Extracting graph features to train an explainable predictive model

Exercise: Predicting retail purchase fraud

In this exercise, we take a synthetic dataset for retail purchase fraud, model it as a graph, extract various graph-based and non-graph-based features, export the feature vectors, and use conventional ML techniques to train a fraud prediction model. The graph model enables the

This exercise is taken directly from an exercise in Andrew Ng's Coursera online course on Machine Learning.

Data Set:
The dataset is 594,643 synthetic retail transactions, each with a Customer and a Merchant, created by a simulator called Banksim. The simulator has modeled fraud, where approximately 2% of the transactions are labeled as fraudulent. The dataset is available from Kaggle: https://www.kaggle.com/ntnu-testimon/banksim1

Lopez-Rojas, Edgar Alonso ; Axelsson, Stefan Banksim: A bank payments simulator for fraud detection research. In Proceedings 26th European Modeling and Simulation Symposium, EMSS 2014, Bordeaux, France, pp. 144–152, Dime University of Genoa, 2014, ISBN: 9788897999324. https://www.researchgate.net/publication/265736405_BankSim_A_Bank_Payment_Simulation_for_Fraud_Detection_Research

Machine Learning Technique:
Random Forest - aggregate result from running multiple Decision Trees, each based on a random subset from the full dataset.

Instructions for Graph Setup and Feature Extraction

1. In a Linux shell for your TigerGraph machine, change to the bank_fraud/setup folder:
   `cd /home/tigergraph/usecases/bank_fraud/setup`

2. In the setup folder, we have a `bank_setup.sh` script written for you to i) create the graph schema for the Bank graph; ii) load the data into the graph; iii) define and install the queries to extract graph features.(You are encouraged to take a look at the setup script) So you just need to run:

```
bash bank_setup.sh
```

3. Now go to GraphStudio. In the upper left corner, select the Bank graph. Switch to the Design Schema page to get familiar vertex types and edge types in this schema. Hover the cursor over each vertex to see the attributes include for each vertex. Note that in addition to attributes from the banksim data (such as Customer age and Transaction amount), each vertex also as a Map<STRING, DOUBLE>. This is an extensible container to hold various graph features as <feature_name, value> pairs.

4. The queries below compute the features of the customers, merchants, and transactions in the graph, and add the features into the attr_map of each vertex. Here is a description of each one:
    i. For each customer, the average amount of money per transaction: `avg_amt_per_transaction.gsql`
    ii. For each customer, number of transactions per day: `avg_num_of_transactions_per_day.gsql`
    iii. For each customer, the average number of different merchants he/she has transaction to per day: `avg_num_of_diff_merchants_per_day.gsql`
    iv. For each customer, the number and percentage of fraud transactions he/she has: `num_of_fraud.gsql`
    v. For each merchant, the number and percentage of fraud transactions it has: `num_of_fraud_merchant.gsql`
    vi. For each transaction, whether its category is one of the top 3 categories (ranked by the number of transactions) for that customer, as well as in the top 3 categories when ranked by the amount of transactions: `top_categories.gsql`
    vii. For each transaction, whether its merchant is one of the top 5 merchants (ranked by the number of transactions) for that customer, as well as whether it is in the top 5 merchants when ranked by the amount of transactions. It will also find the number of merchants each customer has transacted with: `top_merchants.gsql`
    viii. If we start from a transaction and traverse along: Transaction (start)->Merchant->Transaction (1st closest)->Customer->Transaction (2nd closest)->Merchant->Transaction (3rd closest), the number of fraud transactions in 1st, 2nd and3rd closest transactions. `fraud_in_nHops.gsql`
    * the query (viii) depends on the result of queries (iv and v). Please run (iv and v) before (viii)
    ** query (viii) can take up to 1 hour on a typical single server.

5. Due to the limited time for this workshop, we will only run a few of the queries. In GraphStudio, run one of query which has no input parameters, such as

`avg_amt_per_transaction.gsql`
Then examine the atttr_map values for a few transaction vertices.

