Setting a Good Example: Improving Generalization Performance in Support Vector Machines through Outlier Exclusion

P. Dwight Kuo and Wolfgang Banzhaf
Department of Computer Science
Memorial University of Newfoundland
St. John's, NL
A1B 3X5
CANADA
Tel: (709) 737–6108
Fax: (709) 737–2009

February 12, 2004

Abstract. We show that the removal of outliers when training support vector machines both improves the generalization performance of the classification system while simultaneously reducing the number of support vectors. To demonstrate this, outliers were chosen to be the samples in the leave–one–out cross–validation methodology which are misclassified by the support vector machine, the minimum intra–class distance, maximum a posteriori and Fisher's linear discriminant classifiers. The classifiers were then retrained and evaluated using the original dataset with the outliers removed. This leads to an increase in generalization performance of the SVM and a reduction in the number of support vectors. The support vector machine algorithm was tested using linear, polynomial and radial basis function kernels with a variety of penalty parameters.

Keywords: support vector machines, kernel methods, outlier detection, generalization

1. Introduction

Support vector machines are a relatively new machine learning method which has begun to be studied extensively due to its robustness to error and its ability to construct non–linear classification and decision boundaries.

However, it has been stated in the literature that this method can be sensitive to overfitting (the generation of highly non–linear decision surfaces) (Christianini and Shawe–Taylor, 2000). Although there have been attempts to generate both separate training algorithms and modifications which mitigate this problem (Henbrich and Weston, 1999; Zhang, 1999; Song et al., 2002), these solutions have not been generated in a systematic fashion and have not been shown to be effective in the general case.

Here, we show that, for a variety of parameters and kernel types, removal of outliers which have not been classified correctly by both statistical methods and support vector machines, and subsequent retraining of the support vector machine leads to increased generalization performance. This increase in performance is greater than that which we would expect for just the deletion of the outlying points. Therefore, we show that the deletion of these points during the training phase of the support vector machine leads to an increase in generalization ability for a wide variety of parameters and kernels as well as a reduction in the number of support vectors.

It has been shown for the maximal margin support vector machine classifier that a reduction in the number of support vectors leads to better generalization performance (Christianini and Shawe–Taylor, 2000).

2. Support Vector Machines

Support vector machines (SVM) are a relatively new machine learning algorithm introduced by Boser, Guyon and Vapnik (Boser et al., 1992). SVMs has been shown to be successful in a variety of different applications such as text categorization, hand–written character recognition, image classification and bio–sequence analysis (Christianini and Shawe–Taylor, 2000; Muller et al., 2001; Guyon et al., 2000).

SVMs emerged from work on the perceptron algorithm. The perceptron algorithm is an algorithm used in machine learning for supervised learning as well as for developing classifiers (Christianini and Shawe–Taylor, 2000). Earlier work on the perceptron learning rule focused on linearly separable classes. The best generalization performance for such systems was obtained when the discriminant hyperplane was as far as possible from both classes of data points. The use of SVMs generalizes this concept further by allowing for non–linear decision surfaces which can perfectly separate non–linearly separable classes of data in the original measured feature space.

Suppose we are given empirical data of the form:

$$(\mathbf{x}_1, y_1), ..., (\mathbf{x}_m, y_m) \in \chi \times \{\pm 1\} \tag{1}$$

where the domain of χ is some nonempty set of which the feature vectors $\mathbf{x}_i$ with the y_i's representing the class assignment being either ± 1 are sampled from. So for any new feature vector $\mathbf{x}$ not in the set of $\mathbf{x}_1 ... \mathbf{x}_m$ we wish to determine its classification y.

To start with, we consider the general class of hyperplane functions which separate linearly separable classes in some feature space of dimension n. Note that this hyperplane can be defined in a dimension greater than that of the original problem:

$$(\mathbf{w} \cdot \mathbf{x}) + b = 0, \ \mathbf{w} \in \mathbb{R}^N, b \in \mathbb{R}. \tag{2}$$

This corresponds to the following decision function:

$$f(\mathbf{x}) \ = \ sgn\left((\mathbf{w} \cdot \mathbf{x}) + b\right), \tag{3}$$

where evaluated samples with positive values belong to the first class while negative valued samples belong to the second class.

