HELLENIC MEDITERRANEAN UNIVERSITY
DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING

Artificial Neural Networks

Final Project

24. May 2022

Adam Fabo (er1785)

Contents

1 Introduction

Goal of the project is to create and train neural network on given data. Plan is to load the data and visualise them, remove the outliers if there are any and then create single neural network that will learn how to classify this data. Next and main step is to look at different neural networks with varying numbers of layers an number of neurons in each layer and see the results of the experiment. Part of the experiment is to also try differently split dataset (training data and test data) and observe behaviour of neural network. Last step is to find optimal number of clusters for self organizing map using Elbow method and Silhouette analysis.

2 Dataset

Assignment specified 4 datasets from which one can be chosen. I chose the one with the most samples because I think it gives me biggest freedom when manipulating the data and results in best trained neural network. Data in the dataset that I chose were extracted from images that were taken for the evaluation of an authentication procedure for banknotes. Dataset has 5 attributes which are:

- 1. variance of Wavelet Transformed image (continuous)
- 2. skewness of Wavelet Transformed image (continuous)
- 3. curtosis of Wavelet Transformed image (continuous)
- 4. entropy of image (continuous)
- 5. class (integer)

There are only 2 classes to which can specimen belong, and it is described in table by either 0 or 1. Dataset has 1372 rows and first 5 are presented in Table 1.

Row	Variance	Skewness	Curtosis	Entropy	Class
0	3.621600	8.666100	-2.807300	-0.446990	0
1	4.545900	8.167400	-2.458600	-1.462100	0
2	3.866000	-2.638300	1.924200	0.106450	0
3	3.456600	9.522800	-4.011200	-3.594400	0
4	0.329240	-4.455200	4.571800	-0.988800	0

Table 1: First 5 rows of dataset

But better to show whole dataset than only to describe it. Images on Figures 1 - 5 show plotted attributes one by one by their value and their sample number. Images also show their histograms. Samples from dataset are grouped together by Class - meaning that first rows are samples that belong to first class and then follow samples that belong to second class. Since there are only 2 classes to which sample can belong, differences between classes can be seen on a first sight, mainly at Variacne, Skewness and Curtosis. This can not be said about Entropy.

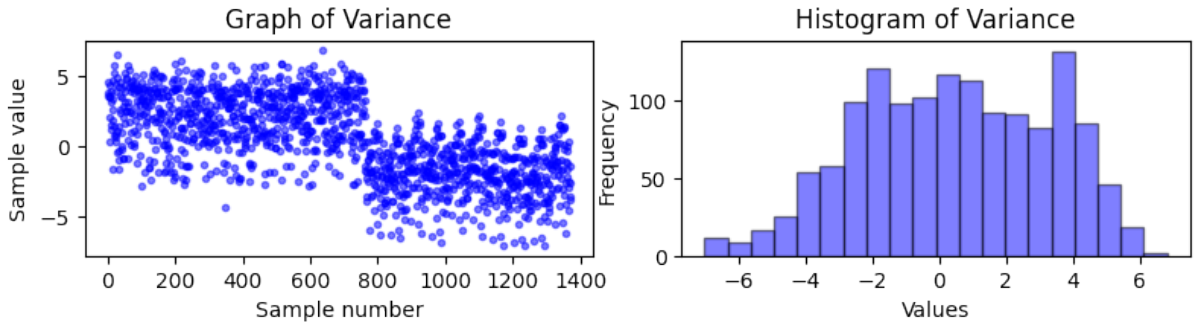

Figure 1: Variance

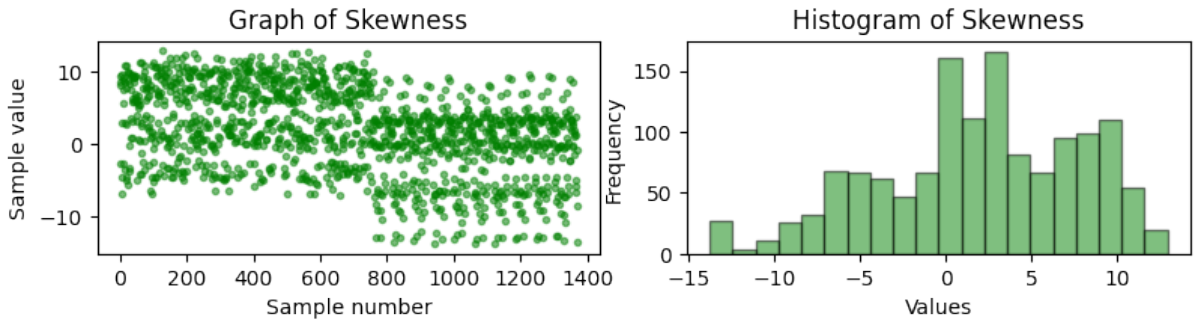

Figure 2: Skewness

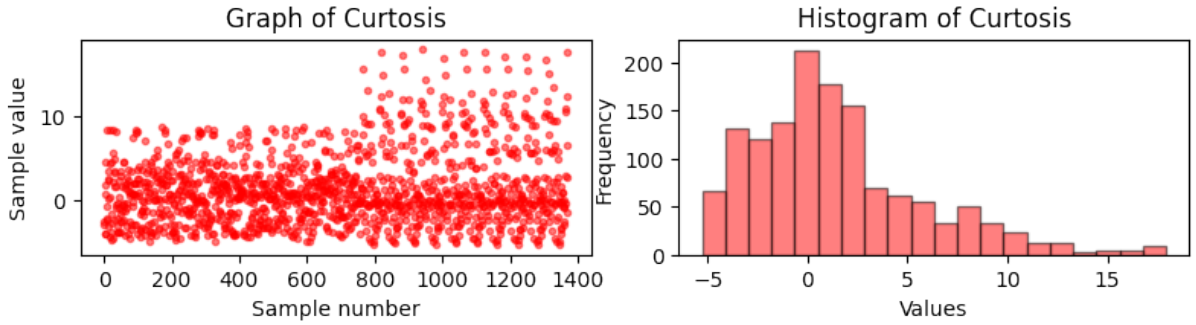

Figure 3: Curtosis

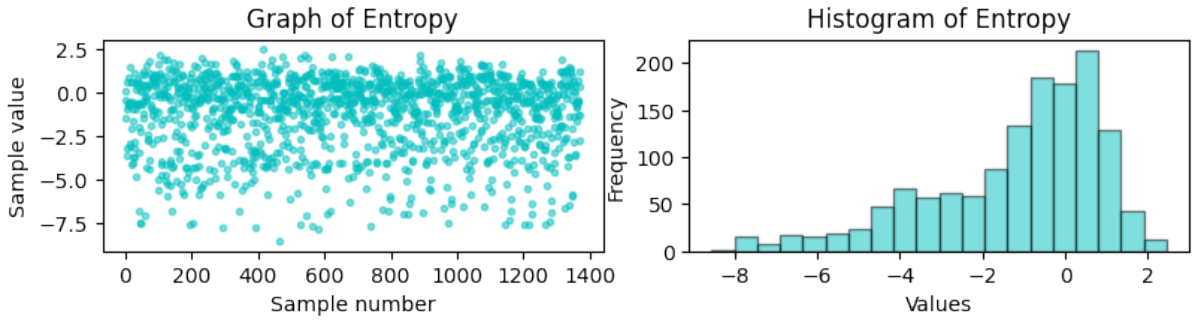

Figure 4: Entropy

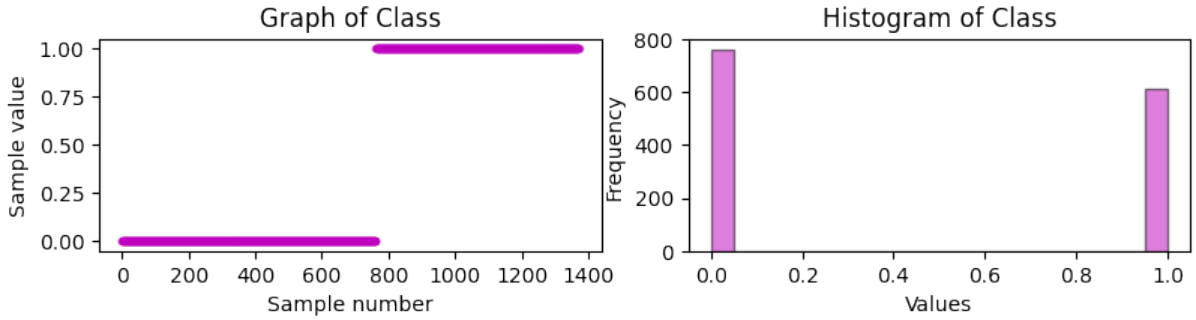

Figure 5: Class

2.1 Outliers

Outliers are data points that lie an abnormal distance from other values. These data points can mess with following data processing, so that is why they should be dealt with.

For detection of outliers I tried to use 2 methods: Interquartile range method and Standard deviation method. Standard deviation method is better suited for datasets without extreme outliers and chosen dataset does not have them. After few tests I decided to go with Standard deviation method over Interquartile range method (second method detected too many values as outliers). Standard deviation method or Z-score gives you an idea of how far from the mean of dataset, that follows normal distribution, a data point is .

