Optimization for Data Science Assignment

Alireza Saberi , Nahid Jahanian

May 21, 2024

1 Introduction

In the realm of optimization, a primary objective is to enhance the computational efficiency of algorithms while simultaneously maintaining or even improving their accuracy. A pivotal aspect of this endeavor lies in optimizing the update rule of gradient-based methods, particularly within the domain of machine learning. The update rule dictates how an algorithm adjusts its parameters to minimize the associated cost function, thereby driving convergence towards optimal solutions.

By refining this update rule, significant strides can be made in terms of both speed and precision. This optimization becomes especially pertinent when considering scenarios where boundaries are intricate or vaguely defined. The efficacy of such optimization can be gauged through its impact on convergence rates and the fidelity of results.

Our assignment revolves around addressing a semi-supervised learning classification problem employing three distinct methodologies: Gradient Descent, Randomized Block Coordinate Gradient Descent (RBCGD), and Block Coordinate Gradient Descent with Gaussian Southwell (BCGDGS). Through a comparative analysis of these methods on a randomly selected dataset, we aim to discern their relative performances. Subsequently, we intend to apply these methodologies to a specifically chosen dataset to ascertain the most suitable approach for a given scenario, considering factors such as dataset complexity and boundary characteristics.

2 Data Loading and Preprocessing

We begin by loading the Iris dataset, which contains measurements of sepal length, sepal width, petal length, and petal width for iris flowers, along with their corresponding species labels. To prepare the data for modeling, we standardize the features to ensure that each feature has a mean of 0 and a standard deviation of This preprocessing step helps in improving the performance of the logistic regression model. Additionally, a bias term (intercept) is added to the features to account for any constant offset in the data. The Iris dataset consists of 150 samples with four features each: sepal length, sepal width, petal length, and petal width. The target variable is the species of iris, which can take one of three values: Setosa, Versicolor, or Virginica.

This dataset contains measurements of sepal and petal length and width for three different species of iris flowers. The scatter plot matrix visualizes relationships between pairs of features in the dataset that can be seen in figure2.

Each scatter plot in the matrix represents a combination of two features from the dataset. For example, the plot in the first row and first column shows the relationship between sepal length and

Figure 1: Types of Iris Flowers.