6. To output the all the features for each transaction to a file, run `print_transaction_features.gsql`

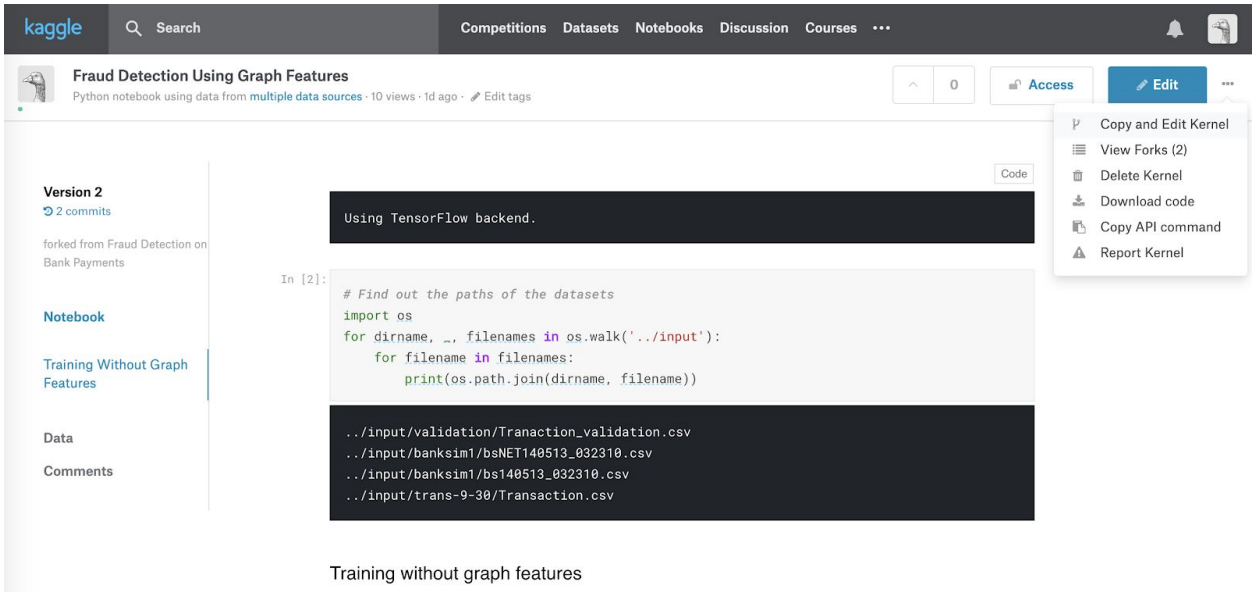
   NOTE 1: If you did not run all the queries, you need to modify this query to include only the features you have.

   NOTE 2: The output file is always.
   `/home/tigergraph/usecases/bank_fraud/data/training_data/Transaction.csv`
   If you run it more than once, it will overwrite the previous results. If you want to keep the earlier output, you need to rename the output file or modify the print_transaction_features query.

