Evaluation of Import Vector Machines for Classifying Hyperspectral Data

Ribana Roscher, Björn Waske, Wolfgang Förstner

Zusammenfassung

We evaluate the performance of Import Vector Machines (IVM), a sparse Kernel Logistic Regression approach, for the classification of hyperspectral data. The IVM classifier is applied on two different data sets, using different number of training samples. The performance of IVM to Support Vector Machines (SVM) is compared in terms of accuracy and sparsity. Moreover, the impact of the training sample set on the accuracy and stability of IVM was investigated. The results underline that the IVM perform similar when compared to the popular SVM in terms of accuracy. Moreover, the number of import vectors from the IVM is significantly lower when compared to the number of support vectors from the SVM. Thus, the classification process of the IVM is faster. These findings are independent from the study site, the number of training samples and specific classes. Consequently, the proposed IVM approach is a promising classification method for hyperspectral imagery.

1 Introduction

Hyperspectral imaging, also known as imaging spectroscopy is used since more than two decades for monitoring the Earth [12]. The spectrally continuous data range from visible to the short-wave infrared region of the electromagnetic spectrum and thus, enables a detailed separation of similar surface materials. Therefore hyperspectral imagery is used for classification problems that require a precise differentiation in spectral feature space, e. g., for mapping geological units [7, 34], classifying urban structures [2, 31], and in context of forest and agricultural applications [19, 29]. Hyperspectral applications become even more attractive, regarding the increased availability of hyperspectral imagery through future space-borne missions, such as the German EnMAP (Environmental Mapping and Analysis Program) [13] and the Italian PRISMA (Hyperspectral Precursor of the Application Mission).

Nevertheless, the special properties of hyperspectral imagery demand more sophisticated image (pre)processing and analysis [25, 24]. Conventional

methods, such as the maximum likelihood classifier, can be limited when applied to hyperspectral imagery, due to the high-dimensional feature space and a finite number of training samples. Consequently, the classification accuracy often decreases, with an increasing number of bands (i. e., the well-known Hughes phenomena). Thus, usually alternative classifier methods are applied on hyperspectral imagery, such as spectral angle mapper, neural networks, multiple classifier systems and Support Vector Machines (SVM). However, among the various developments in the field of pattern recognition, SVM are perhaps the most popular approach in recent hyperspectral applications [24]. SVM can outperform other methods in terms of the classifcation accuracy [23, 33] and still exhibit further modification and improvement, e. g., in context of modifying the kernel functions [3] and semi-supervised learning [9, 20].

SVM discriminates two classes by fitting an optimal separating hyperplane to the training samples of two classes within the multidimensional feature space. The approach aims to maximize the margin between the hyperplane and the closest training samples, the so-called support vectors [32]. In linear non-separable cases, the data are transformed by a kernel function into a higher-dimensional feature space. The newly distributed samples enable the fitting of a linear hyperplane, which appears non-linear in the original feature space. As a matter of fact SVM can describe complex classes with multi-modal distributions in the feature space. Moreover, they seem adequate when classifying high dimensional data with small training sets [23].

Contrary to classifiers that directly provide a class label (e. g., decision trees) or a probability measurement (e. g., Gaussian maximum likelihood classifier) for each pixel in the input image, the primary output image of SVM contain the distance of each pixel to the hyperplane of the binary classification problem. This information is used to determine the final class membership. Whereas other classifiers can directly solve multi-class problem, the binary nature of SVM requires an adequate multi-class strategy. Although different multi-class strategies exist [17], most applications use the one-against-one strategy, which divide a K-class problem into K(K-1)/2 binary classification problems. However, probabilities could be of interest and discriminative probabilistic methods like Relevance Vector Machines (RVM) [30]) were applied to hyperspectral [10] and multispectral data [11]. The results show that the RVM-based classification accuracies on multispectral imagery [11] and in part on hyperspectral data [10] are insignificantly lower than the accuracy achieved by SVM. Moreover, RVM are sparser when compared to SVM (i. e., many of the parameters are zero) and thus, enable a faster testing/classification process. The samples assigned to nonzero parameters are used as relevance vectors and support vectors, respectively. However, due to

the computational complexity during the optimization process, the training time of RVM is longer [10].

Logistic Regression is an alternative, probabilistic discriminative classification model that was used, for example, in context of classification and feature selection of hyperspectral imagery [8, 36]. The approach can be extended to Kernel Logistic Regression, e. g.,[16, 14, 5, 28] and further to the concept of Import Vector Machines (IVM) [37]. Like the well-known Gaussian maximum likelihood classifier and contrary to SVM, for example, IVM are a multi-class concept that directly provides probability outputs. Furthermore it is a discriminative model like the SVM, which often shows superior performance over generative models like Gaussian Maximum Likelihood. In addition, Zhu and Hastie [37] already shows that the IVM are much sparser than the SVM and therefore much faster during the classification process. Behind these facts, IVM seem interesting to evaluate the potential in context of classifying hyperspectral data. Our main objectives are:

- to evaluate the performance of IVM compared to SVM in terms of accuracy;
- to evaluate the performance of IVM compared to SVM in terms of sparsity; and
- to investigate the impact of the training sample set on the accuracy and stability of IVM in terms of the classification result.

