

3-26-2007

Multistage Linear SVM Classification

Qian Xia

Purdue University, qxia@purdue.edu

M. Jameel Shaikh

Purdue University, mshaikh@purdue.edu

Okan Ersoy

ersoy@purdue.edu

Herb Moskowitz

Purdue University, herbert.moskowitz.1@purdue.edu

Follow this and additional works at: <http://docs.lib.purdue.edu/ecetr>

Xia, Qian; Shaikh, M. Jameel; Ersoy, Okan; and Moskowitz, Herb, "Multistage Linear SVM Classification" (2007). *ECE Technical Reports*. Paper 353.

<http://docs.lib.purdue.edu/ecetr/353>

This document has been made available through Purdue e-Pubs, a service of the Purdue University Libraries. Please contact epubs@purdue.edu for additional information.

Multistage Linear SVM Classification

Qian Xia, Jameel Shaikh, Okan Ersoy
School of Electrical and Computer Engineering
Herb Moskowitz
Krannert School of Management

Purdue University
West Lafayette, IN 47907

March 18, 2007

ABSTRACT

Parametric models such as linear models are widely used in constructing discriminant functions. They provide useful, interpretable description of simple structure in data. However, sometimes such simple structure does not extend across an entire data set. A trade-off exists between the ease of interpretation of the model and its predictive power. We propose a multistage linear SVM method to address this problem. For this purpose, we develop a strategy to partition the dataset into a rejected subset and an accepted subset based on the sample-margin induced by a linear SVM model. The samples in the accepted subset are considered as adequately modeled whereas the samples in the rejected subset are not. A second linear SVM model is then trained on the accepted subset, and the rejected subset is forwarded to the next stage for further processing. This procedure proceeds until no further partitioning is necessary or possible. The recursive partitioning of the dataset naturally leads to a multistage classifier. The aggregation of simple models obtained upon termination provides us the power to handle complex data while maintaining the ease of model interpretability. Empirical results with different data sets show that the multistage linear SVM method achieves substantial improvements in training and testing accuracies as compared with a single stage linear SVM model and also circumvents the potential overfitting issue usually faced by more complex nonlinear models.

Key Words:

Support Vector Machines, Multistage Structure, Partitioning of Dataset, Aggregation of Sub-Models

1. Introduction

In classification analysis, we typically have n observations of a vector of k explanatory variables $X = \{x_1, x_2, \dots, x_k\}^T$ on objects belonging to one of J classes and wish to predict their class labels. A “training set” consisting of input vectors and corresponding class labels are assumed available. Based on the information in the training set, a discriminant model is constructed to identify the class corresponding to a particular sample.

Parametric models such as linear models are widely used in constructing the discriminant function. They provide useful, interpretable description of simple structure in data. However, such simple structure rarely extends across an entire data set. One alternative is to use nonlinear models for discriminant analysis. Unfortunately, such complex structures usually lose the power of ease of interpretation. A trade-off exists between the ease of interpretation and the predictive power of the learning algorithm. An alternative to compromise this trade-off is to choose piecewise linear models. In this approach, the entire dataset is partitioned into subsets and separate ‘sub models’ are used for the subsets of the partitions. The support vector machine (SVM) is a machine learning algorithm recently developed by Vapnik [1]. Its idea originates from statistical learning theory, and it can be formulated as a quadratic programming problem with equality and inequality constraints. However, as with many other classical methods, the SVM produces a single model over the entire data space, and therefore also faces the dilemma of model interpretability and the ability to handle complex nonlinear data.

We propose a margin-based multistage framework based on linear SVM models as a new solution. The dataset is divided into a rejected subset and an accepted subset based on sample-margins induced by a linear SVM model per stage. The samples in the accepted region are considered as adequately modeled, whereas the samples in the rejected subset are not. A second SVM model is then trained on the accepted subset and the rejected subset is forwarded to the next stage for further processing. The procedure proceeds in several stages until no further division is necessary or possible. The recursive partitioning of the data space naturally leads to a multistage scheme. The aggregation of linear SVM models at each stage approximates the nonlinear model-structure in the overall data space and provides us the power to handle complex data while maintaining the ease of model interpretability. In other words, the model structure is a chain of successive stages rather

than a binary tree structure. Furthermore, the multistage SVM inherits all the advantages of a regular linear SVM. The merits of the algorithm are demonstrated on a synthetic data and a breast-cancer classification task. The experiments show that the multistage SVM achieves comparable predictive ability with conventional nonlinear SVM using polynomial or Gaussian kernel functions.

