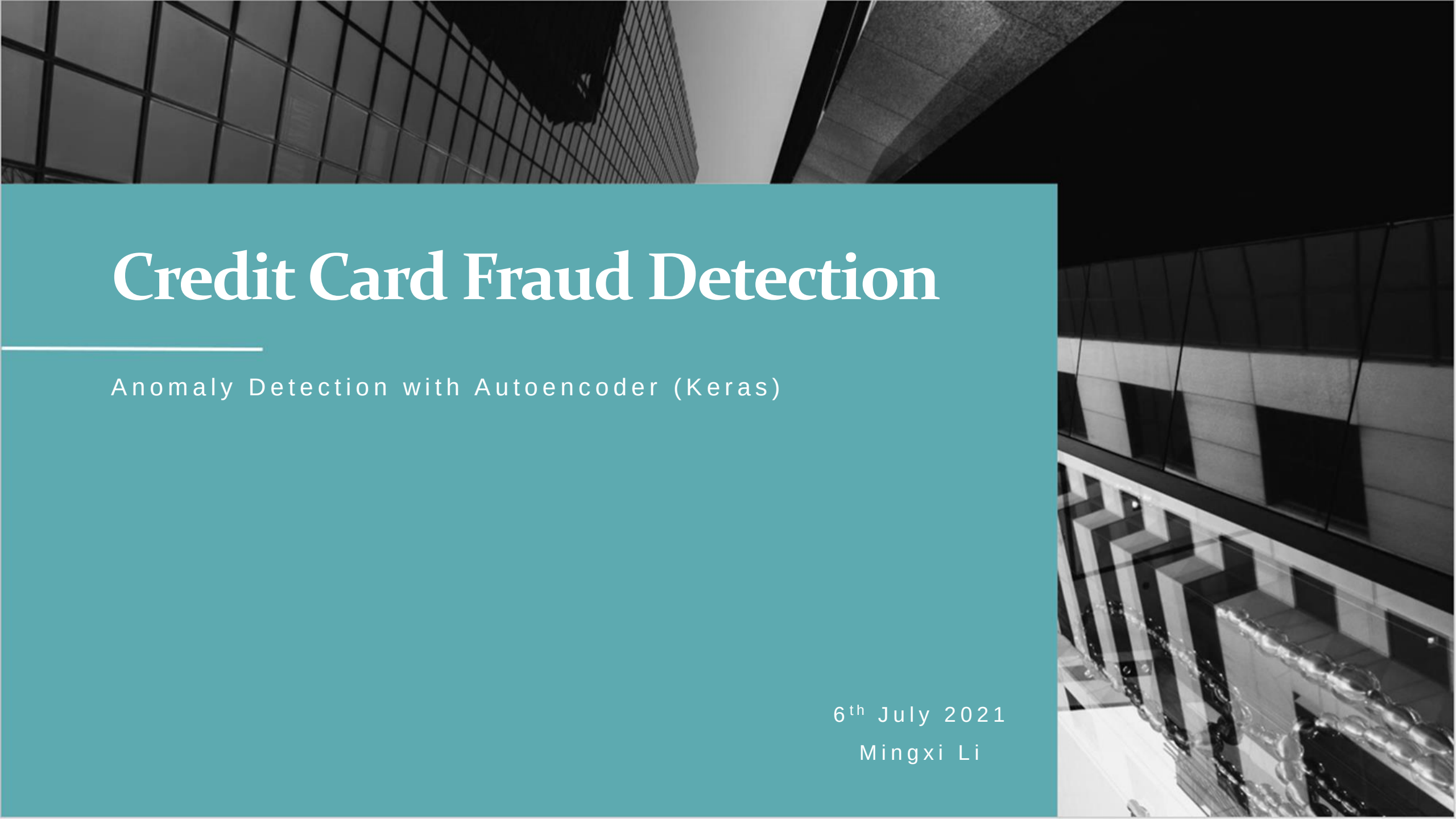


Credit Card Fraud Detection

Anomaly Detection with Autoencoder (Keras)

6th July 2021

Mingxi Li

Contents

Problem Description & Data Exploration

1

Target:

Using unsupervised anomaly detection technique to recognize fraudulent credit card transactions so that customers are not charged for item that they did not purchase.

