Enhancing Logistic Regression Using Neurochaos Learning

Krishna Sajeev [1,*], Niranjana Prajith [1,*] and Akhila Henry[1,†]

Abstract

The rapid expansion of machine learning has intensified the demand for classification algorithms that are both effective and interpretable. Neurochaos Learning (NL), a chaos-based machine learning framework, has demonstrated strong classification performance; however, its computational complexity can limit practical applicability. To overcome this limitation, simplified NL variants have been introduced, employing chaos-inspired features such as the Tracemean and Fluctuation Index, which retain essential dynamical characteristics while reducing computational overhead. In this work, we integrate these simplified Neurochaos features with logistic regression, a classical and interpretable classification model. Logistic regression provides a mathematically transparent framework for evaluating the discriminative power and linear separability of chaos-derived features. The proposed approach is evaluated on several benchmark datasets, including Iris, Breast Cancer, Wine, Statlog, and Penguins datasets. Through systematic experimentation, we investigate the contribution of simplified NL features to classification accuracy, generalization capability, and stability across datasets. The results demonstrate that combining simplified Neurochaos features with logistic regression yields competitive performance while maintaining low computational complexity and high interpretability, making the approach suitable for practical and explainable machine learning applications.

Keywords: Tracemean, Fluctuation index, Logistic regression, Neurochaos learning

Introduction

Machine learning (ML) has become a cornerstone of modern data analysis, enabling systems to learn patterns and make decisions directly from data without explicit rule-based programming. Its applications span diverse domains such as healthcare, finance, signal processing, and pattern recognition. Despite the success of advanced models, a persistent challenge in machine learning is the effective representation of complex, nonlinear relationships present in real-world data.

---

*These authors contributed equally to this work.

1Department of Mathematics, Amrita Vishwa Vidyapeetham, Amritapuri Campus, Kollam

Feature extraction and preprocessing play a crucial role in determining classifier performance, particularly when lightweight or classical learning models are employed. Consequently, there is growing interest in biologically inspired and nonlinear transformation techniques that can enrich data representations while maintaining interpretability and computational efficiency (Mukhamediev et al., 2022).

Logistic Regression (LR) is one of the most widely used supervised learning algorithms for classification problems, especially binary classification. It models the posterior probability of class membership using the sigmoid function, which maps linear combinations of input features to the interval $[0,1]$. Due to its simplicity, interpretability, and low computational cost, logistic regression remains a preferred baseline model in many applications. However, its linear decision boundary limits its ability to capture complex nonlinear patterns inherent in many datasets. As a result, improving the representational power of logistic regression through effective preprocessing and feature transformation—without modifying its internal structure—has become an active area of research (Ba'abbad, Althubiti, Alharbi, Alfarsi, & Rasheed, 2021).

Chaos theory offers a powerful mathematical framework for understanding complex, nonlinear, and deterministic systems that exhibit sensitive dependence on initial conditions. Unlike random processes, chaotic systems follow precise rules but generate rich and unpredictable dynamics. These properties make chaos particularly attractive for machine learning, where diversity, nonlinearity, and sensitivity can enhance feature representation and pattern discrimination (Alligood, Sauer, & Yorke, 1997). Interestingly, neurophysiological studies suggest that the human brain operates near the "edge of chaos," a regime that balances order and randomness to support efficient information processing, adaptability, and learning. Motivated by this observation, Neurochaos Learning (NL) was introduced as a brain-inspired computational framework that integrates chaotic dynamics into machine learning pipelines (N. Harikrishnan & Nagaraj, 2020; Sethi, Nagaraj, & B., 2023). In NL, each input feature is processed by a chaotic neuron—typically governed by maps such as the Generalized Lüroth Series (GLS)—to generate a neural trace that encodes nonlinear transformations of the input. From this trace, chaos-based features are extracted to form a high-dimensional representation that enhances classification performance (Hamidouche, Guesmi, & Essounbouli, 2024)..

