1. Project Overview

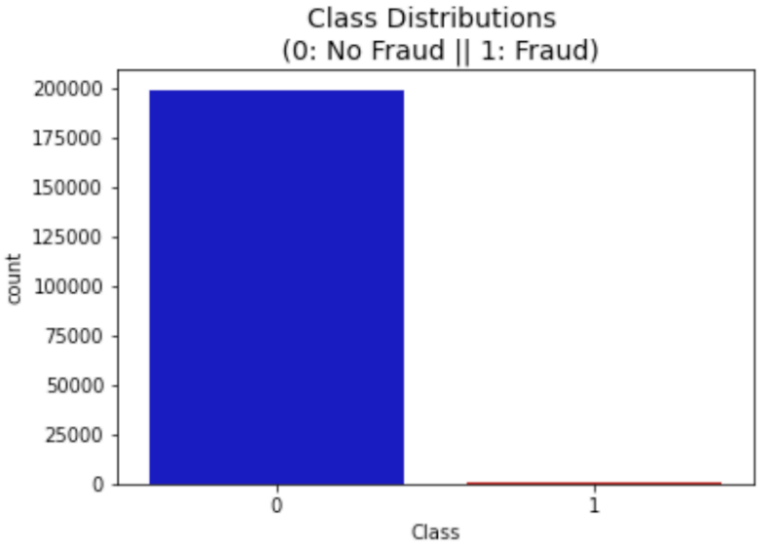
This is a Kaggle competition which is to use advanced classification technique to recognize fraudulent credit card transactions. The dataset contains transactions made by credit cards in September 2013 by European cardholders. For confidentiality issues, most of the features have been PCA transformed. In this project, I include the following approaches and techniques:

- EDA with Pandas and Seaborn
- Deal with imbalance data by different oversample/Under sample techniques
- Dimensionality reduction and visualization
- Regression-based models and tree-based models
- Hyperparameter tuning by GridSearchCV or RandomSearchCV
- Threshold tuning and imbalanced data performance analysis

2. Data Exploration and Data Wrangling

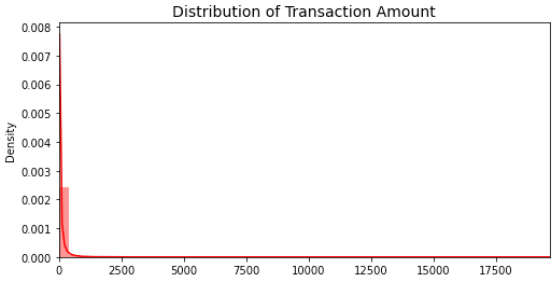
2.1 Data Inspection

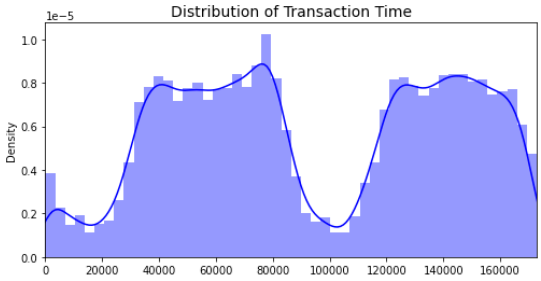
In my dataset, there are 30 predictor variables and 1 response variable at 284,907 observations. Among the 30 predictors, 28 of them are transformed numerical features from PCA to protect the customers' privacy. The other two numerical features "Time" and "Amount" measures the seconds elapsed between each transaction and the first transaction and each transaction amount respectively. In order to have all features on the same scale, I will scale these two features, which will be mentioned in more details later. The target variable is a binary number indicating whether this transaction is fraud (marked as 1) and nonfraud (marked as 0).
There is no duplicated nor missing data. However, the dataset is highly unbalanced with the positive class (frauds) accounts constitutes 0.172% of all transactions. This may raise errors of my classification models since it may "assume" that most of transactions are not fraud instead of detecting patterns that give signs of fraud. I will elaborate in more details about how to deal with imbalanced data in later section.



2.2 Data Exploration

- Distribution: By seeing the distribution, it is clear how skewed these features are. Transaction amount is highly skewed to the right while the distribution of transaction time has two peaks





- Normality: Running a normality test we see that all features are not following a normal distribution.

