UIO

FYS-STK4155
Project 2

November 9th, 2020

Kjell R. Christensen

ABSTRACT

Today's data analysts are faced with a never-ending compromise between gathering data, to the extent that it "clearly" concludes on trends, but on the flip side, what sort of relevance do your conclusion have, if this always drags out in time an becomes yesterday's news.

As a compromise this project will look into the option of limiting the processing of the full data set by using Stochastic Gradient Descent (SGD), on a selected set of the full data.

The project will show that SGD will give a good indicator for a solution that might be sufficient enough, for the stakeholder.

INTRODUCTION

In a fast-developing society where humans leave an escalating amount of digital traces every single day, data analysts are faced with a huge challenge in the amount of data to process.

For a continuous motivation and search for improvement, large amount of data is analyzed every day, to spot correlations, patterns, but also outliers for “your” every day´s expected behavior.

Today’s data analysts are faced with a never-ending compromise between gathering data, to the extent that it “clearly” concludes on trends, but the flip side of this, what sort of relevance do your conclusion have, if this always drags out in time an becomes yesterday’s news.

Even though computer power has shown to follow More’s law, the data for the last couple of decades has out pace this.

We will use the SGD on the logistic regression and also implement as solution for the FFNN and process limited amount of a data set.

METHODS

The analyses is done in a msCode environment and is using standard libraries as specified in the *.py heading. All “imports” are based on standard libraries, commonly used for ML, and routines and code generated for this project, is located only in the python file.

I have use two datasets, one test-set, generated by extracting random numbers within [0,1] and scaling them by subtracting the mean. Second dataset is based on real topographic data. I have been scaling these data in the same manner as for the test-set.

The program is modularized, based on separate tasks, and I have modularized in such a way that the program itself, can run one, several or all modules, in one go for analyses. A separate test environment is, hence, not needed.

The reader can at any time reproduce any test-results or plots, by activating the selected part in the MainModule().

All plots are located in the ./Plots folder and they are named, based on function, parameters and methods. The naming convention is also indicating the size of the sample data, to ease the setup, and re-run the tests.

FORMALISM

For the linear regression we know that the Normal Equations will give a mathematical optimal solution for the design and coefficient matrix. Tempting as it might be, it should be used as a reference rather than a solution. It will give an absolute optimal solution for the equation, hence overfitting our solution for the test data.

At suited solution would be to introduce the OLS and Ridge methods to find a “close” to optimal solution. Analysis with cross-validation will show a crossing of the two curves for variance and bias. The intersection between the curves indicate at stopping point for the training and optimal betha to apply to the test data.

The latter two methods can be computational tough and request a lot of data power, when we have a lot of instances and many features. The curse-of dimensionality is well known, and we often need to apply feature reduction for the model, to make the model computational.

Huge number of instances will still put constraint on computation power, so to come around this problem with introduce the stochastic gradient descent. Due to the fact that we only compute the gradient of one instance at a time, the resources needed is relatively small, but the downside is that the gradient and direction towards global min is bad and the solution will never reach the global min, but then again could be more than satisfying for the stake holder.

For the FFNN we used the breast cancer data in sklearn and with the 30 features it was computational on a normal laptop. This was a binary classification problem and with ReLU activation function for the hidden layers and Sigmoid for the output and classification. The implementation run into division by zero for increased number of epochs, but overall gave a good performance.

For the logistic regression the activation was set to cross-entropy, that based on its mathematical formula, punish and correct the errors (distance for Y) even harder that the Sigmoid. The range for the y-values for both are [0,1] and can easily be interpreted as TRUE/FALSE for a probabilistic solution.

RESULTS & DISCUSSIONS

The Stochastic Gradient Descent (SDG), was implemented using standard functionality and we would have liked to have momentum as part of our solution.

For the FFNN, our first implementation was with just one hidden layer and did not manage to predict, so no conversion. Our test result for MNIST gave an average of 10% for the 10 classes and with the prediction levels set at 0,5 (50%), no predictions were found for the data set. A clear conclusion, that we need a minimum of two hidden-layers for a multi classification problem.

The expansion to a flexible multi-layer solution got stuck based on the timeframe and we used a fallback option where we tested the breast cancer data set. The solution was limited in its implementation and did not handle high numbers of epochs, due to variables running minus infinity.

With a 3-hidden layer solution we managed a prediction of 86%.
The finding was surprising when we reduced “neurons” from 32-16 in the hidden layers, and gained a 2% better prediction? We are currently not certain why this occurred but will be pursued further in the next project.

For the breast cancer data our implementation was good, but we will implement SGD for the FFNN, when we approach larger datasets.

CONCLUSIONS & EXPECTATIONS

Based on the algorithm and the theory we can easily see the convenient of using SGD. The methods avoid the high dimensionality and hence gives a good approximation. The solution included is basic and we will pursue adding the momentum to the functionality for the next phase.

For the logistic regression classification, we found that the sigmoid activation function was surprisingly good, but we could like to implement the cross-entropy in our next implementation.

The FFNN was hard to implement with multiple hidden layers and we needed to use a fall back solution. We were not able to implement a stable solution for large number of epochs, that most likely would have given us even a better prediction on the beast cancer data. We run int div/zero problem and we need to pursue this further in the next project. Also, for this implementation we used Sigmoid for the prediction in the out layer and our findings favor the cross-entropy also for the neural network.

REFERENCES

[1] Morten Hjorth-Jensen, Computational Physics, Lecture Notes and sample code, Fall 2020, https://github.com/CompPhysics/MachineLearning

[2] Trevor Hastie, Robert Tibshirani, Jerome Friedman, The Elements of Statistical Learning, Second Edition.
DOI https://**doi**.org/**10.1007**/978-0-387-84858-7; Copyright Information Springer-Verlag New York 2009; Publisher Name Springer, New York, NY; eBook ...

[3Hands-on Machine Learning with Scikit-Learn, Keras, and TensorFlow, Aurelien Geron. Hands-on Machine Learning with Scikit-Learn, Keras, and TensorFlow [2nd ed.] 978-1-492-03264-9. 174 47 66MB. English Pages 1150 Year 2019.

[4]Data Science from Scratch-First Principles with Python, Grus.
Afficher les collections Cacher les collections. Identifiant. ISBN : **978-1-491-90142-7**. ISBN : **978-1-491-90142-7**. ISBN : 1-491-90142-X. ISBN : 1-491-90142-X.

[5] Practical Statistics for Data Scientists - Peter Bruce & Andrew Bruce
978-1-491-95296-2 [M] www.allitebooks.com Dedication We would like to dedicate this book to the memories of our parents Victor G. Bruce and Nancy C. Bruce, ...

[6]Python Machine Learning, Sebastian Raschka & Vahid Mirjalili
23. sep. 2017 — Python Machine Learning - von Sebastian Raschka, Vahid Mirjalili (ISBN **978-1-78712-593-3**) bestellen. Schnelle Lieferung, auch auf ...

[7]A Primer on Scientific Programming with Python, Hans Petter Langtangen
ISBN **978-3-662-49886-6**; Free shipping for individuals worldwide. Please be advised Covid-19 shipping restrictions apply. Please review prior to ordering.

[8]A Neural Networks And Deep Larning, Michael Nielsen / Dec 2019
http://neuralnetworksanddeeplearning.com/index.html