While the original NL framework demonstrates strong performance across a range of classification tasks, it introduces increased dimensionality, computational overhead, and sensitivity to multiple hyperparameters. To address these limitations, simplified NL models have been proposed that retain the expressive power of chaotic transformations while reducing complexity (Henry, Sundaravaradhan, & Nagaraj, 2025). Building on this idea, the present work focuses on enhancing Logistic Regression through chaos-driven feature transformation rather than architectural modification. Four chaotic maps—Skew Tent ($T(x)$), Skew Binary ($B(x)$), $sin(\pi x)$ ($S(x)$) and the Logistic map ($L(x)$)—are employed to generate neural traces, from which two compact features, namely Tracemean(TM) and Fluctuation Index(FI), are extracted. The proposed Neurochaos-enhanced Logistic Regression model is evaluated on ten benchmark datasets - Bank Note (Gillich & Lohweg, 2010), Breast Cancer (Street, Wolberg, & Mangasarian, 1993), Haberman (Haberman, 1973), Iris (Fisher, 1936), Seeds (Dua & Graff, 2017), Statlog (Dua & Graff, 2017), Wine(Forina et al., 1988), Ionosphere (Sigillito, Wing, Hutton, & Baker, 1989), Penguin (Horst, Hill, & Gorman, 2020) and Sonar dataset (Gorman & Sejnowski, 1988), demonstrating consistent and significant improvements over stand-alone logistic regression. The results highlight that chaos-based preprocessing, particularly using Skew Tent, Skew Binary, and $sin(\pi x)$ maps, provides a lightweight yet powerful mechanism for improving classification accuracy. This study establishes Neurochaos Learning as an effective enhancement strategy for classical machine learning models, offering improved performance without sacrificing simplicity or interpretability (Anusree & Pramod, 2025).

Neurochaos Algorithm and Simplified Variants

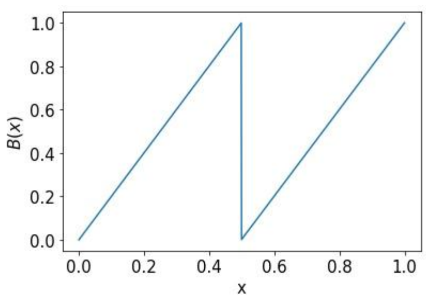
Neurochaos Learning (NL) has emerged as a powerful framework for classification tasks involving complex and highly nonlinear data. By transforming input features through chaotic dynamics prior to classification, NL enhances class separability and enables the extraction of discriminative patterns that may be overlooked by conventional linear or nonlinear models. Empirical evaluations demonstrate that NL consistently achieves superior F1 scores when compared with standard machine learning algorithms, highlighting the effectiveness of chaos-based feature representations. The framework further allows flexibility in classifier design, as evidenced by the integration of ChaosNet, which exploits the intrinsic structure of chaotic features to improve decision boundaries. Despite these advantages, the original NL framework suffers from notable limitations. The fourfold expansion of feature dimensionality significantly increases storage requirements and computational cost, particularly for large-scale datasets. Moreover, the need to carefully tune multiple hyperparameters, namely the initial point q, threshold b, and tolerance ε, introduces additional complexity and sensitivity to initialization. These factors collectively hinder scalability and practical deployment, thereby motivating the development of simplified NL architectures (Henry & Nagaraj, 2025; Henry, Nagaraj, & Sundaravaradhan, 2025).

To address these challenges, a simplified NL model was proposed in (Henry, Sundaravaradhan, et al., 2025), which substantially reduces computational and design complexity while preserving the core benefits of chaos-driven learning. In this formulation, the skew/threshold parameters are fixed at $b = 0.499$ and $\varepsilon = 0.25$, effectively eliminating the need for extensive hyperparameter tuning. Furthermore, instead of producing a fourfold expansion of chaotic features, the simplified approach extracts a single representative feature—either the tracemean or the Fluctuation Index—from the neural trace generated by the Skew Tent Map (Henry, Sundaravaradhan, et al., 2025). This reduction in feature dimensionality leads to lower memory usage and faster computation, while still retaining the ability to capture meaningful nonlinear and chaotic structures inherent in the data (AS, Harikrishnan, & Nagaraj, 2023; N. B. Harikrishnan & Nagaraj, 2021; NB, Kathpalia, & Nagaraj, 2022).

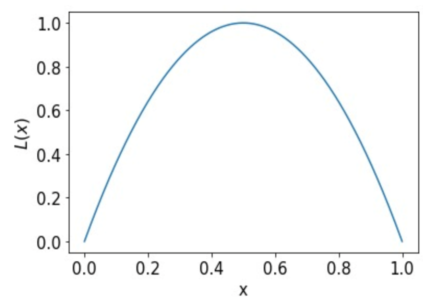
However, the simplified NL framework has so far been evaluated only in conjunction with the Random Forest classifier and is restricted to chaotic learning based on the Skew Tent Map. In this work, we extend the simplified NL paradigm in two important directions. First, we investigate its integration with logistic regression, a widely used linear classifier, to assess the effectiveness of chaos-based features in enhancing linear decision models. Second, we broaden the scope of chaotic transformations by incorporating additional maps, namely the Logistic Map ($r = 4$), Skew Binary Map ($b = 0.499$), skew tent map ($b = 0.499$) and the $\sin(\pi x)$ map. The mathematical definition and graphs of these maps are shown in Table 1 and Figure 1 respectively. These extensions aim to improve the generality and applicability of simplified NL while preserving its low computational overhead and interpretability.

