ROI TRAINING

MAXIMIZE YOUR TRAINING INVESTMENT

Python Program

CHAPTER 9: GRAPH DATABASE

Chapter Objectives

In this chapter, we will:

- Introduce the concept of graph databases
- Explore Python's NetworkX package

Chapter Concepts

Graph Databases

NetworkX

Chapter Summary

Graph Database

- Relational databases are useful for storing fields in a tabular format
- To show one record is connected to another in some way, you would build a related table and create a foreign key between them
- This can get messy to encode complex data with many relationships
- Graph databases encode the data in a different manner which places equal weight to the data of the record and the relationship it has with other records
- Once data is encoded in a graph database format, it is able to be queried in ways that would be difficult and time consuming in a relational database
- Graph databases have many use cases
  - Network and social analysis
  - Fraud detection
  - Supply chain transparency
  - Infrastructure monitoring

Terminology

- Graph databases are composed of nodes and edges instead of tables made up of rows and columns
- A node is like a record in that it stores data, but it is much less structured and more free form
  - Can store any data you like as a key-value pair
  - Much like JSON or Python dictionaries
  - Not all nodes have to have the same data
- Edges are connections between nodes and are like a relationship between objects
  - Edges themselves can have additional data, just like a node
  - A common data element found in an edge is weight, indicating how strong a relationship there is between the two nodes
  - Edges can be directed or undirected, meaning they flow one way or both ways

Common Queries

- There are two categories of queries you can run on graph databases
  - Egocentric answers questions about particular nodes
    - How many other nodes does it connect to?
    - How important is the node?
    - Is it centrally located or near the borders?
    - What other types of nodes is it connected to?
    - How far is it from one node to another?
    - What is the shortest path between two particular nodes?
  - Sociocentric answers questions about the network as a whole
    - Are there particular combinations of nodes that are more important?
    - Are there natural clusters or communities within the network?
    - Are there bottlenecks that could break down the network if they fail?
    - Where could we add additional connections to facilitate a more robust network or better connectivity between nodes?
    - Do nodes tend to connect to other similar nodes?

Chapter Concepts

Graph Databases

NetworkX

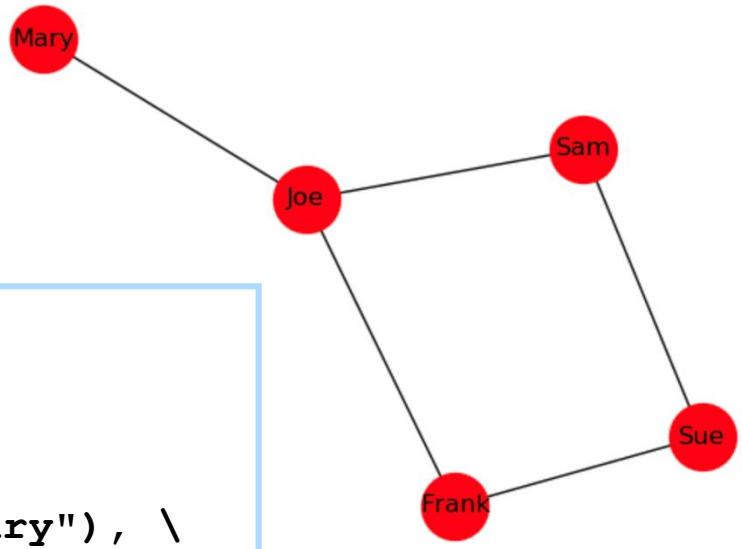
Chapter Summary

Tools

- There are many tools for graph databases
  - Neo4J is a full-fledged database cluster for storing and manipulating large graph databases
  - NetworkX is a Python library that has a lot of features and is suitable for running queries on data that may be stored in regular places, such as text files
  - GraphX is a Spark library that can be used with Python, Scala, and Java and works on a cluster to handle larger datasets
- All use the same concepts and terminologies and can solve the same problems—they just use different syntax to do it

NetworkX

- Easy to use Python package that supports creating in-memory graph databases that can be analyzed
- Integrates nicely with Python plotting packages to make it easy to display graph data
- The circles represent nodes
- The lines between the nodes are edges



