Privacy Preserving Credit Card Fraud Detection Using Synthetic Datasets

Rikako Hatoya
Dept. of Electrical and Computer Engineering
University of California, Los Angeles
Los Angeles, USA
rikakohatoya@ucla.edu

***Abstract*—Machine learning can predict and prevent fraudulent credit card transactions. However, limited amount of data is available to the public due to them being private information vulnerable to adversarial attacks, which limits the improvement of fraud detection models. Furthermore, this limits the accuracy of currently existing detection models due to under-sampling of fraudulent data. In this project, synthetic data sets are generated to compensate for the class imbalance issue caused by this limited availability of data.**

I. Introduction

Accurate detection of fraudulent credit card transactions are essential for financial institutions. Customers become unsatisfied when their fraud detection alerts are false positives, while at the same time, they also face potential risks as millions of dollars are lost each year due to fraud. However, institutions are prevented from sharing transaction data to the public since they are private information that is susceptible to risks such as model inversion, membership inference, and property inference. Furthermore, there are not enough samples to generalize the ML model since positive cases are rare, which leads to poor classification of performance. There are currently several ways to resolve this class imbalance issue including under-sampling, over-sampling, synthetic data generation, cost sensitive learning, ensemble methods, etc. In this project, synthetic data is generated to investigate if they can provide a more secure approach to the development of fraud detection models. In particular, several under- and over-sampling methods along with GANs (Generative Adversarial Networks) to augment synthetic data set are generated and investigated if they are realistic enough to increase the accuracy in predictions.

II. Dataset

The dataset used in this project comes from a previously hosted Kaggle competition, "Credit Card Fraud Detection." The dataset contains transactions made by credit cards in September 2013 by European cardholders in 2 days. There are 284,807 transactions in total with 492 of them being fraud. The dataset consists of "Time" –time of transaction, "Amount" –amount transacted, "Class" –"0" for legal and "1" for fraudulent transaction, and "V1-V28", which are principal components that have been PCA transformed to preserve the confidentiality of data.

III. Methodology

A. Exploratory Data Analysis (EDA)

The distribution of non-fraud and fraud data is as follows. There are 284315 non-fraud cases and 492 fraud cases.

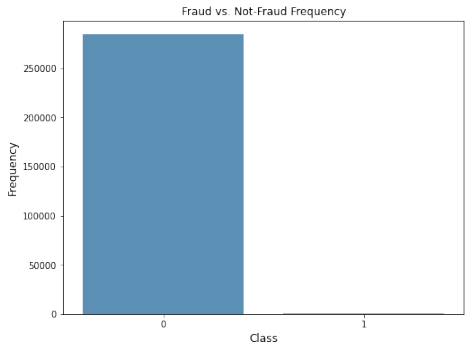


Fig. 1. Distribution of Fraudulent and Non-Fraudulent Data.

The distribution of transactions over time [s] are relatively consistent, although 2 peaks may be identified in the fraudulent case.

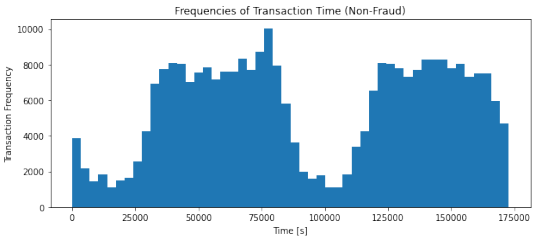


Fig. 2. Distribution of Transactions Over Time.

The frequencies of transaction amounts are as expected with a decay graph on a logarithmic scale and not much difference is observed between the 2 classes.

The feature correlation heatmap indicates that there may be some correspondence between "Time", "Class", and "Amount". However, not many details can be inferred from this map as data is still unprocessed.

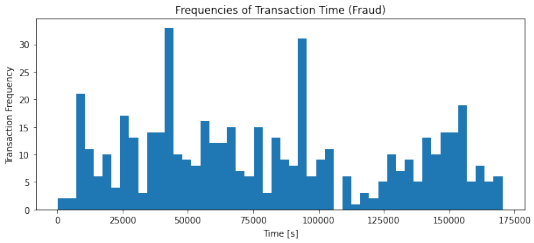


Fig. 3. Distribution of Transactions Over Time.

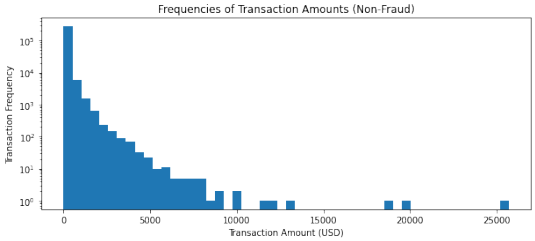


Fig. 4. Distribution of Transaction Over Amounts.