Table 1: Mathematical definitions of chaotic maps used in the study

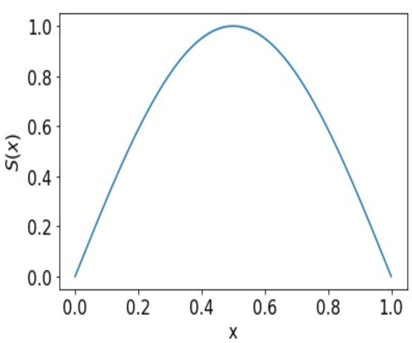
Map	**Mathematical Definition**
Skew Binary Map	$B(x) = \begin{cases} \frac{x}{b}, & 0 \leq x < b, \\ \frac{x-b}{1-b}, & b \leq x \leq 1 \end{cases}$
Logistic Map	$L(x) = r\,x(1-x), \quad 0 < r \leq 4$
Sine Map	$S(x) = \sin(\pi x)$
Skew Tent Map	$T(x_n) = \begin{cases} \frac{x}{b}, & 0 \leq x < b, \\ \frac{1-x}{1-b}, & b \leq x \leq 1 \end{cases}$



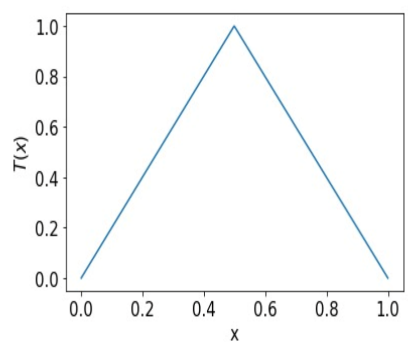
(a) Skew Binary Map with $b = 0.499$ $(B(x))$



(b) Logistic map with $r = 4$ $(L(x))$



(c) $sin(\pi x)$ map $(S(x))$



(d) Skew Tent map with $b = 0.499$ $(T(x))$

Figure 1: Graphs of the chaotic maps used in Neurochaotic Logistic Regression models.

Neurochaotic Logistic Regression Model

This section presents the proposed neurochaotic logistic regression framework, in which original input features are transformed into chaos-driven representations—namely the tracemean (TM) or Fluctuation Index (FI)—and subsequently classified using a logistic regression model.

Step 1: Data Normalization
Suppose the given dataset contains m samples, each sample contains n features, represented as

$$\{(a_{11}, a_{12}, \dots, a_{1n}), (a_{21}, a_{22}, \dots, a_{2n}), \dots, (a_{m1}, a_{m2}, \dots, a_{mn})\}.$$

To ensure uniform scaling across features, Min–Max normalization is applied independently to each feature. The normalized value b_{ij} corresponding to a_{ij} is computed as

$$b_{ij} = \frac{a_{ij} - \min\{a_{ij}: 1 \le i \le m\}}{\max\{a_{ij}: 1 \le i \le m\} - \min\{a_{ij}: 1 \le i \le m\}}, \qquad 1 \le j \le n.$$

The resulting normalized dataset is given by

$$\{(b_{11}, b_{12}, \dots, b_{1n}), (b_{21}, b_{22}, \dots, b_{2n}), \dots, (b_{m1}, b_{m2}, \dots, b_{mn})\}.$$

Step 2: Generation of Chaotic Neural Trace
Corresponding to each b_{ij}, a neural trace is generated using a chaotic map. The chaotic systems considered in this study include the Skew Tent map (T), Skew Binary map (B), Logistic map (L), and the $\sin(\pi x)$ map (S).

The neural trace generation process is initiated from an initial neural activity q. During training, the parameter q is varied within the interval $[0.01, 0.99]$ in increments of 0.01 to identify the value yielding optimal classification performance. The dataset is partitioned such that 80% of the samples are used for training, while the remaining 20% are reserved for testing.