The proposed method falls in the category of subspace pattern recognition, which includes the well-known tree classification methods. In some previous work on combining SVM with tree structure [2,3,4], Chi and Ersoy introduced a Linear Support Vector Machine Decision Tree (LSVM-DT) [2]. Bennett and Auslander introduced a support vector decision tree method for customer targeting in the database marketing [3]. Their algorithms primarily follow the structure of classical decision trees. Based on the decision boundary induced by SVM model at each tree node, the node is bisected into two child nodes, and the process proceeds recursively on every child node. The proposed method differs from such previous work in the following ways: first, the partition criterion we proposed is margin-based rather than boundary-based. By using sample-margins we distinguish samples based on the level of confidence in making predictions for them; secondly, the multistage structure we propose is tail-recursive – only the rejected subset is forwarded for further processing. In other words, the model structure is a chain of successive stages rather than a binary tree structure. Furthermore, the multistage SVM inherits all the advantages of a regular linear SVM. The merits of the algorithm are demonstrated on a synthetic dataset and a breast-cancer classification task. The experiments show that the multistage SVM achieves comparable predictive ability as compared to a nonlinear SVM using polynomial kernel function.

The paper is organized as follows: Section II explains the basic SVM machine learning algorithm, and the margin-based criterion for learning. In Section III, we present the new method of multistage SVM classification. In Section IV, we provide the experimental results and discussion, followed by concluding discussion in Section V.

2. The Original SVM Classification

The original SVM classification attempts to find the best separating hyperplane with the largest margin width and the smallest number of training errors, as depicted in Figure 1.

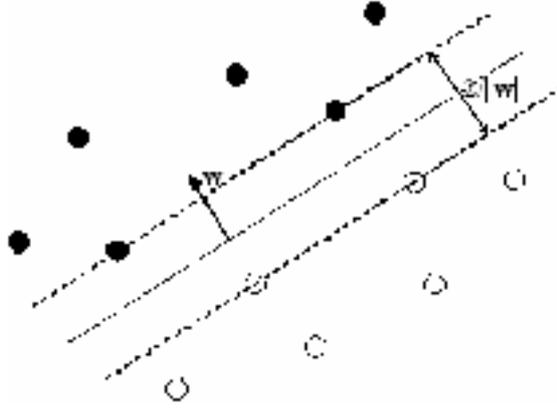


Figure 1. Visualization of margin in a linearly separable problem.

It is easy to imagine that of all the hyperplanes that can separate the training data perfectly, the one that has the maximum margin width has a better chance to give the best generalization performance on future unseen data. In SVM, such a hyperplane with maximum margin is determined by solving a convex quadratic programming problem as shown below:

$$\min \frac{1}{2} \|\mathbf{w}\|^2 + C \sum_i \xi_i \quad (1)$$

subject to the constraints

$$\begin{aligned} y_i (\mathbf{x}_i^T \mathbf{w} + b) &\geq 1 - \xi_i \\ \xi_i &\geq 0 \end{aligned}$$

where \mathbf{x}_i is the i^{th} data vector, y_i is the binary (-1 or 1) class label of the i^{th} data vector, ξ_i is a slack variable, \mathbf{w} is the weight vector normal to the hyperplane, C is the regularization parameter or called the error penalty and b is the bias. It can also be shown that the margin width is equal to $2/|\mathbf{w}|$. Therefore, minimizing the first term in Eq. (1) is equivalent to maximizing the margin width and minimizing the second term is equivalent to minimizing the cost of training errors, Using Lagrangian formulation, the above optimization problem can be solved by minimizing the Lagrangian, which is often known as the primal problem:

$$\min_{\mathbf{w}, b, \xi_i} L_p \equiv \frac{1}{2} \|\mathbf{w}\|^2 + C \sum_i \xi_i - \sum_i \alpha_i (y_i (\mathbf{x}_i^T \mathbf{w} + b) - 1 + \xi_i) - \sum_i \mu_i \xi_i \quad (2)$$

