AI & MACHINE LEARNING

Use graphs for smarter AI with Neo4j and Google Cloud Vertex AI

Ben Lackey

and verbs. Nodes, or the nouns, are things such as people, places, and items. Relationships, or the verbs, are how they're connected. People know each other and items are sent to places. The signal in those relationships is powerful.

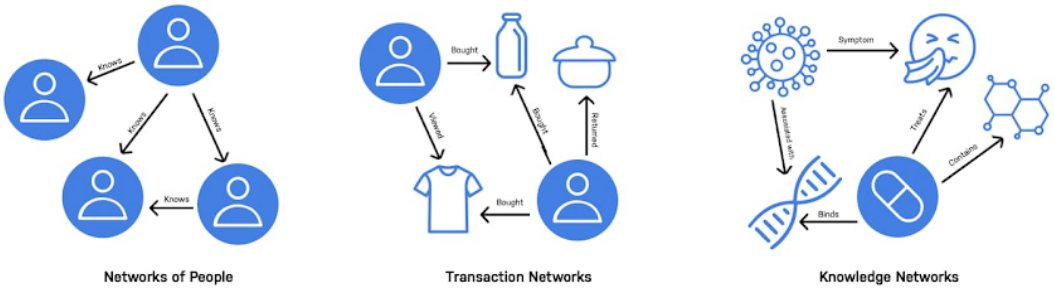


Figure 1. Everything is naturally connected.

Graph data can be huge and messy to deal with. It is nearly impossible to use in traditional machine learning tasks.

Google Cloud and Neo4j offer scalable, intelligent tools for making the most of graph data. Neo4j Graph Data Science and Google Cloud Vertex AI make building AI models on top of graph data fast and easy.

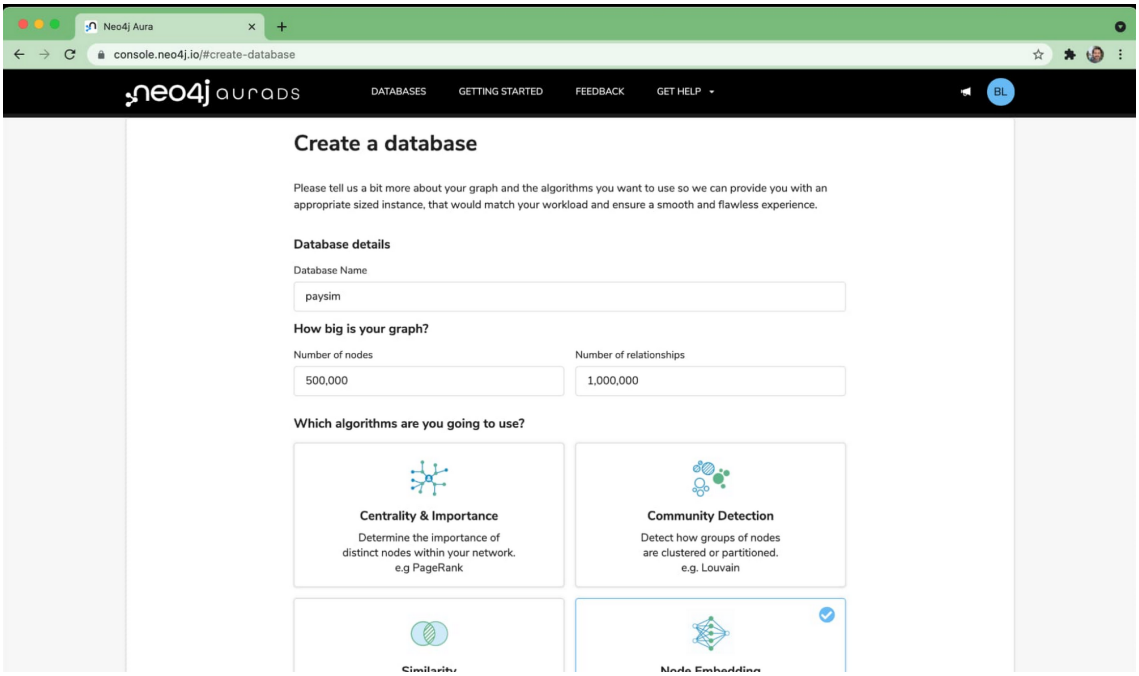
Dataset - Identify Fraud with PaySim

Graph based machine learning has numerous applications. One common application is

graph features.

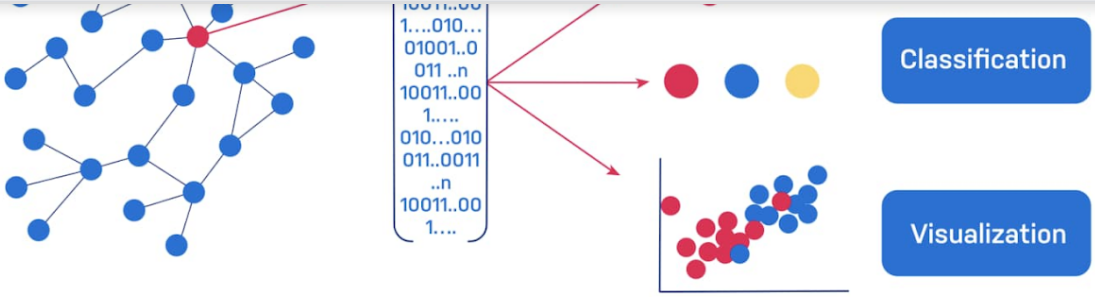
Loading Data into Neo4j

First off, we need to load the dataset into Neo4j. For this example, we're using AuraDS. AuraDS offers Neo4j Graph Database and Neo4j Graph Data Science running as a managed service on top of GCP. It's currently in a limited preview that you can sign up for here.



