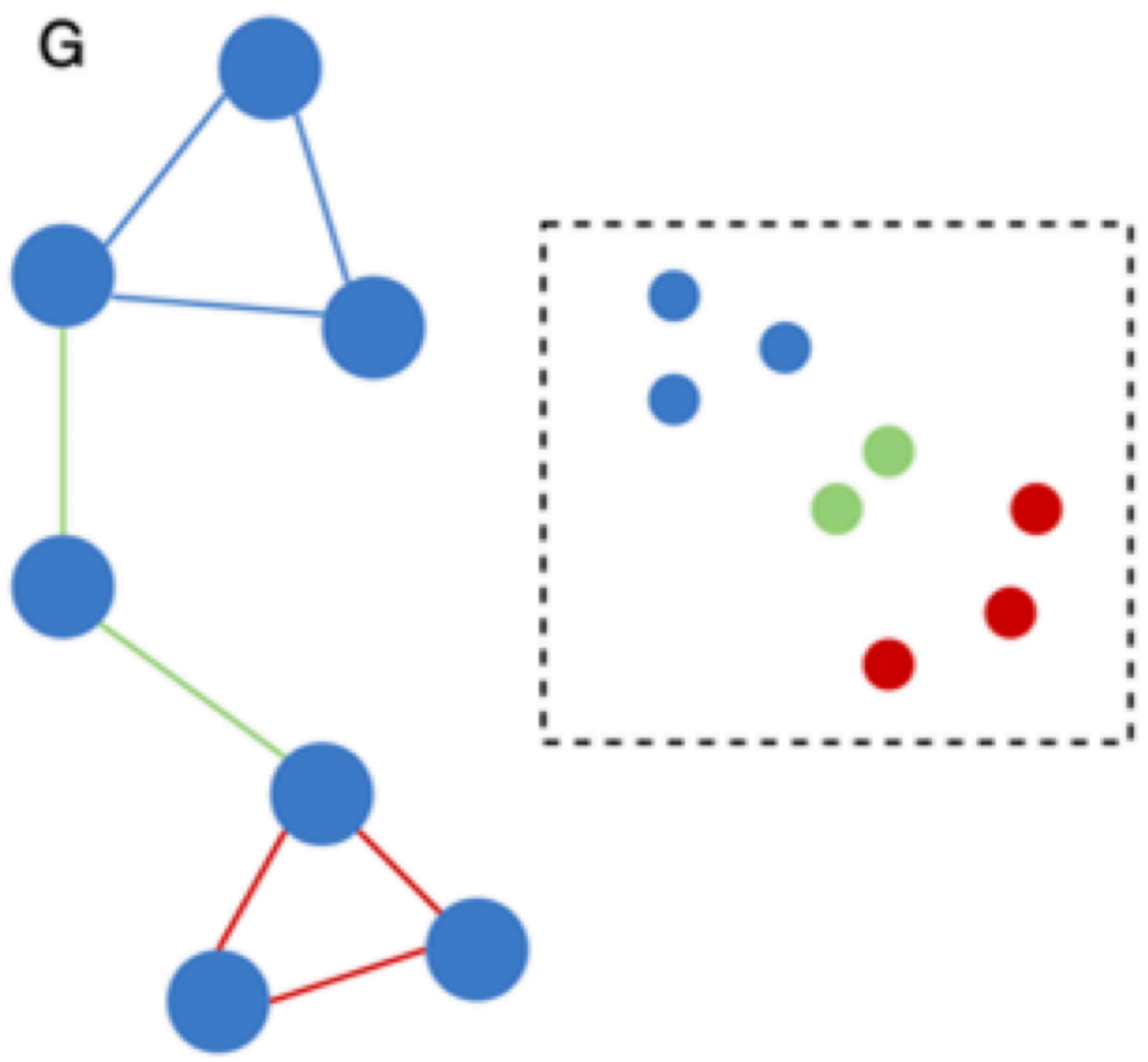
1. VertexCount
2. EdgeCount
3. VertexDegree
4. VertexInDegree
5. VertexOutDegree
6. GraphDistance
7. GraphDistanceMatrix
8. VertexEccentricity
9. GraphRadius
10. GraphDiameter
11. Connectivity Measures
12. VertexConnectivity
13. EdgeConnectivity
14. Centrality Measures
15. ClosenessCentrality
16. BetweennessCentrality
17. DegreeCentrality
18. EigenvectorCentrality
19. KatzCentrality
20. PageRankCentrality
21. HITSCentrality
22. RadialityCentrality
23. StatusCentrality
24. EdgeBetweennessCentrality
25. GraphReciprocity
26. GlobalClusteringCoefficient
27. MeanClusteringCoefficient
28. LocalClusteringCoefficient
29. GraphAssortativity
30. VertexCorrelationSimilarity
31. MeanNeighborDegree
32. MeanDegreeConnectivity
33. VertexDiceSimilarity
34. VertexJaccardSimilarity
35. VertexCosineSimilarity