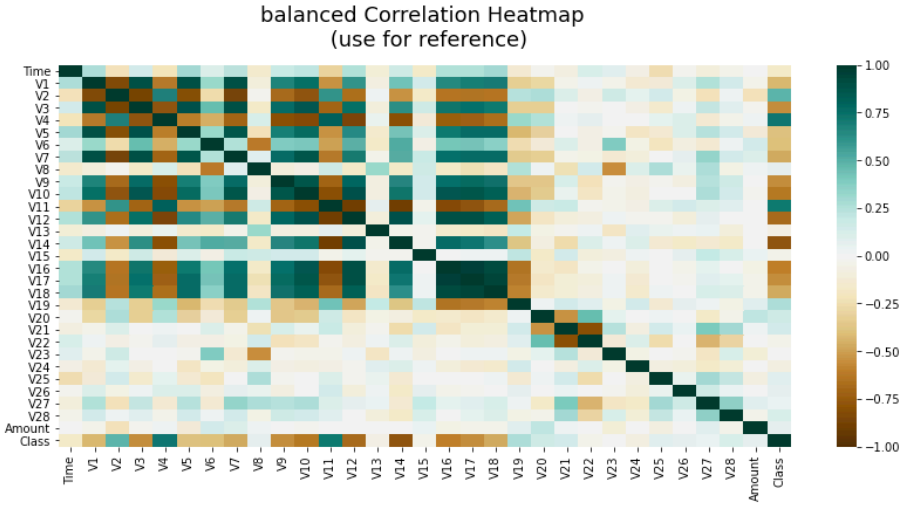
	Feature	Test-statistics	P_value	Decision
0	Time	677579.960345	0.000000e+00	Not Normal
1	V1	154914.102882	0.000000e+00	Not Normal
2	V2	201943.584757	0.000000e+00	Not Normal
3	V3	119229.944534	0.000000e+00	Not Normal
4	V4	23755.308642	0.000000e+00	Not Normal
5	V5	75410.935151	0.000000e+00	Not Normal
6	V6	68630.672233	0.000000e+00	Not Normal
7	V7	102486.775881	0.000000e+00	Not Normal
8	V8	303954.716486	0.000000e+00	Not Normal
9	V9	27471.644374	0.000000e+00	Not Normal
10	V10	89977.754190	0.000000e+00	Not Normal
11	V11	9033.322719	0.000000e+00	Not Normal
12	V12	110162.116432	0.000000e+00	Not Normal
13	V13	379.820368	3.334686e-83	Not Normal
14	V14	106789.715963	0.000000e+00	Not Normal
15	V15	3558.899986	0.000000e+00	Not Normal
16	V16	60957.166525	0.000000e+00	Not Normal
17	V17	191418.689173	0.000000e+00	Not Normal
18	V18	15120.237580	0.000000e+00	Not Normal
19	V19	8111.235674	0.000000e+00	Not Normal
20	V20	100968.729769	0.000000e+00	Not Normal
21	V21	198357.856733	0.000000e+00	Not Normal
22	V22	16231.411289	0.000000e+00	Not Normal
23	V23	266111.668104	0.000000e+00	Not Normal
24	V24	10764.700591	0.000000e+00	Not Normal
25	V25	23653.545712	0.000000e+00	Not Normal
26	V26	12860.571080	0.000000e+00	Not Normal
27	V27	170693.501440	0.000000e+00	Not Normal
28	V28	312595.603950	0.000000e+00	Not Normal
29	Amount	370569.624259	0.000000e+00	Not Normal

- Skewedness: The majority of the features are skewed to the left, which is confirmed by calculating the skewness.

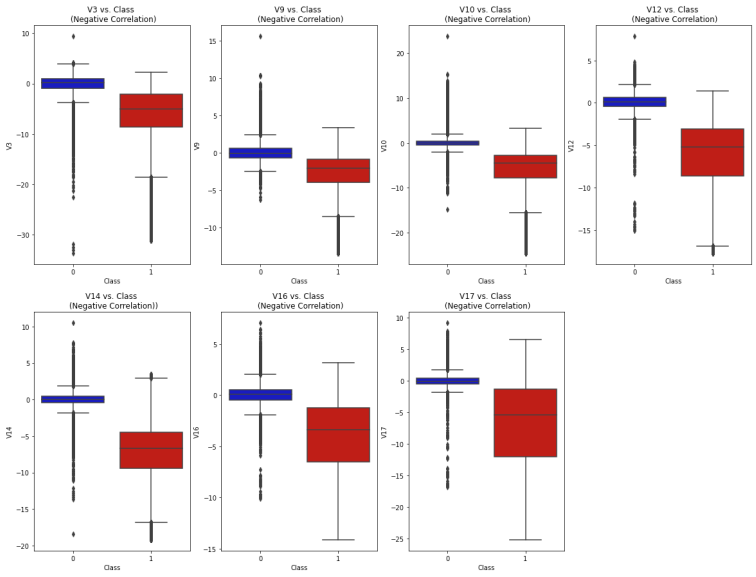
- Scaling: Since all features except for “time” and “amount” went through a PCA transformation, I scale the features that are left by robust scaler to have less outlier influence. To prevent data leakage, the scaler fit from training data will be fit on test data set.

