Fraud Detection Using an Autoencoder and Variational Autoencoder

Raj Popat
CS23MTECH14009

Vaibhav Falgun Shah
AI23MTECH02007

Sreyash Mohanty
CS23MTECH14015

Yash Shukla
CS23MTECH14018

Somya Kumar
SM23MTECH11010

Completed as a assignment of the coursework on Fraud Analytics Utilizing Predictive and Social Network Techniques (CS6890).

This assignment aims to construct and evaluate neural and variational autoencoder models for identifying fraudulent transactions in a credit card dataset. These models seek to develop representations that identify legal from fraudulent transactions by using the data's intrinsic structure and linkages. Detecting fraudulent activity in financial transactions is crucial for maintaining system integrity and protecting stakeholders' interests. Traditional fraud detection systems frequently fail to account for the complexity and diversity of fraudulent activities. Machine learning approaches, notably autoencoders and variational autoencoders, provide promising avenues for detecting fraudulent transactions by capturing underlying patterns in data.

1. Problem Statement

This assignment aims to construct and evaluate neural and variational autoencoder models for identifying fraudulent transactions in a credit card dataset. These models seek to develop representations that identify legal from fraudulent transactions by using the data's intrinsic structure and linkages.

Using **Autoencoders and Variational Autoencoders** (VAEs) for fraud detection, especially with imbalanced datasets where fraudulent instances are scarce, involves anomaly detection. The idea is that autoencoders can learn the normal patterns from non-fraudulent data and detect fraud by recognizing deviations from these patterns

2. Description of the dataset

The dataset consist of one file - 'creditcard.csv'.

The dataset comprises 284,807 credit card transactions, of which 492 are labeled as fraudulent.It consists of 30 attributes, including: 28 principal components derived from the transaction data. The time elapsed between each transaction and the first transaction in the dataset.The amount paid for each transaction.

Consider converting time difference and amount features into log scale for dynamic range compression, enhancing model performance. The dataset appears to be related to credit card transactions, containing several features for each transaction. Like Time i.e the amount of time elapsed since the first transaction in the dataset (in seconds). other features from V1-V28 which are anonymized features likely represent various aspects of the transaction, such as amount, location, and others.we also have amount column which conveys the information about the amount of the transaction.class indicates a binary indicator (0 or 1) specifying whether the transaction is fraudulent (1) or not (0). Each row in the dataset represents a single credit card transaction.

2.1. Statistics of the dataset

we have done dataset analysis using sweetviz library which is written in python.

