Homework 2

ECE 662

4/1/2008

Question 1 Parametric Method Revisited

During the in class discussion on parametric method, the approach of drawing a separation hyper-plane between two classes was discussed. A hyper-plane was found by obtaining the argmax of the cost function, $J(w) = \frac{w^T S_B w}{w^T S_W w}$, known as w_0. The solution is found to be $w_0 = S_w^{-1}(m_1 - m_2)$, where m_1 and m_2 are the sample means of the two classes being separated by the hyper-plane. The issue being raised in this section is can the cost function simply be replaced by $J(w) = w^T S_B w$, which would effectively set S_W as the identity matrix.

Six different numerical experiments were performed exploring this line of thought. First, exploring the simplified cost function, classes that are well separated but not distributed normally is investigated. Specifically in this case, the classes are Africa and Europe and the feature vectors are crude birth rate (x-axis) and life expectancy at birth (y-axis).

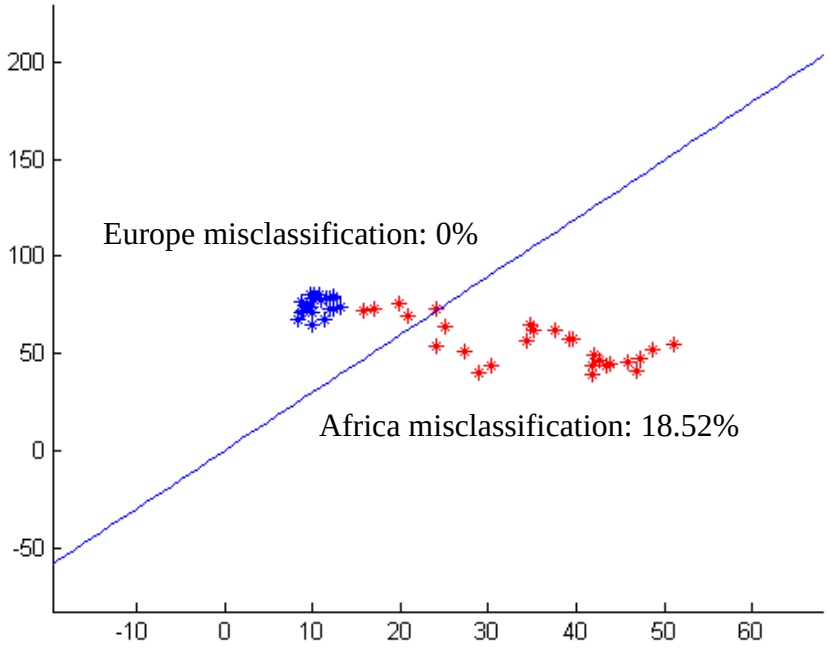

Figure 1 Europe vs. Africa hyper-plane from simplified cost function

As can be seen from the results in Figure 1, the classification for Europe is very good, but can be attributed to a close clustering of the data, and not necessarily to a well drawn hyper-plane. A hyper-plane that would lower the percentage of African countries being misclassified would be more desirable, therefore no definitive conclusions can be drawn from only this test. Next, the same feature vectors and same classes will be used, however, the more complicated cost function will be used.

