ECE662: Pattern Recognition and Decision Making Processes: HW TWO

Purdue University
Department of Electrical and Computer Engineering
West Lafayette,
INDIANA, USA

Abstract. In this report experiments are carried out to evaluate some of the techniques that we have covered in the $ECE662$ course material. In particular in the first exercise a numerical experiment is carried out to test effect of applying a Fischer linear discriminant approach to linearly separable data. An extra step is also take to test the effect of the technique on non-linearly separable data. In the second part of the experiment we look at the classifiers designed using Support Vector Machines and the Neural Network approach. The performance of the two methods on the same dataset is compared. The third part of the experiment covers the comparison of a classifier designed using the Parzen and K-Nearest Neighbor(KNN) techniques. In all the exercises some results are found to confirm the theoretical conclusions the in course. On some cases where results seem to be different than expected, a motivation is given based on the outcome. Three data sets were used and the complex of the data sets vary from linearly separable to non linearly separable.

1 Fisher's Linear Discriminant

This is a linear classification model that is in terms of dimensionality reduction. We will consider the case of 2 classes and suppose we take the D-dimensional input vector $\mathbf{x}$ and project it down to one dimension using:

$$y = \mathbf{w}^T\mathbf{x} \tag{1}$$

If we place a threshold on y and classify $y \geq -w_0$ as class C_1, and otherwise class C_2, then this projection to one dimension might lead us to loss of information and data that is well separated in D-dimensional space will end up being overlapped in one dimension. However, by adjusting the components of the weight vector $\mathbf{w}$, one can select a projection that maximizes the class separation. As mentioned above the experiment for this exercise will be for a two class problem. But first we will derive the cost function for maximizing the choice of the weights and the direction of optimal projection. We consider the problem to consist of N_1 points of class C_1 and N_2 points of class C_2 so that the mean vectors of the two classes are given by

$$\mathbf{m}_1 = \frac{1}{N_1} \sum_{n \varepsilon C_1} \mathbf{x}_n \tag{2}$$

$$\mathbf{m}_2 = \frac{1}{N_2} \sum_{n \varepsilon C_2} \mathbf{x}_n \tag{3}$$

From the above equations: The simplest measure of the separation of the classes, when projected onto $\mathbf{w}$, is the separation of the projected class means. This then suggest that we might choose $\mathbf{w}$ so as to maximize:

$$m_2 - m_1 = \mathbf{w}^T(\mathbf{m}_2 - \mathbf{m}_1)m_{1,2} = \mathbf{w}^T\mathbf{m}_{1,2} \tag{4}$$

However equation 4 can be made to be arbitrarily large simply by increasing the magnitude of $\mathbf{w}$. To solve this problem, we could constrain $\mathbf{w}$ to have a unit length, so that, [1]:

$$\sum_i w_i^2 = 1 \tag{5}$$

Using a Lagrange multiplier to perform the constrained maximization, we then find that:

$$\mathbf{w} \propto (\mathbf{m}_2 - \mathbf{m}_1) \tag{6}$$

Again with this approach we might still run into problems of classes overlapping due to the strongly non diagonal covariances of the class distributions. Fisher's idea to resolve this problem is outlined in the following: The idea is to maximize a function that will give a large separation between the projected class means while also giving a small variance within each class, thereby minimizing the class overlap.

Equation 1 transforms the set of labelled data points in $\mathbf{x}$ into a labelled set in the 1-dimensional space y. The within-class variance of the transformed data C_k is therefore given by:

$$s_k^2 = \sum_{n \varepsilon C_k} (y_n - m_k)^2 y_n \mathbf{w}^T\mathbf{x} \tag{7}$$

We can define the total within-class variance for the whole data set to be simply $s_1^2 + s_2^2$. Fisher's criterion is defined to be the ratio of the between-class variance to the within-class variance and is given by, [1]:

$$J(\mathbf{w}) = \frac{(m_2 - m_1)^2}{s_1^2 + s_2^2} \tag{8}$$

Making use of substitution of equations 1 to 7 equation 8 can be written in the form:

$$J(\mathbf{w}) = \frac{\mathbf{w}^T\mathbf{S}_B\mathbf{w}}{\mathbf{w}^T\mathbf{S}_W\mathbf{w}} \tag{9}$$

$\mathbf{S}_B$ is the $between - class$ covariance matrix and is given by:

$$\mathbf{S}_B = (\mathbf{m}_2 - \mathbf{m}_1)(\mathbf{m}_2 - \mathbf{m}_1)^T \tag{10}$$

$\mathbf{S}_W$ is the total $within - class$ covariance matrix, given by:

$$\mathbf{S}_W = \sum_{n \varepsilon C_1} (x_n - m_1)(x_n - m_1)^T + \sum_{n \varepsilon C_2} (x_n - m_2)(x_n - m_2)^T \tag{11}$$

Differentiating equation 9 with respect to $\mathbf{w}$, we find that J($\mathbf{w}$) is maximized when

$$\mathbf{w}^T \mathbf{S}_B \mathbf{w} \mathbf{S}_W \mathbf{w} = \mathbf{w}^T \mathbf{S}_W \mathbf{w} \mathbf{S}_B \mathbf{w} \tag{12}$$

From equation 10 we see that $\mathbf{S}_B\mathbf{w}$ is in the direction of $\mathbf{m}_2 - \mathbf{m}_1$. Furthermore we are not interested in the magnitude of $\mathbf{w}$, only its direction, and so we can drop the scalar factors, $\mathbf{w}^T\mathbf{S}_B\mathbf{w}$ and $\mathbf{w}^T\mathbf{S}_W\mathbf{w}$. Multiplying both sides of equation 12 by $\mathbf{S}_W^{-1}$ we then obtain:

$$\mathbf{w} \propto \mathbf{S}_W^{-1}(\mathbf{m}_2 - \mathbf{m}_1) \tag{13}$$

From equation 13 we notice that if the within class covariance is isotropic, so that $\mathbf{S}_W$ is proportional to the identity matrix, we find that $\mathbf{w}$ is proportional to the difference between the means as shown by equation 6 The above theory is applied to two different experiments: The first experiment makes use of *Fischer's iris data*. The data consists of 150 observations containing four measurements based on the petals and sepals of three species of iris. The species are: *Iris setosa, Iris virginica, Iris versicolor*. First we choose the two features from the two of any combination from the three species above and we calculate the optimal weights required to separate the data well.

