# CREDIT CARD FRAUD DETECTION

##### Prepared By:

##### Arpan Avvari

##### DS861

##### Data Mining and Advanced Statistical Methods for Business Analysts



**Table of contents**

[CREDIT CARD FRAUD DETECTION 1](#_TOC_250009)

[Abstract 3](#_TOC_250008)

[Introduction 4](#_TOC_250007)

[Data Explanation 5](#_TOC_250006)

[Problem to Address 7](#_TOC_250005)

[Why is it Interesting and How is it relevant to class? 11](#_TOC_250004)

[Literature Review 12](#_TOC_250003)

[Techniques Used and Results 15](#_TOC_250002)

[Results 20](#_TOC_250001)

[Reference 25](#_TOC_250000)

## Abstract

Credit card fraud is a significant concern in today's society, with the rise in fraudulent transactions posing risks to individuals and institutions. This analysis focuses on a dataset obtained from Kaggle, containing credit card purchases made by European consumers in September 2013. The dataset includes both fraudulent and non-fraudulent transactions, with fraud accounting for only 0.172% of all transactions, resulting in a highly imbalanced dataset.

The goal is to detect and prevent fraudulent transactions by employing machine learning algorithms such as Logistic Regression, SVC, Decision Tree, and Random Forest. The analysis includes techniques like stratified sampling, correlation analysis, PCA for dimensionality reduction, and evaluation using the Area Under the Precision-Recall Curve (AUPRC) due to class imbalance. The project explores various aspects, including class distribution, transaction amount and time analysis, feature correlation, model accuracy, overfitting, learning curves, and ROC curve analysis. By addressing these aspects and employing appropriate techniques, the study aims to develop a robust credit card fraud detection system that can effectively identify and prevent fraudulent activities.

## Introduction

Credit cards have become an integral part of people's daily lives, facilitating both offline and online purchases of products and services. However, with the widespread use of credit cards, the incidence of credit card fraud has also been on the rise. Credit card theft poses a significant risk to individuals as well as corporate institutions. Fraudsters employ various methods, including stealing actual cards or obtaining sensitive account information, to carry out fraudulent transactions.

The consequences of credit card fraud are not limited to financial institutions but also impact consumers. The prevalence of identity fraud has been steadily increasing, with a significant rise observed in 2008. Globally, card fraud losses reached a staggering $21 billion in 2015, highlighting the urgency to combat this issue.

In this analysis, we are examining a dataset obtained from Kaggle, which focuses on credit card purchases made by consumers in Europe during September 2013. The dataset segregates purchases into two categories: fraudulent and non-fraudulent. By establishing correlations within these groups, we aim to identify suspicious transactions using machine learning algorithms. Specifically, we will employ Logistic Regression and Random Forest algorithms to detect anomalies and compare their performance to determine the more effective approach.

By conducting this analysis, we seek to enhance our understanding of credit card fraud patterns and develop robust methods for identifying and preventing fraudulent transactions. This research can contribute to the ongoing efforts to safeguard financial institutions and protect consumers from the detrimental impacts of credit card fraud.

## Data Explanation

The dataset at hand contains credit card transactions made by European cardholders in September 2013. It encompasses a two-day period and comprises a total of 284,807 transactions. Among these transactions, there are 492 instances of fraud, making the dataset highly unbalanced, with frauds accounting for only 0.172% of all transactions.

Principal Component Analysis (PCA) is a dimensionality reduction technique commonly used in data analysis and machine learning. It is applied to transform a set of potentially correlated variables into a new set of uncorrelated variables called principal components. The dataset primarily consists of numerical input variables, which have undergone a Principal Component Analysis (PCA) transformation. PCA is a dimensionality reduction technique used to convert potentially correlated variables into a new set of uncorrelated variables called principal components. The original features and additional background information about the data cannot be provided due to confidentiality issues. However, the transformed features resulting from the PCA process are labeled as V1, V2, ..., V28, representing the principal components obtained.

It's worth noting that the 'Time' and 'Amount' features have not undergone the PCA transformation and remain in their original form. The 'Time' feature represents the time elapsed in seconds between each transaction and the first transaction recorded in the dataset. The 'Amount' feature denotes the monetary value of each transaction. These two features provide additional information that can be used in the analysis.

The response variable in the dataset is the 'Class' feature, which takes the value 1 for cases of fraud and 0 for legitimate transactions.

Given the highly imbalanced class distribution, it is recommended to evaluate the model's performance using the Area Under the Precision-Recall Curve (AUPRC) rather than traditional

accuracy metrics based on the confusion matrix. AUPRC provides a more accurate assessment of the model's ability to detect fraud in an unbalanced classification scenario.