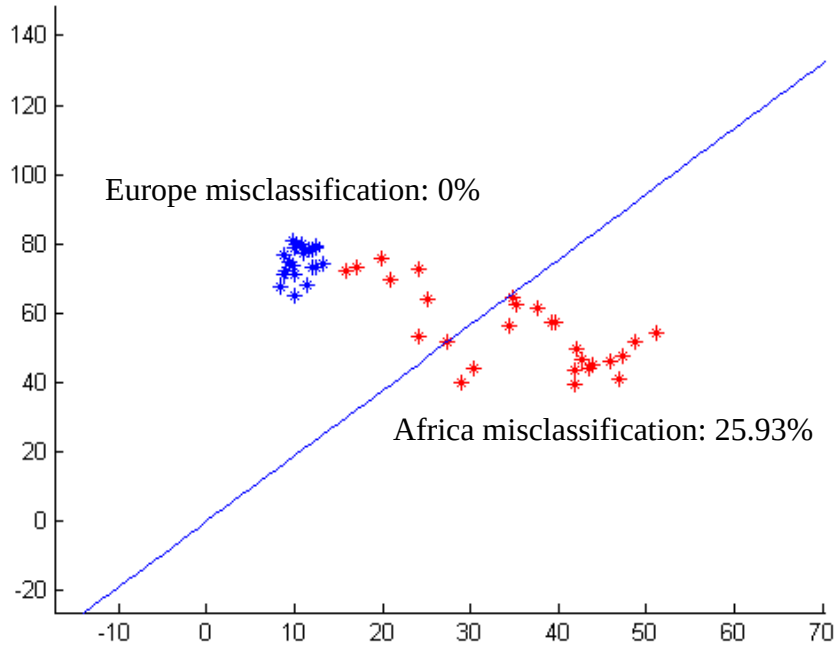

Figure 2 Europe vs. Africa hyper-plane from original cost function

From the results in Figure 2 it would seem that the simpler cost function would result in a better hyper-plane. However, this is not a conclusive result because the Europe class is very well clustered and could be affecting the algorithm. Therefore, in the next test, the same feature vectors are used but the Europe class is changed to Asia. Once again the simpler cost function is tested out first.

As can be seen in Figure 3, neither class is well separated from the other, nor are the data points normally distributed. Still the Africa misclassification percentage is lower than in the previous case with the complex cost function and a well separated Europe class. However, before any conclusions are drawn, the complex cost function must be applied to this new data. Figure 4 demonstrates that the more complex function both

decreased the error of one class, but dramatically increased the error of the other. Therefore, one more set of experiments will be done to demonstrate the effectiveness of the either functions.