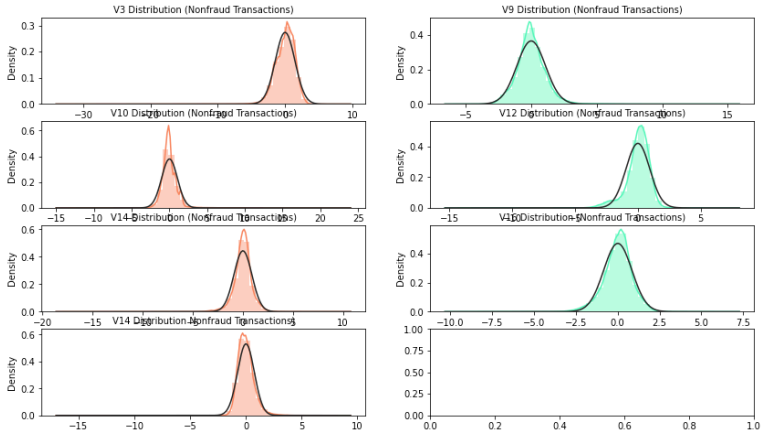
Correlations:

Correlation matrix would be helpful to understand if there are features that influence heavily in whether a specific transaction is fraud. However, since the correlation matrix will be affected by the high imbalance between the two classes, it is critical to use the correlation matrix after balance the dataset

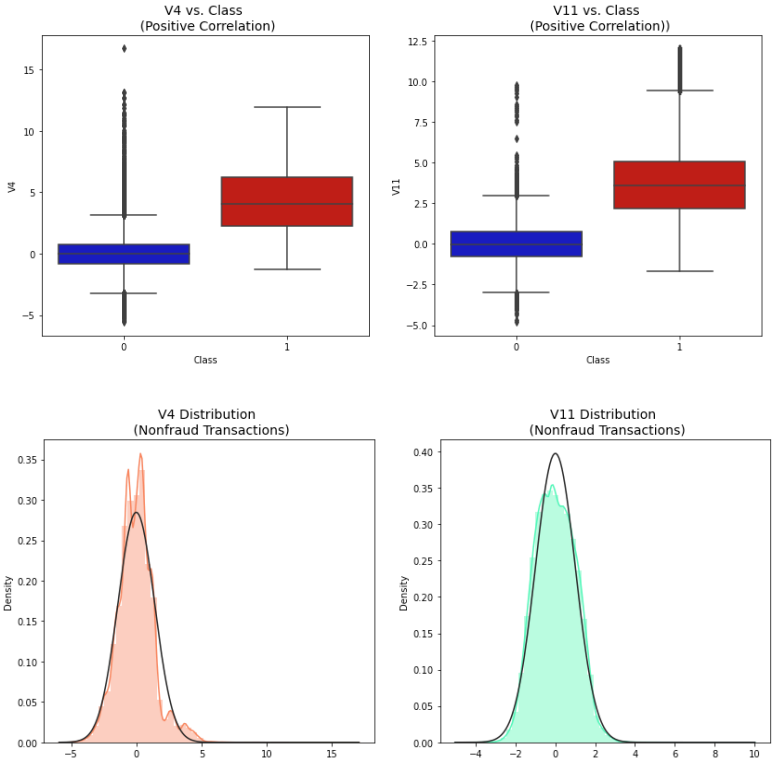


- Negative correlation: V3, V9, V10, V12, V14, V16 and V17 are negatively correlated with the target variable. The boxplots show that the data are fairly concentrated in the interquartile in fraud class while the data are scattered in nonfraud class. The distributions are fairly normal under nonfraud class



