TBMI26 – Computer Assignment Report  
Supervised Learning

Deadline – March 14 2021

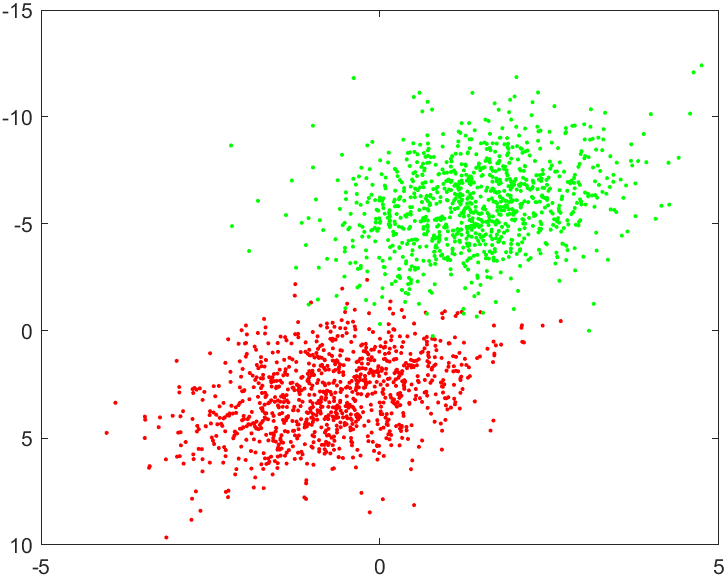
Author/-s:

Martynas Lukosevicius

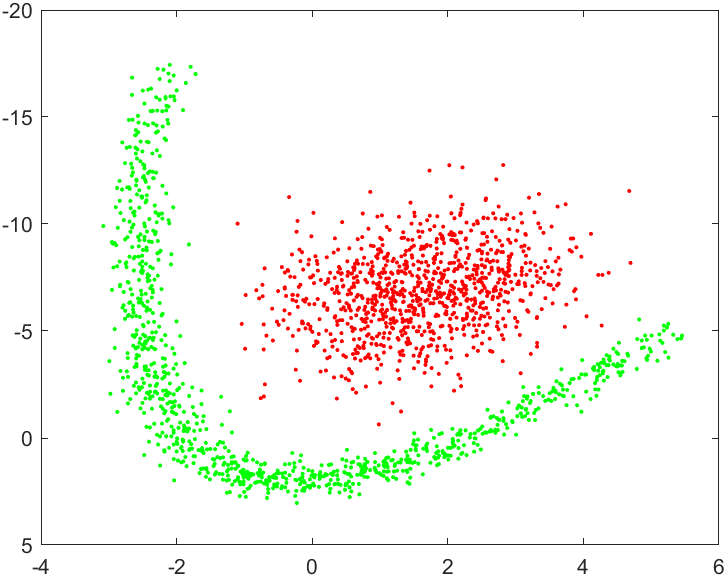
In order to pass the assignment you will need to answer the following questions and upload the document to LISAM. Please upload the document in PDF format. **You will also need to upload all code in .m-file format**. We will correct the reports continuously so feel free to send them as soon as possible. If you meet the deadline you will have the lab part of the course reported in LADOK together with the exam. If not, you’ll get the lab part reported during the re-exam period.

1. **Give an overview of the four datasets from a machine learning perspective. Consider if you need linear or non-linear classifiers etc.**