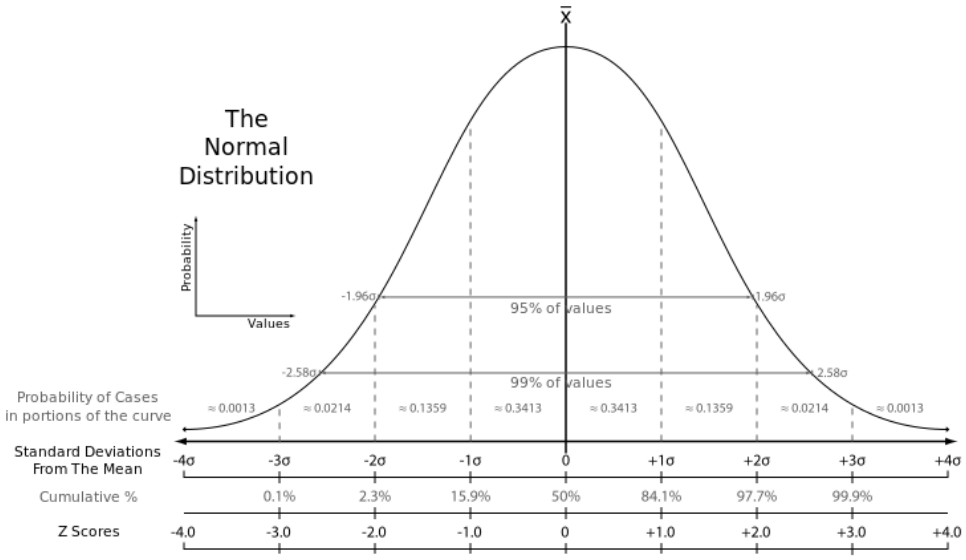

Figure 6: Normal distribution and Z-score

After Z-score of sample is calculated, its absolute value compared to chosen number (here I am using 3). If absolute value of Z-score is bigger than this value, the sample is considered as an outlier and is dealt with. Equation 1 shows formula for Z-score.

$$Z = \frac{x - \mu}{\sigma} \tag{1}$$

By this method there were found 36 outliers from 1372 samples (2.62%). Since that is not that many outliers I decided to remove them and continue with 1336 samples. Images 6-11 show differences of histograms before and after removing outliers. What can be seen from these histograms is that mainly the edge values were removed.

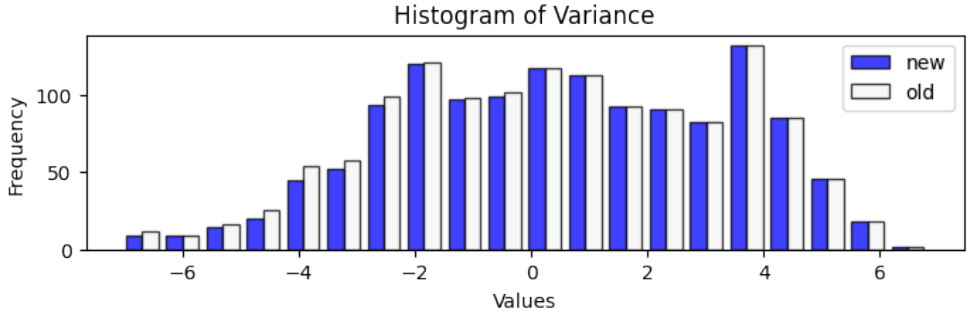

Figure 7: Histogram of Variance

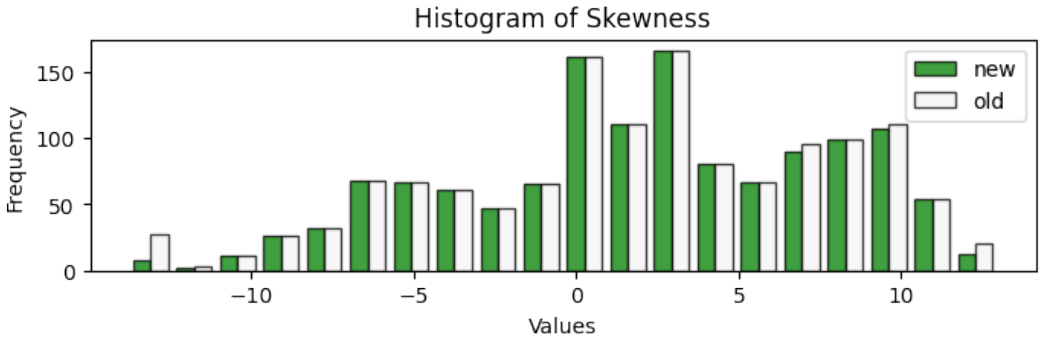

Figure 8: Histogram of Skewness

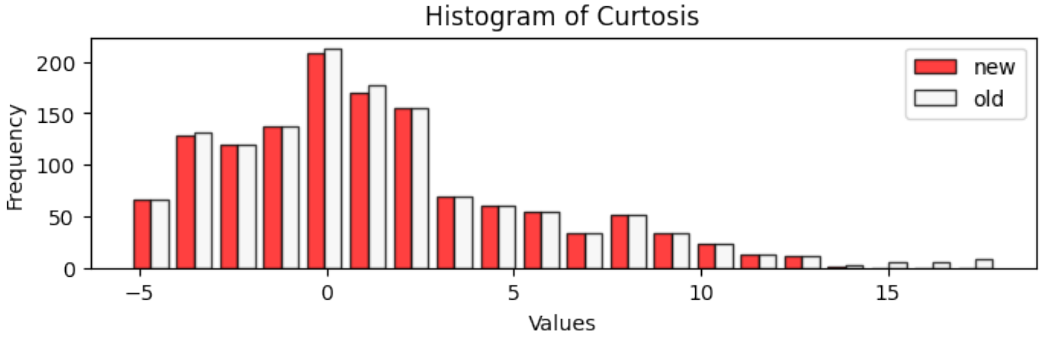

Figure 9: Histogram of Curtosis

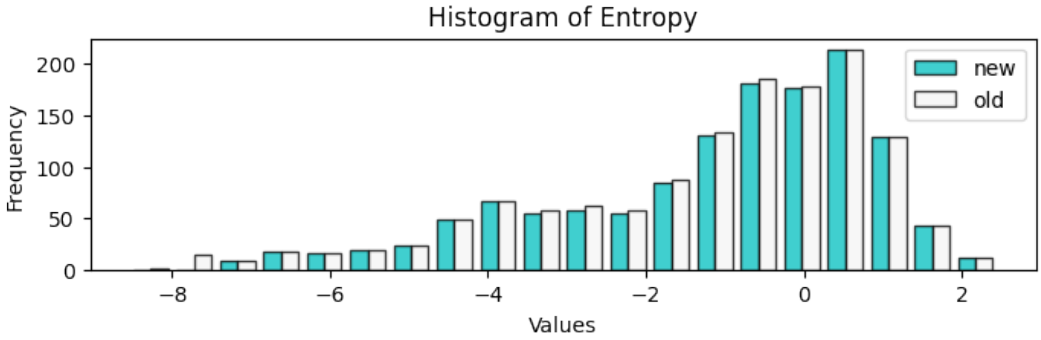

Figure 10: Histogram of Entropy

Figure 11: Histogram of Class

2.2 3D visualisation

From Figures 1-3 can be seen distinct differences between 2 classes of the dataset. That gave me an idea to plot them in 3D as function of Variacne, Skewness and Curtosis.

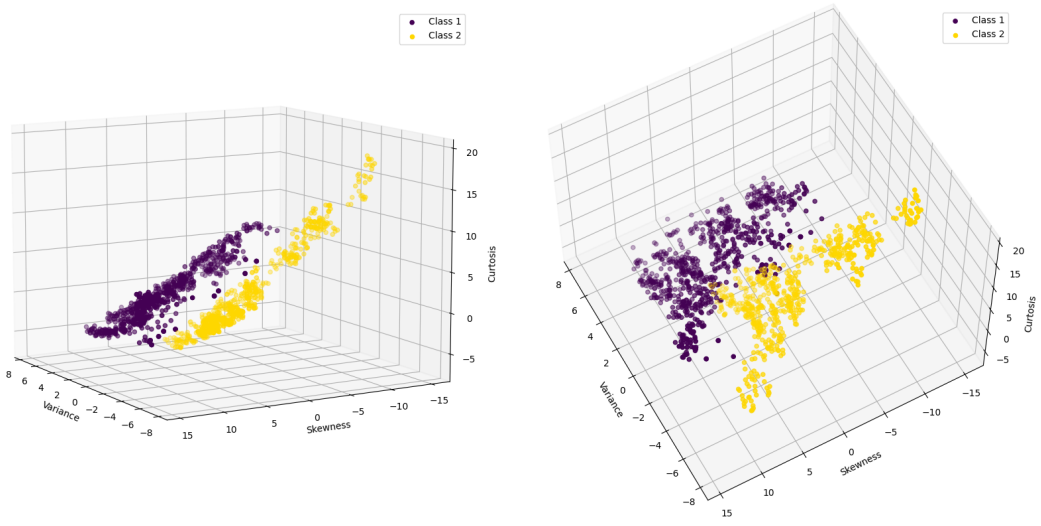

Figure 12: 3D visualisation of data

As a result of 3D scatter what we can see is that data are in mainly two distinguishable clusters. At first glance it might even look like they are linearly separable. That is one of the tests in later chapters.

3 First Neural Network

As per assignment I split the data randomly to 2 groups - train and test set which were in ratio 70% : 30%. Then I created neural network with one hidden layer and 3 neurons in hidden layer. Output layer is set to have 2 neurons (even if 1 could be sufficient) where only one neuron lights up at a time, indicating to which class does data point belong.

