Credit Card Fraud Detection

Springboard DSC Capstone Project I

Presentation Deck
Reuben Yang
2017.10

Introduction

- Business Problem: Identifying the fraudulent transactions from a set of credit card transactions
- Clients: Financial institutions
- Dataset: Credit card transactions in two days in Sep 2013 by European cardholders (from Kaggle)

Introduction (cont.)

- Summary of dataset:

Total Transactions	Legal	Fraudulent	Fraud Ratio	Number of Features
284,807	284,315	492	0.17%	28

- The dataset is highly imbalanced
- Coded features V1 to V28 are all numerical and obtained after PCA transformation
- No missing values in the dataset

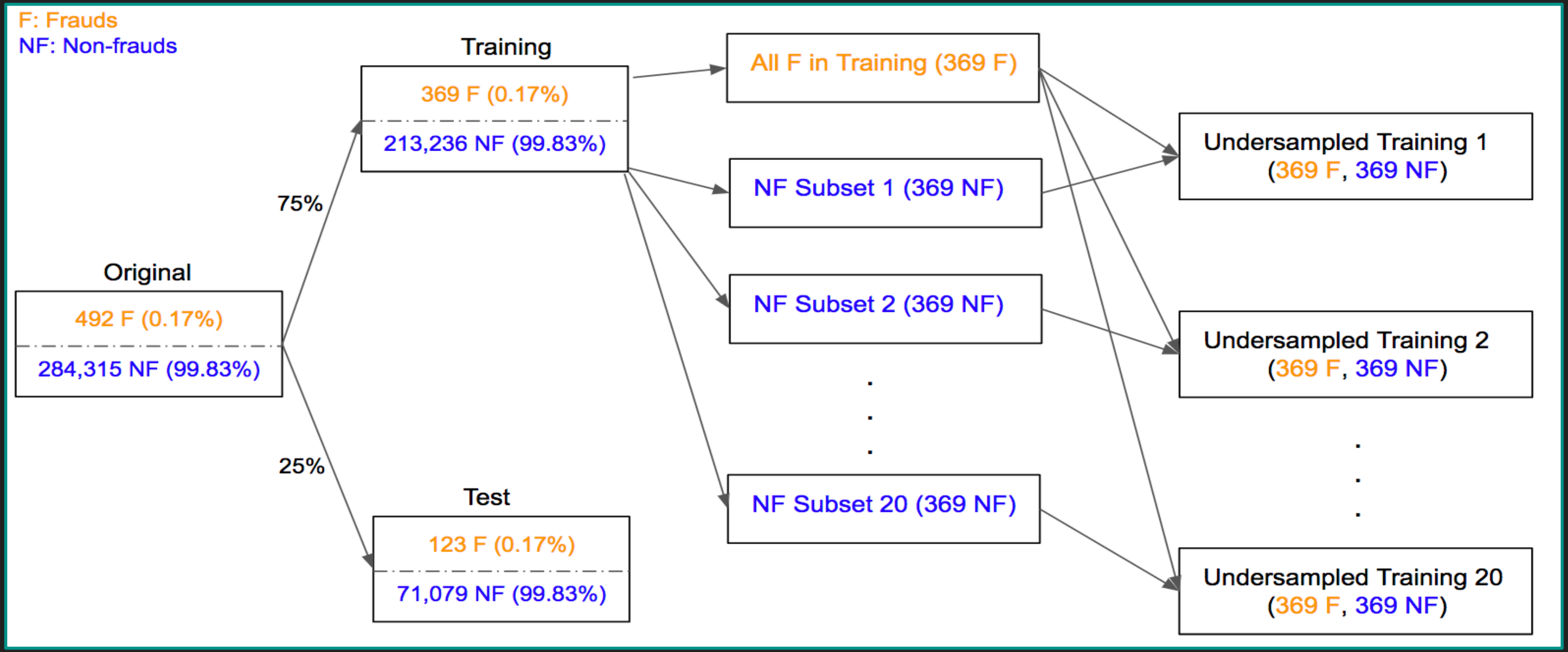
Methodology

- Baseline: train the models directly on the highly skewed data without resampling
- Undersampling: randomly select a fraction of the non-frauds (the number of frauds) and pair them up with all frauds to form a balanced dataset
- Oversampling: generate new samples of frauds to make the two classes balanced
- A 75%-25% training-test split of the original dataset with stratification is used in all 3 approaches