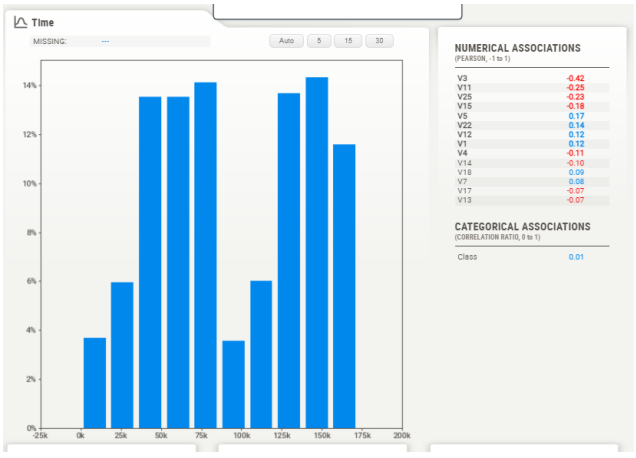

Figure 1. Time plot

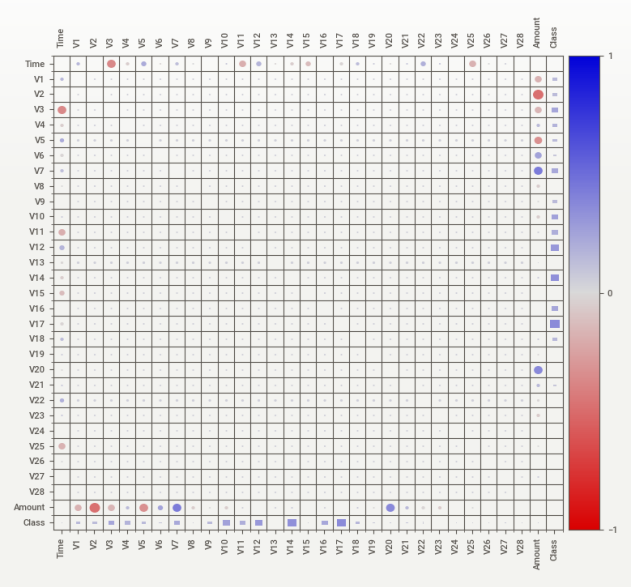

Figure 2. Association Plot

3. Algorithm Used

3.1. Autoencoder Approach

Architecture :

1) Encoder: Maps the input data to a lower-dimensional space (latent space).

2) Bottleneck (Latent space): The compressed representation of the input data.

3) Decoder: Maps the compressed representation back to the original input space.

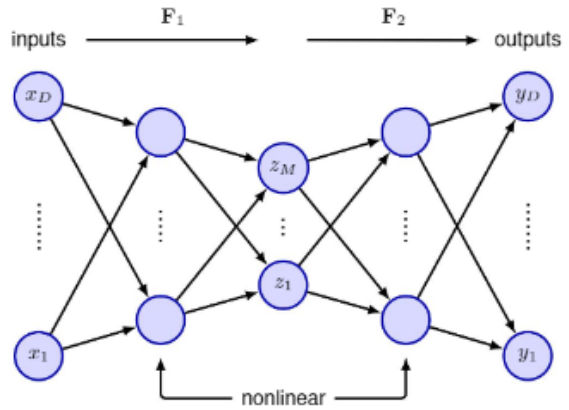

Figure 3. Process of AutoEncoder.

Architecture has Input layer with the number of features in the dataset and Hidden layers gradually reducing in size to a bottleneck (latent) layer and Symmetric decoder layers, gradually increasing in size back to the original input size.

Loss Function:

$$\mathcal{L}(X, \hat{X}) = \frac{1}{N} \sum_{i=1}^{N} \left(X_i - \hat{X}_i \right)^2 \tag{1}$$

Mean Squared Error (MSE) is between the input and reconstructed data, where N is the number of data points and the Training Procedure is to train the autoencoder on non-fraudulent data to minimize the reconstruction error. Inference is Pass each data point through the trained autoencoder and also calculate the reconstruction error (MSE) for each data point.After that, Set a threshold for anomaly detection based on the distribution of reconstruction errors from the non-fraudulent data. Then, Classify a data point as fraudulent if its reconstruction error exceeds the threshold.

Algorithm 1 Autoencoder Algorithm

Training Phase:
Initialize encoder weights W_e and biases b_e.
Initialize decoder weights W_d and biases b_d.
for each training step **do**
  $\boldsymbol{Z} = \boldsymbol{f}(\boldsymbol{X}; W_e, b_e) = \text{Encoder}(\boldsymbol{X})$ ▷ Z is the latent representation.
  $\hat{\boldsymbol{X}} = g(Z; W_d, b_d) = \text{Decoder}(Z)$ ▷ $\hat{X}$ is the reconstructed input.
  Compute reconstruction loss using Mean Squared Error (MSE):
  $\mathcal{L} = \frac{1}{N} \sum_{i=1}^{N} (X_i - \hat{X}_i)^2$
  Backpropagate the loss and update the weights W_e, b_e, W_d, b_d.
end for

Reconstruction Phase:
for each data sample X **do**
  $\boldsymbol{Z} = \boldsymbol{f}(\boldsymbol{X}; W_e, b_\epsilon)$ ▷ Encode the input to obtain the latent representation.
  $\hat{X} = g(Z; W_d, b_d)$ ▷ Decode to reconstruct the input.
  Compute the reconstruction error:
  $\epsilon = \|X - \hat{X}\|^2$ ▷ Set a threshold T to classify X as an anomaly if $\epsilon > T$.
end for

3.2. Variational Autoencoder (VAE) Approach

VAEs extend autoencoders by incorporating a probabilistic approach to the latent space. The encoder learns the parameters of a probability distribution representing the latent space.

The reparameterization trick allows gradients to backpropagate through the sampling step, enabling training with gradient-based optimization. The decoder then reconstructs the input data from the samples drawn from the learned latent distribution.

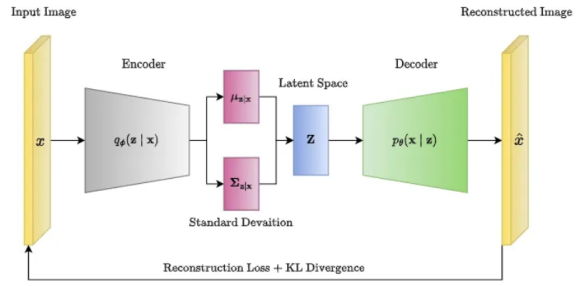

Figure 4. Process of Variational Autoencoder Approach

In this Architecture, Encoder layers to output the mean and standard deviation of the latent space distribution. Decoder layers to map the sampled latent representation back to the original input size.

Encoder :

$$\mu, \sigma = f(X) \tag{2}$$

Reparameterization Trick :

$$Z = \mu + \sigma \cdot \epsilon \quad \text{where} \quad \epsilon \sim \mathcal{N}(0,1) \tag{3}$$

Decoder :

$$\hat{X} = g(Z) \tag{4}$$

Reconstruction Loss: MSE between the input and reconstructed data.

$$\mathcal{L}_{\text{rec}} = \frac{1}{N} \sum_{i=1}^{N} \left(X_i - \hat{X}_i \right)^2 \tag{5}$$

KL Divergence: Divergence between the learned latent distribution and a standard normal distribution.

$$\mathcal{L}_{\text{KL}} = -\frac{1}{2} \sum_{j=1}^{D} \left(1 + \log \sigma_j^2 - \mu_j^2 - \sigma_j^2 \right) \tag{6}$$

Total Loss:

$$\mathcal{L} = \mathcal{L}_{\text{rec}} + \mathcal{L}_{\text{KL}} \tag{7}$$

Training Procedure: Train the VAE on non-fraudulent data to learn a probabilistic latent representation of normal patterns.

For Inference Pass each data point through the trained VAE to compute the reconstruction error. Use the KL Divergence term to gauge the likelihood of the data point being from the learned normal distribution. Set a threshold on both the reconstruction error and the KL divergence. Classify a data point as fraudulent if it exceeds the threshold on either metric.

Algorithm 2 Variational Autoencoder (VAE) Algorithm

Training Phase:
Initialize encoder weights W_e and biases b_c.
Initialize decoder weights W_d and biases b_d.
for each training step **do**
  $\mu, \sigma = f(X; W_c, b_e) = \text{Encoder}(X)$ ▷ μ and σ represent the mean and standard deviation of the latent space.
  $Z = \mu + \sigma \cdot \epsilon$ where $\epsilon \sim \mathcal{N}(0,1)$ ▷ Sample latent representation using the reparameterization trick.
  $\hat{\boldsymbol{X}} = g(Z; W_d, b_d) = \text{Decoder}(Z)$ ▷ $\hat{X}$ is the reconstructed input.
  Compute the reconstruction loss and KL divergence:
  $\mathcal{L}_{\text{rec}} = \frac{1}{N} \sum_{i=1}^{N} (X_i - \hat{X}_i)^2$
  $\mathcal{L}_{\text{KL}} = -\frac{1}{2} \sum_{j=1}^{D} (1 + \log \sigma_j^2 - \mu_j^2 - \sigma_j^2)$
  Combine the losses and backpropagate:
  $\mathcal{L} = \mathcal{L}_{\text{rec}} + \mathcal{L}_{\text{KL}}$ ▷ Update weights W_e, b_e, W_d, b_d.
end for