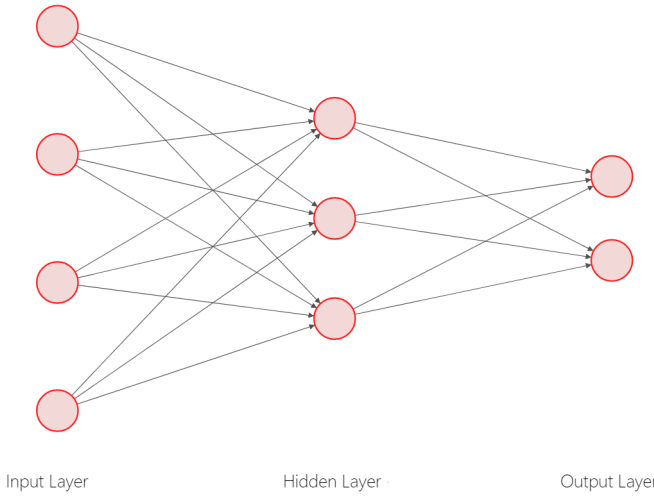

Figure 13: Visualisation of constructed neural network

Other parameters that were given in assignment are:

- Learning rate: 0.3
- Maximum number of epochs to train: 3000
- Performance goal: 1e-5

3.1 Results of trained neural net

Neural network did not reach performance goal before maximum number of epochs was reached. Error goes as follows:

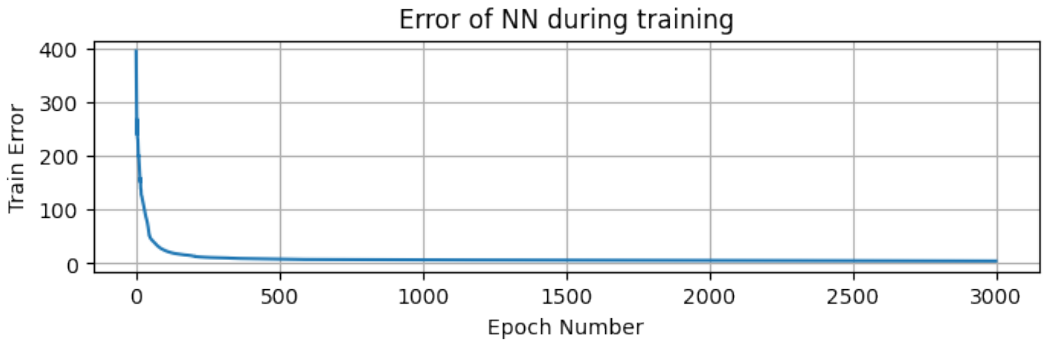

Figure 14: Training error of neural network

Error after last epoch was 4.649. After training neural network was presented with training data and with test data which neural network did not see until this moment. Results of accuracy are presented in Table 2.

	Training set	Test set
Accuracy	99.38%	99.27%

Table 2: Result of trained neural network

From results can be seen that training of the neural network could use some more hundreds epochs in order to reach performance goal. Test set was of size 412 and recognized patterns are scattered at Figure 15

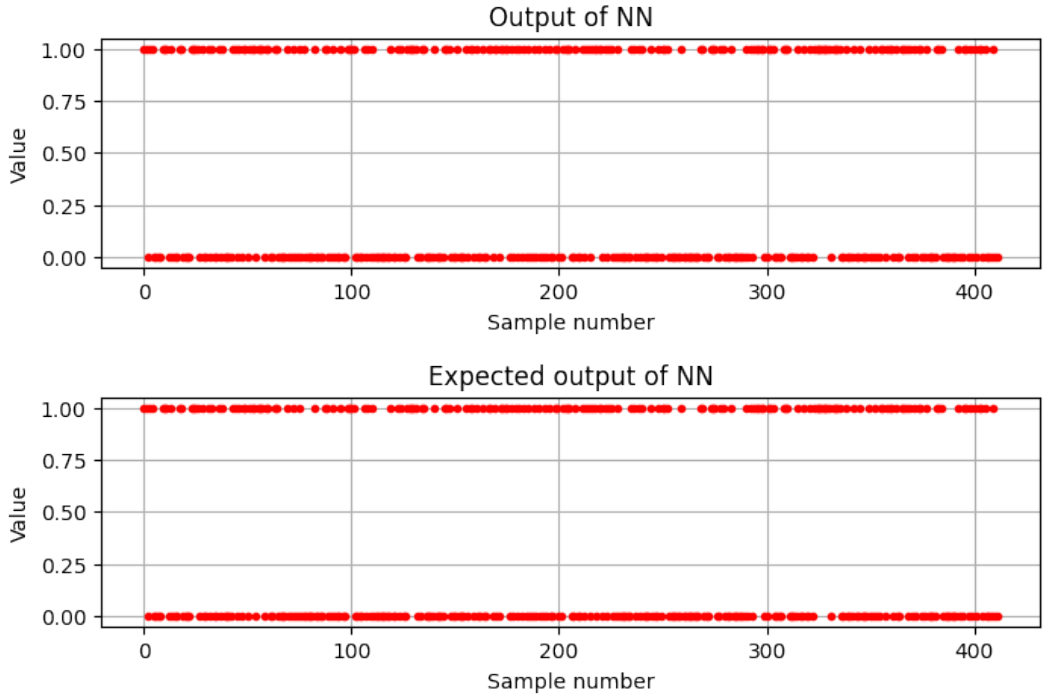

Figure 15: Real and expected output of neural network

Since there are 412 points at each graph and it is hard to see the differences, on Figure 16 are only shown wrongly recognized patters.

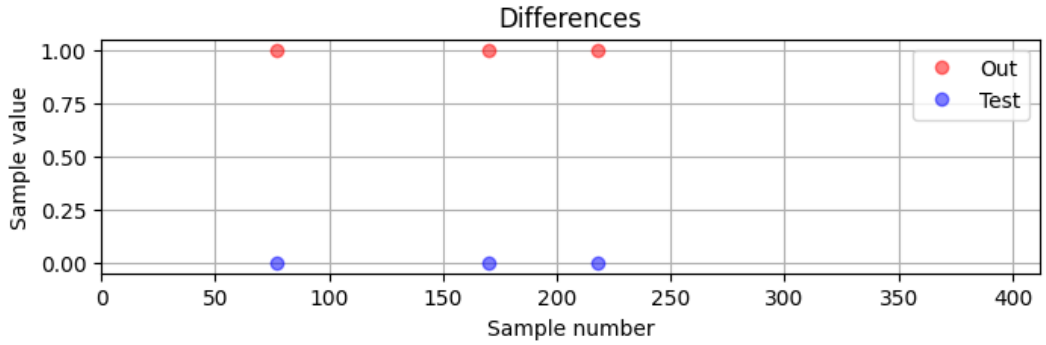

Figure 16: Differences between real and expected output

Overall neural network had high accuracy and it could be considered as successfully trained.

4 Test with perceptron

Goal of the test with perceptron is to find out if the data in dataset is linearly separable. Figure 12 might suggest that it could be possible. Second goal is to look at entropy and its role in classification of this dataset, since on Figure 4 can not be seen distinguishable difference in different classes of the dataset as can be seen in variance (Figure 1), skewness (Figure 2), and curotsis (Figure 3).

4.1 Perceptron model

Perceptron is type of neural network that has only single layer - output layer. It is binary classifier, meaning that it can split the dataset into two categories. In 2 dimensions it is by line in 3 dimensions it is by plane and so on. On Figure 17 is model of perceptron that is used in this test.

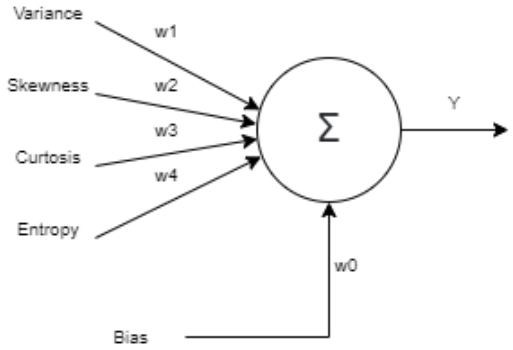

Figure 17: Perceptron

Output of any neural network are trained weights. In perceptron these weights are normal vector of the line/plane that is dividing the dataset. After perceptron is trained, weights can be extracted and and we can plot the line/plane that weights are describing.

4.2 First test

Goal of the first test is to see if perceptron can classify input data, and if not, look at the accuracy.

Firstly perceptron was created and trained on data without outliers. Parameters with which was perceptron trained are written in Table 3.

Epochs	Learning rate	Training set	Test set
500	0.05	70%	30%

Table 3: Parameters of perceptron

Result is that perceptron never reached 0 error as Figure 18 shows. That means the data is not linearly separable. But error of trained perceptron was small, which indicates that perceptron could have high accuracy when trying to classify data.

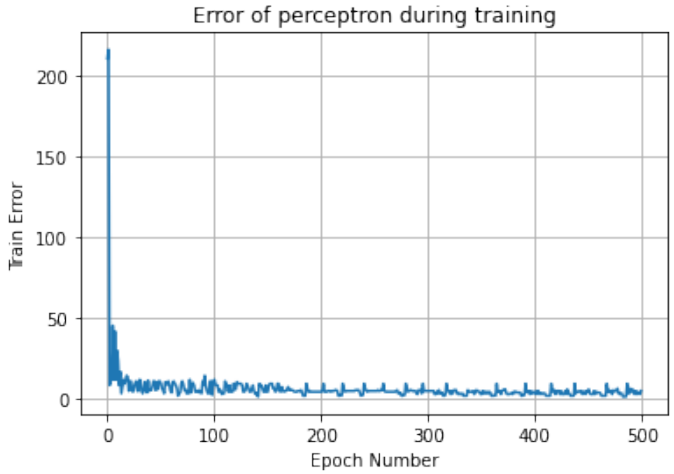

Figure 18: Perceptron error

Lets take a look at weights of the trained perceptron to relation to its input data.

