Lab1 Report

Yuyao Fan 522070910015

April 15, 2025

1 Introduction

This report presents the implementation and comparison of three classification models: Logistic Regression, Support Vector Machine (SVM), and Multi-Layer Perceptron (MLP). The objective is to evaluate their performance in terms of accuracy and training time on a given dataset.

2 Methodology

2.1 Data Preprocessing

The dataset was preprocessed using feature scaling with the ‘StandardScaler‘ from the scikit-learn library. This ensures that all features are standardized to have zero mean and unit variance, which helps in improving the convergence of the models during training.

2.2 Model Implementation

2.2.1 Logistic Regression

Logistic Regression is a linear model used for binary classification. It uses the logistic function to model the probability of a certain class or event. The model was implemented using gradient descent to minimize the logistic loss.

2.2.2 Support Vector Machine (SVM)

SVM is a powerful model for classification tasks. It finds the hyperplane that maximally separates the classes in the feature space. The model was implemented using hinge loss and gradient descent.

2.2.3 Multi-Layer Perceptron (MLP)

MLP is a type of artificial neural network with multiple layers. It consists of an input layer, one or more hidden layers, and an output layer. This implementation uses one hidden layer with ReLU activation and the Sigmoid function for the output layer. The model was implemented using PyTorch.

2.3 Parameter Settings

- **Logistic Regression**: Learning rate = 0.1, Max iterations = 5000
- **SVM**: Learning rate = 0.001, Lambda (regularization parameter) = 0.01, Max iterations = 1000
- **MLP**: Hidden size = 64, Learning rate = 0.001, Epochs = 100

3 Model Comparison

Below are the performance of three models:

The training loss for **Logistic Regression** decreased steadily over the epochs, indicating good convergence. Both training and test accuracy converged to high values, showing that the model generalizes well to unseen data.

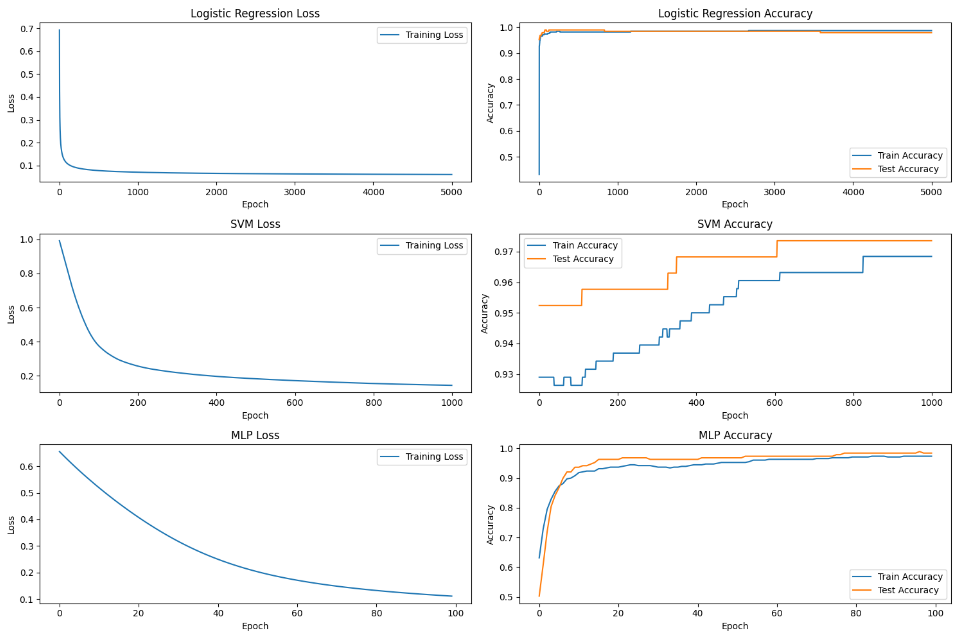

Figure 1: All Models Comparison

The training loss for **SVM** also decreased steadily over the epochs. The accuracy metrics improved over time, indicating effective learning. However, the model required more iterations to converge compared to Logistic Regression.

The **MLP** model showed a rapid decrease in training loss and a quick increase in accuracy during the initial epochs. Both training and test accuracy reached high values, demonstrating the model's ability to capture complex patterns in the data.

Here, we represent the train accuracy, test accuracy and training time for the three models in table format as below:

Model	**Train Accuracy**	**Test Accuracy**	**Training Time (seconds)**
Logistic Regression	0.9868	0.9788	2.98
SVM	0.9684	0.9735	1.45
MLP	0.9737	0.9841	0.36

Table 1: Model Performance Comparison

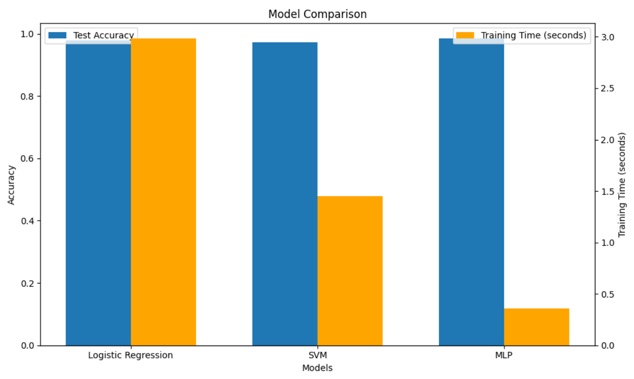

Figure 2: Model comparison

The MLP model achieved the highest test accuracy and required the least training time. The LR model had the most training time but slightly lower accuracy. The SVM model offered a balance between accuracy and training time.

4 Conclusion and Analysis

The experimental results demonstrate that the MLP model achieved the highest test accuracy while requiring the least training time. This can be attributed to the MLP's ability to capture complex patterns in the data efficiently. The architecture of the MLP, with its hidden layers and non-linear activation functions, allows it to learn intricate relationships that simpler models might miss. Additionally, the use of advanced optimization algorithms in frameworks like PyTorch can significantly speed up the training process, making MLPs not only accurate but also relatively quick to train despite their complexity.

On the other hand, the LR model, despite being the simplest among the three, had the longest training time and achieved slightly lower accuracy. This might be due to the fact that LR is a linear model and may struggle to capture non-linear relationships in the data effectively. While LR is generally fast to train, the specific implementation and optimization techniques used might have influenced its training time in this case.

The SVM model offered a balance between accuracy and training time. SVMs are known for their efficiency in high-dimensional spaces and ability to handle non-linear data through kernel tricks. However, the choice of kernel and the model's parameters can significantly impact both its performance and training time. In this case, the SVM was able to achieve a reasonable trade-off, making it a viable option when a balance between speed and accuracy is desired.

In summary, the MLP model's superior performance in both accuracy and training time highlights its effectiveness for complex classification tasks. The LR model, while simpler, may not be as efficient or accurate when dealing with non-linear data. The SVM provides a middle ground, offering a good compromise between computational efficiency and model accuracy. The choice of model should be guided by the specific requirements of the application, considering factors such as data complexity, available computational resources, and the need for high accuracy versus training efficiency.

5 References

- Project code and data available in the provided repository.
- Libraries used: numpy, pandas, torch, scikit-learn, matplotlib.
- Large language models like kimi.