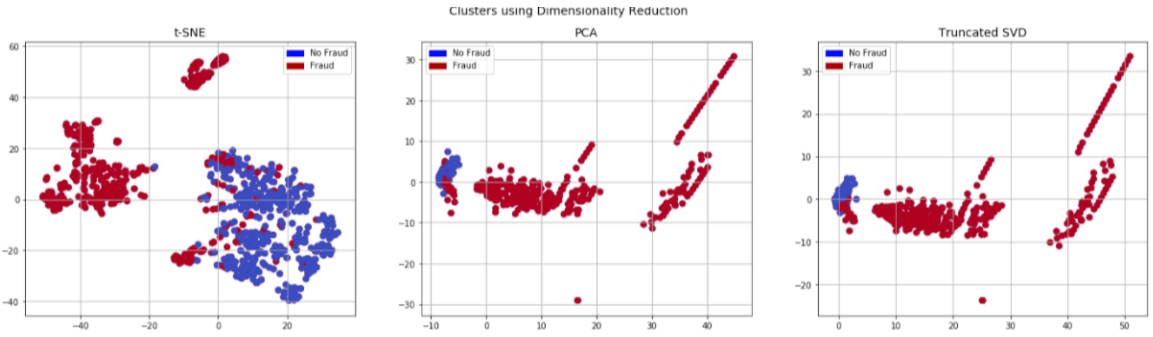


- Positive correlation: V2, V4, V11 are positive correlated with the target variable. The boxplot shows that most of the data are within the interquartile range and the distributions are fairly normal under nonfraud class.



2.3 Dimensionality reduction and visualization

- There are several techniques to visualize high dimension data. For example, t-SNE can accurately cluster the cases that were fraud and non-fraud in our dataset. It is not hard to see that the two classes are separable, and this gives an indication that further predictive models could perform well in separating those two cases.

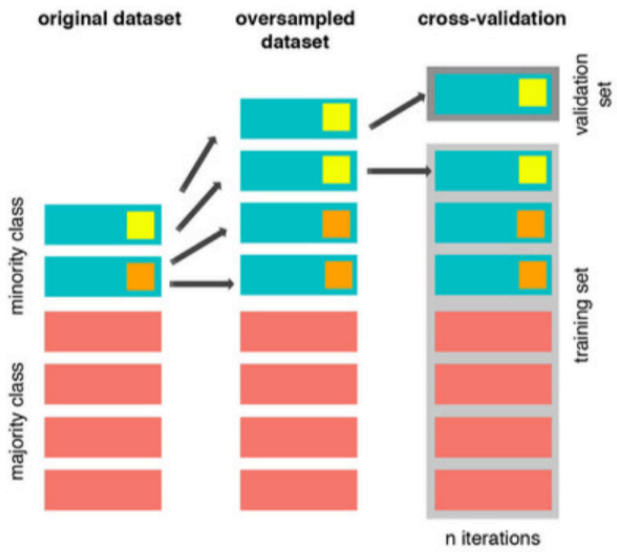


2.4 Oversample/undersample imbalanced data

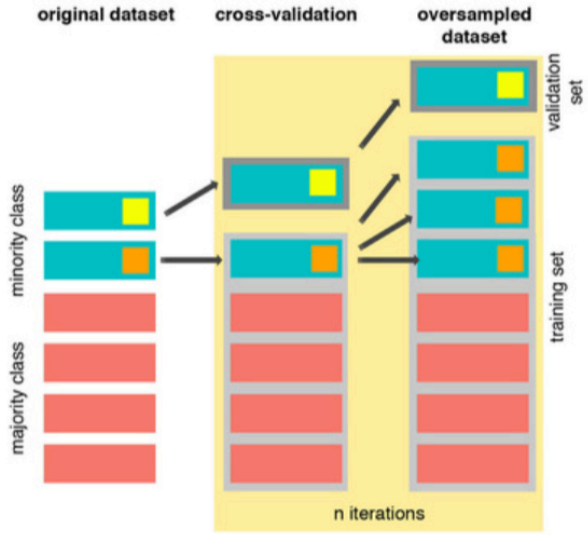
- There are different techniques to balance imbalanced dataset like Nearmiss, random undersampling technique or SMOTE (Synectic Minority Oversampling) technique. Since the data set is not large, undersampling would result in a huge data loss which may impact the model performance, I pursue SMOTE to oversample the data set to make it balanced. After applying SMOTE, the data set becomes more balanced.

2.5 Incorporate SMOTE to data preprocessing pipeline and cross-validation

- A common mistake made in oversample to cross-validate model performance is to oversample before cross validation. This may cause data leakage issue since the validation set may share the same information as training set.
  The Wrong Way



- The correct way to do so is to oversample “during” cross-validation rather before it. To incorporate the whole process into a pipeline, I leverage imblearn pipeline package which only apply oversample on train folds but not validation fold.