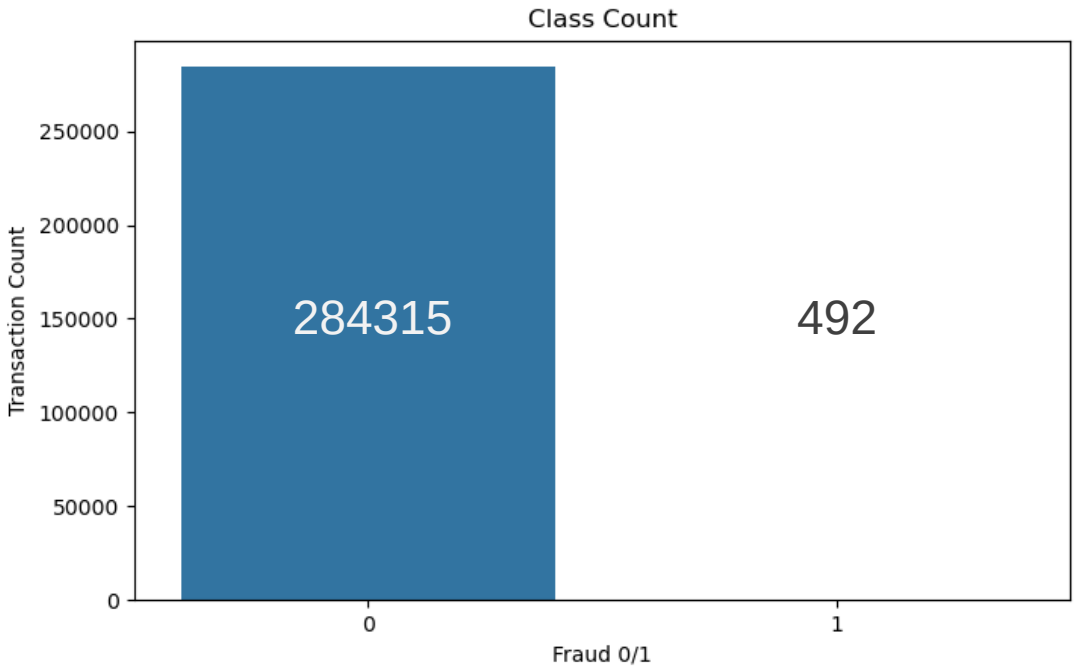
Dataset:

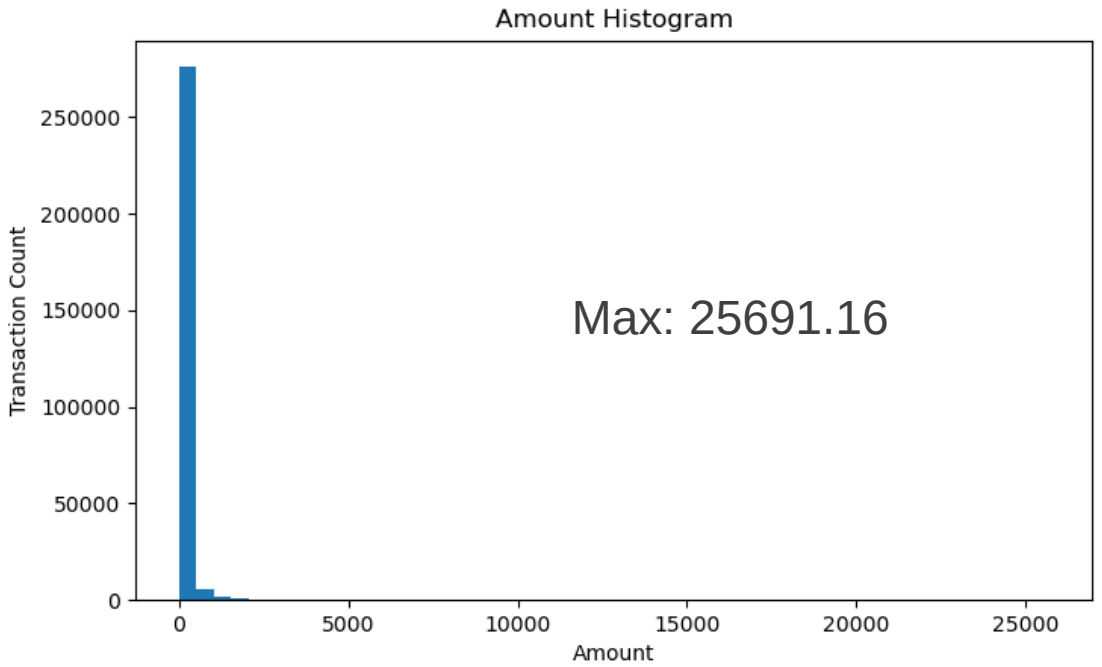
The dataset contains 2 days transection records made by credit cards in September 2013 by EU cardholders.

