



After setting the seed (to ensure a stable outcome) and preparing in the data, we also normalized the data. We used the mean and standard deviation from the labeled training set for all sets (also for unlabeled training and test set).

In a first step, we then trained a neural network on the labeled training set. We included 2 layers with relu activation functions and applied softmax on the output layer. We used a sparse categorical crossentropy loss function, adam as optimizer (better than SGD for large data sets) and accuracy as valuation metric. The model was fitted throughout 200 epochs using batches of 50 observations each. The model was evaluated based on an 80%/20% training/validation split. To allow for different layer sizes, we created a list with several models consisting of layer sizes between 150 and 350 neurons.

In a second step, we let python sort of perform an expectation maximization process. We use the previously trained model to predict the labels of the unlabeled training set. If the model can claim a label with more than 99.5% confidence, the observation is added to the labelled dataset. This combined dataset is then used to newly train the model. This is done in a total of 25 loops.

After each iteration, the current model (trained on the combined set) then makes predictions for the test set.