3. Models

3.1 Evaluation metrics

Since the data set is imbalanced, using accuracy score would be misleading. My goal is to predict classify fraud and nonfraud. Therefore, in order make my model robust to perform consistently over different test datasets, I choose F1 score as my main evaluation metric and use precision/recall score or confusion matrix as a supplement.

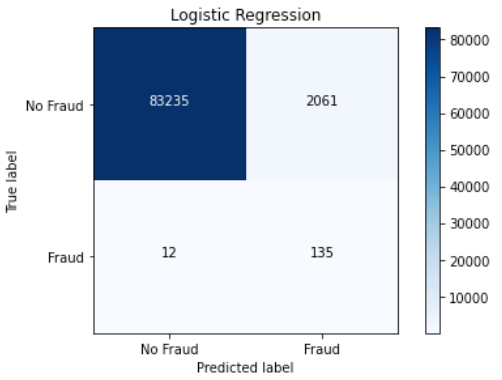
3.2 Baseline models

I created two baseline classification models (See table for a detailed summary). They both suffer from high Type I error which leads to a low precision relative to recall and thus a low F1-score. Each baseline model has its own limitation which I will demonstrated in more details. Since these baseline models were intended as benchmarks, I will develop more advanced classification models which refine the model limitation and achieve a much better performance.

Model	F1-Score	Precision	Recall
Logistic Regression	11.12%	6.11%	91.84%
Gaussian Naïve Bayes	10.72%	5.72%	85.71%

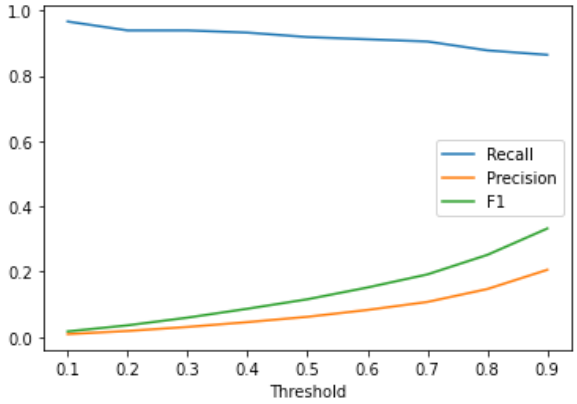
1. Logistic regression model:

- The first baseline model is a logistic regression model with L1 penalty by Grid searching on regularization strength. This model achieves a test F1 score at 11.12% with the test recall score at 91.84% while the test precision score at 6.11%. The high recall versus the low precision suggests that the model suffers from big Type I error, and this has been illustrated in more details using confusion matrix and the classification report below.



	precision	recall	f1-score	support
No Fraud	1.00	0.98	0.99	85296
Fraud	0.06	0.92	0.11	147
accuracy			0.98	85443
macro avg	0.53	0.95	0.55	85443
weighted avg	1.00	0.98	0.99	85443

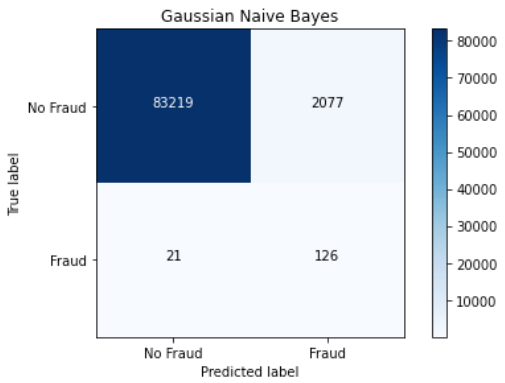
- Precision-recall tradeoff and threshold-moving:
  Considering the nature of this classification problem is to classify fraudulent and nonfraudulent transactions, I would want to keep a descent recall while maintaining a balanced tradeoff between precision and recall based on F1 score.
  If the predicted probability is above the threshold, the sample would be classified as a positive class and the other way around for a negative class. By altering the threshold, F1 score is increased but not with a significant amount. This may suggest that the potential model limitation which will be illustrated in more details in later section



- Model limitation:
  After hyperparameters tuning and threshold-moving, there is no significant improvement in model performance. One hypothesis is that the data is not linearly separable so that it is hard to find a linear decision boundary between the two classes. This hypothesis could be support by observing the data clustering chart using dimensionality reduction technique shown in the earlier section. In addition, this hypothesis has been further confirmed in later section when applying tree-based classification models which achieves a significant performance improvement.