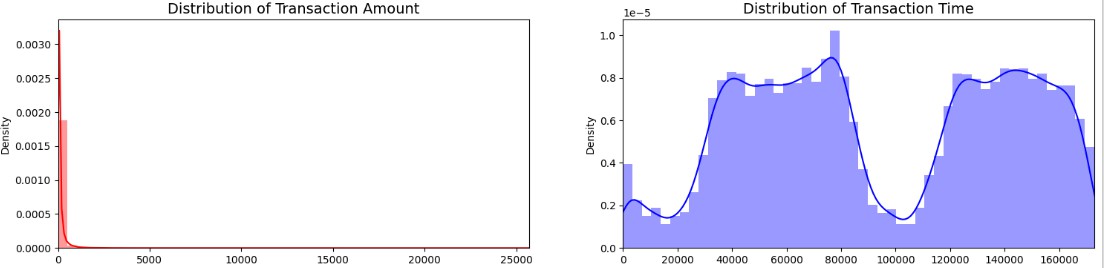
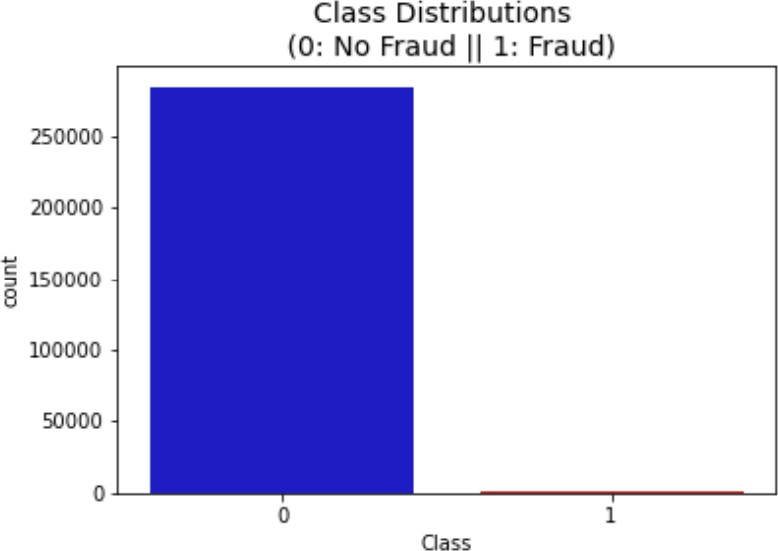
In summary, the dataset contains anonymized credit card transaction data, with most of the numerical features transformed through PCA, resulting in the principal components V1 to V28. The 'Time' and 'Amount' features retain their original values. The 'Class' feature serves as the response variable, indicating fraud (1) or legitimate transaction (0). Evaluating the model's performance using AUPRC is recommended due to the class imbalance, as it provides a more meaningful metric for unbalanced classification problems.



