Project 1

Thi Ai Nguyen Sałkowska, Jakub Seliga, Krzysztof Wolny

March 2025

1 Introduction

This report presents a project for the Advanced Machine Learning class, focusing on the implementation and evaluation of the Cyclic Coordinate Descent (CCD) algorithm for parameter estimation in regularized logistic regression with an L1 (lasso) penalty. The goal is to compare its performance with standard logistic regression without regularization. The report covers the methodology, an analysis of the correctness of the LogRegCCD algorithm, the impact of dataset characteristics on its performance, and a benchmark comparison with the standard logistic regression model.

2 Methodology

2.1 Selection and Generation of Datasets

To conduct the experiments, we selected four real-world datasets that address binary classification problems and contain numerical features. Each dataset used in the experiments had a relatively large number of features, ensuring that the number of variables was at least 50% of the number of observations. The datasets used are:

- **Tecator** – A dataset where the target variable is the binarized fat content of a meat sample, and the features represent its near-infrared absorbance spectrum. The dataset consists of 124 features and 240 observations (dataset1).
- **Sonar** – A dataset aimed at distinguishing between sonar signals reflected from a metal cylinder and those reflected from a rock. The original dataset contained 60 features and 208 observations. To increase the number of features, we introduced dummy variables by adding shuffled copies of the original features. The final version consists of 120 features and 208 observations (dataset2).
- **fri_c4_250_100** - This dataset is a binarized version of the original dataset, where the numeric target feature has been transformed into a two-class nominal target. Instances with a target value lower than the mean are classified as positive ('P'), while all other instances are classified as negative ('N'). The dataset consists of 250 instances, each described by 101 features (dataset3).
- **Semeion** - A dataset with data about handwritten digits, where the aim is to classify between the numbers 1 and 2. The dataset consists of 256 features and 318 observations (dataset4).

Additionally, we generated a synthetic dataset using the method described in Task 1. Its parameters include p, n, d, and g.

2.2 Details about algorithm implementation and applied optimizations

We consider the problem of logistic regression with ℓ_1-regularization. Using the coordinate-wise update rule, we get:

$$\beta_j \leftarrow \frac{S\left(\sum_{i=1}^{N} w_i x_{ij}(y_i - r_i - x_{ij}\tilde{\beta}_j), \lambda\right)}{\sum_{i=1}^{N} w_i x_{ij}^2}, \tag{1}$$

where $S(z, \lambda)$ is the soft-thresholding function:

$$S(z, \lambda) = \text{sign}(z)\max(|z| - \lambda, 0). \tag{2}$$

$$z_i = \tilde{\beta}_0 + x_i^T \tilde{\beta} + \frac{y_i - \tilde{p}(x_i)}{\tilde{p}(x_i)(1 - \tilde{p}(x_i))}, \quad \text{(working response)} \tag{3}$$

$$w_i = \tilde{p}(x_i)(1 - \tilde{p}(x_i)), \quad \text{(weights)} \tag{4}$$

We compute the solutions for a decreasing sequence of values for λ, starting for smallest value λ_{max} where $\tilde{\beta} = 0$. From equation (1) and soft-thresholding we can calculate that:

$$|\sum_{i=1}^{N} w_i x_{ij}(y_i - r_i - x_{ij}\tilde{\beta}_j)| < |\sum_{i=1}^{N} \tilde{(}p_i)(1 - \tilde{(}p_i))x_{ij}y_i| <= \frac{1}{4}|\sum_{i=1}^{N} x_{ij}y_i| < \lambda \tag{5}$$

We then, as in the research paper, select a minimum value $\lambda_{\text{min}} = \epsilon\lambda_{\text{max}}$, and construct a sequence of K values from λ_{max} to λ_{min} on the log scale.

We implemented early stopping mechanisms for likelihood function and for beta:

$$\mathcal{L}_{\text{new}} - \mathcal{L}_{\text{old}} < \epsilon \tag{6}$$

$$\max\left(|\beta - \beta_{\text{old}}|\right) < \epsilon \tag{7}$$

3 Discussion about correctness of the LogRegCCD algorithm

The LogRegCCD algorithm was tested in different ways to make sure it works as expected and follows the theory described in the paper. The main checks focused on how well the model fits the data, how the algorithm reacts to different values of λ, and how the coefficients and validation scores change when λ increases.

