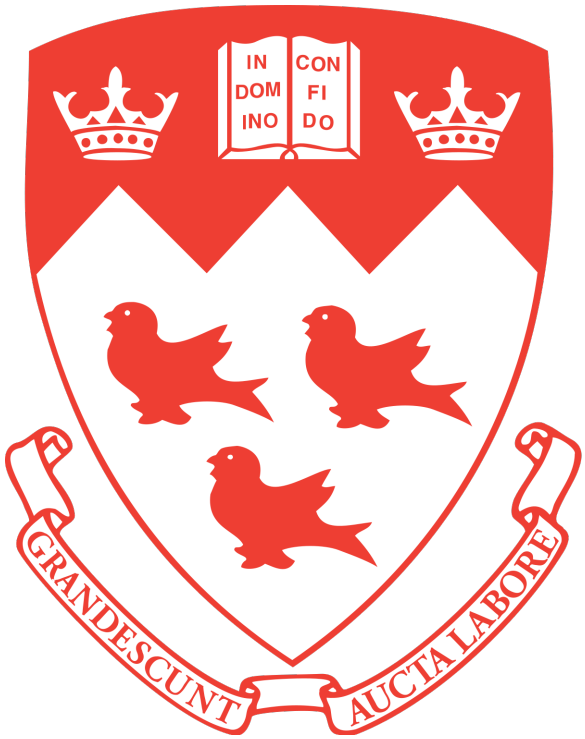


Comp 551: Applied Machine Learning
Mini-Project 2: Multi-Class Logistic Regression and Gradient Descent
Group 49

Doreen Duoduaah - 260995803
Ameer Ibrahim Osman - 260682723
Ella Malone - 260761280

November 22, 2020

1 Abstract

The objective of this project is to implement multi-class Logistic Regression (Softmax Regression) and compare its performance against other classification algorithms, namely K-NN and Decision Trees. Furthermore, we will be optimizing our Softmax Regression model using our implementation of Mini-Batch Stochastic Gradient Descent with momentum and regularization. These models will be trained on two distinct datasets, Scikit-Learn's Digits dataset and OpenML's Cardiotocography dataset. The goal is to implement Softmax Regression and Mini-Batch Stochastic Gradient Descent from scratch. In our results we found that our Regressor has an accuracy of 92.7% for the Digits dataset, and 77.8% for the Cardiotocography dataset.

1.1 Abbreviation and Notations

LR: Logistic Regression
GD: Gradient Descent
Multi-Class Logistic Regression: Equivalent to Softmax Regression
K-NN: K-Nearest-Neighbours
LR: Learning Rate
BS: Batch Size

2 Introduction

In this mini-project, the team is assigned to implement Softmax Regression from scratch, train it on two datasets, and compare it's performance against K-NN and Decision Trees. The goal is to train these models and find their optimal hyper-parameters such that they predict the classes of the two datasets. We utilize the following notable python libraries: Numpy, Scikit-learn, and Matplotlib. We also utilize Pycharm to provide the results in this report. We would also like to note that the results may vary between Jupyter Notebook, Google Colab, and Pycharm as in our experience.

The two datasets used in this project are Scikit-learn's Digits and OpenML's Cardiotocography datasets. The Digits dataset is of shape (1797, 64). The Cardiotocography dataset is of shape (2126, 35). The Carditocography dataset used is version 2.

Our most important findings are that our Softmax Regression Model achieved an accuracy of 92.7 % for dataset 1 and 77.8% for dataset 2. We found that momentum greatly reduces oscillations within the cost function. We also found that implementing 5-fold cross validation within Mini-Batch Stochastic Gradient descent not only greatly reduces the runtime, but also improves the accuracy of our model. Moreover, we found that implementing L1 & L2 Regularization did not significantly improve our model. We also found the optimal hyper-parameters for our model for datasets 1 & 2. Finally, we found that the only case our model outperforms off the shelf K-NN and Decision Trees is for dataset 1 , where Decision Trees performed worse than our model.