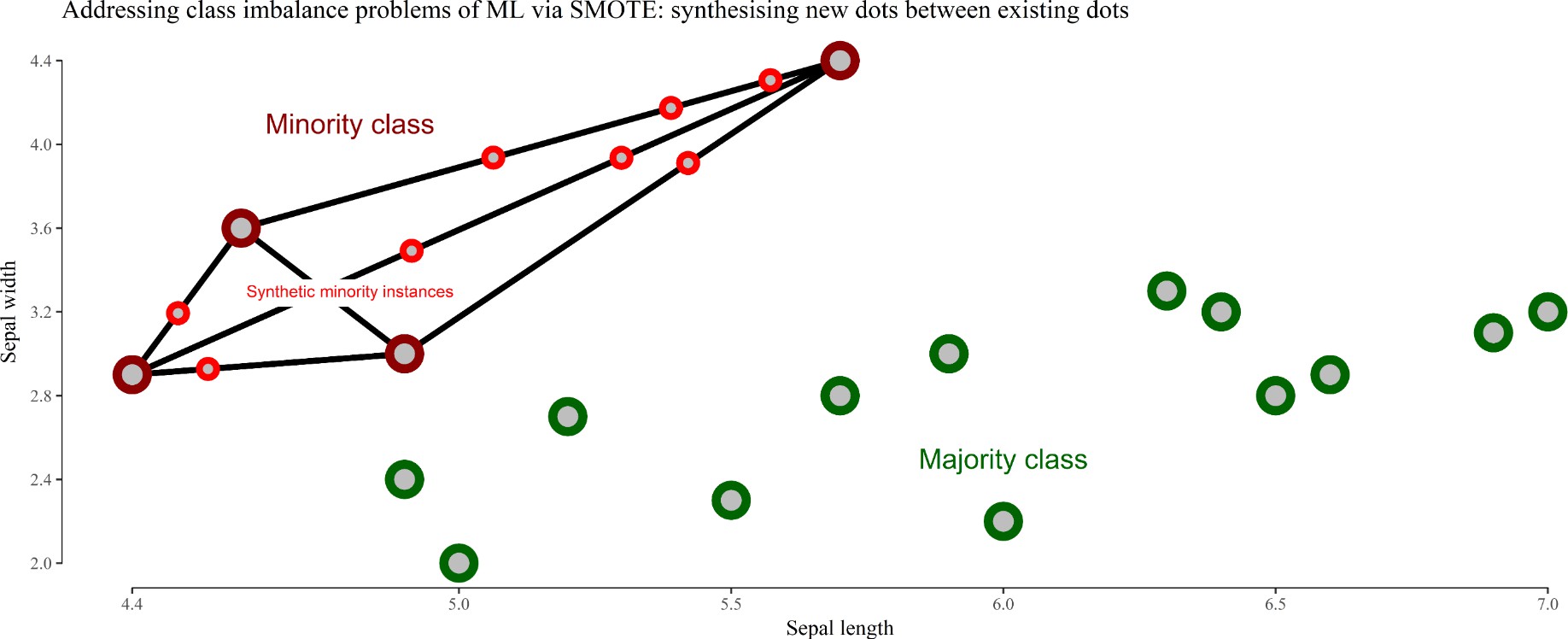
#### Problem to Address

The problem we want to address is the detection of fraudulent transactions in a dataset. This is a common problem in the field of fraud detection, where the goal is to accurately identify transactions that are likely to be fraudulent and distinguish them from legitimate transactions. The dataset used for this problem contains various features related to each transaction, such as transaction amount, timestamp, and other relevant information.

The challenge in this problem is that fraudulent transactions are typically rare compared to legitimate transactions, resulting in an imbalanced dataset. This class imbalance makes it difficult for traditional classification algorithms to effectively learn and classify fraudulent transactions.



Therefore, specific techniques like undersampling and oversampling, such as Random UnderSampling and SMOTE, are applied to balance the classes and improve the performance of the classifiers.



The final effect is, more information is retained since we didn't have to delete any rows unlike in random undersampling. Although it is likely that SMOTE will be more accurate than random under-sampling, it will take more time to train since no rows are eliminated as previously stated. But we realized that SMOTE oversampling is used when we need more data for our dataset in cases where there is missing data for one of the classes or we want to create a perfect class population in the dataset. Therefore, we don’t need to implement SMOTE technique to our dataset as the ratio of the classes should be maintained in the training, cross-validation, and test set. This ratio is vital while training our machine learning models.

Stratified sampling is a sampling method used when a population can be naturally divided into distinct, non-overlapping subgroups or strata. This method ensures that each stratum is represented in the final sample in proportions that reflect its size within the population.

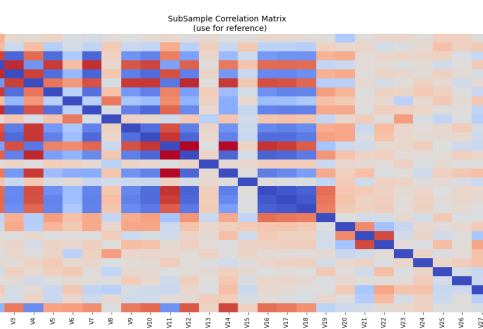
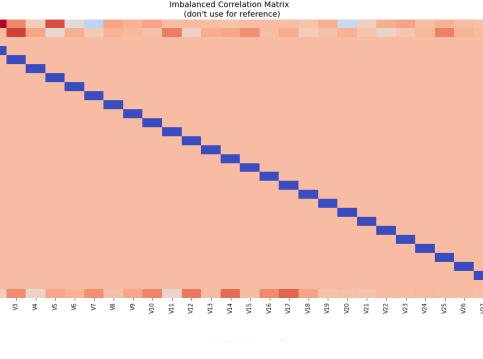
To determine the sample size for each stratum, you can use the following formula: S = (S/N) × n

This formula calculates the sample size for a single stratum based on its proportionate size in the population. Alternatively, if you know the percentage (p%) of the total population that belongs to a single stratum, you can calculate the sample size for that stratum using the following formula:

Sample size for a stratum = (p% × n)

By multiplying the percentage of the population in the stratum by the desired overall sample size, you can determine the sample size for that specific stratum. After calculating the sample sizes for each stratum, random samples are taken independently from each stratum, and these samples are combined to form an overall representative sample of the population.