where α_i 's are the Lagrangian multipliers. However, this primal problem may be very difficult to solve, whereas its dual form may be easier to work with:

$$\max_{\alpha} L_D \equiv \sum_i \alpha_i - \frac{1}{2} \sum_{i,j} \alpha_i \alpha_j y_i y_j \mathbf{x}_i^T \mathbf{x}_j \quad (3)$$

subject to the constraints

$$\begin{aligned} 0 &\leq \alpha_i \leq C \\ \sum_i \alpha_i y_i &= 0 \end{aligned}$$

The optimal \mathbf{w} is given by

$$\mathbf{w} = \sum_i \alpha_i y_i \mathbf{x}_i \quad (4)$$

It should be noted that there is an α_i for each training vector \mathbf{x}_i . Some of these α_i 's turn out to be zero. Therefore, by referring to Eq. (4), the optimal \mathbf{w} is represented by only the training vectors \mathbf{x}_i 's whose α_i 's values are non-zero. These training vectors are called the support vectors. The decision function then becomes

$$f(\mathbf{x}) = \mathbf{w}^T \mathbf{x} + b \quad (5)$$

By substituting Eq. (4) into Eq. (5), the decision function can also be written as

$$f(\mathbf{x}) = \sum_i \alpha_i y_i \mathbf{x}_i^T \mathbf{x} + b \quad (6)$$

The classification can be based on the *sign* of Eqs. (5) or (6).

The above dual form corresponds to linear SVM learning. In nonlinear cases, the data vectors \mathbf{x}_i 's are first mapped to a high dimensional Euclidean space by a mapping function ϕ . After mapping, a linear decision hyperplane is constructed in this high dimensional space. Therefore, in nonlinear SVM, the dot products $\mathbf{x}_i^T \mathbf{x}_j$ in Eqs (3) and (6)

are replaced by the dot products in the high dimensional space $\phi(\mathbf{x}_i)^T \phi(\mathbf{x}_j)$. It may seem that the mapping function ϕ , which may be complex and difficult to compute, has to be determined, and its dot product has to be computed every time. Fortunately, $\phi(\mathbf{x}_i)^T \phi(\mathbf{x}_j)$ can be replaced by a kernel function $K(\mathbf{x}_i, \mathbf{x}_j)$, if Mercer's condition is satisfied (Vapnik 1995). It is simply a function of the input vectors \mathbf{x}_i and \mathbf{x}_j , and it can take on a few different forms, such as polynomial and radial basis functions, as shown in Eqs. (7) and (8), respectively.

$$K(\mathbf{x}_i, \mathbf{x}_j) = (\mathbf{x}_i^T \mathbf{x}_j + 1)^d \quad (\text{polynomial}) \quad (7)$$

$$K(\mathbf{x}_i, \mathbf{x}_j) = \exp\left(-\|\mathbf{x}_i - \mathbf{x}_j\|_2^2 / \sigma^2\right) \quad (\text{radial basis}) \quad (8)$$

where d and σ^2 are user defined parameters.

This implicit kernel mapping helps avoid computation in the high dimensional feature space, thus curing the curse of dimensionality. However, because the kernel mapping is carried out implicitly, the model coefficients in the high dimensional feature space can not be calculated. In other words, when implicit kernel mapping is used, the SVM model is not interpretable. In the proposed multistage SVM method, we restrict the algorithm to linear SVM by working in the input space without any mapping. The dual problem used is exactly the same as Eq.(3). The model coefficients can be calculated using Eq.(4).

3. The Multistage SVM Algorithm

The method proposed in this section extends the original SVM method to a multistage structure. During construction of a SVM model, a subset of samples, known as support vectors, is selected automatically and the discriminant hyperplane with maximum margin is generated. The difficult samples (i.e., support vectors within the margin whose α 's are always non-zero) will be close to the discriminant hyperplane and the easy-to-classify ones will be further away from it. Based on this, the multistage SVM method bisects the dataset into accepted/rejected subsets with samples "easy-to-classify" in the accepted subset and samples "difficult-to-classify" in the rejected subset. The rejected subset is then forwarded to the next stage for further processing, and a second SVM model is trained on

the accepted subset. The process resembles a data filtering framework in terms of distinguishing easy and difficult samples and processes them differently. The overall model consists of a chain of successive stages, with two linear SVM models at every stage but the last stage.

