An Analysis on Better Testing than Training Performances on the Iris Dataset

Marten Schutten and Marco A. Wiering

Institute of Artificial Intelligence and Cognitive Engineering
University of Groningen, The Netherlands

Abstract

The Iris dataset is a well known dataset containing information on three different types of Iris flowers. A typical and popular method for solving classification problems on datasets such as the Iris set is the support vector machine (SVM). In order to do so the dataset is separated in a set used for training and a set used for testing. The error rate, after training, for the training set should be lower than the error rate on the test set. However, in this paper we show that when solving the classification problem for the Iris dataset with SVMs this is not the case. Therefore, we provide an analysis of the Iris dataset and the classification models in order to find the origin of this interesting observation.

1 Introduction

The Iris dataset [7] is a well known dataset used for classifying different types of Iris flowers (the Iris Setosa, Iris Versicolor and Iris Virginica). A version of this dataset can be found in the UCI repository [1], with two slight deviations from the original set [2]. Support Vector Machines (SVMs) [5, 11] are a well known and popular method to solve classification problems such as to be solved for the Iris dataset. Methods such as the Support Vector Machine aim to find a hyperplane that separates observations from different classes from each other. This hyperplane is learned from observations from which it is known to which class they belong, and should then generalize to instances that have not been observed before. One of the risks in learning this hyperplane on the basis of a limited number of examples is the problem of over fitting: The hyperplane is constructed specifically to separate the specific observations that are in the training set, even the ones that would be more likely to belong to a different class, when the class were to be unknown. As a result the hyperplane generalizes very poorly and is unable to correctly classify instances in the test set that have not been observed before. The result is then a very high classification accuracy on the train set and a much lower accuracy on the test set.

In this paper, we show that when using SVMs to solve the classification problem for the Iris dataset, a very different problem occurs: rather than over fitting on the known data, the hyperplane is 'under fitted' and generalizes better to unseen instances than it is able to classify the known instances, which is unknown to happen for any classification problem until now. The aim of this paper is to give an analysis of the Iris dataset and the obtained classification models in order to provide insights into this problem.

The paper is organized as follows. First a short explanation of support vector machines is given, followed by the methods through which the Iris dataset is analyzed. Finally, the dataset and classification models are analyzed and discussed.

2 Support Vector Machines

First, a short introduction to support vector machines [5, 11] will be given. Support vector machines can be used both for regression and classification problems, however, due to the nature of the Iris dataset, the explanation will be limited to their use for classification problems. For the explanation, initially a linear support vector machine will be considered for the separation of two classes. For a more thorough explanation of SVMs, see Burges [3], on which this explanation is based.

Consider a dataset $\mathcal{D}$ containing N instances of one of two classes, where the i-th (with $1 \leq i \leq N$) instance of $\mathcal{D}$ is given by $\{\mathbf{x}_i, y_i\}$, where $\mathbf{x}_i \in \mathbb{R}^d$ and the target class $y_i \in \{1, -1\}$, and where d is the number of features that describe each instance in the dataset. The goal is to construct a hyperplane of the form given by the weight vector $\mathbf{w} \in \mathbb{R}^d$ and offset (or bias) $b \in \mathbb{R}$, such that for each instance i in the dataset it holds that $\mathbf{w} \cdot \mathbf{x}_i + b \geq 1$ for $y_i = 1$, and $\mathbf{w} \cdot \mathbf{x}_i + b \leq -1$ for $y_i = -1$. Given that this holds for all points in $\mathcal{D}$, these formulae can be combined to the constraint given by equation 1.

$$y_i(\mathbf{w} \cdot \mathbf{x}_i + b) - 1 \geq 0 \quad \forall i \tag{1}$$

Now the points that lie closest to the separating hyperplane, are the points given either by $\mathbf{w} \cdot \mathbf{x}_i + b = 1$ (if $y_i = 1$) or $\mathbf{w} \cdot \mathbf{x}_i + b = -1$ (if $y_i = -1$). The points that satisfy these constraints are called the *support vectors*. Note that the support vectors for both classes all lie at the same distance from the separating hyperplane, which is given by $\frac{1}{\|\mathbf{w}\|}$. Let the margin of the separating hyperplane be defined as the sum of the distances to support vectors of both classes, being $\frac{2}{\|\mathbf{w}\|}$. Now the goal is to maximize this margin, by minimizing $\|\mathbf{w}\|^2$, subject to the constraint given by equation 1. Thus the goal is to find a separating hyperplane that maximizes the distance between itself and the support vectors of both classes.