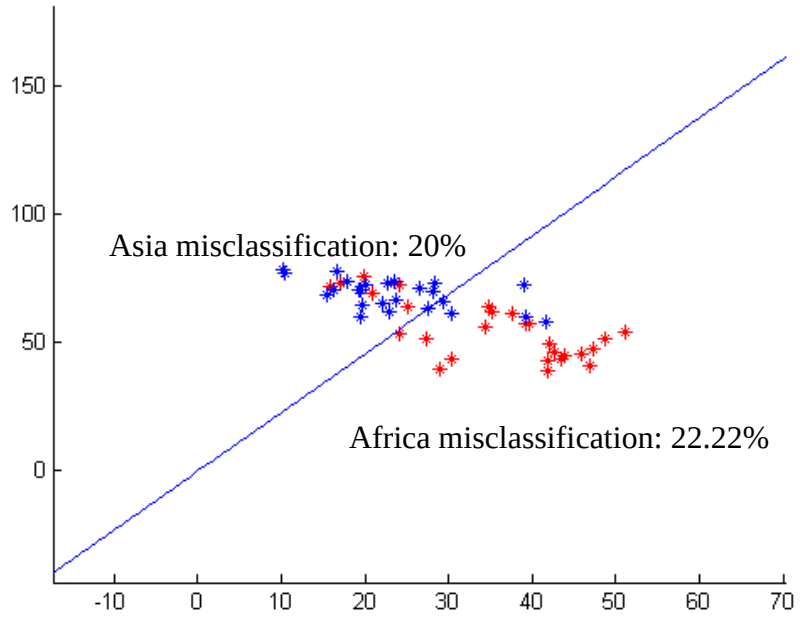

Figure 3 Asia vs. Africa hyper-plane from simplified cost function

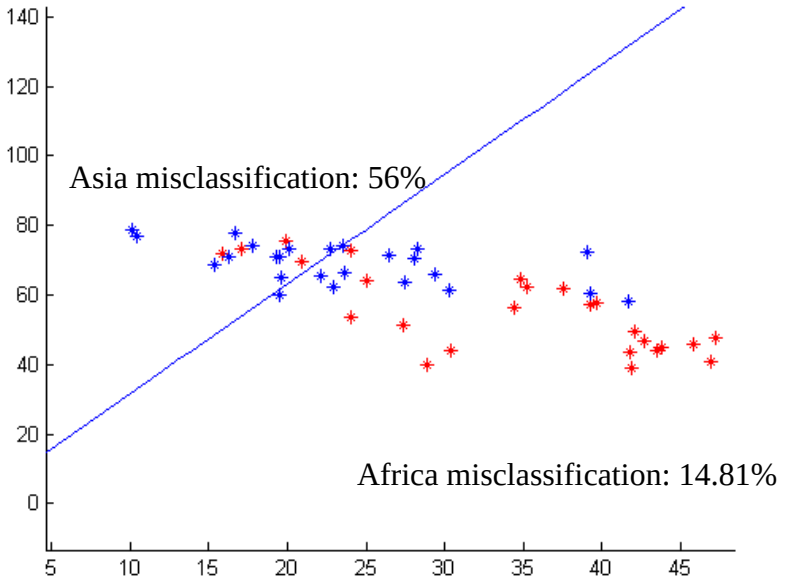

Figure 4 Asia vs. Africa hyper-plane from original cost function

For the next set, data will be created in a similar fashion as that from the previous homework. The data will be somewhat uncorrelated and in a random normal distribution (means of (-8, -8), (8, 8) and variances of 10 for both). Once again the less complex equation is used first.

Once again a reasonable hyper-plane is created, but as before no conclusions can be drawn until the complex function is taken into account. In this case, the percentages of misclassification for both classes were lowered.

Therefore, drawing a definitive conclusion is difficult because of the mixed results from the experiments. The clearest and most concise conclusion that can be given is that it depends on the data being classified. In some cases, such as the clearly clustered data, the simplified equation lends itself to bettering the hyper-plane. However, in other cases such as the two normal distributions, the more complex was better.