Let f denote a chosen chaotic map. The neural trace is obtained through iterative composition of f, yielding

$$N = \{q, f(q), f^2(q), \dots, f^T(q)\},$$

where T denotes the firing time. This sequence characterizes the chaotic evolution of neural activity and forms the basis for subsequent feature extraction.

Step 3: Extraction of chaotic features from neural trace
Given n normalized features, n corresponding chaotic map instances $\{f_1, f_2, \dots, f_n\}$ are employed. For the j-th feature, the neural trace evolves as

$$N_j = \{q, f_j(q), (f_j)^2(q), (f_j)^3(q), \dots\}, \qquad j = 1,2, \dots, n.$$

The evolution continues until the trajectory enters an ε-neighborhood of the stimulus value b_{ij}. In this work, the tolerance parameter is fixed at $\varepsilon = 0.25$. Upon satisfying this stopping criterion, either the tracemean or the Fluctuation Index of the neural trace is computed.

The resulting scalar quantity serves as the transformed representation of the original feature b_{ij}. Repeating this procedure for all features and all samples yields the transformed dataset

$$\{(t_{i1}, t_{i2}, \dots, t_{in}) : i = 1,2, \dots, m\},$$

which encapsulates nonlinear and chaotic characteristics of the input data.

Step 4: Classification Using Logistic Regression
The transformed feature vectors are classified using logistic regression. Let

$$\{(t_{i1}, t_{i2}, \dots, t_{in}) : i = 1,2, \dots, m\}$$

denote the transformed dataset, where each sample consists of n features. For the i-th sample, a linear combination of the input features is computed using a weight vector $w = (w_1, w_2, \dots, w_n)$ and a bias term b:

$$z_i = w_1 t_{i1} + w_2 t_{i2} + \cdots + w_n t_{in} + b = w \cdot t_i + b.$$

The scalar z_i is passed through the sigmoid activation function

$$\sigma(z_i) = \frac{1}{1 + e^{-z_i}},$$

which maps the output to the interval $[0,1]$, representing the estimated probability that the i-th sample belongs to Class 1. A decision threshold of 0.5 is applied to obtain the predicted class label:

$$\hat{y}_i = \begin{cases} 1, & \text{if } \sigma(z_i) \geq 0.5, \\ 0, & \text{otherwise.} \end{cases}$$

Thus, the proposed framework combines chaos-based nonlinear feature transformation with a linear probabilistic classifier to achieve effective classification. The figure 2 shows the flowchart of the above described algorithm.

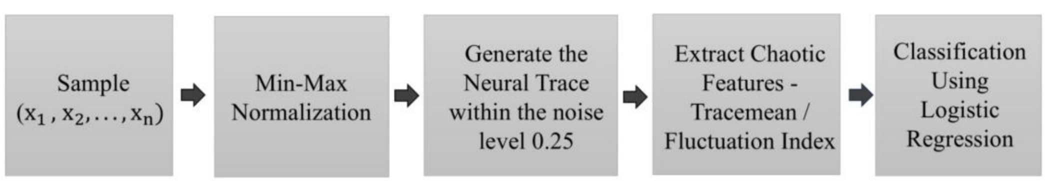


Figure 2: Neurochaotic Logistic Regression Model

Experimental Results

This section outlines the results of logistic regression with chaos based transformation (Tracemean and Fluctuation index) using four different chaotic maps.

Logistic Regression using the mean of the neural trace

The tables 2, 3, 4, 5 reports the results obtained by applying the chaotic maps - Skew Tent map, Skew Binary Map, $sin(\pi x)$ and Logistic Map followed by the Tracemean feature extraction before training the Logistic Regression classifier.

Table 2: Performance of Logistic Regression algorithm with tracemean feature extracted from neural trace generated by Skew Tent Map

Sl. No	Dataset	q	Training F1 Score	Testing F1 Score
1	Bank Note	0.030	0.941	0.989
2	Breast Cancer	0.050	0.966	0.936
3	Haberman	0.420	0.610	0.630
4	Iris	0.110	0.958	1.000
5	Seeds	0.010	0.944	0.874
6	Statlog	0.960	0.815	0.899
7	Wine	0.630	0.960	0.894
8	Ionosphere	0.820	0.893	0.924
9	Penguin	0.090	0.983	1.000
10	Sonar	0.450	0.764	0.667

Table 3: Performance of Logistic Regression algorithm with tracemean feature extracted from neural trace generated by Skew Binary Map