Reconstruction Phase:
for each data sample X **do**
  $\mu, \sigma = f(X; W_c, b_c)$ ▷ Encode to get distribution parameters.
  $Z = \mu + \sigma \cdot \epsilon$ where $\epsilon \sim \mathcal{N}(0,1)$ ▷ Sample from the latent space.
  $\hat{X} = g(Z; W_d, b_d)$ ▷ Decode to reconstruct the input.
  Compute the reconstruction error:
  $\epsilon = \|X - \hat{X}\|^2$ ▷ Calculate the anomaly score using the combined reconstruction error and KL divergence.
  Set a threshold to classify X as an anomaly if the score exceeds the threshold.
end for

4. Methodology

Autoencoders and Variational Autoencoders (VAEs) learn representations of non-fraudulent transactions by training on a dataset predominantly composed of non-fraudulent samples. Here's how they achieve this and why reconstruction loss for non-fraudulent samples is generally less than for fraudulent ones:

Learning Representations of Non-Fraudulent Transactions Training on Normal Data: During training, the model learns to compress and reconstruct the majority data, which in this case is composed of non-fraudulent transactions. The model optimizes its parameters to minimize the reconstruction loss on these non-fraudulent samples. Latent Space Representation: Autoencoder: The encoder learns to map non-fraudulent samples into a lower-dimensional latent

space that captures the significant features of the data. VAE: The encoder learns the parameters of a probability distribution that represents the latent space for non-fraudulent data. Decoder Learning: The decoder learns to map from this latent space back to the original feature space. It becomes particularly good at reconstructing patterns seen during training (non-fraudulent transactions).

Why Reconstruction Loss Is Less for Non-Fraudulent Transactions Fit to Normal Patterns: The encoder-decoder network becomes skilled at reconstructing data that follows the patterns learned during training (non-fraudulent transactions). As a result, the reconstruction loss for these normal transactions is low. Deviation from Known Patterns: Fraudulent transactions often have distinct patterns not present in the training data. When a fraudulent transaction passes through the encoder, it results in a latent representation that the decoder cannot accurately reconstruct. Higher Reconstruction Error: The discrepancy between the original fraudulent transaction and its reconstructed output (high reconstruction error) is due to the model being unable to accurately capture the unseen patterns. This results in a significantly higher reconstruction error compared to non-fraudulent transactions. Intuition Autoencoder: The latent space directly captures the patterns of non-fraudulent transactions. Fraudulent data points fall outside these patterns, resulting in higher reconstruction errors. VAE: The probabilistic latent space further accentuates the inability to reconstruct fraudulent data due to the mismatch in the distribution parameters, in addition to high reconstruction loss.

