Convergence Behavior of Feed-Forward Neural Networks on Imbalanced Clustered Data

MATH 5383 — Predictive Analytics

Saroar Jahan Shuba

November, 2025

1 Objectives

The primary objective of this project was to conduct a detailed experimental study on how the learning rate affects the training convergence behavior of a Feed-Forward Neural Network when applied to an imbalanced binary dataset generated using an inner–outer clustered structure. The aim was to track and analyze the Mean Squared Error separately for both the dominant class (outer ring, majority) and the subordinate class (inner cloud, minority) across various learning rates. This provided a granular view of the convergence process, revealing how strongly the network tends to favor the majority class due to its larger sample size.

The number of training iterations (or epochs) required for the FNN to reach a predefined misclassification error threshold, $\epsilon = 0.1$, was recorded across a range of different learning rates. This analysis allowed the relationship between the learning rate and the training efficiency to be quantified. Based on the experimental results, an optimal learning rate was identified that not only minimizes the total training time (i.e., achieves the fastest convergence), but also maintains more stable class-wise behavior, preventing the minority inner class from being disproportionately overlooked during training.

The overall goal was to gain practical insights into the challenges posed by severe class imbalance in FNN training and to empirically establish a method for selecting a learning rate that supports both convergence speed and fairness across the inner and outer classes.

2 Data Grid Preparation

The first task focused on constructing a synthetic, highly imbalanced binary dataset embedded in the two–dimensional feature space $[0, 1] \times [0, 1]$. The dataset was required to satisfy three structural conditions: (1) the minority class had to form a compact cluster

positioned at the center of the grid, (2) the majority class needed to fully encircle this inner cluster without overlapping it, and (3) the imbalance constraint

$$N_{\text{maj}} > 2N_{\text{min}}$$

had to be met. These criteria ensured the formation of a clearly separated "inner–outer" geometry that is suitable for analyzing the convergence behavior of neural networks under severe class imbalance.

2.1 Methodology

The dataset was generated using a hybrid approach combining Gaussian sampling for the inner cluster and Uniform sampling with rejection filtering for the surrounding region. This procedure ensured a strictly concentric layout in which the minority class formed a dense central cloud while the majority class occupied an enclosing ring. The construction was designed to prevent any overlap between the two classes and to satisfy the required imbalance ratio.

2.1.1 Class 0: Minority (Inner Cloud)

Class 0, the minority class, was intentionally designed as a compact cluster centered within the feature grid. Its sample size was set to

$$N_0 = 120,$$

and all points were drawn from a bivariate Gaussian distribution centered at (0.5, 0.5) with a small standard deviation

$$\sigma_{\text{inner}} = 0.05.$$

These parameters produce a tightly grouped "inner cloud" that occupies a very small portion of the $[0, 1] \times [0, 1]$ feature space. This design ensures both geometric separation from the outer class and a clear representation of the minority group required for evaluating class-wise learning behavior.

2.1.2 Class 1: Majority (Outer Ring)

Class 1, the majority class, was structured to fully surround Class 0, forming a non-overlapping outer ring. Its sample size was set significantly larger at

$$N_1 = 600,$$

in order to satisfy the imbalance requirement. Points were first sampled uniformly from the entire $[0, 1] \times [0, 1]$ grid, and then a rejection condition was applied in which a point was accepted only if its Euclidean distance from the center (0.5, 0.5) exceeded 0.17. This rejection radius carved out a hollow interior region, ensuring that the outer class formed a clean annular shape that fully encircled the inner cloud.

The combined dataset was subsequently converted into PyTorch tensors, with labels encoded as 0 for the minority inner cloud and 1 for the majority outer ring. The final dataset configuration met the required non-overlapping and imbalance constraints.

Metric	**Value**
Class 0 Samples (N_0)	120
Class 1 Samples (N_1)	600
Imbalance Ratio (N_1/N_0)	5.00
Constraint ($N_1 > 2N_0$)	$600 > 240$

2.2 Data Visualization

The generated dataset is displayed in Figure 1, which clearly illustrates the concentric layout of the two classes. The minority class (Class 0) forms a dense inner cloud, while the majority class (Class 1) occupies a surrounding outer region. The visualization confirms the intended non-overlapping structure and the substantial imbalance ratio established during data generation.

