

Dimensionality Reduction

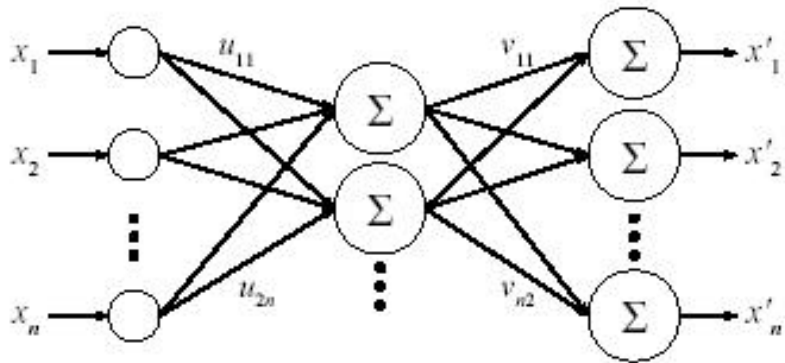
Amin Rahimi
Spring 2005

Problem

- Input data may have some excess dimensions that do not greatly affect the output of the neural network. We want to eliminate these extra dimensions to increase efficiency and speed of the network.

We are given a set of data that may or may not have excess information. We assume that there exist some major components of this data that have a large effect on the output of the data while others have a negligible effect. We would like to reconstruct the original data in terms of an optimal number of these valuable components while yielding a very low reconstruction error.

Neural Network Architecture



We see that the architecture for this network is linear and straight-forward. The input vector components are weighted and summed in the hidden node and the outputs of the hidden nodes are weighted and summed to give the output vector.

Procedure

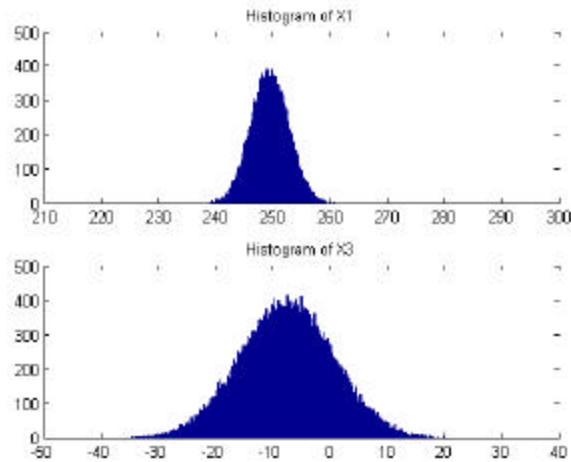
- Z-scale the data
- Find the eigenvectors and eigenvalues of the covariance matrix. Determine the intrinsic dimensionality of the data.
- Determine the intrinsic dimensionality and principal components using the auto-associative neural network

We will first Z-scale the data so that there is no initial bias toward any of the components. The data will be randomized into a training set, a test set, and a validation set.

We will use first use the technique of finding eigenvectors of the covariance matrix to perform PCA.

We will then use the neural network architecture described earlier to perform a similar analysis.

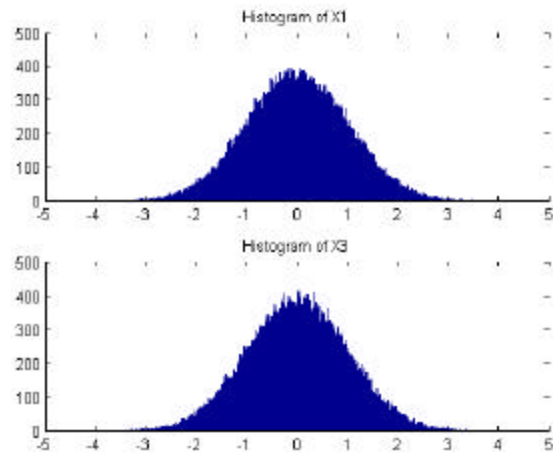
Z-Scaling the Data



Here, we see the histogram of two components, X1 and X3, of the original data. Looking at the x-axes, we see that the mean of X1 lies around 250 while that of X3 is around -7 . We also see, from the width of the two histograms, that the variance of X3 is significantly larger than that of X1.