Sl. No	Dataset	q	Training F1 Score	Testing F1 Score	
1	Bank Note	0.030	0.941	0.989	
2	Breast Cancer	0.050	0.966	0.936	
3	Haberman	0.420	0.610	0.630	
4	Iris	0.110	0.958	1.000	
5	Seeds	0.010	0.938	0.874	
6	Statlog	0.800	0.813	0.809	
7	Wine	0.630	0.967	0.918	
8	Ionosphere	0.820	0.884	0.908	
9	Penguin	0.090	0.983	1.000	
10	Sonar	0.450	0.759	0.697	

Table 4: Performance of Logistic Regression algorithm with tracemean feature extracted from neural trace generated by $sin(\pi x)$ Map

Sl. No	Dataset	q	Training F1 Score	Testing F1 Score
1	Bank Note	0.050	0.943	0.952
2	Breast Cancer	0.030	0.956	0.927
3	Haberman	0.420	0.610	0.630
4	Iris	0.460	0.948	1.000
5	Seeds	0.030	0.943	0.850
6	Statlog	0.980	0.810	0.899

Sl. No	Dataset	q	Training F1 Score	Testing F1 Score
7	Wine	0.630	0.958	0.894
8	Ionosphere	0.760	0.888	0.892
9	Penguin	0.050	0.976	0.909
10	Sonar	0.450	0.764	0.690

Table 5: Performance of Logistic Regression algorithm with tracemean feature extracted from neural trace generated by Logistic Map ($r = 4$)

Sl. No	Dataset	q	Training F1 Score	Testing F1 Score
1	Bank Note	0.980	0.931	0.926
2	Breast Cancer	0.460	0.957	0.840
3	Haberman	0.420	0.610	0.630
4	Iris	0.460	0.948	1.000
5	Seeds	0.069	0.942	0.926
6	Statlog	0.810	0.809	0.878
7	Wine	0.630	0.960	0.918
8	Ionosphere	0.760	0.888	0.892
9	Penguin	0.940	0.976	0.925
10	Sonar	0.460	0.773	0.632

Logistic Regression using Fluctuation Index of neural trace

The tables 6, 7, 8, 9 present the macro F1 scores obtained using FI based feature transformation with Logistic Regression across four different chaotic maps- Skew Tent, Skew Binary, $sin(\pi x)$ and Logistic Map ($r = 4$).

Table 6: Performance of Logistic Regression algorithm with Fluctuation Index feature extracted from neural trace generated by Skew Tent Map

Sl. No	Dataset	q	Training F1 Score	Testing F1 Score
1	Bank Note	0.060	0.882	0.815
2	Breast Cancer	0.690	0.966	0.832
3	Haberman	0.099	0.648	0.617
4	Iris	0.050	0.926	0.932
5	Seeds	0.620	0.904	0.816
6	Statlog	0.390	0.830	0.796
7	Wine	0.650	0.971	0.976
8	Ionosphere	0.069	0.880	0.892
9	Penguin	0.090	0.983	1.000

Sl. No	Dataset	q	Training F1 Score	Testing F1 Score
10	Sonar	0.770	0.759	0.756

Table 7: Performance of Logistic Regression algorithm with Fluctuation Index feature extracted from neural trace generated by Skew Binary Map

Sl. No	Dataset	q	Training F1 Score	Testing F1 Score
1	Bank Note	0.890	0.917	0.846
2	Breast Cancer	0.490	0.959	0.789
3	Haberman	0.410	0.647	0.617
4	Iris	0.050	0.926	0.932
5	Seeds	0.940	0.959	0.873
6	Statlog	0.030	0.818	0.878
7	Wine	0.920	0.978	0.919
8	Ionosphere	0.910	0.896	0.825
9	Penguin	0.090	0.983	1.000
10	Sonar	0.460	0.790	0.711

Table 8: Performance of Logistic Regression algorithm with Fluctuation Index feature extracted from neural trace generated by sin(πx) Map

Sl. No	Dataset	q	Training F1 Score	Testing F1 Score
1	Bank Note	0.090	0.862	0.796
2	Breast Cancer	0.740	0.965	0.954
3	Haberman	0.890	0.632	0.439
4	Iris	0.040	0.929	0.893
5	Seeds	0.750	0.923	0.869
6	Statlog	0.650	0.829	0.779
7	Wine	0.680	0.979	0.911
8	Ionosphere	0.110	0.875	0.878
9	Penguin	0.030	0.967	1.000
10	Sonar	0.850	0.771	0.711

Table 9: Performance of Logistic Regression algorithm with Fluctuation Index feature extracted from neural trace generated by Logistic Map(r=4)

