Lab 2

Brittny Hopwood

Use Case: Fraud Detection

Submitted 3/9/24

Source: https://github.com/aws-samples/aws-fraud-detector-samples/tree/master/data

Amazon Fraud Detector - Data Profiler Notebook

Dataset Guidance

AWS Fraud Detector's Online Fraud Insights(OFI) model supports a flexible schema, enabling you to train an OFI model to your specific data and business need. This notebook was developed to help you profile your data and identify potenital issues before you train an OFI model. The following summarizes the minimimum CSV File requirements:

- The files are in CSV UTF-8 (comma delimited) format (*.csv).
- The file should contain at least 10k rows and the following **four** required fields:
  - Event timestamp
  - IP address
  - Email address
  - Fraud label
- The maximum file size is 10 gigabytes (GB).
- The following dates and datetime formats are supported:
  - Dates: YYYY-MM-DD (eg. 2019-03-21)
  - Datetime: YYYY-MM-DD HH:mm:ss (eg. 2019-03-21 12:01:32)
  - ISO 8601 Datetime: YYYY-MM-DDTHH:mm:ss+/-HH:mm (eg. 2019-03-21T20:58:41+07:00)
- The decimal precision is up to four decimal places.
- Numeric data should not contain commas and currency symbols.

- Columns with values that could contain commas, such as address or custom text should be enclosed in double quotes.

Getting Started with Data

The following general guidance is provided to get the most out of your AWS Fraud Detector Online Fraud Insights Model.

- Gathering Data - The OFI model requires a minimum of 10k records. We recommend that a minimum of 6 weeks of historic data is collected, though 3 - 6 months of data is preferable. As part of the process the OFI model partitions your data based on the Event Timestamp such that performance metrics are calculated on the out of sample (latest) data, thus the format of the event timestamp is important.

- Data & Label Maturity: As part of the data gathering process we want to insure that records have had sufficient time to “mature”, i.e. that enough time has passed to insure “non-fraud" and “fraud” records have been correctly identified. It often takes 30 - 45 days (or more) to correctly identify fraudulent events, because of this it is important to insure that the latest records are at least 30 days old or older.

- Sampling: The OFI training process will sample and partition historic based on event timestamp. There is no need to manually sample the data and doing so may negatively influence your model’s results.

- Fraud Labels: The OFI model requires that a minimum of 500 observations are identified and labeled as “fraud”. As noted above, fraud label maturity is important. Insure that extracted data has sufficiently matured to insure that fraudulent events have been reliably found.

- Custom Fields: the OFI model requires 4 fields: event timestamp, IP address, email address and fraud label. The more custom fields you provide the better the OFI model can differentiate between fraud and not fraud.

- Nulls and Missing Values: OFI model handles null and missing values, however the percentage of nulls in key fields should be limited. Especially timestamp and fraud label columns should not contain any missing values.

If you would like to know more, please check out the Fraud Detector's Documentation.

In [32]:

```
from IPython.core.display import display, HTML
from IPython.display import clear_output
display(HTML("<style>.container { width:90% }</style>"))
from IPython.display import IFrame
# -----------------------------------------------------------------
import numpy as np
import pandas as pd
```

```
pd.set_option('display.max_rows', 500)
pd.set_option('display.max_columns', 500)
pd.set_option('display.width', 1000)
pd.options.display.float_format = '{:.4f}'.format
df = pd.read_csv('transaction_data_100K_full.csv', low_memory=False)
# -- AWS stuff --
import boto3
```

```
/tmp/ipykernel_846/3252863927.py:1: DeprecationWarning: Importing display from IPyth
on.core.display is deprecated since IPython 7.14, please import from IPython display
  from IPython.core.display import display, HTML
```

In [ ]:

```

```

Amazon Fraud Detector Profiling

from github download and copy the afd_profile.py python program and template directory to your notebook

afd_profile.py

- afd_profile.py - is the python package which will generate your profile report.
- /templates - directory contains the supporting profile templates

In [33]:

```
# -- get this package from github --
import afd_profile
```

Intialize your S3 client

https://boto3.amazonaws.com/v1/documentation/api/latest/reference/services/s3.html

In [34]:

```
client = boto3.client('s3')
```

File & Field Mapping

Simply map your file and field names to the required config values.

Map the Required fields

- input_file: this is your CSV file in your s3 bucket

required_features are the minimally required freatures to run Amazon Fraud Detector

- EVENT_TIMESTAMP: map this to your file's Date or Datetime field.
- IP_ADDRESS: map this to your file's IP address field.

- EMAIL_ADDRESS: map this to your file's email address field.
- FRAUD_LABEL: map this to your file's fraud label field.

  note: the profiler will identify the "rare" case and assume that it is fraud

In [35]:

```
# -- update your configuration --
config = {
    "input_file"        : "transaction_data_100K_full.csv",
    "required_features" : {
        "EVENT_TIMESTAMP" : "EVENT_TIMESTAMP",
        "EVENT_LABEL"     : "EVENT_LABEL",
        "IP_ADDRESS"      : "ip_address",
        "EMAIL_ADDRESS"   : "customer_email"
    }
}
```

Run Profiler

The profiler will read your file and produce an HTML file as a result which will be displayed inline within this notebook.

Note: you can also open **report.html** in a separate browser tab.

In [36]:

```
# -- generate the report object --
report = afd_profile.profile_report(config)
```

```
/home/sagemaker-user/data-science-on-aws/02_usecases/fraud_detector/profiler/afd_pro
file.py:39: UserWarning: The argument 'infer_datetime_format' is deprecated and will
be removed in a future version. A strict version of it is now the default, see http
s://pandas.pydata.org/pdeps/0004-consistent-to-datetime-parsing.html. You can safely
remove this argument.
  df['EVENT_TIMESTAMP'] = pd.to_datetime(df[config['required_features']['EVENT_TIMES
TAMP']], infer_datetime_format=True)
0
```

In [37]:

```
with open("report.html", "w") as file:
    file.write(report)

IFrame(src='report.html', width=1500, height=800)
```

Out[37]:

Amazon Fraud Detector CSV

Overview

transaction_data_100K_full.csv

Record count

Column count

Duplicate count

Memory size

Record size

Date range

Day count

Field Summary

Name	Type	Feat
EVENT_LABEL	int64	EVEI
EVENT_TIMESTAMP	datetime64[ns, UTC]	EVEI
ENTITY_ID	object	cate
card_bin	int64	num

In [ ]: