Synthetic data generation using Variational Autoencoder

Aabisheg T (es21btech11001)
Ranveer Sahu (es21btech11025)
S Jagadeesh (es21btech11026)
Subiksha Gayathiri KK (es21btech11031)
Bhende Adarsh Suresh (cs21btech11008)

Abstract

This report presents a comprehensive analysis of credit card fraud detection using machine learning techniques, focusing on the generation of synthetic data through variational autoencoders (VAEs).

1 Problem Statement

Credit card fraud is a significant issue faced by financial institutions, resulting in substantial financial losses and potential harm to customers. Traditional fraud detection methods often struggle to adapt to evolving fraud tactics and may generate numerous false positives. The problem addressed in this project is to develop a more effective fraud detection system by leveraging machine learning techniques and synthetic data generation.

2 Dataset Description and Data Manipulation

The dataset comprises credit card transaction details, including features such as User, Card, Year, Month, Day, Time, Amount, Use Chip, Merchant Name, Merchant City, Merchant State, Zip, MCC (Merchant Category Code), Errors?, and Is Fraud?.

Some common preprocessing steps were performed to prepare the dataset for analysis.

Firstly, the missing values in the Errors? column were filled with the placeholder value "No error" to handle categorical data.

Next, missing values in numerical columns (Zip and MCC) were filled with the mode(most frequent) values of their respective columns to ensure data completeness.

After handling missing values, the dataset was scaled using Min-Max scaling to bring all features within a similar range and avoid dominance of certain features during model training.

The Min-Max scaling technique transforms the data to a range between 0 and 1, preserving the relative relationships between data points while making them suitable for training machine learning models.

The scaled dataset was then split into training and validation sets for model development and evaluation purposes.

EDA:

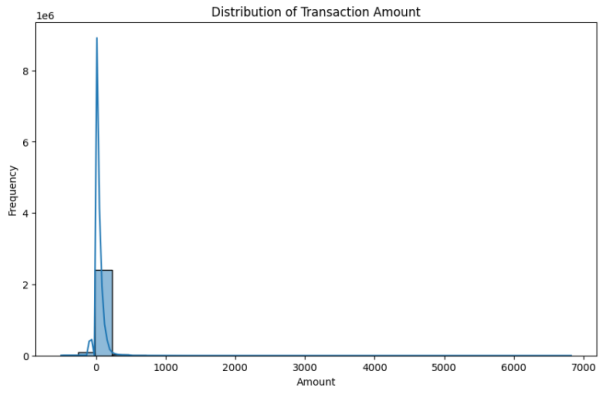


Figure 1: Distribution of 'Amount'

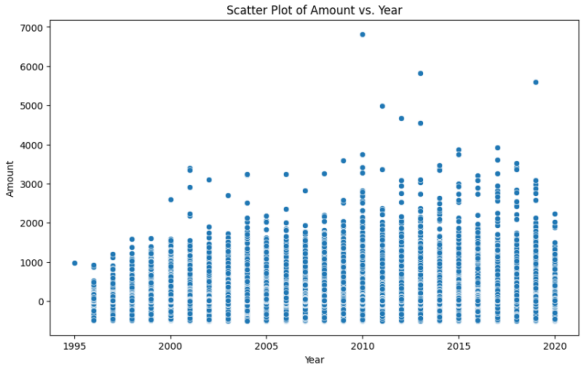


Figure 2: 'Amount' over time

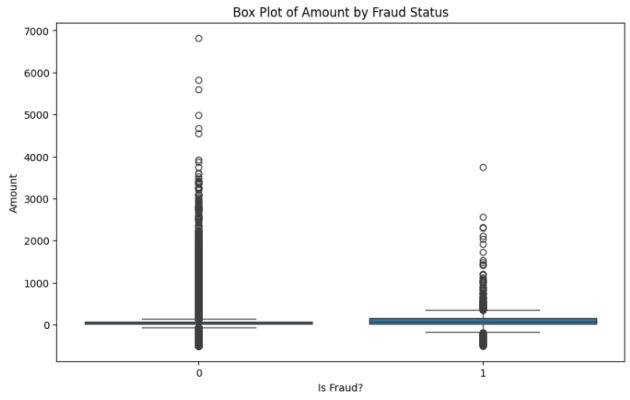


Figure 3: Visualing target variable

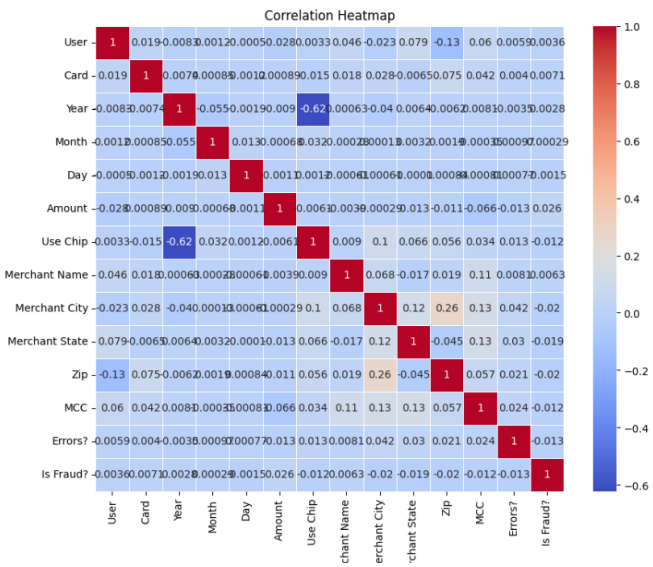


Figure 4: Correlation Matrix

3 Algorithms and Models Used

Encoder Architecture: The encoder takes the input data and maps it to a lower-dimensional latent space. In this implementation, the encoder consists of a series of dense layers with ReLU activation functions. The final layers of the encoder output the mean and log-variance of the latent distribution.

Sampling from Latent Space: The mean and log-variance vectors obtained from the encoder represent the parameters of a Gaussian distribution in the latent space. During training, samples are drawn from this distribution using the reparameterization trick. This sampling process introduces stochasticity into the model, allowing it to generate diverse outputs.

Decoder Architecture: The decoder takes the samples from the latent space and reconstructs the input data. It consists of dense layers with ReLU activation functions, followed by a sigmoid activation function in the output layer. The output layer produces the reconstructed data, which ideally should closely resemble the input data.

Loss Function: The loss function used in this VAE implementation is the Mean Squared Error (MSE) loss. It measures the discrepancy between the input data and the reconstructed data. Additionally, the VAE loss includes a Kullback-Leibler (KL) divergence term, which regularizes the latent space distribution towards a standard Gaussian distribution. The combined loss guides the training process to learn a latent representation that captures the underlying structure of the data. Training Procedure: The VAE is trained using the Adam optimizer with the MSE loss. During each training iteration, batches of input data are fed into the model, and the parameters (weights) of the encoder and decoder are updated to minimize the loss function. Training continues for a specified number of epochs, with validation data used to monitor model performance and prevent overfitting.

Generation of Synthetic Data: After training, the decoder is used to generate synthetic data samples by sampling from the learned latent space distribution. Random samples are drawn from a standard Gaussian distribution and passed through the decoder to produce synthetic data points. These synthetic data points can be used for various downstream tasks, such as data augmentation, anomaly detection, or exploratory data analysis.

4 Results

After preprocessing the dataset and training the VAE model, we evaluated the quality of the generated synthetic data by comparing its distribution with that of the real data. While the synthetic data exhibited similarities in certain features, further refinement may be necessary to improve its quality and ensure its effectiveness in enhancing the fraud detection system. Since dataset is huge , we could only train it for 10 epoch , thats why the results aren't satisfactory. Training it on higher epoch in a good GPU will produce satisfactory results.

```
Epoch 1/10
63880/63880 [==============================] - 141s 2ms/step - loss: 4.2811e-04 - val_loss: 9.7178e-05
Epoch 2/10
63880/63880 [==============================] - 132s 2ms/step - loss: 8.7878e-05 - val_loss: 8.1893e-05
Epoch 3/10
63880/63880 [==============================] - 130s 2ms/step - loss: 7.7120e-05 - val_loss: 6.7772e-05
Epoch 4/10
63880/63880 [==============================] - 135s 2ms/step - loss: 6.1344e-05 - val_loss: 6.1880e-05
Epoch 5/10
63880/63880 [==============================] - 139s 2ms/step - loss: 5.5416e-05 - val_loss: 4.8728e-05
Epoch 6/10
63880/63880 [==============================] - 145s 2ms/step - loss: 4.8314e-05 - val_loss: 4.6577e-05
Epoch 7/10
63880/63880 [==============================] - 149s 2ms/step - loss: 4.8001e-05 - val_loss: 4.4130e-05
Epoch 8/10
63880/63880 [==============================] - 154s 2ms/step - loss: 3.9405e-05 - val_loss: 3.4837e-05
Epoch 9/10
63880/63880 [==============================] - 148s 2ms/step - loss: 3.6693e-05 - val_loss: 3.5106e-05
Epoch 10/10
63880/63880 [==============================] - 147s 2ms/step - loss: 3.4234e-05 - val_loss: 3.1950e-05
62500/62500 [==============================] - 83s 1ms/step
```

Figure 5: Training process with loss and val. loss after every epoch

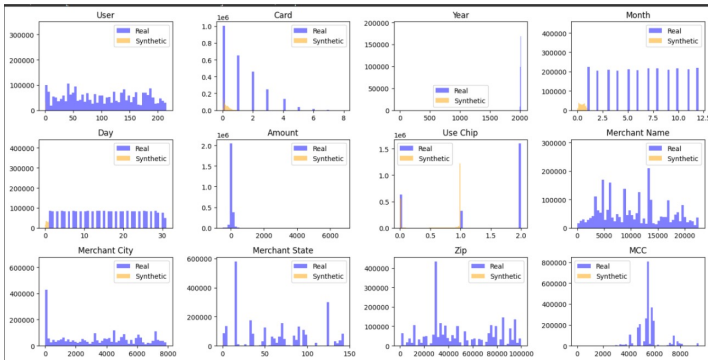
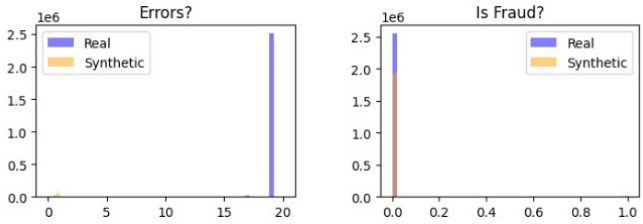


Figure 6: Real data vs synthetic data

In analyzing the results and challenges encountered, it's apparent from the visualizations that the model struggled to capture the patterns of certain features. Notably, features such as card, Use Chip, Month, and Fraud? were well-represented in the synthetic data, indicating that the model performed effectively in these instances. However, for other features, the model's performance was less optimal, suggesting that the training data may not have been sufficiently diverse or representative.

This observation underscores the importance of comprehensive data preparation and training to ensure that the model can learn and replicate the underlying patterns accurately. Despite the model's apparent capabilities, its performance was likely hindered by the limited computational resources available. The inability to train the model beyond 10 epochs due to computational constraints highlights the need for more robust infrastructure, such as GPU acceleration, to facilitate longer and more intensive training sessions.



5 Conclusion

the process of synthetic data generation plays a crucial role in various domains, including fraud detection and analysis. By creating synthetic datasets that mimic the characteristics of real-world data, we can augment the size and diversity of our datasets, thereby improving the performance and robustness of machine learning models.

One of the key advantages of synthetic data generation is its ability to address issues related to data scarcity and privacy concerns. In scenarios where obtaining large volumes of real data is challenging or impractical, synthetic data provides a viable alternative for training and testing machine learning algorithms. Additionally, synthetic data allows researchers and practitioners to explore and experiment with different scenarios and conditions without the constraints imposed by real-world data limitations.

However, it's important to acknowledge that synthetic data generation also has its limitations and potential drawbacks. One such limitation is the inherent difficulty in accurately replicating the complexity and variability present in real-world datasets. While synthetic data can closely resemble real data in some aspects, it may fail to capture certain nuances or nuances, leading to biased or inaccurate model predictions.