1	Client	11270
2	Bank	5
3	Merchant	3465
4	Mule	0
5	CashIn	746751
6	CashOut	424574
7	Debit	130284
8	Payment	542443
9	Transfer	0

The notebook gives examples of other queries including relationship types and transaction types as well. You can explore those yourself here.



visualization of two weakly connected components

A different approach is to use Neo4j to generate graph embeddings. Graph embeddings boil down complex topological information in your graph into a fixed length vector where related nodes in the graph have proximal vectors. If graph topology, for example who fraudsters interact with and how they behave, is an important signal, the embeddings will capture that so that previously undetectable fraudsters can be identified because they have similar embeddings to known fraudsters.

Task: Similarity of embeddings between nodes is reflective of the similarity in the actual graph

Example: Who is Zachary….?

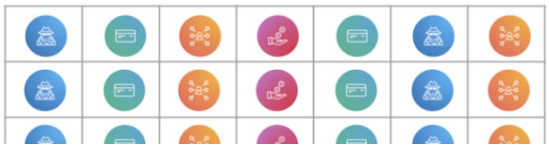
```
  propertyRatio: 0.25,
  nodeSelfInfluence: 0.15,
  embeddingDimension: 16,
  randomSeed: 1,
  mutateProperty:'embedding'
})
```

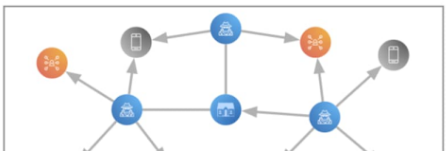
That creates a 16 dimensional graph embedding using the Fast Random Project algorithm. One neat feature in this is the nodeSelfInfluence parameter. This helps us tune how much nodes further out in the graph influence the embedding.

With the embedding calculated, we can now dump it into a pandas dataframe, write that to a CSV and push that to a cloud storage bucket where Google Cloud's Vertex AI can work with it. As before, these steps are detailed in the notebook here.

Machine Learning with Vertex AI

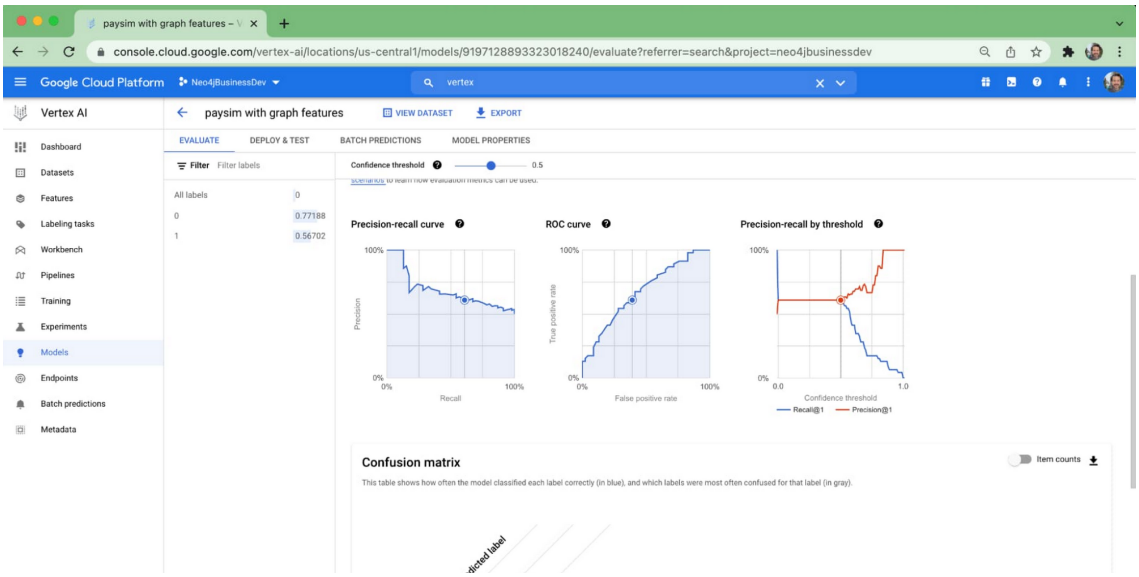
Now that we've encoded the graph dynamics into vectors, we can use tabular methods in Google Cloud's Vertex AI to train a machine learning model.





```
    target_column="is_fraudster",
    training_fraction_split=0.8,
    validation_fraction_split=0.1,
    test_fraction_split=0.1,
    model_display_name="paysim-prediction-model",
    disable_early_stopping=False,
    budget_milli_node_hours=1000,
)
```

You can view the results of that call in the notebook. Alternatively, you can login to the GCP console and view the results in the Vertex AI's GUI.



Specific areas we'd like to explore in future work include:

Improved Dataset - For data privacy reasons, it's very difficult to publicly share fraud datasets. That led us to use the PaySim dataset in this example. That is a synthetic dataset. From our investigation, both of the dataset and the generator that creates it, there seems to be very little information in the data. A real dataset would likely have more structure to explore.

In future work we'd like to explore the graph of SEC EDGAR Form 4 transactions. Those forms show the trades that officers of public companies make. Many of those people are officers at multiple companies, so we anticipate the graph being quite interesting. We're planning workshops for 2022 where attendees can explore this data together using Neo4j and Vertex AI. There is already a loader that pulls that data into Google BigQuery here.

Boosting and Embedding - Graph embeddings like Fast Random Projection duplicate the data because copies of sub graphs end up in each tabular datapoint. XGBoost, and other boosting methods, also duplicate data to improve results. Vertex AI is using XGBoost. The result is that the models in this example likely have excessive data duplication. It's quite possible we'd see better results with other machine learning methods, such as neural networks.

Graph Features - In this example we automatically generated graph features using the embedding. It's also possible to manually engineer new graph features. Combining