CSL2050: Pattern Recognition and Machine Learning

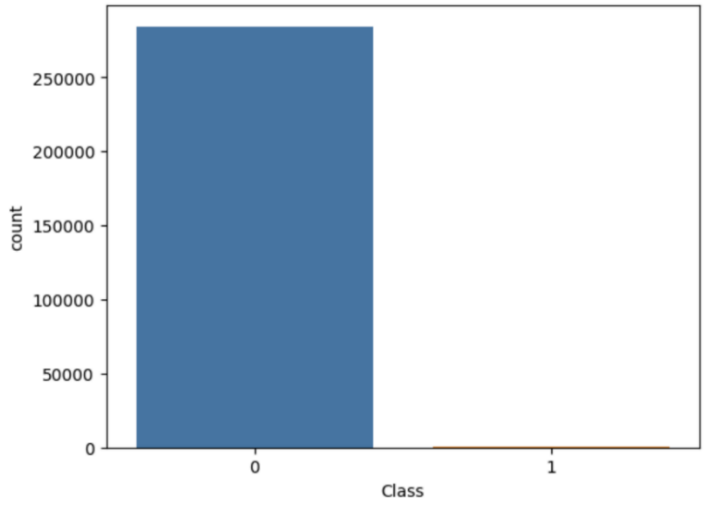
Report on Minor Project

"Credit Card Scam Detection"

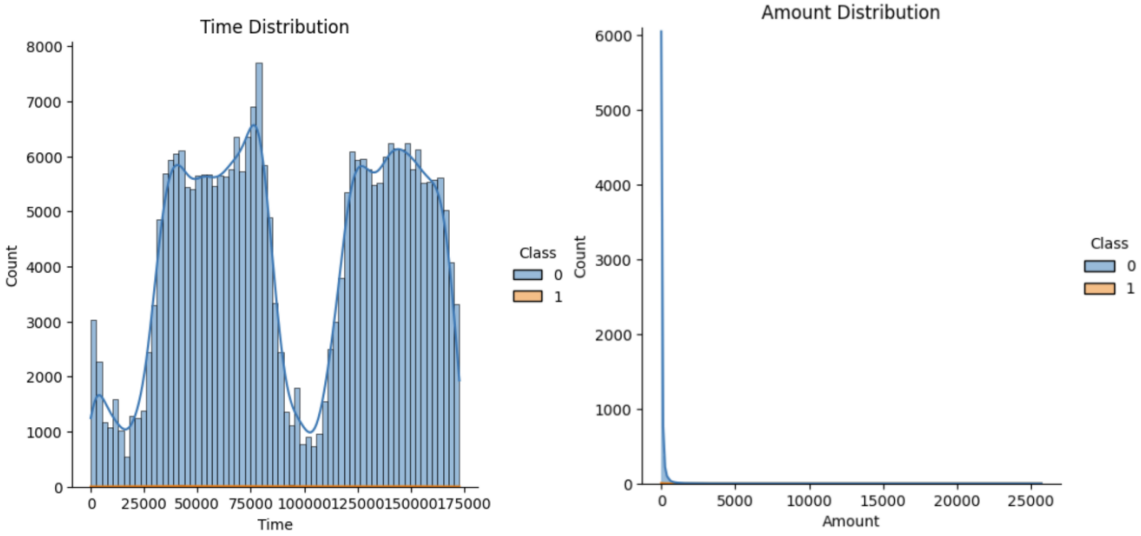
Submitted by: Shreyas Vaidya, B21CS072
Soham Parikh, B21CS074
Yash Shrivastava, B21CS079

Data Pre-processing

The dataset is imported using Pandas. There are no null values and all the features are found to be continuous ratio data. The targeted variable is binary. So, the process is that of binary classification.
The data is found to be highly skewed with label 0 of 284315 samples and label 1 of 492 samples, i.e., 99.83% and 0.17% respectively, as shown:



Other noteworthy features are expressed below:

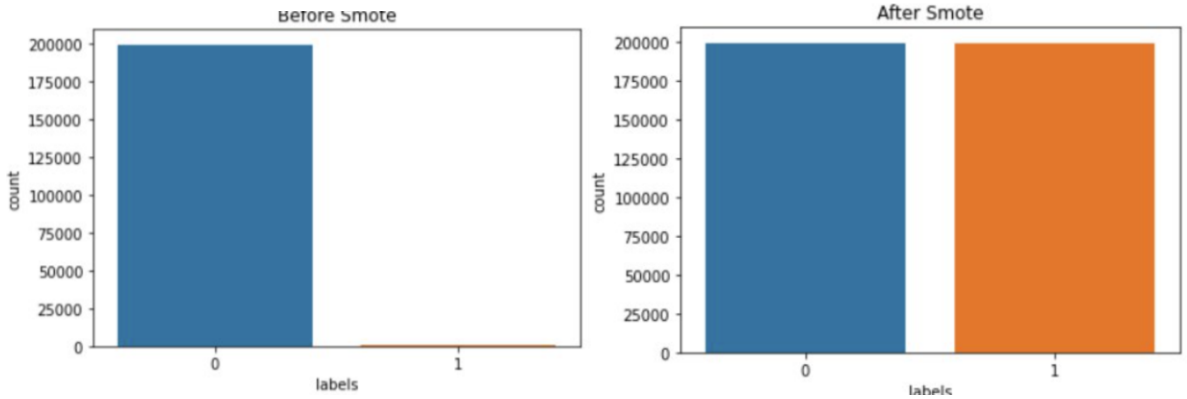


The distribution of transactions with time is not concentrated much, but the amount of money for transaction is highly fixed towards value lesser than 1000.

The other features that are provided are already produced using PCA beforehand. So, those features are already scaled. The remaining features to be scaled are 'Time' and 'Amount' which are plotted above. They are scaled using sklearn.preprocessing.RobustScaler. The scaled values are then included into the dataset and the non-scaled values are dropped.