Iris data optimal weights using:

$$S_W = \begin{pmatrix} 0.0492 & 0.0776 \\ 0.0776 & 0.2460 \end{pmatrix} \tag{14}$$

$$W = \begin{pmatrix} -7.9797 \\ -8.8588 \end{pmatrix} \tag{15}$$

The projection of the data to W is shown in fig 1. As can be seen from scatter plot of the original data, when this data is projected onto the one-dimensional plane, the data is well separated. It must be noted that the effect of Fischer's discriminant does not bring about any improvement in separating the data because the data was well separated already. In fig 2 we consider data that is not well separated. On this data we perform the calculation of optimal weights twice: *(a)* calculate optimal direction with S_w not set to an identity matrix and *(b)* calculate optimal direction with S_w set to I. The projection of the data to planes of optimal directions are shown in fig 3.

Setting S_W to the identity matrix:

$$S_{W_2} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \tag{16}$$

$$W_2 = \begin{pmatrix} -1.0800 \\ -2.7980 \end{pmatrix} \tag{17}$$

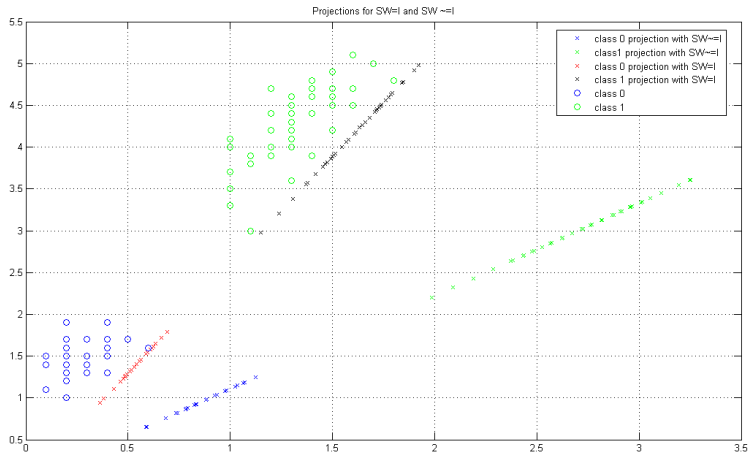

Fig. 1. Projection of Iris Setosa Feature 3 and 4 vs Versicolor Features 2 and 3

The second experiment consist of two class gaussian clouded data([2]).The optimal weights for the data in fig 4 are calculated and the direction of optimal class separation is established. The projection of data along the weights direction is shown in fig 5.

$$S_W = \begin{pmatrix} 2.0649 & 0.0514 \\ 0.0514 & 1.9720 \end{pmatrix} \tag{18}$$