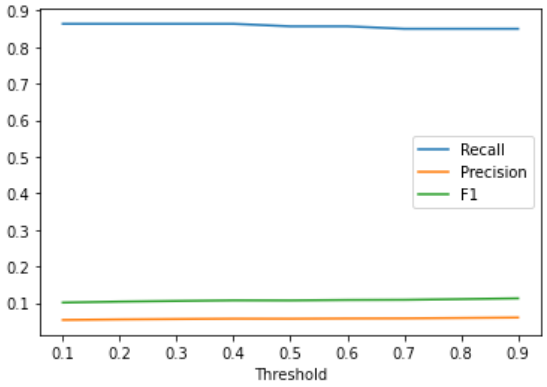
2. Gaussian Naïve Bayes model:

- The second baseline is a Gaussian Naïve Bayes model by Grid searching on largest variance proportion. This model achieves a test F1 score at 10.72% with the test recall score at 85.71% while the test precision score at 5.72%. The high recall versus the low precision suggests that the model suffers from big Type I error, and this has been illustrated in more details using confusion matrix and the classification report below



	precision	recall	f1-score	support
No Fraud	1.00	0.98	0.99	85296
Fraud	0.06	0.86	0.11	147
accuracy			0.98	85443
macro avg	0.53	0.92	0.55	85443
weighted avg	1.00	0.98	0.99	85443

- Precision-recall tradeoff and threshold-moving:
  Similar with logistic regression, altering the threshold doesn't lead to a significant increase in F1 score, which suggests the potential model limitation which will be mentioned in more details in later section

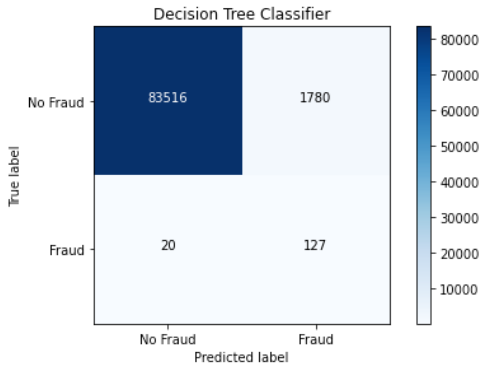


- Model limitation:
  After hyperparameters tuning and threshold-moving, there is no significant improvement in model performance. One hypothesis is that the features violate the assumptions that they follow a Gaussian distribution with no co-variance. This can be confirmed by the feature distribution charts in the earlier section. This limitation can be remediated using more advanced tree-based classification models mentioned in later section.

3.3 Tree-based models

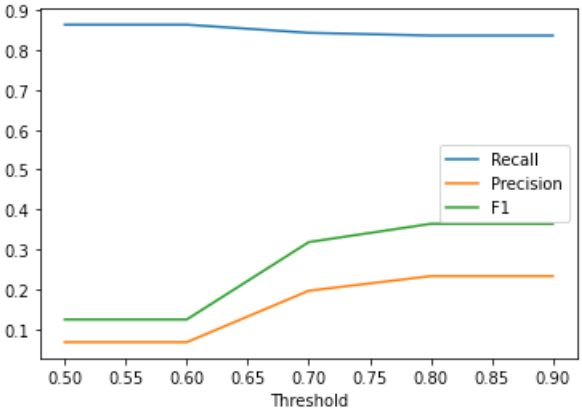
3.3.1 Decision Tree Classifier

I fit the data with Decision Tree classifier and find the best *max_depth* for the tree based on 5-fold cross validation. The best model chooses a *max_depth* = 4, which is not so complex to overfit while also includes a satisfying number of features to avoid underfitting. The test F1-score is 16.91% better than baseline models, which confirms the non-linear decision boundary hypothesis. Since I built a basic tree which may fail to fit the underlying complexities in the predictor-response, test F1-score is not significantly increased. But enhancement will be made in other tree-based models in later sections.