Time	V1	V2	V3	...	V26	V27	V28	Amount	Class
0	-1.35981	-0.07278	2.536347	...	-0.18911	0.133558	-0.02105	149.62	0
0	1.191857	0.266151	0.16648	...	0.125895	-0.00898	0.014724	2.69	0
1	-1.35835	-1.34016	1.773209	...	-0.1391	-0.05535	-0.05975	378.66	0
1	-0.96627	-0.18523	1.792993	...	-0.22193	0.062723	0.061458	123.5	0
2	-1.15823	0.877737	1.548718	...	0.502292	0.219422	0.215153	69.99	0
2	-0.42597	0.960523	1.141109	...	0.105915	0.253844	0.08108	3.67	0
4	1.229658	0.141004	0.045371	...	-0.25724	0.034507	0.005168	4.99	0
7	-0.64427	1.417964	1.07438	...	-0.05163	-1.20692	-1.08534	40.8	0
7	-0.89429	0.286157	-0.11319	...	-0.38416	0.011747	0.142404	93.2	0
9	-0.33826	1.119593	1.044367	...	0.094199	0.246219	0.083076	3.68	0
10	1.449044	-1.17634	0.91386	...	-0.12948	0.04285	0.016253	7.8	0
10	0.384978	0.616109	-0.8743	...	-0.49221	0.042472	-0.05434	9.99	0
10	1.249999	-1.22164	0.38393	...	-0.35499	0.026416	0.042422	121.5	0

- Besides ‘Time’, ‘Amount’ and ‘Class’, the dataset only contains numerical input variables which are the result of a PCA transformation (V1 ~ V28).
- ‘Time’ is the seconds elapsed between each transaction and the first transaction in the dataset, so the time range is [0, 172792].
- ‘Class’ is the response variable and it takes value 1 in case of fraud and 0 otherwise.

Data Exploration (1)



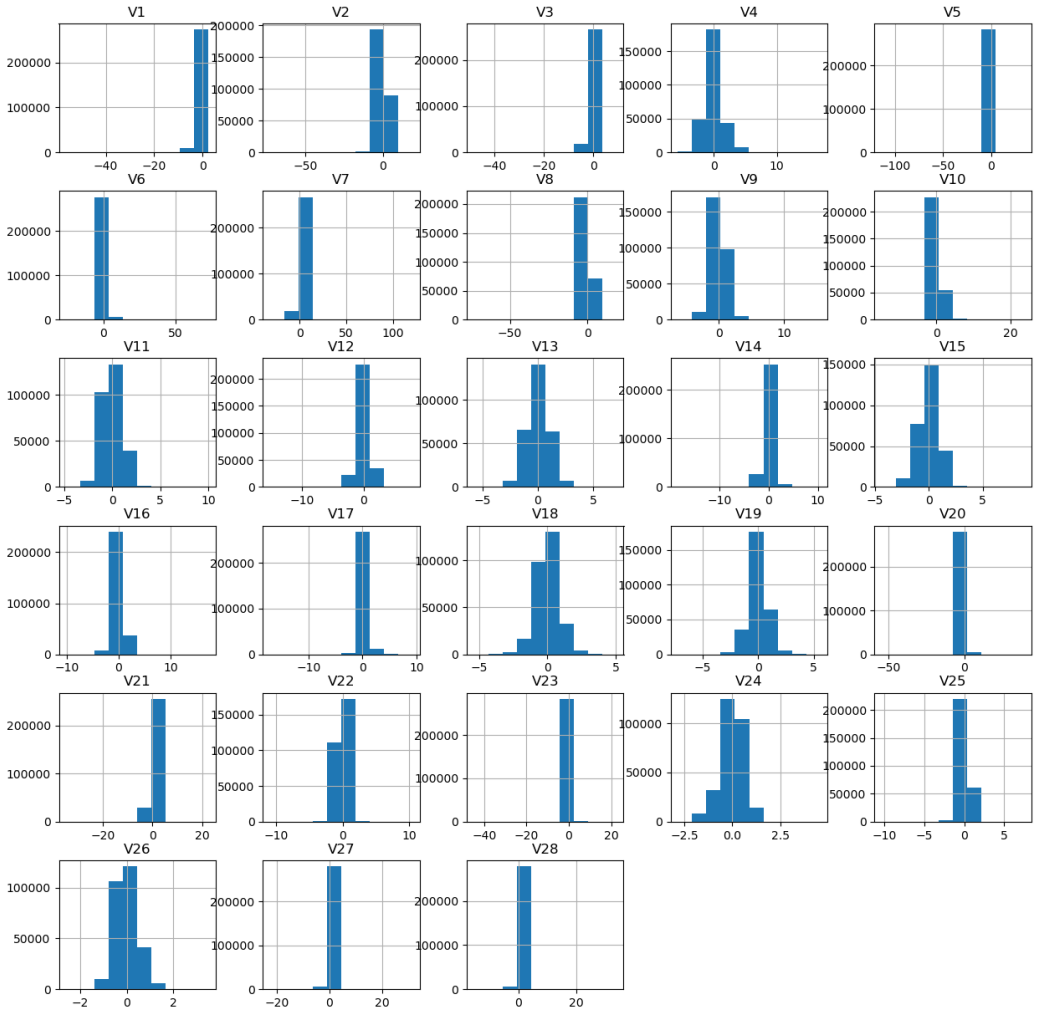


Unbalanced class (0.17% fraud):