The performance of this kind of classifier is based on the fact that, among the set of hyperplanes that separate the data, there exists a unique hyperplane that yields the maximum margin of separation between classes given in Equation 4. Theoretically, a hyperplane with such a maximum margin will have a lower error rate when generalized over samples that have not been used to train the classifier.

$$\max_{\mathbf{w},b} \ \min\{\|\mathbf{x} - \mathbf{x}_i\| : \mathbf{x} \in \mathbb{R}^N, \ (\mathbf{w} \cdot \mathbf{x}) + b = 0, \ i = 1, ..., m\} \tag{4}$$

This problem may then be reformulated into the following optimization problem:

$$\begin{aligned} minimize \quad & \tau(\mathbf{w}) = \frac{1}{2}\|\mathbf{w}\|^2 \\ s.t. \quad & y_i \cdot ((\mathbf{w} \cdot \mathbf{x_i}) + b) \geq 1, \ i = 1, ..., m. \end{aligned} \tag{5}$$

We can then use Lagrange multipliers, $\alpha_i \geq 0$ and a Lagrangian as follows:

$$L(\mathbf{w}, b, \alpha) \; = \; \frac{1}{2}\|\mathbf{w}\|^2 - \sum_{i=1}^{m} \alpha_i (y_i((\mathbf{x}_i \cdot \mathbf{w} + b) - 1) \tag{6}$$

The Lagrangian L must now be minimized with respect to the primal variables $\mathbf{w}$ and b and maximized with respect to the dual variables α_i (i.e. a saddle point must be found). For any of the constraints that are not precisely met as inequalities, (i.e. where $y_i \cdot ((\mathbf{w} \cdot \mathbf{x}_i) + b) - 1 > 0$), the corresponding α_i must be 0. This is the case since L is maximized when these values of $\alpha_i = 0$. This is also known as the Karush–Kuhn–Tucker (KKT) conditions.

By the KKT conditions, the derivatives of L with respect to the primal variables must disappear:

$$\frac{\partial}{\partial b} L(\mathbf{w}, b, \alpha) = 0, \; \frac{\partial}{\partial \mathbf{w}} L(\mathbf{w}, b, \alpha) = 0, \tag{7}$$

which leads to

$$\sum_{i=1}^{m} \alpha_i y_i = 0 \tag{8}$$
$$\mathbf{w} = \sum_{i=1}^{m} \alpha_i y_i \mathbf{x}_i$$

This means that our solution can be expressed as a subset of some of the training patterns which are denoted by the $\alpha_i s$ that are non–zero. It is these non–zero vectors that are referred to as support vectors as they define the optimum marginal hyperplane (Christianini and Shawe–Taylor, 2000).

$$\alpha_i [y_i((\mathbf{x}_i \cdot \mathbf{w} + b) - 1] = 0, \; i = 1, ..., m. \tag{9}$$

Only these support vectors are used to construct the classifier system. The remaining training vectors are therefore irrelevant to the performance of the system. Therefore, we can think of the support vectors as defining the hyperplane. If we substitute Equation 9 into L we arrive at the most common statement of the SVM method as a quadratic programming problem in Equation 10.

$$\begin{aligned} &\mathit{maximize} \quad W(\alpha) = \sum_{i=1}^{m} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{m} \alpha_i \alpha_j y_i y_j (\mathbf{x}_i \cdot \mathbf{x}_j) \\ &\mathit{s.t.} \quad \alpha_i \geq 0, \; i = 1, ..., m, \; \mathit{and} \; \sum_{i=1}^{m} \alpha_i y_i = 0. \end{aligned} \tag{10}$$

As well, substituting Equation 9 into our original decision function Equation 3, allows the hyperplane decision function to be written as:

$$f(\mathbf{x}) = sgn(\sum_{i=1}^{m} y_i \alpha_i (\mathbf{x} \cdot \mathbf{x}_i) + b) \tag{11}$$

From Equation 11, the $\mathbf{x} \cdot \mathbf{x}_i$ dot product may be thought of as a distance measure between the two vectors $\mathbf{x}$ and $\mathbf{x_i}$. In the most general case, this may be replaced by $k(x, x')$ where k is the kernel which operates on the input patterns x and x'. Such a substitution is allowed since we defined the feature vectors in a space which only occurs in dot products (refer to Equation 3).