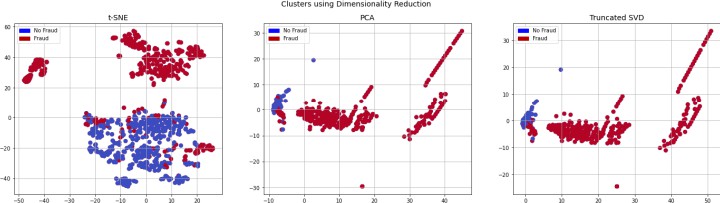
An imbalanced dataset can present challenges when trying to uncover important correlations between features and the target variable. In cases where one class is heavily underrepresented compared to the other, traditional correlation measures may not accurately capture the relationships between features and the target. To address this issue, using a balanced subsample that accurately represents the distribution of the target variable can provide a more reliable reference for analyzing feature-target correlations.



By creating a subsample where both classes are adequately represented, the impact of class imbalance on the correlation analysis can be mitigated. Visualizing the correlation using a

heatmap in this balanced subsample allows for a clearer understanding of the relationship in the context of fraud detection. The heatmap enables the identification of patterns and strengths of correlation between individual features and the target variable, leading to a more accurate comprehension of the factors contributing to fraud detection. When working with imbalanced datasets, it is essential to employ appropriate techniques to address the class imbalance, such as oversampling, undersampling, or synthetic sampling methods like SMOTE. These techniques help improve the representation of the minority class and reduce bias in the correlation analysis.

The ultimate objective is to develop a model that can accurately detect fraudulent transactions, minimizing both false positives (legitimate transactions incorrectly classified as fraudulent) and false negatives (fraudulent transactions incorrectly classified as legitimate). This will help financial institutions and businesses mitigate losses due to fraudulent activities and enhance their security measures.



t-SNE algorithm can pretty accurately cluster the cases that were fraud and non-fraud in our dataset. Although the subsample is small, the t-SNE algorithm is able to detect clusters in every scenario. This gives us an indication that further predictive models will perform pretty well in separating fraud cases from non-fraud cases.

## Why is it Interesting and How is it relevant to class?

Fraud detection is crucial in various industries, especially finance. Accurately identifying fraudulent transactions has significant implications, preventing financial losses, protecting customers, maintaining trust, and ensuring compliance. Studying fraud detection in our class is relevant for several reasons:

1. Practical Application: Fraud detection techniques are used in real-world scenarios, essential for careers in data science, machine learning, or finance.
2. Imbalanced Data Handling: Fraud detection addresses imbalanced datasets, providing opportunities to learn techniques like undersampling, oversampling, and ensemble methods.
3. Feature Engineering: Fraud detection involves extracting meaningful features from complex datasets, and providing insights into feature selection, transformation, and engineering techniques.
4. Ethical Considerations: Fraud detection raises ethical considerations such as privacy, bias, and fairness, expanding students' understanding of social implications and ethical responsibilities in data analysis.

By studying fraud detection, students gain valuable knowledge applicable to data analysis and machine learning, reinforcing core concepts in our class.

## Literature Review

The problem of credit card fraud detection has been extensively studied in the literature, and various approaches and techniques have been proposed to address this issue. Researchers have focused on developing effective methods to detect fraudulent transactions and minimize the impact of such activities on individuals and organizations. Here is a brief overview of what has been said and done in the literature regarding credit card fraud detection:

Several traditional machine learning algorithms, such as Decision Trees, Random Forests, Support Vector Machines (SVM), and Logistic Regression, have been applied to credit card fraud detection. These methods utilize features extracted from transaction data to train models and make predictions about fraudulent transactions. Researchers have explored different combinations of features, feature selection methods, and model architectures to improve the accuracy of fraud detection. Authors continue to go over feature selection, on how they play a crucial role in credit card fraud detection, as it helps identify the most relevant attributes that contribute to fraud patterns. Various feature selection techniques, such as Genetic Algorithms, Information Gain, Principal Component Analysis (PCA), and Recursive Feature Elimination, have been employed to select the most informative features and reduce the dimensionality of the data. By selecting a subset of relevant features, these methods aim to improve the efficiency and effectiveness of fraud detection models.

Ensemble methods, such as Random Forests, Boosting, and Bagging, have been widely used in credit card fraud detection. These methods combine multiple base models to make collective predictions, leveraging the diversity and complementary strengths of individual models. Ensemble methods have been shown to enhance the robustness and generalization capability of fraud detection systems, improving the detection accuracy and reducing false

positives. With the advancements in deep learning, researchers have explored the application of neural network architectures, such as Convolutional Neural Networks (CNNs), Recurrent Neural Networks (RNNs), and Long Short-Term Memory (LSTM) networks, for credit card fraud detection. Deep learning models have demonstrated the ability to automatically learn complex patterns and representations from raw transaction data, leading to improved fraud detection performance.

These methods focus on capturing deviations from normal behavior and flagging suspicious transactions for further investigation.