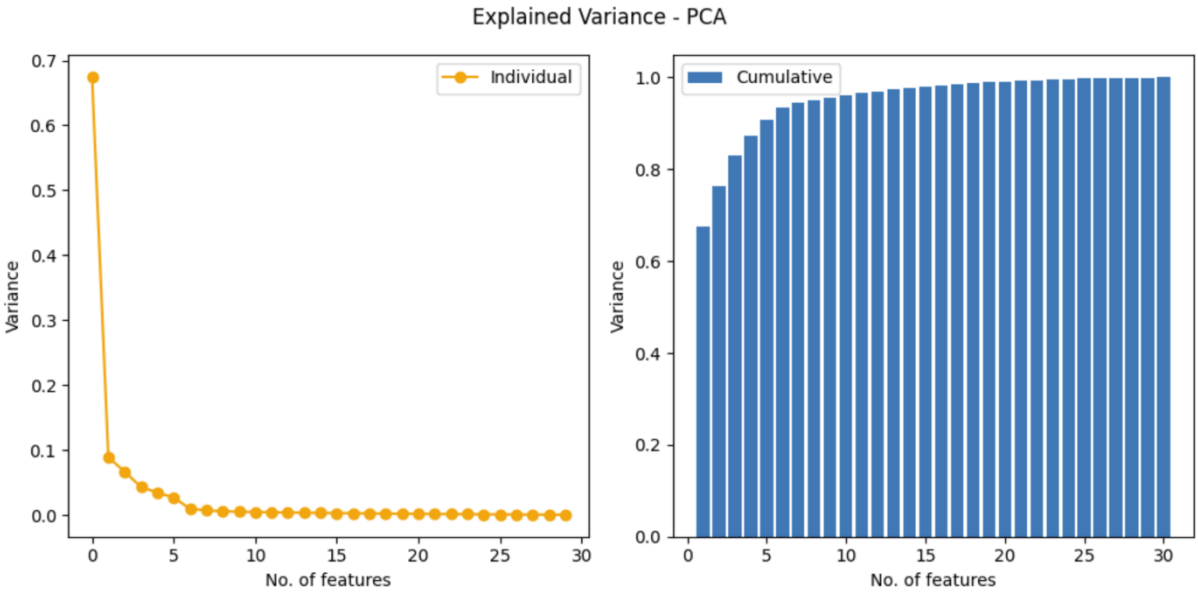
The data is split into training, validation and testing sets in the ratio of 70:10:20 using sklearn.model_selection.train_test_split.

The skewness of the data is in contrast to the requirement of obtaining a model to identify scams. The skewed data will lead to less weight on the minority class which is fraud data in this case. The fraud data is the one which is the core of the analysis, hence to deal with the skewness of the dataset, SMOTE(Synthetic Minority Oversampling Technique) is applied using imblearn.over_sampling.SMOTE. It produces a dataset that has equal values of data for both the labels. The data is modified in such a way that the number of values of under sampled label gets increased to the level of over sampled one, i.e., both 199029 samples.



By oversampling the minority class, SMOTE can help the model learn the patterns of fraudulent transactions and improve the model's ability to distinguish between fraudulent and legitimate transactions. This can lead to better model performance on the minority class and overall. The **testing and validation datasets are not modified using SMOTE**. They are kept at their original states to **ensure true performance** of the models not the one with artificial data.

Next, we apply PCA through sklearn.decomposition.PCA on the augmented data and see the possibility of dimensionality reduction for the data as the number of features(30) is large. The resultant is:



It can be seen by elbow method and the ratio of cumulative explained variance that the optimum number of principal components is 5. Later, both the reduced after PCA data as well as the original augmented data will be taken for training the models and would be compared on their resultant performances.

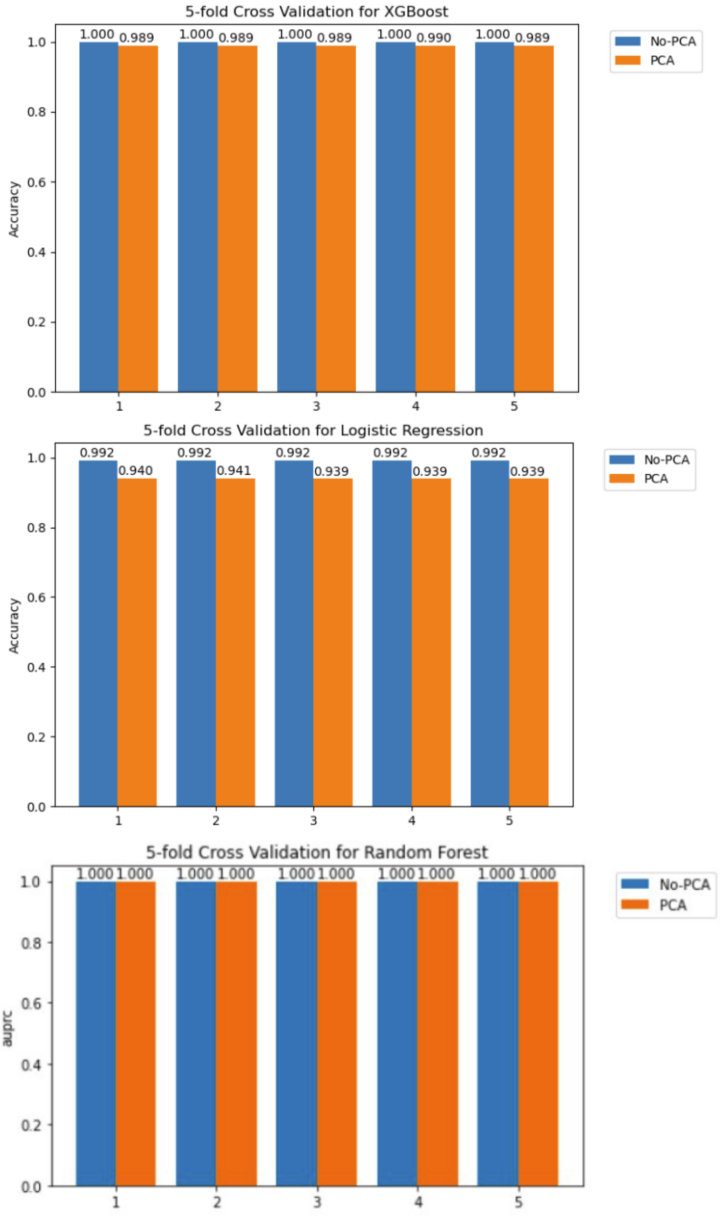
Model Training & Validation

The models that are chosen to be trained are:

- Random Forest Classifier
  An ensemble learning technique, employed due to it being highly effective for large datasets while having a high degree of accuracy.
- XGBoost Classifier
  A powerful machine learning algorithm that provides low bias, low variance, and high performance on highly skewed datasets. It is also fast, scalable, and provides useful feature importance scores.
- Logistic Regressor
  A fast and efficient algorithm that can handle large datasets with a large number of features with high bias and low variance. It provides a coefficient for each independent variable that can be used to identify the most important features in the dataset.
- Artificial Neural Network
  A powerful machine learning algorithm that can model complex relationships between variables and handle class imbalances in highly skewed datasets. They can be computationally expensive to train and number of layers is minimized to check overfitting.