3 Dataset Source, Size, & Processing

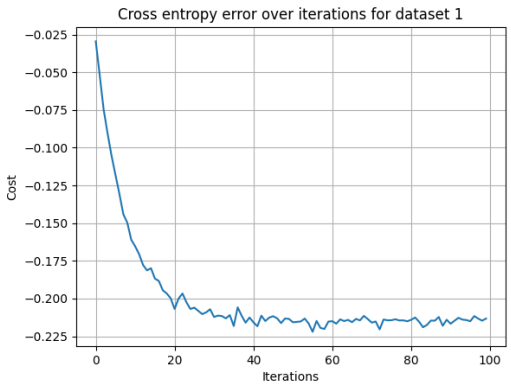
The Digits dataset is from Scikit-learn's datasets. It consists of 1797 instances of features that are 8x8 images corresponding to numbers 0 to 9. The goal with this dataset is to classify numbers 0 to 9. This dataset was pre-processed by Scikit-learn upon import. No processing was necessary as there were no missing or NaN entries. Hereinafter we will refer to this dataset as dataset 1.

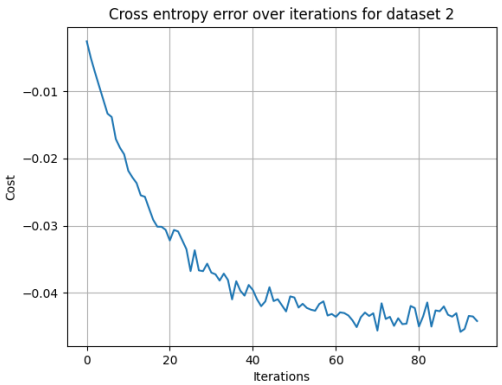
The Cardiotocography dataset is from OpenML's database. It consists of 2126 instances of 35 different medical features corresponding to fetal heartbeat parameters. The classes correspond to 3 fetal states (N, S, P). The preprocessing was performed as follows. Firstly, non-numeric entries such as the dtype tag were removed. Then due to an overflow in the softmax function, we Min-Max normalize the dataset. Hereinafter we will refer to this dataset as dataset 2.

4 Discussion & Results

4.1 Softmax Regression Discussion & Results

We implemented Softmax Regression and Mini-Batch Stochastic Gradient Descent with momentum from scratch. Its performance is a function of the learning rate, momentum, batch size, and number of iterations. These parameters influence the convergence of our model towards the optimal weights. Our implementation utilizes a termination condition that is dependent on the 5-fold cross validation accuracy. It checks that the accuracy in the last 20 iterations of cross validation are strictly increasing, otherwise it terminates. This returns the model with the optimal cross-validation accuracy, not the model at the end of gradient descent. Moreover, it greatly decreases the convergence speed especially when the maximum number of iterations is 10,000. The cost function from gradient descent with optimal parameters for both datasets are shown below. The run time, accuracy, and cost for dataset 1 respectively are 3.7 seconds, 92.7%, and -0.214. For dataset 2 these values respectively are and 4.9 seconds, 77.8%, and -0.046. Moreover, we implemented L1 & L2 Regularization with a lambda of 0.1, but found that it did not improve the performance of our model.





4.2 Hyper-parameter Optimization Discussion & Results

We implemented a Hyper-parameter optimizer that utilizes 5-fold cross validation to find the optimal parameters that result in the highest performance in terms of accuracy. This utilizes a 80% 20% train-test split. The runtime performance of the hyper-parameter optimizer along with the ranges utilized are shown in the table below.

Dataset	BS Range	BS step size	LR Range	LR step size	Momentum Range	Momentum Step Size	Run Time(s)
1	50 - 1797	100	0.01 - 0.1	0.018	0.9 - 0.99	0.018	824.6
2	50 - 2126	100	0.01 - 0.1	0.018	0.9 - 0.99	0.018	991.2

It is noteworthy that a batch size of 1 provides the best accuracy, although it has a trade-off with runtime. In that case Mini-Batch Gradient Descent reduces to Gradient Descent. Thus, we did not consider a batch size of 1. The optimal hyper parameters found for each dataset are displayed in the table below, along with our models performance.

Dataset	Batch Size	Learning Rate	Momentum	Run Time (s)	Accuracy
1	950	0.027	0.9	3.7	92.7%
2	50	0.01	0.9	4.9	77.8%

We also plot the Validation accuracy, Training accuracy, and cost for each hyper-parameter for each dataset in the appendix. These figures illustrate the evolution of the hyper-parameters in our hyper-parameter optimizer. These figures re-affirm our optimal hyper-parameter values discussed here.

