IOWA STATE UNIVERSITY

Cross-Validation and Hyperparameter Tuning for Breast Cancer Classification

Ehsan Ghafourian

Elnaz Bashir

April 2023

Introduction

- Breast cancer is a prevalent health issue, but early and accurate diagnosis can save lives.
- The Breast Cancer Wisconsin (Diagnostic) dataset is a valuable resource for developing and testing predictive models.
- In this project, we use linear classification techniques to predict whether a breast mass is **benign** or **malignant**.
- This presentation will cover the **data importing process**, **hyperparameter tuning**, and **cross-validation** for model evaluation.

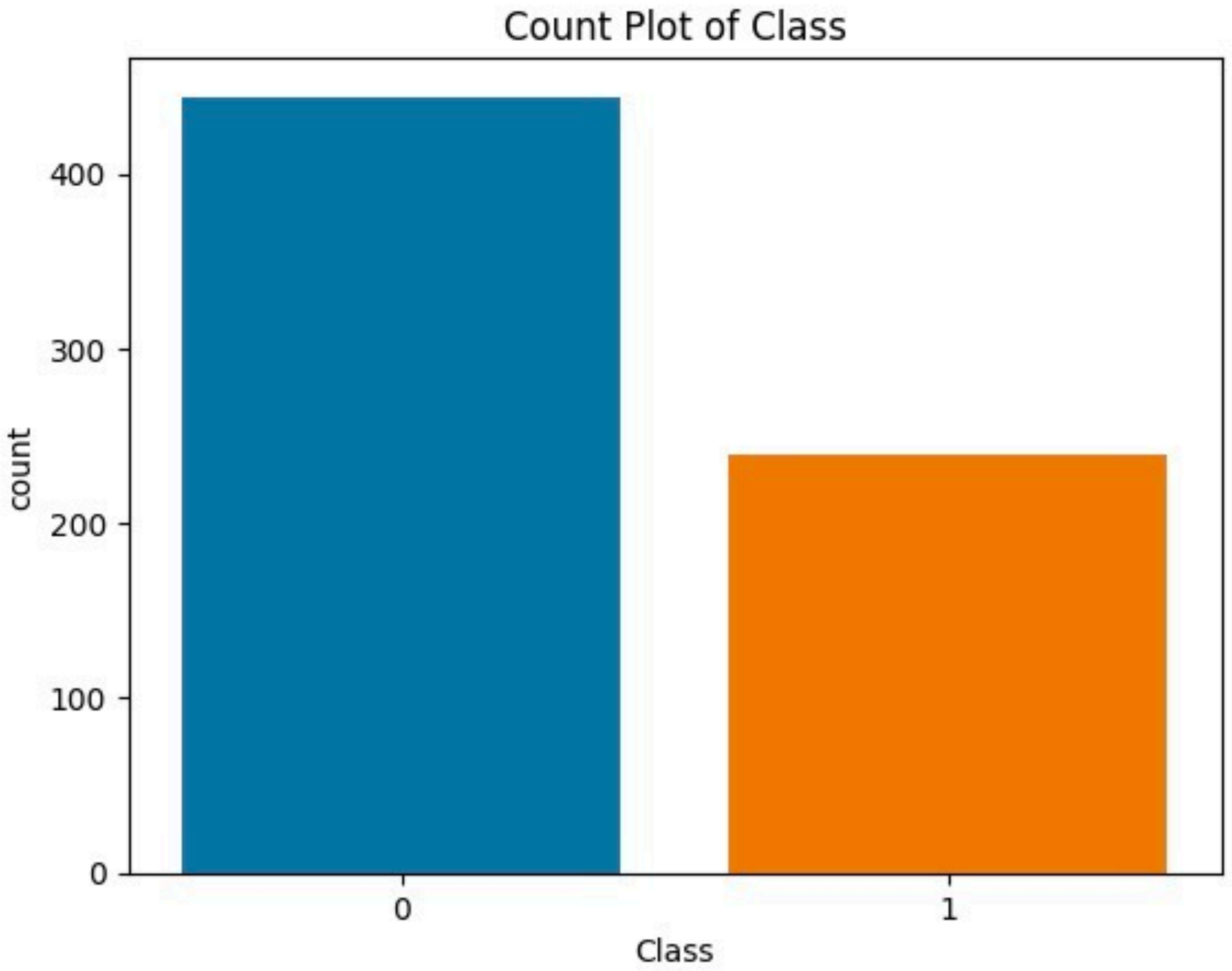
Introduction

Model Evaluation Challenges:

- Limitations of a single train-test split: Potential bias and limited data utilization.
- Importance of robust evaluation: Ensuring the linear classifier's performance is reliable and generalizable.