- Support Vector Machine Classifier
  A powerful machine learning algorithm that can model both linear and non-linear relationships between variables and handle class imbalances in highly skewed datasets with low bias and high variance.
- Gaussian Naïve Bayes Classifier
  A simple and fast algorithm that can handle large datasets with a large number of features. It has low variance and is less prone to overfitting than more complex models.

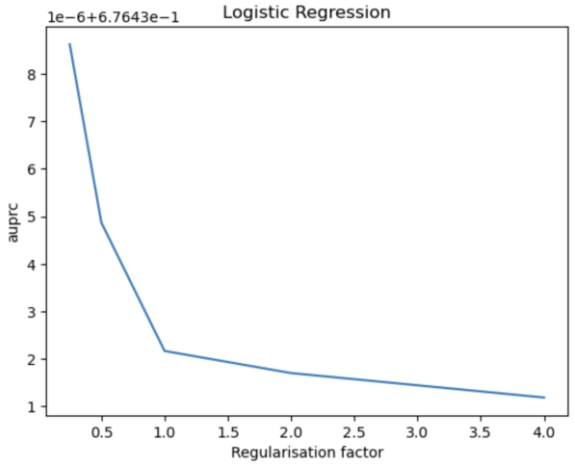
All the models undergo 5-fold Cross Validation over the reduced as well as the augmented data. To do this a function cvs_5_fold is created. It takes the classifier object, the reduced data and the non-reduced data as inputs and generates the results of 5-fold cross validation in text and bar-plot graphs. It does so by using KFold and cross_val_score of sklearn. The results are:

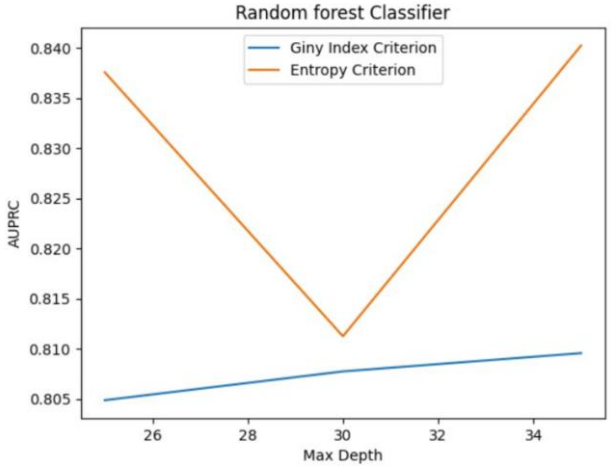


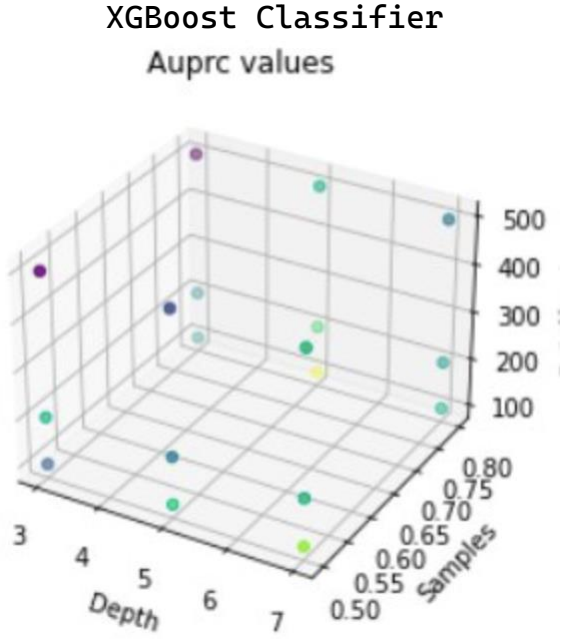
It can be seen that the effect of PCA on performance of model is not much. So, we utilise the reduced data for all the models. The hyperparameters for each of the models are tuned using train_and_tune functions defined independently for each of the models. The function takes in train and validation data as input and utilises the AUPRC(Area under precision-recall curve) scores of the models on a range of hyperparameters to get the tuned hyperparameters.

The cross-validation as well as hyperparameter tuning was not done for the SVM Classifier as the required computations were extremely lengthy. Similarly, GridSearchCV of sklearn was considered for the hyperparameter tuning, but due to time constraints it was eventually dropped.

The results of tuning are obtained utilising the following graphs:







