KU Leuven

Data Mining and Neural Networks: **Update**

Akshat Dwivedi (Student Number: [redacted])

Instructor: Prof. Johann Suykens
Course: G9X29A

January 22, 2015

Contents

1 Curse of Dimensionality

1.1 m = 2

Similar to the case for m=1, the aim is to use a neural network to train the `sinc` function for 2 dimensions. This function is also known as the mexican hat function. A plot of this function over the 2-dimensional interval $[-5, 5]^2$ is given in figure 1.

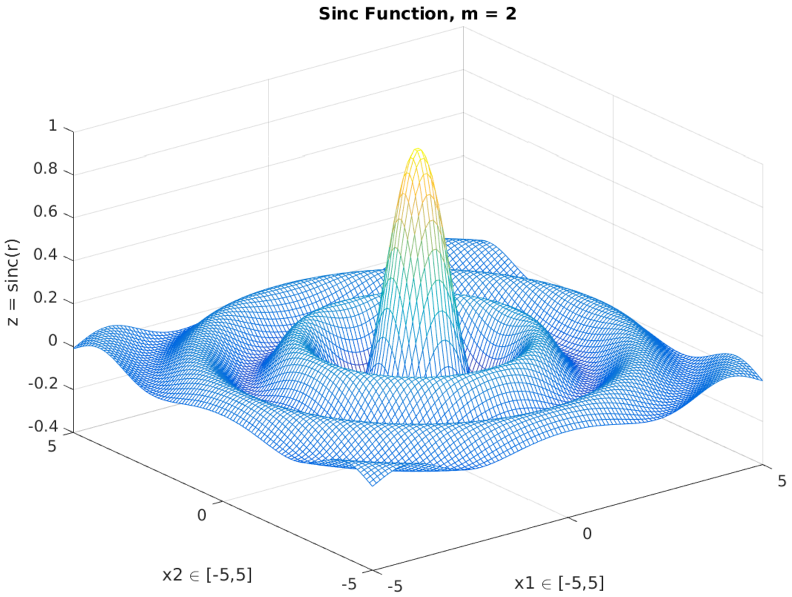
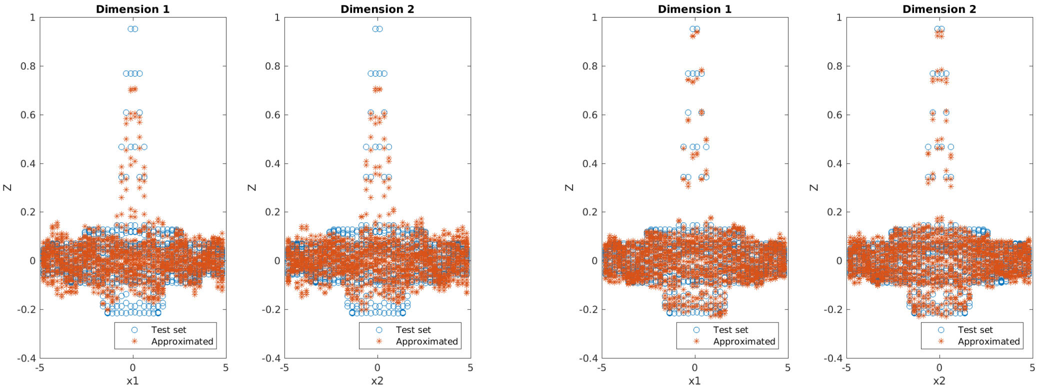


Figure 1: Mexican hat function on the 2-dimensional interval $[-5, 5]^2$

It is interesting to note, that in the 1-dimensional case, 100 points were used (training set) and there was a lesser amount of ”empty space” in the plot. Comparing this to the 2-dimensional case, we see that there is a lot more ”empty space” in 3-dimensions compared to 2-dimensions. If we take 100 equally spaced points for x_1 and $x_2 \in [-5, 5]$, we get a grid of $100^2 = 10{,}000$ points in the training set. Thus, we observe that the number of cases n has now increased from 100 to 10,000. Generally speaking, for an m-dimensional input space, the training set would contain 10^m cases. This is known as the *curse of dimensionality.*

We decide to go with the full training set with 10,000 points and use early stopping with a validation set consisting of 53 equally spaced points on [-4.9,4.9]. Thus, the validation set consists of $53^2 = 2809$ points. Finally, a separate test set of 40^2=1600 points on $[-4.8,4.8]^2$ is created to evaluate the network performance and is not used at the time of training the network. The advantage of selecting these three sets is that they are mutually disjoint and hence, the network is evaluated on sets which are similar, but not the same as the training set. The activation function function used is the radial basis function which is selected due to it's advantage in using a smaller number of neurons. Levenberg-Marquadt algorithm is used to train the network due to it's fast convergence rate, however resilient backpropagation was tried and resulted in a poorer fit compared to the LM algorithm. The network was trained with 25 and 35 neurons.



(a) 25 neurons

(b) 35 neurons

Figure 2: Mexican hat function in 2 dimensions. Plot of the test set with the approximated function on the test set. 25 and 35 neurons were tried.

In figure 2(a), we see that the network with 25 neurons is not able to adequately approximate the underlying function. The MSE on the test set is 0.0039. In 2(b) we see that the performance of the network with 35 neurons results in a rather good fit. The MSE was found to be 0.000494.

1.2 m = 5

Now we arrive at the interesting part of the exercise: evaluating the `sinc` function in 5 dimensions. The first problem we run into is that the training and test sets cannot be chosen in a manner that was employed for the 1- and 2-dimensional cases. Selecting 100 equally points on a 5-dimensional grid, for example, will lead to 100^5=10,000,000,000 observations which is simply too impractical. Furthermore, it would be challenging to visualize the approximated function due to the dimension being greater than 3.

Thus, we compare the fitted models based on MSE on the training and test sets. 5 networks are trained with 100, 200, 300, 400 and 500 nodes in the hidden layer. Furthermore, we used the scaled conjugate gradient algorithm to train the network as it is suitable for large scale problems as it does not store large matrices in memory. Training set was chosen to be 15 equally spaced points on the $[-5, 5]^5$ interval which results in $15^5 = 759375$ cases used to train the network. A validation set of 10 equally spaced points was chosen on the $[-4.9, 4.9]^5$ which resulted in 100000 cases in the validation set. Similarly, a test set of 10 equally spaced points was chosen on the $[-4.8, 4.8]^5$ which resulted in 100000 cases in the test set. Early stopping was used to train the network, however, due to time constraints, we trained the network for 200 epochs and training was stopped whichever of these two occurred first. It was observed that increasing the number of neurons did not improve the fit. An attempt was made to assess convergence with the Levenberg-Marquadt algorithm, however, the training time was simply too high.

This indicates that the successes of this approach for the 1- and 2-dimensional cases would not be appropriate here. Thus, we would have to resort to sampling methods to choose the training set on $[-5, 5]^5$. Uniformly sampling from this interval will not result in adequate training sets, as it will be give higher weight to the observations away from the center than the case for the sinc function. Sampling from a weighted mixture distribution might result in an adequate training set. This appears to be a non-trivial task.