AML Project 1

Mikołaj Roguski
Mateusz Borowski
Szymon Matuszewski

March 2024

1 Introduction

In recent years, machine learning algorithms have gained significant traction across various domains, revolutionising how tasks are automated and decisions are made. Among these algorithms, logistic regression stands out as a fundamental method for binary classification tasks. Its simplicity, interpretability, and effectiveness make it a cornerstone in the toolkit of data scientists and machine learning practitioners.

The primary objective of this project is to implement and compare the performance of three optimisation algorithms SGD, Adam, and IWLS applied to logistic regression. Additionally, we aim to implement early-stopping rule with intent of preventing overfitting and improving model's ability to generalise to unseen data.

1.1 Project objectives

In this project we will implement:

1. a `LogisticRegression` classifier,
2. SGD, ADAM and IWLS optimisers,
3. a DataLoader,
4. an early stopping rule.

Additionally we will:

1. perform convergence analysis of our optimisers,
2. compare classification performance of our models with and without variable interactions,
3. compare classification performance of our models to popular implementations of LDA, QDA, Decision Tree and Random Forest.

2 Methodology

In this section we will provide details about our implementations, datasets we used and other insights into our methology.

2.1 Implementation Details

To assure no runtime errors in IWLS algorithm, which by default is sensitive to some datasets, we introduced two variables: λ – regularization parameter, explained below set to 10^{-10} and δ – minimum value which a diagonal element of W may reach set to 10^{-10}.

Lambda parameter reduces the chance of X^TWX being a singular matrix by replacing the expression for new weights

$$\beta^{new} = (X^TWX)^{-1}X^TWz$$

with

$$\beta^{new} = ((1-\lambda)X^TWX + \lambda I)^{-1}X^TWz$$

where p is the probability response of the model with weights from previous epoch and

$$z = X\beta^{old} + W^{-1}(y-p); W = diag(p(1-p))$$

2.2 Datasets

To evaluate the classification performance and perform convergence analysis we needed data. For this reason we've chosen 9 different datasets, 3 small ones (meaning they contain at most 10 features) and 6 larger ones (meaning they contain more than 10 features). Number of observations ranged from 1109 to 16599 and number of features ranged from 2 to 64.

Due to the fact that the aim of the project was to implement and compare the performance of different optimization algorithms for logistic regression, which is an algorithm that deals with a binary classification problem, we converted each of our datasets to the form that can be used in a binary classification task. To do so, for any given dataset the majority class was assigned to class 0 and the rest of the classes were assigned to class 1.

2.2.1 Small datasets

We obtained all the small datasets from `https://www.openml.org/`. Those datasets had no missing features, had at least 1000 observations and at most 8 features. We list their names and ids below:

1. phoneme – 1489
2. banknote verification – 1462
3. kin8nm – 807

2.2.2 Large datasets

Similarly to small datasets we obtained the large ones from `https://www.openml.org/`. These datasets have no missing features. We list their names below:

1. elevators
2. jm1
3. kdd_JapaneeseVowels
4. mfeat-karhunen
5. mfeat-zernike
6. pc1

2.3 Stopping Rule

To ensure no overtraining and limit duration of an experiment we implemented an early stopping rule.

This rule keeps track of the best log likelihood and model seen during training, and if no improvements are made in a set number of epochs it terminates the training returning the best model. As a bonus we combined this early stopping rule with simple iteration limit.

All our experiments were concluded with 500 max iterations and patience of 5 iterations unless stated otherwise.

2.4 Training

Before the training began, the data was checked for missing rows and if any were found they were dropped from the dataset. Furthermore, since we came across a few features, which consisted almost exclusively of zeros, the data was checked for such sparse features and if any were detected, they were dropped from the dataset (threshold was set at 85%). Finally each feature was separately normalized using `StandardScaler` from scikit-learn package.

The learning rate for SGD was set to 0.001 and to 0.01 for Adam. The IWLS method did not require specifying any hiperparameters. Additionally we set exponential decay rate for the first moment estimate to 0.9, exponential decay rate for the second moment estimate to 0.999 and epsilon to 10^{-8}. Although we implemented SGD and Adam optimizers in a way that lets the user specify the size of a mini-batch, we set it to 1 in all of our experiments in order for the comparison to be fair.