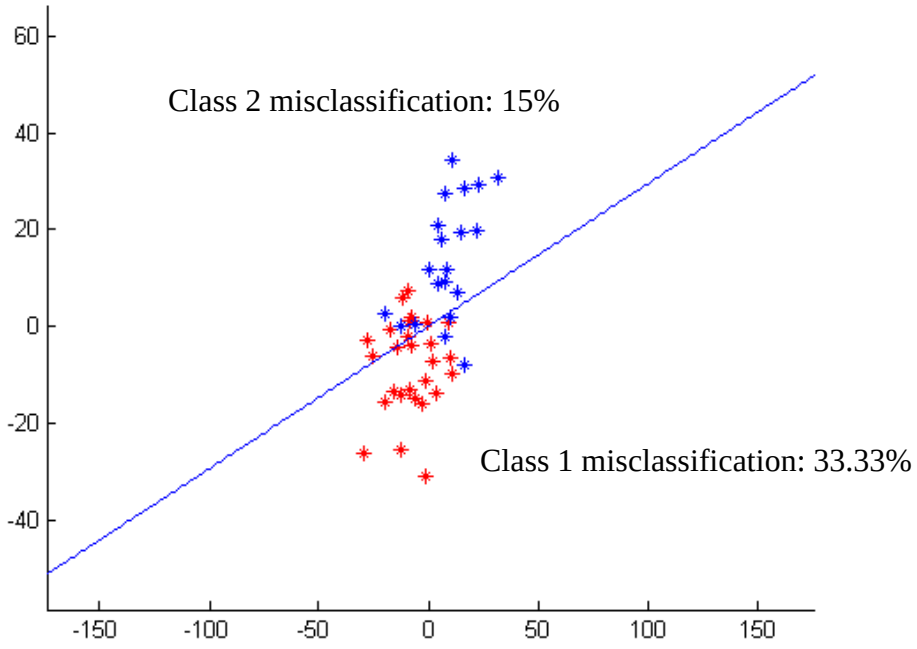

Figure 5 Normally distributed data hyper-plane from simplified cost function

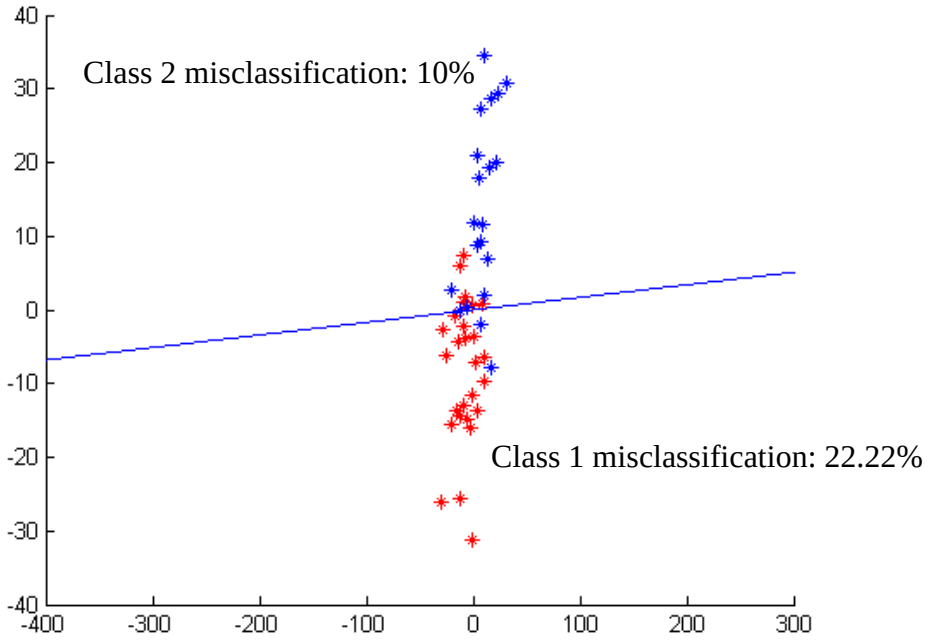

Figure 6 Normally distributed data hyper-plane from original cost function

Question 2 ANN vs. SVM

The basic point of this problem is to design a classifier based on each method and then compare them. To make a fairly straightforward and relatively simple experiment, for both the artificial neural network and support vector machine, all the available feature vectors will be used. This will make for an interesting experiment because some of the vectors are well separated as in Figure 1 and 2, while others are not so clearly separated as in Figure 3 and 4.

For the ANN, the type of function to be used in the hidden layer is decided upon. During the first phase each training point is put through the network for the purpose of adjusting the weights through back propagation. The cost function that is used is $J\left(\bar{w}\right)=\frac{1}{2}\sum_{i=1}^{d}\left\|\vec{t}_i-\vec{z}_i\right\|^2$ and optimized using the gradient descent. The output is then determined by the output of the hidden layer functions and the weights between the hidden layer and output layer. After the outputs have been determined, they are compared to the actual data, and the weights are adjusted accordingly. Once the training data has

been run through, the test data is put through and no adjustments are made. Whereas, for the SVM, the training data is loaded into the program. Then the hyper-plane that maximizes the distance between the two classes is then determined. Since there are only two classes being classified, there is no real need for a kernel trick of any kind.

For the neural network the sigmoid function $f(x) = \frac{1}{1+e^{-x}}$ was the main consideration. A Gaussian function was also considered, but the error was much greater than that of the sigmoid functions. Therefore, it was not pursued further. The mean-squared error was calculated for different number of nodes in the training set as well as in the test set. The results are shown in Table 1.

Table 1 Mean-squared error for different number of nodes in a neural network

	Sigmoid			Gaussian
number of nodes	1	3	10	3
mean-squared error (training)	11.10%	10.50%	10.10%	
mean-squared error (testing)	11.4522%	11.5310%	11.8255%	13.0547%

Upon inspection of the results, several interesting occurrences take place. One being, that as the number of nodes increase the mean-squared error based on the training data decreases. However, namely, that as the number of nodes increases so does the mean-squared error based on the testing data. One possible explanation as to why this problem arises is that as the number of nodes increases, so does the over-fitting. One way to reduce the chances of that from happening is to add more data points. Unfortunately, in this case, there are a fixed number of points, half of which must be set aside for testing purposes.