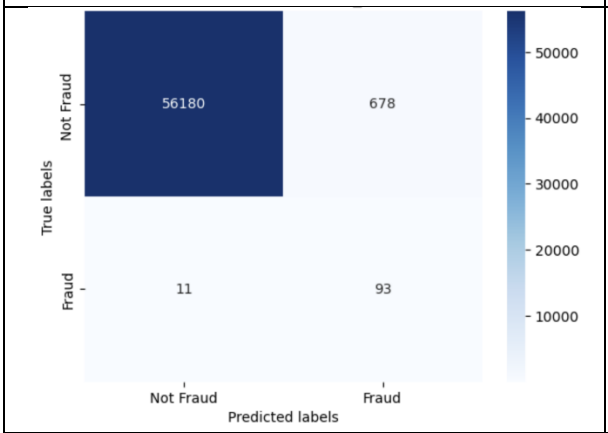
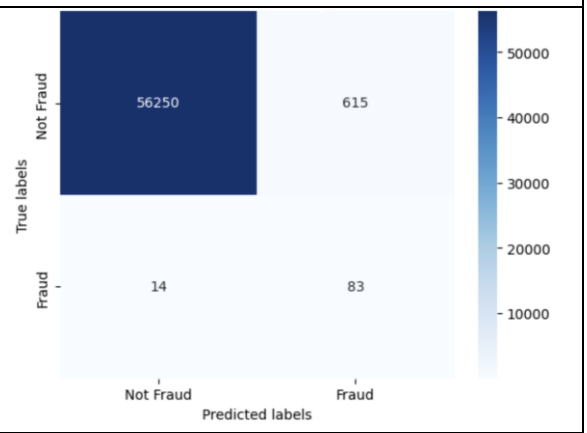
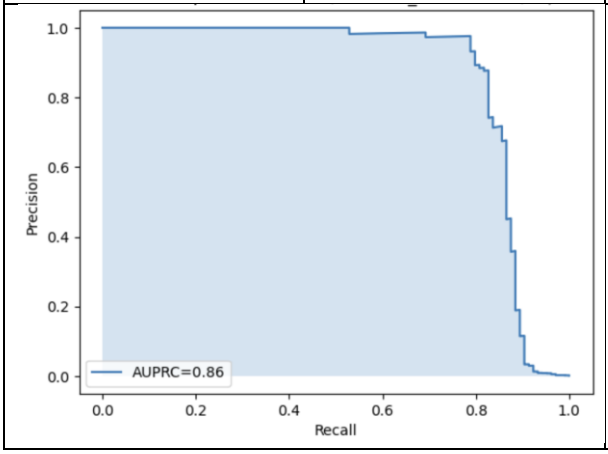
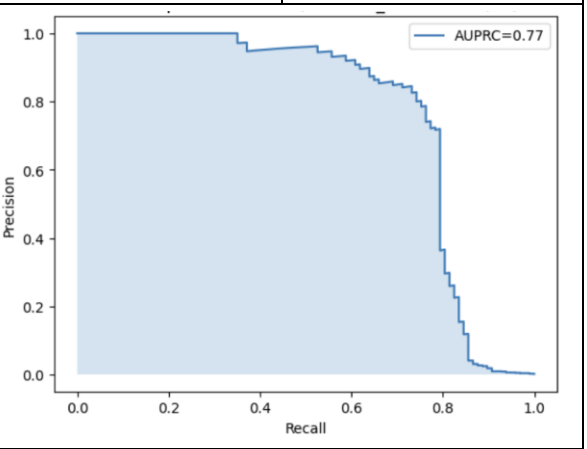
Model Performance

A print_report function was defined to print the conclusive reports of the model, it comprises of a:

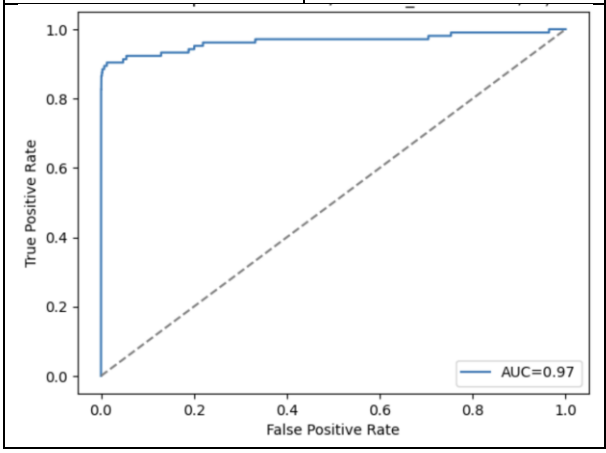
- Confusion Matrix,
- Classification report, with highlighted accuracy score, balanced accuracy score, F1 Score, precision and recall, ROC_AUC_Score and average precision score
- Area Under Precision-Recall curve(this curve is used best to understand the performance of the models on imbalanced data, since ROC curve is dependent on the change in FPR, which would be very slow, while PCR curve would be dependent on the Precision)
- Receiver Operating Characteristic curve

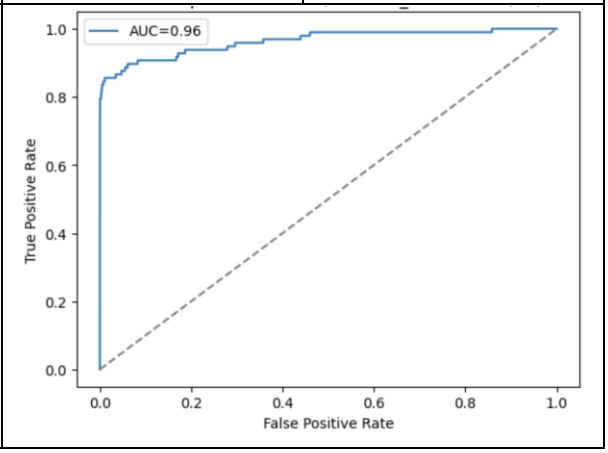
The performance of each of the models is summarised as follows:

❖ XGBoost Classifier

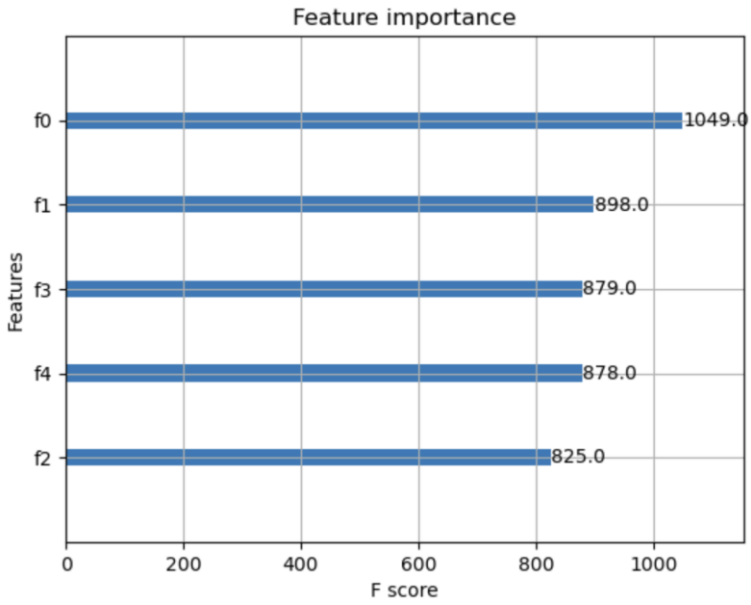
<table>
<tr><th colspan="2">Tuned</th><th colspan="2">Untuned</th></tr>
<tr><td colspan="2"> 
</td><td colspan="2"> 
</td></tr>
<tr><td>Accuracy</td><td>0.99</td><td>Accuracy</td><td>0.99</td></tr>
<tr><td>Balanced Accuracy</td><td>0.94</td><td>Balanced Accuracy</td><td>0.92</td></tr>
<tr><td>F1 Score</td><td>0.21</td><td>F1 Score</td><td>0.21</td></tr>
<tr><td>AUPRC</td><td>0.857</td><td>AUPRC</td><td>0.767</td></tr>
<tr><td colspan="2"> 
</td><td colspan="2"> 
</td></tr>
</table>