$$W_{optimal} = \begin{pmatrix} 0.4745 \\ 0.4726 \end{pmatrix} \tag{19}$$

Setting S_W to the identity matrix

$$S_{W_2} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \tag{20}$$

$$W_2 = \begin{pmatrix} 1.0041 \\ 0.9565 \end{pmatrix} \tag{21}$$

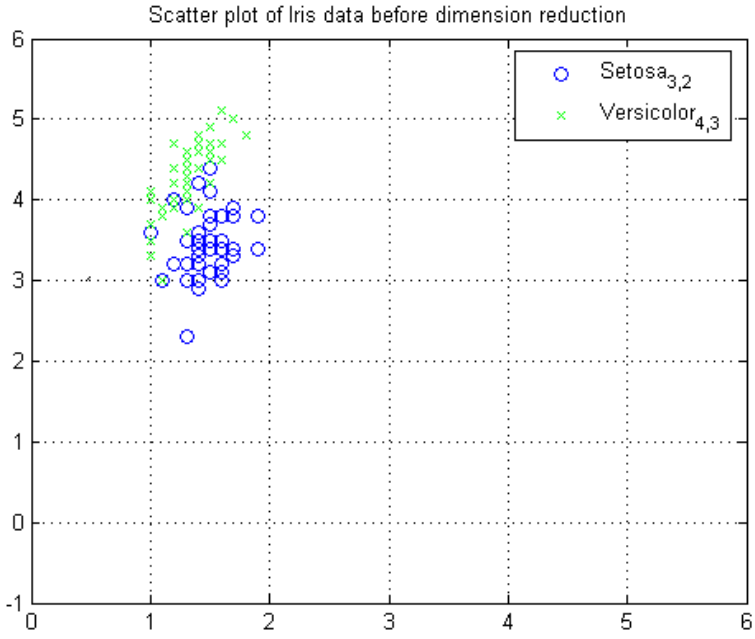

Fig. 2. Projection of Iris Setosa Feature 3 and 2 vs Versicolor Features 4 and 3

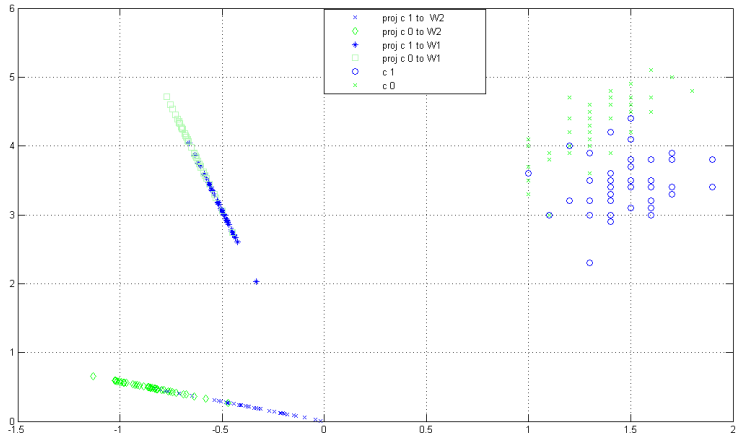

Fig. 3. Projection of Iris Setosa

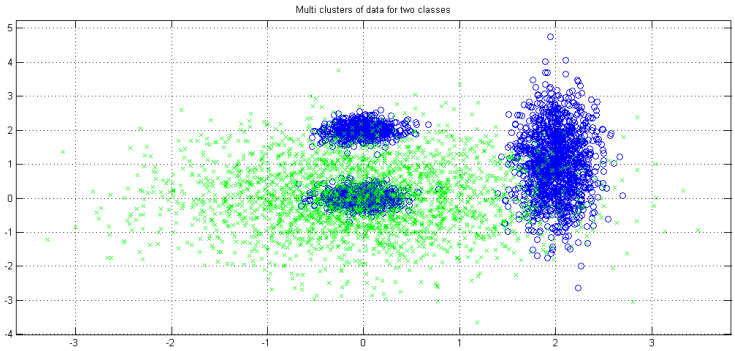

Fig. 4. Gaussian clouded data

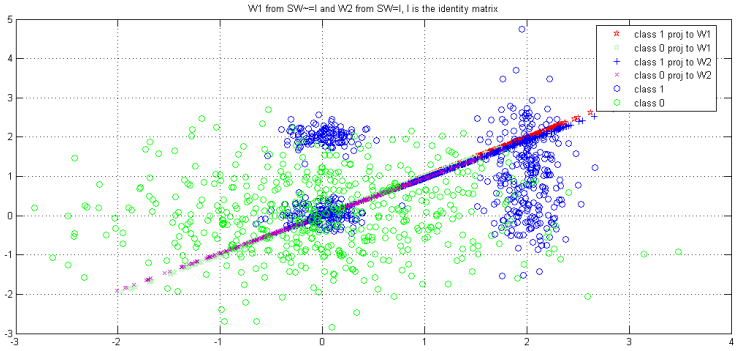

Fig. 5. Projection of clouds

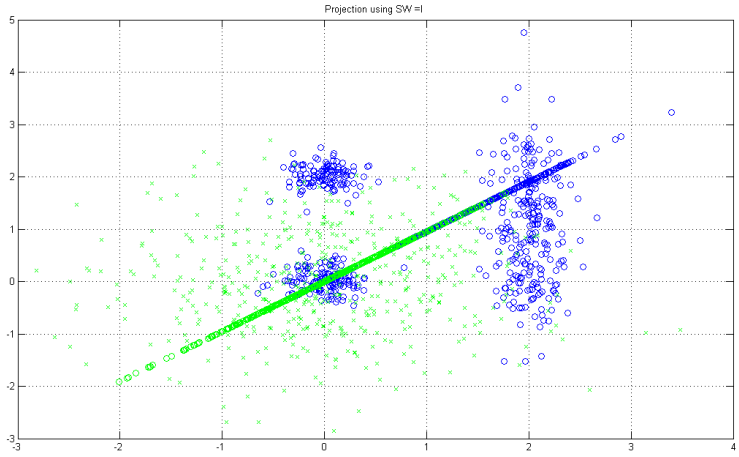

Fig. 6. Projection of clouds with few data points $S_W \neq I$

Classifying the data in fig 5 using the Maximum Likelihood to computer the parameters: MLE Classifier parameters are:

$$Class_0 \quad mean_0 = \begin{pmatrix} 1.1 \\ 0.85 \end{pmatrix} \tag{22}$$

$$Cov_0 = \begin{pmatrix} 0.838 & 1.08 \\ 1.08 & 1.4 \end{pmatrix} \tag{23}$$

$$Class_1 : \quad mean_1 = \begin{pmatrix} -0.304 \\ -1.48 \end{pmatrix} \tag{24}$$

$$Cov_1 = \begin{pmatrix} 1.88 & 0.788 \\ 0.788 & 0.526 \end{pmatrix} \tag{25}$$

1.1 Error Computation Table for the Bayes Decision Region

	Class 0	Class 1	Total error
Train Error	0.24	0.23	0.24
Test Error	0.26	0.24	0.23
Optimal Bayes Error	0.056	0.15	0.103

The classification performance from both of these experiments was carried out with 40 percent hold out of data and the rest used for training the classifier. Note that classification was only performed on the data in fig 4 and the projection planes shows no significant difference between $S_W = I$ and $S_W \neq I$. As can be observed from fig 5 for highly correlated data setting S_W to I yields no significant improvement in making the data more separable. Thus the conclusion that one

can draw from both the experiments above is: When data is linearly separable setting S_W to I can improve the computation of the weights and also the result is a weight direction that is not necessarily the same as when S_W is not set to the identity matrix. For non linear data setting S_W results in a weight direction almost the same as non identity S_W and also choosing the identity can improve the computation if the data is all isotropic.

2 Classification using Neural Network and Support Vector Machines

2.1 Neural Network (NN): What is a NN

An Artificial Neural Network (ANN) is an information processing paradigm that is inspired by the way biological nervous systems, such as the brain, process information. The key element of this approach is the novel structure of the information processing system. It is composed of a large number of highly interconnected processing elements (neurones) working in unison to solve specific problems. ANNs, like people, learn by example. An ANN is configured for a specific application, such as pattern recognition or data classification, through a learning process. Learning in biological systems involves adjustments to the synaptic connections that exist between the neurones . This is true of ANNs as well. ANN entails a learning process as described below.[for diagrams see (Bishop 2006)] The learning method to be used for this experiment is the Supervised learning:

2.2 Supervised Learning

Supervised learning incorporates an external form of teaching, so that each output unit is told what its desired response to input signals should be. During the learning process global information may be required. Paradigms of supervised learning include error-correction learning, reinforcement learning and stochastic learning. An important issue concerning supervised learning is the problem of error convergence, ie the minimization of error between the desired and computed unit values. The aim is to determine a set of weights which minimizes the error. In this exercise we will make use of the backpropagation algorithm for computing the weights and their correction.

2.3 Network Design

The neural network classifier used for this experiment consisted of: three layer Multilayer Perceptron optimized using the back projection algorithm,two inputs, one output, ten hidden nodes,a hyperbolic function for the hidden nodes and a sigmoid function for the output layer.

The behavour of an ANN (Artificial Neural Network) depends on both the weights and the input-output function (transfer function) that is specified for

the units. We present the network with training examples, which consist of a 100 training patterns of activities from each of the 2 classes, for the input units together with the desired pattern of activities for the output units (the target values). We determine how closely the actual output of the network matches the desired output. We change the weight of each connection so that the network produces a better approximation of the desired output using the backpropagation algorithm.(See [3] for the algorithm)

2.4 Neural Network Results

First we plot the distribution of the data from the two classes before classification.

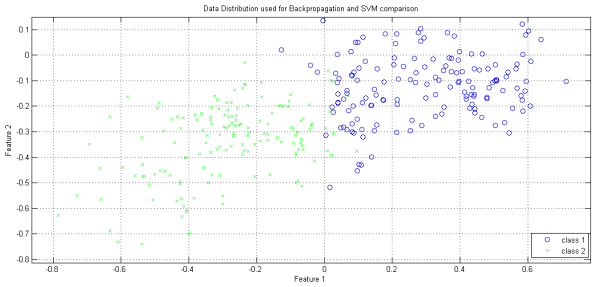

Fig. 7. Scattered gaussian data from two classes

The results from applying a Neural Network to the above data is displayed in fig 8 where the optimal bayes decision region is also shown.The data was divided using the holdout approach into: training and testing sets. The training data consisted of 2/3 of the original data shown in fig 8 and the test data consisted of a 1/3. The results for both sets including the bayes decision error are shown in the table below.