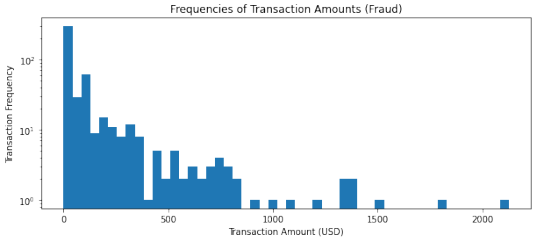


Fig. 5. Distribution of Transaction Over Amounts.

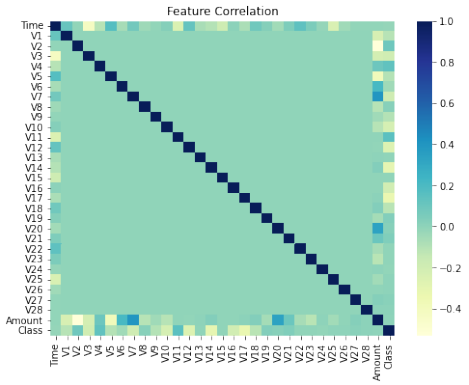


Fig. 6. Feature Correlation Heatmap.

B. Data Pre-Processing

While "V1-V28" are PCA transformed, the values in "Amount" are raw data and therefore they need to be scaled. A standard scaler is used to transform the values in the "Amount" column. Then, data is split into train and test data with a test size of 30% (199008 samples=Non-Fraud, 356 samples=Fraud.)

C. Synthetic Data Generation: Re-sampling Methods

To resolve the class imbalance issue, re-sampling methods were first considered. For over-sampling, 1.SMOTE, 2.SVM SMOTE, 3.Borderline SMOTE, 4.ADASYN, 5.RandomOver-Sampler and for under-sampling, 1. RandomUnderSampler, 2.NearMiss were investigated. The number of samples obtained for each method in the end were as follows.

	SMOTE	SVM	BSM	ADASYN	ROS	RUS	NM
Fraud	199008	199008	199008	198935	199008	356	356
Non Fraud	199008	199008	199008	199008	199008	356	356

Furthermore, the correlation heatmaps for each of the sampling methods are as follows.

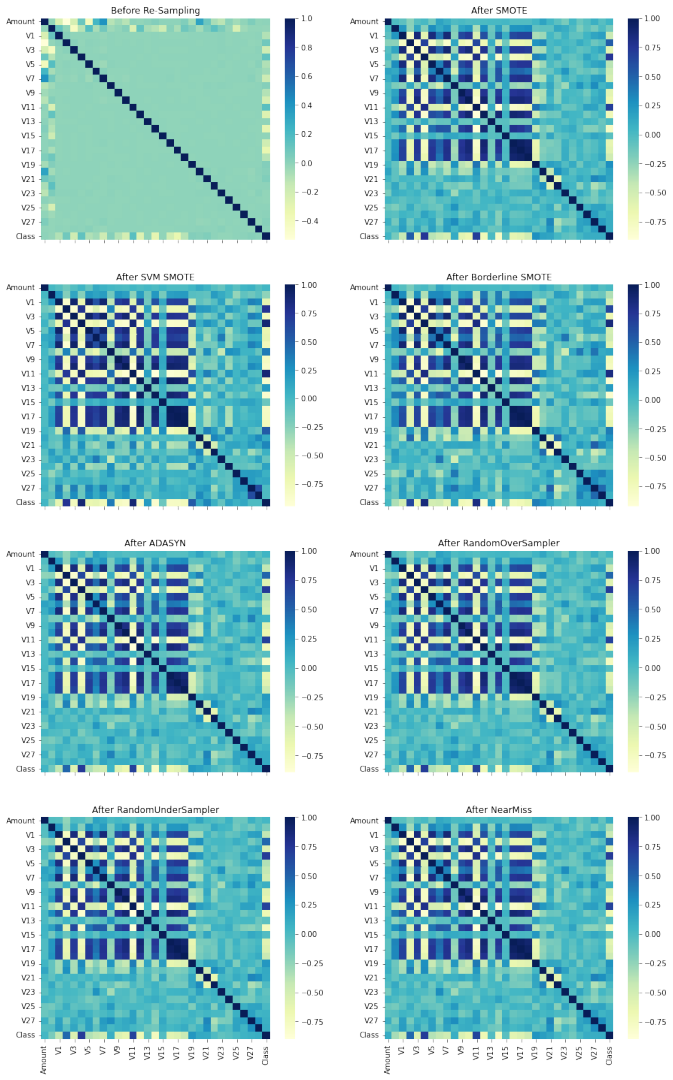


Fig. 7. Feature Correlation Heatmap.

One may observe that features V1-V18 have the most strongest correlations. The strongest features may be removed at this point, however, all features are selected for analysis.