Instructions for Machine Learning Training

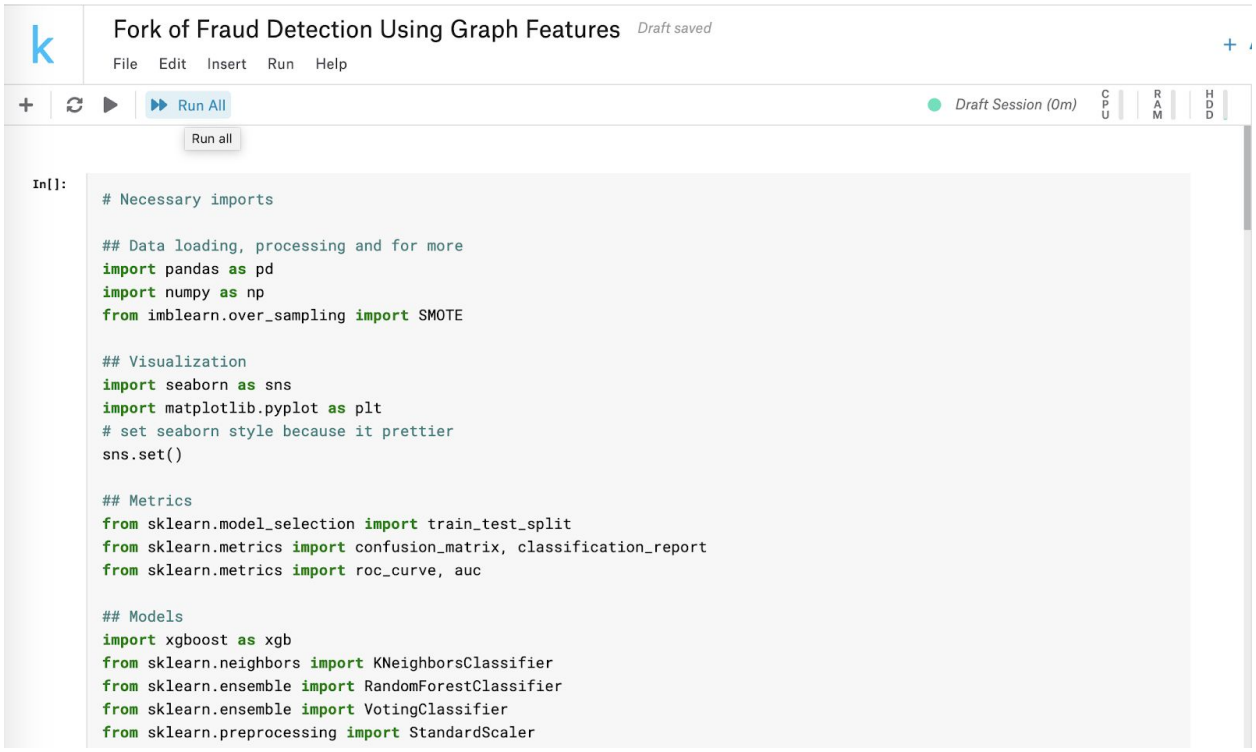
1. Go to the following analytics notebook page which we have created on Kaggle:
   https://www.kaggle.com/victortiger/fraud-detection-using-graph-features
   We have already loaded the feature data `transaction.csv` into the notebook. To see how to upload data, see the Appendix below.

2. Sign in to your Kaggle account or create one if you need to.
3. In the upper right corner, click the blue box "**Copy and Edit Kernel**".



Training without graph features

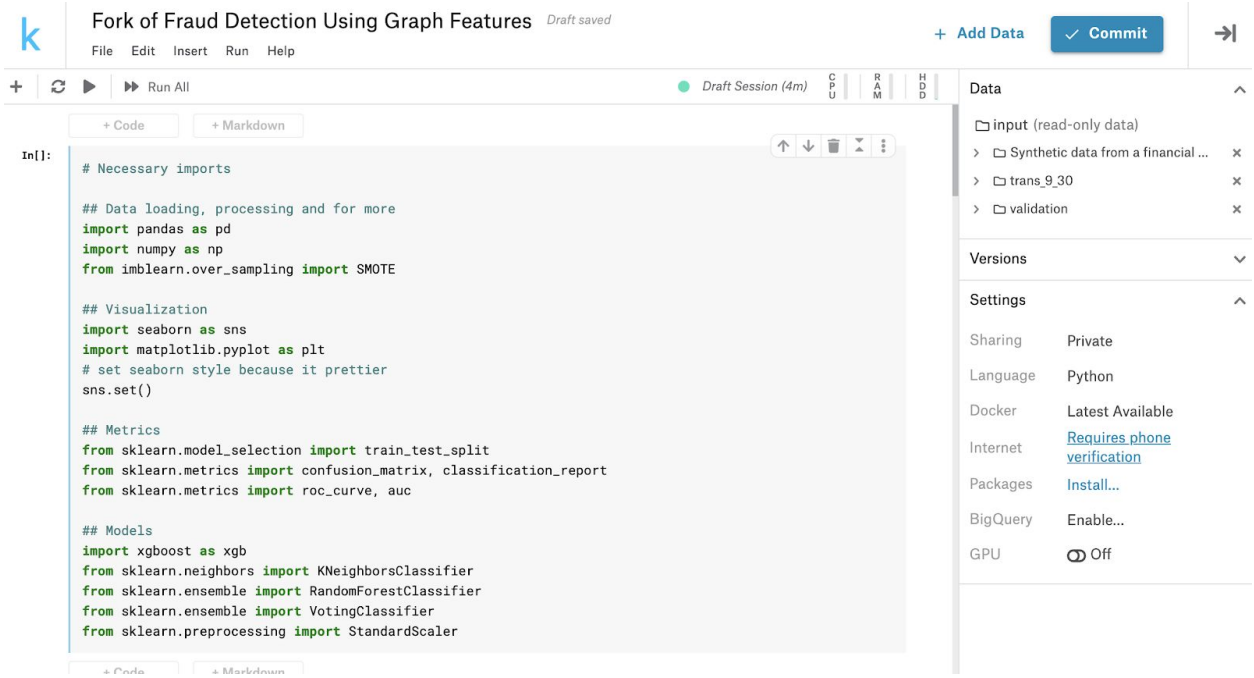
4. In the upper left corner, click the "**Run All**" button.