	Error Computation for NN Classifier		
	Class 0	Class 1	Total error
Hidden units	7		
Train Error	0	0.083	0.033
Test Error	0.23	0.067	0.05
Bayes Error	0.02	0.04	0.03

3 Support Vector Machines

In this section we will compare the neural network approach versus the support vector machines. As covered in class, we found that traditional neural networks approaches have suffered difficulties with generalization, producing models that

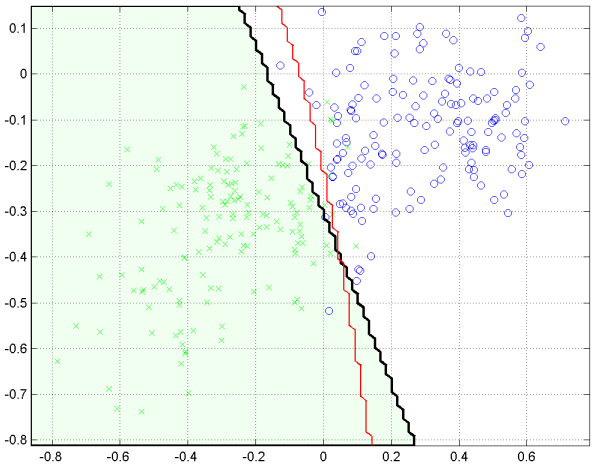

Fig. 8. NN Decision plane and Bayes Optimal Separation

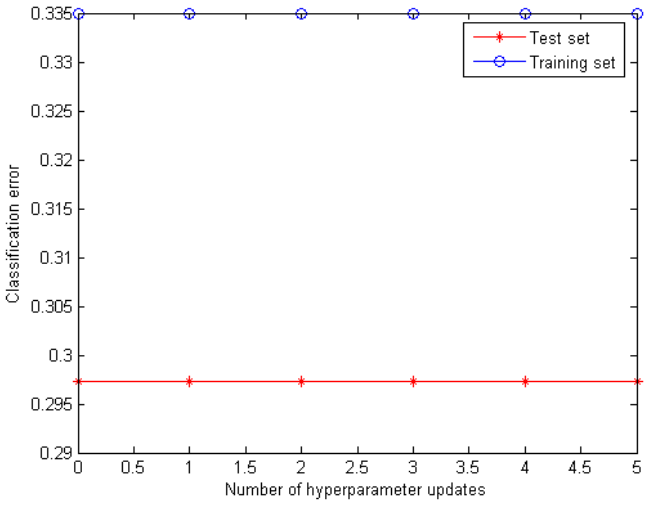

Fig. 9. Neural Network Classification Error on clouded data

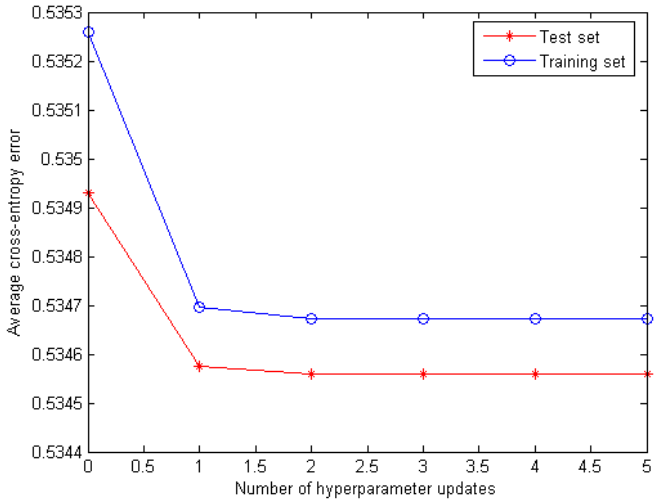

Fig. 10. Average Entropy

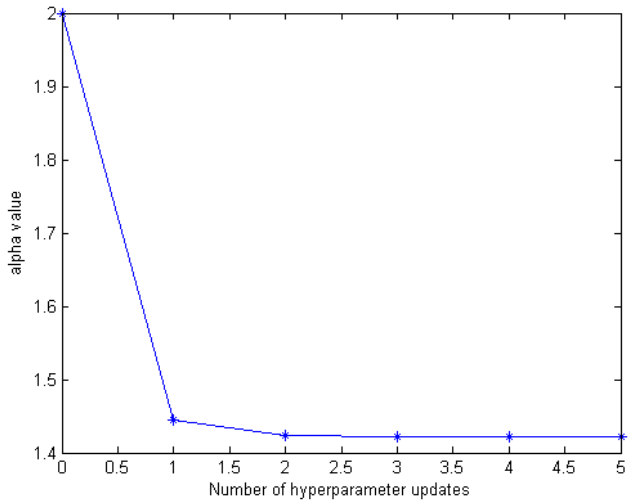

Fig. 11. The Value of alpha for regularization

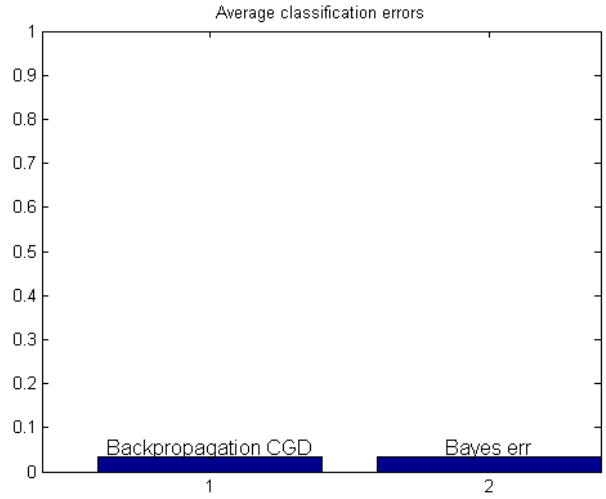

Fig. 12. Average Error Comparison between the two decision surfaces shown in fig 8

can over-fit the data. This in theory can be shown that it is a consequence of the optimization algorithms used for parameter selection and the statistical measures used to select the best model.On the other hand the formulation surrounding the support vector machines, "embodies the Structural Risk Minimization" (SRM) principle which has been shown to be superior,[4],compared to the Empirical Risk Minimization (ERM) principle which was used in the formulation of Neural Networks. SRM minimizes an upper bound on the expected risk, as opposed to ERM which minimizes the error on the training data, [4]. The mathematical outline of the above differences is out of the scope of this report.