The support vector machine in this experiment is only able to handle two classes at a time. As in a similar fashion to the neural network, all the feature vectors were included. Two specific cases were explored using the SVM. First, was the case of Africa versus Asia. As previously seen in Figure 3 and 4, there is a portion of the data overlapping resulting in misclassification. In Figure 7 it can be seen that as the rate of false positives increases, the rate of true positives rises. It rises slowly at first and then

quickly at the very end. However, a rate of 60% false positives is needed to do slightly better than guessing (50%) true positives.

There was another case discussed in the first question. Namely, deciding between the classes of Africa and Europe. This set contained a noticeably separate Europe cluster from a spread out Africa. Even with just a hyper-plane in that case, the decision surface was fairly accurate (< 20% error). When put through the SVM with a false positive rate of approximately 5%, the SVM can attain approximately 90% true positive accuracy.

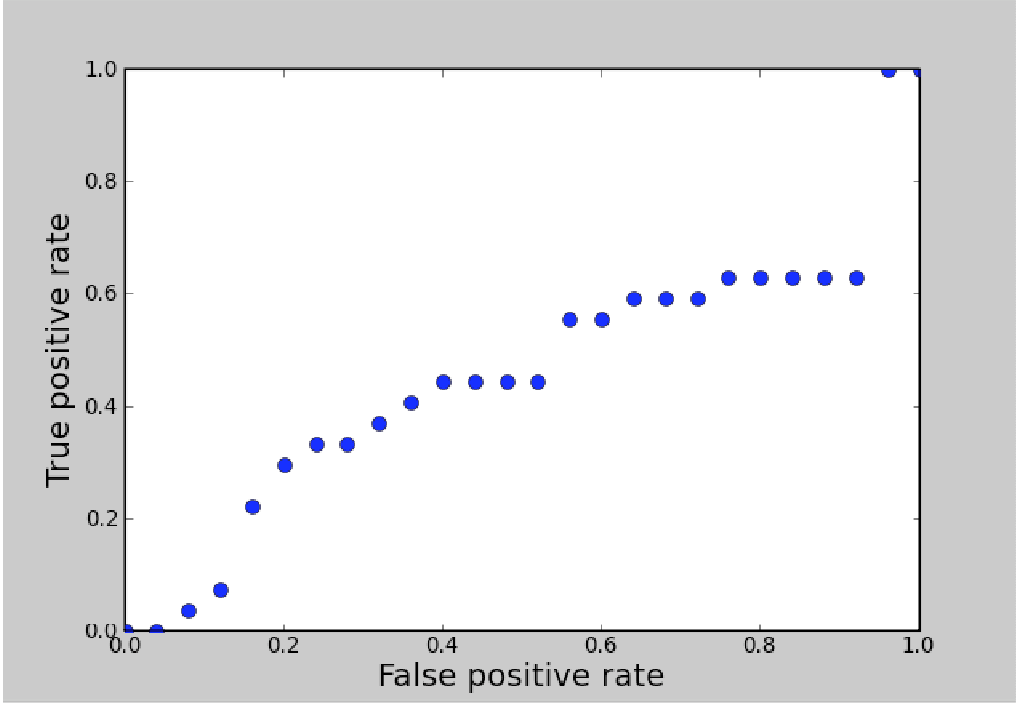

Figure 7 False positives vs. True positives for the Africa vs. Asia SVM

Comparing the results of the ANN and SVM, in one case the ANN performs much better, and in the other, the SVM performs better.

In side by side comparison it would seem that the neural network would tend to be better because of cases such as Africa versus Asia. However, that is only a tendency, as was shown with Europe versus Africa, under certain conditions the SVM is able to outperform the neural network. This is similar to the conclusion drawn earlier about the results are not definitive, but rather, depend on the data being used.