AUC	0.965	AUC	0.963

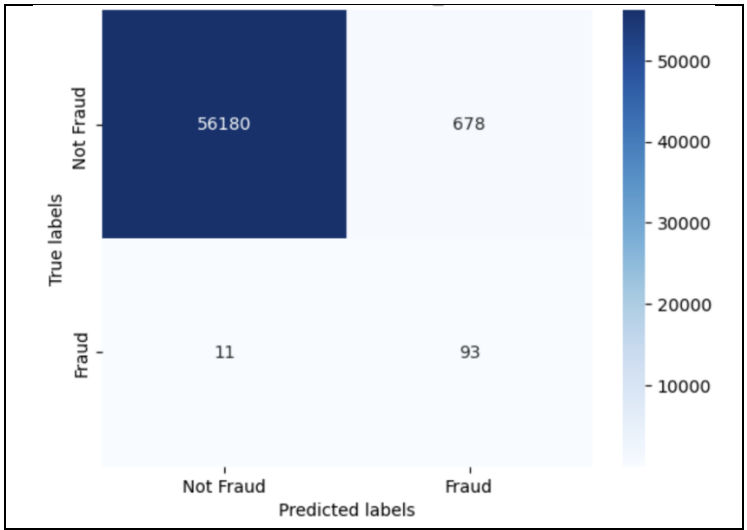




The feature importance for tuned model is as shown:

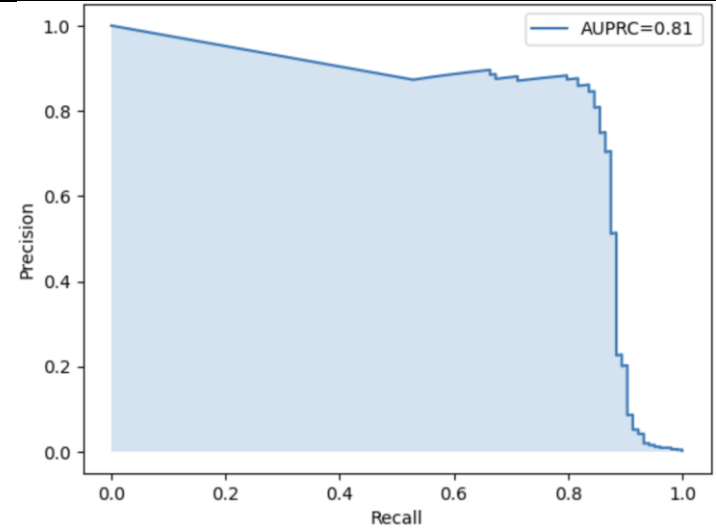


❖ Logistic Regressor(tuned)

