Fraudulent Transaction Detection

Business Objective

In the Banking or payment industry, fraud is an illegal usage of credit card details without the real cardholder's knowledge. A stolen credit card/card number is usually the cause of a fraudulent charge. Once a cardholder sees a payment transaction, he did not make on his credit card statement, he/she has the right to dispute the charge by contacting his/her bank. The bank or Credit Card Company conducts an investigation and returns the money to the cardholder.

But what if we can detect fraudulent transaction activity in real-time? This way we can take the required action to stop the same. We can use machine learning techniques to detect fraudulent transactions. We can use supervised or unsupervised methods of learning depending upon the dataset. For this project, we will be opting for unsupervised learning using Isolation Forest and Local Outlier Factor (LOF) algorithms.

Isolation Forests are similar to Random forests that are built based on decision trees. There are no pre-defined labels here and hence it is an unsupervised algorithm. It was built based on the fact that anomalies or outliers are the data points that are "few and different." In an Isolation Forest, randomly sub-sampled data is processed in a tree structure based on randomly selected features. The sub-samples that travel deeper into the tree are less likely to be anomalies as they required more cuts to isolate them. The ones which ended up in shorter branches indicated anomalies as it was easier for the tree to separate them from other observations.

Local Outlier Factor or LOF is an unsupervised ML algorithm that identifies outliers with respect to the local neighborhoods as opposed to using the entire data distribution. The advantage of using a LOF is identifying points that are outliers relative to a local cluster of points. This algorithm is based on a concept of local density, where locality is given by k nearest neighbors, whose distance is used to estimate the density.

Data Description

The dataset that we will use for this project consists of 15 numerical columns with 140000 rows which are a result of PCA transformation. These numerical values are nothing but masked credit card transactions. We do not have any background information on the features due to confidentiality reasons.

Aim

To build a model that is able to correctly identify fraudulent credit card transactions from the valid transactions using Isolation Forest and Local Outlier Factor.

Tech Stack

- Language - Python
- Libraries - sklearn, pandas, matplotlib, numpy, seaborn

Approach

1. Importing the required libraries and packages
2. Open the config.ini file. (This is a configuration file that can be edited according to your dataset)
3. Read the dataset (audio files)
4. Perform exploratory data analysis
5. Handle Missing Values
6. Find contamination amount
7. Model Training
8. Making predictions

Modular code overview

```
input
|_config.ini
|_ credit_card_transactional_data.csv

src
|_engine.py
|_ml_pipeline
        |_utils.py
        |_processing.py
        |_model.py

lib
|_ Fraudulent Transaction using Isolation Forest.ipynb
|_ Local Outlier Factor for Anomaly Detection.ipynb

output
|_IF_model.pkl
|_LOF_model.pkl
```

Once you unzip the modular_code.zip file you can find the following folders within it.

1. input

2. src

3. output

4. lib

1. input folder **-** It contains all the data that we will need for analysis.
    - A config file, with some basic configuration parameters which can be edited according to your dataset.
    - A credit_card_transactional_data.csv file, that has 140000 processed and masked credit card transactional data with 15 features.

2. src folder - This is the most important folder of the project. This folder contains all the modularized code for all the above steps in a modularized manner. This folder consists of:
    - engine.py
    - ml_pipeline

    The ml_pipeline is a folder that contains all the functions put into different python files which are appropriately named. These python functions are then called inside the engine.py file.

3. output folder **–** The output folder contains the model that we trained for this data. This model can be easily loaded and used for future use and the user need not have to train all the models from the beginning.

4. lib folder **-** This is a reference folder. It contains the original ipython notebook that we saw in the videos.

Project takeaways

1. Understanding problem statement
2. Understanding anomaly detection
3. Understanding fraudulent transaction
4. Understanding the nature of data
5. Learning the approach of choosing the ideal algorithm
6. Learning Isolation Forest Algorithm
7. Learning missing value imputation
8. Learning how to find correlations between features
9. Learning how to find contamination amount of Isolation Forest
10. Model training using isolation
11. Plotting countplots and boxplots
12. Using libraries like matplotlib, sklearn, seaborn, etc.

13. Using pandas and numpy libraries
14. Creating config files
15. Learning how to plot heatmaps
16. Understanding the functioning of Local Outlier Factor Algorithm
17. Implementing Local Outlier Factor Algorithm
18. Calculating anomaly scores