Input Data class	Weight	Value
Variance	w1	-1.97
Skewness	w2	-1.84
Curtosis	w3	-2.00
Entropy	w4	0.05
Bias	w0	2.50

Table 4: Weights of trained perceptron

What can be seen from weights is that weight that is connected to Entropy has small value - meaning that entropy does not play a big role when trying to classify this dataset. Lets see how this trained perceptron handles on test data.

	Training set	Test set
Accuracy	99.06%	97.57%

Table 5: Result of trained perceptron

From Table 5 can be seen that perceptron has high accuracy at this dataset

4.3 Second test

As can be seen from Table 4, entropy does not play big role when classifying this dataset. That begs the question - can this dataset be classified even without the entropy? Same as before perceptron was created with same parameters as are mentioned in Table 3.

Result of training is similar. Perceptron can not divide the dataset into 2 categories since error never reached 0 as can be seen at Figure 19.

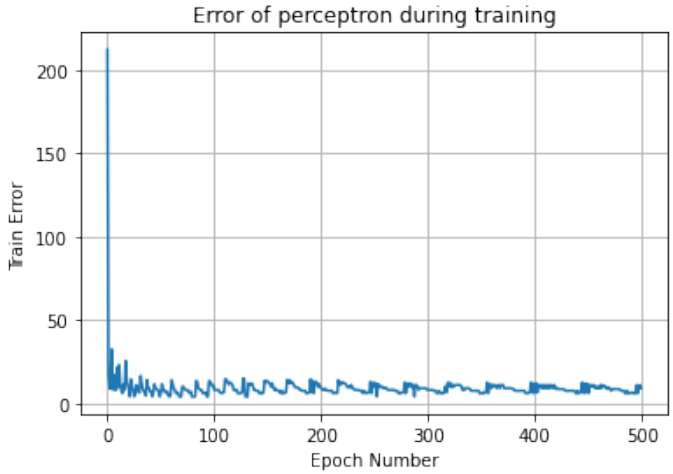

Figure 19: Perceptron error

The results of accuracy are similar if not the same as in the the test with entropy because accuracy highly depends on how was the dataset splitted. Accuracy of trained perceptron can be seen in Table 6.

	Training set	Test set
Accuracy	98.12%	96.84%

Table 6: Result of trained perceptron

What is important are the values of weights shown in Table 7. Values of weights are not that different from previous test, but now there is one less value.

Input Data class	Weight	Value
Variance	w1	-1.26
Skewness	w2	-1.43
Curtosis	w3	-1.49
Bias	w0	1.9

Table 7: Weights of trained perceptron

One less value means that there are now only 4 values which in mathematical view describe plane that tries to divide the dataset. These values can be seen as a normal vector which, is perpendicular to a plane. And this plane can be visualised in 3 dimensional graph, like one that is shown on Figure 12. This can be seen on Figure 20.

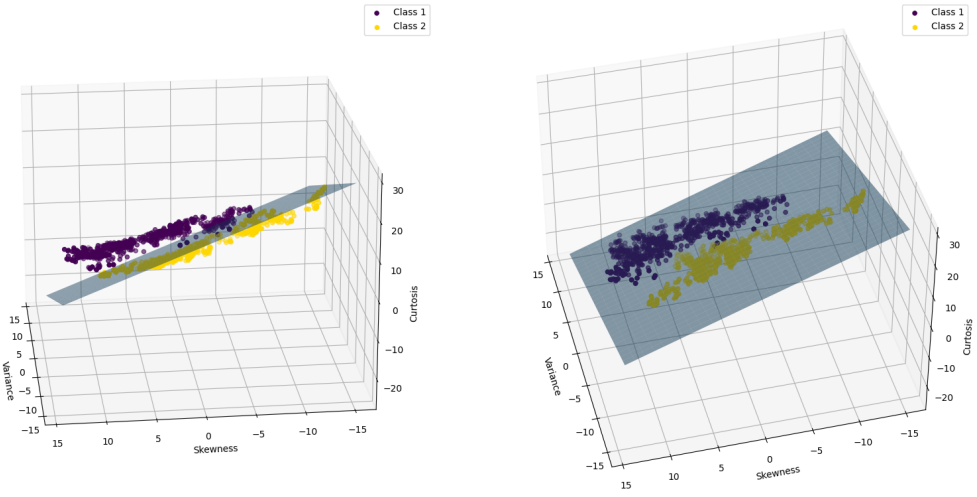

Figure 20: 3D visualisation of trained percepton

4.4 Conclusion

Perceptron proved itself as adequate tool for classification of this dataset. Of course it all depends on the application where this perceptron would be used if the provided accuracy would be sufficient. This test also showed that entropy provides small to none information when trying to classify this dataset.

Second test proved that dataset can be classified even without the entropy. It has also visualised the weights of trained perceptron, as mathematical model suggest - a plane in its relation to the dataset.

5 Test with one hidden layer

Last chapter has shown test of neural network as single layer perceptron. This chapter expands the neural network from only output layer to neural network with one hidden layer and one output layer. Neural network constructed like this should be better at recognizing patters than simple perceptron. While perceptron can separate dataset only linearly - by line, plane etc., neural network with one hidden player can separate dataset with more complex convex shapes.

5.1 Description

Goal of this test is to take look at behaviour of multiple neural networks with different number of neurons in hidden layer and different number of epochs for which is the neural network trained. Data that are observed are: *error* during training, *time* of training and *accuracy* on training and test dataset.

Parameters that do not change during the test are shown in Table 8.

Learning Rate	Learning goal	Training set	Test set	Transfer function	Training function
0.05	1e-5	70%	30%	LogSig	Resilient Backpropagation

Table 8: Static variables of test

Parameters that are iterated trough are defined in Table 9.

Min Neurons	Max Neurons	Neuron Step	Min Epochs	Max Epochs	Epoch step	Repeats
1	10	1	100	3000	100	10

Table 9: Definition of variables

Before each test weights are assigned randomly and dataset is split to training and test set randomly as well. To avoid statistical errors, each test will be repeated 10 times, as parameter Repeats shows in Table 9. Pseudo code of algorithm used in this test is described in Algorithm 1.

Algorithm 1 TEST ALGORITHM

```
for n in < 1, 10 >, step = 1 do
    for e in < 100, 3000 >, step = 100 do
        for r in < 1, 10 >, step = 1 do
            Split the dataset
            Create neural network with n neurons in hidden layer
            Train neural network for e epochs
            Collect data from training
        end for
        Average the results and save them
    end for
end for
```

5.2 Error with different training

First thing that needs to be shown is that training error behaves similarly when training different neural networks with same number of neurons, for different number of epochs. This knowledge reduces complexity of test in later sections, which would be more complicated not knowing this.

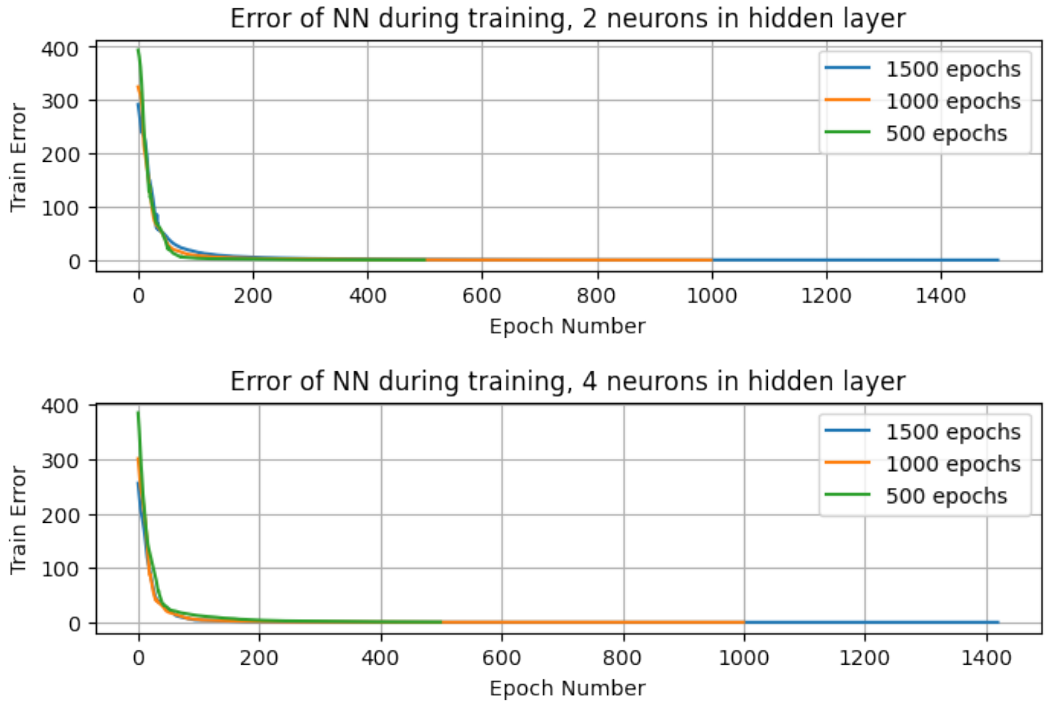

Figure 21: Error with varying number of epochs

Figure 21 shows 2 different plots, in each are plotted errors of 3 different neural networks. Only difference is these networks is number of epochs for which they were trained. They were trained for 500, 1000, 1500. What can be seen is that errors behave similarly no matter the number of epochs. This test was the only one that was not repeated multiple times, so there can be seen differences that come out of statistical noise, that was random weights number and randomly split test and training set. Note: even tough these 2 graph are similar, it is not point to show that error behaves same with different number of neurons in hidden layer. Point is to show that error behaves similarly when there is same number of neurons in hidden layer.