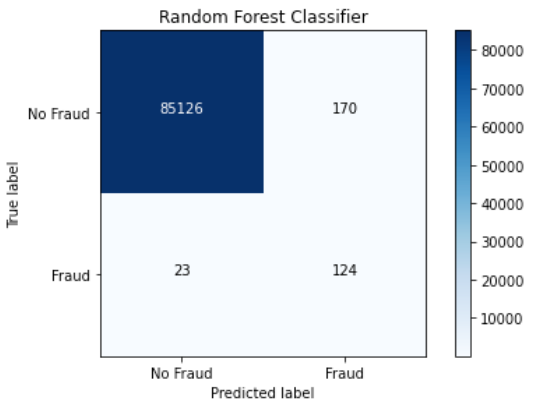
	precision	recall	f1-score	support
No Fraud	1.00	0.98	0.99	85296
Fraud	0.07	0.86	0.12	147
accuracy			0.98	85443
macro avg	0.53	0.92	0.56	85443
weighted avg	1.00	0.98	0.99	85443

Following the similar procedure to move the threshold to classify fraudulent and nonfraudulent cases, there is much more obvious increase in F1-score while putting stricter threshold.



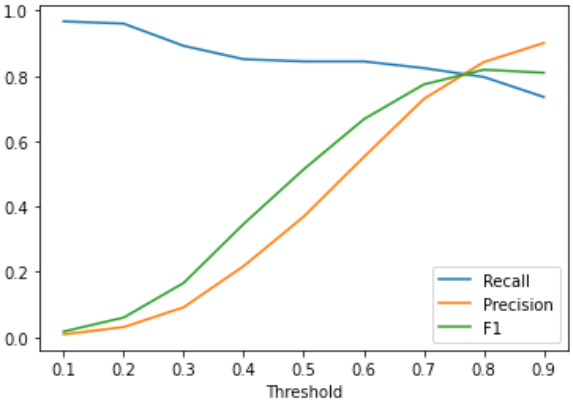
3.3.2 Random Forest Classifier

I pick Random Forest classifier since it tends to reduce the impact of the strongest feature through decorrelation of trees. Random Forest is pretty robust to overfitting in that the model randomly chooses one feature from sqrt(number_of_features) for every split in every tree. For the sake of computational time, I don't construct really deep trees, fit the model with only 150 trees and then ensemble them into a forest. By grid search, the best model chooses a *max_depth* = 7 , which is way deeper than the optimal depth *max_depth* = 4 in Decision Tree Classifier.



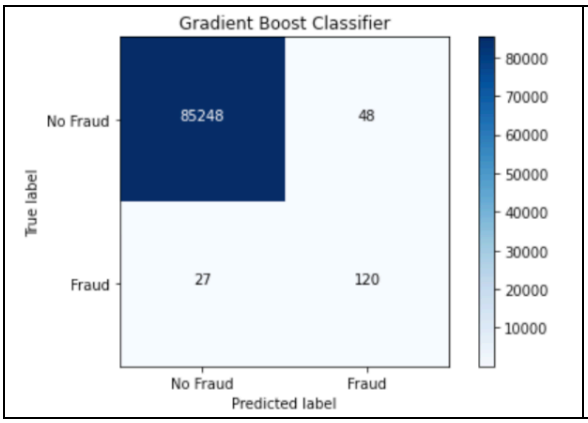
	precision	recall	f1-score	support
No Fraud	1.00	1.00	1.00	85296
Fraud	0.42	0.84	0.56	147
accuracy			1.00	85443
macro avg	0.71	0.92	0.78	85443
weighted avg	1.00	1.00	1.00	85443

Like before, I adjust the threshold to find the balance between recall and precision and use F1-score as my main evaluation metric to assess the tradeoff. F1-score is largely increased to 81.82% with the threshold cutoff at 0.8.



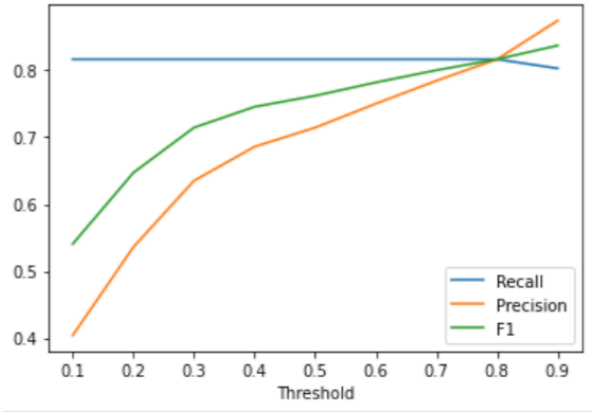
3.3.3 Gradient Boost Classifier

Continue my journey by fitting my model with Gradient Boosted Classifier with a default learning rate of 0.1. As Gradient Boost Classifier is more powerful algorithm in that it can learn to boost the performance by learning the previous tree. Thus, I have a better F1-score than the two tree-based classifier mentioned above with the optimal *max_depth* = 7 and ensemble 150 trees.