To treat the non–linear case and to make the above formulation the most general case possible, one additional change is required. In practice, a separating hyperplane between two classes may not exist. In this case, we wish to position the decision hyperplane such that the classification error rate is minimized. We do this by adding the slack variable $\varepsilon_i \geq 0, i = 1, ..., m$. There are many possible formulations for adding the slack variables into our optimization problem, one such objective function is:

$$minimize \quad \tau(\mathbf{w}, \varepsilon) = \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{m} \varepsilon_i \tag{12}$$

for some value of the constant $C > 0$. Again, rewriting the objective in terms of Lagrangian multipliers leads to the addition of the constraint $0 \leq \alpha_i \leq C$, for $i = 1, ..., m$ and $\sum_{i=1}^{m} \alpha_i y_i = 0$ to Equation 10.

A more detailed description of the SVM algorithm may be found in Christianini and Shawe–Taylor (Christianini and Shawe–Taylor, 2000), Műller et al. (Muller et al., 2001), Burges (Burges, 1998) and Cortes and Vapnik (Cortes and Vapnik, 1995).

3. Statistical Classifiers

In order to detect outliers in the data, we choose to use the minimum intra–class distance, maximum a posteriori, and Fisher's linear discriminant classifiers. Points which are misclassified by these statistical classifiers and the support vector machine are considered outliers.

3.1. Minimum Intra–Class Distance Classifier

The minimum intra–class distance (MICD) classifier can be considered to be a variation on the use of the minimum euclidean distance (MED) classifier. The MED is not appropriate for the general classification problem as it is only valid for the special case of uncorrelated, equal variance features. If we assume the mean vector of all samples in class A to be μ_A and the mean vector of all samples in class B to be μ_B, then we can classify any new samples by the Euclidean distance from these means. If the new sample is closer (in the Euclidean distance sense) to the mean of class A, classify it as a member of class A. Similarly, if the sample is closer to the mean of class B, then classify it as class B. This can be expressed mathematically as:

$$\mathbf{x} \in A \;\; iff \;\; (\mathbf{x} - \mu_A)^T(\mathbf{x} - \mu_A) < (\mathbf{x} - \mu_B)^T(\mathbf{x} - \mu_B) \qquad (13)$$

However, as previously stated, this assumes uncorrelated, equal variance features. Of course, this is not true in the general case. Therefore, in order to generalize the MED we require a transformation of the feature space into a space which has uncorrelated equal variance features. We can then use the MED in this new space as opposed to directly in the original feature space.

It can be shown that this transformation may be achieved by using the inverse of the class covariance matrices (S^{-1}) as shown in Equation 14:

$$\mathbf{x} \in A \;\; iff \;\; (\mathbf{x} - \mu_A)^T S_A^{-1}(\mathbf{x} - \mu_A) < (\mathbf{x} - \mu_B)^T S_B^{-1}(\mathbf{x} - \mu_B) \quad (14)$$

with

$$S = \frac{1}{N}\sum_{i=1}^{N}(\mathbf{x}_i - \mu)(\mathbf{x}_i - \mu)^T \qquad (15)$$

3.2. Maximum A Posteriori Classifier

The maximum a posteriori (MAP) classifier is very similar to the previously presented MICD. The MICD uses the covariance matrices of the two classes to increase the accuracy of the classifiers. We can attempt to improve upon this by including the expected class probabilities in our classifier.

$P(A|\mathbf{x})$ and $P(B|\mathbf{x})$ are referred to as the “a posteriori” class probabilities given an input pattern $\mathbf{x}$. This leads to the following classifier which is referred to as the maximum a posteriori probability classifier:

$$\mathbf{x} \in A \;\; iff \;\; P(A|\mathbf{x}) > P(B|\mathbf{x}) \tag{16}$$

Using Bayes Theorem and choosing a class distribution (in this case a multi–variate Gaussian distribution) we can rewrite Equation 16 as:

$$\frac{P(A)\exp\left[-\frac{1}{2}(\mathbf{x}-\mu_A)^T\Sigma_A^{-1}(\mathbf{x}-\mu_A)\right]}{(2\pi)^{\frac{n}{2}}|\Sigma_A^{\frac{1}{2}}|} > \frac{P(B)\exp\left[-\frac{1}{2}(\mathbf{x}-\mu_B)^T\Sigma_B^{-1}(\mathbf{x}-\mu_B)\right]}{(2\pi)^{\frac{n}{2}}|\Sigma_B^{\frac{1}{2}}|} \tag{17}$$

3.3. Fisher's Linear Discriminant

Fisher's Linear Discriminant (FLD) is used to find features which maximize the distance between classes while simultaneously minimizing the intra–class variances (Duda et al., 2000). This is most commonly formulated as follows:

$$J(\mathbf{w}) \;=\; \frac{(m_1 - m_2)^2}{s_1^2 + s_2^2} \tag{18}$$

where J is the function to be maximized, m_1 and m_2 are the class means, and s_1 and s_2 are the class covariances.