GRAPH MACHINE LEARNING AT DIFFERENT SCALE

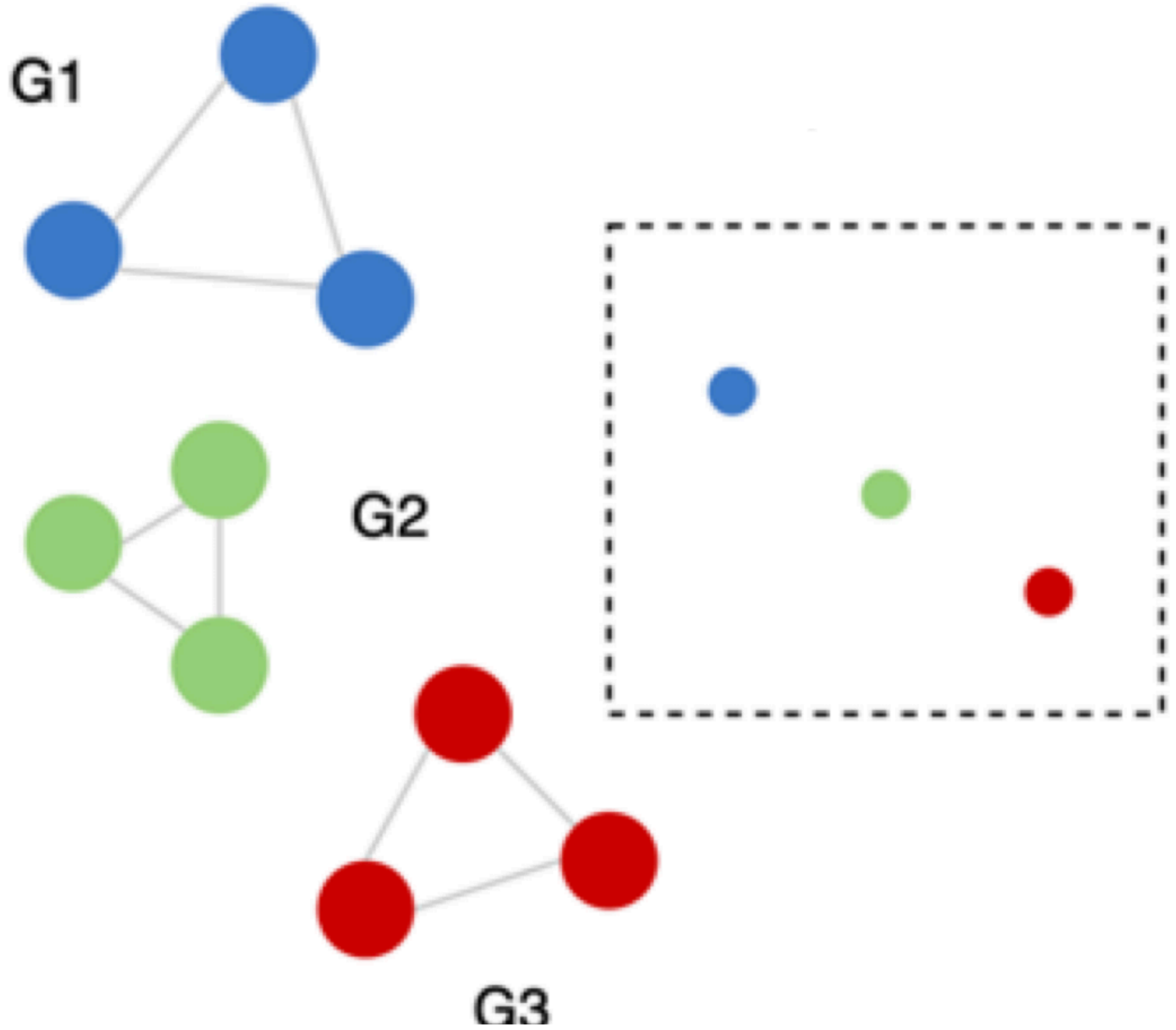
- Node Level Classification



- Edge Level Classification

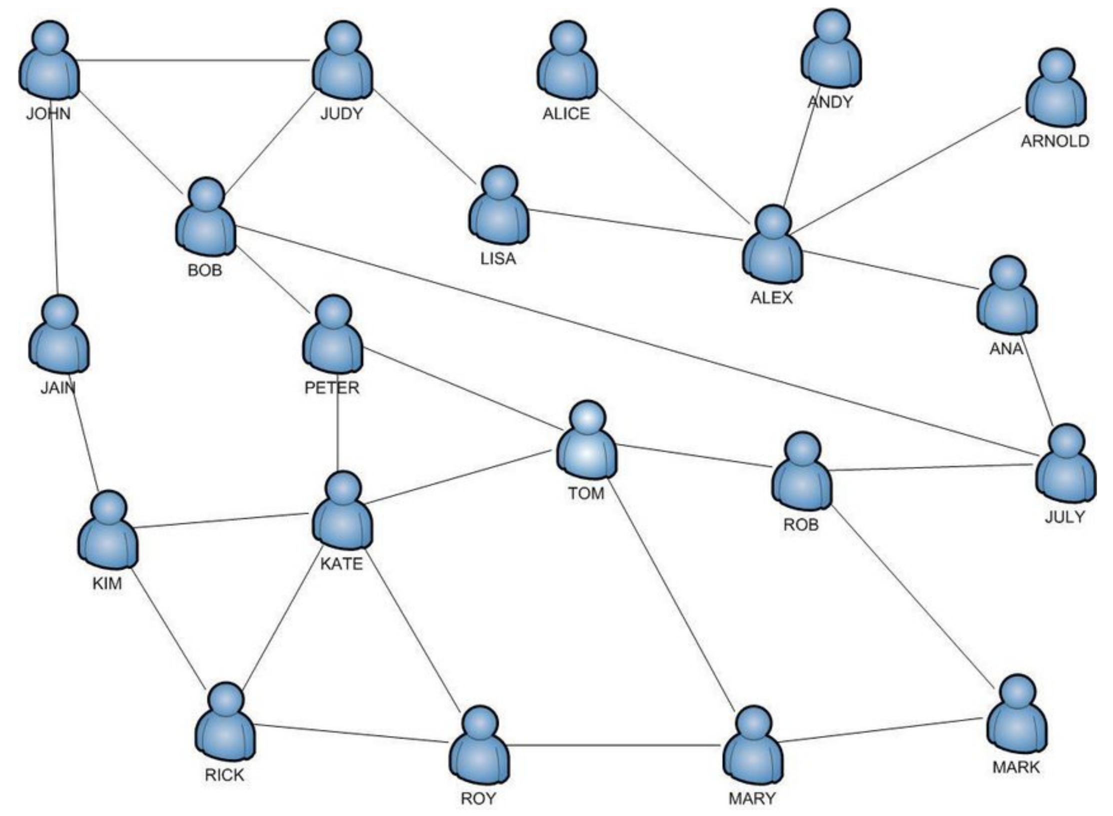


- Graph Level Classification



MACHINE LEARNING ON GRAPH - A STRAIGHTFORWARD APPROACH

- In «classical» machine learning applications, a common way to process the input data is to build a set of features
- For a *node classification* task, we compute a set of metrics or known features to generate the feature vector



Feature engineering

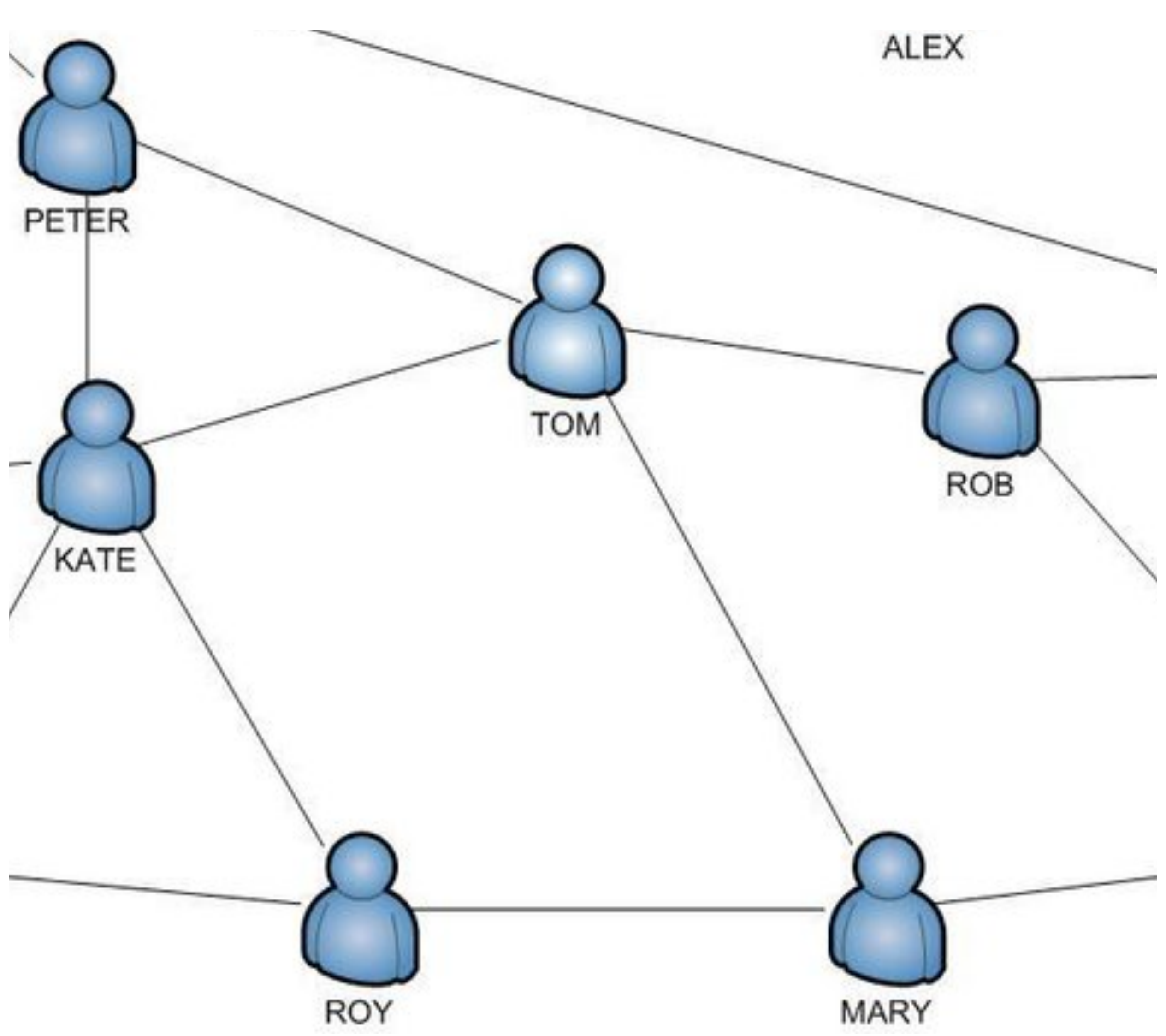
NodeId	Node Degree	…	Age
Alice	0.9	0.12	28
..	0.20	0.88	…
Bob	0.25	0.76	31

- *Node Attributes*: We know that the nodes in a graph represent entities and these entities have their own characteristic attributes (Location, Area, Average number of passenger, etc..).
- *Local Structural Features*: Node features like degree (count of adjacent nodes), mean of degrees of neighbor nodes, number of triangles a node forms with other nodes, etc.