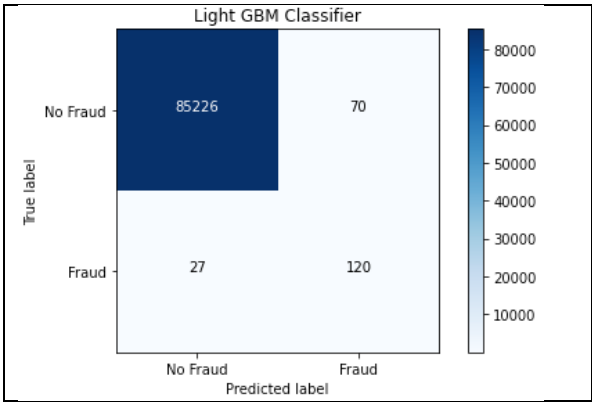
	precision	recall	f1-score	support
No Fraud	1.00	1.00	1.00	85296
Fraud	0.71	0.82	0.76	147
accuracy			1.00	85443
macro avg	0.86	0.91	0.88	85443
weighted avg	1.00	1.00	1.00	85443

Again, I adjust the threshold to find the balance between recall and precision and use F1-score as my main evaluation metric to assess the tradeoff. The model achieves 81.63% F1-score with the threshold cutoff at 0.8.



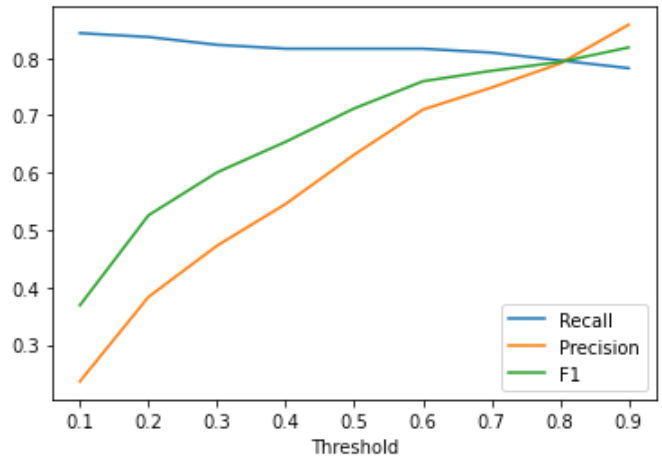
3.3.4 LightGBM Classifier

Light GBM is a fast, distributed, high-performance gradient boosting framework based on decision tree which splits the tree leaf wise with the best fit whereas other boosting algorithm split the tree depth wise or level wise. The best model chooses *max_depth* = 20, which is way deeper than the optimal depth in Decision Tree and Random Forest Classifier.



	precision	recall	f1-score	support
No Fraud	1.00	1.00	1.00	85296
Fraud	0.63	0.82	0.71	147
accuracy			1.00	85443
macro avg	0.82	0.91	0.86	85443
weighted avg	1.00	1.00	1.00	85443

I adjust the threshold to find the balance between recall and precision and use F1-score as my main evaluation metric to assess the tradeoff. The model achieves 79.32% F1-score with the threshold cutoff at 0.8.



3.3 Model Selection

Both Random Forest and Gradient Boost classifiers achieve a significant improvement in F1-score compared to baseline L1-regularized logistic regression and Gaussian Naïve Bayes model. This also implies that the features are not linearly separable and thus a tree-based model is needed to make classification. The goal of the model is to classify fraudulent cases, so in addition to a good F1-score, I also want to have a relatively high recall score. In Random Forest classifier, the test F1-score is 81.82% and the recall is 75.60%. But in the Gradient Boost Classifier, recall is 6.03% higher with only 0.19% trade off from F1-score. After a comprehensive model performance analysis, I choose Gradient Boost Classifier as my final model with the threshold at 0.8.

Model	F1-Score	Precision	Recall	Threshold
Decision Tree Classifier	36.40%	23.25%	83.67%	0.8
Random Forest Classifier	81.82%	84.18%	75.60%	0.8
Gradient Boost Classifier	81.63%	81.63%	81.63%	0.8
LightGBM Classifier	79.32%	79.05%	79.60%	0.8

4. Feature Interpretation for the Best Model