The most common feature chosen is the projection of the data onto the eigenvector whose corresponding eigenvalue is largest. This eigenvector corresponds to the direction of maximum variance of the data.

However, in the most general case, such a choice may not be the best one since it is possible to construct two classes where the maximum variance direction does not correspond to a projection where the two classes are linearly separable.

4. Classification Methodology

In order to test the efficacy of the classification schemes, we use the leave–one–out cross–validation classification error. This methodology requires that the data be divided into n folds. The classifier is then trained on a training set of $n-1$ folds and tested on the n^{th} fold. This is repeated for all n combinations of $n-1$ training sets. The performance of the classifier can then be assessed based on the classification error for each test case. The leave–one–out cross–validation methodology is known to give an almost unbiased estimate of the generalization error of the classification accuracy (Gretton, 2003) especially as the value of n approaches the number of samples in the dataset.

In many instances, with a large number of training examples it becomes impractical to use the leave–one–out cross–validation methodology. Often, heuristic methods for calculating the leave–one–out cross–validation method are then implemented (Gretton, 2003). However, in the case of the datasets selected, there are a relatively small number of samples. Therefore, we choose to have as many folds as there are data points in order to obtain a relatively unbiased estimate of the generalization performance of the classification schemes.

5. Data Sets

All input data was obtained from the University of California Irvine Machine Learning Repository. Data was scaled such that it was between 0 and 1. Three test sets were chosen: the Wisconsin Breast Cancer Database (WBCD), the Pima Indians Diabetes Database (PIDD), and the Wisconsin Prognostic Breast Cancer Database (WPBCD).

5.1. Wisconsin Breast Cancer Database

The data in the WBCD was collected by Dr. William H. Wolberg of the University of Wisconsin Hospital, Madison, Wisconsin (Mangasarian and Wolberg, 1990). The database contains data from 699 patients taken from 1989–1991. From each patient, a sample of tumor tissue was extracted. From each patient sample, nine features were recorded which were used to train the classifier: clump thickness, uniformity of cell size, uniformity of cell shape, marginal adhesion, single epithelial cell size, bare nuclei, bland chromatin, normal nucleoli, and mitoses. Each feature was given a subjective integer value of between 1 and 10 by a technician. The samples were then classified as either benign or malignant with the following distribution: 458 cases (65.5%) benign and 241 (34.5%) malignant.

It must also be noted that there are feature entries within the database which have no recorded information. In total there are 16 instances within the database with a single missing attribute value. These missing values were replaced with the mean value for that attribute.

5.2. Pima Indians Diabetes Database

The PIDD was collected by the National Institute of Diabetes and Digestive and Kidney Diseases. Samples were collected from 768 females of 21 years or older of Pima Indian heritage. The classification task is to determine whether the patient is diabetic according to World Health Organization criteria. The 8 numerical feature measurements are: Number of times pregnant, plasma glucose concentration in an oral glucose tolerance test, Diastolic blood pressure (mm Hg), triceps skin fold thickness (mm), 2–hour serum insulin (mu U/ml), body mass index (weight in kg /(height in m)^2), diabetes pedigree function, and age (years).

The class distribution is as follows: 65.10% tested negative for diabetes, 34.90% tested positive for diabetes.

5.3. Wisconsin Prognostic Breast Cancer Database

There are 198 records in the WPBCD. Each record represents the follow–up for one breast cancer case which exhibits invasive breast cancer and no evidence of distant metastases at the time of diagnosis. Each record contains 30 attributes. The classification task is to classify each record as being a case of nonrecurrent or recurrent cancer. The class distribution is as follows: 151 nonrecurrent, 47 recurrent. There were 4 cases of missing values in the database. These values were replaced with the mean value for that given feature.

6. Removal of Outliers

Using the SVM, MICD, MAP, and FLD classifiers, we calculate the leave–one–out cross–validation accuracy for each of the three datasets. By determining which samples were misclassified by all four classification methods we then can determine in a somewhat systematic fashion which samples in each of the datasets can be considered outliers. By deleting these samples from their respective datasets, and reanalyzing the leave–one–out cross–validation accuracy for each classifier we can then assess the effect of the deletion of the outliers on each method.