D. Synthetic Data Generation: GANs

Furthermore, GANs are used to generate synthetic data and resolve the class imbalance issue. GANs creates realistic datasets by propagating between its generator and discriminator, which competes with each other in a zero sum game. The following methods were adopted: 1. GAN (Vanilla GAN), 2.WGAN, 3.CramerGAN. Since the original training data is a huge matrix that contains nearly 200,000 samples, the number of samples created by GANs are limited to 40,000 samples each to save on computing costs. The number of samples obtained for each method in the end were as follows.

	GAN	WGAN	CramerGAN
Fraud	411	303	419
Non Fraud	357	465	349

The correlation heatmap for GANs generated data is as follows. One may observe here that Vanilla GAN and CramerGAN was able to create a synthetic dataset with feature correlation similar to those seen in the re-sampling methods.

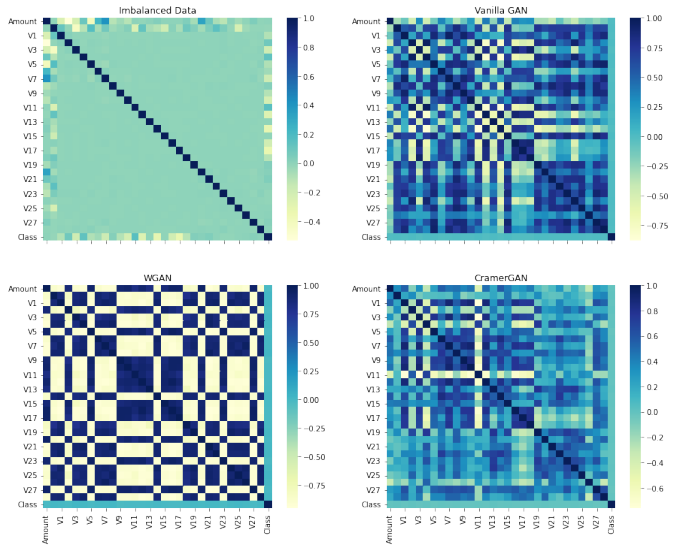


Fig. 8. Feature Correlation Heatmap.

E. Subset Sampling

In order to further save on computing costs and to also make the number of total samples consistent among different sampling methods, datasets created by the re-sampling methods are randomly sampled and reduced down to 40,000 samples to match the samples of GANs generated data. The number of samples for the original test data set are also randomly sampled and reduced to match the number of rows (51 samples=Fraud, 39949 samples=Non Fraud.)

IV. Evaluation

A. Metrics

In order to evaluate the performance of each of the synthetic datasets, the following parameters are observed: 1.ROC Curve, 2.Confusion Matrix, 3.Accuracy Score, 4.Recall Score, 5.Precision Score, 6.F-1 Score.

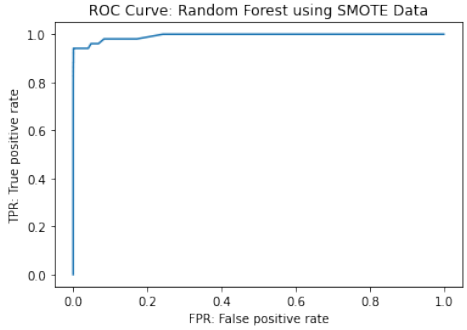
B. Random Forest

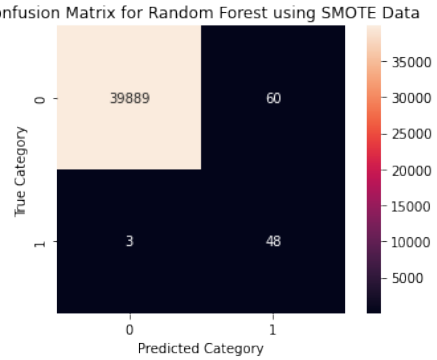
The metrics obtained by applying random forest classification on the datasets are as follows.

	SMOTE	SVM	BSM	ADASYN	ROS
Accuracy	0.998	0.999	0.999	0.999	0.999
Recall	0.970	0.931	0.921	0.970	0.961
Precision	0.722	0.849	0.871	0.773	0.873
F-1	0.801	0.886	0.894	0.845	0.912

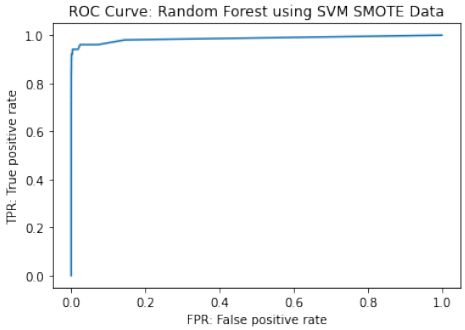
	RUS	GAN	WGAN	CramerGAN
Accuracy	0.963	0.394	0.002	0.636
Recall	0.481	0.481	0.500	0.563
Precision	0.5	0.500	0.501	0.500
F-1	0.490	0.283	0.002	0.390

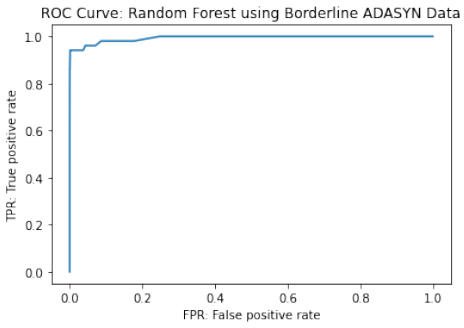
ROC Curve and Confusion Matrix for each dataset is as given in the following.