The implemented optimizers are compared against LDA, QDA, Decision Tree, and Random Forest. The training of the aforementioned methods was conducted in the following way:

- LDA – implementation provided by scikit-learn package with default values of hyperparameters;
- QDA – implementation provided by scikit-learn package with `reg_param` set to 0.1 and `tol` set to 10^{-8};
- Decision Tree – implementation provided by scikit-learn package with 5-fold cross-validation performed on the `max_depth` parameter taking the values $4, 6, 10, 16,$ `None`;
- Random Forest – implementation provided by scikit-learn package with 5-fold cross-validation performed on the `max_depth` parameter taking the values $4, 6, 10, 16,$ `None`.

During evaluation, each algorithm was run 7 times, each time with a different seed in order to ensure reliability of the results by capturing the variability introduced by random initialization or sampling processes.

3 Convergence analysis

To evaluate convergence of our algorithms we tracked log likelihood calculated after each training epoch. The results are shown in Figure 1 and training balanced accuracy in Figure 2.

We can see, that IWLS converges extremely fast, in just 2 or 3 iterations, while Adam and SGD often require 10 to 15 epochs, never reaching convergence on certain datasets like phoneme of elevators. In general we can a decrease of Log-Likelihood loss on every dataset as the training progresses.

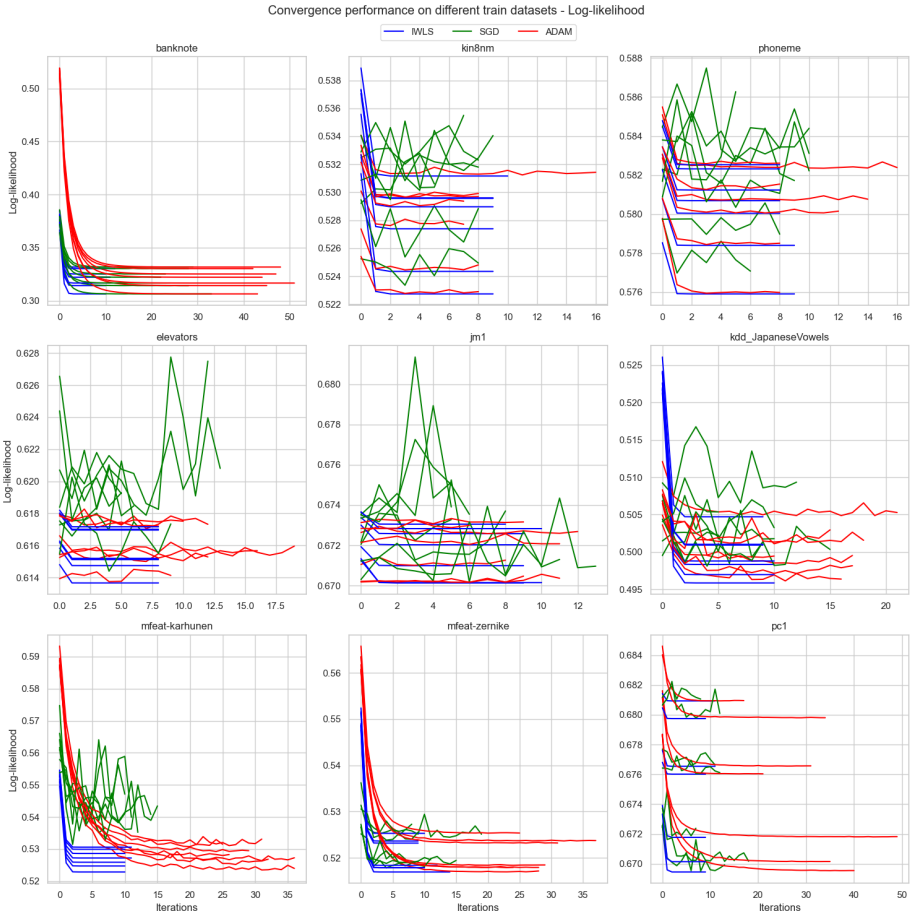

Figure 1: Relationship between log-likelihood loss and training progress on all datasets after each iteration.