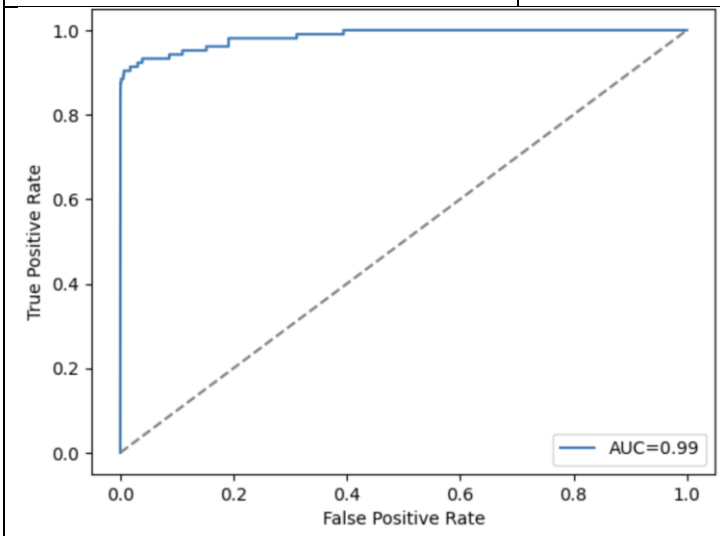


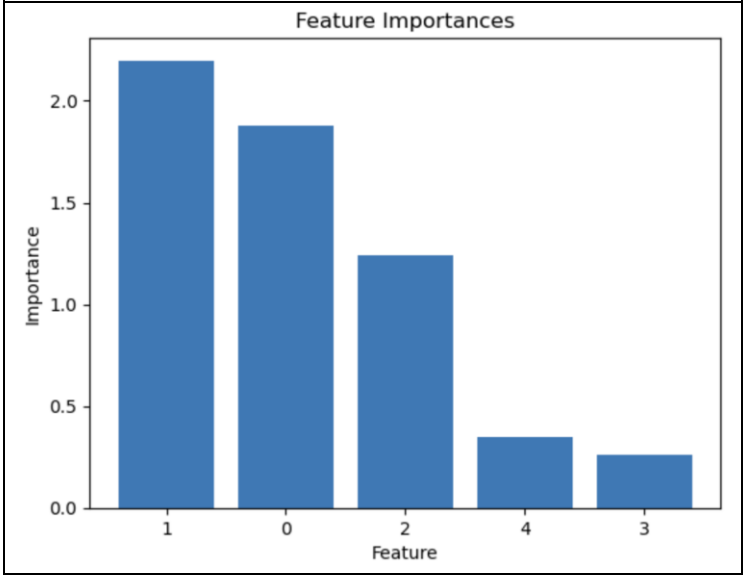
Accuracy	0.98

Balanced Accuracy	0.94
F1 Score	0.12
AUPRC	0.807

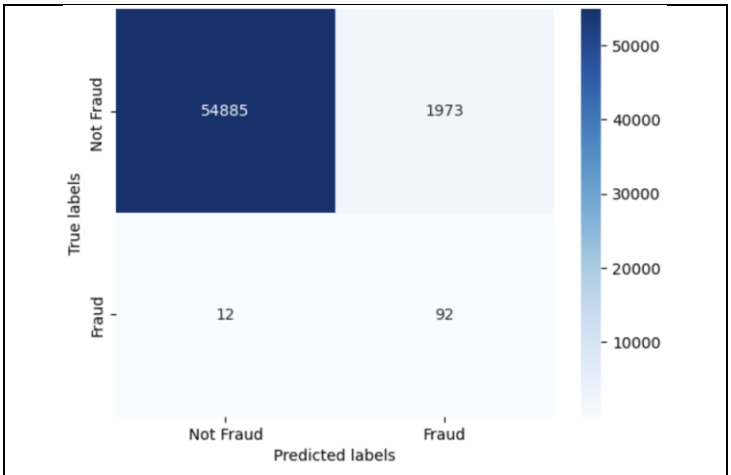


AUC	0.985

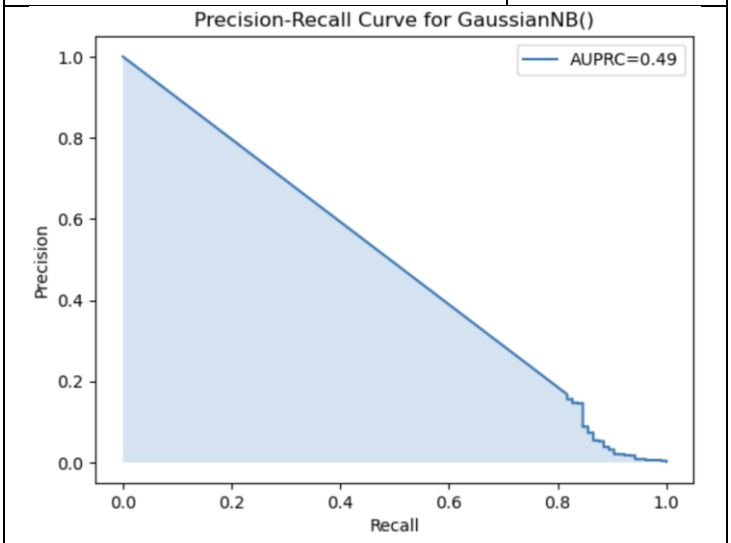




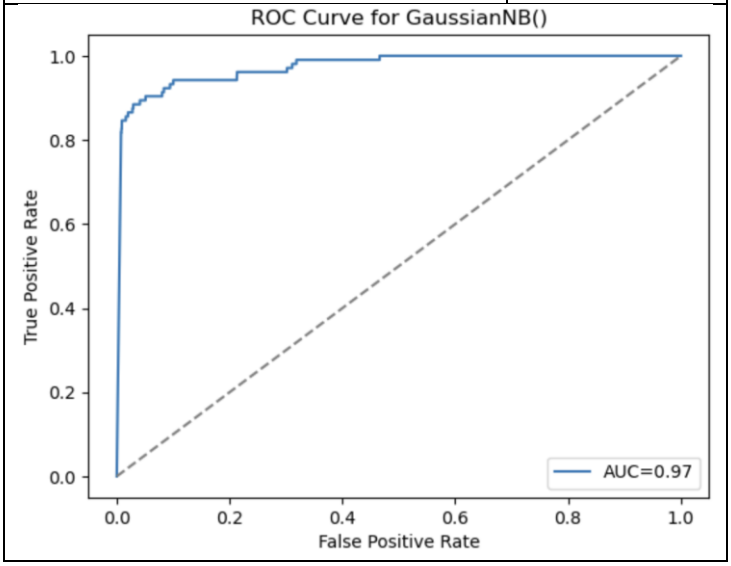
❖ Gaussian Naïve Bayes



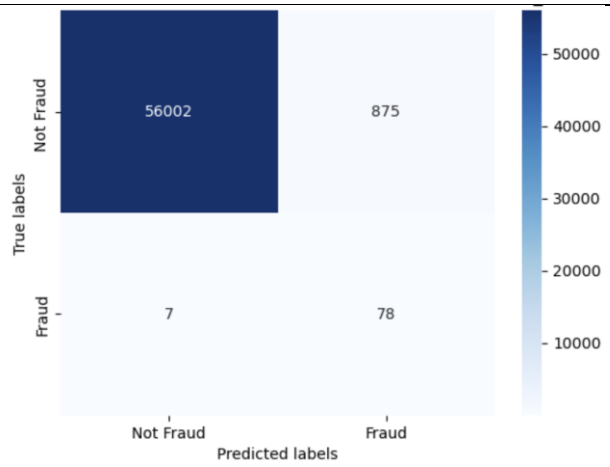
Accuracy	0.97
Balanced Accuracy	0.92
F1 Score	0.08
AUPRC	0.485