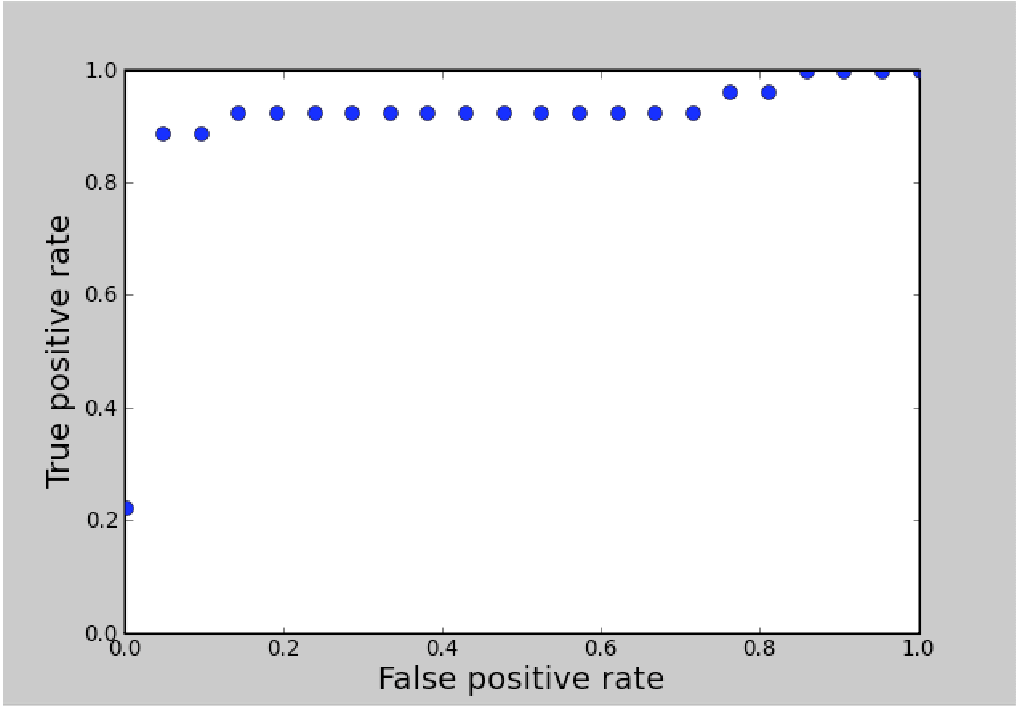

Figure 8 False positives vs. True positives for the Africa vs. Europe SVM

Question 3 Parzen Window vs. KNN vs. NN

Similar to the previous question, in this case the objective is to design three different types of classifiers based on the three methods of Parzen windows, K nearest neighbor, and nearest neighbor. Unlike in the previous question with the SVM that dealt with two classes at a time. In this case, all six classes will be included. This results in a different guessing accuracy of 16.67% than in previous cases of 50%. This means that anything below 83.33% error will be considered better than guessing.

All three of the programs follow the same basic method. A test point is taken and compared to all the training points one at a time. A distance is calculated for each training point using the Minkowski metric for various values of p. That value is put into a matrix along with its respective class. This distance matrix is then sorted showing the closest training point at the top.

For the Parzen window method, each distance is weighted using a Gaussian window. The value of the distance is given to be the mean while the variance is varied to experiment with different levels of accuracy. The function is then evaluated at zero. This has the effect of translating the coordinate system of a given test point to the origin and

then rotating it so that the training point being weighted is on the x-axis. All the weights for the different classes are added, and the class with the largest tally is assigned to that particular test point. This is repeated for every test value.

For K nearest neighbor, the same sorting algorithm is used at the beginning. However, instead of weighting the distances as in Parzen window, each class is given a weight of one. Also, since there is no variance to vary, the value for K is varied. The value of K is used to determine how many of the closest training points to consider. What that relates to in the distance matrix is the number of top rows to consider. Then, as in Parzen window, all the votes for each class are added, and the largest tally is assigned to that particular test point.