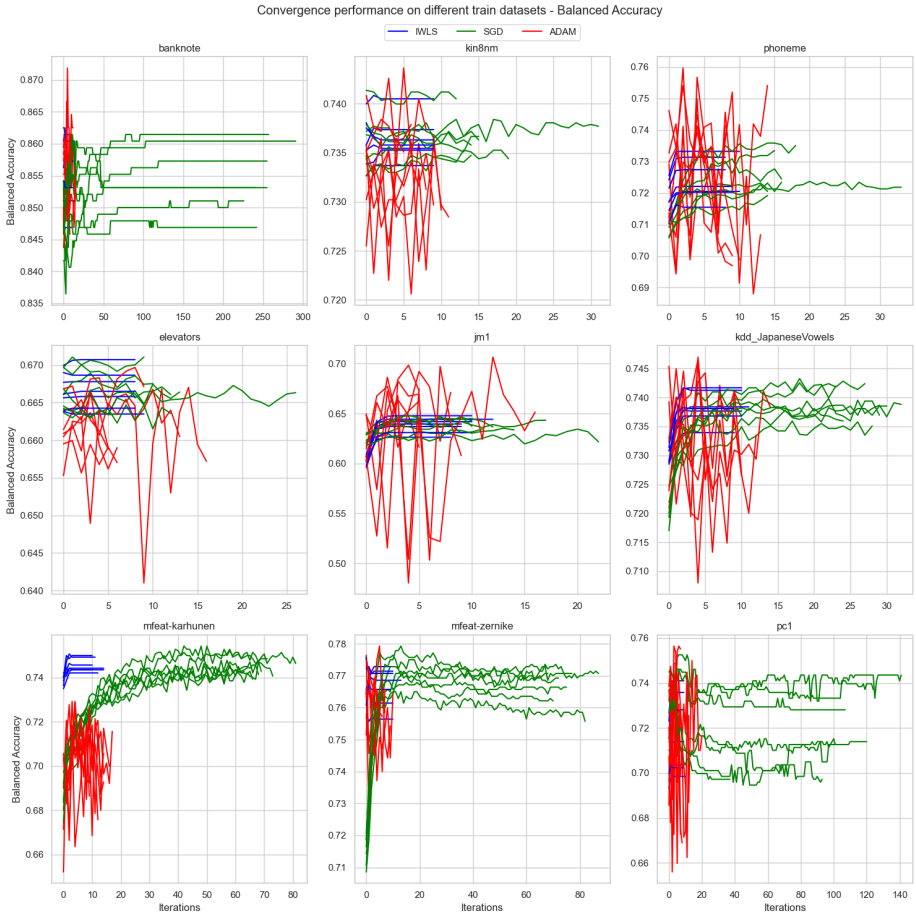

Figure 2: Training balanced accuracy progress with training on all datasets after each iteration.

4 Comparison of classification performance

We tested the optimizers for logistic regression that we wrote against LDA, QDA, Decision Tree, and Random Forest algorithms.

Each of the optimizers that we have implemented achieves similar performance on all datasets, which is visible in Figure 3. Moreover all of them achieve comparable results with the banchmark methods from the scikit-learn library on all datasets that we have chosen, which is visible in Figure 4. Logistic regression even outperforms other methods on elevators, jm1, and pc1 datasets.