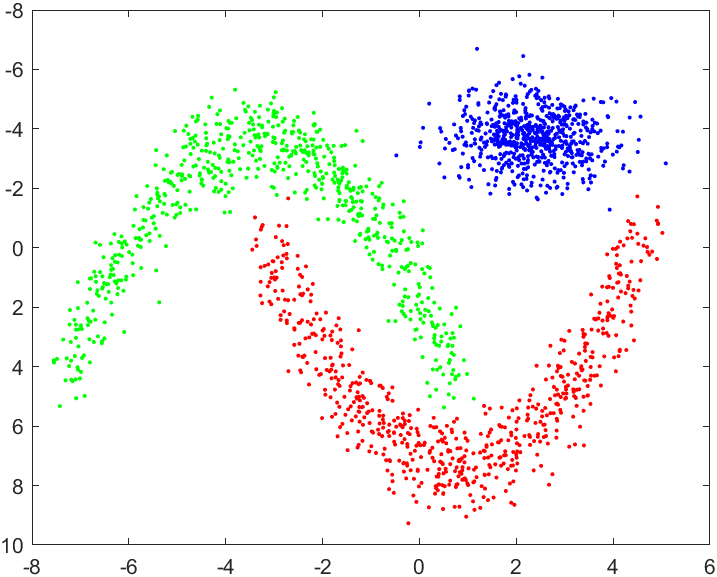
Dataset 1: we can see 2 separated classes, which can be classified by linear model (draw a line between). However, some data points overlay. As a result, there will be misclassification error.



Dataset 2: 2 classes which can be classified by non-linear classifier, data points don’t overlay, so its possible to classify everything correctly



Dataset 3: 3 classes, can be classified using nonlinear classifier, some data points are close to other class groups, so they might be misclassified.



1. **Explain why the down sampling of the OCR data (done as pre-processing) result in a more robust feature representation. See** [**http://archive.ics.uci.edu/ml/datasets/Optical+Recognition+of+Handwritten+Digits**](http://archive.ics.uci.edu/ml/datasets/Optical+Recognition+of+Handwritten+Digits)
2. **Give a short summary of how you implemented the kNN algorithm.**

Find k closest neighbors by Euclidean distance, use majority voting to classify

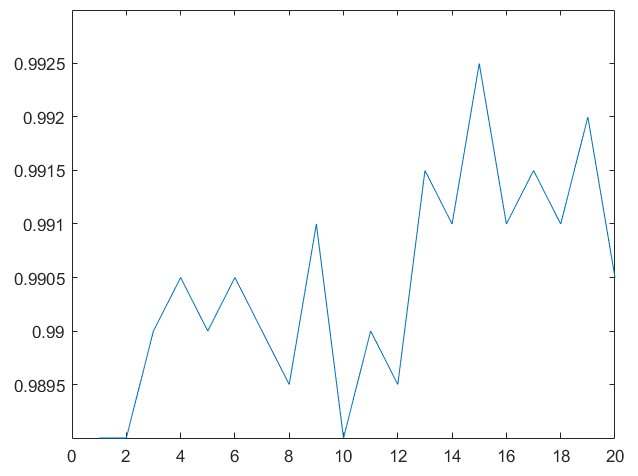
1. **Explain how you handle draws in kNN, e.g. with two classes (k = 2)?**

If there is draw, calculate sum of distances to every class, choose label which sum is min

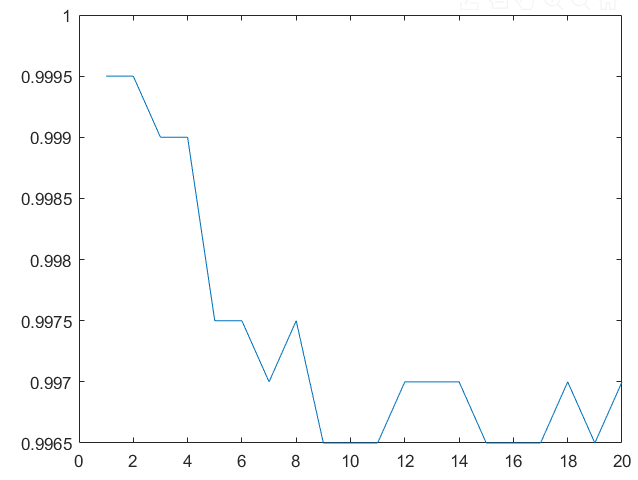
1. **Explain how you selected the best k for each dataset using cross validation. Include the accuracy and images of your results for each dataset.**

Did 10 fold cross validation. Divide data into 10 parts and train on 9 parts , test one 1 for every model, 10 times with different test fold