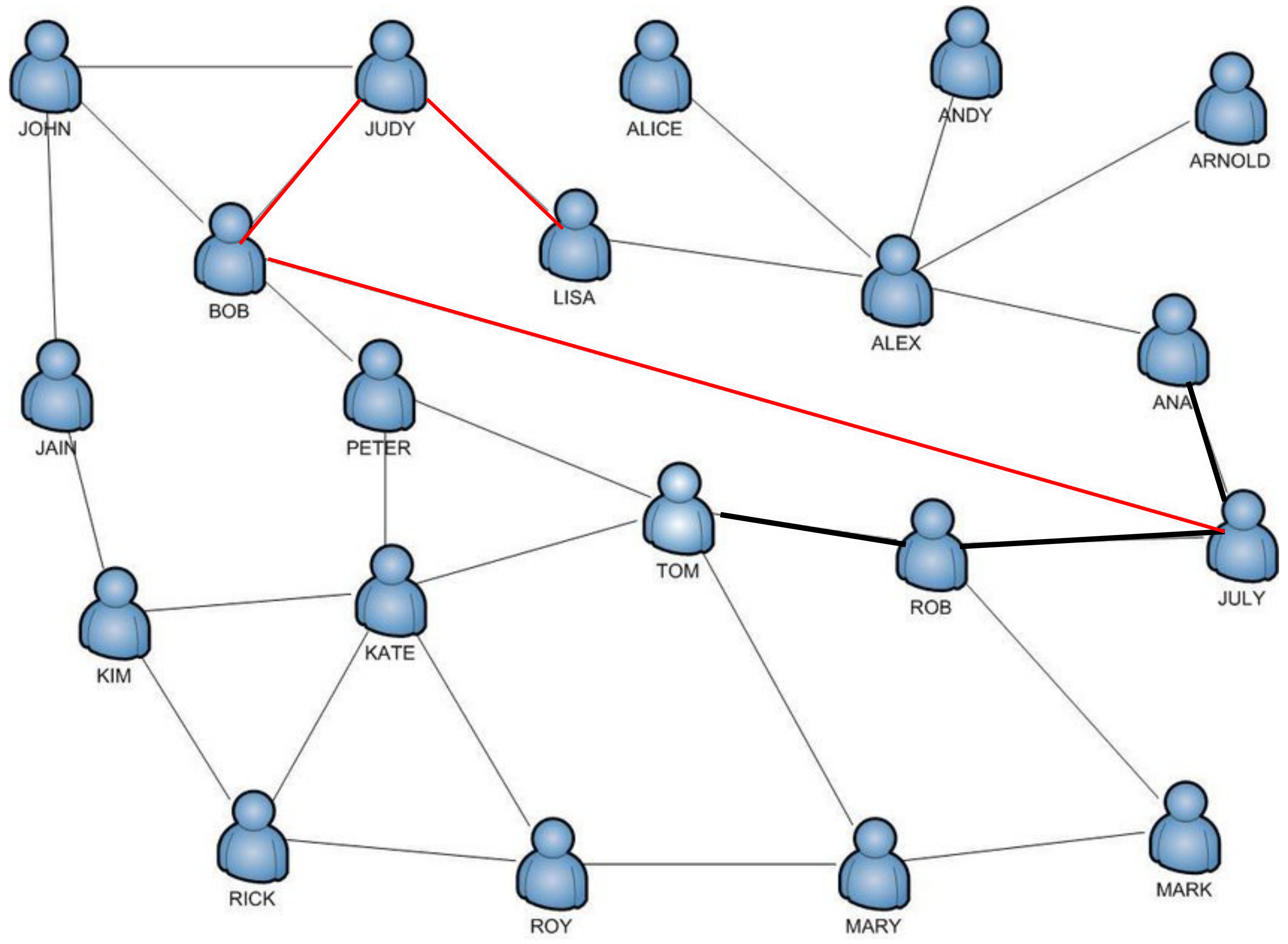
A STRAIGHTFORWARD APPROACH - LIMITATIONS

1. *Node Attributes:* We know that the nodes in a graph represent entities and these entities have their own characteristic attributes (Age).
2. *Local Structural Features:* Node features like degree (count of adjacent nodes), mean of degrees of neighbor nodes, number of triangles a node forms with other nodes, etc.
3. ***Embeddings:*** **The above-discussed features carry only node related information. They do not capture the information about the context of a node (the surrounding nodes).**