As already explained in the Neural Networks section above, the problem covered here is of separating two classes. The objective is to design a classifier that performs within acceptable tolerance on unseen data, thereby generalizing well. For data that is linearly separable there could be more than one way to separate the data but the design of the classifier should an optimal separation plane that maximizes the margin.

The mathematical formulation of the above is as follows: Since we are considering the problem of separating the set of training vectors belonging to two separate classes,

$$D = \{(x^1, y^1), ..., (x^n, y^n)\}, \quad x\varepsilon R^n, y\varepsilon\{-1, 1\} \tag{26}$$

with hyperplane

$$< w, x > +b = 0 \tag{27}$$

An optimal separating hyperplane of the above data should satisfy the constraint:

$$y^j[< w, x^j > +b] \geq 1, \quad j = 1, .., n \tag{28}$$

The distance of the nearest points from the plane is given by,

$$d(w, b; x) = \frac{| < w, x^j > +b|}{\|w\|} \tag{29}$$

The optimal hyperplane can be found by maximizing the margin, g $w.r.t$ equation 28,

$$l(w, b) = min_{x^i:y^i=-1} d(w, b; x^i) + min_{x^i:y^i=1} d(w, b; x^i) \tag{30}$$

$$= min_{x^i:y^i=-1} \frac{| < w, x^j > +b|}{\|w\|} + min_{x^i:y^i=1} \frac{| < w, x^j > +b|}{\|w\|} \tag{31}$$

$$= \frac{1}{\|w\|} min_{x^i:y^i=-1} | < w, x^j > +b| + min_{x^i:y^i=1} | < w, x^j > +b| \tag{32}$$

$$= \frac{2}{\|w\|} \tag{33}$$

Thus the plane that optimally separates the data is the one that minimizes:

$$L(w) = \frac{\|w\|^2}{2} \tag{34}$$

The solution to this equation is given by saddle points of the Lagrange multiplier:

$$L(w, b, \alpha) = \frac{\|w\|^2}{2} - \sum_{i=1}^{n} \alpha_i (y^j[< w, x^j > +b] - 1) \tag{35}$$

where α are lagrange multipliers. the Lagrangian has to be minimized $w.r.t$ w, b and maximized $w.r.t$ $\alpha \geq 0$. The solution to equation 35 can be found in text,[4],[1],. It is formulated as follows:

$$\alpha = argmin_\alpha \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j < x_i, x_j > - \sum_{k=1}^{n} \alpha_k \tag{36}$$

with constraints:

$$\alpha_i \geq 0 \quad i = 1, ..., n \sum_{j=1}^{n} \alpha_j y_j = 0 \tag{37}$$

For non-linear non-separable data the above procedure is followed but with an introduction of non-negative variables, $\xi_i \geq 0$ and a penalty function:

$$F_\sigma(\xi) = \sum_i \xi_i^\sigma \tag{38}$$

Equation 28 is modified to:

$$y^j[< w, x^j > +b] \geq 1 - \xi_i, \quad j = 1, .., n \tag{39}$$

The optimization problem of equation 30 under the new constraints is given by:

$$L(w, b, \alpha, \xi, \beta) = \frac{\|w\|^2}{2} + C \sum_i \xi - \sum_{i=1}^{n} \alpha_i (y^i[< w^T, x^i > +b] - 1 + \xi_i) - \sum_{j=1}^{n} \beta_j \xi_j \tag{40}$$

where $\alpha_i \beta_i$ are the lagrange multipliers. The Lagrangian in this case should be minimized $w.r.t\ w, b, x$ and maximized $w.r.t\ \alpha, \beta$. The solution to this problem is found to be:

$$\alpha = argmin_\alpha \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j < x_i, x_j > - \sum_{k=1}^{n} \alpha_k \tag{41}$$

with constraints:

$$0 \leq \alpha_i \leq C \quad i = 1, ..., n \sum_{j=1}^{n} \alpha_j y_j = 0 \tag{42}$$

The solution above is identical as to the one found for the linearly separable data with the only difference being on the constraints : equation 42, which is the bounds on the Lagrange multipliers.The parameter C should be determined as it introduces additional capacity control within the classifier. This is the parameter that is can directly be linked to the the regularization parameter in the Neural Network approaches. In our experiment we will use a trial and error method to obtain this value for a separation that can best regularize the optimal decision plane. Theoretical methods of finding this parameter are out of the scope of the experiment. Plots of support vectors and Neural Network on non-linear non-separable data. These plots where generated from the SVM toolbox in Matlab that incorporates a comparison of multialgorithms applied to one dataset. The *SPRTool* has a tutorial and the full toolbox can be downloaded from [5]. Another multi-algorithms comparison that was used is the Matlab classification toolbox,[2]

3.1 support vector machine error table: fig 13

	Class 0	Class 1	Total error
Train Error	0	0.033	0.017
Test Error	0.017	0.058	0.037
Bayes Error	0.02	0.04	0.03
Support Vectors	300		
Margin	1.7508		