Some studies in the article have proposed hybrid approaches that combine multiple techniques to enhance fraud detection performance. For example, combining traditional machine learning algorithms with anomaly detection methods or integrating deep learning models with ensemble techniques. These hybrid approaches aim to leverage the strengths of different methods and improve the accuracy and efficiency of fraud detection systems. As Credit card fraud datasets are highly imbalanced, with a large number of legitimate transactions compared to a small fraction of fraudulent transactions. Researchers have addressed thisue by applying various techniques, such as oversampling the minority class, undersampling the majority class, and using cost-sensitive learning algorithms. These methods aim to mitigate the imbalance problem and ensure that fraud detection models can effectively learn from the limited fraudulent instances.

In evaluating the performance of credit card fraud detection models, researchers have used metrics such as accuracy, precision, recall, F1-score, and Area Under the Curve (AUC) of the Receiver Operating Characteristic (ROC) curve. These metrics provide insights into the

effectiveness and efficiency of the models in detecting fraudulent transactions and differentiating them from legitimate ones.

Overall, the literature on credit card fraud detection emphasizes the importance of developing accurate and efficient models that can effectively detect fraudulent activities while minimizing false positives. Researchers are incorporating advancements in machine learning and deep learning, as well as leveraging new data sources and technologies, to further improve credit card fraud detection systems.

## Techniques Used and Results

Aspects, Techniques, and Key Learnings from Credit Card Fraud Detection Project

The credit card fraud detection project focused on developing an effective solution through the exploration of various aspects of the dataset and the application of appropriate techniques. The key aspects examined in the project included class distribution analysis, transaction amount and time analysis, correlation analysis, dimensionality reduction with PCA, accuracy score calculation, overfitting evaluation, learning curve analysis, stratified KFold cross-validation, ROC curve analysis, and comparison of classification models.

Class distribution analysis was performed to understand the imbalance between fraudulent and non-fraudulent transactions, visualizing the distribution of the target variable. Transaction amount and time analysis involved studying the patterns and ranges of transaction values, as well as identifying specific time periods with increased transaction activity. Correlation analysis helped uncover significant relationships between different features in the dataset, potentially indicating fraudulent activity.

Dimensionality reduction techniques, specifically PCA, were employed to reduce the number of variables while retaining most of the information. Accuracy scores were calculated to assess the precision of classification models in distinguishing between fraudulent and non-fraudulent transactions. Overfitting evaluation ensured that the models could generalize well to unseen data.

Learning curve analysis provided insights into model performance with varying training sample sizes, while stratified KFold cross-validation ensured reliable performance estimates by maintaining a similar distribution of target classes in each fold. The effectiveness of

classification models was evaluated using the ROC curve, which depicted true positive and false positive rates.

Furthermore, different classification models such as random forest, gradient boosting trees (GBT), logistic regression, decision trees, and support vector classifiers were compared for their performance in credit card fraud detection.

The key learnings from the project encompassed understanding the dataset's imbalance, recognizing transaction amount and time patterns, utilizing dimensionality reduction techniques like PCA, the significance of accuracy scores for model evaluation, the need to address overfitting, insights gained from learning curves, the effectiveness of the ROC curve, and the importance of comparing different classification algorithms.

In conclusion, the project aimed to develop a robust credit card fraud detection system by exploring these aspects and employing suitable techniques.

Hyperparameters

Logistic Regression: The training and cross-validation scores converge and plateau as the training size increases. This suggests that the model is not overfitting or underfitting the data and is performing consistently.

K-Nearest Neighbors: The training score decreases slightly as the training size increases, indicating that the model becomes more sensitive to noise. The cross-validation score improves with more data but remains lower than the training score, suggesting some degree of overfitting. Support Vector Classifier: The training and cross-validation scores converge and plateau at a high level, indicating a good fit to the data and generalization capability.

Decision Tree Classifier: The training score remains high, but the cross-validation score is lower and does not improve significantly with more data. This suggests overfitting and poor generalization.

Random Forest Classifier: The training score is high and increases slightly with more data. The cross-validation score also improves with more data, indicating good generalization.

In summary, these learning curve plots indicate that the KNN, logistic regression, SVC, random forest, and GBT models perform well in credit card fraud detection, showing high training accuracy and stable or improving validation accuracy. The decision tree model also performs well but may have slightly lower generalization capabilities compared to the other models.

*Logistic regression:*

“penalty”: The regularization penalty to be used. This can take on the values of 'l1' for L1 regularization, or 'l2' for L2 regularization. L1 regularization encourages sparse coefficient values (i.e., some coefficients are exactly zero), while L2 regularization encourages small coefficient values (but none are exactly zero).

“C”: The inverse regularization strength. Smaller values of C correspond to stronger regularization, and larger values correspond to weaker regularization. The values specified in the dictionary range from 0.001 to 1000, with larger values indicating weaker regularization.



*SVC :*

“C”: The C hyperparameter in SVC controls the trade-off between the complexity of the decision boundary and the degree of misclassification in the training data.