4.3 Comparison against another classifier Discussion & Results

We compared our Softmax Regressor against K-NN and Decision tree Regressors. Firstly, we found the optimal K and Tree depth using our hyper-parameter optimizer that utilizes 5-fold-cross validation. These optimal parameters for dataset 1 are K = 1 and tree depth of 20 with accuracies of 98.9% and 91.7% respectively. For dataset 2, the optimal parameters are K = 3 and tree depth of 5 with accuracies of 99.1% and 98.6% respectively. As expected, off the shelf implementations provide equal or higher accuracies for datasets 1 & 2 as compared to our accuracies of 92.7% and 77.8% for the datasets respectively. It is noteworthy that for dataset 1, the decision tree was unable to perform better than our Softmax Regressor.

5 Conclusion

In this project, we implemented Softmax Regression with Momentum and Mini-Batch Stochastic Gradient Descent. We've deployed our model on two distinct datasets, the Digits dataset and the Cardiotocography dataset. In the Digits dataset, we predict digits 0-9 based on an input 8x8 pixel image. In the Cardiotocography dataset, we predict the fetal state as one of three classes (N, S, P). We've implemented a version of Mini-Batch Stochastic Gradient Descent that utilizes 5-fold cross-validation to improve our models performance with respect to the various hyper-parameters involved. Our model achieved accuracies of 92.7% & 77.8% for datasets 1 & 2 respectively. We also implemented L2 & L1 Regularization but found no significant improvement in performance. This could be due to the fact that in the weight space, there is already a unique global minima.

We also utilized Scikit-learn's off the shelf implementation of K-NN and Decision trees to compare the performance of our model. We used our own hyper-parameter optimizer to find the optimal model parameters, and found the accuracies of these models. We found that It generally outperforms our model, except for the decision tree in dataset 1, as our model achieved a higher prediction accuracy.

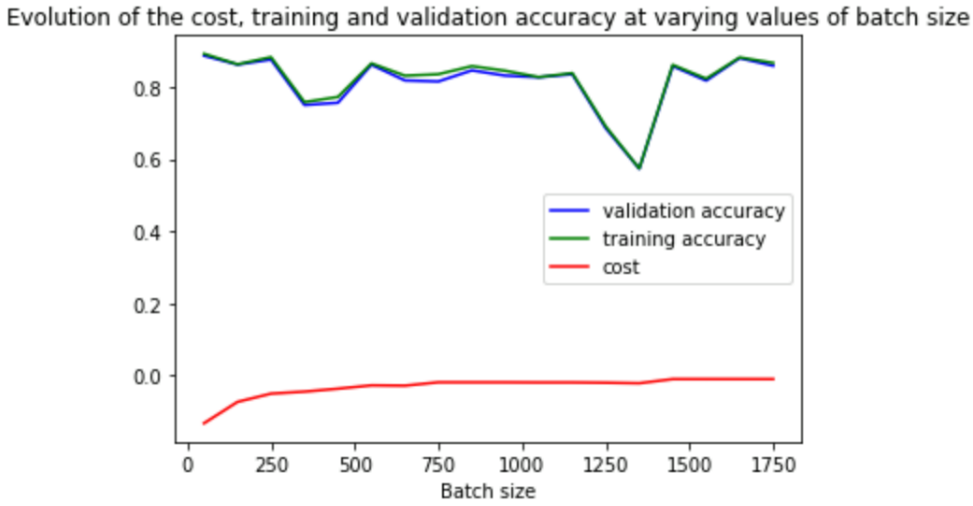
Further Investigation would involve exploring the reason for the low effect of our L1 & L2 Regularization on the accuracy or performance of our model. Furthermore, we would implement Adaptive Gradient methods such as Root-Mean Squared Propagation or Adaptive Momentum Estimation.