Dataset Overview

- Number of instances: 699
- Number of attributes: 10



Attribute	Domain
Sample code number	id number
Clump Thickness	1 - 10
Uniformity of Cell Size	1 - 10
Uniformity of Cell Shape	1 - 10
Marginal Adhesion	1 - 10
Single Epithelial Cell Size	1 - 10
Bare Nuclei	1 - 10
Bland Chromatin	1 - 10
Normal Nucleoli	1 - 10
Mitoses	1 - 10
Class	2 for benign 4 for malignant

Data Importing

- Tools and libraries used: Pandas, NumPy, scikit-learn
- Process:

    1. Load the dataset from the UCI repository
    2. Convert data to Pandas DataFrame
    3. Split the dataset into features (X) and target (y) variables

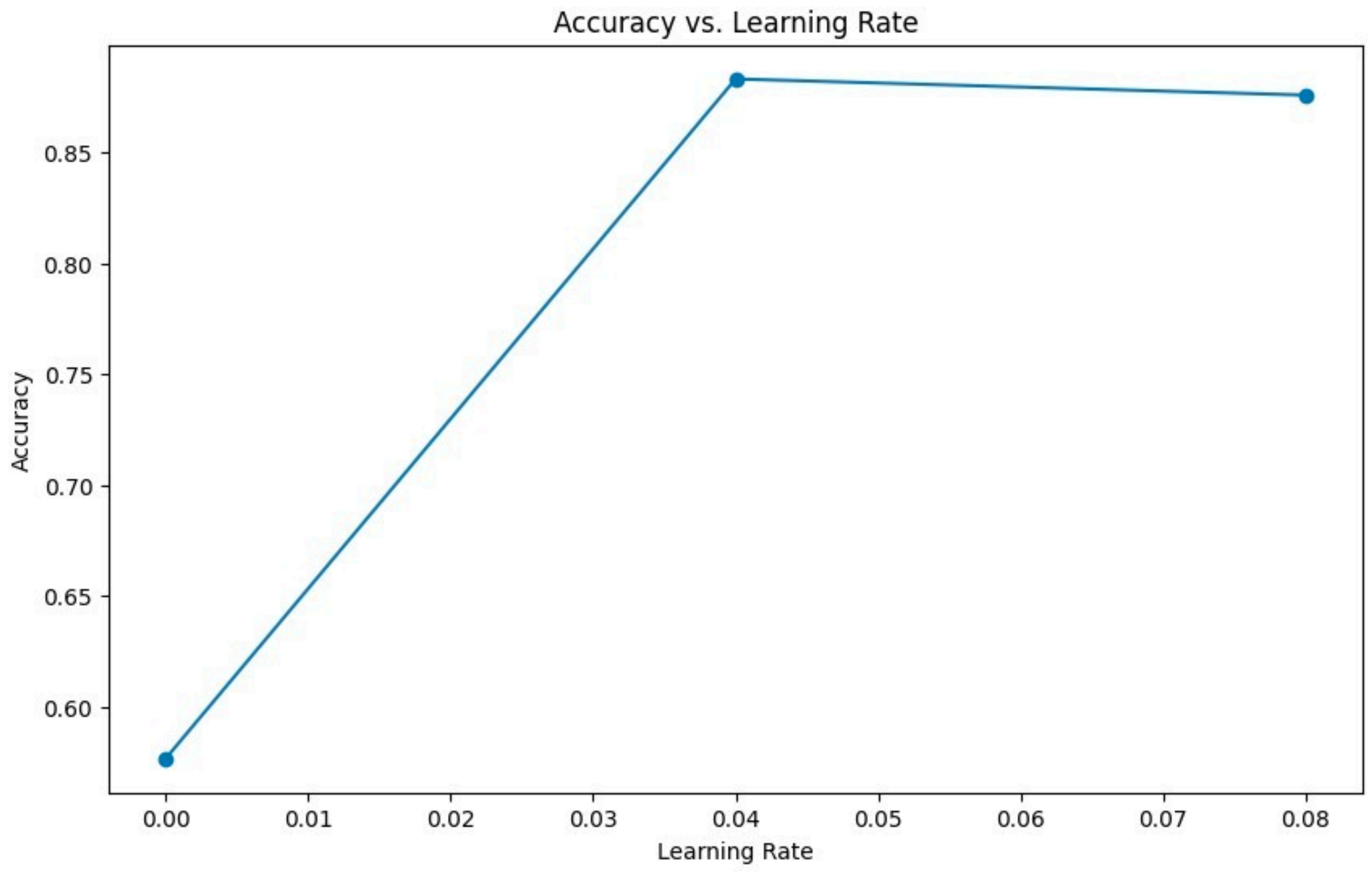
Hyper Parameters

- Learning rate
- Number of iterations
- Regularization Parameters

Hyper Parameters

Learning rate

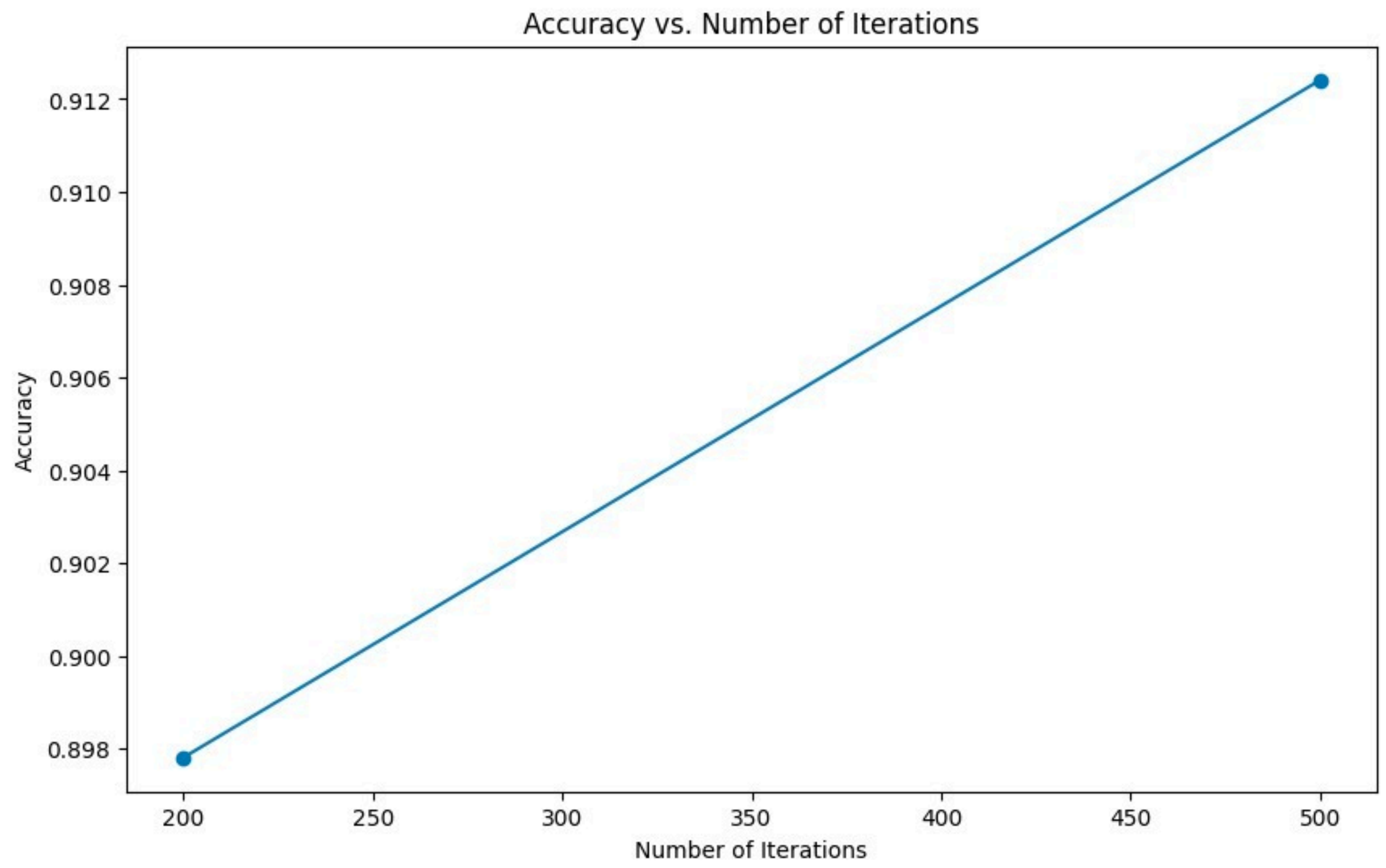
The learning rate is a hyperparameter that determines the step size taken during each iteration of the model training process, influencing the speed and stability of convergence.



Hyper Parameters

Number of iterations

The number of iterations is a hyperparameter that determines how many times the model will update its weights during training.

