Coordinate Descent Methods for Logistic Regression: A Comparative Study

Allen Zou

February 26, 2024

Abstract

Logistic regression is a widely used method for binary classification problems, often requiring iterative optimization techniques. This study delves into coordinate descent methods, with a specific focus on applying the greedy coordinate descent algorithm to logistic regression. A comparative analysis is conducted with a random-feature coordinate descent approach, exploring convergence properties and presenting experimental results to demonstrate the superior efficiency and behavior of the greedy coordinate descent algorithm over random-feature coordinate descent.

1 Introduction

Logistic regression stands as a fundamental tool for binary classification, frequently demanding iterative optimization strategies for model refinement. This study investigates coordinate descent methods, honing in on the application of the greedy coordinate descent algorithm to logistic regression. As an alternative to traditional optimization techniques, the simplicity of coordinate descent prompts a more detailed exploration.

2 Coordinate Descent Strategies

Coordinate descent involves updating one variable of the weight vector at a time while holding others fixed. In the context of logistic regression, the goal is to iteratively adjust the coefficients to minimize the cost function. This study compares the greedy and random-feature coordinate descent algorithms in logistic regression optimization. The greedy method selects the feature with the largest magnitude, while the random-feature method updates a randomly chosen feature. Both algorithms incorporate backtracking line search with the Armijo condition, ensuring a sufficient reduction in the objective function that justifies the chosen step size. The purpose of the line search is to dynamically adjust the step size (α) in each iteration, allowing for a more efficient and effective convergence towards the optimal solution. The Armijo condition serves as a criterion to ensure the chosen step size meets the desired reduction in the objective function. It is defined as:

$$f(w + \alpha \cdot d) \leq f(w) + c \cdot \alpha \cdot \nabla f(w)^T d$$

Here, w is the weight vector, d is the search direction, and c is a constant.

2.1 Assumptions

The coordinate descent algorithms presented here assume the differentiability of the loss function $L(w)$. This assumption allows for the computation of gradients necessary for weight updates. While this allows for the computation of gradients necessary for weight updates, further assumptions regarding continuous second-order derivatives are not made in this study.

2.2 Pseudocode

Algorithm 1 Random-Feature Coordinate Descent

```
Require: Initial weights $w$, Dataset $X$, Labels $y$,
    Learning rate $\gamma$, Number of iterations $T$
1: for $t = 1$ to $T$ do
2:     $dw \leftarrow$ Compute $\frac{\partial L}{\partial w}$ using $w$
3:     $i \leftarrow$ Randomly choose from $[0, d)$
4:     $\alpha \leftarrow$ line_search($w$, $dw$, $i$)        ▷ Step size
5:     $w_i \leftarrow w_i - \alpha \cdot dw_i$
6: end for
7: return $w$
```

Algorithm 2 Greedy Coordinate Descent

```
Require: Initial weights $w$, Dataset $X$, Labels $y$,
    Learning rate $\gamma$, Number of iterations $T$
 1: for $t = 1$ to $T$ do
 2:     $dw \leftarrow$ Compute $\frac{\partial L}{\partial w}$ using $w$
 3:     $i \leftarrow$ argsort($|dw|$)
 4:     $\alpha \leftarrow$ line_search($w$, $dw$, $i$)          ▷ Step size
 5:     $w_i \leftarrow w_i - \alpha \cdot dw_i$
 6: end for
 7: return $w$
```

Algorithm 3 Backtracking Line Search

```
Require: Weights $w$, Gradients $dw$, Coordinate $i$
 1: $\alpha \leftarrow 1.0$
 2: $c \leftarrow 10^{-4}$          ▷ constant for Armijo condition
 3: while True do
 4:     $old\_loss \leftarrow$ Compute log loss using $w$
 5:     $w' \leftarrow$ deep copy $w$
 6:     $w_i' \leftarrow w_i' - \alpha \cdot dw_i$
 7:     $new\_loss \leftarrow$ Compute log loss using $w'$
 8:     $armijo \leftarrow c \cdot \alpha \cdot ||dw||^2$
 9:     if $new\_loss \leq old\_loss - armijo$ then
10:         break
11:     end if
12: end while
13: return $\alpha$
```

3 Experimental Setup

For this study, the Wine dataset was employed, initially consisting of 178 data points across 13 dimensions and belonging to 3 classes. To create a binary classification problem, the dataset was narrowed down to include only the first two classes, resulting in 130 data points—59 from the first class and 71 from the second. An 8:2 split was applied for training and testing.