6.1. Results

The results of the classification of the three datasets both with the original dataset, and the modified one are shown in Tables I to IX. The SVM results are shown for the error penalty parameter, C, equal to 0.1, 1.0, 10.0, 100.0 and 1000.0. Three kernel types were investigated: linear, polynomial (degree 2) and radial basis function (RBF). The RBF kernel was tested with the γ parameter equal to 0.1, 0.25, 0.5, 1.0, 2.0, 5.0. Note that the performance of the statistical classifiers is not shown for the use of the RBF kernel as it is comparable to the previously presented results.

The second line of each row indicates the performance of the classifiers with the outliers removed. The last row of each table shows the average change for runs of the classifiers with and without exclusion of outliers.

As can be seen, in the vast majority of cases the SVM generalization performance improves with the removal of outliers. Only in two cases was there a decrease in performance for the SVM with the removal of outliers. Using a Polynomial kernel of degree 2 with $C = 0.1$ for both the PIDD and WPBCD led to a slight decrease in classification performance. For the PIDD the SVM misclassified an additional three samples with the removal of outliers while use of the SVM with the WPBCD led to an additional misclassification of two samples. As well, the number of support vectors (shown in brackets) which define the SVM classifier was reduced in all cases. The performance of the statistically based classifiers either decreased or remained relatively unaffected.

In the formulation of the SVM we know that samples closest to the decision boundary between classes are used to form the decision function. These points were referred to as the support vectors. Reviewing the support vectors used to form our SVM classifier, it was seen that many of these points were also the outliers previously mentioned. Intuitively, if these points really are outliers, we would prefer to completely ignore them and not use them in the training of our classifier. However, the SVM treats these points as ones from which it can learn a great deal and therefore incorporates them into the decision function with a high weighting. This leads to a skewed decision function which weights the importance of these outliers disproportionately high. This may be thought of as a form of overfitting the dataset when training a classifier.

For all datasets, kernels and parameters tested it was found that all of the sample data points which were considered outliers by the above method were included in the support vectors which were used to build each classifier. Since these points were also points that were misclassified by leave–one–out cross–validation, this would indicate that their removal might increase the generalization performance of the classifier. This was indeed the case.

The traditional statistical methods while still suffering from this drawback are much less sensitive to outliers. Since they pool together statistics for each class as a whole (i.e. the mean and covariance), the effect of a few outliers on the system is negligible. For instance, all of the statistical measures applied use the class means as part of the decision function. A small number of outliers does little to affect the location of the mean since they are in relatively small number as compared to the non–outlier samples. Therefore, in this case we may hypothesize that the statistical approaches will be less sensitive to outliers, but may not be more accurate in generalization ability.

Showing that the removal of specific points from the training set improves the generalization performance of the SVM methodology indicates that methods should be devised for detecting such points. There have been many approaches for outlier detection in the literature (Breunig et al., 2000; Aggarwal and Yu, 2001; Knorr et al., 2000). However, such approaches detect outliers in the sense of distance, density or through projections of the dataset. Thus, such methods do not take into account class membership. In this application outliers may be defined as those points which are most likely to be misclassified by the SVM and are also potentially isolated from other points from the same class in the feature space. As such, current outlier detection algorithms are inadequate for this specific application. The development of such an algorithm when applied to a dataset before SVM training would have the ability to improve the generalization performance of the SVM while reducing the number of support vectors.

7. Conclusions

We have shown that when omitting outliers from the dataset during the training phase of building a support vector machine classifier that the generalization performance increases as shown by an increase in the number of correctly classified samples. Outliers were selected as those points that were misclassified by the SVM, MAP, MICD and FLD classifiers.

The actual SVM classifier produced when omitting the outliers is simpler than one produced by training using the entire dataset as is demonstrated by the reduced number of support vectors. This performance increase was demonstrated for three different kernel types with a variety of parameters. This would indicate that for these datasets the SVM is sensitive to outliers and overfitting.

The results presented here would indicate that the removal of outliers from the dataset proves to be beneficial both from the point of improved classifier performance and a reduction in the number of required support vectors. However, we previously detected outliers by running four different classification schemes on the training data. Clearly, this is not a reasonable solution to the general classification problem. For this reason, we believe that more research is required in order to find appropriate outlier exclusion algorithms which work for the support vector machine learning methodology in the most general case.