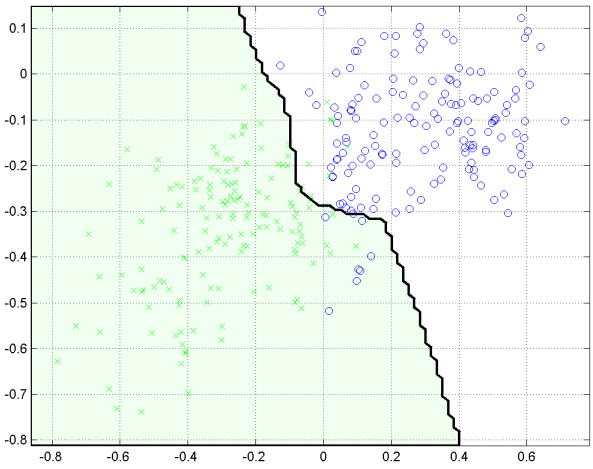

Fig. 13. Support Vector Machines on non-linearly separable data

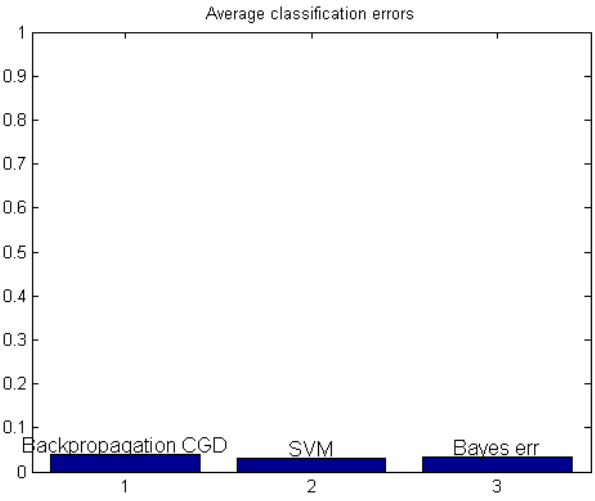

Fig. 14. Support Vector Machines and Neural Network average training error

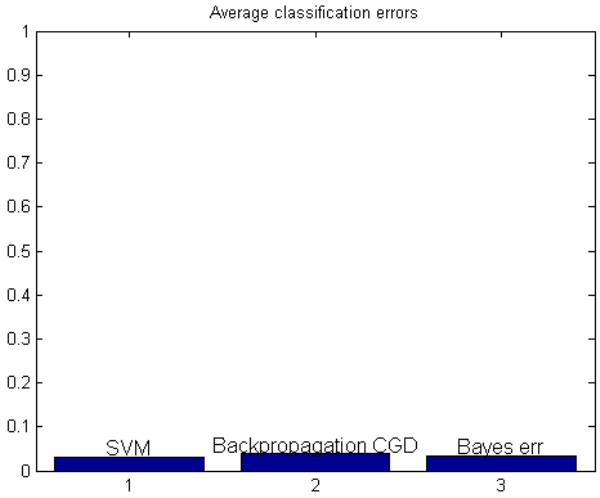

Fig. 15. Support Vector Machines and Neural Network average test error

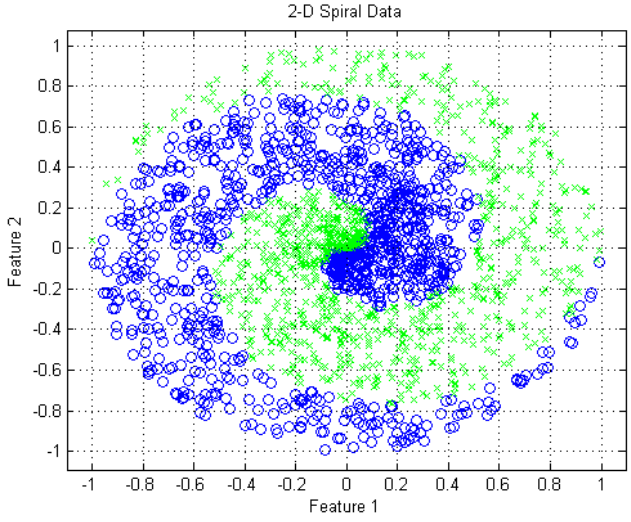

Fig. 16. Two class Spiral Data

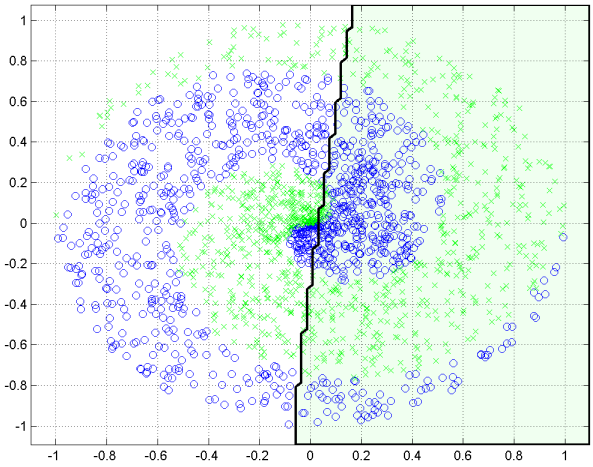

Fig. 17. Neural Network:BP on Spiral Data

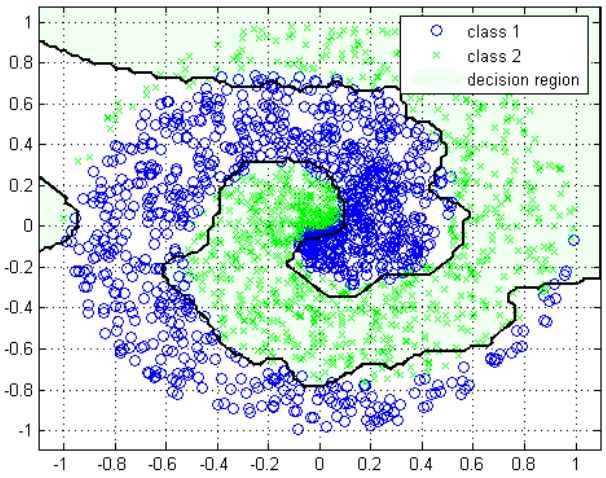

Fig. 18. Support Vector Machine on Spiral Data

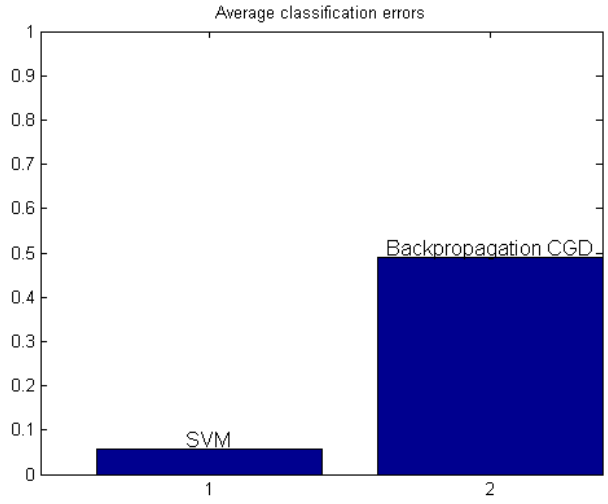

Fig. 19. Average Error Comparison for Support Vector Machine and NN on Spiral Data

4 Spiral Dataset

In this section we consider a dataset set that is very complex for classification models.