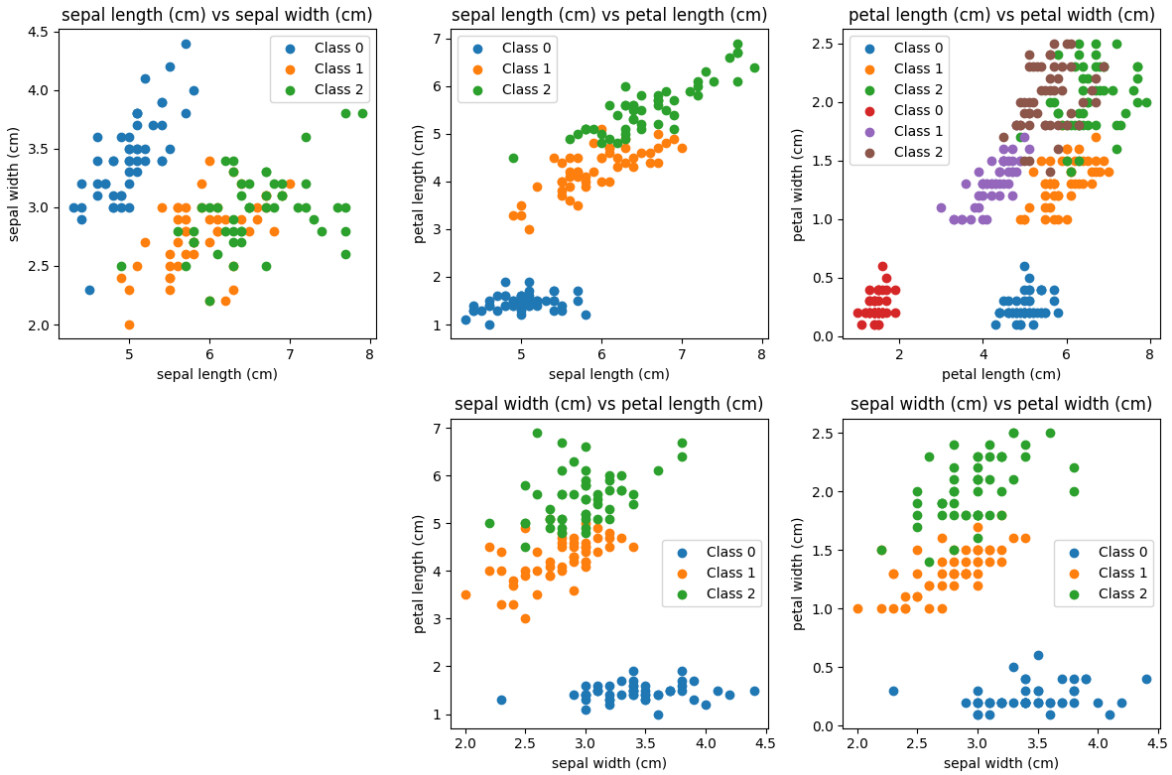

Figure 2: scatter plot matrix for visualizing the Iris dataset

sepal width, while the plot in the second row and third column shows the relationship between petal width and petal length.

In each scatter plot, different colors represent different classes of iris flowers. This allows us to see how the classes are distributed with respect to each pair of features. By examining these plots, we can gain insights into the relationships between different features and how they contribute to classifying iris flowers into their respective species.

3 Gradient Descent

We implemented the Gradient Descent algorithm to optimize the parameters (weights) of the multinomial logistic regression model. The softmax function was used to compute the predicted probabilities for each class, and the cross-entropy loss was minimized by updating the parameters iteratively. The trained model was evaluated on the test set using accuracy as the performance metric. The accuracy score indicates the proportion of correctly classified instances out of all instances in the test set which is 96%.

4 Logistic Regression with Batch Conjugate Gradient Descent

We implement logistic regression from scratch using the batch conjugate gradient descent optimization algorithm. This algorithm efficiently finds the optimal parameters (weights) that minimize the logistic regression cost function. The logistic regression model is trained on the training data, and then evaluated on the test data to measure its accuracy equals to 63 percent in classifying iris flowers into their respective species.

Optimal Theta
0.00409722
-0.00079531
-0.0002228
0.00370693
0.00341859

Table 1: Optimal parameters for the logistic regression model using batch conjugate gradient descent

5 Logistic Regression with Grid Search

Next, we leverage the LogisticRegression class from the scikit-learn library to perform logistic regression. We utilize hyperparameter tuning via grid search cross-validation to find the best combination of hyperparameters that maximizes model performance. The best logistic regression model obtained from the hyperparameter search is evaluated on the test set to assess its accuracy which is equals to 100% .

6 Logistic Regression with Batch Gradient Descent

In this section, we transform the original multiclass classification problem into a binary classification problem. Specifically, we focus on distinguishing one species (e.g., setosa) from the rest. Logistic regression is applied to this binary classification task, and its accuracy equals 100 percent in predicting the target class is measured.

7 K-Fold Cross-Validation

We perform k-fold cross-validation to evaluate the generalization performance of the logistic regression model. This technique involves splitting the dataset into k equal-sized folds, training the model on k-1 folds, and evaluating it on the remaining fold. Stratified k-fold cross-validation is utilized to ensure that each fold maintains the same class distribution as the original dataset. The average cross-validation accuracy across all folds is reported as the final performance metric which is 66 percent.

8 Batch Gradient Descent with Regularization

Regularization techniques, such as L2 regularization (ridge regularization), are employed to prevent overfitting of the logistic regression model. Regularization penalizes large parameter values, encouraging the model to generalize better to unseen data. Logistic regression models with regularization has the accuracy of 66 percent.