Unfortunately, it is not always the case that two classes are perfectly separable, and no solution can be found according to these specific constraints. In order to deal with this non-separability it is required to relax the constraints of the classification problem. This is done with the introduction of a positive slack variable ξ_i for all examples in $\mathcal{D}$. With the introduction of this slack variable the constraints are now given by:

$$\mathbf{w} \cdot \mathbf{x}_i + b \geq 1 - \xi_i \quad \text{if } y_i = 1 \tag{2}$$

$$\mathbf{w} \cdot \mathbf{x}_i + b \leq -1 + \xi_i \quad \text{if } y_i = -1 \tag{3}$$

$$\xi_i \geq 0 \quad \forall i \tag{4}$$

Note that with these constraints an instance is falsely classified whenever $\xi_i > 1$. However, since miss classifications are undesirable, the objective function that is to be minimized is rewritten from $\|\mathbf{w}\|^2$ to the formula given in equation 5. In this formula C defines the cost parameter that determines the severity of errors. Note that minimizing $\frac{1}{2}\|\mathbf{w}\|^2$ yields the same weight vector as minimizing $\|\mathbf{w}\|^2$, but $\frac{1}{2}$ is added to simplify future equations.

$$\frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{N}\xi_i \tag{5}$$

In order to solve this problem it can be rewritten in the form of a Lagrangian formulation, introducing a Lagrange multiplier α_i for each of the i constraints provided by equations 2 and 3. The problem can then be defined according to a Wolfe dual formulation, defining a primal and a dual Lagrangian formulation, given respectively by formulas 6 and 7. In the former formula μ_i represent the Lagrange multipliers that are required to enforce positivity of ξ_i, in accordance with constraint 4.

$$L_P \equiv \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_{i=1}^{N}\xi_i - \sum_{i=1}^{N}\alpha_i\{y_i(\mathbf{w} \cdot \mathbf{x}_i + b) - 1 + \xi_i\} + \sum_{i=1}^{N}\mu_i\xi_i \tag{6}$$

$$L_D \equiv \sum_{i=1}^{N}\alpha_i - \frac{1}{2}\sum_{i,j=1}^{N}\alpha_i\alpha_j y_i y_j \mathbf{x}_i \cdot \mathbf{x}_j \tag{7}$$

The solution can then be found by either maximizing L_D or by minimizing L_P, which yield the same solutions. The constraints of both problems are given in Table 1 and the solution is given by formula 8. Using this weight vector a classification output can be given for any unseen example $\mathbf{x}$ by using equation 9.

$$\mathbf{w} = \sum_{i=1}^{N}\alpha_i y_i \mathbf{x_i} \tag{8}$$

Table 1: Constraints for the primal and dual Lagrangian formulations L_P and L_D.

L_P	L_D
$\frac{\delta L_P}{\delta \mathbf{w}} = \frac{\delta L_P}{\delta b} = \frac{\delta L_P}{\delta \xi_i} = 0$	$0 \leq \alpha_i \leq C$
$\xi_i, \alpha_i, \mu_i \geq 0$	$\sum_{i=1}^{N} \alpha_i y_i = 0$
$\alpha_i\{y_i(\mathbf{w} \cdot \mathbf{x}_i + b) - 1 + \xi_i\} = 0$	
$\mu_i \xi_i = 0$	

$$f(\mathbf{x}) = sign(\sum_{i=1}^{N} \alpha_i y_i \mathbf{x_i} \cdot \mathbf{x} + b) \tag{9}$$

Now this solution holds for linear separating hyperplanes. However, often non-linear hyperplanes are better suited to solve the classification problem. These non-linear hyperplanes can be described using so-called kernel functions. Rather than using the linear form of $\mathbf{x_i} \cdot \mathbf{x}$ that can be found in the solution given by equation 9, the solution can then be found using an alternate form defined by the kernel function $K(\mathbf{x_i}, \mathbf{x_j})$, as given by equation 10.