	Training	Test
Fraud	369 (0.17%)	123 (0.17%)
Non-fraud	213,236 (99.83%)	71,079 (99.83%)

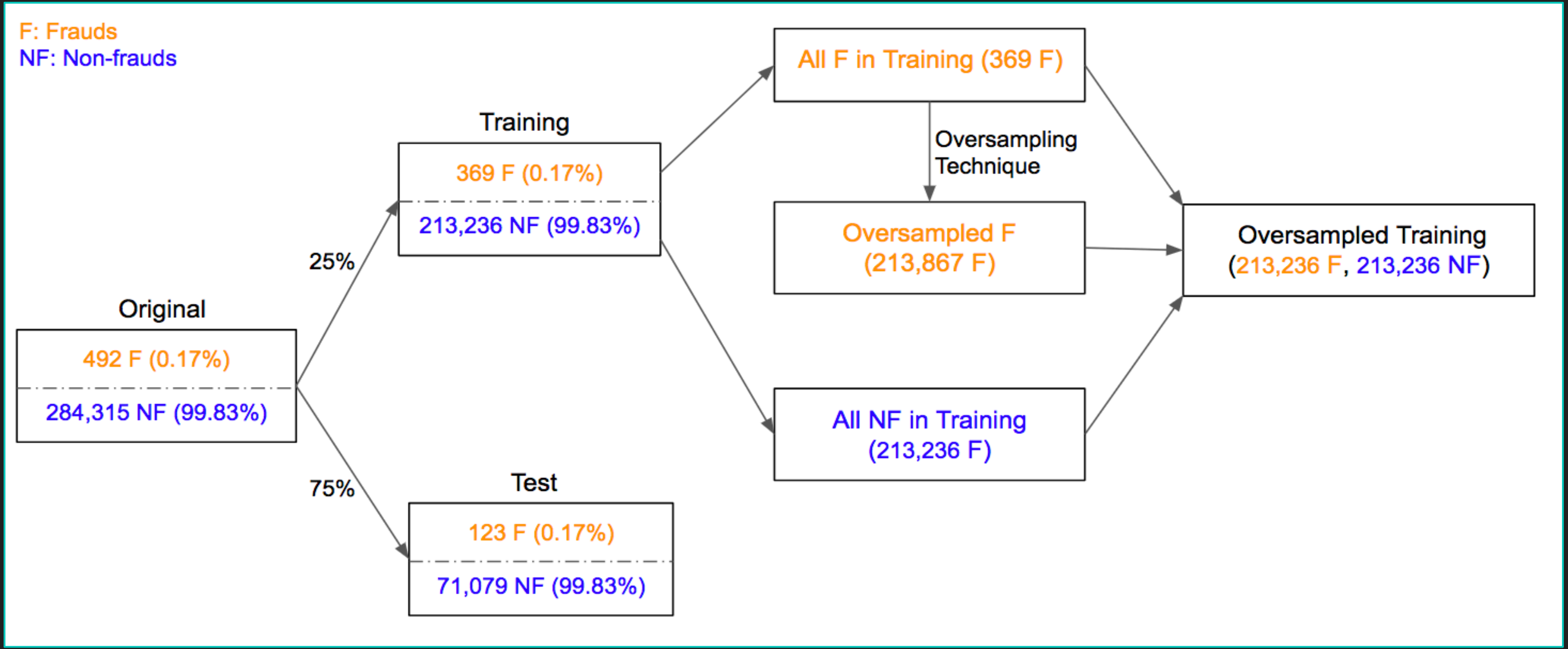
Methodology (cont.)

Illustration of undersampling



Methodology (cont.)

