TDT4171 — Artificial Intelligence Methods
Assignment 7 - Neural Networks Intro

Erik Storås Sommer - 535006

March 2023

Feedforward Neural Network

The hyperparameters used for the neural network are shown in Figure 1. The weights and biases in the network is tuned using the gradient descent algorithm, and the loss function is the mean squared error. The network is trained for 100000 epochs, and the learning rate is set to 0.0001. Tried different epochs and learning rates, and found that these values gave the best results.

```
128     # Hyperparameters
129     input_dim = 2
130     hidden_dim = 2
131     output_dim = 1
132     learning_rate = 0.0001
133     n_epochs = 100000
134
135     # Initialize the neural network
136     fnn = FeedforwardNeuralNetwork(input_dim, hidden_dim, output_dim, learning_rate)
```

Figure 1: Hyperparameters used for the neural network

The output from running the attached python file is shown in Figure 2. The output for the loss function is gradually decreasing as the network is trained, hinting that the network is learning. To see the performance of the network, the mean squared error for the train and test set is calculated and printed before and after training. As shown in the output, the mean squared error for the train set is ≈ 0.126, and the mean squared error for the test set is ≈ 0.130, both after training. This means that the network is able to generalize well, and is not overfitting the data.

```
eriksommer@Eriks-MBP neural_networks_intro % python3 feedforward_nn.py
Mean squared error before training:
MSE on training set: 1.4581820317956977
MSE on test set: 1.422638613516768

Training...
Epoch: 10000, Loss: 0.034590634412170654
Epoch: 20000, Loss: 0.006313296673767849
Epoch: 30000, Loss: 0.0059759118231347596
Epoch: 40000, Loss: 0.005864734875517536
Epoch: 50000, Loss: 0.00581583862786607
Epoch: 60000, Loss: 0.005791149316773886
Epoch: 70000, Loss: 0.005777150969968153
Epoch: 80000, Loss: 0.005768332671284548
Epoch: 90000, Loss: 0.005762220655606435
Epoch: 100000, Loss: 0.00575761924494556

Mean squared error after training:
MSE on training set: 0.1267457922344287
MSE on test set: 0.13068611337586902
```

Figure 2: Output from running the attached python file