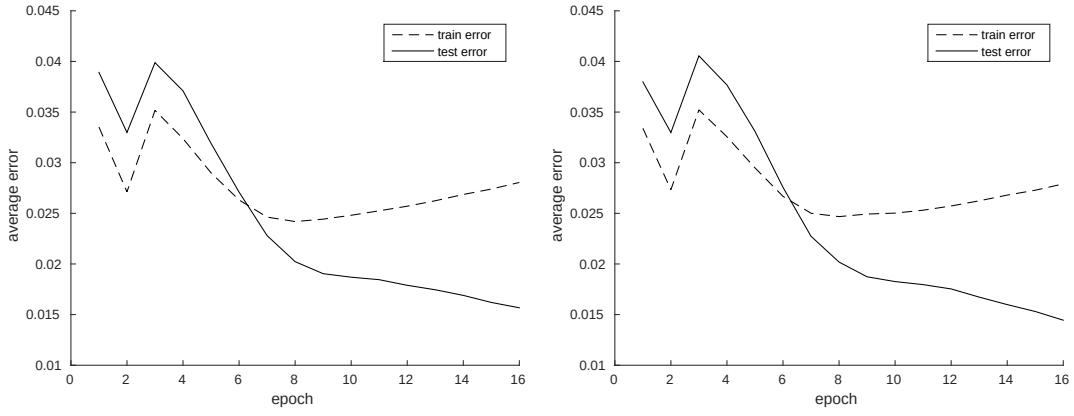
$$f(\mathbf{x}) = sign(\sum_{i=1}^{N} \alpha_i y_i K(\mathbf{x_i}, \mathbf{x}) + b) \tag{10}$$

A well-known kernel function is the radial basis function (RBF), which computes similarities between two examples according to equation 11. In this paper, a truncated version of this kernel was used, so that whenever the output of the kernel function was below a certain threshold θ, it was set to 0. The value of θ was optimized in the same manner as the other parameters for the SVM, as discussed in the following section.

$$K(\mathbf{x_i}, \mathbf{x}) = e^{-\|\mathbf{x_i} - \mathbf{x}\|^2 / 2\sigma^2} \tag{11}$$

3 Methods

In Figure 1, we show the development of the error rate over the 16 epochs for which the SVM was trained using both a regular RBF kernel (Figure 1a) and a truncated RBF kernel (Figure 1b). The error rate is the average error rate over 10000 runs that were performed.



(a) regular RBF kernel

(b) truncated RBF kernel

Figure 1: Learning curves for both the training and the testing set as developed over the 16 epochs for which the SVM was trained. The error rates are averaged over 10000 runs.

The error rate is slightly lower for the truncated RBF kernel than for the regular RBF kernel (0.0144 versus 0.0157 resp. after 16 epochs for the testing set and 0.0279 and 0.0281 resp. after 16 epochs for the training set). Interestingly, and in contrast to expectations, the error rate for the training set starts increasing after 7 epochs for both kernels, while the error on the test set keeps decreasing.

In order to examine how the Iris dataset is classified, the first two principal components were determined using Principal Component Analysis (PCA) [10, 8]. These principal components were used to get an insightful look at the separability of the data. Furthermore, each of the individual features, laid out against each other is examined as well. Since the overall training error is higher than the testing error for the Iris dataset, there must be a number of points in the dataset for which the same holds. In order to identify these points the data were being trained and tested upon using the SVM with the truncated RBF kernel for 10000 times, in which the data were randomly separated in a training set (90%) and a testing set (10%). Particle Swarm Optimization (PSO) [9] was used to find the optimal parameters for the SVM. As SVM, we used the gradient descent SVM explained in [4].

During the runs with the support vector machines a number of things were kept track of for each data point: (1) the number of times the point was added to the testing set; (2) the number of errors that was made when the point was added to the training set and (3) the number of errors that was made when the point was added to the testing set. From this, the error rates can be computed both for when the data point was added to training set and when it was added to the testing set. In order to determine whether the error rates in the testing set were significantly higher than in the training set, Fisher's exact test [6] was used with $\alpha = 0.01$. Finally records were kept for the α-coefficients for each point.