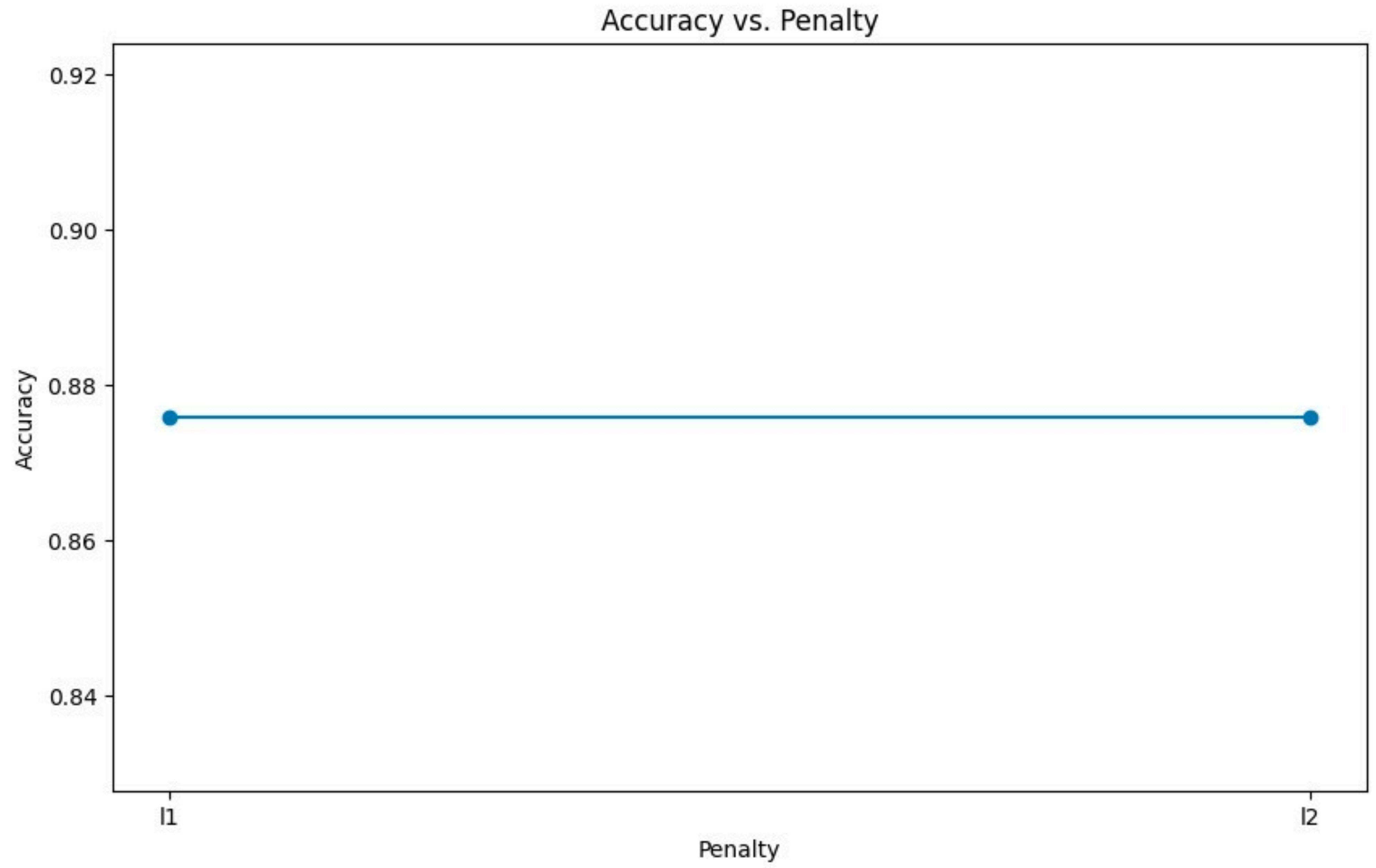


Hyper Parameters

Regularization

Regularization is a technique used in machine learning to prevent overfitting by adding a penalty term to the loss function, encouraging simpler models with smaller parameter values.

- **Penalty**: Penalty in regularization refers to the additional term added to the loss function

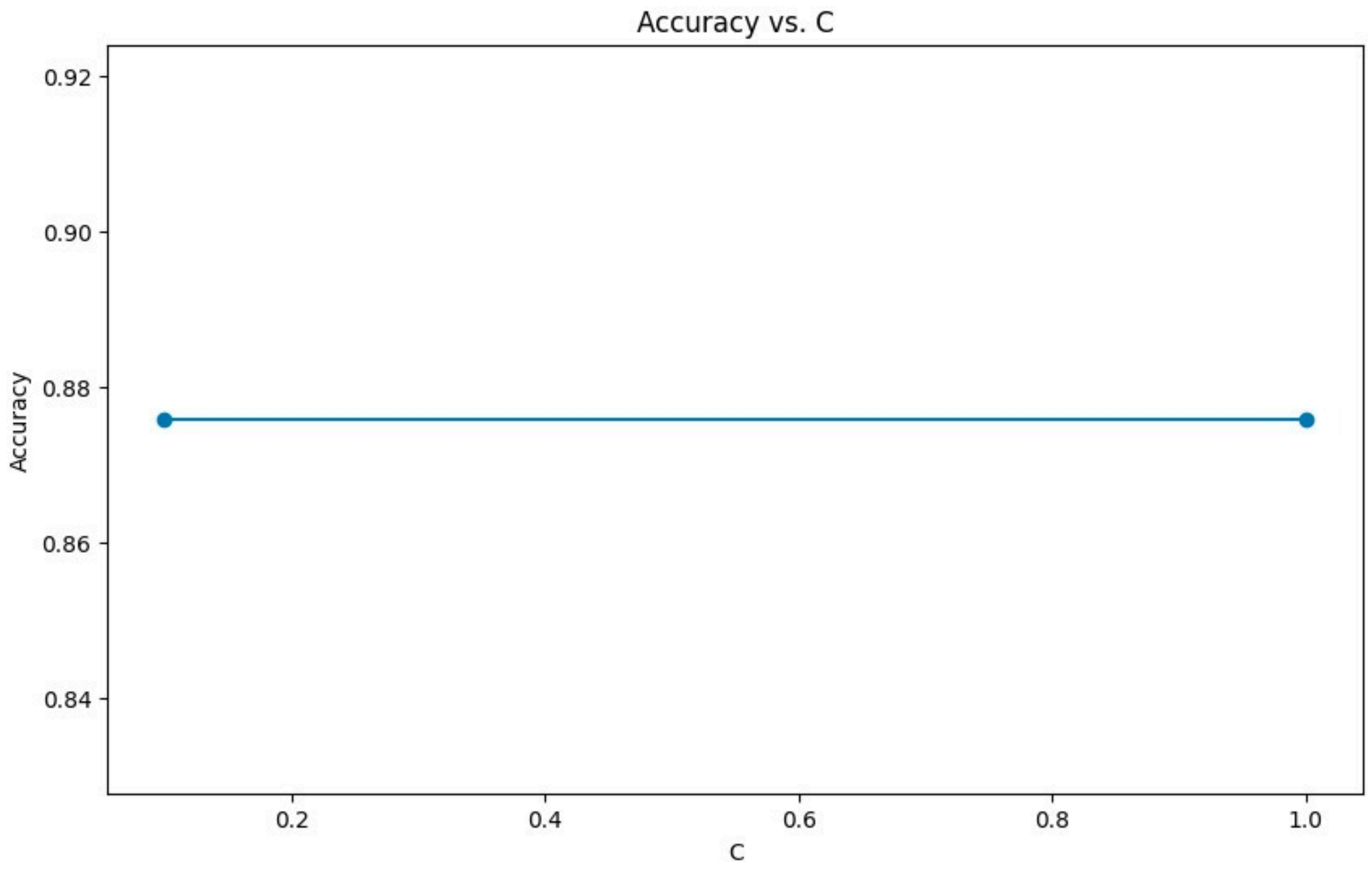


Hyper Parameters

Regularization

Regularization is a technique used in machine learning to prevent overfitting by adding a penalty term to the loss function, encouraging simpler models with smaller parameter values.