In order to present the method further, we will discuss the criterion used for data partitioning at each stage and the rules used to stop the iterative process, followed by illustrations of the overall training and testing procedure.

3.1 Margin-based dataset partitioning

Margin plays a crucial role in modern machine learning research [5]. It measures the confidence of a classifier for its predictions. Sample-margin is defined as the distance between the response of the sample and the discriminant hyperplane induced by the classifier [6,7]. As mentioned earlier, SVM is a margin-based learning algorithm. The sample-margin produced by SVM model measures the confidence of the SVM classifier with respect to its prediction for that sample. In other words, the samples whose responses are closer to the decision boundaries are more likely to be misclassified than those that are further away. If a SVM model has high level of confidence in making prediction for a sample, it is natural to expect that the underlying model structure of that sample is well represented or approximated by that SVM model. Furthermore, the set of samples that are well represented or approximated by the same SVM model should share a similar underlying model-structure. So, using the sample-margin information, we can pick out the subset of samples that are easy to classify in terms of the underlying model-structure and based on this “cleaned” subset, we can train a new SVM model to better capture the common underlying model-structure of the subset.

One natural way to do this is to use the ± 1 SVM margin, and consider the samples that fall within the margin as being close to the boundary and the samples that fall outside the margin as being adequately far away from the boundary. For separable cases, it is the set of support vectors that will fall on the ± 1 margin. For nonseparable cases, all the samples that fall within the margin are not necessarily just support vectors, and it is possible that some support vectors fall outside this border region as well. However, the ± 1 margin can still be a good choice for the following reasons: first, usually only a small

portion of the support vectors will fall outside the margin area; secondly, the second SVM model trained on the “cleaned” dataset is expected to correct some of the misclassified responses produced by the first SVM model at each stage; thirdly, widening the rejection margin tends to reject more samples and, therefore, will produce more stages during training, which can lead to potential over-fitting during testing.

3.2 Termination Rules

As the process proceeds, the set of samples that are easy to classify in the underlying model-structure are picked out, leaving only the difficult to classify subset for further processing. As more stages are created, it will be more and more difficult to generate an effective partitioning because the samples left are inconsistent in nature and act like noise. On the other hand, since the partitioning is tail-recursive, there will be fewer and fewer samples left, which will increase the risk of overfitting for later stages.

Based on the analysis above, we propose two termination rules:

Rule 1: If the number of the accepted samples is too small, then stop.

Since a second SVM model is trained on the accepted subset, it is important that the size of the accepted subset is sufficiently large. For example, the size should not be smaller than the number of explanatory variables in the model.

Rule 2: Terminate at the earliest stage which maximizes the prediction accuracy.

Cross validation can be used to evaluate the prediction accuracy, and to find the number of stages which maximizes the prediction accuracy. If there are no unique maxima, an earlier stage is preferred to later stages.

3.3 Training and testing procedures

The training procedure for the multistage SVM method is described below:

Step 1: Set $k:=1$ for the first stage. Let $S(0)$ represents the entire data set, and $S(k-1)$ represent the subset forwarded from the previous stage, in other words, the rejected subset at the $k-1^{\text{th}}$ stage.

Step 2: Train an initial linear SVM model on the data set $S(k-1)$ for the k th stage. Divide $S(k-1)$ into two parts based on the ± 1 SVM margin. The points falling outside this region are the accepted subset, and the points inside the rejection region are the

rejected subset. Then, a second linear SVM model is trained on the accepted subset and saved as the appropriate model for that stage.

Step3: Check the termination conditions. If either of the two conditions is/are satisfied, then stop; otherwise, let $k:=k+1$, $S(k)$ be the rejected subset of samples and go to step 2.