Illustration of oversampling

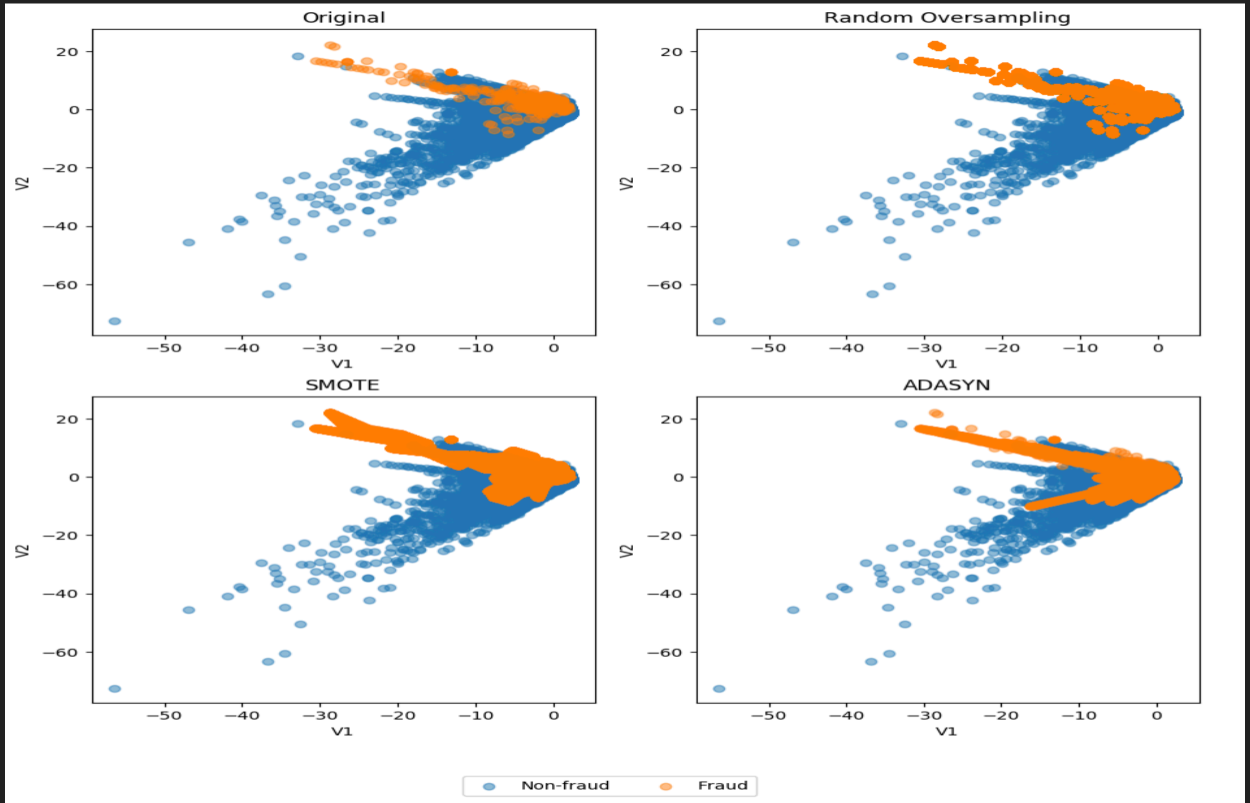


Methodology (cont.)

- Three different oversampling techniques are applied
- Random Oversampling (RO): randomly sample, with replacement, the current available samples of frauds
- Synthetic Minority Oversampling Technique (SMOTE): creates synthetic examples which lie on the line segment joining a sample of fraud in the original dataset and one of its k nearest neighbors
- Adaptive Synthetic (ADASYN): like SMOTE, ADASYN also creates synthetic samples but it focuses on generating samples next to the original samples which are wrongly classified using a k-Nearest Neighbors classifier (called "difficult to learn")

Methodology (cont.)

Illustration of oversampling based on variable V1 and V2



Methodology (cont.)

- Algorithms: logistic regression and random forest
- Logistic regression is trained for all three approaches. L2 norm regularization is used and 0.01, 0.1, 1, 10, and 100 are the candidates.
- Random forest is trained for the baseline and undersampling approach. Integers in [40, 50] are considered as the number of trees and “sqrt” is assigned to max_features.
- For the undersampling approach, 20 logistic regressions and random forests are trained, respectively. The final prediction for each test sample is based on the majority vote.