“kernel”: The kernel parameter in SVC specifies the type of kernel function used to transform the input data into a higher-dimensional space. The kernel function essentially measures the

similarity between pairs of data points in the transformed space, allowing SVC to find a decision boundary that separates the classes.



*KNN:*

“n\_neighbors”: The number of neighbors to consider when making a prediction. This hyperparameter controls the level of model complexity and can affect both the bias and variance of the model. In this case, the values specified in the dictionary range from 2 to 4, with step 1, meaning that the number of neighbors considered will be 2, 3, and 4.

“algorithm”: The algorithm used to compute the nearest neighbors. There are several options available in the sci-kit-learn library, including auto, ball\_tree, kd\_tree, and brute. These different algorithms have different time and space complexity and may perform differently depending on the characteristics of the data.



*Decision Tree Classifier:*

"criterion": It specifies the measure used for the quality of a split, either "gini" or "entropy". "max\_depth": It sets the maximum depth of the decision tree, controlling the complexity of the model.

"min\_samples\_leaf": It sets the minimum number of samples required to be at a leaf node.

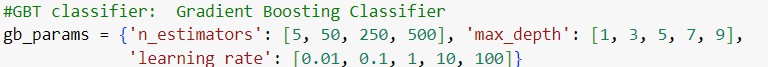


*Gradient Boosting Classifier (GBT):*

"\_estimators": It determines the number of weak decision tree models to be trained.

"max\_depth": It sets the maximum depth of each weak decision tree.

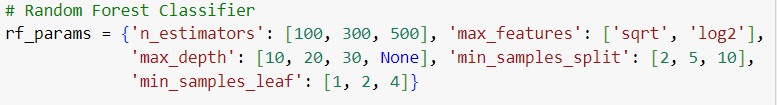
"learning rate": It controls the contribution of each weak tree to the final prediction.



*Random Forest Classifier:*

"\_estimators": It specifies the number of decision trees to be included in the random forest. "max\_features": It determines the number of features to consider when looking for the best split. "max\_depth": It sets the maximum depth of each decision tree in the random forest. "min\_samples\_split": It sets the minimum number of samples required to split an internal node. "min\_samples\_leaf": It sets the minimum number of samples required to be at a leaf node.

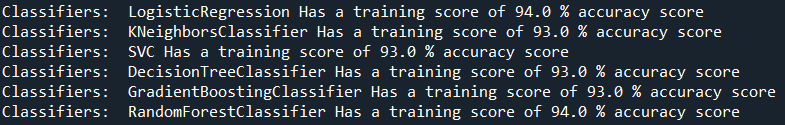
Adjusting these hyperparameters can significantly impact the model's performance. Finding the optimal values often involves experimentation and model evaluation using techniques like cross-validation. The goal is to select hyperparameter values that result in a well-performing model, avoiding overfitting (the model being too complex and fitting noise) or underfitting (the model being too simple and unable to capture the underlying patterns).



### Results

Training / Validation Set Data:

Accuracy Score:

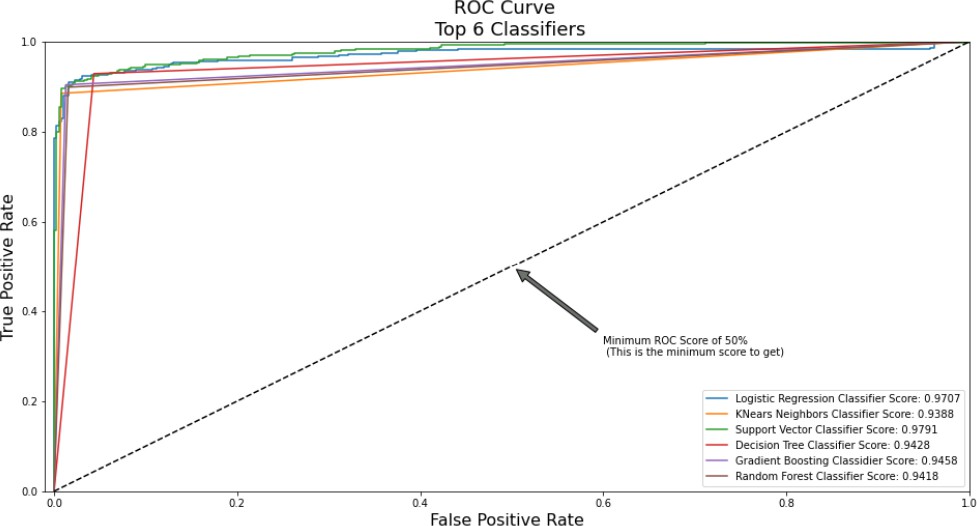


These scores indicate the performance of each classifier on the training set and during cross-validation. The cross-validation scores are generally higher than the training scores, which suggests that the models are performing well and not overfitting the training data. However, it's important to note that these scores alone do not provide a complete picture of the model's performance, and other metrics such as precision, recall, and F1-score should be considered as well. Additionally, the choice of the best classifier depends on the specific problem and dataset being used.