5. Results

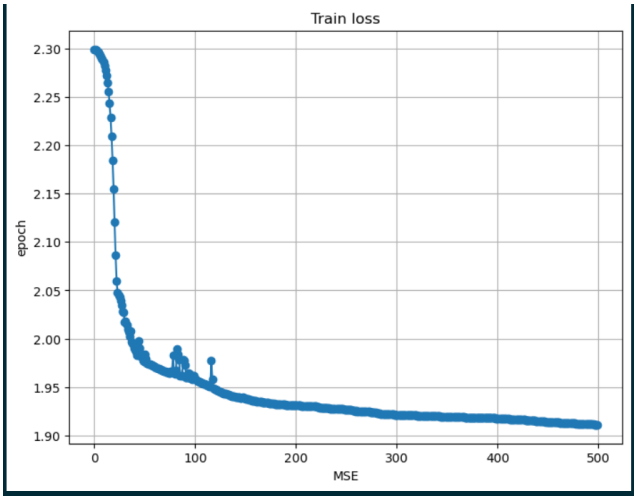

Figure 5. Output 1

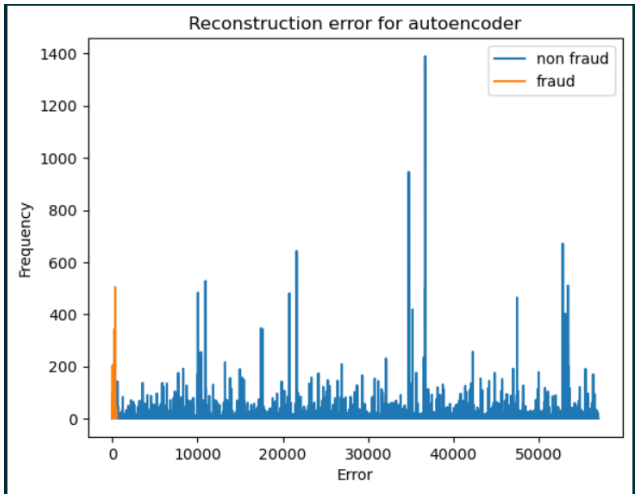

Figure 6. Output 2

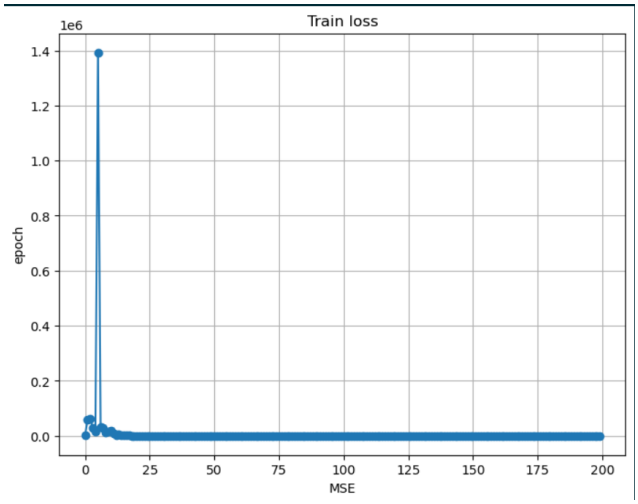

Figure 7. Iteration-wise scores of all the nodes and bad nodes are highlighted.

6. Conclusion

This assignment intends to improve fraud detection capabilities in credit card transactions by utilizing autoencoder and variational autoencoder models, giving stakeholders useful tools for recognizing and mitigating fraudulent behaviors. The research will rigorously evaluate and analyze these machine learning algorithms to validate their efficacy in tackling real-world financial security concerns.

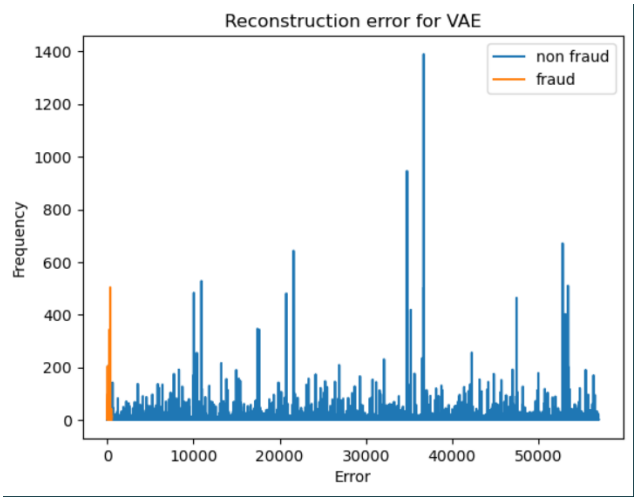

Figure 8. Iteration-wise scores of all the nodes and bad nodes are highlighted.

References

[1] amilton, W.L., Ying, R. and Leskovec, J., 2017. Representation learning on graphs: Methods and applications. arXiv preprint arXiv:1709.05584.