4.1 NN-BP error table on spiral data: fig 17

	Class 0	Class 1	Total error
Train Error	0.4245	0.5745	0.4950
Test Error	0.4695	0.5025	0.4863

4.2 SVM error table on spiral data: fig 18

	Class 0	Class 1	Total error
Train Error	0.019	0.057	0.037
Test Error	0.033	0.014	0.087
Support Vectors	400		
Margin	4.4658		

4.3 Summary of Neural Networks and SVMs

The results displayed in figs 8-15 where obtained from classifying the data in fig 7. From fig 14 we notice how the support vector machines constructs a decision

boundary that is very close to the bayes optimal decision surface. The computation of both SVM training and testing errors as shows in table **??** is very close to the bayes optimal boundary. In Figures 16- 19 we observe a dataset in which Neural Networks performs no better that the outcome of tossing a fair coin while the support vector machines does well in separating the two classes. A significant advantage noted on the SVMs is that whilst Backpropagation can suffer from multiple local minima, the solution to an SVM is global and unique. Two more advantages of SVMs are that that have a simple geometric interpretation and give a sparse solution. Unlike NNs, the computational complexity of SVMs does not depend on the dimensionality of the input space. ANNs use empirical risk minimization, whilst SVMs use structural risk minimization. As in all models the structure of the data does play a huge role in the classification error and decision boundary to be constructed by the classifier.

5 Parzen Window Technique

The Parzen technique is considered for a simple case covered in class where **px** is a zero mean, unit variance, unit variance normal density. The window function is described by the equation:

$$\Psi(u) = \frac{1}{2\pi} \exp^{\frac{-u^2}{2}} \tag{43}$$

Letting $h_n = \frac{h_1}{\sqrt{n}}$. Thus $p_n(x)$ is an average of normal densities centered at the samples.

$$p_n(x) = \frac{1}{n} \sum_{i=1}^{n} \frac{1}{h_n} \Psi(\frac{x - x_i}{h_n}) \tag{44}$$

In this experiment the classifier is based on Parzen-window estimation. The densities are estimated for both classes and test points are classified by the class category corresponding to the maximum posterior.

5.1 Parzen error table on spiral data

	Class 0	Class 1	Total error
h = 0.1			
Train Error	0.1571	0.0431	0.0975
Test Error	0.2052	0.0493	0.1281
h = 0.5			
Train Error	0.2476	0.4485	0.3450
Test Error	0.2657	0.4404	0.3538

6 K-Nearest Neighbor Technique

A method to remedy the problem of estimating using the "best" window function is to let the cell volume be a function of the training data, rather than some

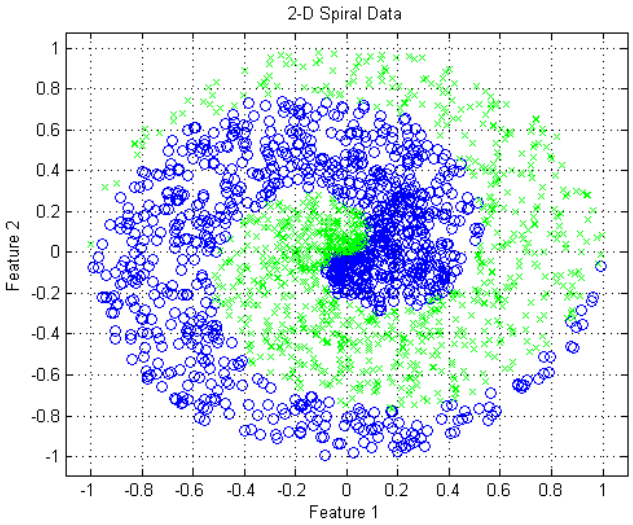

Fig. 20. Spiral 2D Data

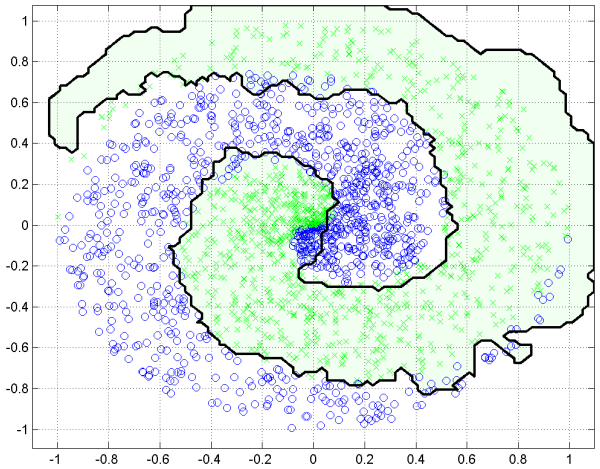

Fig. 21. Parzen on Spiral data with h =0.1. See classification error in table below

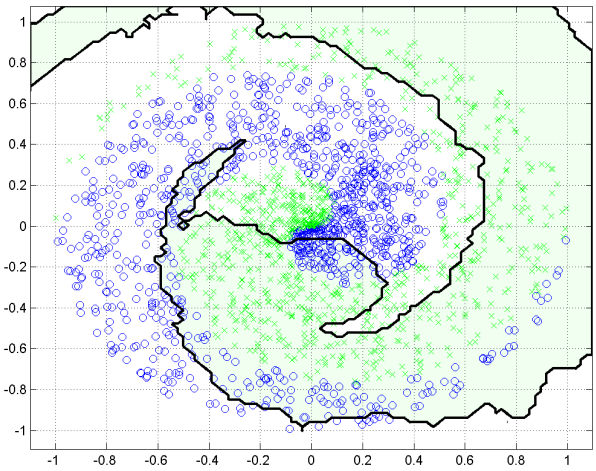

Fig. 22. Parzen on Spiral data with h =0.5

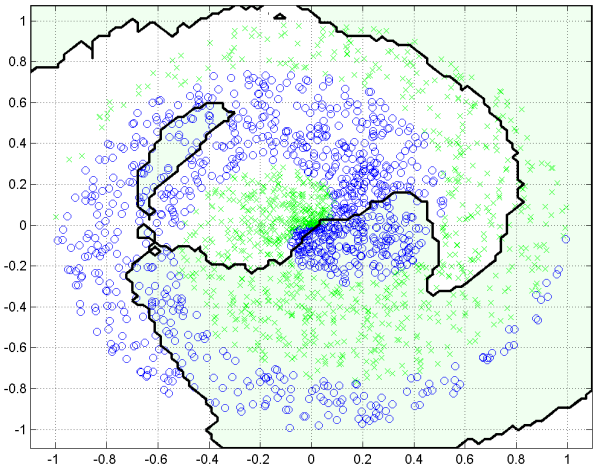

Fig. 23. Parzen on Spiral data with h =0.8

arbitrary function of the overall number of samples. For example, to estimate $p(x)$ from n training samples,we can center a cell about x and let it grow until it captures k_n samples, where k_n is a specified function of n. These samples become the $k - nearestneighbors$ of x. If the density if high near x, the cell will be relatively small, which leads to a good resolution. If the density is too low, the cell will grow large and will eventually stop once it has entered the regions of higher density.

$$p_n(x) = \frac{\frac{k_n}{n}}{V_n} \tag{45}$$

Combining the above with some of theory covered in [3] we discovered that the posterior probabilities can be estimated by:

$$p_n(\omega_i|x) = \frac{p_n(x,\omega_i)}{\sum_{j=1}^{c} p_n(x,\omega_j)} \tag{46}$$

The results that were obtained from using the above methods and the theory covered in class are shown below.