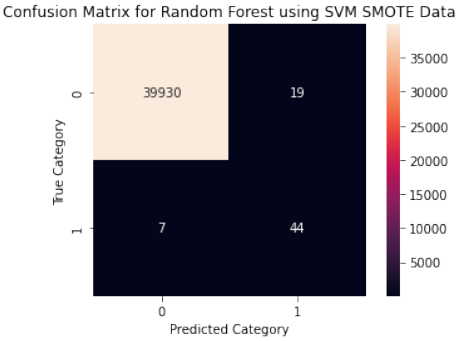


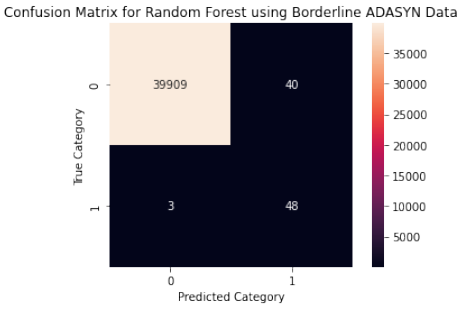


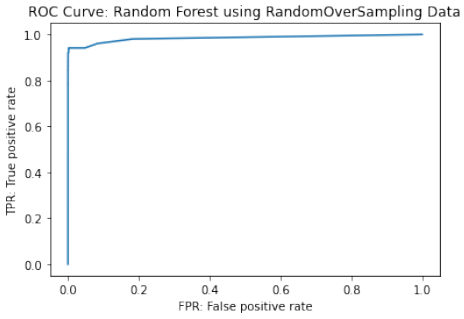
Under-sampling of data is crude with extremely small amount of fraudulent data since it is very difficult to obtain even a single fraudulent result in the test set. Therefore, under-sampling methods (RandomUnderSampling and NearMiss) will be ignored from this point. Furthermore, WGAN predictions are terrible and both Vanilla GAN and CramerGAN have a lot of false positives. The synthetic data models with the highest recall scores are: SMOTE, ADASYN, and RandomOverSampling. To achieve a better prediction model, ADASYn, which has the highest recall score is further experimented with hyper-parameter tuning based on

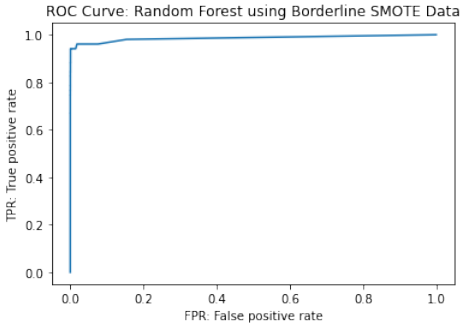

ROC Curve: Random Forest using SVM SMOTE Data
TPR: True positive rate
FPR: False positive rate

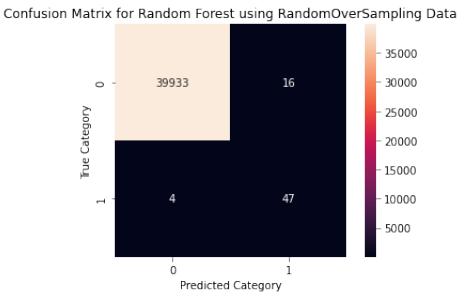

ROC Curve: Random Forest using Borderline ADASYN Data
TPR: True positive rate
FPR: False positive rate

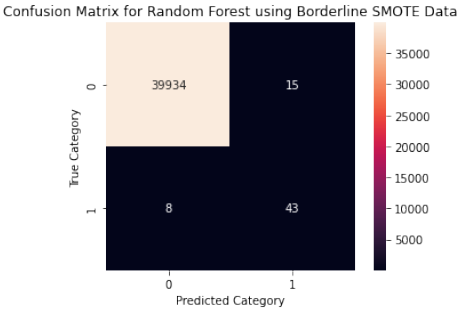

Confusion Matrix for Random Forest using SVM SMOTE Data
True Category
Predicted Category
39930
19
7
44

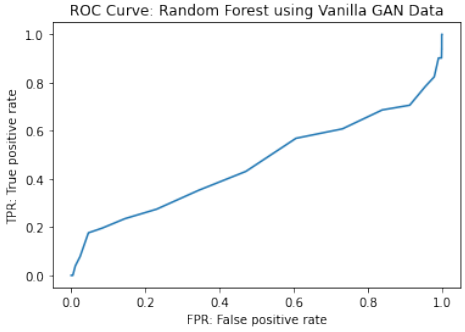

Confusion Matrix for Random Forest using Borderline ADASYN Data
True Category
Predicted Category
39909
40
3
48