During testing, we first check to see whether the testing sample is accepted or rejected using the first SVM model at the current stage. If the response value falls inside the rejection region, then the testing sample is rejected; otherwise, it is accepted. The process proceeds, until the testing sample is accepted by a stage (the first stage that accepts it), and the final classification result is calculated by the second SVM model of that stage.

The testing procedure of the multistage SVM algorithm is described below:

Step 1: Set $k:=1$ for the first stage, and let $f_k(x)$ represent the first linear SVM model, and $g_k(x)$ represent the second linear SVM model for the k th stage.

Step 2: Calculate the value of $f_k(x)$ for the testing sample; if the value is outside the rejection region, then interpret $g_k(x)$ as the final classification label and stop. If the value falls within the rejection region, then let $k:=k+1$, and do step 2 again.

The training and testing procedures are visualized in Figure 2.

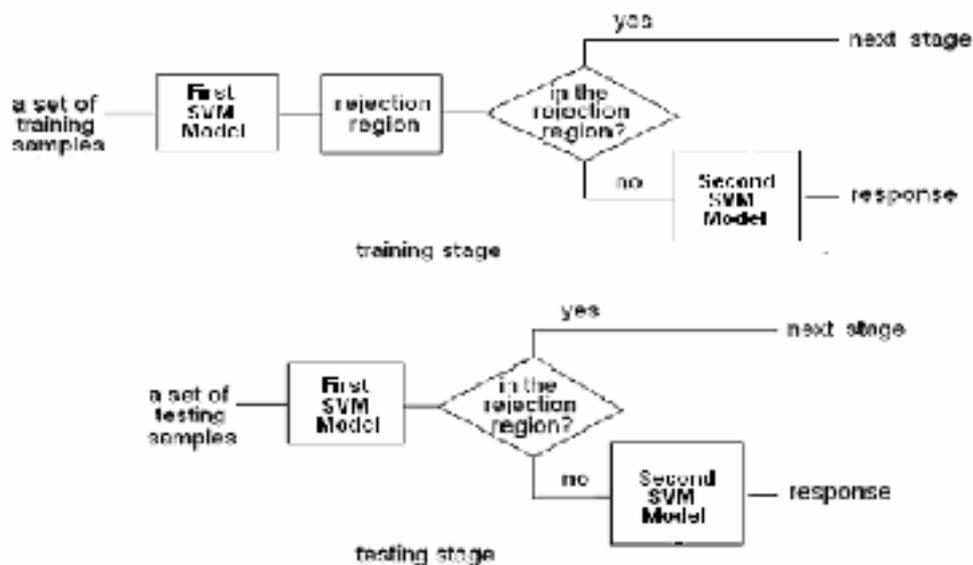


Figure 2. Internal structure of multistage SVM.

4. Experiments and Discussion

The multistage SVM algorithm is designed to approximate the nonlinear model structure by an aggregation of linear submodels. In order to test this, we compare multistage SVM with the original linear SVM model and the 2nd order SVM polynomial model. The behavior of the three methods is compared on a synthetic and a real world classification problem. The synthetic one is a two-dimensional nonlinear classification problem. The real one is a breast cancer task for diagnosing benign and malignant cases.

4.1 The cocentric data set

We designed a simulated data set in the form of cocentric circles shown in Figure 3. The crossed points in the middle correspond to the first class, and the circled points outside correspond to the second class. There are 281 samples in the first class, 448 samples in the second class, and altogether 729 samples in the dataset. Each sample has two explanatory variables x_1, x_2 and one response variable y .

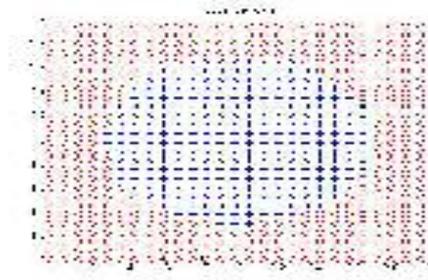


Figure 3. Visualization of the cocentric data set.

The cocentric dataset is a nonlinear classification problem. We randomly select 600 samples as training set, of which there are 228 samples in the first class and 372 samples in the second class. The remaining 129 samples are used as testing set, with 53 samples in the first class and 76 samples in the second class. The experimental results of single stage linear SVM, single stage 2nd order polynomial SVM and multistage SVM are summarized in Table 1.