4 Description and Analysis of the Data

The dataset contains 3 classes of types of Iris flowers: the Iris Setosa, Iris Versicolor and the Iris Virginica. Each class has 50 instances in the set of which one is linearly separable from the other classes, while the other classes are not linearly separable from each other. Each instance in the dataset has the following attributes: 1) sepal length in cm; 2) sepal width in cm; 3) petal length in cm; 4) petal width in cm. There are no instances in the dataset that miss any of these attributes. Figure 5 shows how the data is scattered for all permutations of the different features.

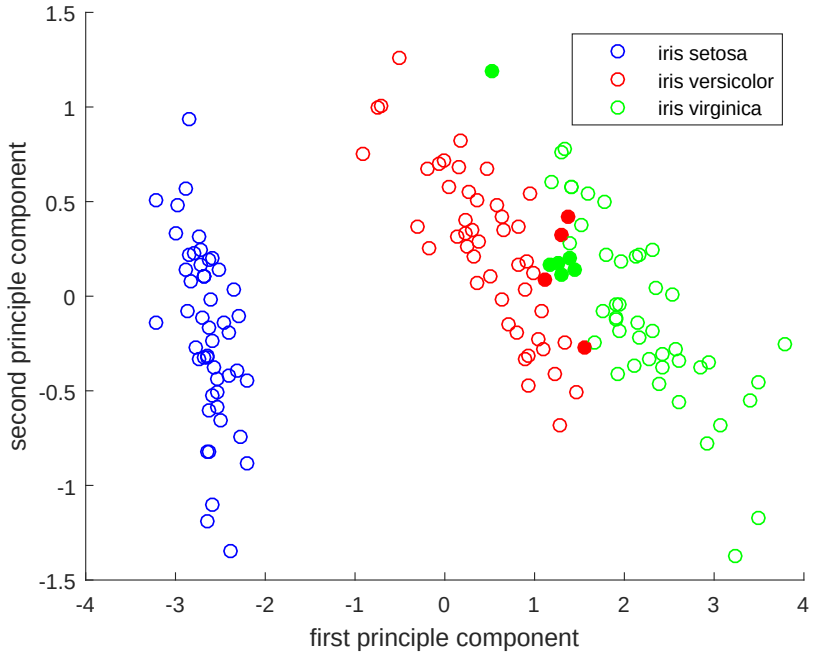


Figure 2: The examples plotted using the first two principal components of the Iris dataset. The filled points represent the examples for which the error rate is significantly higher when the example is used in the training set than in the testing set.

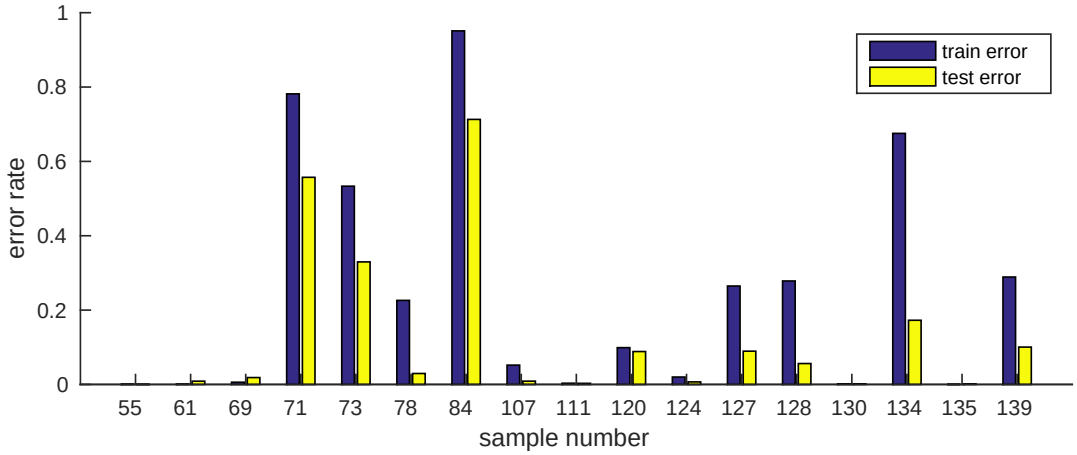


Figure 3: The error rates for all points for which at least one error was made in either the training set or the testing set.