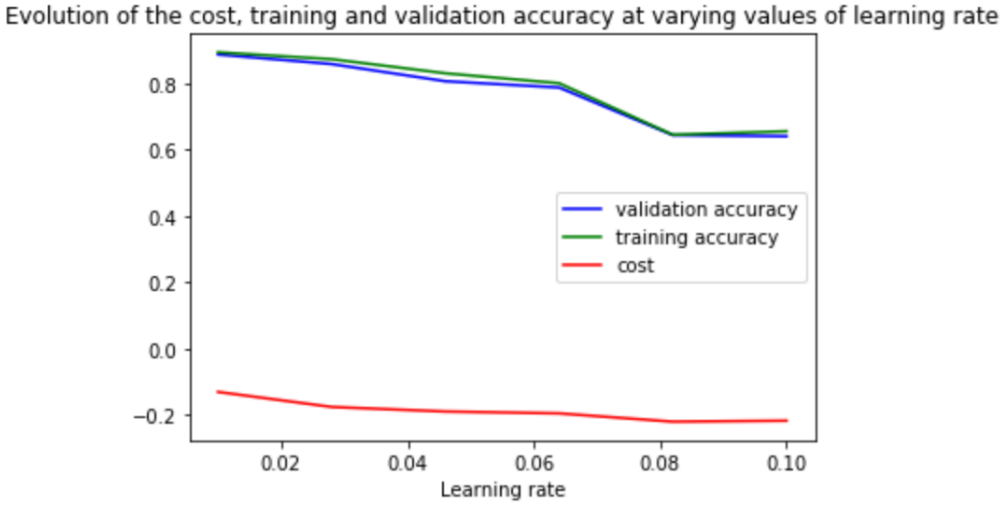
6 Statement of Contributions

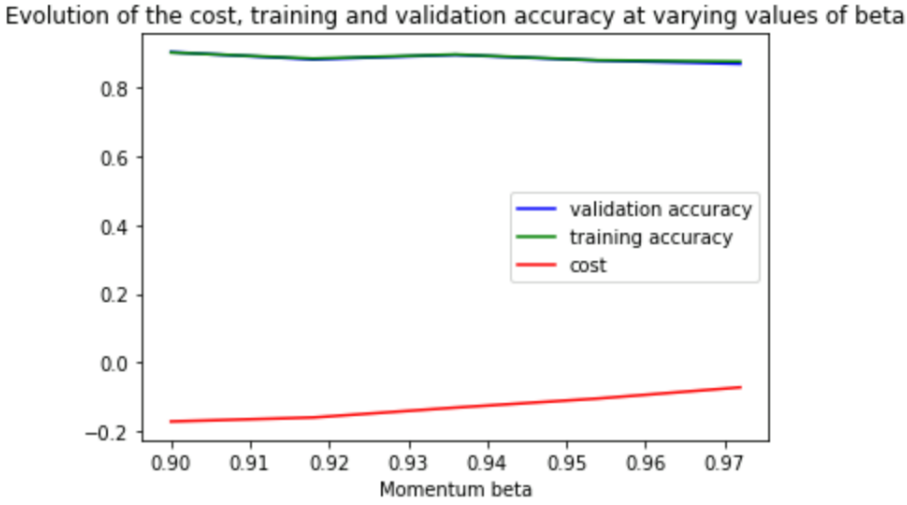
The work done on this project was split as follows. Ameer and Doreen worked on Implementing Softmax Regression equally. Doreen and Ella worked on Hyper-parameters optimization. Ameer worked on Termination condition. Doreen worked on Comparison against another classifier. The report was written by Ameer.

7 Appendix

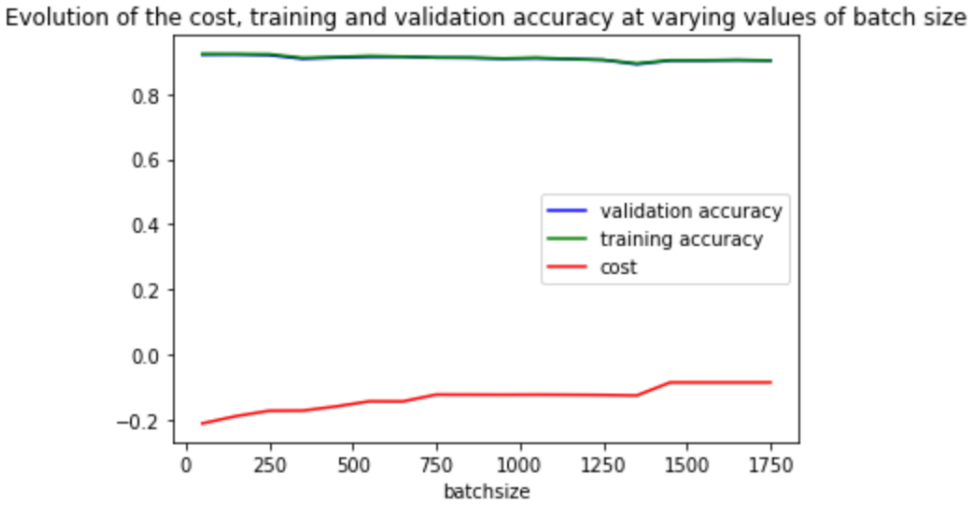
7.1 Hyper-Parameter Evolution for Dataset 1