- **C**: hyperparameter that determines the inverse of the regularization strength, allowing control over the trade-off between fitting the training data and the extent of regularization

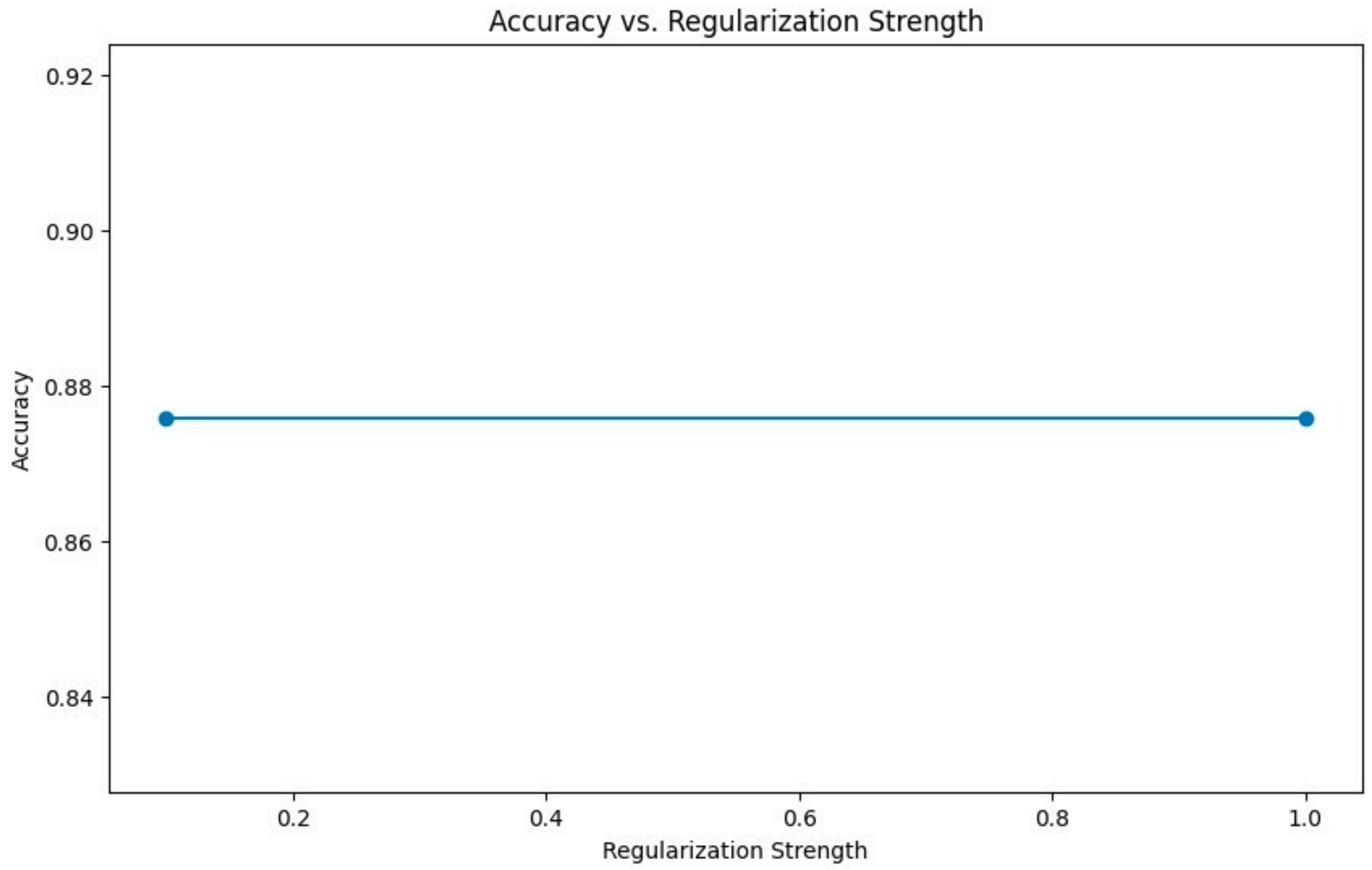


Hyper Parameters

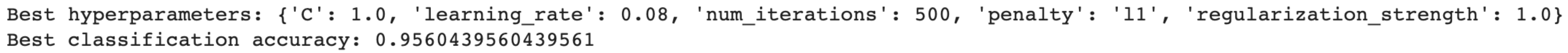
Regularization

Regularization is a technique used in machine learning to prevent overfitting by adding a penalty term to the loss function, encouraging simpler models with smaller parameter values.