#	true class	error in training set	error in testing set	p	#	true class	error in training set	error in testing set	p
55	ver.	0.000	0.000	1.000	120	vir.	0.099	0.089	0.332
71	ver.	0.782	0.557	$< 2.2e-16$	*124*	vir.	0.020	0.007	0.002
73	ver.	0.533	0.330	$< 2.2e-16$	*127*	vir.	0.265	0.090	$< 2.2e-16$
78	ver.	0.226	0.029	$< 2.2e-16$	*128*	vir.	0.278	0.056	$< 2.2e-16$
84	ver.	0.951	0.713	$< 2.2e-16$	130	vir.	0.001	0.001	1.000
107	vir.	0.052	0.011	$7.939e-13$	*134*	vir.	0.675	0.173	$< 2.2e-16$
111	vir.	0.003	0.003	1.000	*139*	vir.	0.289	0.101	$< 2.2e-16$

Table 2: Error rates for all examples in which the error rate in the training set is higher than the error rate in the testing set, along with the probability that this error rate is due to non-random chances. The points for which this difference is significant (for $p < 0.01$) are emphasized. The true classes were abbreviated to ver. (for the Iris Versicolor) and vir. (for the Iris Virginica). No instances of the Iris Setosa class need to be examined.

The Iris dataset was first published by Fisher [7]. Over time different versions of the dataset have been published [2] with slight alterations. For this paper the version that can be found in the UCI repository [1] was used. This version contains two deviations from the original dataset [2] (in the 35th and 38th example), which were corrected.

Figure 2 shows the data plotted using the first two principal components that can be obtained for this dataset. It can be seen that the Iris Setosa class lies far apart from the other two classes and is easily separable. The other two classes lie more closely to each other and even though most examples should be easy to separate, around the boundary area it might be hard to differentiate between the two of them. Note that a similar observation can be made when looking at the different features plotted together in Figure 5.

The results of the 10000 runs performed with the SVM can be seen in Figure 3. This plot shows the error rate for all points in both the training and testing sets. Note that only points are included in which at least one error is made in either the training or testing set. The first 50 examples in the dataset belong to the Iris Setosa class and as expected no errors are made when classifying instances of it. For each of the points that have a higher error rate in the training set than in the testing set, Table 2 shows the error rates for both sets and the results of the Fisher test (with $\alpha = 0.01$). It can be seen that for four of the examples (55, 111, 120 and 130) the difference is not significant while in all other points the error rate in the training set is significantly higher than in the testing set. Figure 3 together with Table 2 show that only for three examples (61, 69, and 135), the error is higher when the example is used in the test set compared to being part of the training set.

Both in Figure 5 and 2 the examples that score better in the testing set than in the training set are

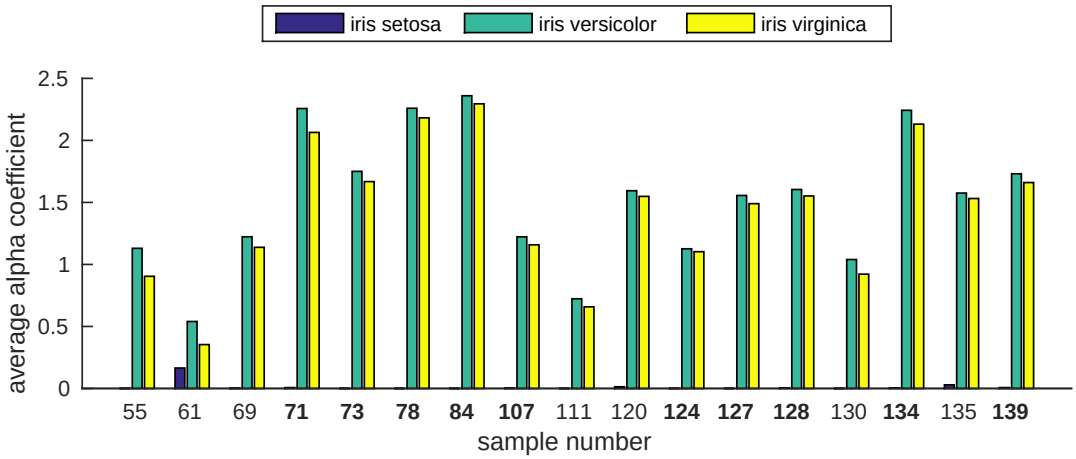


Figure 4: α-coefficient for all points in which an error was made in either the training or testing set. Points printed bold are points in which the error rate was higher in the training set than in the test set.

highlighted. Especially in the plot containing the principal components (Figure 2) it can be seen that all of these 10 points lie around the boundary between the Iris Versicolor and Iris Virginica class. A similar observation can be made in Figure 5, albeit not as clear for all feature combinations.