The single stage linear SVM fails to distinguish the two classes and classifies all the samples as the second class (i.e., all of the samples of the first class are classified as the

Table 1. The classification results with three types of SVM models.

	Training (# = 600)			Testing (# = 129)		
	Mis #	Cor #	Per	Mis #	Cor #	Per
Linear	228	372	62.0%	53	76	58.9%
2 nd order poly	3	597	99.5%	1	128	99.2%
Multistage	120	480	80.0%	30	99	76.7%

second class). Since the true underlying model is 2nd order polynomial, the SVM model using 2nd order polynomial kernel function obtains very good result in both training and testing. The multistage SVM algorithm ends with 4 stages. Although, the performance of the multistage linear SVM is not as good as the 2nd order polynomial SVM, which happens to have the same form as the true underlying model, it improves substantially as compared to the single stage linear SVM. The overall training and testing accuracies are 18% and 17.8% higher than that of the single stage linear SVM, respectively. The detailed results of the multistage SVM are summarized in the table below. The number of samples accepted (total#), number of samples misclassified and correctly classified (mis# and cor#) and the accuracy percentage at each stage are given in Table 2 for both the training and testing results.

Table 2. The classification result obtained at the successive stages of the multistage SVM,.

Stage	Training				Testing			
	Total#	Mis#	Cor#	Per	Total#	Mis#	Cor#	Per
1	276	41	235	85.1%	80	21	59	73.8%
2	160	10	150	93.8%	30	4	26	86.7%
3	58	7	51	88.6%	16	4	12	75.0%
4	106	39	67	63.2%	3	1	2	66.7%
Sum	600	120	480	80%	129	30	99	76.7%

As observed in Table 2, the stagewise training and testing accuracies are quite high, except for the last stage. This result is consistent with our expectation. For previous stages, the prediction was made by the second SVM model which was trained on the “cleaned”

dataset. However, for the last stage, no dataset partitioning is generated, and a single SVM model is trained on all samples available there. Since the samples forwarded to the last stage are noise like in nature, it is natural to observe a substantial decrease in training and testing accuracies for the last stage. In summary, when making predictions, the multistage SVM model has higher level of confidence when the prediction is made by models at previous stages, and a lower level of confidence when the prediction is made by the model at the last stage.

4.2 Experiments with the Real World Data Set

The breast cancer dataset [6] was obtained from the repository of a machine learning database at University of California, Irvine. This dataset has 9 attributes (3 continuous features, 6 nominal features) with 277 instances of which 196 are of benign class and 81 are of malignant class. We randomly select 230 samples as training set, of which 160 samples are benign and 70 samples are malignant, and the rest 47 as testing set, of which 36 samples are benign and 11 samples are malignant.

Table 3 summarizes the overall training and testing result for the breast cancer dataset. The multistage SVM method ends with 3 stages, and it has training and testing accuracy 8.7% and 10.6% higher than that of the single stage linear SVM model. The 2nd order polynomial SVM model has the highest training accuracy, but also the lowest testing accuracy, which indicates over-fitting during training. This result demonstrates that by using the multistage linear SVM model, we obtain a substantial improvement in training and testing accuracy when compared to single stage linear model and also circumvent the risk of overfitting by piecewise linear models as compared to complex nonlinear models.

Table 3. The results with the cancer data.

	Training (# = 230)			Testing (# = 47)		
	# mis	# cor	Per	#mis	#cor	Per
Linear	60	170	73.9%	15	32	68.1%
2 nd order poly	23	207	90.0%	19	28	59.6%
Multistage	44	186	80.9%	10	37	78.7%

The stagewise training and testing accuracies are summarized in Table 4. We observe similar result as with the cocentric dataset - the multistage SVM has higher accuracy in previous stages and lower accuracy in the last stage.

Table 4. The stagewise results with the cancer data.