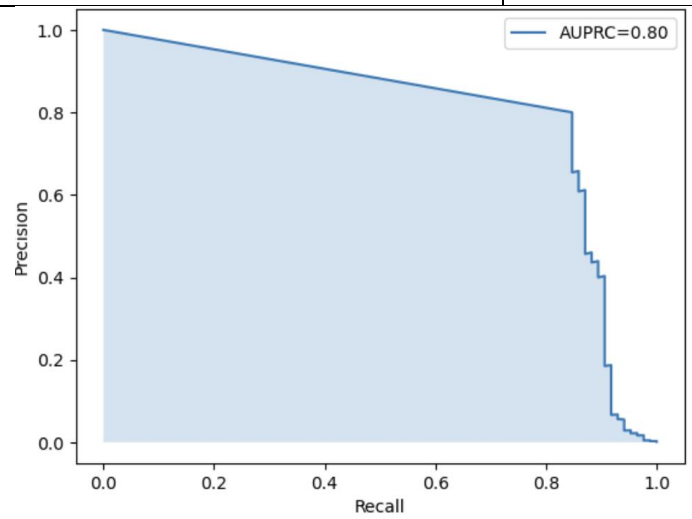
AUC	0.985



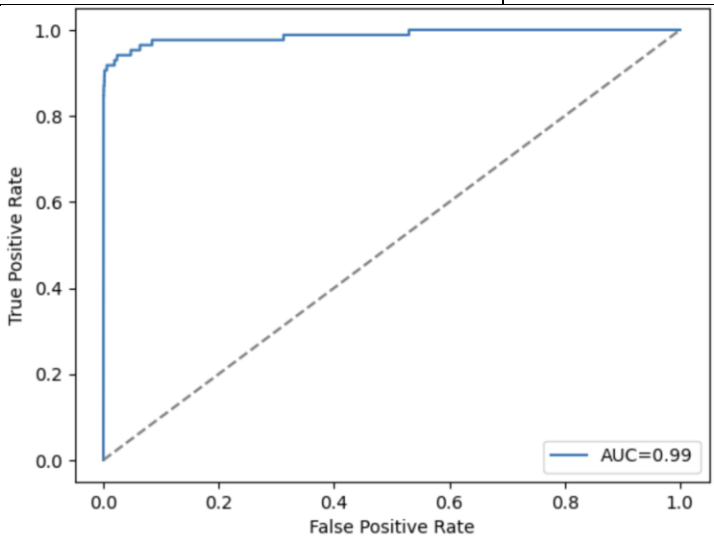
❖ SVM Classifier



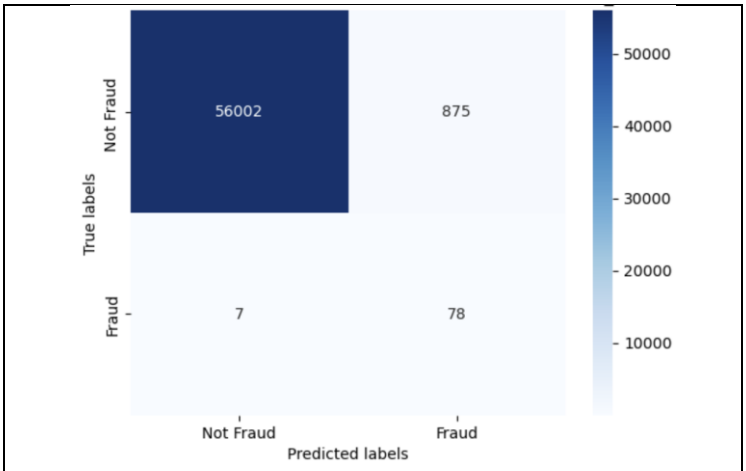
Accuracy	0.98
Balanced Accuracy	0.95
F1 Score	0.15
AUPRC	0.797



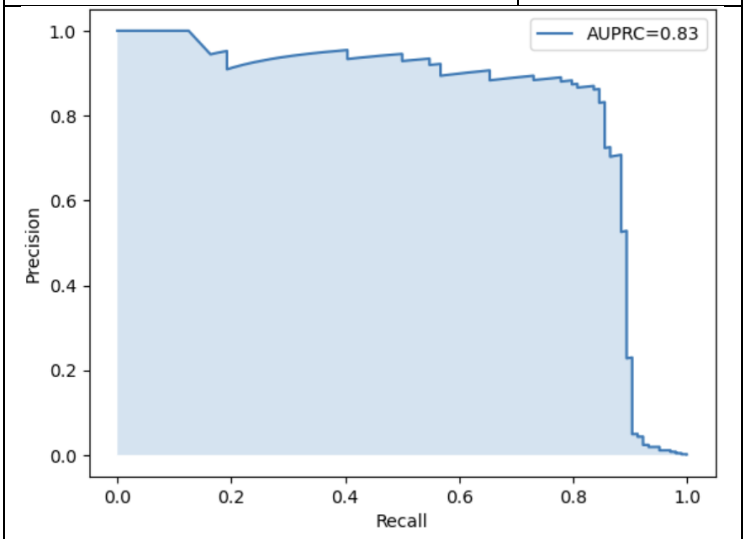
AUC	0.985



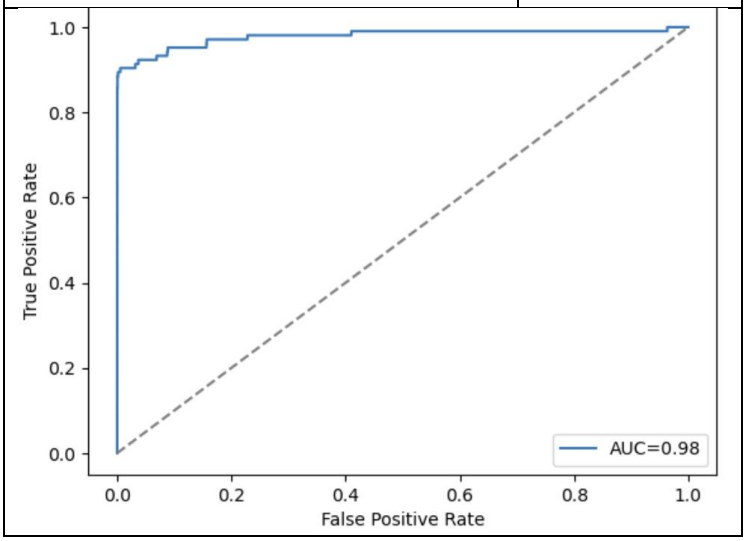
❖ Artificial Neural Network



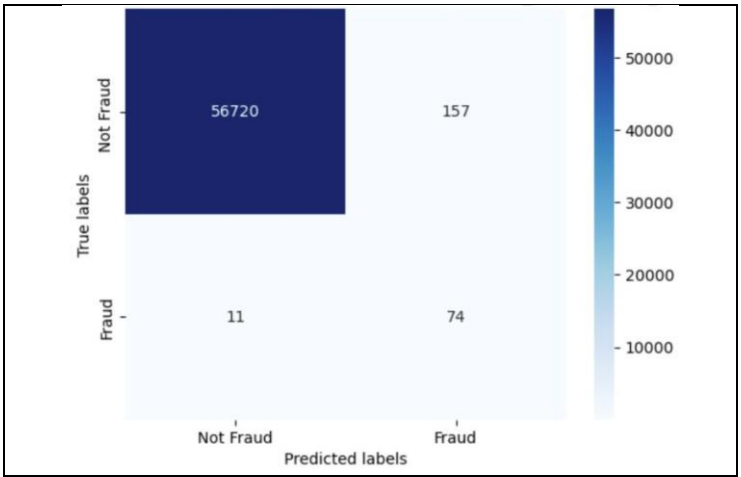
Accuracy	0.95
Balanced Accuracy	0.94
F1 Score	0.06
AUPRC	0.826