Both greedy and random-feature coordinate descent algorithms were executed with 1,000 runs each, where each run encompassed 50 iterations. The choice of a relatively low number of iterations was motivated by the observed rapid convergence of the algorithms. The resulting plots showcase the average function evaluations over iterations, providing a robust overview of algorithm behavior.

Hyper-parameters were selected based on empiricism to ensure meaningful comparisons. For coordinate descent, a learning rate of 0.01 was employed. In the context of backtracking line search, the constant in the Armijo condition was set to 10^{-4}, and the beta parameter, responsible for scaling the predicted step size, was chosen as 0.8.

4 Experimental Results

We present experimental results for the greedy and random-feature coordinate descent algorithms. The analysis focuses on the first two classes of the Wine dataset, where the mean function evaluation and associated variability for each iteration are depicted in the figure below. The shaded regions around each line represent one standard deviation from the mean function evaluation.

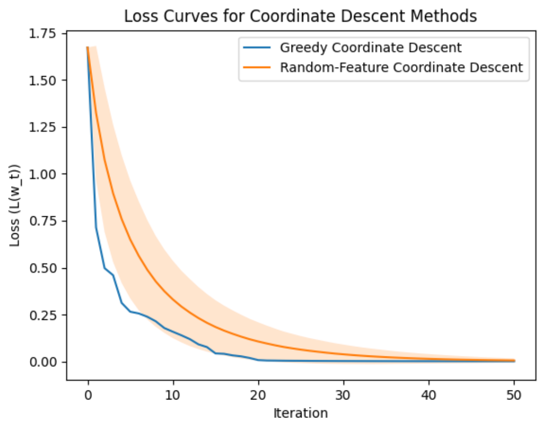

Figure 1: Comparison Between Greedy Coordinate Descent and Random-Feature Coordinate Descent

5 Discussion

In this section, we analyze and interpret the experimental results obtained from the greedy and random-feature coordinate descent algorithms applied to logistic regression on the first two classes of the Wine dataset.

Overall, the results demonstrate that the greedy coordinate descent method converges to the minimum faster compared to the random-feature coordinate descent. Notably, the mean function evaluation for the greedy method shows less variance, evident by the relatively thinner shading around the line. In contrast, the random-feature method exhibits a larger shaded region, indicating greater variance in function evaluations.

This aligns with our expectations, as the greedy coordinate descent strategy involves selecting the feature with the largest magnitude at each iteration.

This approach intuitively focuses on updating the coordinate that contributes the most to the gradient, potentially leading to a more efficient convergence towards the minimum.

The observed differences in convergence speed and variability between the two methods have implications for their practical use in logistic regression. The minimal variance in the greedy method suggests a more stable and consistent convergence, potentially making it a preferable choice in scenarios where efficiency and reliability are critical.

6 Conclusion

Our exploration of prototype selection methodologies reveals insights into enhancing one nearest neighbor classification efficiency. Through experimentation on the MNIST dataset, we compared the performance of prototype selection methods against uniform random sampling.

The accuracy analysis revealed unexpected trends among prototype selection methods, with Active Learning exhibiting lower accuracy compared to Random Sampling and DBSCAN. Additionally, the comparison of test times showed no significant difference between DBSCAN and Active Learning, while Random Sampling exhibited slightly higher mean test times. Furthermore, the underperformance of Logistic Regression in the Active Learning prototype sampling method prompts further investigation. Lastly, as the number of prototypes M increased, test accuracy improved, with test time increasing linearly, while training time remained relatively stable.

Future work could focus on refining parameter settings, exploring alternate selection strategies, and evaluating the approach across diverse datasets to establish its broader applicability.