Sl. No	Dataset	q	Training F1 Score	Testing F1 Score
1	Bank Note	0.020	0.865	0.790
2	Breast Cancer	0.690	0.959	0.840
3	Haberman	0.880	0.647	0.617

Sl. No	Dataset	q	Training F1 Score	Testing F1 Score
4	Iris	0.020	0.892	0.853
5	Seeds	0.780	0.944	0.917
6	Statlog	0.320	0.829	0.801
7	Wine	0.760	0.977	1.000
8	Ionosphere	0.069	0.892	0.892
9	Penguin	0.720	0.963	0.945
10	Sonar	0.780	0.763	0.808

Discussion

The impact of chaotic transformations on both the Tracemean (TM) and Fluctuation Index (FI) features demonstrates their ability to consistently enhance classification performance across a wide range of datasets. Analyzing each chaotic map reveals complementary and reinforcing effects on these two feature representations.

Using the *tracemean* feature, chaotic maps outperform the standalone model on 8 out of 10 datasets (80%). Among the evaluated maps, the Skew Tent and Skew Binary maps exhibit the strongest and most consistent performance, each achieving the best or joint-best F1-score in 5 datasets. The Logistic and $\sin(\pi x)$ maps follow, providing best performance in 3 and 2 datasets, respectively. Across all datasets, the average F1-score improvement obtained using tracemean features is approximately 5.5%, with modest gains of 0.3–3% observed for simpler datasets such as Seeds, Statlog, and Iris, and substantially higher improvements for more complex datasets. The Breast Cancer dataset shows the highest improvement of 16.1%, while Haberman and Ionosphere record gains of approximately 8%. For the Wine and Sonar datasets, chaotic maps achieve performance comparable to the standalone model, indicating saturation of baseline separability.

When the *Fluctuation Index* feature is employed, chaotic maps outperform the standalone logistic regression baseline on 6 out of 10 datasets (60%), while yielding significantly larger improvements in F1-score. In this setting, the Logistic and $\sin(\pi x)$ maps dominate, each producing the best performance in 4 datasets, followed by the Skew Tent and Skew Binary maps with best results in 2 datasets each. The average F1-score improvement increases to approximately 9.6%, nearly double that obtained using tracemean features. Notably, the Haberman dataset exhibits a dramatic improvement of 63.4%, demonstrating the effectiveness of chaos-induced fluctuations in enhancing linear classifiers on overlapping and imbalanced data. Additional gains of 18.4% for Breast Cancer, 5.9% for Sonar, and 4.2% for Ionosphere further highlight the advantage of fluctuation-based chaotic features.

Tables 10 and 11 present a detailed comparative analysis of classification performance obtained using different chaotic maps in terms of F1-score for the tracemean and Fluctuation Index feature representations, respectively.

Table 10: Comparison of performance of Logistic Regression with tracemean feature generated by different chaotic maps.

Sl. No	Dataset	Standalone	Skew Tent	Skew Binary	$sin(\pi x)$	Logistic
1	Bank Note	0.951	**0.989**	**0.989**	0.952	0.926
2	Breast Cancer	0.806	**0.936**	**0.936**	0.927	0.840
3	Haberman	0.584	**0.630**	**0.630**	**0.630**	**0.630**
4	Iris	0.965	**1.000**	**1.000**	**1.000**	**1.000**
5	Seeds	0.923	0.874	0.874	0.850	**0.926**
6	Statlog	0.878	**0.899**	0.809	**0.899**	0.878
7	Wine	**0.968**	0.894	0.918	0.894	0.918
8	Ionosphere	0.856	**0.924**	0.908	0.892	0.892
9	Penguin	0.981	**1.000**	**1.000**	0.909	0.925
10	Sonar	**0.763**	0.667	0.697	0.69	0.632

Table 11: Comparison of performance of Logistic Regression with fluctuation index feature generated by different chaotic maps.