5.3 Results of the test

One of criteria of successfully trained neural network is that training was completed, meaning that learning goal was reached. Results of training multiple neural networks as is described in Algorithm 1 can be seen on Figure 22.

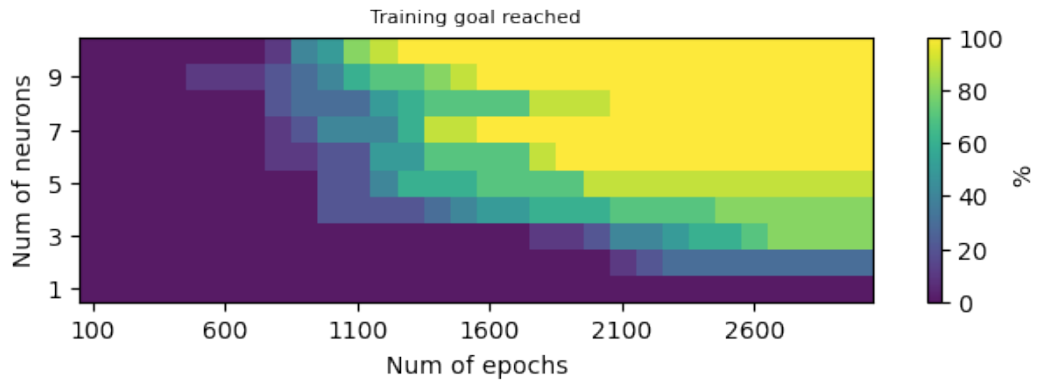

Figure 22: Training goal reached

X-axis shows for how long was neural network trained and Y-axis shows how many neurons were in hidden layer. This format of showing data stays consistent as can be seen other figures. Each test with neural network was conducted 10 times with random values, so naturally incidents can occur, where from 2 neural networks with same parameters one reached learning goal and second one does not. That is why in Figure 22. can be seen % of reached training goals. That % means how many neural networks from 10 reached goal before reaching maximum number of epochs. What can be seen from this figure is that neural network with 1 neuron in hidden layer is not able to reach learning goal no matter of number of epochs and neural network with 2 neurons barely can. What can be said about other neural network with higher number of neurons in hidden layer is that, they are able to reach learning goal but they need around 1500-2000 epochs in order to be trained.

Seeing if neural network reached goal is fine, but in case that training goal was not reached it does not tell with what error was the end of training. Figure 23 shows this.

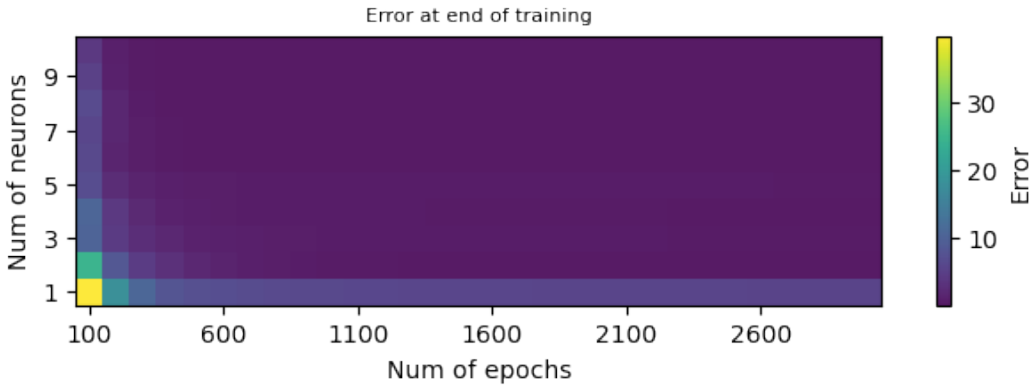

Figure 23: Error at the end of training

Intuitively Figure 23. shows that biggest error is at small number of neurons and at small number of epochs. Considering the nature of this graph, it might be to better visualise it in 3D. This plot can be seen on Figure 24.

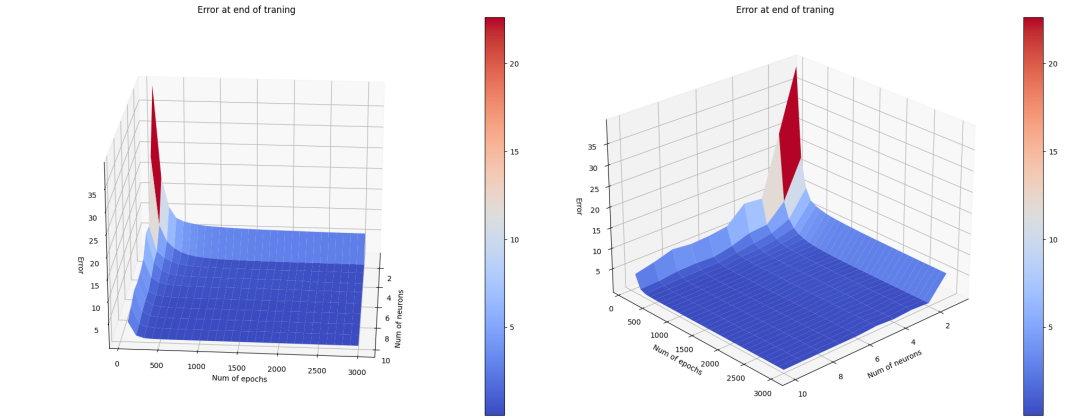

Figure 24: 3D visualisation of errors at end of the training

Figure 24 shows that for majority of neural networks errors goes down rapidly after few hundredths training epochs but reaches learning goal of 1e-5 only after few thousands epochs as is shown on Figure 22.

Another attribute of training that can be observed is how long it took to train neural network. Figure 22 shows if training goal was reached but does not tell how long it took to train neural network. For example, it

shows neural network with 10 neurons which training was set to 3000 epochs, reach training goal but does not tell how many epochs it took. This is what next Figure 25 shows.

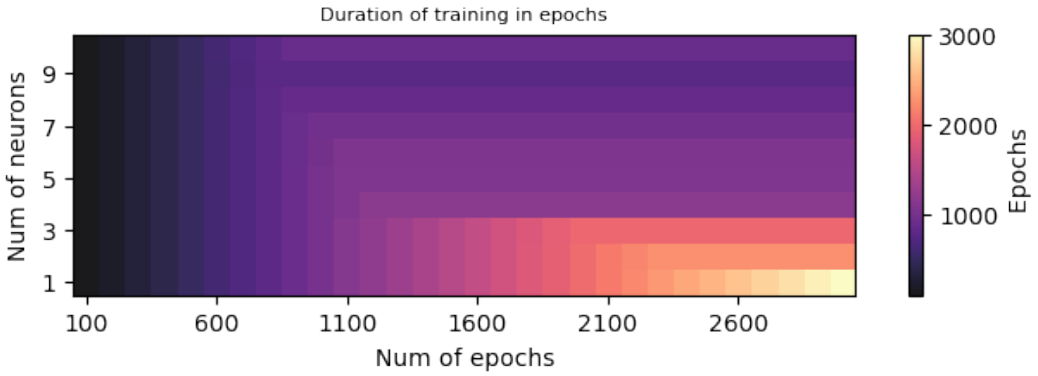

Figure 25: Duration of training in epochs

Figure is constructed in a way, that if learning goal is not reached, duration of training is number of epochs that neural network was trained for. However if goal was reached, number of training epochs is lower than epochs that neural network was set to be trained for. It is the actual number of epochs it took to train the neural network.

What can be observed from Figure 25 is that higher the number of neurons in hidden layer, less time it took to train neural network. Also it shows that same neural network is trained in same number of epochs no matter the number of epochs it was set to be trained for. Additionally plot shows that paradoxically longest time that it took to train neural network was with neural networks that had the least amount of neurons in hidden layer.

But it would be also interesting to see how long did it take to train the neural networks in real time. It could be said that 2000 epochs with single neuron in hidden layer could take the same as ten neurons in hidden layer with 200 epochs (10 times more weights but 10 times less epochs). That is what Figure 26 disproves.

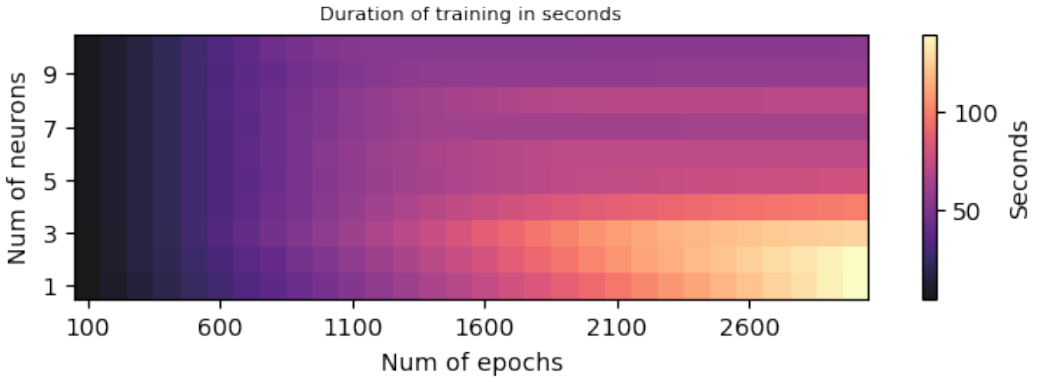

Figure 26: Duration of training in seconds

Figure 26 shows that trend of duration of training is the same in real time - seconds, as it is in epochs. Longest time it took to train the neural networks is with the least number of neurons in hidden layer. Just for info - sum of all values in this graph is over 18000 seconds. It took over 5 hours of real time on Intel Core i5-8300. Of course time on powerful GPU would be by orders of magnitude smaller, but point is to show trend of training time.