Because these components vary in scale and offset, and because we want all components to be favored equally, we will use a process called Z-scaling. That is, we will subtract from each component its overall mean and we will divide by the overall standard deviation of that component. Evaluation of the expected value and variance equations shows that this process results in data with zero mean and a standard deviation of 1.

Z-Scaling the Data



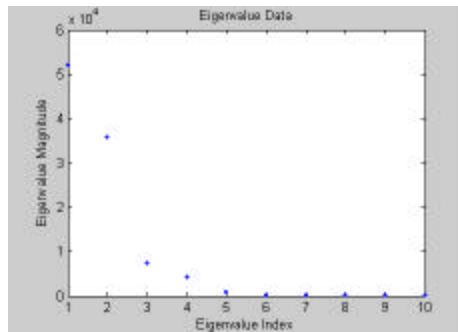
This is a histogram of the X1 and X3 data after Z-scaling. We can see that both components now have zero mean and equal variance.

Finding Eigenvectors

Find the covariance matrix, $R = 1/N XX^T$

Determine the eigenvalues and eigenvectors of this matrix

The largest eigenvalues correspond to principle components



Intrinsic dimensionality = 5

If X is our input data matrix, we calculate the covariance matrix as $R = 1/N XX^T$ where N is the number of elements in the set.

We then use MATLAB to find the eigenvectors and corresponding eigenvalues of this matrix. The eigenvectors corresponding to the largest eigenvalues of the matrix are the principal components of the data set.

From the plot above, showing the magnitude of the eigenvalues with respect to their indices, we can see that eigenvalues with indices higher than 5 are very small. We therefore conclude that the intrinsic dimensionality, or the number of principal components of the data, is 5.

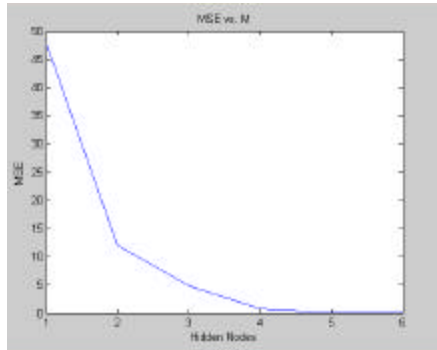
Neural Network Implementation

- Training set size = 80,000 samples. Validation set size = 20,000 samples.
- Set learning rate, α , to 10^{-6} .
- We want the output of the network to approximate the input. Mean square error is the output measured with respect to the input.
- We begin with 1 hidden node, allow the network to train, and freeze the weights.
- Once converged, we add another hidden node.
- This process is continued until there is no marked improvement in MSE.

The goal of this network is to reconstruct the input accurately with the smallest number of hidden nodes possible.

We begin with 1 hidden node. In this case, to minimize the mean square error, the weights will be updated such that the outputs will lie along the major axis of the data (the principal component). Once this network has been allowed to converge, we freeze the hidden layer weights and add another hidden node. The weights for this hidden node will then be updated such that the output vector can also vary along the first minor axis, or the second principle component, of the data. This process is continued until the decrease in mean square error is negligible when adding hidden nodes.

Results



Intrinsic Dimensionality = 5

Reconstruction error = .0138

The plot above shows the minimum mean square error achieved as a function of the number of hidden nodes in the network. We see that above $M = 5$, there is no noticeable decrease in mean square error, so our intrinsic dimensionality is 5 as we had expected from earlier calculations.

When the validation set was run through this network with 5 hidden nodes, the reconstruction error was found to be .0138. The data can therefore be very accurately approximated with only 5 components.

Upon inspection of the network weights, we find that the vector of input weights to the first hidden node is approximately equal to the eigenvector corresponding to the largest eigenvalue. Similarly, the vector of input weights to the second node is approximately equal to the eigenvector corresponding to the second largest eigenvalue.

This comparison was made by subtracting the n th row of the weighting matrix, U (transposed), by the n th eigenvector and summing all the components. This yielded values lower than .5 for matching vectors and greater than 10 for those that didn't match.

We conclude that our neural network was able to implement principal component analysis through the use of gradient descent.

