Multi-Class Logistic Regression and Gradient Descent

COMP 551, Fall 2020, Mcgill University

November 22, 2020

Team members:

Narry Zendehrooh Kermani 260700556

Abdul Rahman Alseiari 260899182

Abdelhadi Ghalmi 260587573

Abstract

Multiclass regression or softmax regression is a classification method that generalizes logistic regression to problems with more than two possible discrete outcomes. This model can be used to predict the probabilities of the different possible outcomes of a categorically distributed dependent variable, given a set of independent variables. In order to gain insight about softmax regression and its gradient descent optimization, both will be implemented from scratch. We found that the batch size and learning rate affect the performance and run time of the multiclass logistic regression. When compared against the Decision tree, multiclass logistic regression has better validation accuracy but worse training accuracy and run-time performance.

Introduction

The datasets used to compare the performance are the Digit Dataset which consists of hand-written digits and Credit-G Dataset which predicts a person's credit as good or bad. The primary tasks consist of processing the data, implementing a softmax regression model, hyper-parameter search, comparing the performance against Decision Trees and exploring the advantages of random search. The softmax regression model accepts an optimizer class based on gradient descent to find the weight parameters needed for the linear boundary. For this project we use stochastic gradient descent with momentum. The training termination condition is to stop when validation accuracy has not decreased in the T previous iterations. An overall maximum iteration must still be added for safety to prevent infinite looping.

Datasets

The Digit dataset is made up of 1797 8x8 images that each image is of a hand-written digit which is flattened into a 1797x64 array. Each feature is an integer in range 0-100, and the label is in range 0-9. Credit-G dataset classifies 1000 people described by a set of 21 attributes as good or bad (1 or 0) credit risk. Credit-G has some nominal attributes that we encode using one-hot encoding. We standardize features using the StandardScalar function in Sklearn.

Results

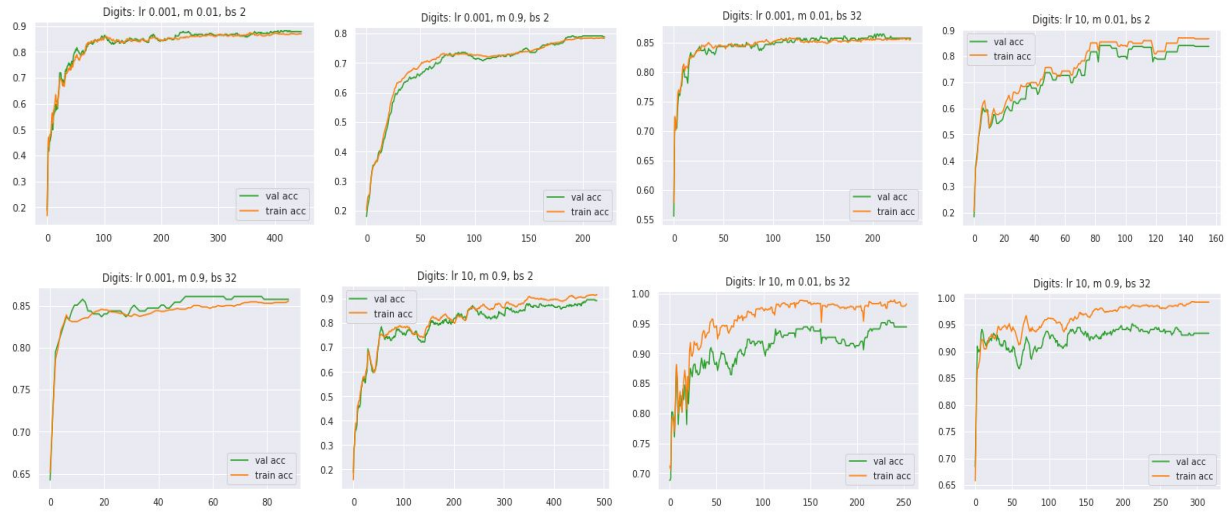
Best hyper-parameters found

Our model on Credit-G seems to be underfitted, which could be due to logistic regression being too simple of a model (not enough model capacity) for this particular dataset.

