Machine Learning Models Comparison Report

Ivan Seslija

October 26, 2023

Abstract

This report evaluates the accuracy of three machine learning models—Stochastic Gradient Descent (SGD) Support Vector Machine (SVM), Gaussian SVM, and a Neural Network (NN)—using a binary classification task on a modified Fashion MNIST dataset featuring sandals and sneakers. Models are optimized through k-fold cross-validation and grid search to identify the best parameters, followed by a performance comparison based on accuracy.

1 Introduction

In the rapidly evolving landscape of machine learning, choosing the right model for a specific task is paramount. This study delves into the comparative analysis of three prominent models: SGD SVM, Gaussian SVM, and NN. Using the tailored Fashion MNIST dataset, which has been narrowed down to sandals and sneakers, we aim to determine which model excels in binary classification. To ensure that each model performs at its peak, a methodical approach involving k-fold cross-validation and grid search is employed for parameter tuning. The culmination of this study is a clear assessment of each model's ability to classify images accurately, shedding light on their respective strengths and areas of improvement.

2 Individual Model Testing

In testing each of the models, we made the decision to add noise to the training data. For each label, there was a 20% chance to flip it, 0 to 1 or 1 to 0 respectively.

Otherwise, each classifier is configured and tested independently of one another.

2.1 SGD SVM

2.1.1 Parameter Selection

In our experiment involving the SGD SVM, selecting appropriate hyperparameters was a pivotal step. A total of 30 distinct values were chosen for the regularization parameter C, logarithmically spaced between 10^{-3} and 10^{3}. This approach allowed us to thoroughly assess the model's responsiveness to various C values, ensuring a robust evaluation process.

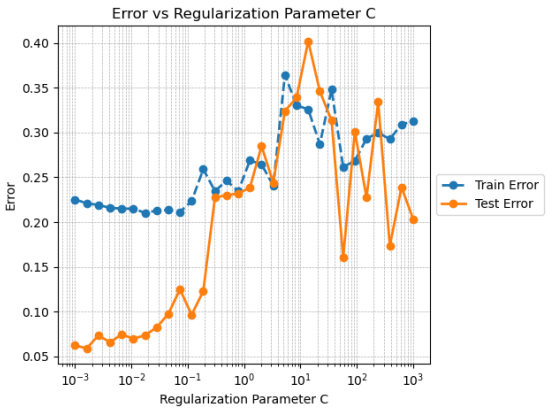
2.1.2 Parameter Conversion: C to Alpha

An essential aspect to note is the conversion of the C parameter to α, as facilitated by the scikit-learn model's internal mechanics. This conversion is represented by the equation:

$$\alpha = \frac{1}{n \cdot C}$$

Where n is the sample size. This conversion is intrinsic to the model's configuration within scikit-learn, playing a critical role in fine-tuning and optimizing the classifier's performance.

2.1.3 Results



In the error analysis, it is observed that between 10^{-3} and slightly before 10^{-1}, the test error consistently outperforms the training error, exhibiting an error range of 0.05-0.1, compared to the 0.2-0.25 range seen in the training error.

Beyond this point, the patterns of test and training errors start moving in closer alignment, although the test error maintains a slight edge, generally showing better performance.

A plausible explanation for this behavior could be the presence of noise in the initial segments of the test set. This noise likely complicates the model's adaptation process, making it challenging to achieve lower error rates in the earlier stages of testing.

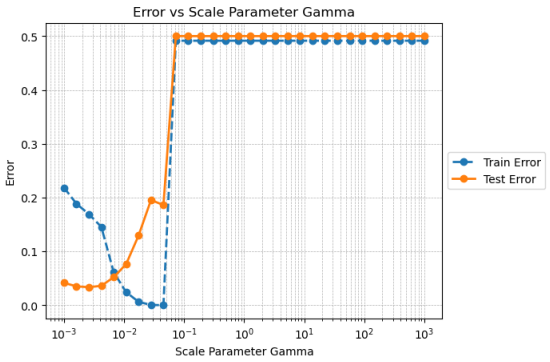
Furthermore, an increase in the regularization parameter, C, is associated with elevated error rates in both training and test sets. Excessive regularization appears to hinder the model's learning capability, thereby escalating the error rates. This observation emphasizes the necessity of a meticulously chosen regularization level, ensuring an optimum balance in the bias-variance tradeoff to enhance the model's generalization proficiency.

2.2 Gaussian SVM

2.2.1 Parameter Selection

In the Gaussian SVM analysis, we systematically tested combinations of C and γ, each varying from 10^{-3} to 10^{3}. In the illustrated graph, C remains constant at the scikit-learn default of 1.0, while γ is varied to ascertain its impact on model performance.

2.2.2 Results



In the illustrated graph, various γ values are analyzed to discern their influence on the Gaussian SVM's performance. At lower γ values, the model exhibits a lower test error compared to the training error, demonstrating effective generalization despite the noise present in the training data. However, a shift is observed as γ increases, where the test error begins to exceed the training error, signaling that the model might be adjusting more to the training noise, diminishing its generalization capabilities.