- Local Structure Features



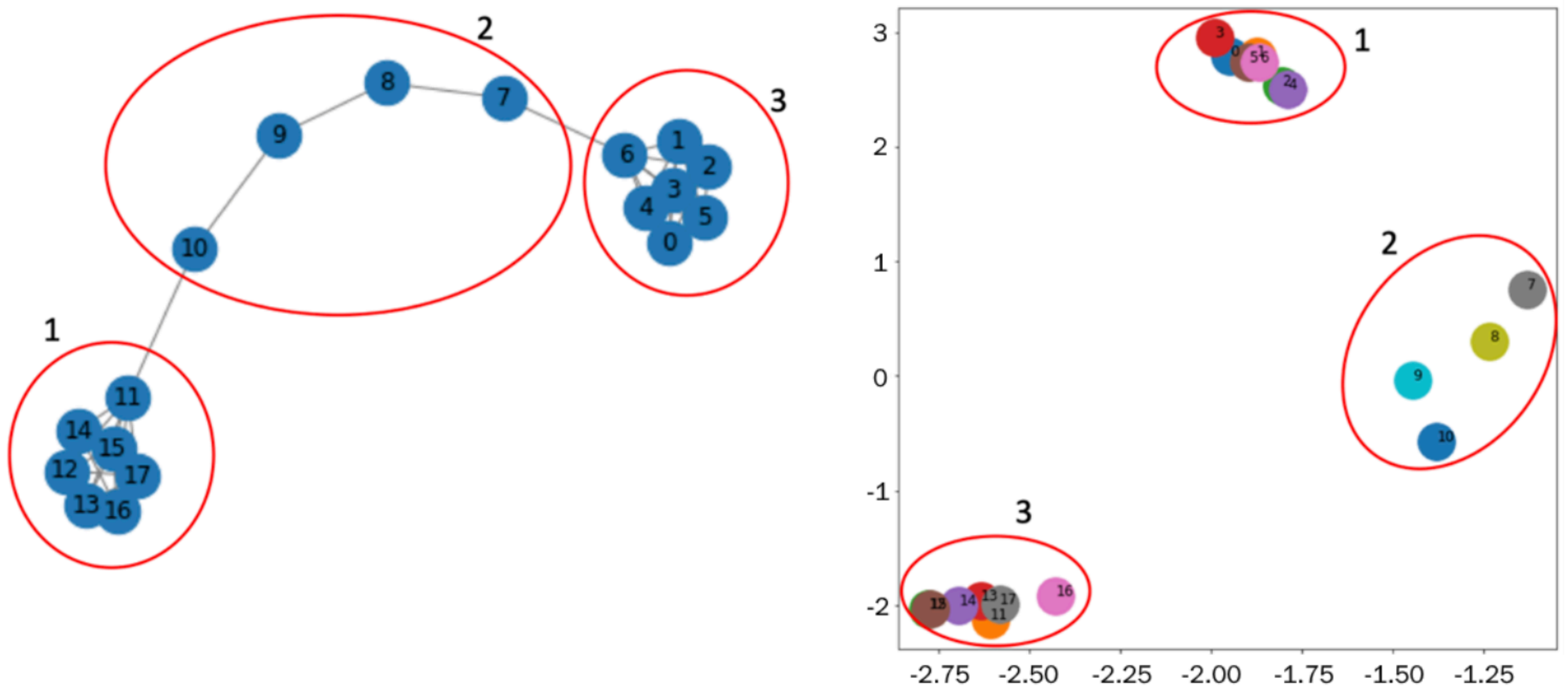
- Embeddings



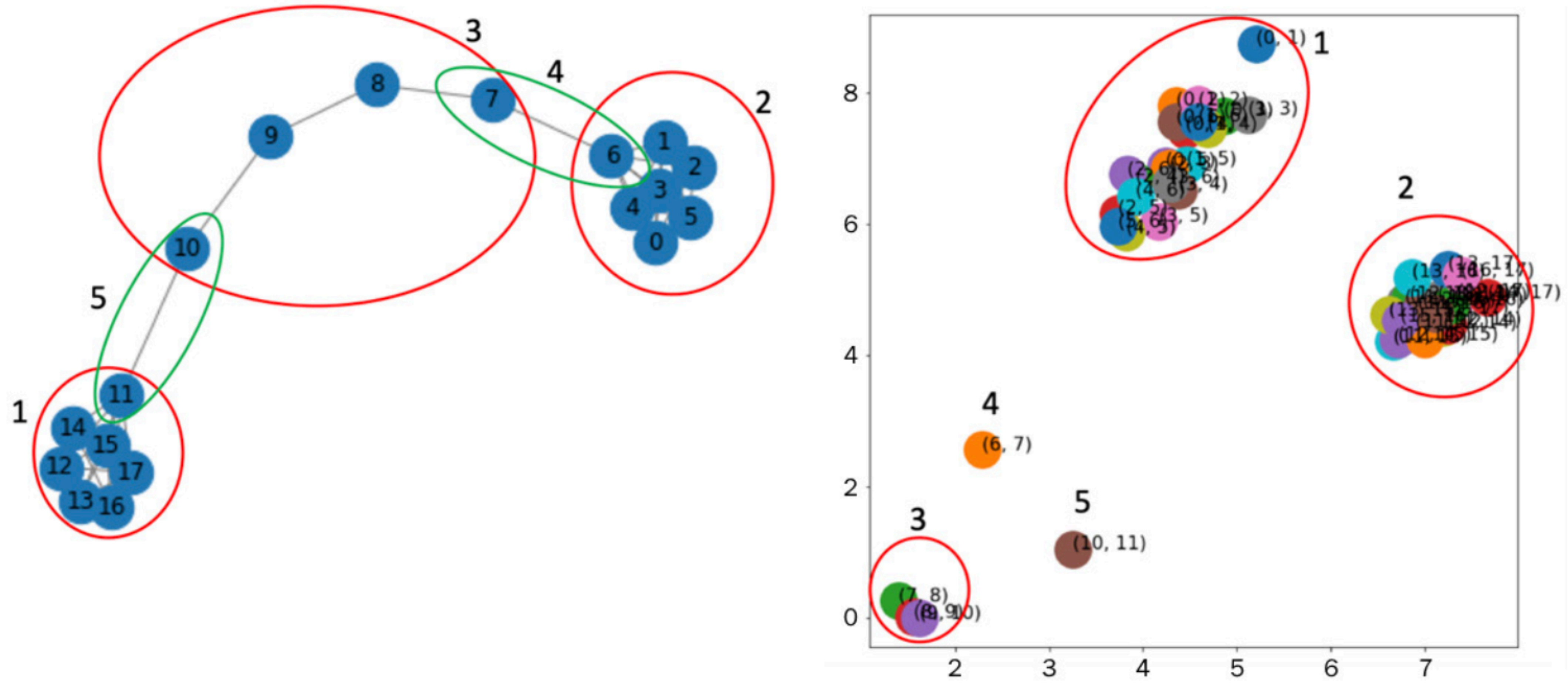
THE GENERALIZED GRAPH EMBEDDING PROBLEM – REPRESENTATION LEARNING

- *Representation learning* (network embedding) is the task that aims to learn a mapping function $f: G \rightarrow \mathbb{R}^d$ from a discrete graph to a continuous domain.
- Function will be capable of performing a low-dimensional vector representation such that the properties (local and global) of graph are preserved.