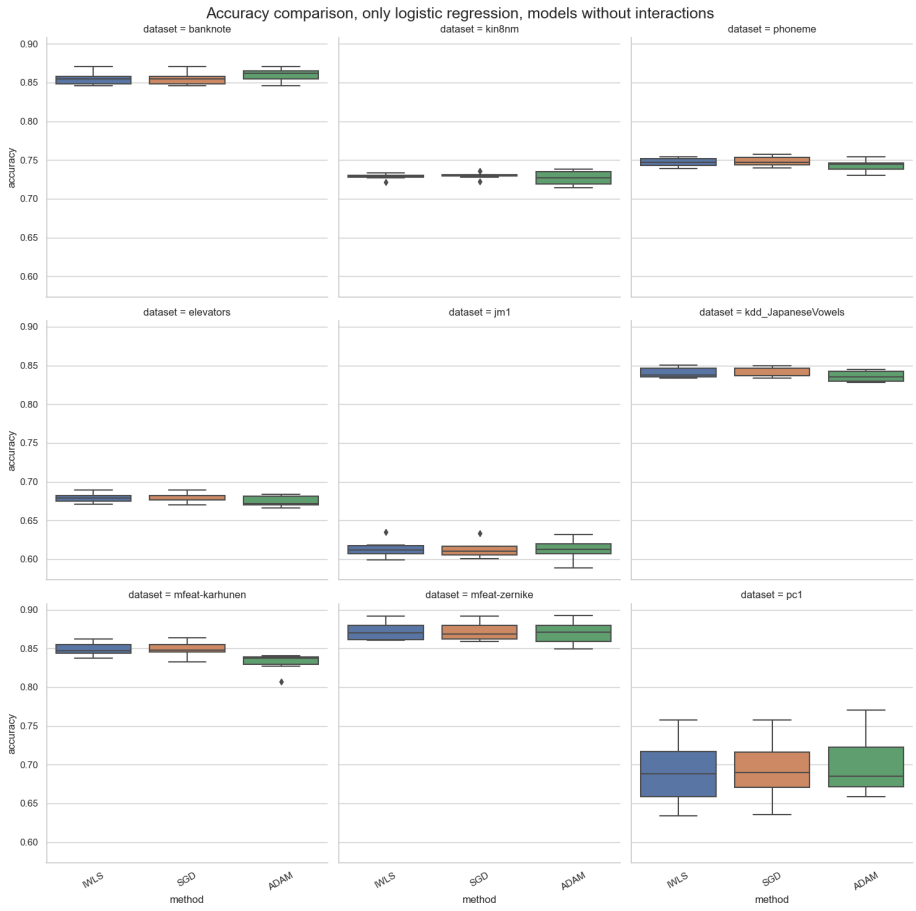

Figure 3: Comparison of testing accuracy of logistic regression models on all sets without feature interactions.

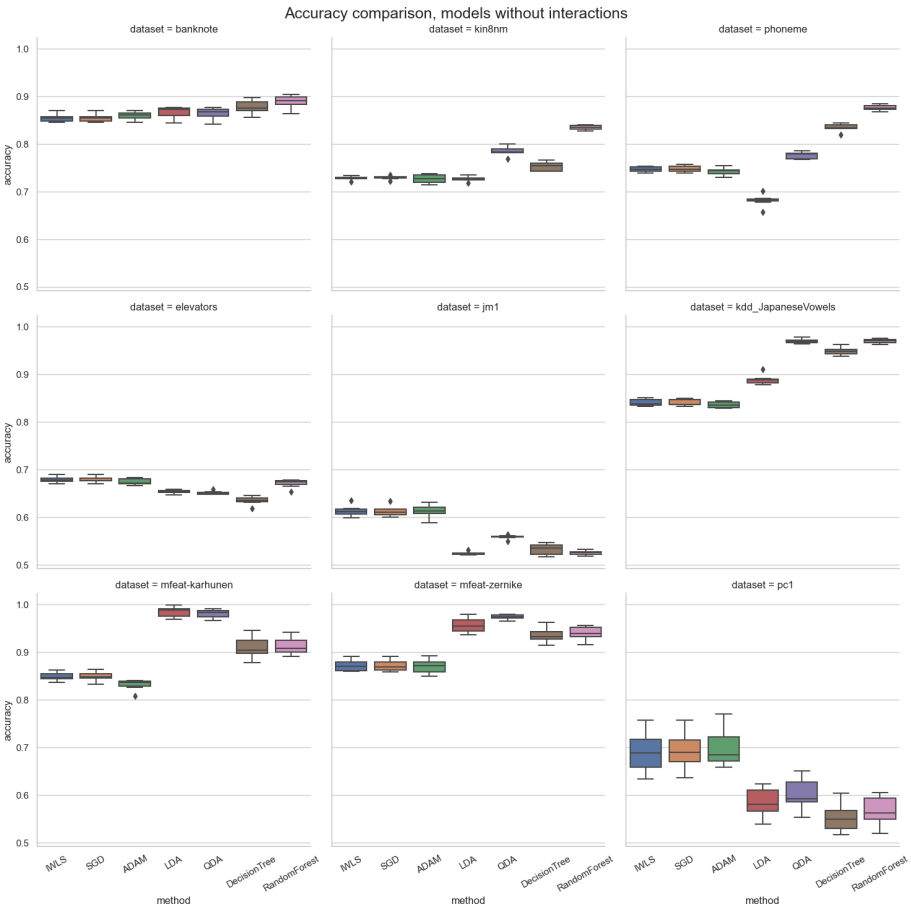

Figure 4: Comparison of testing accuracy of different models on all sets without feature interactions.

5 Impact of interactions on classification performance

Since logistic regression model assumes linear dependence of log odds on features it might make learning higher order and more complex relationships challenging, to mitigate this we include pairwise products of features. We would expect such a modification to raise the accuracy of our models, however this can be only observed to a significant degree on kin8nm dataset and to a lesser degree on phoneme dataset. What is interesting is that adding interactions significantly improved the accuracy of random forest on each tested dataset except phoneme. Those results are portrayed in Figure 5.