Last, and the most important attribute that was observed is accuracy of trained neural network is accuracy. Accuracy was acquired for both - the training and the test dataset. Even if accuracy on training dataset is not

that important, it shows insight on that how well was the neural network trained. Figure 29 shows accuracies of trained neural networks on both datasets.

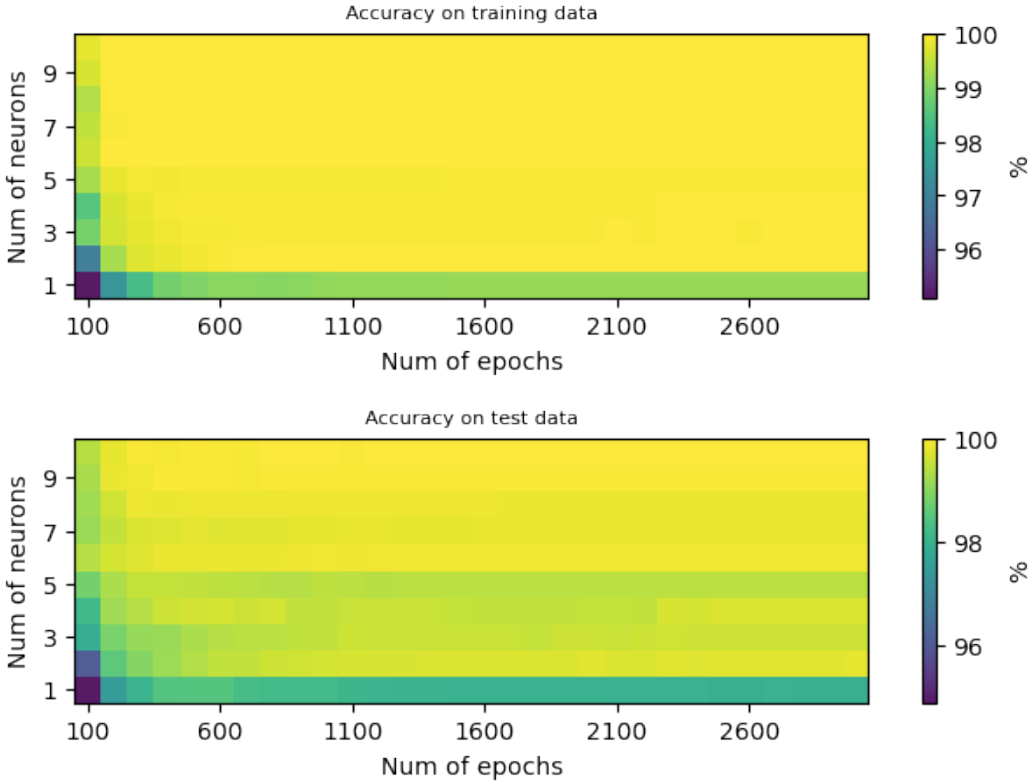

Figure 27: Accuracy on training and test data

As could be expected, accuracy on training data is better than accuracy on test data. Also as expected worst accuracy had the neural networks that were trained for least number of epochs and neural networks with the least number of neurons. But overall every neural network, even the worst performing, had accuracy $> 95\%$, which could be considered as well trained neural network.

What is interesting to see from Figure 27 and Figure 22, which shows training goal reached, is that some neural networks did not reach learning goal but had close to 100% accuracy on both datasets. For example neural network with 10 neurons in hidden layer that was trained for 600 epochs. That might mean that learning goal was set too low and neural networks are being trained unnecessarily too long.

Final information that can be extracted from this data is difference between the two accuracies. Accuracy on test data was subtracted from accuracy on training data and result is shown in Figure 28.

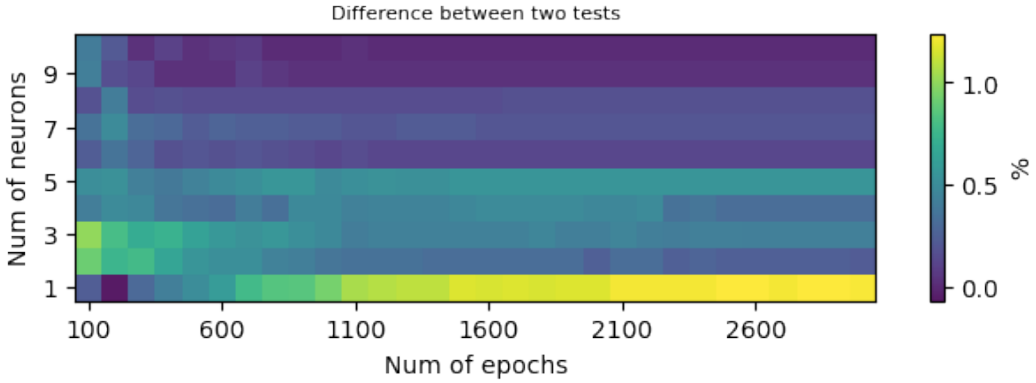

Figure 28: Accuracy on training and test data

What can be seen is that difference between accuracies was never $> 2\%$, and biggest difference was at neural networks with single neuron in hidden layer.

5.4 Conclusion

This tests shown that neural network with one hidden layer is sufficient in recognizing patterns in given dataset. It shows that higher the number of neurons, faster the neural network learns. Also what results might suggest is that learning goal was set too low, judging but the accuracy of trained neural networks, that did not reach learning goal but had 100% accuracy.

6 Test with 2 hidden layers

This test is expansion of the test with one hidden layer as was the test with one hidden layer to perceptron. Now the neural network consists of 2 hidden layers. Neural network with 3 and more layers can distinguish arbitrary complex regions/patters in dataset, so in way this neural network should give best results.

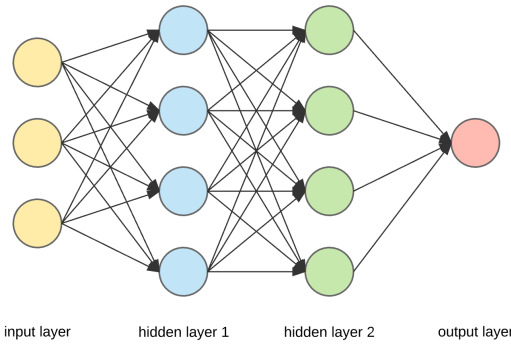

Figure 29: Neural network with 2 hidden layers

Naming convection of hidden layers used in this test can be seen on Figure 29. Layers are numbered from left to right, and in this chapter are called first and second.

6.1 Description

Goal of this test is to take a look at behavior of neural networks with different number of neurons in both hidden layers. As it would be too complex to also iterate trough number of epochs while iterating trough number of neurons in both layers, number of epochs is set to static 3000. Other static parameters of this test are same as they were in previous test, and can be seen at Table 8. Table 10 shows ranges of variables that is iterated trough.

1st layer Min Neurons	1st layer Max Neurons	2nd layer Min Neurons	2nd layer Max Neurons	Repeats
1	10	1	10	10

Table 10: Definition of variables

Algorithm is also similar to the last test, only difference is the mentioned iteration trough neurons in first and second layer, while number of epochs is static.

Algorithm 2 TEST ALGORITHM

for *n1 in* $< 1, 10 >$, *step = 1* **do**
  for *n2 in* $< 1, 10 >$, *step = 1* **do**
    for *r in* $< 1, 10 >$, *step = 1* **do**
      Split the dataset
      Create neural network with $n1$ neurons in 1st and $n2$ neurons in 2nd hidden layer
      Train neural network for 3000 epochs
      Collect data from training
    end for
    Average the results and save them
  end for
end for

6.2 Results of the test

Results from this test are analogy of results from test with one hidden layer. First data that was collected is if network reached training goal, can be seen on Figure 30.

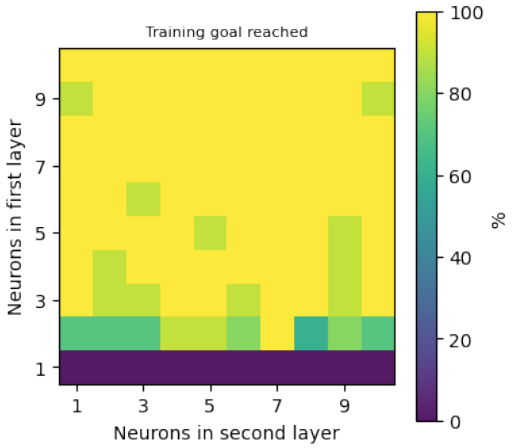

Figure 30: Training goal reached

Figure 30 shows asymmetry between number of neurons in different layers. As can be seen, if neural network is constructed with 1 neuron in first layer, it does not behave the same as neural network with 1 neuron in second layer. That may be because first hidden layer has input of 4 dimension and has output of only one, therefore there is loss of information and network behaves similarly to perceptron. And it does not matter how many neurons there are in second layer, they can not create information out of nothing, they can only work with output of first layer.

Another data point is how long it took for neural network to be trained. Duration of training was measured in epochs and is seconds (Again seconds are here to show the trend, not the absolute value).