- Node Level Embedding



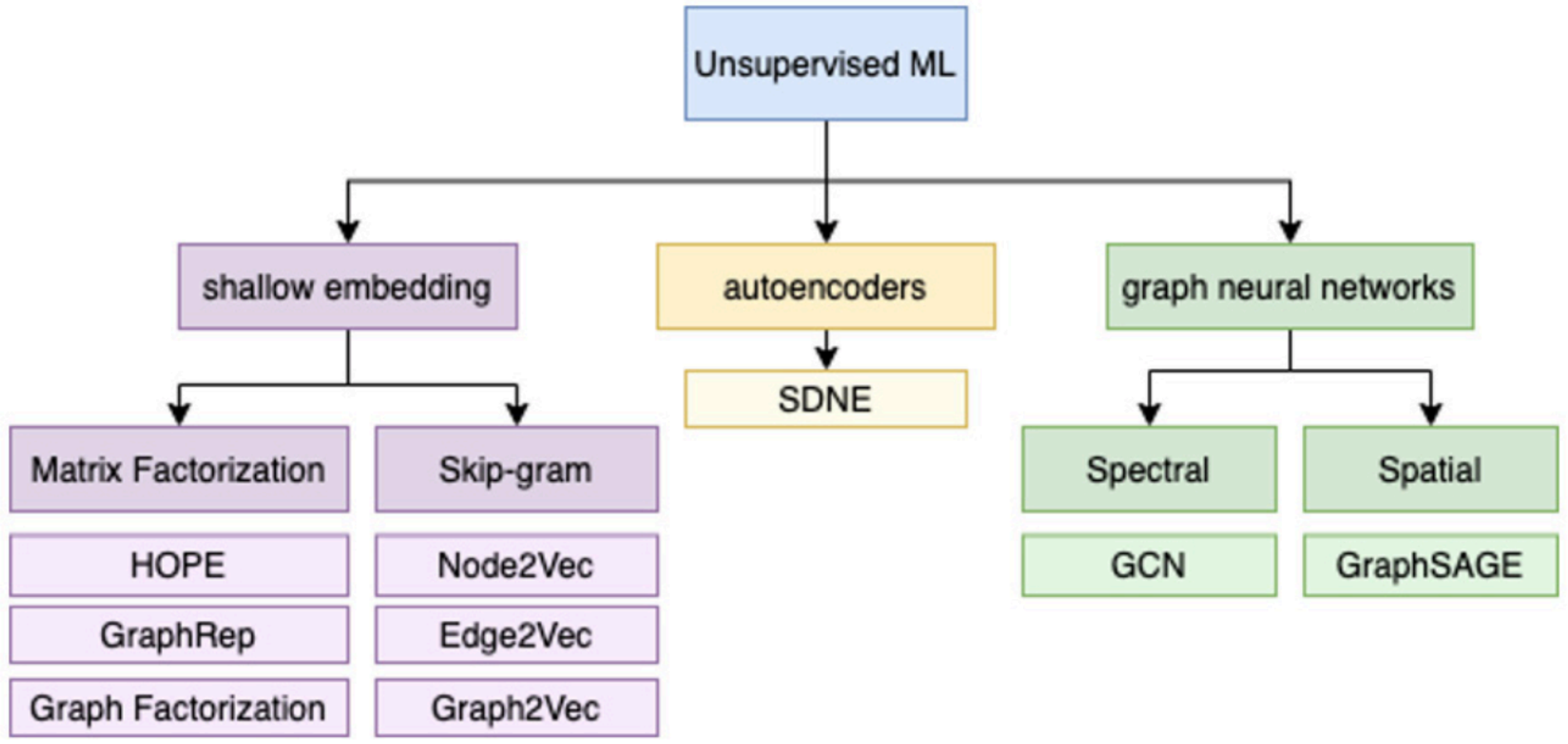
- Edge Level Embedding

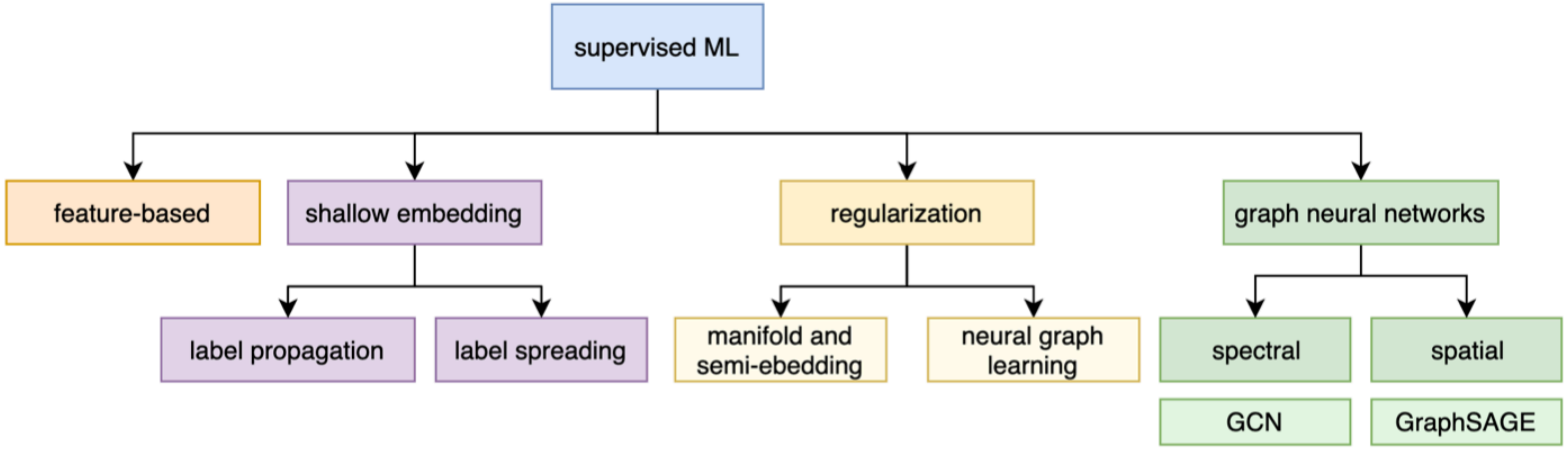


GRAPH EMBEDDING ALGORITHMS

Machine Learning on Graphs:
A Model and Comprehensive Taxonomy

Ines Chami*†, Sami Abu-El-Haija‡, Bryan Perozzi††, Christopher Ré‡‡, and Kevin Murphy††