Stage	Training				Testing			
	Total#	Mis#	Cor#	Per	Total#	Mis#	Cor#	Per
1	107	13	94	87.9%	23	4	19	82.6%
2	56	11	45	80.4%	18	4	14	77.8%
3	67	20	47	70.1%	6	2	4	66.7%
Sum	230	44	186	80.9%	47	10	37	78.7%

Besides percentage accuracy, a task like the breast cancer problem should be analyzed by using the model response when making diagnostic decisions. In order to investigate this, we examined the relationship between odds of prior probability and odds of posterior probability. Let $\{T = t_0\}$ and $\{T = t_1\}$ represent the event that the true result is benign and malignant, respectively and $\{M = m_0\}$ and $\{M = m_1\}$ represents the event that the model response is benign and malignant respectively. From Bayesian theory, we have,

$$\frac{P(T = t_0 | M = m_0)}{P(T = t_1 | M = m_0)} = \lambda * \frac{P(T = t_0)}{P(T = t_1)} \quad \text{where } \lambda = \frac{P(M = m_0 | T = t_0)}{P(M = m_0 | T = t_1)} \quad (9)$$

The ratio on the left is the odds of posterior probability, and the second term on the right is the odds of prior probability. The odds of prior probability represents the confidence in diagnosing using only prior information, and the odds of posterior probability represents the confidence in diagnosing using both prior information and the model response. The likelihood ratios

$$\lambda_0 = \frac{P(M = m_0 | T = t_0)}{P(M = m_0 | T = t_1)} \quad \text{and} \quad \lambda_1 = \frac{P(M = m_1 | T = t_1)}{P(M = m_1 | T = t_0)} \quad (10)$$

represent how much we gain or lose in diagnosing when the model response is benign and malignant, respectively. When the likelihood ratio $\lambda < 1$, the posterior odds is not as good as prior odds in diagnosing; when the likelihood ratio $\lambda > 1$, we gain from using the posterior odds, and the larger the value of λ is, the more benefit we obtain from it. In Table

5, we summarize details of classification results in the tables on the left, and the likelihood ratio λ in the tables on the right.

In order to evaluate the informative value of response of each model, we compare the value of λ . Since we are interested in diagnosing unknown patients, we will mainly concentrate on comparing the values of λ for testing results. As shown in Table 5, for the 2nd order polynomial SVM model, the λ values for the testing result are smaller than 1, which implies that the response of this model is not helpful but may make it even worse for diagnostic practice. For single stage linear SVM model, the λ values for testing are slightly larger than 1 for both benign and malignant cases, which means they are of little help in diagnosing. The method of multistage linear SVM has the largest values of λ for testing result, which implies that this method is most helpful in diagnosing. The multistage model is especially helpful when the response given by the model is malignant.

5. Conclusions and Further Research

The main thrust of this report has been to propose a multistage SVM approach and to discuss some of its features. The multistage SVM algorithm partitions the original data set into an accepted subset and a rejected subset based on margins induced by a linear SVM model. The samples in the accepted subset are known to be easy to classify in nature, and a second linear SVM model is trained on this set to capture their common underlying model structure. The rejected subset is forwarded to the next stage for further processing. The process proceeds until no further partitioning is possible or necessary. The aggregation of a set of linear models obtained upon termination enables us to approximate the non-linear nature of the dataset while maintaining the ease of interpretation by piecewise linear models.

The experimental results indicate that by using an aggregation of simple linear SVM models, the multistage SVM method achieves better overall training and testing result than a single stage linear SVM model, and circumvents the issue of potential overfitting usually faced by nonlinear parametric models. Furthermore, early stages are expected to have higher level of accuracy, and the last stage is expected to give worse result.

Several research directions can be further investigated. One of them is to find a better way to partition the dataset. The way we use now is the ± 1 SVM margin. For linearly

Table 5. Classification accuracies per class and the likelihood ratios.