Acknowledgements

The SVM algorithm used was provided by Gavin Cawley of the School of Information Systems, University of East Anglia (Cawley, 2000). This code implements the sequential minimal optimization (SMO) training algorithm of Platt (Platt, 1999).

References

Aggarwal C. C. & Yu, P. S. (2001) Outlier Detection for High Dimensional Data. In *Proceedings of ACM SIGMOD Int. Conf. on Management of Data*, 37–46.

Boser, B. E., Guyon, I. & Vapnik, V. N. (1992). A training algorithm for optimal margin classifiers. In *Proceedings of 5th Annual Workshop on Computational Learning Theory*, Vol. 5, 144–152.

Breunig, M. M., Kriegel, H. P., Ng, R. T., & Sander, J. (2000). LOF: identifying density-based local outliers. In *Proceedings of ACM SIGMOD Int. Conf. on Management of Data*, 93–104.

Burges, C. J. C. (1998). A Tutorial on Support Vector Machines for Pattern Recognition. *Knowledge Discovery and Data Mining.* 2:2, 121–167.

Cawley, G. C. (2000). MATLAB Support Vector Machine Toolbox (v.0.55β). http://theoval.sys.uea.ac.uk/.

Christianini, N., & Shawe–Taylor, J. (2000). *An Introduction to Support Vector Machines (and other kernel–based learning methods).* Cambridge University Press.

Cortes, C. & Vapnik, V. N. (1995). Support Vector Networks. *Machine Learning.* 20:3, 273–297.

Duda, R. O., & Hart, P. E. (2000). *Pattern Recognition and Scene Analysis, 2nd Edition.* Wiley.

Gretton, A. (2003). Kernel Methods for Classification and Signal Separation. *PhD thesis, University of Cambridge.*

Guyon, I., Weston, J., Barnhill, S., & Vapnik, V. (2000). Gene selection for cancer classification using support vector machines. *Machine Learning*, 46:1–3, 389–422.

Henbrich, R. & Weston, J. (1999). Adaptive margin support vector machines for classification. In *Proceedings of ICANN*, Vol. 2, 880–885.

Knorr, E. M., Ng, R. T., & Tucakov, V. (2000). Distance-Based Outliers: Algorithms and Applications. *VLDB Journal: Very Large Data Bases*, 8:3–4, 237–253.

Mangasarian, O. L. & Wolberg, W. H. (1990). Cancer diagnosis via linear programming. In *SIAM News*, 23:5, 1–18.

Műller K. R., Mika, S., Rấtsch, G., Tsuda, K., & Schőlkopf, B. (2001). An introduction to kernel–based learning algorithms. *IEEE Transactions on Neural Networks*, 12:2, 181–202.

Platt, J. C. (1999). Fast training of support vector machines using sequential minimal optimization. In B. Scholkopf, C. Burges and A. J. Smola, editors, *Advances in Kernel Methods – Support Vector Learning*, MIT Press, 1999.

Song, Q., Hu, W., & Xie, W. (2002). Robust Support Vector Machine with Bullet Hole Image Classification. *IEEE Transactions on Systems, Man and Cybernetics – Part C: Applications and Reviews*, 32:4, 440–448.

Zhang, X. G. (1999) Using class–center vectors to build support vector machines. *IEEE Proc. Neural Net. Signal Processing IX.*

Table I. Classifier performance on the WBCD using the original and modified datasets with the SVM using a linear kernel. Numbers in brackets refer to the number of support vectors.

	SVM	*MICD*	*MAP*	*FLD*
C=0.1	667/699(112) 669/688(94)	629/699 614/688	665/699 662/688	671/699 666/688
C=1.0	636/699(69) 647/688(42)	629/699 614/688	665/699 662/688	671/699 666/688
C=10.0	643/699(56) 663/688(25)	629/699 614/688	665/699 662/688	671/699 666/688
C=100.0	645/699(54) 665/688(23)	629/699 614/688	665/699 662/688	671/699 666/688
C=1000.0	645/699(54) 667/688(21)	629/699 614/688	665/699 662/688	671/699 666/688
Avg Δ	+15.0(-28.0)	-15.2	+2.2	-5.0

Table II. Classifier performance on the PIDD using the original and modified datasets with the SVM using a linear kernel. Numbers in brackets refer to the number of support vectors.