Sl. No	Dataset	Standalone	Skew Tent	Skew Binary	$sin(\pi x)$	Logistic
1	Bank Note	**0.951**	0.815	0.846	0.796	0.790
2	Breast Cancer	0.806	0.832	0.789	**0.954**	0.840
3	Haberman	0.584	0.832	0.789	**0.954**	0.840
4	Iris	**0.965**	0.932	0.932	0.893	0.853
5	Seeds	**0.923**	0.816	0.873	0.869	0.917
6	Statlog	**0.878**	0.796	**0.878**	0.779	0.801
7	Wine	0.968	0.976	0.919	0.911	**1.000**
8	Ionosphere	0.856	**0.892**	0.825	0.878	**0.892**
9	Penguin	0.981	0.983	**1.000**	**1.000**	0.945
10	Sonar	0.763	0.756	0.711	0.711	**0.808**

A comparative analysis of both feature representations shows that chaotic maps provide more consistent improvements with the tracemean feature in terms of dataset coverage (80% versus 60%), whereas the Fluctuation Index feature yields substantially higher absolute improvements in F1-score. Overall, chaotic maps outperform standalone models in 14 out of 20 experimental cases (70%). Skew-based maps demonstrate stable and reliable gains across different classifiers, while the Logistic and $\sin(\pi x)$ maps are particularly effective when coupled with fluctuation-driven features. These quantitative findings clearly establish that chaotic map–based feature generation enhances classification performance, especially for datasets with nonlinear or overlapping class distributions.

Conclusion

This work investigated the effectiveness of incorporating chaotic features into the Logistic Regression algorithm using the Neurochaos Learning (NL) framework. Neural traces were generated using four chaotic maps—Skew Tent, Skew Binary, $Sin(\pi x)$, and the Logistic map—and their impact on classification performance was evaluated across ten benchmark datasets.

Within the NL framework, chaos-induced neural traces were first generated and subsequently summarized using two feature extraction measures: tracemean and Fluctuation Index. These single-feature representations were then classified using Logistic Regression. Experimental results consistently demonstrated that integrating chaotic transformations significantly improves performance compared to the stand-alone Logistic Regression model.

Among the chaotic maps, Skew Tent and Skew Binary emerged as the most effective, delivering substantial and consistent gains across multiple datasets, including Bank Note, Breast Cancer, Haberman, Ionosphere, and Penguin. The $Sin(\pi x)$ map also provided notable improvements, particularly for the Breast Cancer and Statlog datasets. Although the Logistic map produced comparatively modest gains, it exhibited dataset-specific advantages, especially in the Seeds and Wine datasets.

A comparative analysis of the extracted features revealed that the tracemean feature consistently outperformed the Fluctuation Index, offering more stable and higher performance improvements across nearly all datasets.

Overall, the findings confirm that Neurochaos Learning combined with suitable chaotic transformations—especially Skew Tent, Skew Binary, and $Sin(\pi x)$ can significantly enhance classification accuracy. This demonstrates that chaos-driven preprocessing and feature extraction serve as a lightweight yet powerful augmentation to classical machine learning models such as Logistic Regression, achieving performance gains without modifying the classifier's internal architecture.

Future research may extend this framework by exploring a broader range of chaotic systems capable of generating richer neural traces. Additionally, integrating chaotic feature representations into deep learning architectures and evaluating their effectiveness on real-world datasets represent promising directions for further investigation.

Appendix

Dataset Description

Table 12 summarizes the characteristics of the datasets used in this study, including the number of features, number of classes, and total number of samples.

Table 12: Number of features and samples in each dataset used in this study.

Dataset	Number of Features (Attributes)	Number of Classes	Number of Samples
Iris	4	3	150
Haberman's Survival	3	2	306
Seeds	7	3	210
Statlog (Heart)	13	2	270
Ionosphere	34	2	351
Bank Note Authentication	4	2	1372
Breast Cancer Wisconsin	31	2	569
Wine	13	3	178
Penguin	4	3	342
Sonar	60	2	208

Performance of Standalone Logistic Regression Algorithm

Table 13 shows the F1 Scores for the 10 datasets using Stand Alone Logistic Regression. The set of candidate values considered for C, regularisation parameter, was $[0.001, 0.01, 0.1, 1, 10, 100]$.

Table 13: Performance of standalone logistic regression algorithm on different datasets.

Sl. No	Dataset	C	Training F1	Testing F1
1	Bank Note	100	0.987	0.951
2	Breast Cancer	100	0.971	0.806
3	Haberman	100	0.550	0.584
4	Iris	100	0.946	0.965
5	Seeds	100	0.947	0.923
6	Statlog	0.1	0.820	0.878
7	Wine	1	0.985	0.968
8	Ionosphere	10	0.848	0.856
9	Penguin	100	0.980	0.981
10	Sonar	100	0.735	0.763

References

1. Alligood, K. T., Sauer, T. D., & Yorke, J. A. (1997). *Chaos: An introduction to dynamical systems*. Springer.

2. Anusree, M., & Pramod, P. N. (2025). Understanding chaotic neural networks: A comprehensive review. *Nonlinear Dynamics*, 1–16.