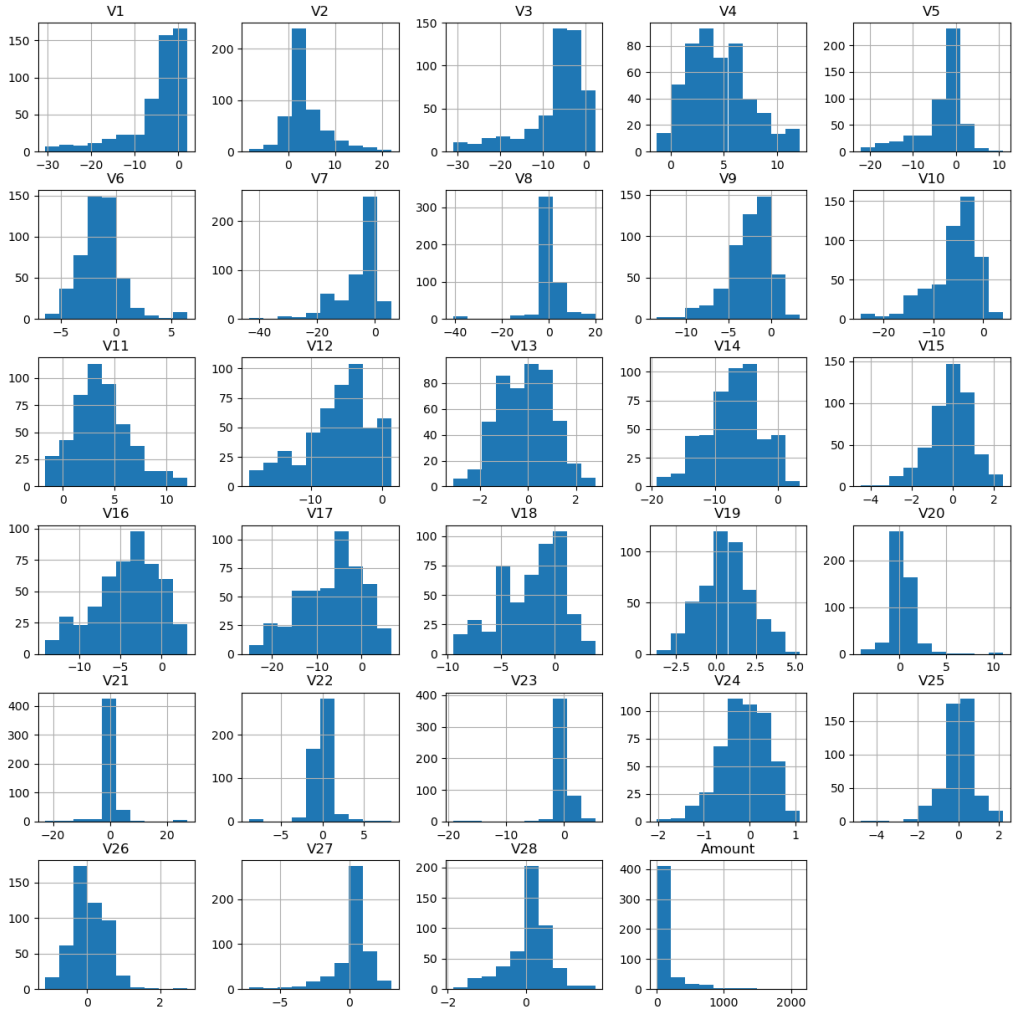
- StratifiedShuffleSplit() for train & test dataset split;
- SMOTE combine with RandomUnderSampler() to increase class 1 ratio in training dataset (***if necessary***).

Transaction amount range is way larger than other PCA features, apply MaxMinNorm to normalize it in case the activation function get twisted.

Data Exploration (2)