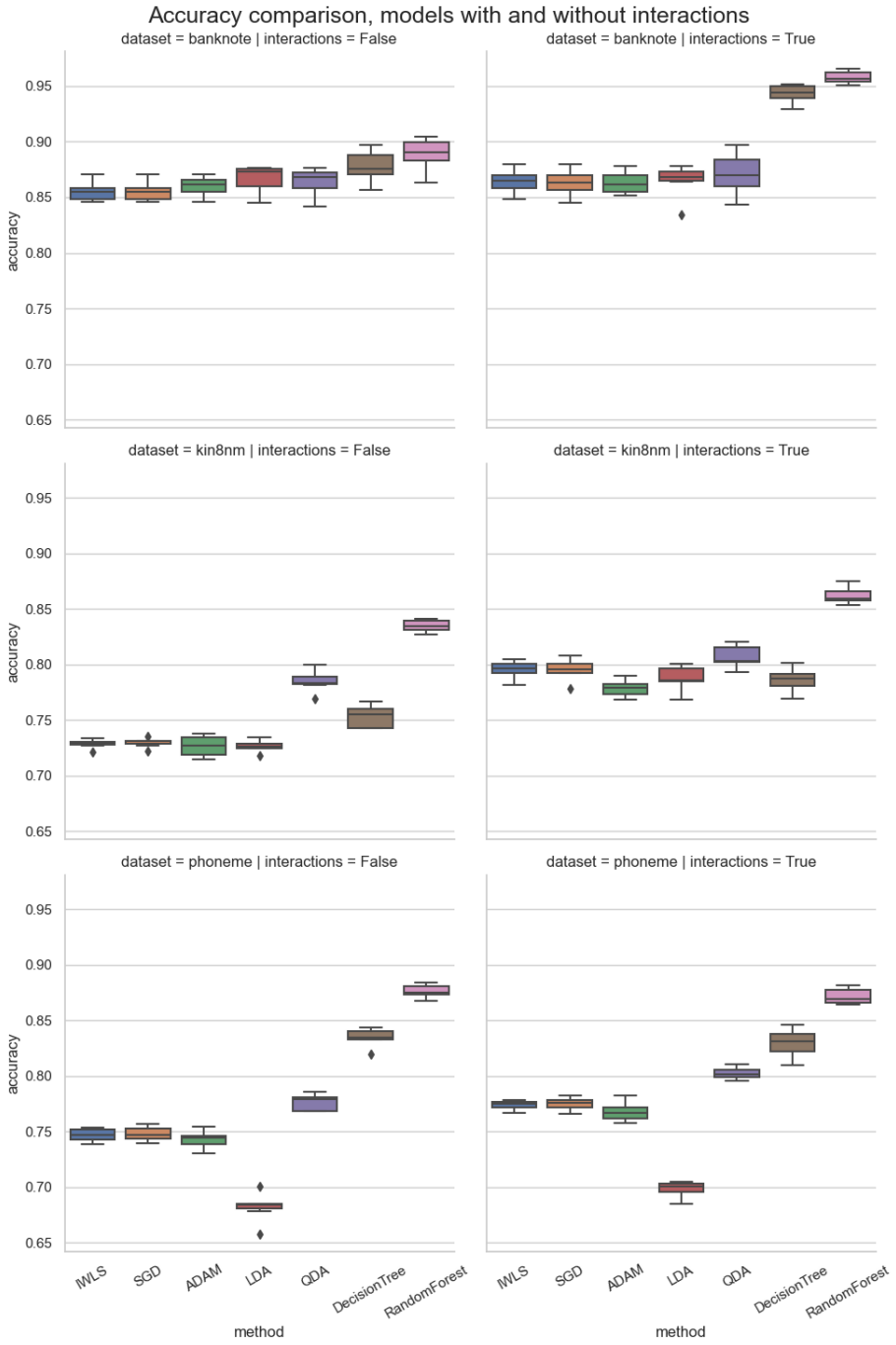

Figure 5: Comparison of testing accuracy of models with and without feature interactions on small datasets.

6 Discussion

In summary, in this project we implemented a logistic regression model, Iterative reWeighted Least Squares optimizer, Stochastig Gradient Descent optimizer and Adam optimizer. We also impremented a log likelihood improvement sensitive stopping rule, experimented with convergence of our optimizers, importance of addition of pariwise products of variables, compared our models with popular implementations of Decision Tree, Random Forest, and Linear and Quadratic Discriminant Analysis models.
We have obtained satisfactory results on model convergence with IWLS algorithm reaching an optimum in just 2-3 epochs while Adam and SGD reached good results in around 10-15 epochs.
We achieved satisfactory classification results on all datasets, even beating Random forest on pc1 dataset by a significant margin.