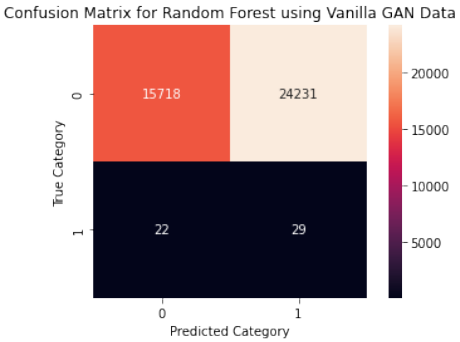

ROC Curve: Random Forest using RandomOverSampling Data
TPR: True positive rate
FPR: False positive rate

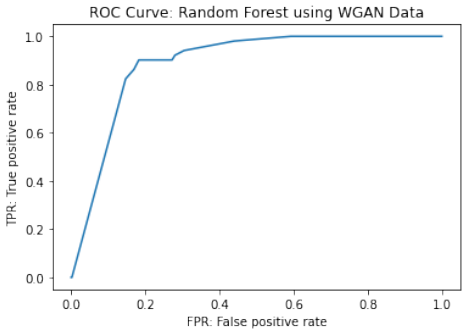

ROC Curve: Random Forest using Borderline SMOTE Data
TPR: True positive rate
FPR: False positive rate

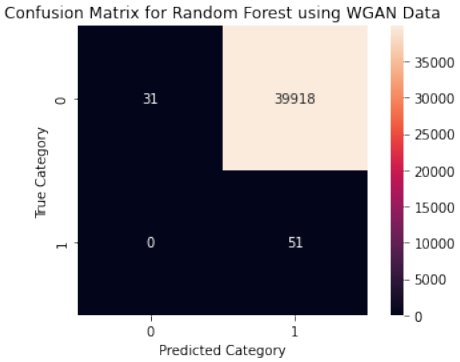

Confusion Matrix for Random Forest using RandomOverSampling Data
True Category
Predicted Category
39933
16
4
47

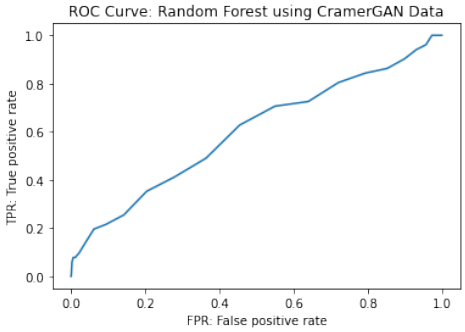

Confusion Matrix for Random Forest using Borderline SMOTE Data
True Category
Predicted Category
39934
15
8
43

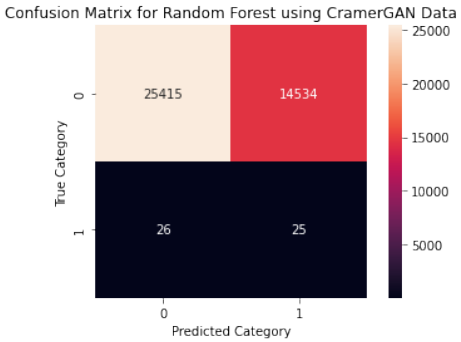

ROC Curve: Random Forest using Vanilla GAN Data
TPR: True positive rate
FPR: False positive rate



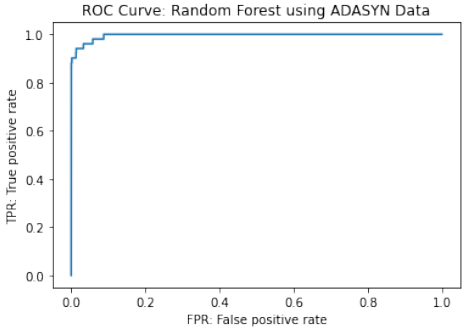


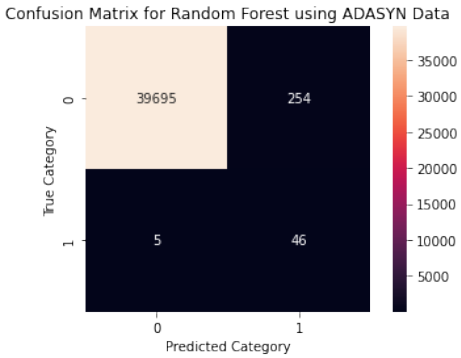






a negative RMSE scoring using GridSearchCV. GridSearch is conducted on 3-folds and the best parameters are when "maximum depth"=8, "maximum features"=0.75, "number of estimators"=50, which gives an RMSE of 0.08. The following graph is another random forest classification on ADASYN with the best parameters.





