**Assignment 2**

Machine Learning, SS2021

|  |  |  |
| --- | --- | --- |
| **Team members** | | |
| Last name | First name | Matriculation Number |
| Rodiga | Luca | 1620972 |
| David | Tamara | 11813320 |

Task 1.1.2)

Underfitting: Underfitting can be recognized, when both the accuracy of the training set and the test set are not satisfyingly high. If for example the train and test accuracy are both around 40 to 50 % the model is probably just to weak for good results. This can be changed by adding more layers, neurons, iterations etc.   
  
Overfitting: Overfitting occurs, when the model gets too sophisticated at recognizing the training data and therefore does not perform well for any other data like the test data. It can be identified, when the training accuracy gets really good, but the test accuracy gets worse. For example, if the model is trained for more epochs and the training accuracy rises, but the test accuracy suffers, the model is overfitting. We can combat this by using regularization techniques and early stopping.

10 neurons

Train accuracy: 0.9054. Test accuracy: 0.7772

Loss: 0.3214

100 neurons

Train accuracy: 1.0000. Test accuracy: 0.8426

Loss: 0.0080

200 neurons

Train accuracy: 1.0000. Test accuracy: 0.8329

Loss: 0.0064

With 200 neurons overfitting starts to become a problem, because the test accuracy actually decreased. Therefore we chose 100 neurons for the next task.

1.1.3)

----- Task 1.3 -----

Amount of neurons: 10

Early Stopping

Train accuracy: 0.7471. Test accuracy: 0.6804

Loss: 0.7742

alpha = 1

Train accuracy: 0.8720. Test accuracy: 0.7990

Loss: 0.6369

alpha = 1 and Early Stopping

Train accuracy: 0.7605. Test accuracy: 0.6973

Loss: 0.8940

Amount of neurons: 100

Early Stopping

Train accuracy: 0.8945. Test accuracy: 0.7772

Loss: 0.3202

**alpha = 1**

**Train accuracy: 0.9945. Test accuracy: 0.8620**

**Loss: 0.4361**

alpha = 1 and Early Stopping

Train accuracy: 0.9642. Test accuracy: 0.8305

Loss: 0.5200

Amount of neurons: 200

Early Stopping

Train accuracy: 0.8945. Test accuracy: 0.7845

Loss: 0.2800

alpha = 1

Train accuracy: 0.9970. Test accuracy: 0.8668

Loss: 0.4327

alpha = 1 and Early Stopping

Train accuracy: 0.9297. Test accuracy: 0.8039

Loss: 0.5962

Yes both alpha = 1 and early stopping combat overfitting, with alpha = 1 and no early stopping being the best option for all tests. So this is our choice.

1.1.4)

The seed of the random state determines the initialization of the weights, biases etc. of the model. So in short it determines the starting point from which the model starts learning. Depending on the other parameters this can lead to different results, but without early stopping and 500 epochs the model converges to very similar values.