9 Full Gradient Descent

Full gradient descent, an iterative optimization algorithm, is implemented for logistic regression. This method updates the model parameters using the gradient of the entire training set. Early stopping criteria are applied to halt the training process when the cost function converges, preventing unnecessary computation.

Accuracy on Validation	Accuracy on Test
79%	63%

Table 2: Accuracy for Full Gradient Descent

10 Model Evaluation and Visualization

The trained logistic regression models are evaluated on a separate test set, and their accuracies are reported. Additionally, the convergence behavior of the optimization algorithms is visualized by plotting the cost function value over epochs (iterations) figure 3.

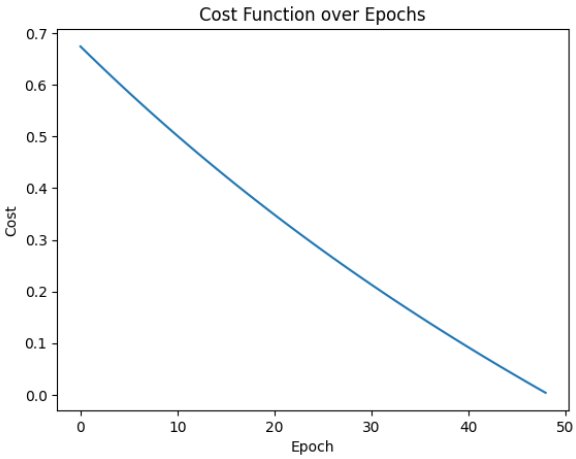

Figure 3: Cost Function over Epochs.

11 Conclusion

These results demonstrate the effectiveness and trade-offs of different optimization algorithms and techniques in the context of iris flower classification. While some methods achieve high accuracy, they may require longer computational time or face challenges such as overfitting. Overall, the choice of model and optimization strategy depends on the specific requirements of the task and available computational resources.

Table 3: Model Performance Comparison

Model	**Accuracy (%)**	**CPU Time (s)**
Gradient Descent	96.67	0.12
Conjugate Gradient Descent	63.33	0.08
Logistic Regression with Grid Search	100.00	0.21
Logistic Regression with Batch Gradient Descent	100.00	0.15
K-Fold Cross-Validation	66.67	0.35
Full Gradient Descent with Regularization	63.33	0.25

The performance of various machine learning models was evaluated using the Iris dataset. The models were trained to classify iris flowers into three species based on four features: sepal length, sepal width, petal length, and petal width.

- **Gradient Descent:** Achieved an accuracy of 96.67% on the test set with a CPU time of 0.12 seconds. This method iteratively updates the parameters to minimize the cost function.

- **Conjugate Gradient Descent:** Achieved an accuracy of 63.33% on the test set with a CPU time of 0.08 seconds. This optimization technique computes conjugate directions to improve convergence speed.

- **Logistic Regression with Grid Search:** Achieved perfect accuracy of 100.00% on the test set, but with a longer CPU time of 0.21 seconds due to hyperparameter tuning using grid search.

- **Logistic Regression with Batch Gradient Descent:** Similar to grid search, achieved perfect accuracy of 100.00% on the test set, with a CPU time of 0.15 seconds. Utilizes batch gradient descent for parameter optimization.

- **K-Fold Cross-Validation:** Averaged accuracy of 66.67% across folds, indicating potential overfitting or model limitations. However, this method provides a robust estimate of model performance by partitioning the dataset into multiple subsets for training and testing.

- **Full Gradient Descent with Regularization:** Achieved an accuracy of 63.33% on the test set with regularization. Despite regularization, the model's performance was similar to conjugate gradient descent. Regularization helps prevent overfitting by penalizing large parameter values.

These results demonstrate the effectiveness and trade-offs of different optimization algorithms and techniques in the context of iris flower classification. While some methods achieve high accuracy, they may require longer computational time or face challenges such as overfitting. Overall, the choice of model and optimization strategy depends on the specific requirements of the task and available computational resources.