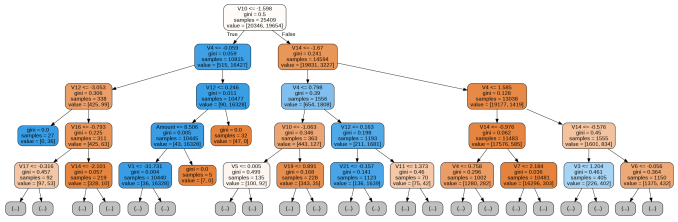


Fig. 9. Random Forest Classifier Tree.

RF classifier on ADASYN data with maximum depth=4 is as seen in the above graph. At the root node, V10 is selected as

the branching feature. Furthermore, V4, V12, and V14 seem to be often used as the branching features, indicating their saliency.

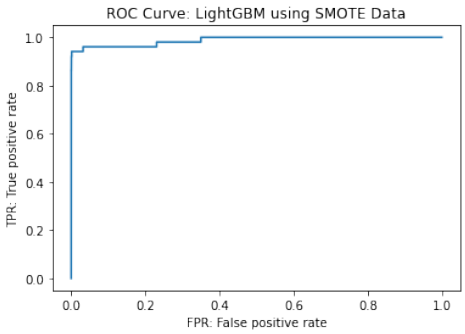
C. LightGBM

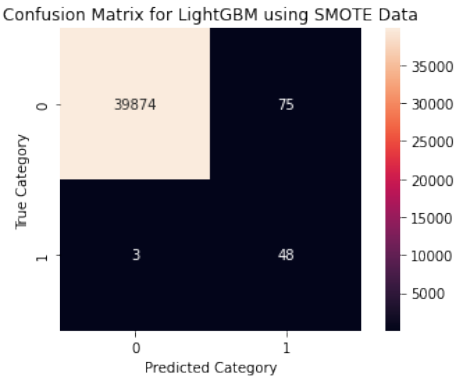
On top of Random Forest, LightGBM is considered as an alternative classification method. The metrics score results are as follows.

	SMOTE	SVM	BSM	ADASYN	ROS
Accuracy	0.998	0.999	0.999	0.998	0.999
Recall	0.970	0.941	0.941	0.950	0.970
Precision	0.695	0.841	0.857	0.680	0.753
F-1	0.775	0.884	0.895	0.756	0.828

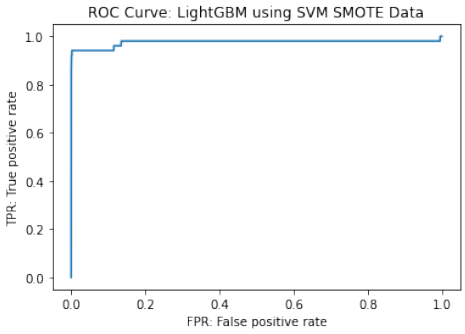
	GAN	CramerGAN
Accuracy	0.159	0.462
Recall	0.471	0.505
Precision	0.500	0.500
F-1	0.138	0.317

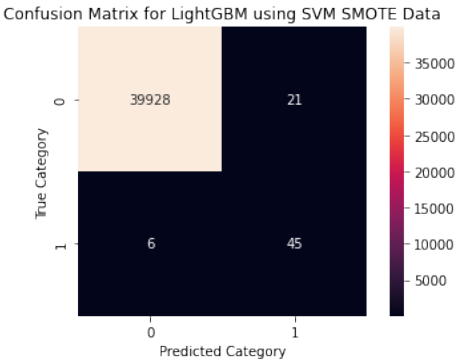
ROC Curve and Confusion Matrix for each dataset is as given in the following.