3.1 Performance of the algorithm at lambda=0

When $\lambda = 0$, no regularization is applied, so the algorithm should work like standard logistic regression without penalties. This means that all features should have nonzero coefficients unless they are completely irrelevant or redundant. Testing this case helps confirm that our algorithm correctly reproduces logistic regression when no regularization is used.
Selected results on validation datasets obtained with algorithm with $\lambda = 0$:

Measure	Tecator	Sonar	Synthetic (p=0.5, n=1000, d=12, g=0.95)
Accuracy score	0.625	0.524	0.855
Precision score	0.583	0.556	0.865
Recall score	0.350	0.455	0.838
F-measure score	0.438	0.500	0.851
Balanced Accuracy score	0.586	0.527	0.855
AUC-ROC score	0.586	0.527	0.855
AUC-PR score	0.475	0.538	0.805

Table 1: Performance Metrics for Different Datasets with λ=0

3.2 Likelihood function values and coefficient values depending on iteration

As the algorithm iterates, the negative log-likelihood (cost) decreases, indicating an improved fit. The coefficient values change gradually over iterations until they converge for a given λ. Plot below shows, how the cost changes in relation to step of the iteration and value of regularization parameter:

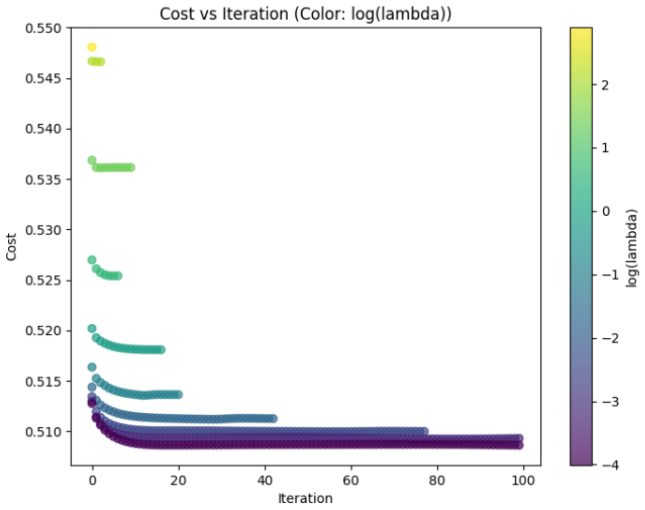

Figure 1: NLL in relation to iteration and value of lambda for training on Tecator dataset

The strength of regularization in the algorithm is controlled by λ parameter. The larger the value of λ parameter is, the stronger is the effect of regularization. Since the algorithm uses L1 regularization (lasso), increasing λ should reduce the size of the coefficients, eventually shrinking some of them to zero. If λ is large enough, all coefficients will become zero, leaving only the intercept. This is exactly what happens in our implementation of the algorithm - plots below show values of coefficients from models trained with different λ, on datasets described in section 2:

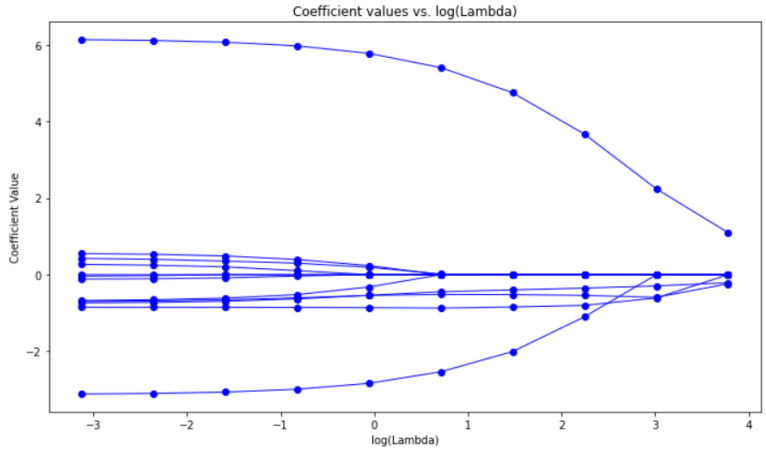

Figure 2: Coefficients of algorithm for different lambda values, trained on synthetic dataset (p = 0.5, n = 1000, d = 12, g = 0.95)

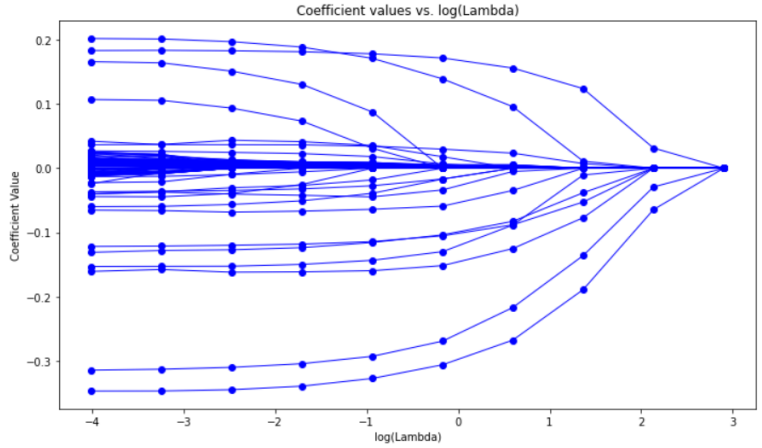

Figure 3: Coefficients of algorithm for different lambda values, trained on Tecator dataset

3.3 Comparison with ready implementation of logistic regression with L1 penalty

In comparison with the standard implementation of logistic regression with an L1 penalty, our LogRegCCD algorithm shows some differences. The main distinction is in the optimization method: our implementation uses Cyclic Coordinate Descent (CCD), while most built-in models mostly rely on coordinate descent or other solvers like SAGA.