Performance Metrics

- Recall: fraction of fraud transactions that are successfully detected, i.e. number of correctly predicted frauds divided by the total number of actual frauds
- Precision: fraction of predicted fraud transactions that are accurate, i.e. number of correctly predicted frauds divided by the total number of predicted frauds
- This project aims at achieving the highest recall and uses precision as auxiliary

Results

Recall for Logistic Regressions:

	Baseline		Undersampling		Random Oversampling		SMOTE		ADASYN	
Case	Training	Test	Training	Test	Training	Test	Training	Test	Training	Test
1	0.645	0.642	0.930	0.902	0.924	0.902	0.920	0.902	0.716	0.943
2	0.645	0.618	0.932	0.886	0.922	0.878	0.923	0.869	0.732	0.943
3	0.623	0.650	0.942	0.862	0.938	0.862	0.938	0.862	0.731	0.894
4	0.604	0.642	0.924	0.919	0.922	0.911	0.918	0.919	0.720	0.975
5	0.615	0.602	0.924	0.911	0.919	0.894	0.918	0.902	0.717	0.959
Avg.	0.627	0.627	0.930	0.896	0.925	0.889	0.924	0.891	0.723	0.943

Results (cont.)

Recall for Logistic Regressions:

	Baseline		Undersampling		Random Oversampling		SMOTE		ADASYN	
Case	Training	Test	Training	Test	Training	Test	Training	Test	Training	Test
1	0.898	0.859	0.973	0.045	0.976	0.064	0.976	0.059	0.839	0.012
2	0.875	0.894	0.973	0.046	0.977	0.068	0.975	0.063	0.854	0.013
3	0.871	0.870	0.967	0.036	0.976	0.058	0.974	0.055	0.849	0.012
4	0.861	0.868	0.967	0.038	0.975	0.060	0.973	0.056	0.826	0.011
5	0.866	0.851	0.969	0.038	0.975	0.060	0.973	0.056	0.847	0.013
Avg.	0.874	0.869	0.970	0.041	0.976	0.062	0.974	0.058	0.843	0.012

Results (cont.)

Recall and Precision for Random Forest[1]:

	Recall				Precision			
	Baseline		Undersampling		Baseline		Undersampling	
Case	OOB	Test	OOB	Test	OOB	Test	OOB	Test
1	0.800	0.732	0.907	0.886	0.955	0.928	0.966	0.060
2	0.794	0.732	0.913	0.886	0.948	0.928	0.966	0.050
3	0.789	0.724	0.919	0.862	0.957	0.947	0.966	0.046
4	0.786	0.813	0.908	0.886	0.954	0.901	0.965	0.053
5	0.770	0.789	0.907	0.886	0.944	0.942	0.965	0.052
Avg.	0.788	0.758	0.911	0.881	0.952	0.929	0.966	0.052

[1]The OOB performance is used as a measure for training performance

Results (cont.)

- For logistic regression:
  - The recall is materially enhanced using the resampling techniques as opposed to the baseline approach
  - ADYSYN has the highest recall for test data although its training performance suffers due to the hard cases it created during oversampling

- For random forest:
  - Better performance under baseline than logistic regression
  - Using undersampling, it has an overall performance that is pretty identical to logistic regression

- It is expected that the precision for models with resampling would drop tremendously for the test sets given the serious imbalance of the test data

Conclusion

- Both logistic regression and random forest with resampling techniques are effective for fraud detection in the exceedingly imbalanced data and have recall scores superior to the baseline approach

- The models with resampling approaches clearly have room for improvement in their precision scores as the falsely predicted frauds may cause inconvenience for credit card holders

Recommendations for Client

- Financial institutions should consider using the undersampling and oversampling approaches when developing the machine learning models for fraud detection

- Based on the experiments, logistic regression with ADYSYN is especially suggested as it has the best test performance for recall

Future Work

- Reduce the false positives: use F-score as performance metrics in training or build a second-stage model on the initially predicted frauds to improve precision
- For undersampling, assign different weights to each individual classifier based on its training performance when assembling the predictions
- Combination of oversampling and undersampling, e.g. SMOTETomek and SMOTEENN
- Perform resampling on the original dataset first and then split the new dataset to training and test