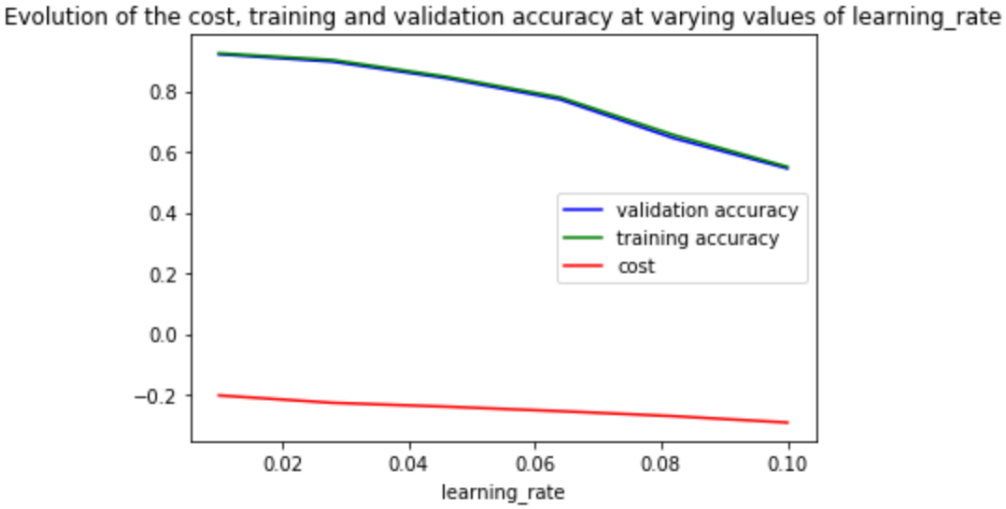


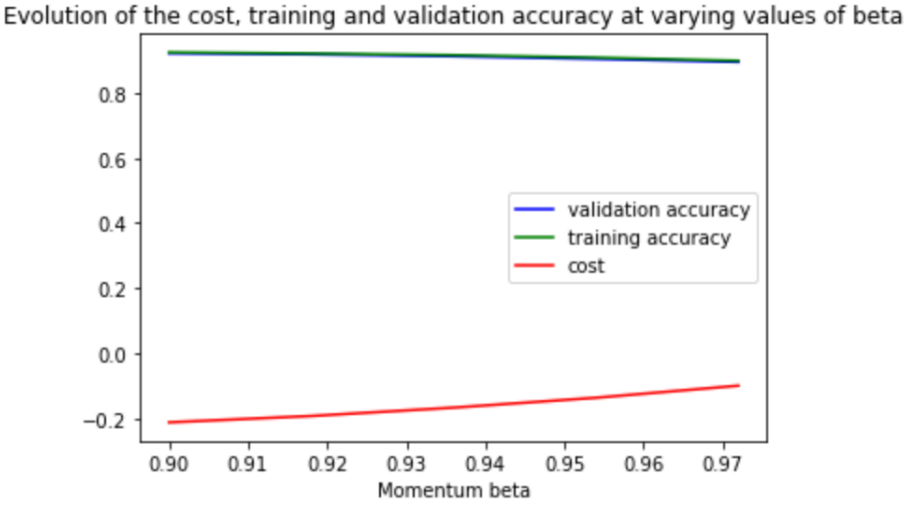




7.2 Hyper-Parameter Evolution for Dataset 2







8 References

- Cardiotocography Dataset: https://www.openml.org/d/1466