Lastly, the nearest neighbor uses the same sorting algorithm as the others. However, the number of rows to consider in the vote depends on a specified value within the range of distances. This could create a problem because some feature vectors represent percentages of populations while others represent portions of populations in the thousands. In order to minimize the necessary work to determine an appropriate value limit, a percentage of the range is used and added to the closest distance.

There are many degrees of freedom that can be manipulated in each of these techniques: The dimensionality of the feature space, the grouping of feature vectors, the type of metric, and the parameters specific to each method. The results in the following tables are the best of the method specific parameter (variance, K, percent of range) manipulations that produced the lowest percentage of error. Also, for the KNN and PW, the best training feature vector was paired with each test feature vector.

From these results it can be seen that parameter values depend largely on the data being used. In many cases, as the method specific parameters were increased, the error would decrease up to a point, and then start to increase. In this case the actual class of each test point was known, so calculating the error and seeing the various trends was relatively simple. However, in a real system, where the class of the test points are unknown, a great deal of time would need to be invested in analyzing what values the parameters need to be set at.

One of the biggest problems, especially in KNN, was the fact that there was a disproportionate amount of data for some classes. When it came to large values of K the

shear number of Africa training points would cause misclassifications. North America was never assigned to test vector because there was only one training point, and it could not get a majority vote.

Drawing a simple conclusion of which technique is better is very difficult with the results of the various experiments. If based on accuracy, the ranking would go KNN, NN, PW because the best for each was 32.6%, 34.7%, and 35.7%, respectively. If based on precision, the order would be NN, KNN, PW because the worst for each was 48%, 67%, 74.5%, respectively. However, despite the poor performance of Parzen window, based on the assumption made at the beginning of the question, it is still more accurate than guessing between 6 regions.

Table 2 Percent error for various parameters of KNN and PW

	K Nearest Neighbor			**Parzen Window**		
	vector(s)	kval	%error	vector(s)	variance	%error
1D	7	12	37.7551	4	3	40.8163
Manhattan	2,8	25	50	2,8	8	55.102
	3,2	15	61.2745	3,8	17	61.2245
	4,7	30	36.7347	4,6	3	63.2653
	5,4	16	67.3469	5,4	10	74.4898
	6,4	9	51.3469	6,8	12	56.1224
	7,4	30	36.7347	7,4	3	41.8367
	8,4	24	43.8776	8,4	7	44.898
Euclidean	2,8	14	45.9184	2,4	15	44.898
	3,6	4	51.0204	3,8	8	54.0816
	4,5	14	38.7755	4,5	15	35.7143
	5,4	16	35.7143	5,4	5	35.7143
	6,8	1	47.9592	6,4	12	47.9592
	7,7	12	37.7551	7,4	3	40.8163
	8,4	7	36.7347	8,4	2	41.8367
p=100	2,8	15	45.9184	2,7	1	61.2245
	3,8	8	50	3,4	11	60.2041
	4,5	23	32.6531	4,8	19	51.0204
	5,4	23	32.6531	5,4	4	36.7347
	6,8	1	48.9796	6,8	6	50
	7,7	12	37.7551	7,4	3	40.8163
	8,4	6	38.7755	8,4	4	42.8571

Table 3 Percent error for various parameters of NN

Nearest Neighbor								
% of Range	1D		2D manhattan		2D Euclidian		2D p=100	
	vector	% error	vector pair	% error	vector pair	% error	vector pair	% error
5	4	42.8571	7,4	42.8571	8,4	41.8367	4,4	42.8571
10	7	41.8367	7,4	41.8367	4,5	36.7347	4,5	37.7551
15	7	44.898	7,4	44.898	4,5	35.7143	4,5	36.7347
20	4	41.8367	7,4	40.8163	4,7	40.8163	4,5	34.6939
25	4	47.9592	7,4	43.8776	4,5	41.8367	4,5	40.8163
30	4	46.9388	7,4	44.898	4,5	45.9184	4,5	42.8571