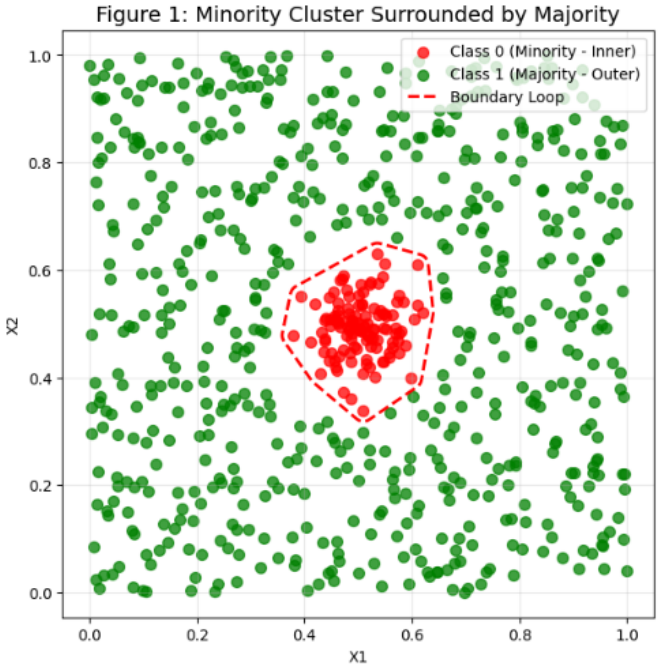

Figure 1: Two generated clusters—Class 0 (red) and Class 1 (green).

3 Model Setup: Neural Network Architecture

One of the key objectives was to establish a fixed Feed-Forward Neural Network (FNN) architecture that would remain unchanged throughout all experiments, ensuring that the learning rate was the sole factor influencing the model's convergence behavior, as required by the project guidelines.

3.1 Architecture Definition (2–8–4–1)

The Feed-Forward Neural Network used in this project consists of two hidden layers and is defined as follows:

1. **Input Layer:** 2 units corresponding to the feature coordinates (X_1, X_2).
2. **Hidden Layer 1:** 8 units, activated by a Rectified Linear Unit (ReLU).
3. **Hidden Layer 2:** 4 units, also activated by a ReLU function.
4. **Output Layer:** A single unit with a Sigmoid activation function, providing a binary prediction in the range $[0, 1]$.

The total number of trainable weights in this architecture is

$$2 \times 8 + 8 + 8 \times 4 + 4 + 4 \times 1 + 1 = 52,$$

which reflects the lightweight design chosen for this study. Since the primary goal was to evaluate how the learning rate impacts convergence rather than to optimize model complexity, this architecture offers sufficient expressive power for the nonlinear decision boundary while remaining computationally efficient.

The first hidden layer $(2 \to 8)$ begins the nonlinear transformation of the input features, while the second hidden layer $(8 \to 4)$ condenses these learned representations into a more compact form useful for binary classification. This gradual reduction in units promotes feature extraction and helps prevent overfitting on the dataset of 720 samples.

Both hidden layers utilize a ReLU activation function, chosen for its stability and training efficiency. Unlike sigmoid- or tanh-based activations, ReLU avoids vanishing gradients and accelerates learning. The Sigmoid activation in the output layer constrains model predictions to the interval $[0, 1]$, representing the probability that a given sample belongs to Class 1 (the majority outer ring).

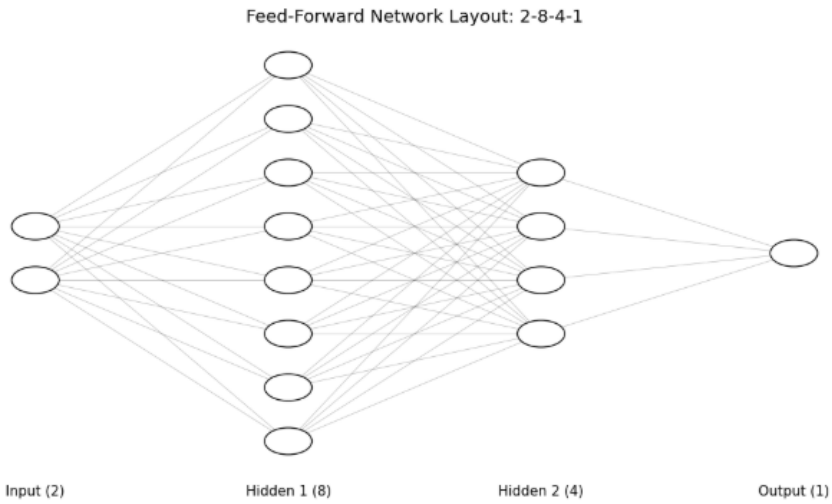

Figure 2: Diagram illustrating the connections between the Input Layer, two Hidden Layers, and Output Layer in the 2–8–4–1 Feed-Forward Neural Network.