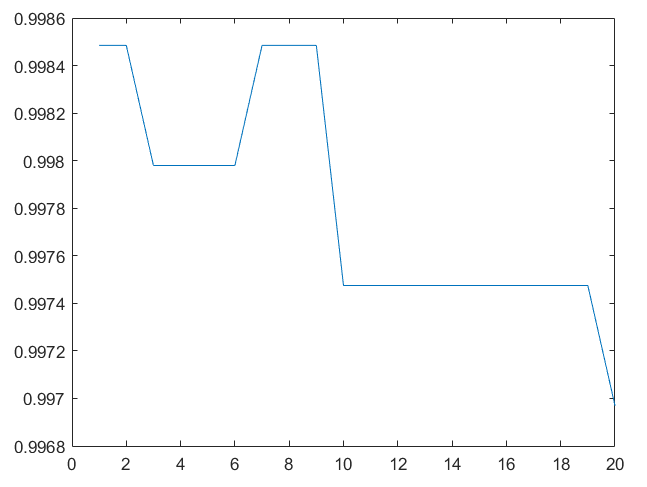
Dataset1: k = 15, accuracy = 0.9925,



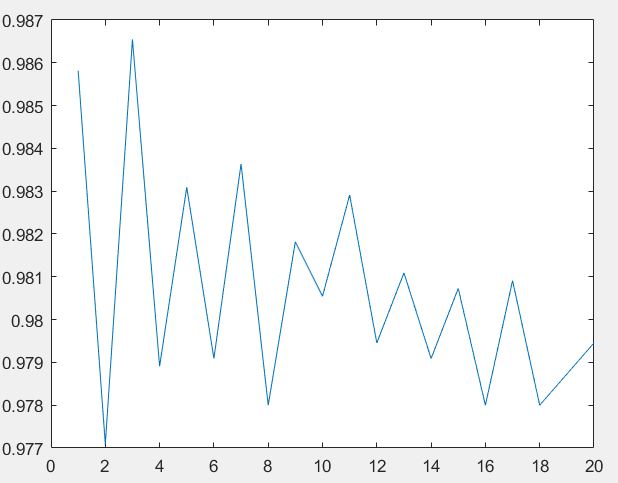
Dataset2: k = 1, accuracy = 0.9995,



Dataset3: k = 1, accuracy = 0.9985,

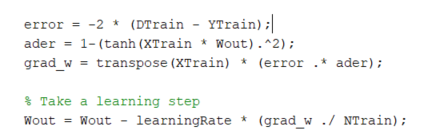


Dataset4: k = 3, accuracy = 0.9865,



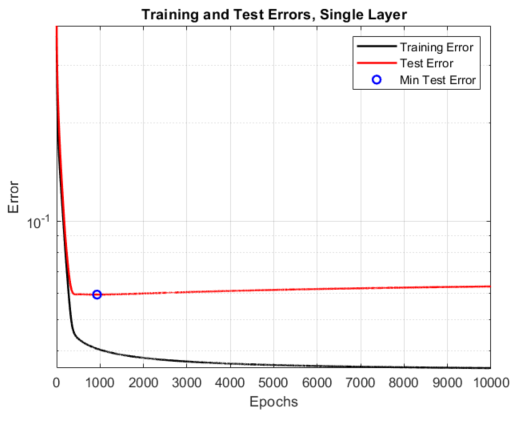
1. **Give a short summary of your backprop implementations (single + multi). You do not need to derive the update rules.**

For backpropagation we will use gradient descent. To derive gradients for weight matrixes, we used chain rule.



1. **Present the results from the neural network training and how you reached the accuracy criteria for each dataset. Motivate your choice of network for each dataset. Explain how you selected good values for the learning rate, iterations and number of hidden neurons. Include images of your best result for each dataset, including parameters etc.**

For dataset1, as its linearly separatable we use single layer NN.