Another difference is in computational efficiency. CCD updates one coefficient at a time, which can be more efficient for sparse solutions where many coefficients become zero. However, especially for large datasets, built-in implementations are far more optimized and therefore work much faster.

4 Impact of of synthetic dataset parameters on the performance of the algorithm

The algorithm's performance is most influenced by the parameter d, which is the number of dimensions in the synthetic dataset and thus the number of beta coefficients to estimate. Once the sample size n is large enough for the algorithm to capture the data patterns well, changes in n have little effect on the results.

Accuracy	Precision	Recall	F1	Balanced_Accuracy	p	n	d	g
0.55	0.50	0.33	0.40	0.53	0.5	100	3	0.9
0.60	0.60	0.60	0.60	0.60	0.5	100	10	0.9
0.45	0.50	0.55	0.52	0.44	0.5	100	15	0.9
0.53	0.54	0.47	0.50	0.53	0.5	300	3	0.9
0.60	0.61	0.72	0.66	0.59	0.5	300	10	0.9
0.32	0.32	0.33	0.33	0.32	0.5	300	15	0.9
0.49	0.50	0.57	0.53	0.49	0.5	500	3	0.9
0.46	0.48	0.57	0.52	0.46	0.5	500	10	0.9
0.45	0.46	0.47	0.47	0.45	0.5	500	15	0.9

Table 2: Performance Metrics for Different Synthetic Dataset Parameters

5 Benchmark of LogRegCCD with LogisticRegression algorithm

5.1 Performance of algorithms regarding different metrics

Measures	**fri_c4_250_100 (dataset3)**		**semeion (dataset4)**	
	Logistic Regression	**LogRegCCD**	**Logistic Regression**	**LogRegCCD**
Accuracy score	0.64	0.58	1.0	0.641
Precision score	0.625	0.526	1.0	0.604
Recall score	0.455	0.455	1.0	0.813
F-measure score	0.526	0.488	1.0	0.693
Balanced Accuracy score	0.620	0.567	1.0	0.640
AUC-ROC score	0.662	0.567	1.0	0.640
AUC-PR score	0.622	0.479	1.0	0.585

Table 3: Performance comparison between Logistic Regression and LogRegCCD

Measures	**semeion (dataset4)**		
	Logistic Regression	**Logistic Regression l1**	**LogRegCCD**
Accuracy score	1.0000	0.8413	0.8958
Precision score	1.0000	0.9214	0.9412
Recall score	1.0000	0.9214	0.8000
F-measure score	1.0000	0.9048	0.9865
Balanced Accuracy score	1.0000	0.9500	0.8821
AUC-ROC score	1.0000	0.8636	0.8821
AUC-PR score	1.0000	0.9167	0.8363

Table 4: Performance comparison between Logistic Regression and LogRegCCD no standarization

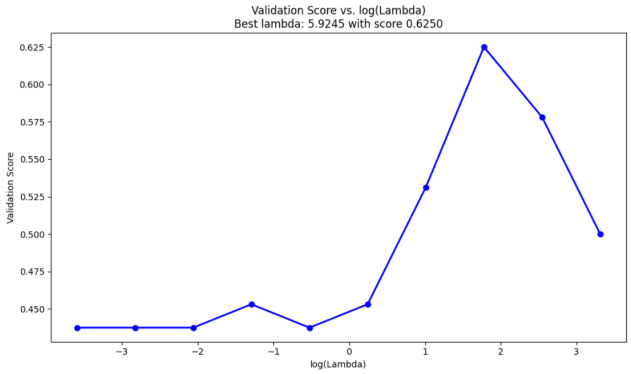

Figure 4: Accuracy scores obtained by implemented algorithm on validation dataset for different lambda values, trained on Semeion dataset

5.2 Values of coefficients obtained in both methods

Our weights differ from the ones we get from sklearn model. It can be caused by early stopping mechanisms, that we implemented and different method of calculation. For no standardized data we were getting better results, but very big weights. For standardized data we were getting worse results, but smaller weights.

	tecator		
β	**Logistic Regression (none)**	**Logistic Regression (l1)**	LogRegCCD
β_0	-19.6066	-0.1639	-0.3121
β_1	-20.3188	-0.1639	-0.3121
β_2	-20.9583	-0.1639	-0.3121
β_3	-21.5108	-0.1639	-0.3121
β_4	-21.9929	-0.1639	-0.3121
β_6	-22.4155	-0.1639	-0.3121

Table 5: First β values for Logistic Regression with penalty=none, penalty=l1 and LogRegCCD for standardized tecator dataset

References

1. Friedman, J. H., Hastie, T., & Tibshirani, R. (2010). Regularization Paths for Generalized Linear Models via Coordinate Descent. *Journal of Statistical Software, 33*(1), 1–22. https://doi.org/10.18637/jss.v033.i01

2. Scikit-learn: Logistic Regression Documentation

3. OpenML Tecator dataset1

4. OpenML Sonar dataset2

5. OpenML dataset3

6. OpenML Semeion dataset4