Statistical Machine Learning 2018
Assignment 4
Deadline: 11th of January 2019

Christoph Schmidl
s4226887
`c.schmidl@student.ru.nl`

Mark Beijer
s4354834
`mbeijer@science.ru.nl`

December 3, 2018

Exercise 1 - Logistic regression (weight 5)

Part 1 - The IRLS algorithm

Part 2 - Two-class classification using logistic regression

Exercise 2 - Neural network regression (weight 5)

We train a neural network using backpropagation, to learn to mimic a 2D multimodal probability density. First, we implement the network and test its regression capabilities on a standard Gaussian; then we train it on the real data set. Visualization of the network output plays an important role in monitoring the progress.

2.1

Create a plot of an isotropic 2D Gaussian $y = 3 \cdot \mathcal{N}(\mathbf{0}|\frac{2}{5}\mathbf{I}_2)$ centered at the origin using the `meshgrid()`, `mvnpdf()` and `surf()` functions. Sample the density at 0.1 intervals over the range $[-2, 2] \times [-2, 2]$ and store the data in colum vector variables $\mathbf{X}$ (2D) and $\mathbf{Y}$ (1D).

Answer:

2.2

Implement a 2-layer neural network with $D = 2$ input nodes, $K = 1$ output nodes and M hidden nodes in the intermediate layer that can be trained using a sequential error backpropagation procedure, as described in Bishop §5.3. Use $tanh(\cdot)$ activation functions for the hidden nodes and a linear activation function (regression) for the output node. Introduce appropriate weights and biases, and set the learning rate parameter $\eta = 0.1$. Initialize the weights to random values in the interval $[-0.5, 0.5]$. Plot a 2D graph of the initial output of the network over the same $[-2, 2] \times [-2, 2]$ grid as the Gaussian (again using `surf()`).

Answer:

2.3

Train the network for $M = 8$ hidden nodes on the previously stored $\mathbf{X}$ and $\mathbf{Y}$ values (the $\{x_1, x_2\}$ input coordinates and corresponding output probability density y), by repeatedly looping over all datapoints and updating the weights in the network after each point. Repeat for at least 500 complete training cycles and monitor the progress of the training by plotting the output of the network over the $\mathbf{X}$ grid after each full cycle. Verify the output starts to resemble the Gaussian density after some 200 cycles (all be it with lots of 'wobbles').

Answer:

2.4

Permute the **X** and **Y** arrays to a random order using the `randperm()` function, keeping corresponding x and y together. Repeat the network training session using this randomized data set. Verify that convergence is now much quicker. Can you understand why? Try out the effect of different numbers of hidden nodes, different initial weights and different learning rates on speed and quality of the network training. Explain your results.

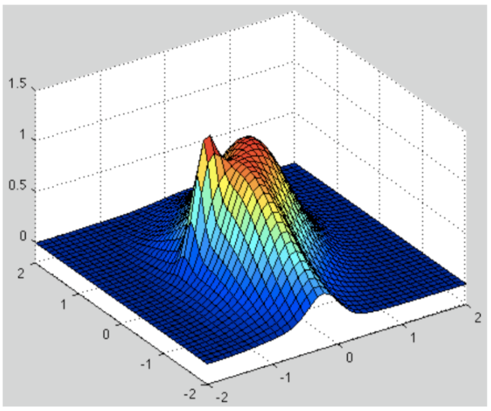

Figure 1: Multi-modal probability density

After these preliminaries we are now going to train the network on the real data set.
Load the data using

```
data = load('a017_NNpdfGaussMix.txt', '-ASCII');
X = data(:,1:2); Y = data(:,3);
```

Answer:

2.5

Create a $2D$-plot of the target probability density function. Notice that the data is in the correct sequence to use in `surf()`.

Answer:

2.6

Train the network on this data set. Use at least 40 hidden nodes and a learning rate parameter no higher than $\eta = 0.01$. Make sure the input data is properly randomized. Run the training phase for at least 2000 complete cycles and follow the progress by plotting the updated network output after every 20 full cycles. How does the final output of the network compare to the target distribution in the data? Explain. How could you improve the neural network in terms of speed of convergence and/or quality of the approximation?

Exercise 3 - Gaussian processes (weight 5)

Part 1 - Sampling from Gaussian stochastic processes

Part 2 - Gaussian processes for regression

Exercise 4 - EM and doping (weight 5)

3.2d

3.2e