- **Regularization strength**: refers to a hyperparameter that determines the intensity of the regularization effect applied to the model



Hyper Parameters

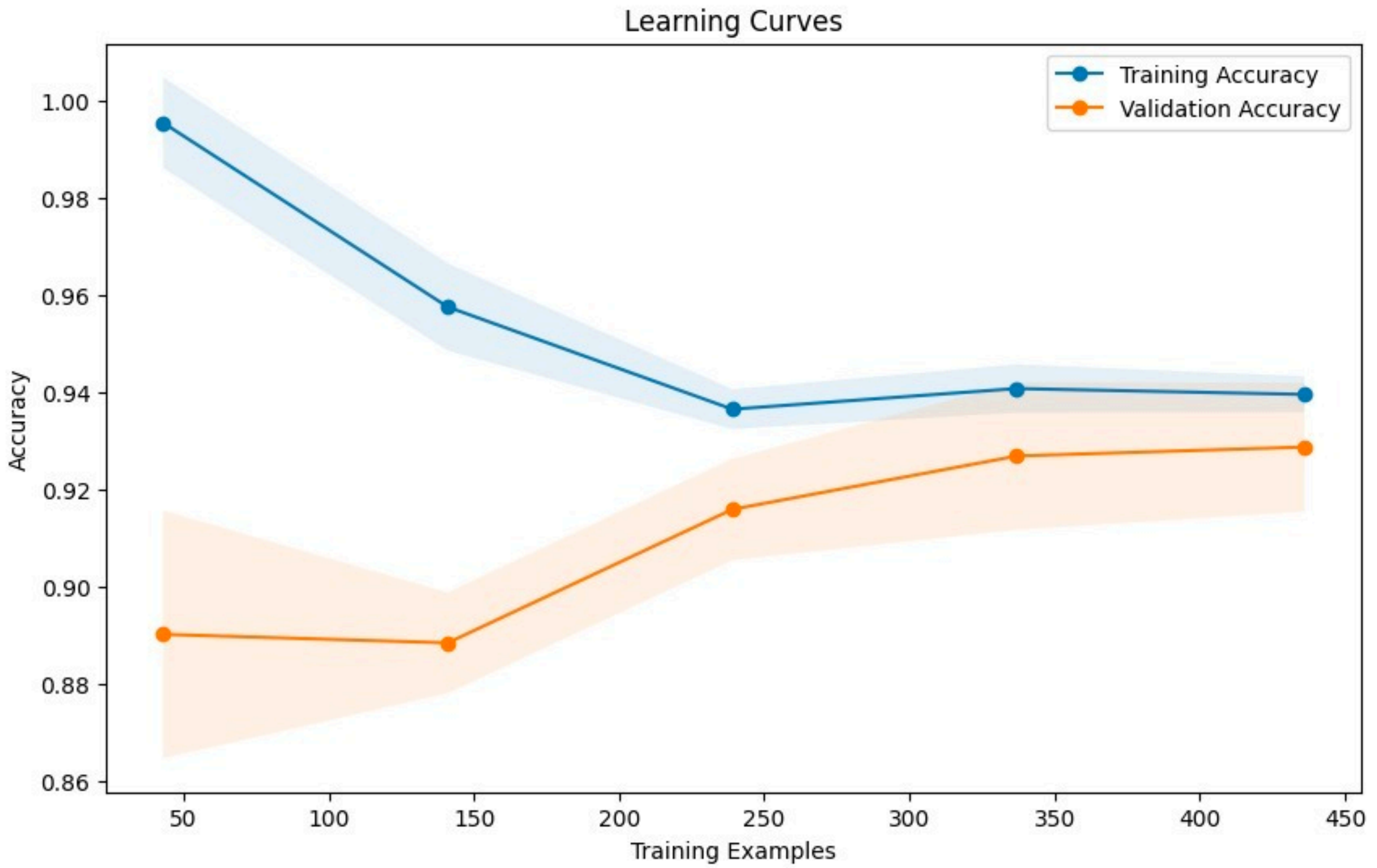


```
Best hyperparameters: {'C': 1.0, 'learning_rate': 0.08, 'num_iterations': 500, 'penalty': 'l1', 'regularization_strength': 1.0}
Best classification accuracy: 0.9560439560439561
```