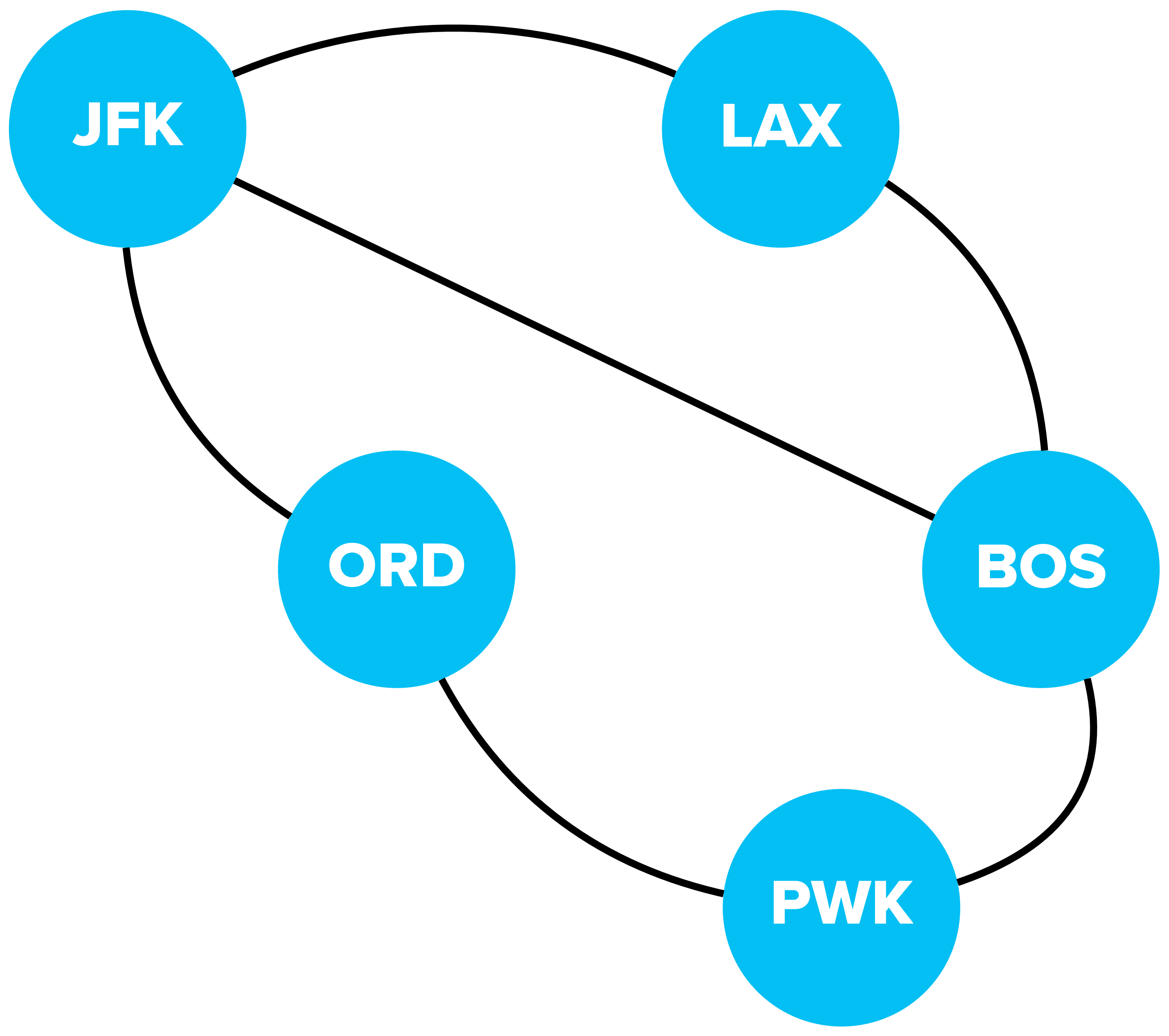




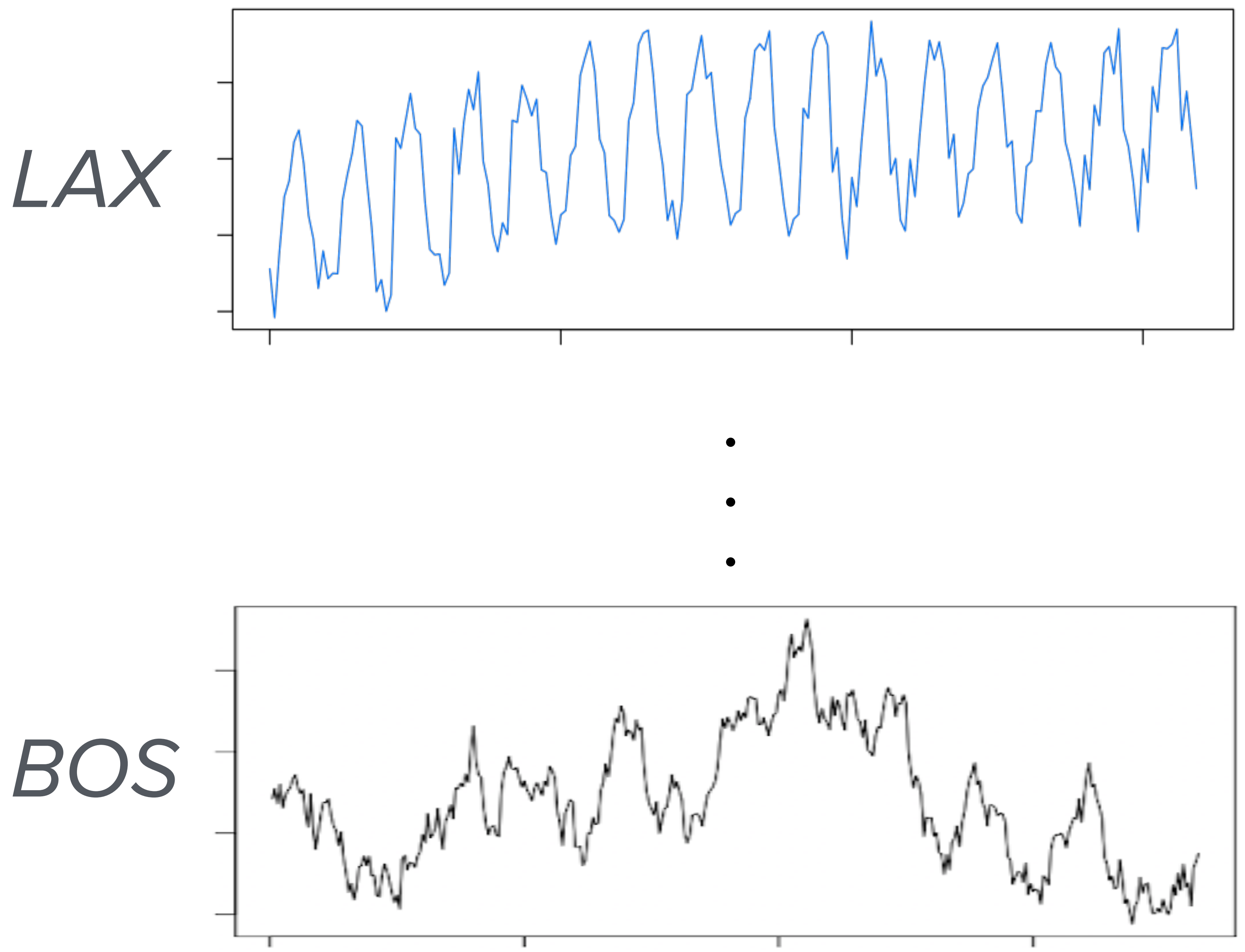
OK GRAPHS LOOK COOL. BUT HOW CAN I USE THEM IN THIS PROJECT ?