MULTI STAGE LINEAR

Training				230 total
	Model Response			Summary
True	Y=0	Y=1	Total	Correct
Y=0	141	19	160	88.1%
Y=1	25	45	70	64.3%
Over all	80.9%			
Testing				47 total
	Model Response			Summary
True	Y=0	Y=1	Total	Correct
Y=0	33	3	36	91.7%
Y=1	7	4	11	27.3%
Over all	78.7%			

Training		
λ	Model Response	
True	Y=0	Y=1
Y=0	2.47	
Y=1		5.41
Testing		
λ	Model Response	
True	Y=0	Y=1
Y=0	1.44	
Y=1		4.36

Linear SVM

Training				230 total
	Model Response			Summary
True	Y=0	Y=1	Total	Correct
Y=0	141	19	160	88.13%
Y=1	41	29	70	41.43%
Over all	73.9%			
Testing				47 total
	Model Response			Summary
True	Y=0	Y=1	Total	Correct
Y=0	30	6	36	83.33%
Y=1	9	2	11	18.18%
Over all	68.1%			

Training		
λ	Model Response	
True	Y=0	Y=1
Y=0	1.50	
Y=1		3.49
Testing		
λ	Model Response	
True	Y=0	Y=1
Y=0	1.02	
Y=1		1.09

POLY SVM

Training				230 total
	Model Response			Summary
True	Y=0	Y=1	Total	Correct
Y=0	154	6	160	96.25%
Y=1	17	53	70	75.71%
Over all	90.00%			
Testing				47 total
	Model Response			Summary
True	Y=0	Y=1	Total	Correct
Y=0	26	10	36	72.22%
Y=1	9	2	11	18.18%
Over all	59.57%			

Training		
λ	Model	
True	Y=0	Y=1
Y=0	3.96	
Y=1		16.1
Testing		
λ	Model	
True	Y=0	Y=1
Y=0	0.88	
Y=1		0.63

separable cases, the responses of all the support vectors fall on the ± 1 margins. However, for the more general nonlinearly separable cases, all the instances falling within the border region are not necessarily just the support vectors, and some of the support vectors may fall outside this region as well. In such cases, we will fail to pick out all the difficult vectors (i.e. support vectors) selected by the SVM model. Given a more effective way to pick out the difficult vectors, the subset accepted at early stages will be more consistent in terms of

the underlying model structure, and it is expected that the performance of those early stages can then be further improved.

Another interesting direction is to improve the performance of the last stage. As previously discussed, the model in the last stage is trained on a set of noise like samples, and it is difficult to construct a good parametric model based on them. An alternative way could be to use nonparametric methods such as the K-nearest-neighbor model to handle the problem.

References

- [1]: Vapnik, V. (1995). *The Nature of Statistical Learning Theory*, Springer-Verlag, New York.
- [2]: Chi, H., Ersoy, O., “Support Vector Machine Decision Trees with Rare Event Detection,” *International Journal of Smart Engineering System Design*, pp. 225-242, Vol 4, Oct 2002.
- [3]: Wu, D., Bennet, K. P., Christianini, N., and Shawe-Taylor, J., “Large Margin Trees for Induction and Transduction,” *ICML’99 Proceedings*..
- [4]: Bennet, K. P., and Auslender, L., “On Support Vector Decision Trees for Database Marketing” *R.P.U. Math Report 98-100*, March 1998.
- [5]: Bachrach, R.G., Navot A., Tishby N., (2004) “Margin Based Feature Selection – Theory and Algorithms,”. *Proceedings of the 21th International Conference on Machine Learning*, Banff, Canada..
- [6] Crammer, K., Gilad-Bachrach, R., Navot, A., &Tishby, N. (2002) “Margin Analysis of the LVQ Algorithm,”. *Proc. 17th Conference on Neural Information Processing System.s*.
- [7]: Schapire, R.E., Freund, Y., Bartlett, P., & Lee, W.S. (1998). “Boosting the Margin: A New Explanation for the Effectiveness of Voting Methods,”. *Annals of Statistics*.
- [8]: Shawe-Taylor, J., Bartlett, P., Williamson, R., & Anthony, M. (1998) “Structural Risk Minimization over Data-dependent Hierarchies,”. *IEEETransactions on Information Theory*, 44, pp.1926-1940.
- [9]: Quinlan, J.R. (1990). “Introduction of Decision Trees,”. In J.W. Shavlik and T.G. Dietterich (Eds.), *Readings in Machine Learning*. Morgan Kaufmann. Originally published in *Machine Learning* 1: 81-106 (1986).

[10]: Burges, C. (1998).” A Tutorial on Support Vector Machines for Pattern Recognition,”
Data Mining and Knowledge Discovery, 2(2)..