----- Task 1.4 -----

seed= 1

Train accuracy: 0.9964. Test accuracy: 0.8620

Loss: 0.4361

seed= 2

Train accuracy: 0.9958. Test accuracy: 0.8838

Loss: 0.4376

seed= 8

Train accuracy: 0.9945. Test accuracy: 0.8644

Loss: 0.4361

seed= 32

Train accuracy: 0.9958. Test accuracy: 0.8717

Loss: 0.4348

seed= 64

Train accuracy: 0.9951. Test accuracy: 0.8668

Loss: 0.4360

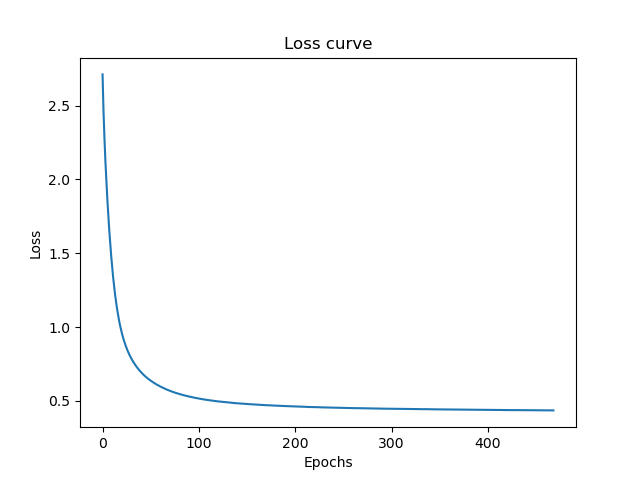
On the train set: 0.9955 +/- 0.0006

On the test set: 0.8697 +/- 0.0077

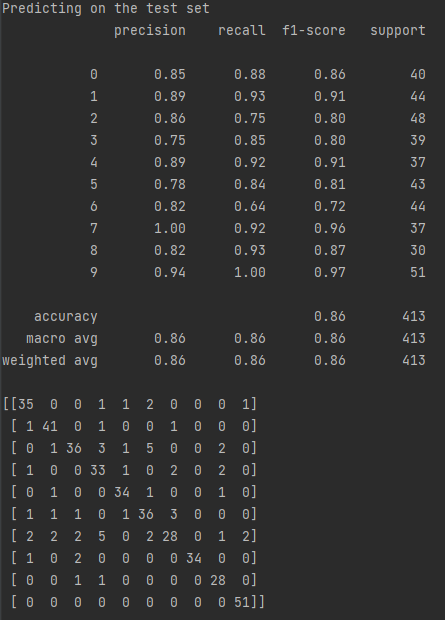
Max accuracy: 0.8838

Min accuracy: 0.8620

1.1.5)



1.1.6)



Recall = true positives / true positives + false negatives

This means recall is the amount of all correct positive classifications divided by the total amount of predictions made for this class. We can get the number of correct positive classifications from the confusion matrix, by looking at the diagonal values, where actual class equals predicted class and divide it by the support.

6 was the most misclassified image.

1.2)

Every possible configuration of the parameters defined gets checked. Therefore 4\*2\*2\*2 = 32 architectures get checked.

Best score:

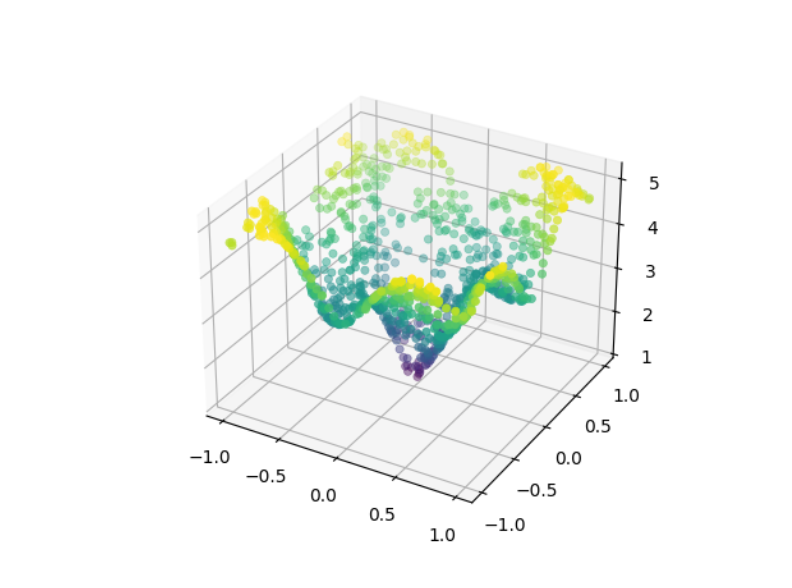
0.852638850511191

Best params:

{'alpha': 1, 'hidden\_layer\_sizes': (100,), 'learning\_rate\_init': 0.001, 'solver': 'lbfgs'}

Task 2)

2.1)



2.3)

To find a good model we used Gridsearch with following parameters:

parameters = {  
 'alpha': [0, 0.001, 0.003],  
 'learning\_rate\_init': [0.001, 0.002],  
 'solver': ['lbfgs', 'adam'],  
 'hidden\_layer\_sizes': [(50,), (30,)]  
}

n\_hidden\_neurons\_list = [5, 100, 300]

2.4)

The final Loss was 0.0014 (Train MSE). The best loss in this case would be 0.0018 (Test MSE)