Normal records



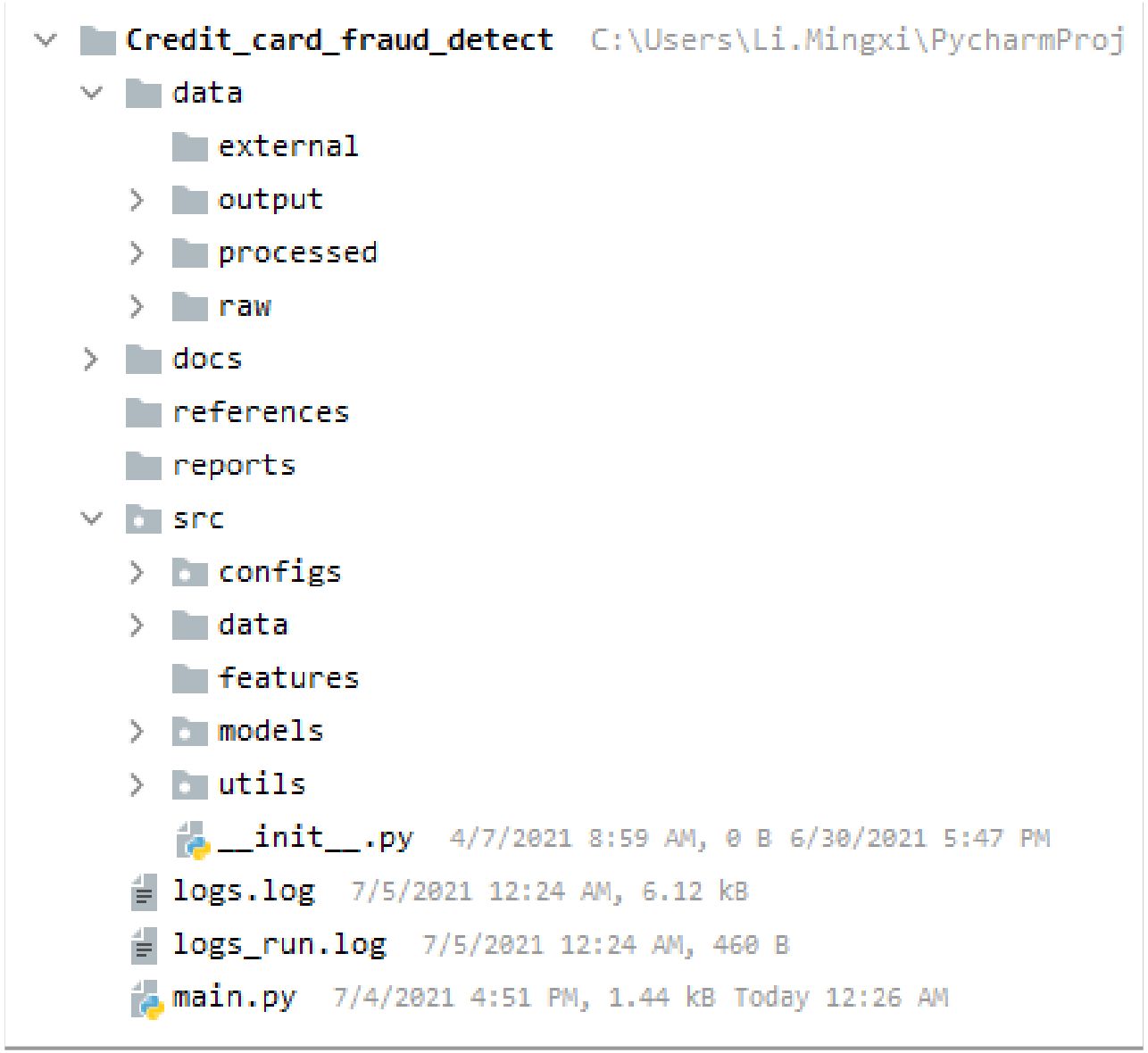
Fraud records

Project Structure

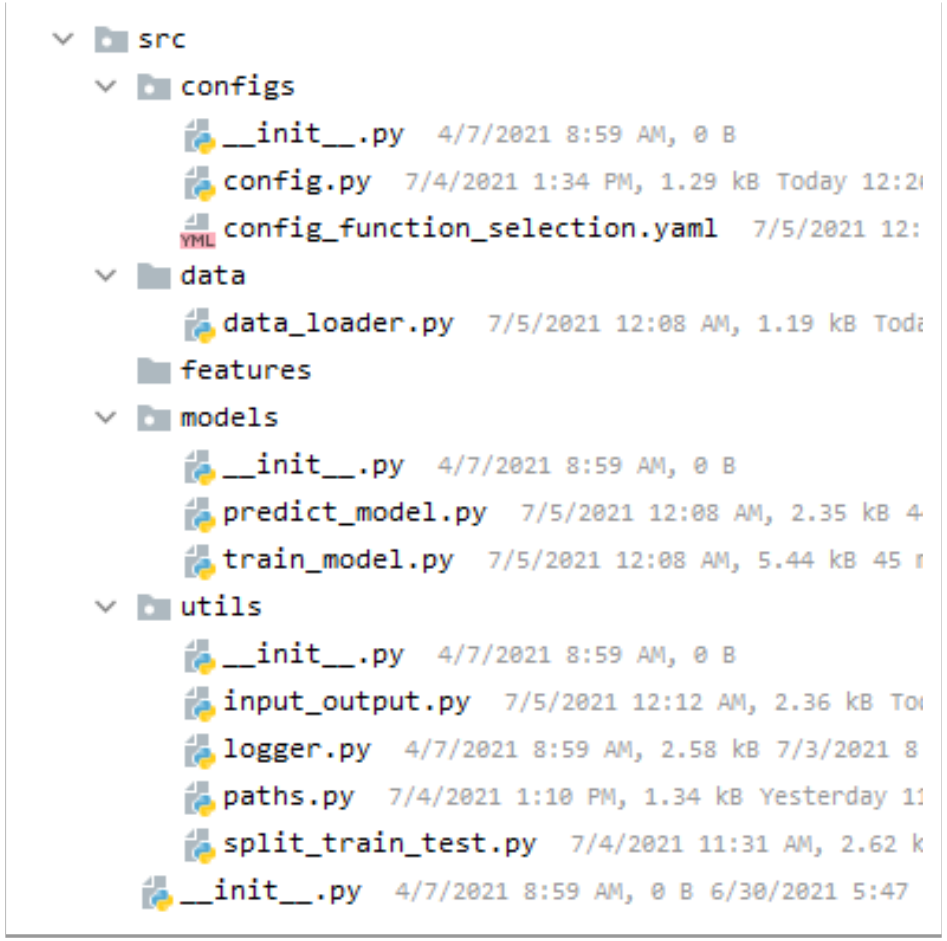
2

A logical and standardized project structure for doing and sharing project.

- **Data** folder contains all the data inputs and outputs:
  - Raw stores all the input dataset, which will be loaded by dataloader.py later.
  - Processed stores all the data & models generated in the middle, incase we want to test code from a random point.
  - Output contains the model results: data, plot, etc.
- **Docs** contains yaml files for feature selection, the team can easily add or remove model features.
- **References** is the folder for model knowledge.
- **Reports** can store analysis results.
- **SRC** is source code for use in this project: see next page
- Main.py : Run the model!



A logical and standardized project structure for doing and sharing project.



SRC is source code for use in this project:

- Configs: all the adjustable model configures are stored in yaml file and can be changed anytime by anyone. Modeling steps can also be selected and unselected.
