Big Data Group 9 Project

Real-Time Credit Card Fraud Detection

SUMMARY

Team members: Lydia Kwan, Steeve Laberge, Aiqing Lan, Ann Li, Sigao Li, Chun Li

# Objectives

This custom project involved the implementation of a real-time framework for credit card fraud predictions. Our specific objectives were as follows:

1. Conduct standard ML pipeline development activities, including exploratory data analysis, feature engineering, and model design and evaluation as taught in the program.
2. Explore the techniques for streamed transaction processing and ML predictions using Spark Streaming and Kafka.

# Analysis

## Dataset Selection

Locating authentic credit fraud data proved to be challenging, which is understandable given the sensitivity of such information. The depersonalized datasets we found mostly had dimension-reduced features, which would have made data analysis much less meaningful to us. As such, we settled for this synthetic data set which was seen by many as realistic and rich in patterns: <https://www.kaggle.com/datasets/kartik2112/fraud-detection/data>

## Exploratory Data Analysis (EDA) Results

The dataset consisted of 1.2 million records stored as a 500MB CSV file. The main conclusions reached at the end of the EDA phase were as follows:

* The target class (is\_fraud) is heavily imbalanced. Less than 1% of transactions are fraudulent
* The fraud percentage is higher during the evenings, especially between 22:00 and 02:00.
* Online shopping and miscellaneous internet-based transactions exhibit the highest rates of fraud
* Seniors aged 80+ are more likely to be victims of fraud than folks in other age groups.
* Fraudulent activity is at its peak for transactions under <1500$

## Model Training and Evaluation

Once the EDA phase was completed, we selected the features that exhibited the most variance and then trained two tree-based classification models, RandomForest and GradientBoostingTrees, and one feed forward neural network, MultiLayerPerceptron. For the tree-based classifiers, we balanced the dataset by computing a “weight” column and then instructing the classifiers to use that column as a weight factor. For the neural network, we downsampled the dataset to make both classes equal.

A challenge of this phase was to keep training time within reasonable limits. At 1.2M records, we used subsampling and a single model approach with limited scope parameters for the better part of the analysis. We cross-validated and tuned the hyper-parameters near the end of the project with the best machines we had on hand.

The table below compares the performance of each model

| **Evaluation metrics** | **Model #1: Random Forest** | **Model #2: Gradient Boosting Trees** | **Model #3: Feedforward Neural Network** |
| --- | --- | --- | --- |
| areaUnderPR | 0.525 | 0.770 | 0.419 |
| Confusion matrix |  |  |  |
| Precision | 0.142 | 0.277 | 0.644 |
| Recall | 0.819 | 0.912 | 0.516 |
| F1 Score | 0.243 | 0.425 | 0.573 |

## Real-Time Pipeline Implementation

After evaluation of different algorithms and tuning, the best model was retained and used to support the real-time prediction process. The first streaming prototype was based on CSV file inputs. The second prototype was based on Apache Kafka.

One major challenge we faced in this phase of the project was to incorporate into the real-time pipeline the various data transformations and extra features needed to establish spending patterns. The Spark MLlib package provided a number of common transformations (e.g. VectorAssembler, StringIndexer, OneHotEncoder, etc..). However, to achieve our goal, we had to define our own transformers and design intermediate aggregate queries that were then joined with the main event stream.

# Conclusions

This project proved to be as challenging as it was rewarding. From a data analysis and fraud detection standpoint, our main conclusions were as follows:

* Most fraudulent transactions involve larger than usual amounts and occur late in the evening.
* Credit card fraud detection is very much a matter of compromise. We tuned our model to strike the best balance between precision and recall (best F1 score). However a credit card agency may choose to prioritize one over the other as a function of business objectives.

Regarding the technical ML pipeline aspects, our main takeaways were as follows:

* Out of the classification algorithms we evaluated, GradientBoostClassification yielded the highest degree of accuracy for this particular use case.
* In spite of similarities between static and streamed dataframe APIs, an ML pipeline must be adjusted to the micro-batch approach used for all operations.

**Links**

Main project notebook

<https://databricks-prod-cloudfront.cloud.databricks.com/public/4027ec902e239c93eaaa8714f173bcfc/1268838149283039/2479943025611017/2477172884221098/latest.html>

Kafka notebook

<https://databricks-prod-cloudfront.cloud.databricks.com/public/4027ec902e239c93eaaa8714f173bcfc/1268838149283039/387308155055381/2477172884221098/latest.html>