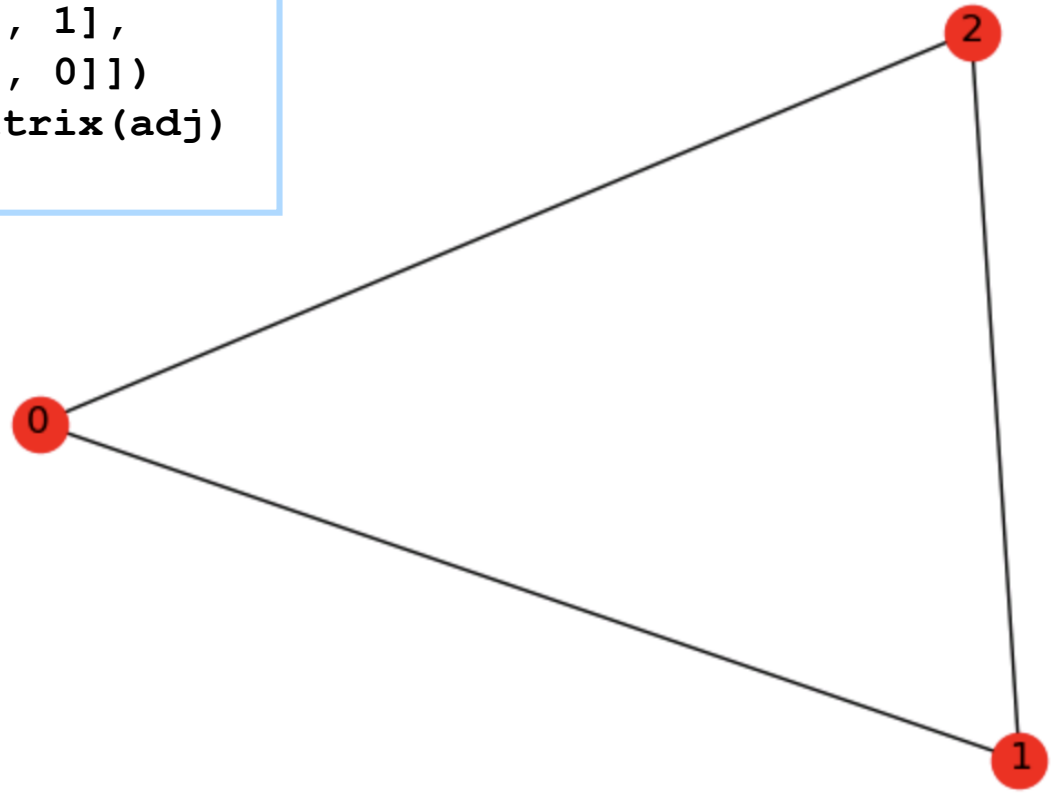

```
pip install networkx

import networkx as nx
G_names = nx.Graph()
G_names.add_edges_from([("Joe", "Mary"), \
("Joe", "Frank"), ("Sue", "Frank"), \
("Sam", "Joe"), ("Sue","Sam")])
nx.draw_networkx(G_names, node_size = 1000)
```

Reading Data from NumPy

- Data can be easily read from NumPy arrays

```
import numpy as np
adj = np.array([[0, 1, 1],
                [1, 0, 1],
                [1, 1, 0]])
G2 = nx.from_numpy_matrix(adj)
nx.draw_networkx(G2)
```

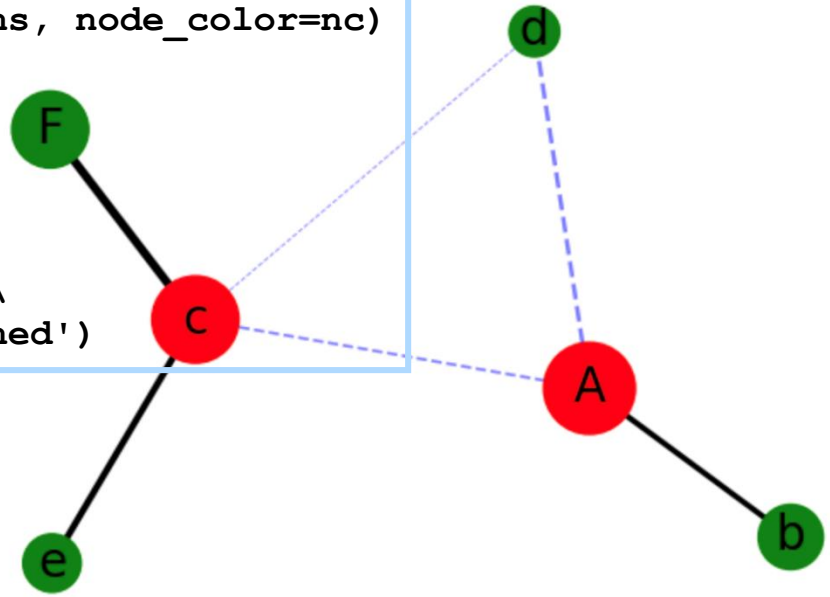


Changing Plot Attributes

- Using additional data in the nodes and edges, the plot can customize features such as size, shape, color, line thickness, and style to create a visual representation of the features that will help identify meaningful information

```
eigen = nx.eigenvector_centrality(G)
ns = [15 * x['size'] for _,x in G.nodes(data=True)]
nc = ['r' if x >= .5 else 'g' for x in eigen.values()]
nx.draw_networkx_nodes(G, pos, node_size=ns, node_color=nc)

# edges
nx.draw_networkx_edges(G, pos, \
    edgelist=elarge, width=elargeweight)
nx.draw_networkx_edges(G, pos, \
    edgelist=esmall, width=esmallweight, \
    alpha=0.5, edge_color='b', style='dashed')
```



Common Functions

- Once you have data encoded as nodes and edges, there are tons of functions that can be run on the data to find interesting features that can be used to find particular data or display it by changing various visual attributes
- Some common egocentric functions:
  - **`degree_centrality`** – number of links in or out
  - **`closeness_centrality`** – how quickly info can pass to other from here
  - **`eigenvector_centrality`** – how well connected to important nodes
  - **`betweenness_centrality`** – how likely to be in communication path
  - **`shortest_path`** – shortest route between two nodes
  - **`diameter`** – shortest distance between two furthest nodes
- Some common sociocentric functions:
  - **`degree_assortativity_coefficient`** – useful in finding how homogeneous the data is
  - **`find_cliques`** – find sections of graph where all nodes are connected to each other
  - **`density`** – describes how many potential connections actually exist

Chapter Concepts

Graph Databases

NetworkX

Chapter Summary

Next Steps

- We only had time to explore the tip of the iceberg for graph databases
- Learn about use cases and solutions that graph databases can solve and see how companies like Facebook, LinkedIn, and more have applied these techniques
- Explore technologies such as:
  - Neo4J
  - NetworkX
  - GraphX for Spark

Chapter Summary

In this chapter, we have:

- Introduced the concept of graph databases
- Explored Python's NetworkX package