- Data: contains scripts to turn raw data into features for modelling.
- Features: feature engineering is done here.
- Models: model training and testing.
- Utils: stores all the functional code for modeling. Here are data input loading & output saving format/ path. Logger will generate informations when you running code, to inform you which step the model at.

Training Data Prep

3

1. Data normalization

Only normalize ‘Amount’ column, then the data set now contains V1, V2, …, V28, Amount_norm (29 columns).

2. Splitting Train & Test (20%)

StratifiedShuffleSplit() returns stratified randomized folds. The folds are made by preserving the percentage of samples for each class.

Count	y_train	y_test
0	227451	56864
1	394	98

3. SMOTE + RandomUnderSampler

- SMOTE generates the synthetic samples for the minority class, which helps to overcome the overfitting problem posed by random oversampling.
- SMOTE normally combined with under sampling function to balance the class distribution.

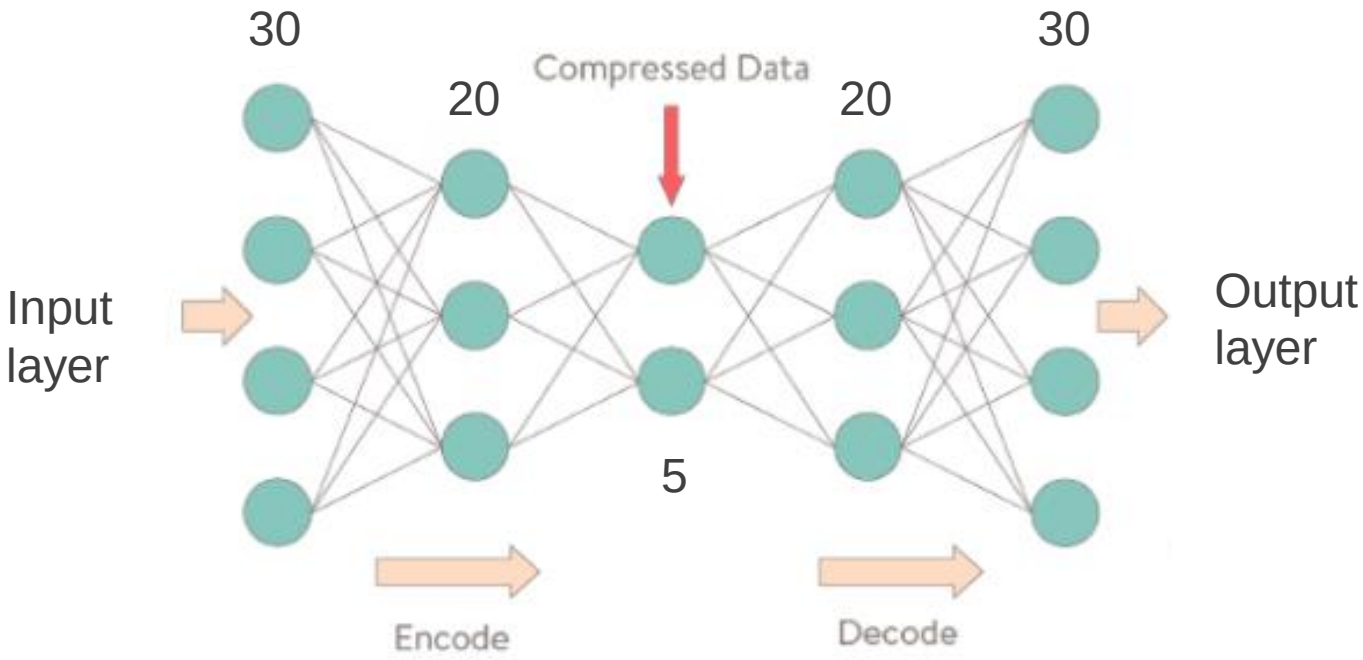
Count	y_train	y_smote
0	227451	64985
1	394	45490