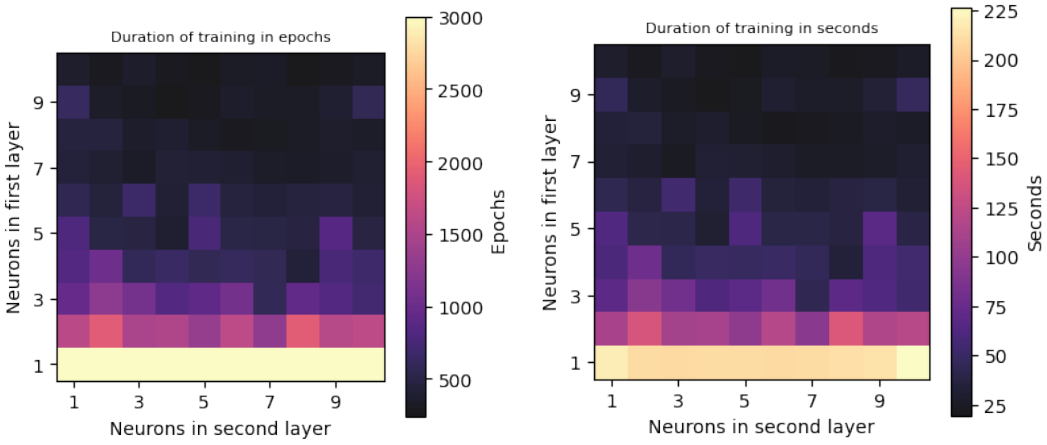

Figure 31: Duration of training

What Figure 31 shows is that counterintuitively, is that it took longer to train networks with lesser number of neurons than networks with higher number of neurons. Also it shows that duration of training is inversely correlated number of neurons in first layer. As for the number of neurons in second layer, it does not seem to have any significant impact for training time.

Last, but most important result of this test is accuracy of the neural network. Accuracy was measured after the neural network was trained. For illustration/comparison neural network was presented with two datasets: one it was trained on and second unknown test dataset.

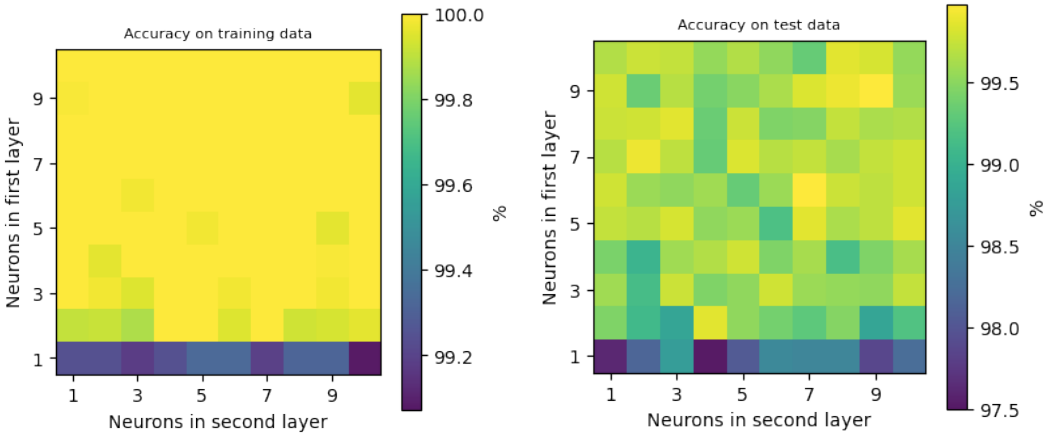

Figure 32: Accuracy

What Figure 32 shows is regards to the accuracy of the neural network with 1 neuron in first layer is that, it had quite good accuracy of 99% on training data and circa 97% on test data. Even though neural network had 3 layers, its accuracy is comparable to the perceptron.

Other feature that can be observed is that neural network had lesser accuracy than expected. When comparing result of this test with result of previous test (Figure 27), neural network with 3 layers shows worse accuracy on test data. That might be because of possible overfitting - when neural learned noise and nuances of training data.

6.3 Conclusion

Neural network with 2 hidden layers has shown little bit worse results than expected. When comparing its accuracy to accuracy of neural network with 1 hidden layer, it showed possible overfitting that was happening. But its accuracy regardless of number of neurons was around 98% which can be considered as successful.

7 Test with differently split dataset

Previous tests were dealing with architecture of the neural network and many parameters stayed the same as can be seen in Table 8. This test takes a look at one of them - ratio between training and test sample, which was until this moment 70:30.

7.1 Description

Goal of this test is to take a look at behaviour of neural network with differently split datasets. Plan is to split the dataset in different rations, train the neural network and look at the outcome. Main data that will be measured is accuracy of the neural network a time it took to train.

Architecture of neural network used in this test is static. Given results from previous tests, in this test is used neural network with single hidden layer with 6 neurons. Static variables can be seen in Table 11.

Learning Rate	Learning goal	Epochs	Transfer function	Training function
0.05	1e-5	3000	LogSig	Resilient Backpropagation

Table 11: Static variables of test

Parameters that are iterated trough are defined in Table 12.

Min % of training set	Max % of training set	% step	Repeats
1	99	1	10

Table 12: Definition of variables

Algorithm for this test is similar to the algorithms of previous tests. But now there is iteration trough differently split dataset and architecture of neural network stays the same.

Algorithm 3 TEST ALGORITHM

for *t in* $< 1, 99 >$*, step = 1* **do**
    Split the dataset, with training size of t% of total dataset
    for *r in* $< 0, 10 >$*, step = 1* **do**
        Create neural network with 6 neurons in hidden layer
        Train neural network for 3000 epochs
        Collect data from training
    end for
    Average the results and save them
end for

7.2 Results of the test

Main data points that are collected in this test are: *accuracy* on test and train data and *time* it took to train the neural network in seconds and epochs.

Every neural network was trained before limit of epochs it was set to be trained for, meaning that every neural network reached goal of training. It is not surprise then, that accuracy on training data was 100% in every case. Because of that it does not have a graph. But what is more interesting is accuracy on test data which can be seen on Figure 33.

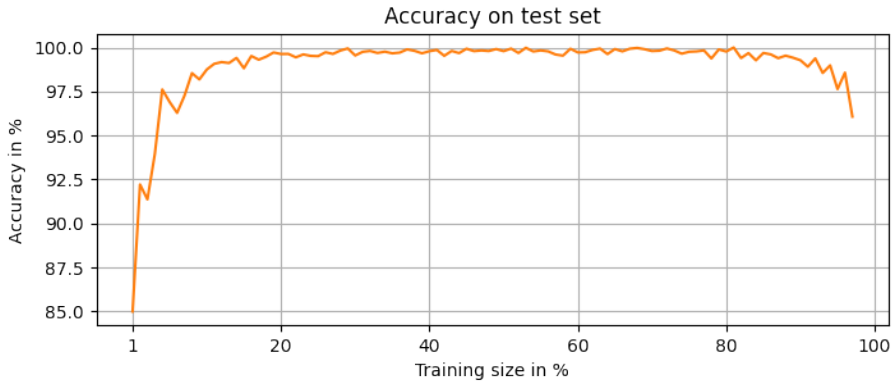

Figure 33: Accuracy

What Figure 33 shows is that even when neural network was presented with small % of the dataset, it was able to successfully classify rest of the dataset.

Another data that was collected and is important, is time that it took to train neural network with differently split datasets. This measurement can be seen on Figure 34, where training time was measured in epochs as well as in real time - seconds.

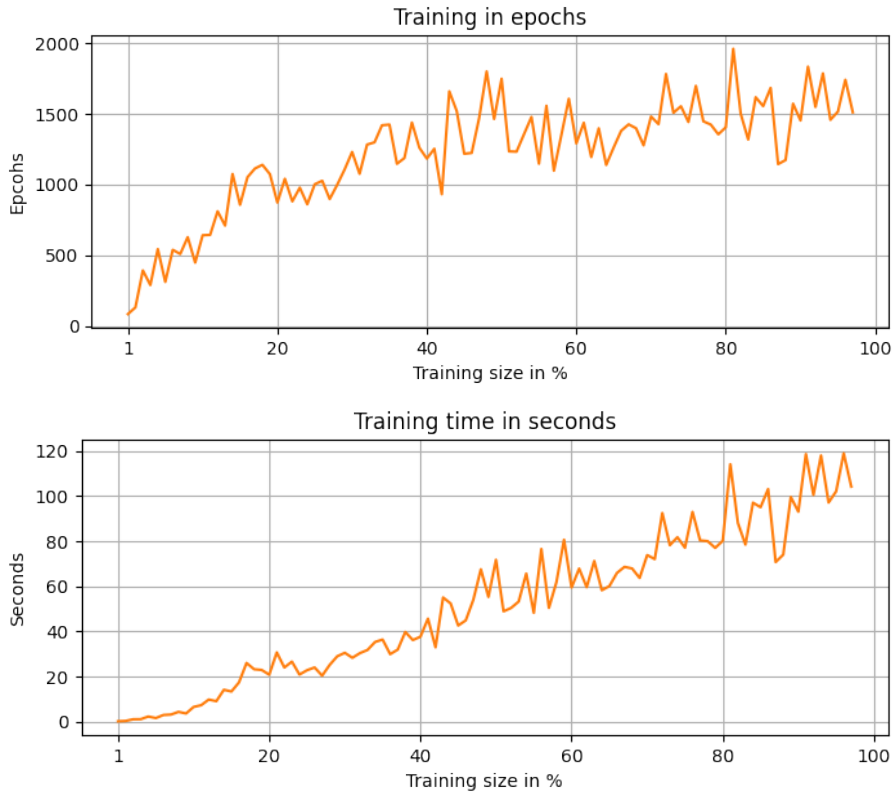

Figure 34: Training times

As can be seen from Figure 34 time to train the neural network goes roughly linearly with the % of the training data that it was presented to be trained on. But as can be seen on Figure 33, accuracy gets to the 100% mark when presented with 20% of the dataset. After that accuracy can not be improved more and additional training is unnecessary.

7.3 Conclusion

This test has shown that on given dataset neural network needs small percentage of the data to be presented with, on order to classify whole dataset successfully. When presented even with 1% of whole dataset neural network had accuracy of 85% on the rest of the whole dataset. Another observation that can be made is that training time positively correlates with size of the presented data. This means that in order to improve the accuracy for a few %, the training time can raise in order of magnitude. Of course in this project it is not a big problem because number of samples is relatively small, but this needs to be thought about of when making neural network with bigger datasets.