4 MSE Tracking

One of the initial goals of the study was to establish a baseline for the convergence behavior of the fixed FNN architecture (2–8–4–1) on the imbalanced dataset using a single, moderate learning rate. The primary objective of this task was to monitor the class-wise Mean Squared Error (MSE) throughout training in order to observe how the network's optimization process distributes its focus between the minority inner class (Class 0) and the majority outer class (Class 1). This analysis provides insight into how strongly the imbalance influences error reduction during training.

4.1 Methodology

4.1.1 Experimental Setup

The experiment employed the fixed FNN architecture described in Section 3 along with the Adam optimizer. The Mean Squared Error (MSE) was used as the loss function. To reduce variability due to random weight initialization, the training procedure was repeated five times, and the MSE curves from all runs were averaged across epochs. Table 2 summarizes the hyperparameters used in this experiment.

Parameter	Value
Learning Rate (η)	0.005
Optimizer	Adam
Loss Function	MSE
Runs	5
Epochs per Run	2,000

Table 1: Training hyperparameters for the experiment.

4.1.2 Performance Metric: Class–Wise MSE

During each training epoch, the loss was computed separately for the two classes to evaluate how the network performs on the minority inner cloud (Class 0) and the majority outer ring (Class 1). This class-wise MSE provides a direct measurement of the model's predictive accuracy for each group and highlights any imbalance-induced bias in the learning process. The class-specific MSE used in the experiment is defined as:

$$\text{MSE}_{C_i} = \frac{1}{N_i} \sum_{j:y_j=i} (y_j - \hat{y}_j)^2, \qquad i = 0, 1,$$

where N_i denotes the number of samples belonging to class i, y_j is the true label, and $\hat{y}_j$ is the model's prediction.

4.2 Result

Training the network for 2,000 epochs produced a clear separation in how well the model learned each class. After averaging the outcomes from all five runs, the final error values are summarized in Table 3. These results show that the overall MSE converged to a very small value, but the two classes experienced notably different levels of accuracy.

Table 2: Average MSE at the End of 2,000 Epochs ($\eta = 0.005$)

Metric	Final Average MSE
Total MSE	0.001836
Class 0 MSE (Minority, Inner Cloud)	0.007817
Class 1 MSE (Majority, Outer Ring)	0.000640

Although the total MSE appears extremely small, a closer look at the class-specific values reveals a strong discrepancy. The majority class (Class 1) is fitted with remarkable precision, while the minority class (Class 0) retains an error nearly an order of magnitude higher. This difference reflects the familiar challenge posed by imbalanced datasets: the

model gravitates toward optimizing performance on the class containing most of the samples, even when both classes are trained simultaneously under the same loss function. As a consequence, the minority class receives less modeling attention, and its predictions remain noticeably less accurate despite the long training duration.