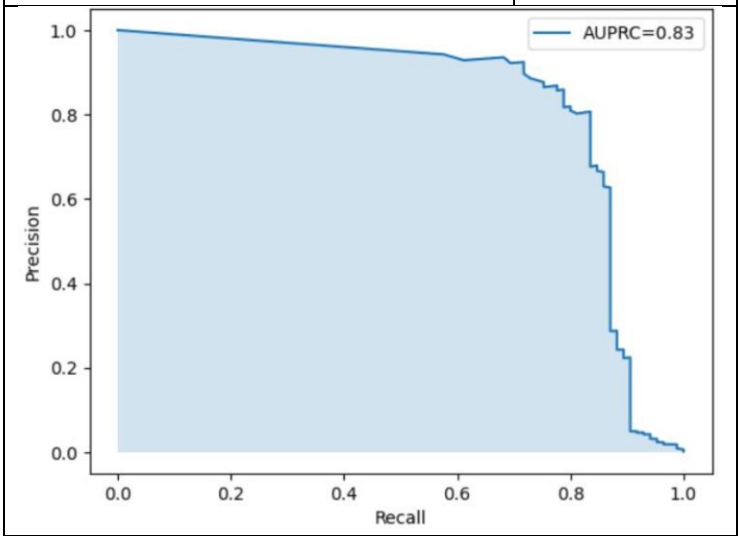
AUC	0.978



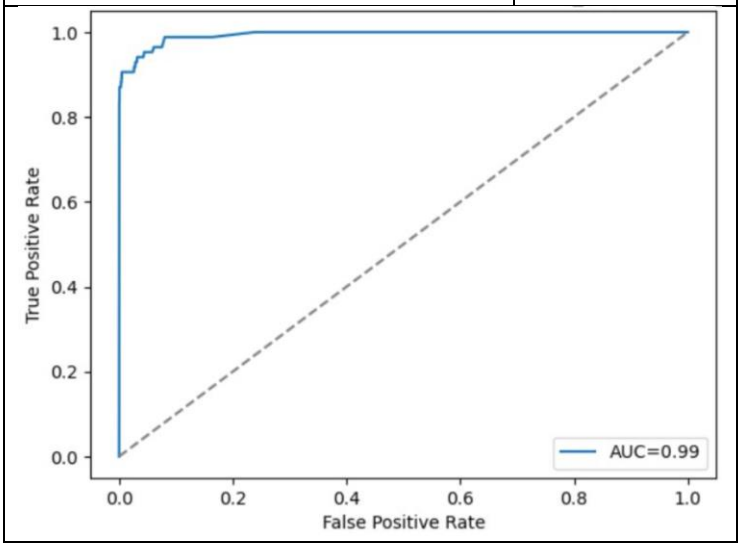
❖ Random Forest Classifier



Accuracy	1
Balanced Accuracy	0.93
F1 Score	0.47
AUPRC	0.826



AUC	0.993

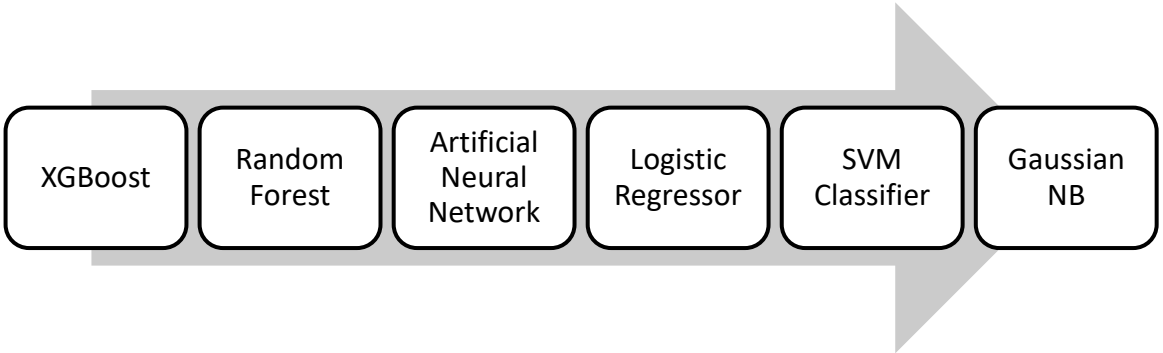


Inferences

The main criteria to judge the performances of the models turns out be the AUPRC as it successfully deals with the issue of highly unbalanced data by incorporating the values of Precision and Recall both simultaneously. AUPRC is a more suitable metric than Accuracy, F1 score or ROC-AUC score in highly skewed datasets since it accounts for the imbalance between the two classes and puts more weight on the minority class. So, the observation is:

Model	AUPRC
XGBoost	0.857
Logistic Regressor	0.807
Gaussian Naïve Bayes	0.485
ANN	0.826
SVC	0.797
Random Forest	0.826

It can be seen that XGBoost has the best performance, followed by equally performing ANN and RFC. But the RFC is more suitable for the task at hand as it is less computationally expensive. The logistic regressor, surprisingly has the next position in terms of performance. It exceeds SVC and obviously as had been expected, the GaussianNB has the worst performance.



Model Performances in decreasing order of AUPRC & computational efficiency

The reason behind the XGBoost and Random Forest can be attributed to their ensemble nature. The low bias and low variance of these models is also followed in case of highly skewed datasets. The ANN can deal with multiple samples and their skewness due to its complex nature leading to higher performance but is computationally expensive. The Regressor produces such results due to interpolation nature of SMOTE producing values that can be better analysed by the logistic regressor. SVC is not able to handle the skewness of the dataset as good as others, but still produces a great performance of around 80% AUPRC. The data is clearly not with

independent features and also has no multivariate-normalized distribution leading to poor performance.

It is also interesting to note that the F1-score gives very high values if the dataset that is used for training has small sample size in case of SVC but the AUPRC is not that high when compared to the ensembles such as XGBoost or Random Forest. In general, it is also seen that the smaller sample sizes produce good results compared to their larger counterparts considering the drastic reduce in processing times.

Summary of Process