Scores are getting even high scores even when applying cross-validation.

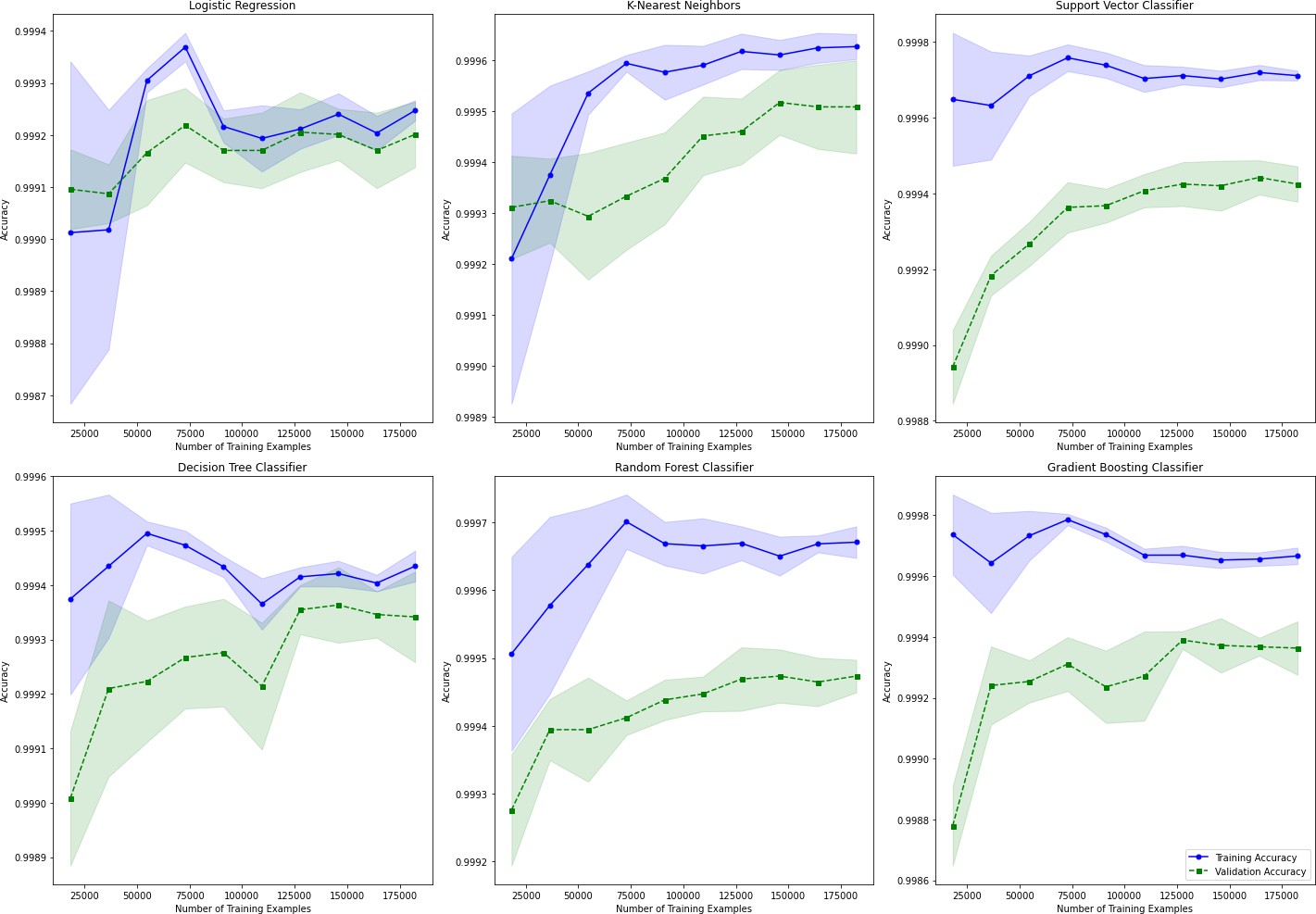
ROC Curve

A common evaluation metric for binary classification issues is the ROC curve. They demonstrate the trade-off between the true positive rate and the false positive rate at various classification thresholds.



Learning Curves

Learning curves help us understand how the model's performance improves as we increase the training size. They show the training and cross-validation scores as a function of the number of training instances.