Model Building

4

Autoencoder Model - Build

```
Model: "model_1"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
input_2 (InputLayer)         [(None, 29)]              0
_________________________________________________________________
dropout_2 (Dropout)          (None, 29)                0
_________________________________________________________________
dense_6 (Dense)              (None, 30)                900
_________________________________________________________________
dense_7 (Dense)              (None, 20)                620
_________________________________________________________________
dense_8 (Dense)              (None, 5)                 105
_________________________________________________________________
dense_9 (Dense)              (None, 20)                120
_________________________________________________________________
dense_10 (Dense)             (None, 30)                630
_________________________________________________________________
dropout_3 (Dropout)          (None, 30)                0
_________________________________________________________________
dense_11 (Dense)             (None, 29)                899
=================================================================
Total params: 3,274
Trainable params: 3,274
Non-trainable params: 0
_________________________________________________________________
```



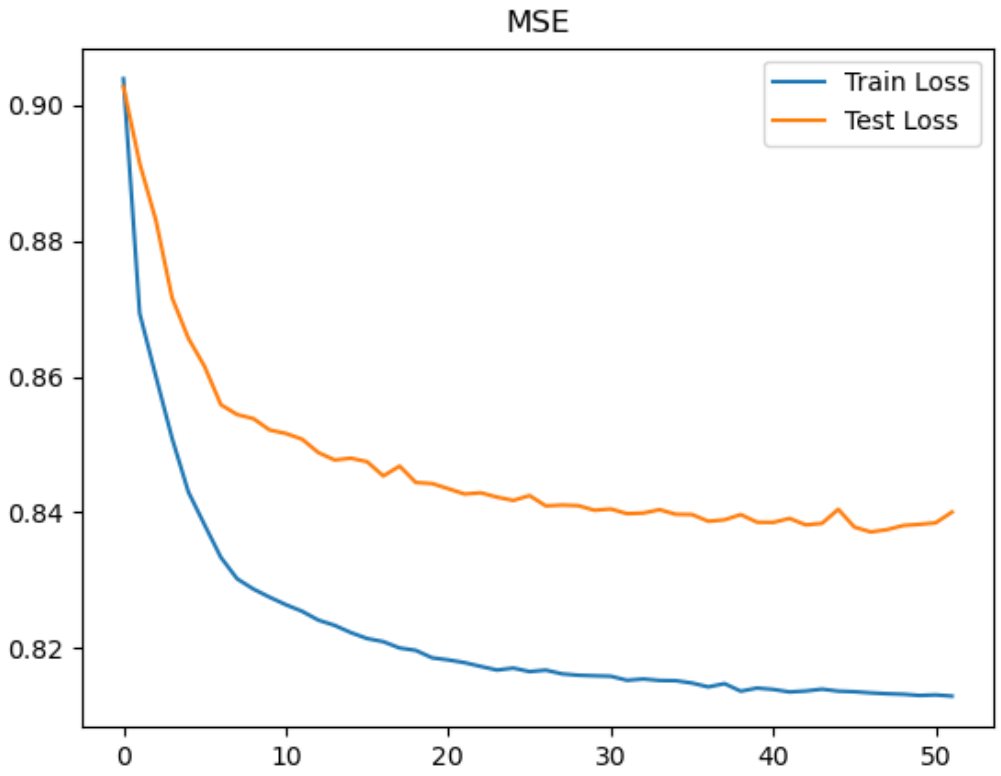
- Hidden layers activation function: ReLu
- Output layer's activation function: sigmoid(x) = 1 / (1 + exp(-x))
- Dropout layer: 0.1 (avoid overfitting)

Autoencoder Model – Train

```
autoencoder.compile(optimizer='adam', loss='mse')
```

