**Project Title: Enhancing Fraud Detection Using Synthetic Transactions Generated by CTGAN**

**Submitted By**

Aditi Bhoir

**Internship Organization**

GLOBAL NEXT CONSULTING INDIA PRIVATE LIMITED (GNCIPL)

**Date of Submission**

5th September 2025

**Executive Summary**

Credit card fraud detection suffers from extreme class imbalance, with fraud cases representing only 0.17% of the dataset. Traditional machine learning models struggle to detect rare fraud instances, resulting in poor recall and performance.  
This project leverages CTGAN (Conditional Tabular Generative Adversarial Network) to generate synthetic fraud transactions, augment the dataset, and retrain machine learning models to improve performance.

The final results show a significant improvement in recall (from 0.41 to 0.67), F1-score (from 0.54 to 0.74), and AUC (from 0.76 to 0.89). This enhances the model’s ability to detect rare fraudulent transactions while preserving customer privacy.

**1. Project Objectives**

* Improve fraud detection accuracy and recall.
* Address class imbalance by augmenting the minority class using synthetic data.
* Maintain customer data privacy by not exposing real transaction records.

**2. Data Overview**

* **Source:** Kaggle Credit Card Fraud Dataset (~285,000 transactions, 30 features).
* **Target Column:** Class (0 = normal transaction, 1 = fraud).
* **Preprocessing Steps:**
  + Normalized Amount and Time columns.
  + Removed duplicate records.
  + Checked for missing values (none found).
* **Key Issue:** Only 492 fraud examples (0.17%), making it challenging for models to generalize.

**3. Methodology**

**3.1 CTGAN Synthetic Data Generation**

Using the SDV (Synthetic Data Vault) library, we trained a CTGAN model on the fraud data to generate synthetic samples:

from sdv.tabular import CTGAN

ctgan = CTGAN(epochs=300)

ctgan.fit(real\_fraud\_data) # real\_fraud\_data contains only fraud transactions

synthetic\_data = ctgan.sample(5000) # Generate 5,000 synthetic fraud samples

**3.2 Data Augmentation & Model Training**

* Combined the original dataset with generated synthetic fraud samples.
* Split the data into training and testing sets (70%-30% split).
* Trained a Random Forest Classifier on the augmented data:

from sklearn.ensemble import RandomForestClassifier

clf = RandomForestClassifier()

clf.fit(X\_train, y\_train)

**3.3 Model Evaluation Metrics**

| **Model** | **Precision** | **Recall** | **F1-Score** | **AUC** |
| --- | --- | --- | --- | --- |
| Real Data Only | 0.80 | 0.41 | 0.54 | 0.76 |
| Augmented Data (CTGAN) | 0.84 | 0.67 | 0.74 | 0.89 |

**3.4 Visualizations**

* **t-SNE / PCA Plots**: Showed the distribution overlap between real and synthetic fraud data.
* **ROC Curve**: Demonstrated improved Area Under Curve (AUC).
* **Confusion Matrix**: Indicated higher True Positive Rate after augmentation.
* **Precision-Recall Curve**: Validated improved performance under skewed class distribution.

**4. Business Impact**

* **Enhanced Fraud Detection**: Detects rare fraud events more accurately.
* **Cost Savings**: Reduces potential financial losses from undetected fraud.
* **Privacy Compliance**: No real customer data is exposed during training or testing.
* **Scalability**: Synthetic data generation allows for scalable augmentation as needed.

**5. Risks & Limitations**

| **Risk/Challenge** | **Mitigation Strategy** |
| --- | --- |
| Synthetic Overfitting | Validate with t-SNE and use a separate test set. |
| Bias | Monitor fairness metrics during training. |
| Generalization Limit | Carefully balance real and synthetic data. |

**6. Conclusion & Recommendations**

This project demonstrates that Generative AI-powered data augmentation using CTGAN significantly improves the detection of rare fraudulent transactions.  
The model’s recall improved by 33%, and the overall F1-score increased by 20%.  
We recommend integrating synthetic data pipelines into regular ML workflows, retraining models periodically, and using explainability tools like SHAP to analyze decision behavior.

**Appendix**

**Key Libraries Used**

* pandas, numpy, scikit-learn
* sdv.tabular.CTGAN
* matplotlib, seaborn

**Sample Code Snippet**

from sdv.tabular import CTGAN

ctgan = CTGAN(epochs=300)

ctgan.fit(real\_fraud\_data)

synthetic\_data = ctgan.sample(5000)

**Files Included**

* EDA Report (PDF)
* Augmented Dataset (CSV)
* Trained Model (.pkl)
* Evaluation Notebook (.ipynb)
* PowerPoint Presentation (.pptx)