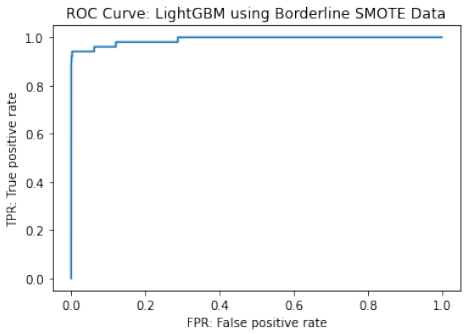


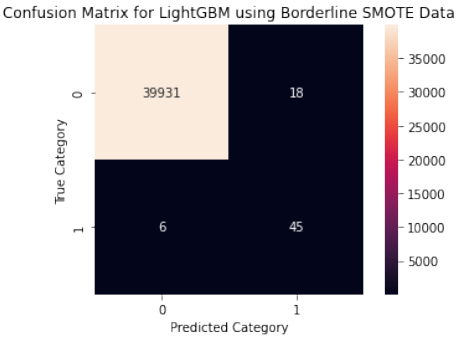


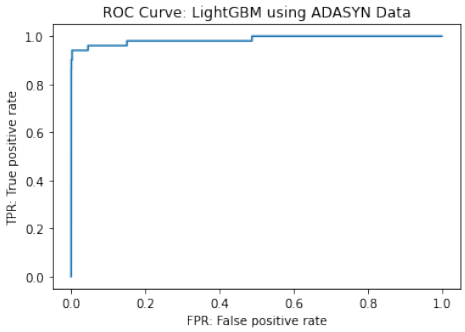
The datasets with the best recall scores are SMOTE and RandomOverSampling. It is once again observed that the synthetic sets created using GANs indicate poor performance. Using GridSearchCV, hyper-parameter tuning is done for SMOTE data. The best parameters are: ”maximum depth”=8, ”number of leaves”=8, ”number of estimators”=100, which gives an RMSE of 0.07. The following graph is another LightGBM classification on SMOTE with the best parameters.

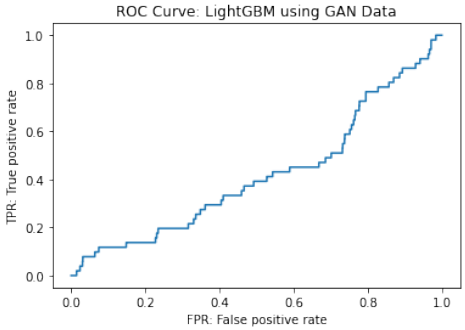


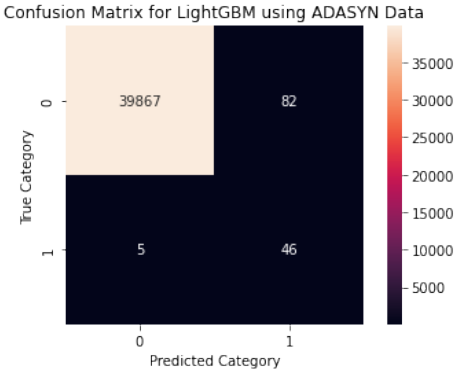


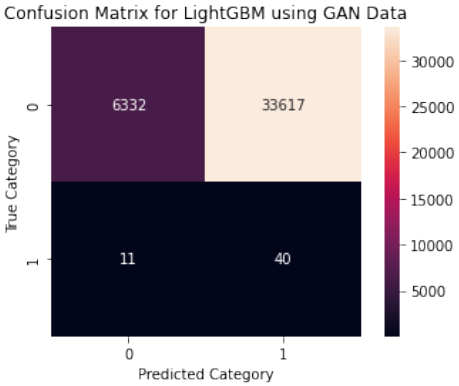


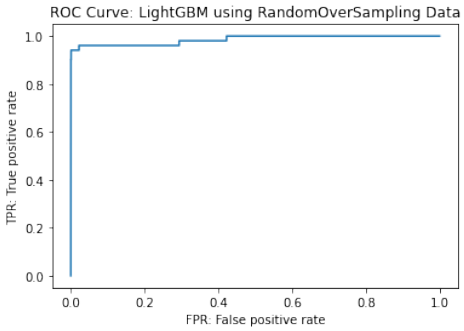


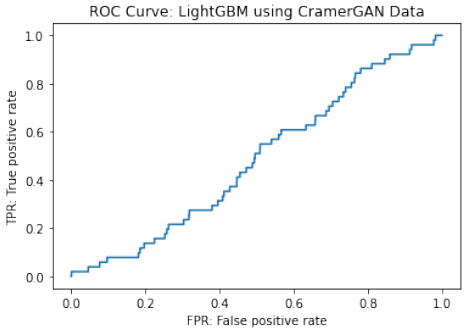

ROC Curve: LightGBM using ADASYN Data
TPR: True positive rate
FPR: False positive rate

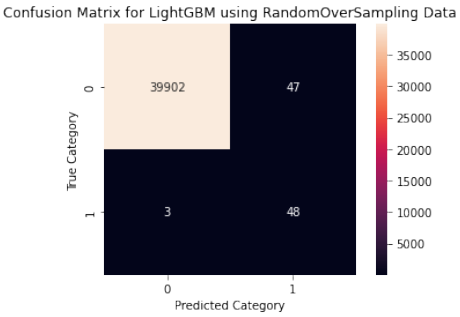

ROC Curve: LightGBM using GAN Data
TPR: True positive rate
FPR: False positive rate

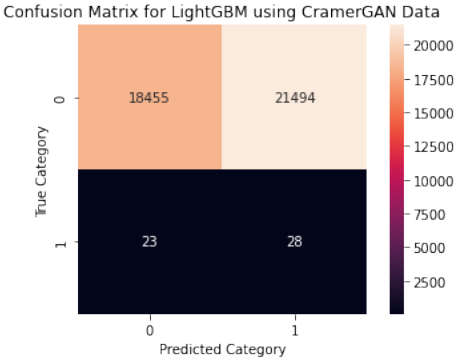

Confusion Matrix for LightGBM using ADASYN Data
39867
82
5
46
True Category
Predicted Category

