Edi Begovic edbe@itu.dk 17-02-2020

Introduction

In this small project I will be implementing logistic regression and an MLP classifier using the ML-libraries *Scikit-Learn* and *Keras* respectively. Both models are relatively simple, but are trained in an iterative manner which leverages the efficient implementations provided by the above mentioned libraries.

Considerations

As the data represents values of pixel-intensity (0-255) for a 28x28 grid, I start out by normalising the values in a flattened representation (784, 1). Other minor technicalities such as having the labels be *one-hot* encoded to use with Keras had to be done.

The concrete implementation of the models were mostly based on the 'getting started' examples provided by the documentation of the libraries. The MLP model was initialised with two hidden layers each consisting of 521 neurons and using the *ReLU* activation function for the output. Due to the simplicity of the models and from past experience, changing the hyper parameters had very little effect on the performance of the models.

Evaluation

Due to the scope of this exercise and the models chosen, I haven't introduced a development set, although that *should be a given* in any machine learning project. While the logistic regression model doesn't have many *hyper parameters* to tweak, the MLP does, including the number of neurons and layers, activation functions etc. Adjusting these parameters and solely relying on the evaluation metrics on the test-set as a measure of how *good* ones model is easily isn't sensible, as one is just adjusting the model to fit the test data.

As both the training and test data are uniformly distributed across the five classes, I've chosen to rely on (micro)accuracy as the main metric in combination with inspecting the confusion matrix. The latter helps with quickly getting an overview of which classes were the most *confused* by the models.

Model	Accuracy
Multi-Layer Perceptron	0.90
Logistic Regression	0.87