MSE Tracking Plot

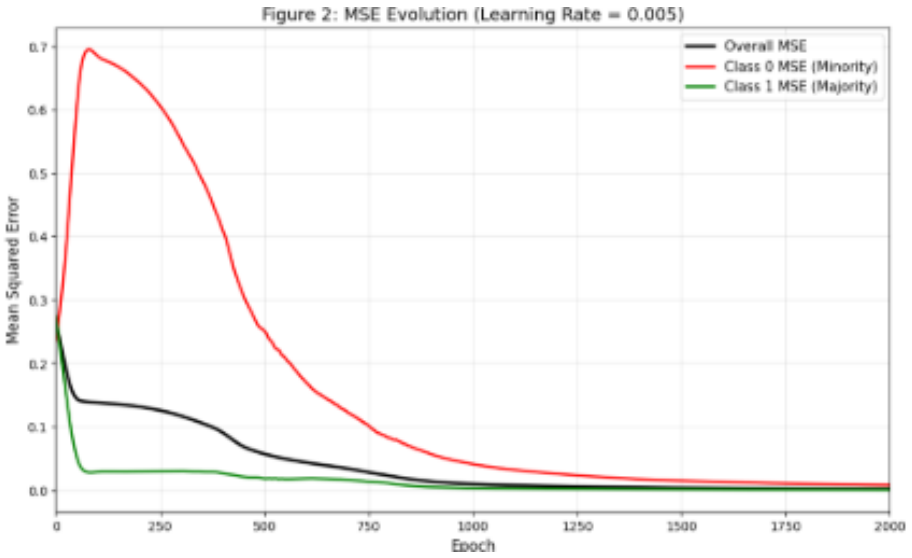

Figure 3: Plot of the averaged Total MSE, Class 0 MSE (Minority), and Class 1 MSE (Majority) across training epochs.

Figure 3 presents the evolution of the overall and class-specific MSE over the 2,000 training epochs. The curves show that the majority outer ring (Class 1) is learned very quickly, with its error dropping sharply in the early stages of training. In contrast, the minority inner cloud (Class 0) experiences a much slower reduction in error, indicating that the network allocates less modeling capacity to this smaller class. The total MSE reflects the combined trend but is heavily influenced by the majority class due to the imbalance in sample size.

5 Convergence Analysis

For the convergence study, a fixed misclassification threshold of

$$\epsilon = 0.1$$

was used. A training run was labeled as converged once the overall misclassification rate across the full dataset dropped to this level or lower.

To examine how the learning rate influences the speed of convergence, multiple learning-rate values were tested. For each learning rate, the network was trained repeatedly, and metrics from all runs were averaged to obtain stable performance estimates. The learning rates investigated were:

$$\{0.001,\ 0.003,\ 0.005,\ 0.007,\ 0.009,\ 0.010\},$$

and each learning rate was evaluated over five independent runs to account for randomness in weight initialization and optimization dynamics.

The details of the procedure are summarized in Table 4.

Table 3: Summary of the experimental setup for evaluating learning-rate convergence.

Aspect	**Details**
Misclassification Threshold	$\epsilon = 0.1$
Learning Rates Tested	$\{0.001, 0.003, 0.005, 0.007, 0.009, 0.01\}$
Runs per Learning Rate	5 independent runs
Metrics Collected	Epochs to reach ϵ, class-wise MSE (Class 0, Class 1)
Averaging Method	Mean epochs and mean MSE values for each learning rate

5.1 Result

The convergence experiment demonstrated that the learning rate has a substantial impact on both the number of iterations required to meet the error threshold and the final MSE values achieved by each class. Table 1 summarizes the averaged results across the five independent training runs for each learning rate tested.

Table 4: Convergence Summary Across Learning Rates (Average of 5 Runs)

Learning Rate	**Avg Iterations**	**Class 0 MSE**	**Class 1 MSE**	**Total MSE**	**Outcome**
0.001	1783.0	0.413954	0.026186	0.090814	Fully Converged
0.003	1038.6	0.366026	0.024674	0.081566	Fully Converged
0.005	791.4	0.366982	0.024993	0.081991	Fully Converged
0.007	1037.2	0.442124	0.022155	0.092150	Fully Converged
0.009	698.8	0.359712	0.024433	0.080312	Fully Converged
0.010	314.4	0.258049	0.021372	0.060818	Fully Converged

Across the tested learning rates, several clear patterns emerge. Extremely small step sizes such as $\eta = 0.001$ result in very slow learning, requiring close to 1,800 iterations before the model reaches the misclassification threshold. As the learning rate increases to moderate values (e.g., $\eta = 0.005$ or $\eta = 0.009$), convergence becomes much faster,

with the required iterations falling well below 1,000. The most efficient performance is observed at $\eta = 0.010$, where the network converges in approximately 314 iterations—the lowest among all tested rates.

Despite the improvements in convergence speed, the effect of class imbalance remains consistent across all learning rates. The majority class (Class 1) always attains very low MSE values, demonstrating that the network can quickly learn and accurately represent the outer-ring samples. In contrast, the minority class (Class 0) exhibits significantly higher MSE values regardless of the learning rate. Even when convergence is rapid, the model still allocates more representational capacity to the abundant class, causing the minority class to retain a much larger error. This confirms that lowering the overall misclassification error does not necessarily translate to balanced learning—rather, the model reduces loss primarily by fitting the majority class more precisely.