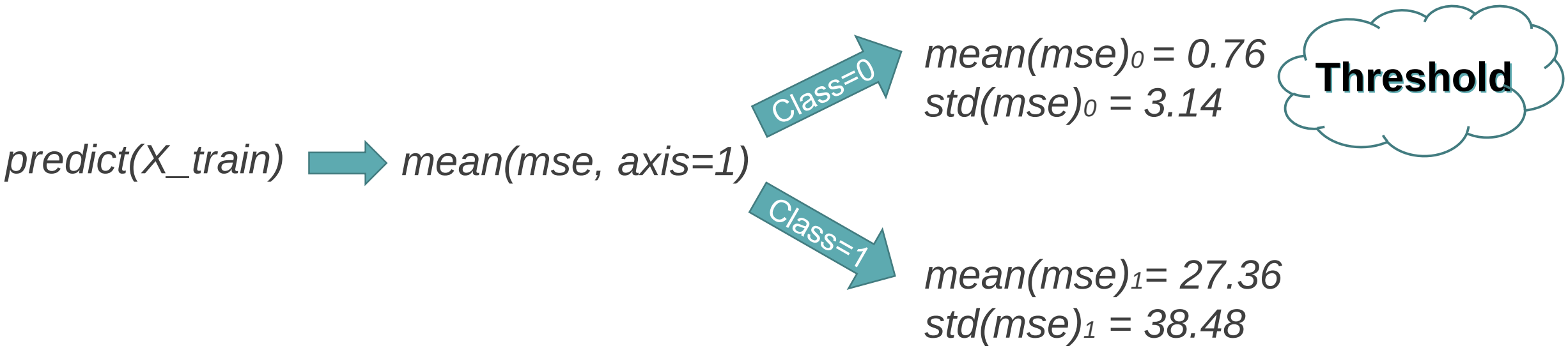
- Reconstruction error (loss): MSE
- Optimizer: Adam

```
autoencoder.fit(x=X_train,
                y=X_train,
                epochs=200,
                batch_size=1024,
                verbose=2,
                validation_data=(X_test, X_test),
                shuffle=True,
                callbacks=[early_stop, checkpointer])
```



This loss plot is without SMOTE oversampling.

Autoencoder Model – Threshold

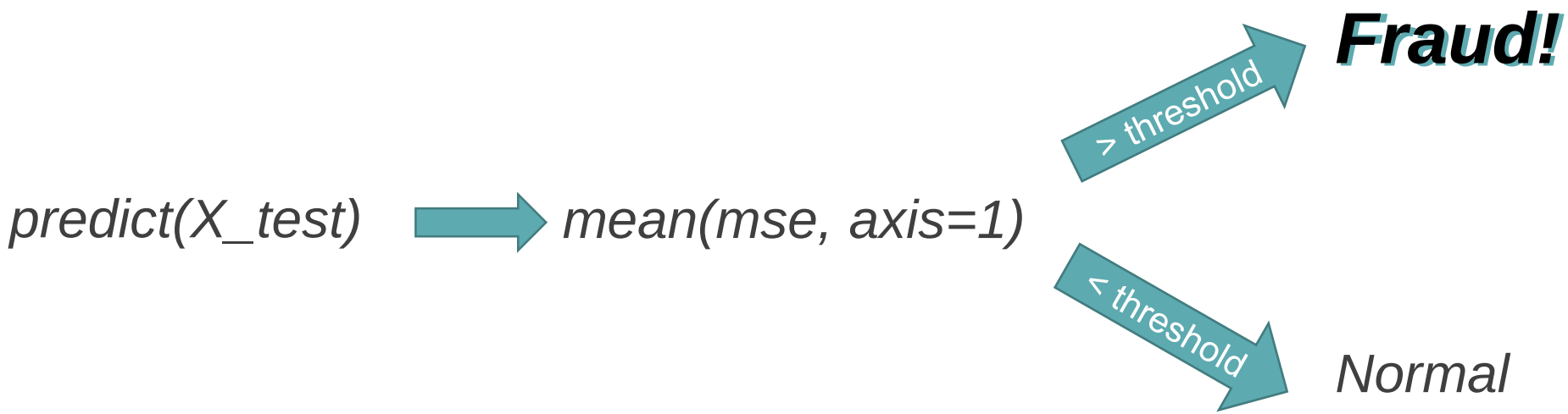


Trained model predicting on train data, then calculate mean MSE by each observation (row), grouping observations by 0/1 class then calculate mean MSE for each class, the mean MSE of class 0 with standard deviation can be the threshold for predicted dataset.

Anomaly Detection

5

Autoencoder Model – Threshold = 3.9



Test \ Predict	Normal	Fraud
Normal	55740	1124
Fraud	25	73

Discussion

Appendix

- ReLU (Rectified Linear Unit) function: *f(x) = max(0, x)*
- Sigmoid function: *s(x) = 1/ (1+e^(-x))*
- Adam optimization: a *stochastic gradient descent method* that is based on adaptive estimation of first-order and second-order moments.