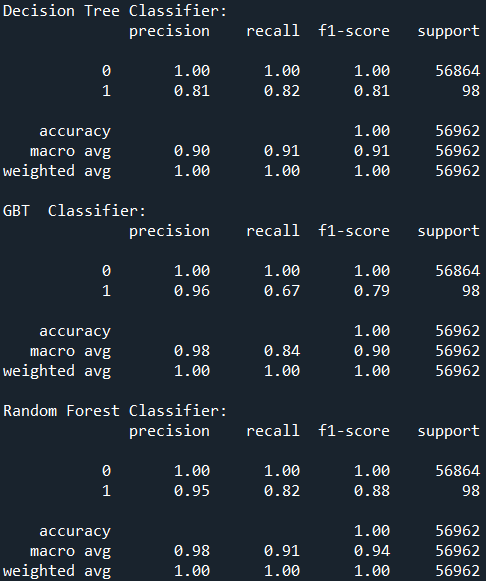
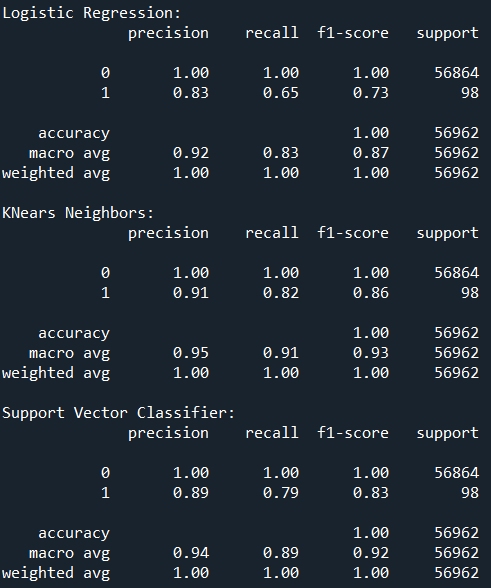
From the learning curves, we can observe the following trends:

* Logistic Regression: The training and cross-validation scores converge and plateau as the training size increases. This suggests that the model is not overfitting or underfitting the data and is performing consistently.
* K-Nearest Neighbors: The training score decreases slightly as the training size increases, indicating that the model becomes more sensitive to noise. The cross-validation score improves with more data but remains lower than the training score, suggesting some degree of overfitting.
* Support Vector Classifier: The training and cross-validation scores converge and plateau at a high level, indicating a good fit to the data and generalization capability.
* Decision Tree Classifier: The training score remains high, but the cross-validation score is lower and does not improve significantly with more data. This suggests overfitting and poor generalization.
* Random Forest Classifier: The training score is high and increases slightly with more data. The cross-validation score also improves with more data, indicating good generalization.
* Gradient Boosting: The GBT model demonstrates strong performance, with high training accuracy and stable validation accuracy.

In summary, these learning curve plots indicate that the KNN, logistic regression, SVC, random forest, and GBT models perform well in credit card fraud detection, showing high training accuracy and stable or improving validation accuracy. The decision tree model also performs well but may have slightly lower generalization capabilities compared to the other models.

###### Test Set Data

Classification Model Performance Metrics



The above values represent the performance metrics of different classification models on a dataset. Let's go through the key metrics and what they mean:

1. Precision: Precision measures the proportion of true positive predictions (1s predicted correctly) out of all positive predictions (both true positives and false positives). Higher precision indicates a lower false positive rate.
2. Recall: Recall, also known as sensitivity or true positive rate, measures the proportion of true positives predicted correctly out of all actual positive instances. Higher recall indicates a lower false negative rate.
3. F1-score: The F1-score is the harmonic mean of precision and recall. It provides a single metric that combines both precision and recall, with a higher value indicating a better balance between the two.
4. Support: Support represents the number of instances of each class in the dataset.
5. Accuracy: Accuracy is the overall correctness of the model, calculated as the ratio of correct predictions to the total number of predictions. Higher accuracy indicates a better-performing model.

The "weighted avg" and "macro avg" values are calculated by averaging the metrics for each class, weighted by the number of instances in each class. The weighted average gives more weight to the larger class, while the macro average treats all classes equally.

Random Forest Classifier model demonstrated the highest overall performance by achieving the macro avg of 0.94. This suggests that the Random Forest Classifier is effective in accurately predicting the fraud. Whereas, Logistic Regression achieved a macro avg of 0.87 which is less compared to the other five models.

### Reference

1. Emmanuel.I, Yanxia.S & Zenghui.W. Feb 22, 2022. A machine learning-based credit card fraud detection using the GA algorithm for feature selection.
2. <https://journalofbigdata.springeropen.com/articles/10.1186/s40537-022-00573-8>
3. <https://docs.python.org/>
4. https://scikit-learn.org/stable/

### Data Used

1. [https://www.kaggle.com/datasets/mlg-ulb/creditcardfraud](http://www.kaggle.com/datasets/mlg-ulb/creditcardfraud)