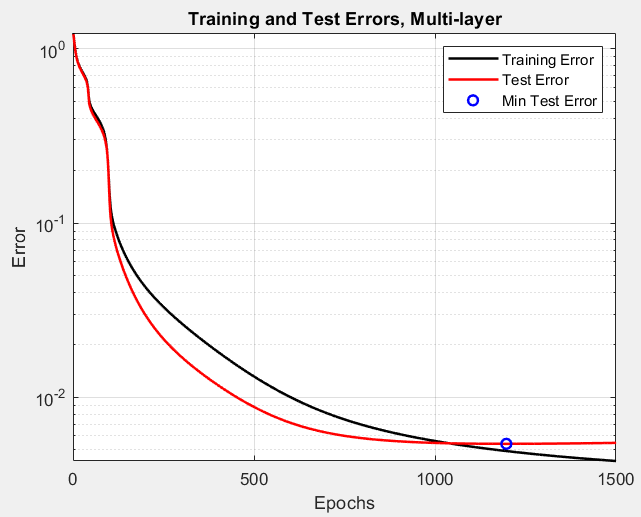


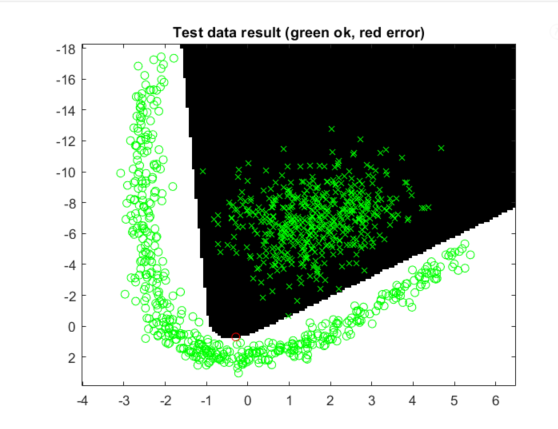
From the plot we can see that 1000 epochs should be chosen for training, as later it overfits.

Test accuracy: 0.987

Dataset2, is not linearly separatable. We are choosing multilayer NN.

We chose to have 2 hidden neurons, learning rate chose to be 0.05, because learning doesn’t oscillate and is stable. By having 1197 epochs test error reaches smallest value.





Test accuracy: 0.999

Data set 3:

numHidden = 7;

numIterations = 10000;

learningRate = 0.1;

****

****

Test accuracy: 0.999

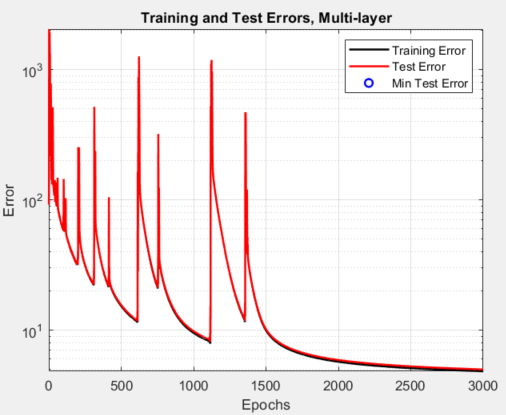
Dataset 4: increased hidden layer until reached Test accuracy: 0.96354

numHidden = 200;

numIterations = 3000;

learningRate = 0.2;

Test accuracy: 0.96354



1. **Present the results, including images, of your example of a non-generalizable backprop solution. Explain why this example is non-generalizable.**

# of train data = 9

# of test data =

numHidden = 7;

numIterations = 1000;

learningRate = 0.1;

****

****

Test accuracy: 0.84534

Because of insufficient amount of training data, model was not able to “catch” real patterns, as a result it can’t predict well on training data.

1. **Give a final discussion and conclusion where you explain the differences between the performances of the different classifiers. Pros and cons etc.**
2. **Do you think there is something that can improve the results? Pre-processing, algorithm-wise etc.**

We could do regularization by dropout method, try to get faster convergence by introducing adaptive learning rate, to add more layers.

1. **Optional task (but very recommended). Simple gradient decent like what you have implemented can work well, but in most cases we can improve the weight update by using more sophisticated algorithms. Some of the most common optimization algorithms are summarized nicely here:**

[**https://towardsdatascience.com/optimizers-for-training-neural-network-59450d71caf6**](https://towardsdatascience.com/optimizers-for-training-neural-network-59450d71caf6)

**Implement one or a few different optimizers and compare the speed at which the training converges. A good starting point is to implement momentum gradient decent.**