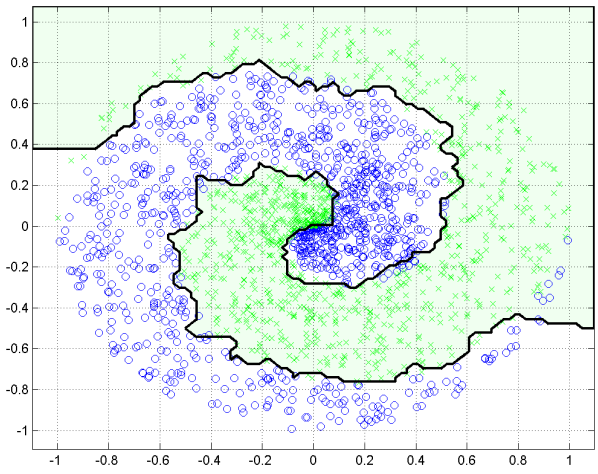

Fig. 24. K-NN on Spiral Data with k=3

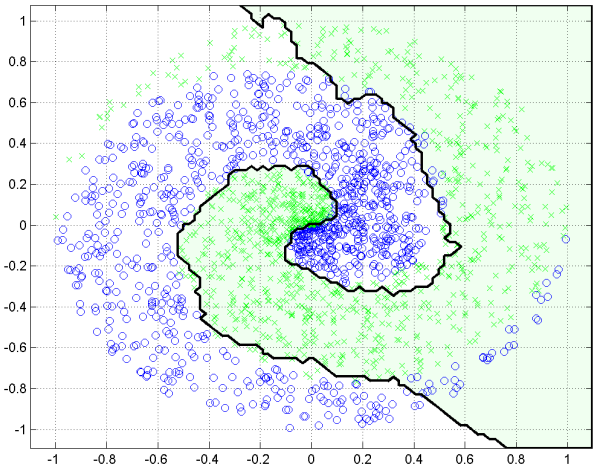

Fig. 25. K-NN on Spiral Data with k=17

6.1 K-NN error table on Spiral data

	Class 0	Class 1	Total error
k = 3			
Train Error	0.0503	0.0199	0.0350
Test Error	0.0911	0.0451	0.0681
k = 5			
Train Error	0.0484	0.028	0.0375
Test Error	0.0823	0.0852	0.0838
k = 17			
Train Error	0.057	0.0918	0.075
Test Error	0.1004	0.1084	0.1044

7 Summary on Parzen and K-Nearest Neighbor Classifiers

The decision regions for Parzen-window classifier (fig21-fig 23) depend upon the number of data points and also the choice of the window function. As can be observed: As the window width gets smaller the classification accuracy of the classifier is increased. This is due to the general characteristic as we discovered in class that if we know the underlying distribution of the data we can use a known window function: in this case a gaussian. The density estimation of this underlying distribution as n gets large does approximates a smooth gaussian estimate with small error (see error table). Assuming that we did not know

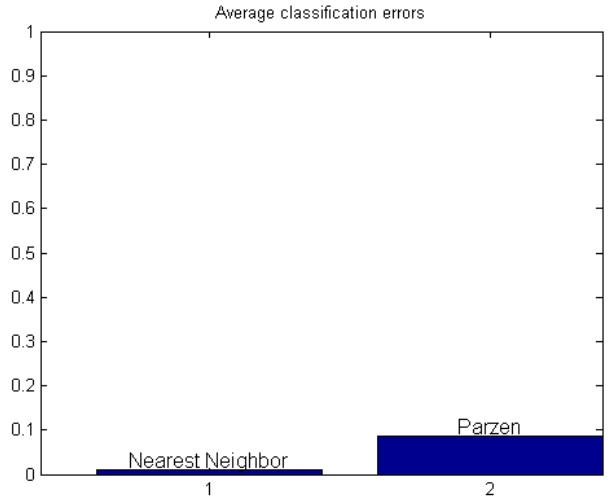

Fig. 26. Average classification error for K-NN vs Parzen Window

that the data was generated from a gaussian density: the Parzen-window has its limitations as was covered in class. In this case the we then let the cell volume be a function of the training data. As explained in section this result in the new estimate (K-NN) based on the number of neighbors surrounding the test point. The results as shown in fig 24 and the $K - NN$ table above does show how superior this method is over the Parzen-window estimation.

8 Conclusion

In this report a design of different classifiers was pursued with the goal of comparing their performance. The results obtained from all experiments indicate that the choice one classifier over the other can not be generalized due to the complexity of either the classifier design or the complexity of the category patterns. However it is realized that if we decide to do a cost effective analysis based on: classification accuracy, design complexity some classifiers can be chosen over the other. On all the data that was used in this experiment the K-Nearest Neighbor is identified to accumulate very low classification error on data that is very complex to separate. The same conclusion is observed from the classification accuracy of the Support Vector Machine (SVM). The SVM is observed to achieve a misclassification error rate that is close to the optimal Bayes error in two dimension. The cost though for choosing the SVM will be the number of support vectors required to correctly separate the categories. The support vectors are observed to increase with the increase in data complexity e.g spiral data set. The classification error rate of the Neural Network, Parzen Classifier are observed to be more than the K-NN and the SVM.

References

1. Bishop, C.: Pattern Recognition and Machine Learning. Springer (2006)
2. Yom-Tov: Computer manual in matlab to accompany pattern classification, 2nd edition (2004) URL: .
3. Hart, D.: Pattern Classification. China Machine Press (2006)
4. Gunn, S.: Support Vector Machines for Classification and Regression. Technical Report: University of Southampton (1998)
5. Franc, V.: Sprtool support vector machine toolbox (2008) URL: `http://cmp.felk.cvut.cz/cmp/software/stprtool/index.html`.