- *Airport are strongly connected*
- *Less passenger in one airport can cause more or less in another*
- *T-GCN: A Temporal Graph Convolutional Network for Traffic Prediction (https://github.com/lehaifeng/T-GCN)*

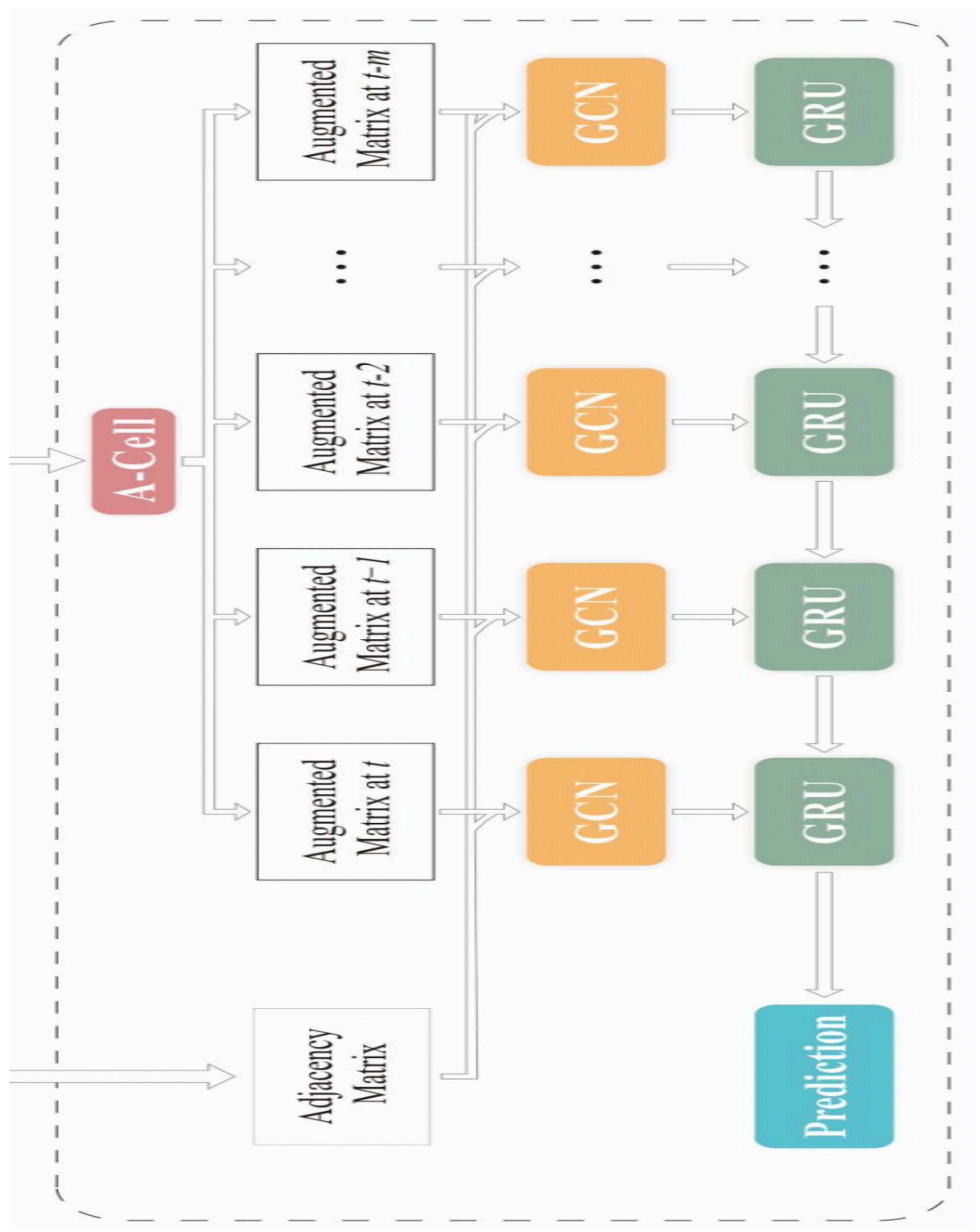
- **Connections between airports**



- **Number of passenger per airpot**



- **T-GCN**



GRAPH MACHINE LEARNING IS ALSO A BOOK