	SVM	*MICD*	*MAP*	*FLD*
C=0.1	505/768(537) 518/722(484)	513/768 520/722	568/768 574/722	519/768 518/722
C=1.0	330/768(442) 376/703(330)	513/768 513/703	568/768 571/703	519/768 524/703
C=10.0	363/768(406) 424/704(281)	513/768 514/704	568/768 572/704	519/758 524/704
C=100.0	367/768(401) 436/704(268)	513/758 514/704	568/758 572/704	519/758 524/704
C=1000.0	401/758(367) 437/704(267)	513/758 514/704	568/758 572/704	519/758 524/704
Avg Δ	+47.0(-111.4)	+2.0	+4.2	+3.8

Table III. Classifier performance on the WPBCD using the original and modified datasets with the SVM using a linear kernel. Numbers in brackets refer to the number of support vectors.

	SVM	*MICD*	*MAP*	*FLD*
C=0.1	151/198(99) 151/181(64)	151/198 151/181	153/198 154/181	122/198 125/181
C=1.0	104/198(101) 151/181(65)	151/198 151/181	153/198 154/181	122/198 125/181
C=10.0	103/198(95) 125/181(56)	151/198 151/181	153/198 154/181	122/198 125/181
C=100.0	117/198(82) 135/181(46)	151/198 151/181	153/198 154/181	122/198 125/181
C=1000.0	121/198(77) 143/181(38)	151/198 151/181	153/198 154/181	122/198 125/181
Avg Δ	+21.8(-37.0)	+0.0	+1.0	+3.0

Table IV. Classifier performance on the WBCD using the original and modified datasets with the SVM using a polynomial kernel of degree 2. Numbers in brackets refer to the number of support vectors.

	SVM	*MICD*	*MAP*	*FLD*
C=0.1	673/699(75) 676/688(52)	629/699 614/688	665/699 662/688	671/699 666/688
C=1.0	646/699(61) 665/688(31)	629/699 614/688	665/699 662/688	671/699 666/688
C=10.0	639/699(62) 665/688(24)	629/699 614/688	665/699 662/688	671/699 666/688
C=100.0	635/699(65) 666/689(24)	629/699 616/689	665/699 616/689	671/699 666/689
C=1000.0	644/699(55) 666/689(23)	629/699 616/689	665/699 662/689	671/699 666/689
Avg Δ	+20.2(-32.8)	-14.2	-12.2	-5.0

Table V. Classifier performance on the PIDD using the original and modified datasets with the SVM using a polynomial kernel of degree 2. Numbers in brackets refer to the number of support vectors.

	SVM	*MICD*	*MAP*	*FLD*
C=0.1	560/768(460) 557/714(361)	513/768 516/714	568/768 574/714	519/768 523/714
C=1.0	366/768(409) 424/705(284)	513/768 514/705	568/768 575/705	519/768 524/705
C=10.0	374/768(397) 443/704(262)	513/768 512/704	568/768 572/704	519/758 523/704
C=100.0	389/768(379) 457/702(245)	513/758 512/702	568/758 572/702	519/758 524/702
C=1000.0	395/758(373) 468/702(233)	513/758 512/702	568/758 572/702	519/758 524/702
Avg Δ	+53.0(-126.6)	+0.2	+5.0	+4.6

Table VI. Classifier performance on the WPBCD using the original and modified datasets with the SVM using a polynomial kernel of degree 2. Numbers in brackets refer to the number of support vectors.

	SVM	*MICD*	*MAP*	*FLD*
C=0.1	160/198(99) 158/181(64)	151/198 151/181	153/198 154/181	122/198 125/181
C=1.0	121/198(93) 152/181(55)	151/198 151/181	153/198 154/181	122/198 125/181
C=10.0	117/198(86) 137/181(51)	151/198 151/181	153/198 154/181	122/198 125/181
C=100.0	121/198(80) 136/183(49)	151/198 151/183	153/198 151/183	122/198 124/183
C=1000.0	116/198(83) 136/183(52)	151/198 151/183	153/198 151/183	122/198 125/183
Avg Δ	+16.8(-34.0)	+0.0	-0.2	+2.8

Table VII. Classifier performance on the WBCD using the original and modified datasets with the SVM using a radial basis function kernel with varying parameter γ and varying penalty parameter C. Numbers in brackets refer to the number of support vectors.