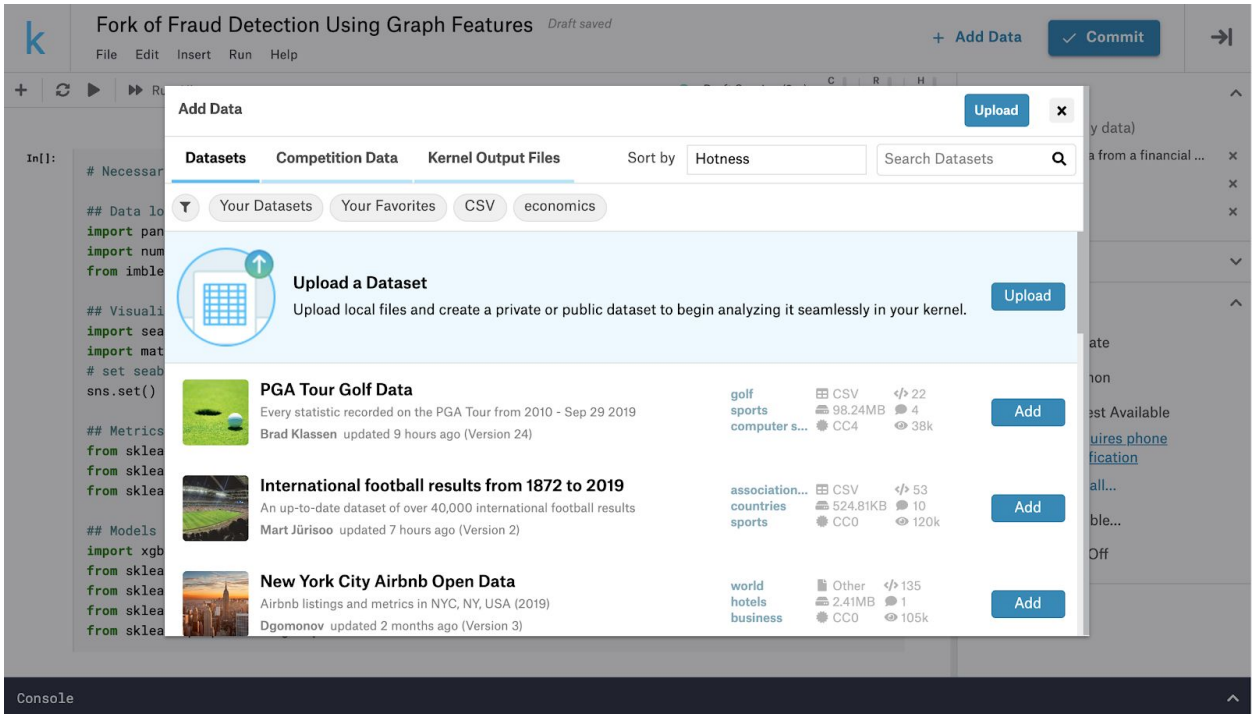
5. Look at the Random forest results for training without graph features and with graph features.

Appendix: Uploading Data into a Kernel in Kaggle

1. Click the "**+ Add Data**" button



2. Click "**Upload a Dataset**"



3. Select the path of the dataset and give it a name.

4. Run the second code chunk in the notebook to see the path of your dataset on Kaggle.

```
# Find out the paths of the datasets
import os
for dirname, _, filenames in os.walk('../input'):
    for filename in filenames:
        print(os.path.join(dirname, filename))
```

```
../input/validation/Tranaction_validation.csv
../input/banksim1/bsNET140513_032310.csv
../input/banksim1/bs140513_032310.csv
../input/trans-9-30/Transaction.csv
```