Finally, Figure 4 shows the α-coefficients for all points in which an error was made in either the training or the testing set. The points that are printed bold are the points in which the error rate was higher in the training set than in the testing set. This figure shows that the alpha coefficients for one class (Iris Versicolor) are generally higher than those for Iris Virginica, which indicates a preference of the model to classify examples as being part of the former class. Due to other examples and the kernel function that uses distances to all examples, this does not mean that these examples are all classified as Iris Versicolor.

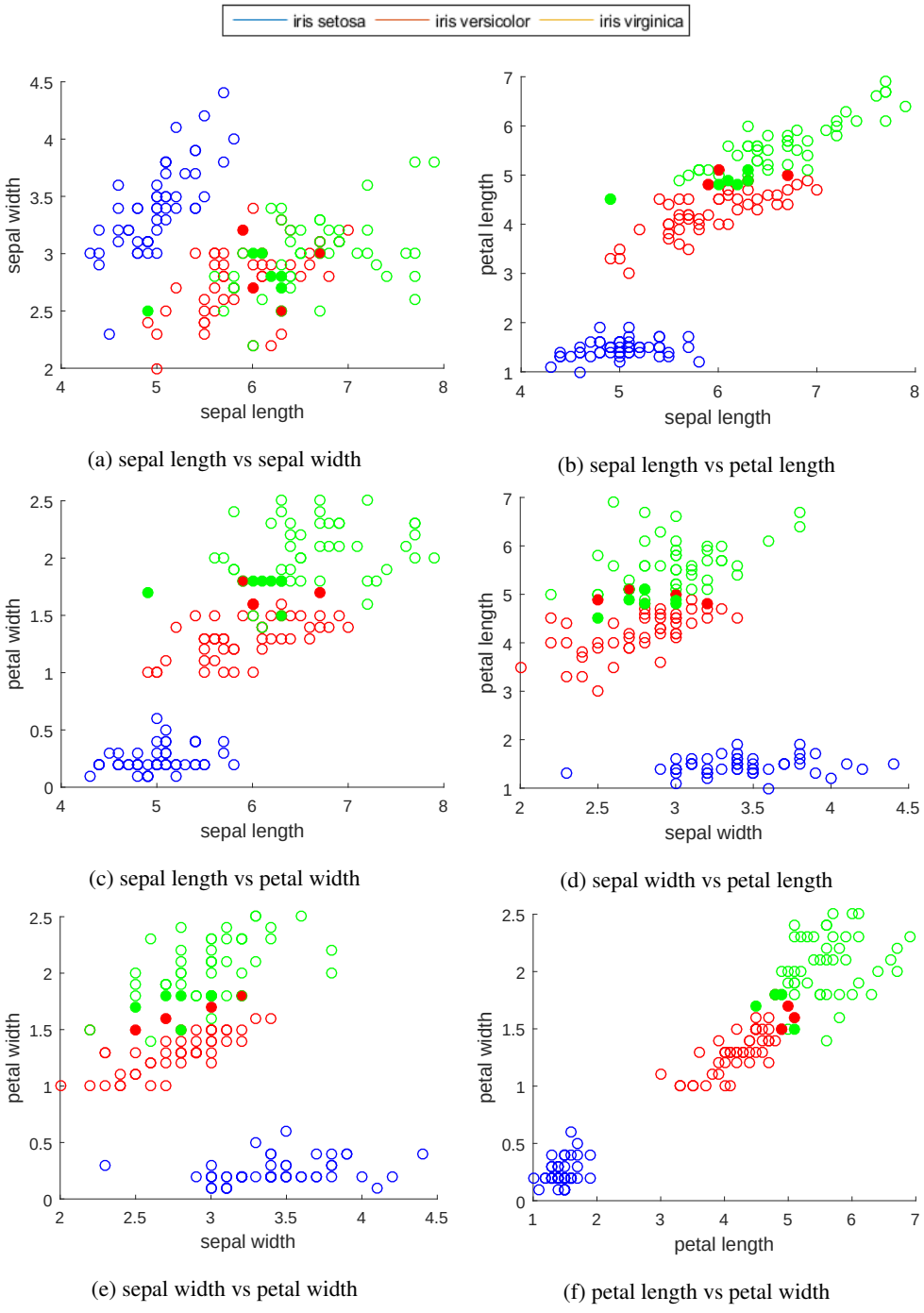
5 Discussion

The aim of this paper is to identify and clarify the unusual behavior in the Iris dataset with regards to the error rates in the training and error phases. Insight has been given how the SVM classifies different data points, and crucial data points that are hard to classify correctly in both the training and testing phase have been identified. We showed that particular data points close to the decision boundary are more often classified correctly when they are in the test dataset than in the train dataset.

One of the ways this behaviour can be explained is through the use of PSO as an optimization method. With PSO a wide variety of parameter settings is tried to find the setting that optimizes the results for their respective test sets using many different random cross validation runs. This means that parameter settings might be chosen that neglect performance in the training phase, in order to be able to generalize better to unseen cases in the testing phase. Because the Iris dataset is very small, meta-parameters were found by PSO that are useful to miss classify training points in order to obtain better results on the test data. Somehow, the extensive parameter search with PSO has led to overfitting on the cross validation results.

This is an important finding, and shows the importance of using a separate unpolluted dataset for the final test. However, the Iris dataset is much too small to be split into a train-validation-test set. Therefore, this option is not available for all datasets (also some datasets with for example fMRI scans are very small and do not contain more than 150 examples).