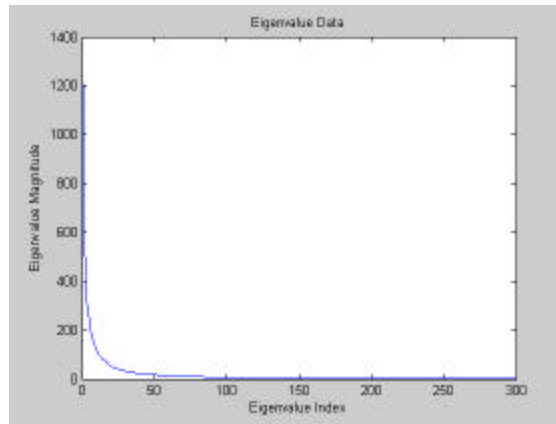
Isolet Data

- Repeat process with Isolet data set
- We let the training set be 4/5 of the data and the validation set be 1/5
- Set learning rate, α , to 10^{-6}
- Initial MSE = 622.14, approximately 617

We will perform the same procedure to analyze the Isolet data. This time, we will use 4/5 of the data as our training set and hold the remaining 1/5 for the validation set. Alpha will again be 10^{-6} .

The initial MSE is approximately 617, or 1 for each input component, as expected.

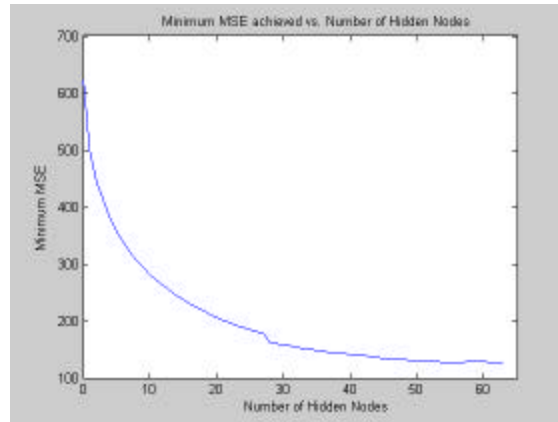
Finding Eigenvectors



The intrinsic dimensionality is about 140.

The graph above shows the eigenvalues that correspond to the eigenvectors of the covariance matrix. We see that these eigenvalues become small after about the 140th index. Therefore, we conclude that the intrinsic dimensionality is 140.

Results



Estimated Intrinsic Dimensionality = 80
Reconstruction error = 126.347

The plot above shows the minimum MSE that was achieved as a function of the number of hidden nodes. Though a clear intrinsic dimensionality is not easily seen, we can estimate it as the value of M where the MSE curve flattens out.

Unfortunately, the continually increasing dimensionality of the neural network slowed the computation down significantly, so only up to 64 hidden nodes were attempted. By the shape of the curve, we can estimate the intrinsic dimensionality to be somewhere around 80. This is significantly smaller than what we estimated using the eigenvalues of the covariance matrix. However, if we take another look at the plot of the eigenvalues, we see that there is no clear cut-off present in that plot.

Because both of these estimations were made without much prior experience, we do not conclude that this method for finding intrinsic dimensionality is not valid. In fact, we find that in this case, the hidden layer weights for each node again correspond to the eigenvectors that we calculated.

We found the reconstruction error to be 126.347 using the validation set using 64 nodes.

Conclusions

- PCA can help to greatly reduce the amount of unnecessary data in a given set
- The neural network is effective in finding the intrinsic dimensionality of a data set
- The process can require extensive computation
- Data may not have unnecessary components

We were able to find a good model for the generated data that relied on only 5 components. The neural network, along with the eigenvector calculation, was able to find the intrinsic dimensionality fairly well.

The clear disadvantage of the network was the long computation time required to carry out the calculations. We saw that adding a single node greatly decreased the speed of the network.

Our evaluation of the Isolet data revealed that there may not have been an overwhelming number of unnecessary components. We saw that at $M=64$ hidden nodes, we still had a very large mean square error. Thus, we wouldn't have been able to find a good set of approximated data without several days' worth of computation time.