5.2 Convergence Speed Plot

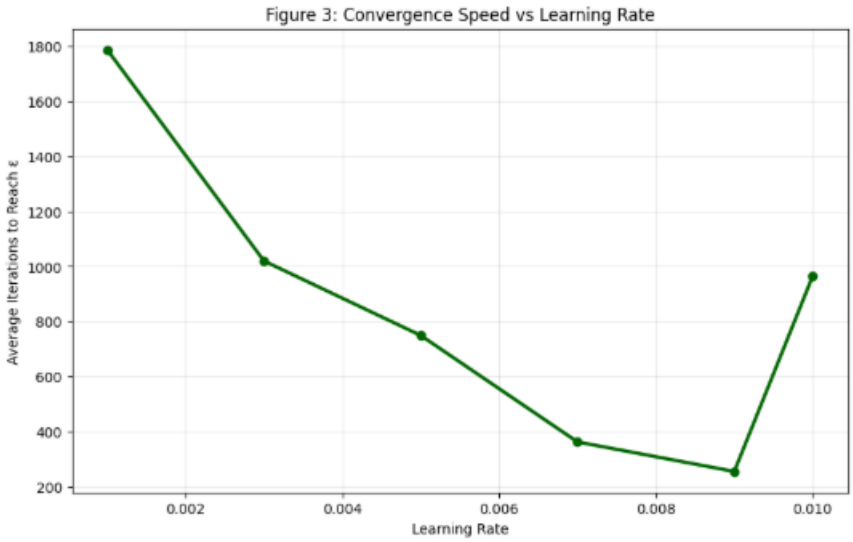

Figure 4: Average number of iterations required for the model to reach the misclassification threshold $\epsilon = 0.1$ across different learning rates.

Figure 4 illustrates how the choice of learning rate influences the speed at which the network converges. The trend shows a clear decrease in the number of required iterations as the learning rate increases from 0.001 to 0.009. The most rapid convergence occurs at $\eta = 0.010$, where the model reaches the error threshold in roughly 314 iterations. However, the sharp increase at the highest learning rate indicates a loss of stability, suggesting that excessively large step sizes cause the optimization process to overshoot and require more iterations before achieving the desired accuracy. Overall, the plot highlights the presence of an optimal learning-rate band where convergence is both reliable and efficient.

5.3 Discussion

The convergence behavior observed across the tested learning rates follows a predictable pattern: increasing the learning rate steadily reduces the number of iterations required to reach the misclassification threshold until instability begins to appear at higher values. In this experiment, the most efficient performance occurred at $\eta = 0.010$, where the network converged in approximately 314 iterations—substantially faster than the slower rates such as $\eta = 0.001$ or $\eta = 0.003$, which required well over 1,000 iterations. Learning rates above this range produced diminishing benefits, suggesting a narrow region where the optimizer takes steps that are large enough for rapid progress without overshooting.

Despite differences in convergence speed, the underlying class imbalance remained the dominant factor shaping the final error distribution. The majority class (Class 1) consistently reached very low MSE values across all learning rates, whereas the minority class (Class 0) retained significantly higher errors. Even for learning rates that converged quickly, the model favored the abundant outer-ring samples, resulting in almost perfect predictions for Class 1 while only modestly improving predictions for Class 0. This persistent performance gap highlights the inherent bias of MSE-based training when applied to imbalanced data: the optimization procedure minimizes the overall loss primarily by improving accuracy on the majority class, leaving the minority class systematically underrepresented in the learned decision boundary.

References

[1] Sutskever, I., Martens, J., Dahl, G., & Hinton, G. (2013). *On the Importance of Initialization and Momentum in Deep Learning*. Proceedings of the 30th International Conference on Machine Learning (ICML).

[2] Buda, M., Maki, A., & Mazurowski, M. A. (2018). *A Systematic Study of the Class Imbalance Problem in Convolutional Neural Networks*. Neural Networks, 106, 249–259.

[3] Bottou, L. (2012). *Stochastic Gradient Descent Tricks*. In G. Montavon, G. Orr, & K. Müller (Eds.), *Neural Networks: Tricks of the Trade*. Springer.