We found that models can be trained that sacrifice train performance in order to obtain better results on test examples. Still, it is complex to understand how this is exactly done in the training process, as the conventional way of overfitting is exactly the other way around. Therefore, this effect can not be fully accounted for by the use of extensive parameter tuning with PSO and cross validation. Furthermore, no other cases of this phenomenon have been reported to the best of our knowledge. One of the main reasons for this effect to occur in the Iris dataset and not in other datasets, is probably the small amount



(a) sepal length vs sepal width

(b) sepal length vs petal length

(c) sepal length vs petal width

(d) sepal width vs petal length

(e) sepal width vs petal width

(f) petal length vs petal width

Figure 5: all different permutations of the four features in the Iris dataset plotted against each other. The filled points represent the examples for which the error rate is significantly higher when the example is used in the training set than in the testing set.

of examples as well as the limited number of features for each example. However, we would gladly be pointed to other datasets and/or cases where the testing results outperform the training results.

References

[1] K. Bache and M. Lichman. UCI machine learning repository, 2013.

[2] James C Bezdek, James M Keller, Raghu Krishnapuram, Ludmila I Kuncheva, and Nikhil R Pal. Will the real Iris data please stand up? *IEEE Transactions on Fuzzy Systems*, 7(3):368–369, 1999.

[3] Christopher J.C. Burges. A tutorial on support vector machines for pattern recognition. *Data Mining and Knowledge Discovery*, 2:121–167, 1998.

[4] V. Codreanu, B. Dröge, D. Williams, B. Yasar, P. Yang, B. Liu, F. Dong, O. Surinta, L.R.B. Schomaker, J.B.T.M. Roerdink, and M.A. Wiering. Evaluating automatically parallelized versions of the support vector machine. *Concurrency and Computation, Practice and Experience*, 28(7):2274–2294, 2016.

[5] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine Learning*, 20:273–297, 1995.

[6] Ronald A Fisher. On the interpretation of χ 2 from contingency tables, and the calculation of p. *Journal of the Royal Statistical Society*, 85(1):87–94, 1922.

[7] Ronald A Fisher. The use of multiple measurements in taxonomic problems. *Annals of eugenics*, 7(2):179–188, 1936.

[8] Ian Jolliffe. *Principal component analysis*. Wiley Online Library, 2002.

[9] J. Kennedy and R. Eberhart. Particle swarm optimization. In *Neural Networks, Proceedings of the IEEE International Conference on*, volume 4, pages 1942–1948, 1995.

[10] Karl Pearson. On lines and planes of closest fit to systems of points in space. *Philosophical Magazine Series 6*, 2(11):559–572, 1901.

[11] Vladimir Vapnik. *The nature of statistical learning theory*. Springer Science & Business Media, 2013.