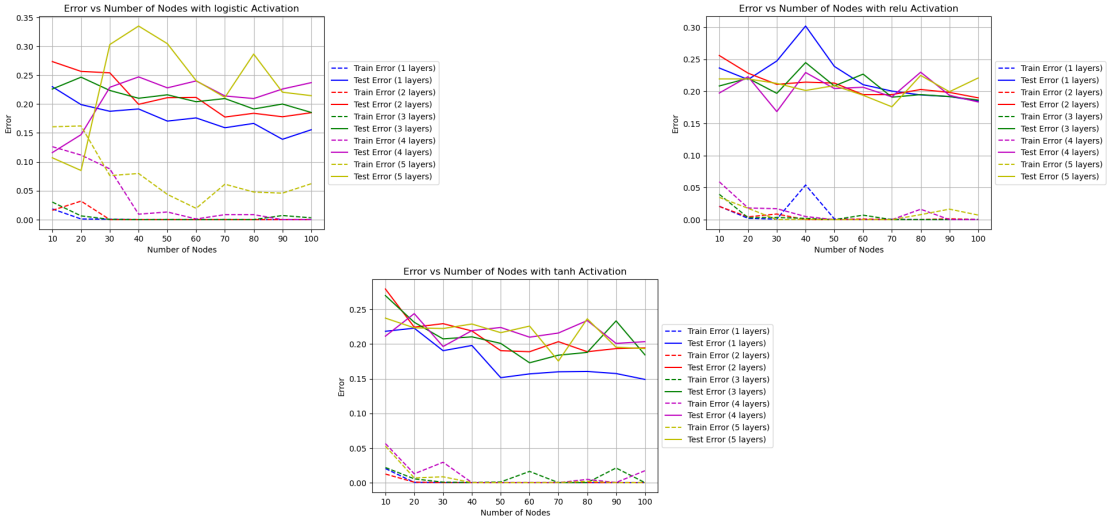
As the γ value increases beyond 0.1, a stabilization in error rates is observed, with both stabilizing around 50%. This illustrates that escalating γ values do not offer additional advantages. It seems that by increasing γ, the model becomes overly restricted and loses its effective decision-making ability, making its performance comparable to a mere coin flip due to limited adaptability to the noise in the training set.

2.3 Neural Network (NN)

2.3.1 Parameter Selection

We experimented with various neural network configurations to optimize performance. Three activation functions: logistic, ReLU, and tanh were tested. The architecture was varied from one to five hidden layers, and the node count in each layer ranged from 10 to 100, in increments of 10. This experimentation aimed to identify the most effective combination of activation function, layer depth, and node count for the task.

2.3.2 Results



The comparative analysis of different activation functions revealed a relatively similar performance, with error rates ranging from 15 to 25 percent. Among the tested functions, ReLU exhibited the least sensitivity to parameter variations, demonstrating more consistent performance across different configurations.

Focusing on architectural complexity, it was found that networks with a single hidden layer generally outperformed those with multiple hidden layers. Further-

more, an increase in the number of nodes within the layer showed a tendency to enhance the model's effectiveness, reflecting in improved error percentages.

3 Model Comparison

In this section, we conduct a comparative analysis of the selected models, evaluating their performance across the entire dataset. The models are configured using optimal parameters, identified through rigorous k-fold cross-validation grid searches.

The optimal parameters established for each model are as follows:

- SGD SVM:
  - C: 0.001
- Gaussian SVM:
  - Gamma: 0.0016102620275609393
  - C: 0.7880462815669912
- NN:
  - Activation Function: Logistic
  - Number of Hidden Layers: 5
  - Number of Nodes per Hidden Layer: 40

Utilizing these optimized parameters, a comprehensive performance assessment of each model was conducted, ensuring a robust and insightful evaluation.

From the analysis of the graphical data, it is observed that the Gaussian SVM exhibits superior performance, maintaining remarkable accuracy of 97% even in the presence of training noise. Following closely, the SGD SVM achieves an accuracy of 94%, while NN accomplishes an accuracy of 92%.

3.1 Neural Network Hidden Layers Analysis

The k-fold cross-validation suggested an optimal configuration of five hidden layers for the NN. However, upon closer examination of the graphs and performance metrics, a single hidden layer appears to be more effective and efficient for the model.

This discrepancy likely arises due to the limitation in the parameter selection methodology on the limited training set made available during cross-validation.

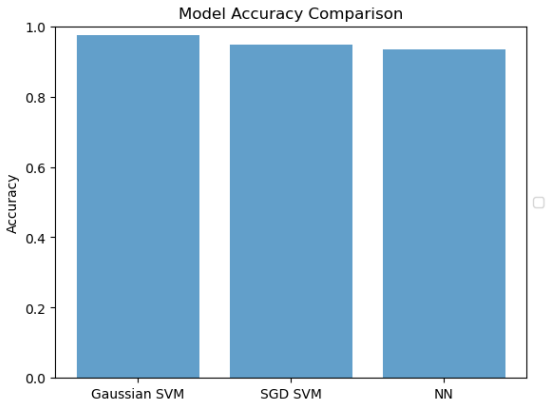


Figure 1

4 Conclusion

Our meticulous evaluation and comparative analysis across different machine learning models reveal that the Gaussian Support Vector Machine (SVM) triumphs as the most competent model among those examined. It exhibited exceptional resilience and adaptability to the noise embedded within the training data, consistently delivering accurate labels and classifications with an impressive accuracy of 97%.

In contrast, the Stochastic Gradient Descent (SGD) Support Vector Machine (SVM) and Neural Network (NN) models yielded accuracies of 94% and 92%, respectively. These results, while commendable, were slightly overshadowed by the outstanding performance of the Gaussian SVM.

Additionally, our analysis has unearthed intriguing insights into the potential limitations of k-fold cross-validation in hyperparameter optimization. The prevalent methodology, while generally reliable, may not always pinpoint the most advantageous hyperparameters, as evidenced in our experiments with neural network configurations.

4.1 Future Inquiry

Our findings beckon further exploration into the efficacy of k-fold cross-validation in the realm of hyperparameter tuning. It sparks curiosity regarding its unwavering capability to discern the optimal hyperparameters across diverse scenarios and conditions, warranting additional research and experimentation to unveil deeper understanding and insights.