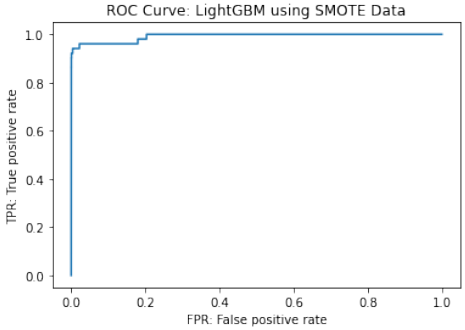

Confusion Matrix for LightGBM using GAN Data
6332
33617
11
40
True Category
Predicted Category

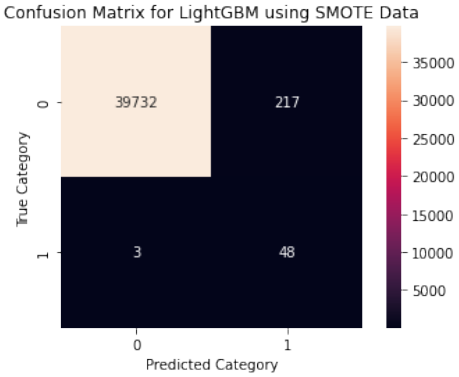

ROC Curve: LightGBM using RandomOverSampling Data
TPR: True positive rate
FPR: False positive rate


ROC Curve: LightGBM using CramerGAN Data
TPR: True positive rate
FPR: False positive rate


Confusion Matrix for LightGBM using RandomOverSampling Data
39902
47
3
48
True Category
Predicted Category

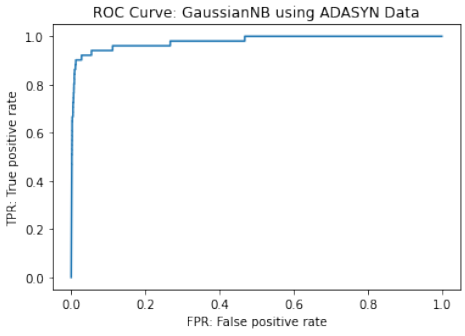

Confusion Matrix for LightGBM using CramerGAN Data
18455
21494
23
28
True Category
Predicted Category





D. Naive Bayes

GaussianNB was also conducted on the data sets and ADASYN performed the best, however, the performance is degraded compared to that of the previous classifiers. (Refer to creditcard_fraud.ipynb for full details.)

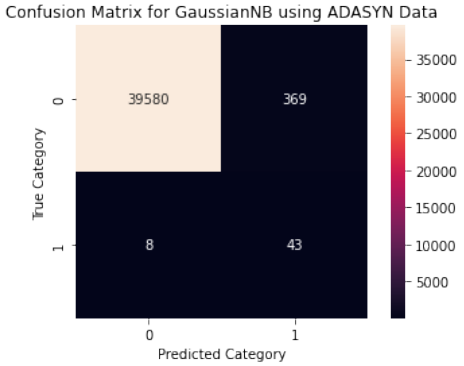


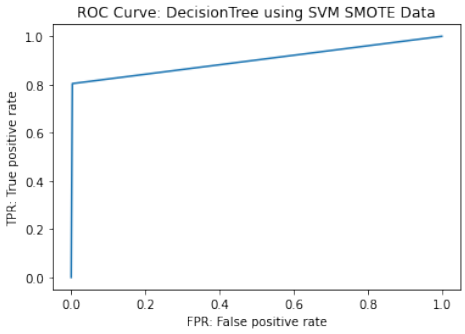
E. Decision Tree

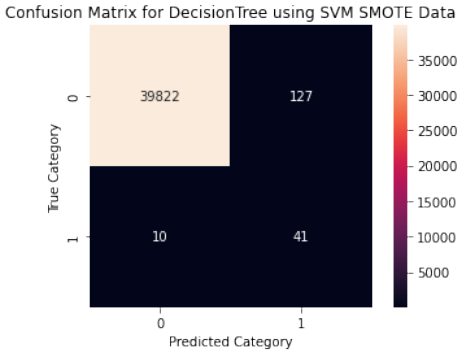
Lastly, Decision Tree performed better than Naive Bayes with its best performance dataset being SMOTE.

V. Conclusion

In summary, re-sampling techniques worked best for creating synthetic data. This may have been due to a lack in the process of generating GANs dataset and also due to a lack in the number of actual fraudulent cases, which limited the number of results that could possibly be created for under-sampling and the size of validation set. For future work, further methodology in generating synthetic data with GANs can be looked into as well as using VAEs (autoencoders) as another alternative source of synthetic data.







VI. Links

Code on Github: https://github.com/rikakks/creditcard_fraud
Dataset: https://www.kaggle.com/mlg-ulb/creditcardfraud