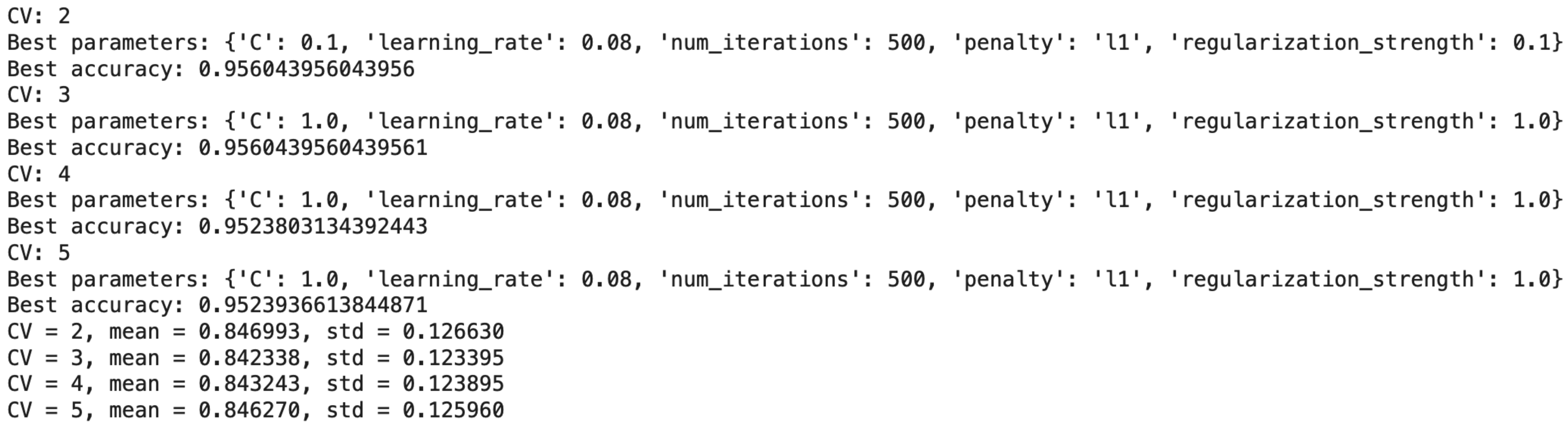
Cross Validation

Benefits of cross-validation include:

- Comprehensive model assessment: By dividing the dataset into multiple folds and iteratively training and evaluating the model, we obtain a more comprehensive understanding of its performance.
- Reducing dependence on a single split: Instead of relying on a single train-test split, cross-validation allows us to assess the model's performance across different subsets of the data, providing a more reliable evaluation.