8 Kohonen network

Kohonen networks are a type of neural network that perform clustering, also known as self-organizing map. This type of network can be used to cluster the dataset into distinct groups when you do not know what those groups are at the beginning.

In a case of this project we know to which groups are data points belonging, but before the test they are removed. This way, with knowing what the result should be, we can look at the accuracy of the methods.

Firstly, before creating the self organizing map, it is needed to know to how many clusters should data belong. For this are used 2 methods: Elbow method and Silhouette analysis

8.1 Elbow method

Elbow method works in a way that it iterates trough different number of clusters. After the cluster centres are find, distortion value is calculated. Distortion is the sum of squared distances from each point to its assigned center. This can be plotted as can be seen on Figure 33.

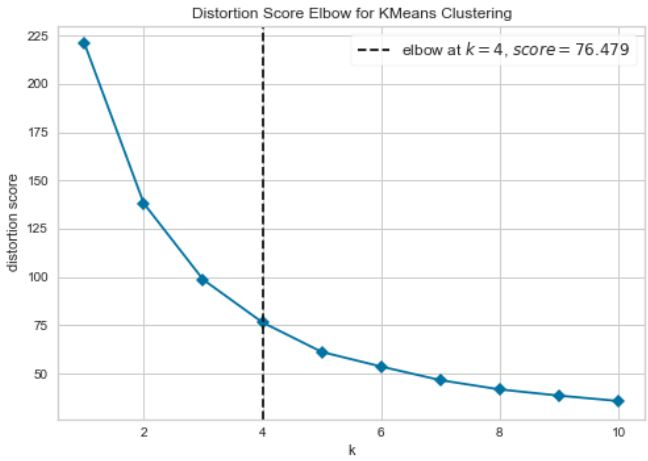

Figure 35: Elbow method

This method is also know as graphical method, because it takes the result from a graph. Number of clusters that should be used can be found on the elbow (or knee) of the graph, which is the point of inflection on the curve. In this case this point occurs when number of clusters is 4.

8.2 Silhouette analysis

Silhouette analysis is more computationally demanding method, but offers deeper information about the clustering. At first it works the same as elbow method. Number of clusters is chosen and their centers are found. After that Silhouette coefficient is calculated for each point. This coefficient says how much does the point belong to its group and not to the closest other group. Coefficient is scaled to range $< -1, 1 >$ where 1 indicates that point is classified well, and -1 that point is classified poorly.

Figure 34 shows the Silhouette coefficient for each data point in the dataset, for different number of clusters. Vertical red line shows average Silhouette coefficient of the whole dataset.

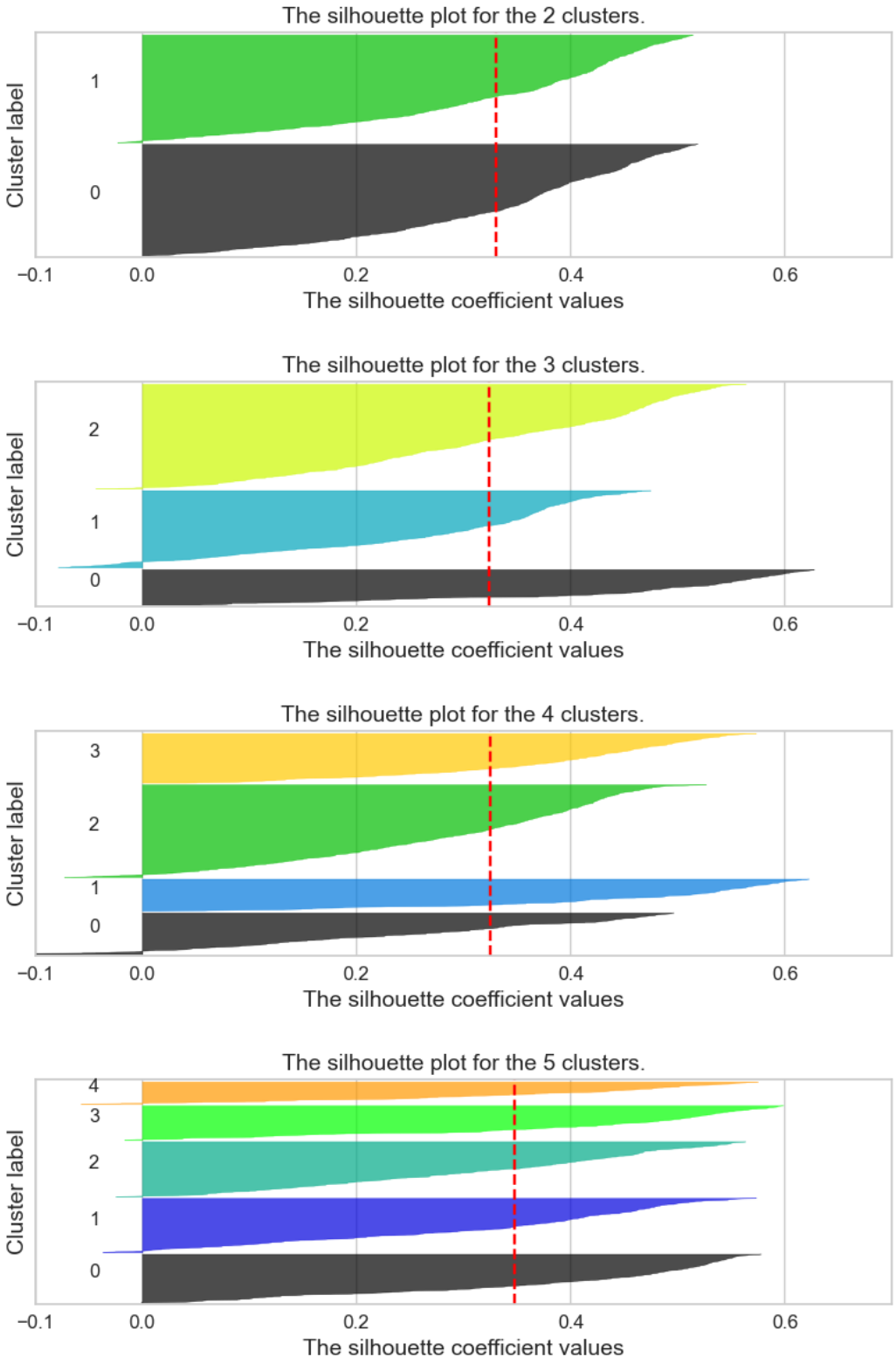

Figure 36: Silhouette coefficient with different number of clusters

These average values can be taken and plotted at graph.

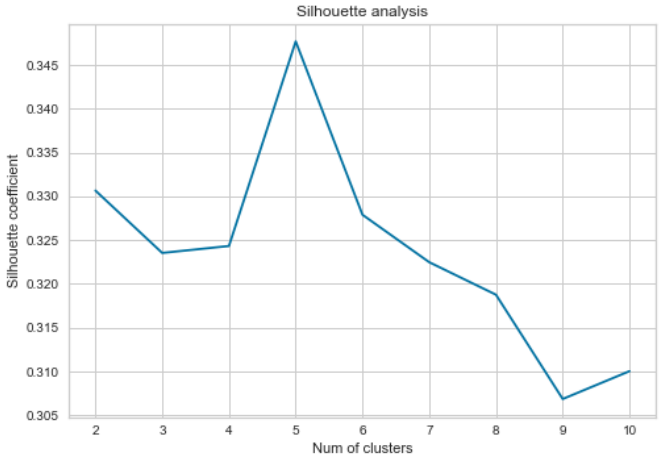

Figure 37: Average Silhouette coefficient

Figure 37 shows that best average Silhouette coefficient was with 5 clusters.

8.3 Conclusion

Judging by the given information from elbow method and silhouette analysis I would probably chose number 5 as number of clusters for this data. Even when elbow method suggests 4 clusters, distortion score is similar for 5 clusters. Especially when looking at Figure 34, which shows for 4 clusters more uneven distribution and also more values with negative score than plot with 5 clusters.

Also what can be seen from Figure 34 when knowing the actual number of clusters is 2, is that for 2 clusters there was least number of points that had negative silhouette coefficient when compared to different number of clusters.

9 Final conclusion

Chapter 2 has visualised the banknote dataset and shown that the dataset contained minimum number of outliers. From the 3D plot of dataset was born idea for the following test - to try to split the dataset linearly - by a plane. Chapter 3 as per assignment was first neural network that successfully classified the dataset. Chapter 4 tried to split the dataset by a plane by a perceptron, and was relatively successful. From extracted weights of trained perceptron could be observe that one of the attributes of the dataset - Entropy - does not give a much of information about the class of data point. Chapter 5 tried experiments with neural network with one hidden layer. It has shown that this dataset can be fully classified by neural network with 2 layers. Another characteristic of the neural network that was observerd, is that lesser the number of neurons, longer time it took the train the neural network. Also accuracy of neural networks in this test suggests that learning goal could be set too low. Chapter 6 experimented with neural networks that had 2 hidden layers. Accuracy of trained networks was worse than the ones from previous chapter, which could suggest possible overfitting. Chapter 7 contains test with differently split dataset, which had until this moment unchanged ratio. Result was that neural network on given dataset needs small % of whole dataset to be trained on, in order to successfully classify the whole dataset. Chapter 8 was about self organizing maps, and about a way to find optimal number of clusters. Both methods - Elbow method and Silhouette analysis suggested wrong number of clusters, but experienced person could from the data provided by Silhouette analysis guess the right number of clusters.