3. AS, R. A., Harikrishnan, N. B., & Nagaraj, N. (2023). Analysis of logistic map based neurons in neurochaos learning architectures for data classification. *Chaos, Solitons & Fractals*, *170*, 113347.

4. Ba'abbad, I., Althubiti, T., Alharbi, A., Alfarsi, K., & Rasheed, S. (2021). A short review of classification algorithms accuracy for data prediction in data mining applications. *Journal of Data Analysis and Information Processing*, *9*(3), 162–174.

5. Dua, D., & Graff, C. (2017). *UCI machine learning repository*. University of California, Irvine, School of Information; Computer Sciences. Retrieved from http://archive.ics.uci.edu/ml

6. Fisher, R. A. (1936). The use of multiple measurements in taxonomic problems. *Annals of Eugenics*, *7*(2), 179–188. https://doi.org/10.1111/j.1469-1809.1936.tb02137.x

7. Forina, M., Leardi, R., Armanino, C., Lanteri, S., Conti, P., & Princi, P. (1988). *PARVUS: An extendable package of programs for data exploration, classification and correlation*. Amsterdam: Elsevier Science Software.

8. Gillich, E., & Lohweg, V. (2010). Banknote authentication. *1. Jahreskolloquium Bildverarbeitung in Der Automation*, 1–8.

9. Gorman, R. P., & Sejnowski, T. J. (1988). Analysis of hidden units in a layered network trained to classify sonar targets. *Neural Networks*, *1*(1), 75–89.

10. Haberman, S. J. (1973). The analysis of residuals in cross-classified tables. *Biometrics*, *29*(1), 205–220. https://doi.org/10.2307/2529686

11. Hamidouche, B., Guesmi, K., & Essounbouli, N. (2024). Mastering chaos: A review. *Annual Reviews in Control*, *58*, 100966.

12. Harikrishnan, N. B., & Nagaraj, N. (2021). When noise meets chaos: Stochastic resonance in neurochaos learning. *Neural Networks*, *143*, 425–435.

13. Harikrishnan, N., & Nagaraj, N. (2020). Neurochaos inspired hybrid machine learning architecture for classification. *2020 International Conference on Signal Processing and Communications (SPCOM)*, 1–5. IEEE.

14. Henry, A., & Nagaraj, N. (2025). Neurochaos learning for classification using composition of chaotic maps. *Chaos Theory and Applications*, *7*(2), 107–116.

15. Henry, A., Nagaraj, N., & Sundaravaradhan, R. (2025). Universal orbits: Unveiling the connection between chaotic dynamics, normal numbers, and neurochaos learning. *Chaos Theory and Applications*, *7*(1), 61–69.

16. Henry, A., Sundaravaradhan, R., & Nagaraj, N. (2025). Simplified neurochaos learning architectures for data classification. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, *35*(6), 063109. https://doi.org/10.1063/5.0263796

17. Horst, A. M., Hill, A. P., & Gorman, K. B. (2020). *Palmerpenguins: Palmer archipelago (antarctica) penguin data*. Retrieved from allisonhorst.github.io

18. Mukhamediev, R. I., Popova, Y., Kuchin, Y., Zaitseva, E., Kalimoldayev, A., Symagulov, A., et al.others. (2022). Review of artificial intelligence and machine learning technologies: Classification, restrictions, opportunities and challenges. *Mathematics*, *10*(15), 2552. https://doi.org/10.3390/math10152552

19. NB, H., Kathpalia, A., & Nagaraj, N. (2022). Causality preserving chaotic transformation and classification using neurochaos learning. *Advances in Neural Information Processing Systems*, *35*, 2046–2058.

20. Sethi, D., Nagaraj, N., & B., H. N. (2023). Neurochaos feature transformation for machine learning. *Integration*, *90*, 157–162. https://doi.org/https://doi.org/10.1016/j.vlsi.2023.01.014

21. Sigillito, V. G., Wing, S. P., Hutton, L. V., & Baker, K. B. (1989). Classification of radar returns from the ionosphere using neural networks. *Johns Hopkins APL Technical Digest*, *10*(3), 262–266.

22. Street, W. N., Wolberg, W. H., & Mangasarian, O. L. (1993). Nuclear feature extraction for breast tumor diagnosis. *Biomedical Image Processing and Biomedical Visualization*, *1905*, 861–870. SPIE. https://doi.org/10.1117/12.148698