	γ					
	0.1	*0.25*	*0.5*	*1.0*	*2.0*	*5.0*
C=0.1	660/699(236) 660/688(222)	668/699(168) 669/688(149)	668/699(136) 670/688(117)	675/699(125) 674/688(105)	672/699(156) 672/688(130)	663/699(272) 664/688(247)
C=1.0	610/699(96) 665/688(31)	624/699(80) 637/688(58)	633/699(73) 645/688(49)	640/699(92) 651/688(51)	643/699(120) 655/688(85)	649/699(214) 656/688(188)
C=10.0	636/699(63) 652/688(36)	640/699(61) 657/688(32)	635/699(69) 660/688(31)	632/699(71) 658/688(41)	632/699(87) 660/688(69)	630/699(183) 656/689(177)
C=100.0	638/699(61) 662/688(26)	635/699(64) 662/688(26)	632/699(68) 664/689(26)	633/699(68) 658/689(37)	634/699(80) 660/689(64)	630/699(183) 656/689(177)
C=1000.0	636/699(63) 666/688(23)	635/699(64) 666/689(23)	638/699(61) 661/689(31)	637/699(65) 656/690(41)	634/699(80) 660/689(64)	630/699(183) 656/689(177)
Avg Δ	+25.0(-36.2)	+17.8(-29.8)	+18.8(-30.6)	+16.0(-29.2)	+18.4(-22.2)	+17.2(-13.8)

Table VIII. Classifier Performance on the PIDD using the original and modified datasets with the SVM using a radial basis function kernel with varying parameter γ and varying penalty parameter C. Numbers in brackets refer to the number of support vectors.

	γ					
	0.1	*0.25*	*0.5*	*1.0*	*2.0*	*5.0*
C=0.1	500/768(537)	500/768(538)	492/768(538)	532/768(534)	542/768(520)	543/768(524)
	500/745(491)	500/745(492)	512/730(492)	533/711(454)	545/710(430)	552/715(443)
C=1.0	257/768(516)	301/768(471)	327/768(446)	346/768(433)	361/768(424)	375/768(436)
	281/702(425)	339/702(367)	372/702(333)	395/702(314)	415/703(309)	422/705(317)
C=10.0	345/768(423)	355/768(413)	363/768(408)	370/768(399)	381/768(391)	366/768(410)
	395/703(309)	414/702(291)	428/703(278)	432/704(274)	444/705(261)	444/707(273)
C=100.0	368/768(400)	376/768(392)	379/768(389)	387/768(384)	384/768(385)	389/768(384)
	432/703(271)	440/704(264)	451/704(253)	458/706(248)	459/708(249)	456/709(258)
C=1000.0	480/758(388)	388/758(380)	395/758(373)	388/758(381)	406/758(363)	423/758(348)
	447/703(256)	460/704(244)	466/706(240)	464/710(246)	474/712(238)	475/717(244)
Avg Δ	+19.0(-102.4)	+46.6(-107.2)	+54.6(-111.6)	+51.8(-119)	+52.6(-119.2)	+50.6(-113.4)

Table IX. Classifier Performance on the WPBCD using the original and modified datasets with the SVM using a radial basis function kernel with varying parameter γ and varying penalty parameter C. Numbers in brackets refer to the number of support vectors.

	γ					
	0.1	*0.25*	*0.5*	*1.0*	*2.0*	*5.0*
C=0.1	151/198(103) 151/181(69)	151/198(105) 151/181(68)	151/198(109) 151/181(70)	151/198(114) 151/181(73)	151/198(139) 151/181(93)	151/198(171) 151/181(139)
C=1.0	103/198(101) 121/181(68)	105/198(105) 121/181(68)	105/198(109) 120/181(70)	108/198(114) 122/181(73)	111/198(137) 128/181(93)	128/198(173) 137/181(139)
C=10.0	97/198(102) 115/181(69)	99/198(101) 122/181(60)	93/198(113) 121/181(62)	90/198(124) 113/181(74)	83/198(135) 114/181(93)	89/198(175) 128/181(140)
C=100.0	102/198(96) 128/181(53)	97/198(101) 124/181(57)	95/198(105) 119/181(64)	90/198(114) 151/181(71)	88/198(139) 113/181(93)	89/198(175) 128/181(140)
C=1000.0	110/198(88) 128/181(53)	105/198(94) 122/183(62)	103/198(96) 114/183(72)	90/198(113) 115/181(71)	88/198(139) 113/181(93)	89/198(175) 113/181(140)
Avg Δ	+16.0(-35.6)	+16.6(-38.2)	+15.6(-39.0)	+24.6(-43.4)	+19.6(-44.8)	+25.2(-34.2)