Dataset Name	lr	batch size	momentum	val acc	train acc	avg time (s)
Digits	10.0	32	0.5	0.959	0.997	0.53
Credit-G	10.0	2	0.5	0.813	0.793	5.86

Figure 1: Plots of training and validation curves for several representative choices of hyper-parameters

Digits dataset



Credit dataset

The graphs in *Figure 1* show different validation and train accuracies in hyperparameter grid search tuning. The plots show that learning rate and batch size are important hyperparameters. We also demonstrate the training time for each set of hyperparameters depicted in Figure 1. Setting too small of a learning rate can lead to slow convergence, and setting too large of a learning rate can lead to divergence. A larger momentum value helps alleviate the problem of getting stuck on local minima and noisy gradients, and thus leads to faster convergence. Also, a larger batch size helps the model converge faster by estimating more accurate gradients.

Training times (per fold) on Digits dataset (s)

batch size	lr=0.001, m=0.01	lr=0.001, m=0.9	lr=10, m=0.01	lr=10, m=0.9
2	0.60	0.28	0.21	0.59
32	0.32	**0.10**	0.35	0.42

Training times (per fold) on Credit dataset times (s)				
batch size	lr=0.001, m=0.01	lr=0.001, m=0.9	lr=10, m=0.01	lr=10, m=0.9
2	0.31	0.18	1.17	3.00
32	0.15	**0.04**	6.31	6.29

Table 2: Performance of Softmax Grid Search vs Softmax Random Search vs Decision Tree validation, train and runtime

Digits dataset

Model	Validation Accuracy	Training Accuracy	Average Time
Softmax (Grid)	0.95895	0.99739	0.53059
Softmax (Random)	**0.97008**	**0.99948**	0.50563
Decision Tree	0.84134	0.98817	**0.01981**

Credit dataset

Model	Validation Accuracy	Training Accuracy	Average Time
Softmax (Grid)	0.81250	0.79250	5.86259
Softmax (Random)	**0.81500**	0.81781	6.37463
Decision Tree	0.68499	**0.93312**	**0.00723**

Table 2 shows how Softmax performs against Decision Tree in a 5 fold cross validation. In both datasets, Softmax using Random Search has the accuracy. However, the run-time performance of Decision Tree is better than Softmax.

Bonus and Creativity

We experiment with random search for finding the hyperparameters of our model as opposed to grid search. We use a reciprocal (log uniform) distribution for our learning rate (with a max value of 10), which has logarithmic spacing between samples. For sampling momentum, we use a uniform distribution in the range [0, 1]. For sampling minibatch size, we use a discrete uniform distribution in

range [1, 128]. We use the same number of search iterations as our grid search (3*3*3=27) for fair comparison. As can be seen from the below table, random search performs better for both Digits and Credit-G dataset compared to grid search.

Dataset Name	lr	batch size	momentum	val acc	train acc	avg time (s)
Digits	1.72	66	0.41	0.97	0.9994	0.51
Credit-G	0.76	66	0.78	0.815	0.818	6.37

Discussion and Conclusion

The takeaway of this project is that hyperparameters can play a big role in how long a model takes to converge, and whether it converges at all or not. This hyperparameter search can be a long and compute intensive process, and if using grid search will take exponentially longer the more hyperparameters that we have.
For improvements, we implemented random search as opposed to grid search. Random search theoretically should perform better whenever one hyperparameter is more important than another (which is almost always the case). Moreover, empirically we found a better model using random search.
Potential improvements which we did not find the time to implement and could have led to improved results:

- Using fancier optimization techniques, such as **nesterov momentum** or **Adam optimizer**
- Using better initialization techniques for initial weights, such as **He initialization**
- Using a model with more capacity, such as a **MLP (Multi-Layer Perceptron)** could help, as we witnessed underfitting on the Credit-G dataset

Statement of Contributions

Narry Zendehrooh Kermani: Datasets, Multi-class logistic regression, bonus
Abdul Rahman Alseiari: Grid search, 5-fold cross-validation, Analysis
Abdelhadi Ghalmi: Gradient descent, Termination conditions, Analysis