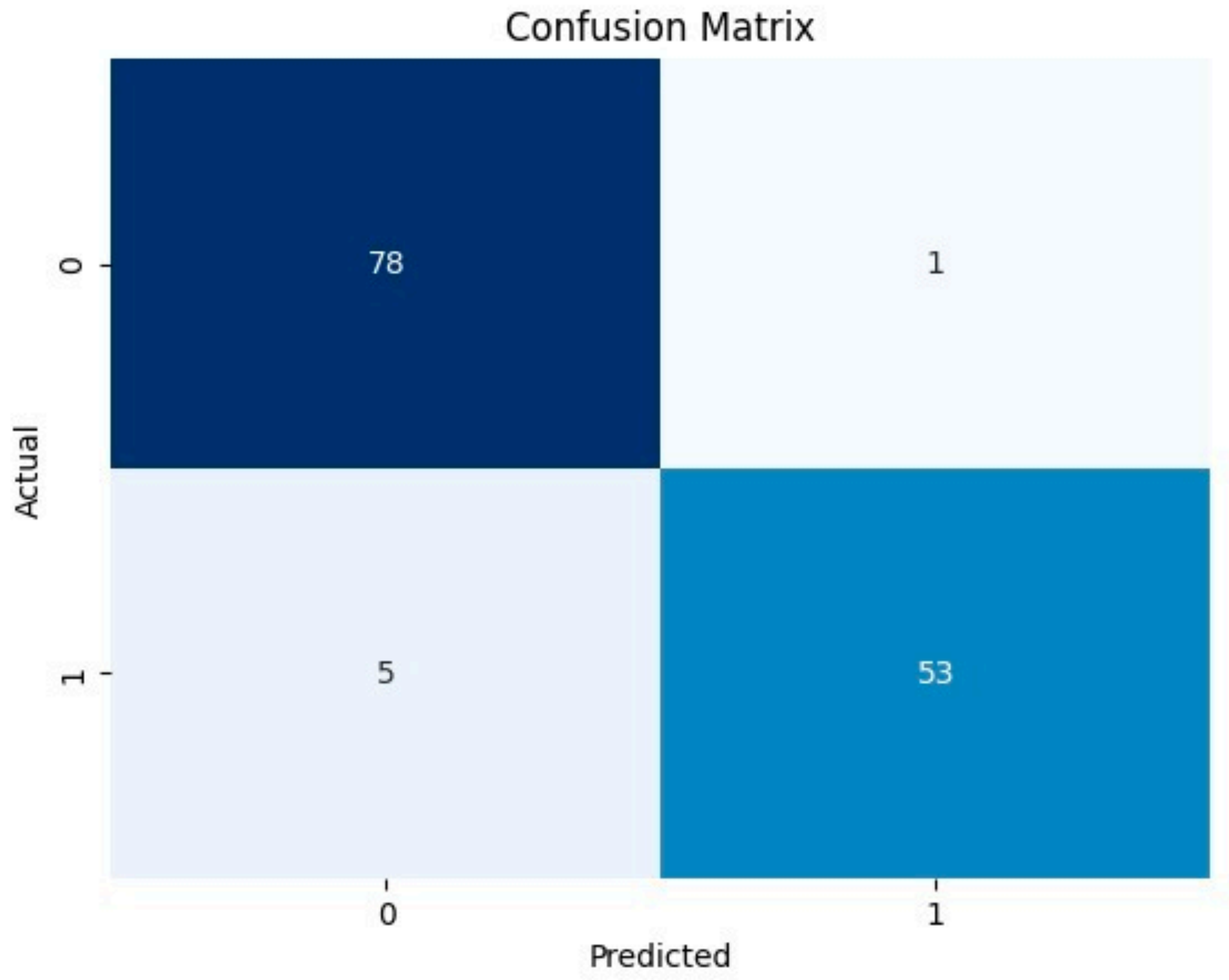


Results

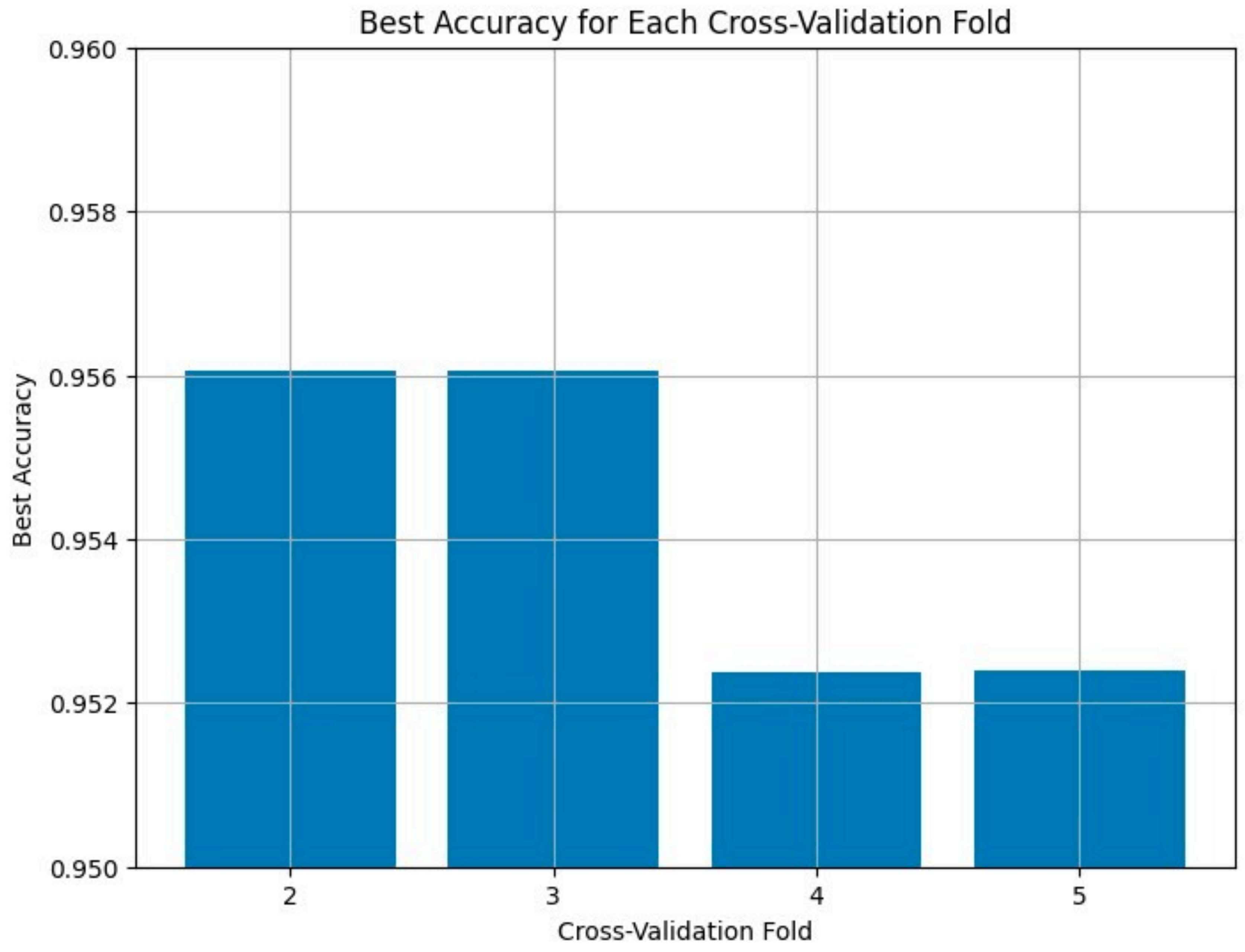


```
CV: 2
Best parameters: {'C': 0.1, 'learning_rate': 0.08, 'num_iterations': 500, 'penalty': 'l1', 'regularization_strength': 0.1}
Best accuracy: 0.956043956043956
CV: 3
Best parameters: {'C': 1.0, 'learning_rate': 0.08, 'num_iterations': 500, 'penalty': 'l1', 'regularization_strength': 1.0}
Best accuracy: 0.9560439560439561
CV: 4
Best parameters: {'C': 1.0, 'learning_rate': 0.08, 'num_iterations': 500, 'penalty': 'l1', 'regularization_strength': 1.0}
Best accuracy: 0.9523803134392443
CV: 5
Best parameters: {'C': 1.0, 'learning_rate': 0.08, 'num_iterations': 500, 'penalty': 'l1', 'regularization_strength': 1.0}
Best accuracy: 0.9523936613844871
CV = 2, mean = 0.846993, std = 0.126630
CV = 3, mean = 0.842338, std = 0.123395
CV = 4, mean = 0.843243, std = 0.123895
CV = 5, mean = 0.846270, std = 0.125960
```

Results



Actual \ Predicted	0	1
0	78	1
1	5	53



Thanks for your attention!!