To investigating these objectives, our study is aiming on the classification of two different hyperspectral data sets, i. e., an urban area from the city of Pavia, Italy, and agricultural areas from Indiana, USA, using two classifiers (i. e., IVM and SVM). In addition, the size of the training sets is varied, to investigate the possible impact of the number of training samples (i) on the classification accuracy and stability of the classifier and (ii) on the computational complexity.

Our paper is organized as follows. Section 2 introduces the Logistic Regression and Kernel Logistic Regression, which is the basis algorithm of the IVM. Moreover, the concept of SVM is briefly introduced. Section 3 explains the conceptual and algorithmic framework of IVM. The experimental setup and results are presented in Section 4 and discussed in Section 5. We conclude in Section 6.

2 Background

In this section we first introduce the Logistic Regression, the basis of the IVM model. Starting from the Logistic Regression, we incorporate kernels and sparsity and end up with a Sparse Kernel Logistic Regression model in Section 3, called IVM. We also briefly introduce the Support Vector Machine model.

2.1 Logistic Regression

We assume to have a training set $(\boldsymbol{x}_n, y_n)$, $n = 1, \ldots, N$ of N labeled samples with vectors $\boldsymbol{x}_n$ of observations and class labels $y_n \in \mathcal{C} = \{1, \ldots, K\}$. The observations are collected in a matrix $\mathsf{X} = [\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N]$.

In the two-class case the posterior probability p_n of a feature vector $\boldsymbol{x}_n$ is assumed to follow the Logistic Regression model

$$p(y_n|\mathbf{x}_n; \mathbf{w}) = \frac{1}{1 + \exp(-\mathbf{w}^\mathsf{T}\mathbf{x}_n)} \tag{1}$$

with the extended feature vector $\mathbf{x}_n^\mathsf{T} = [1, \boldsymbol{x}_n^\mathsf{T}] \in \mathbb{R}^M$ and the extended parameters $\mathbf{w}^\mathsf{T} = [w_{k0}, \boldsymbol{\omega}^\mathsf{T}] \in \mathbb{R}^M$ containing the bias w_{k0} and the weight vector $\boldsymbol{\omega}$.

The objective function $\mathcal{Q}_0(\mathbf{w})$ of the standard logistic regression model is given by the negative log-likelihood function

$$\mathcal{Q}_0(\mathbf{w}) = -\frac{1}{N} \sum_N \left[t_n \log p_n + (1 - t_n) \log (1 - p_n) \right]. \tag{2}$$

The binary target vector $t \in \{0, 1\}$ of length N codes the labels with $t_n = 0$ for $\mathcal{C}_1$ and $t_n = 1$ for $\mathcal{C}_2$.

The Newton-Raphson iteration scheme for the minimization of (2) is given by

$$\mathbf{w}^{(i)} = \mathbf{w}^{(i-1)} - \mathsf{H}^{-1} \nabla \boldsymbol{E} \tag{3}$$

with the gradient $\nabla \boldsymbol{E} = \mathsf{X}(\boldsymbol{p} - \boldsymbol{t})$ and the Hessian $\mathsf{H} = \mathsf{X}^\mathsf{T}\mathsf{R}\mathsf{X}$. The $(N \times N)$-dimensional diagonal matrix R has the elements $r_{nn} = p_n(1 - p_n)$, which can be obtained from (1).

We can reformulate the Newton-Raphson iteration scheme in (3) and obtain the iterated re-weighted least squares solution

$$\mathbf{w}^{(i)} = \left(\frac{1}{N} \mathsf{X}^\mathsf{T}\mathsf{R}\mathsf{X} + \lambda \mathsf{I} \right)^{-1} \mathsf{X}^\mathsf{T}\mathsf{R}\boldsymbol{z}, \tag{4}$$

$$\boldsymbol{z} = \frac{1}{N} \left(\mathsf{X}\mathbf{w}^{(i-1)} + \mathsf{R}^{-1}(\boldsymbol{p} - \boldsymbol{t}) \right) \tag{5}$$

to obtain adjusted parameters in each iteration i. We also introduce a regularization parameter λ to prevent overfitting, especially in the case of separable or nearly separable data.

2.2 Kernel Logistic Regression

To use the linear classifier for solving a non-linear problem we introduce kernels to map the original observations from the input space into a higher-dimensional kernel space.

We introduce kernels and transform the features $\mathbf{\Phi} \in \mathbb{R}^M$ to a higher dimensional feature space $\mathcal{F}$ making use of the kernel trick [1]. The kernel function K is given by

$$K\left(\mathbf{x}_n, \mathbf{x}_m\right) = \boldsymbol{\phi}^{\mathsf{T}}\left(\mathbf{x}_n\right) \boldsymbol{\phi}\left(\mathbf{x}_m\right). \tag{6}$$

Following the Representer Theorem the parameters W lie within the span of the feature vectors $\mathbf{\Phi}$:

$$W = \sum_n \alpha_n \boldsymbol{\phi}_n = \mathbf{\Phi}^{\mathsf{T}} \boldsymbol{\alpha}. \tag{7}$$

The vector $\boldsymbol{\alpha}$ contains the parameters which define the linear decision boundaries in kernel space.

With (6) and (7), (4) and (5) become

$$\boldsymbol{\alpha}^{(i)} = \left(\frac{1}{N} K^{\mathsf{T}} R K + \lambda K\right)^{-1} K^{\mathsf{T}} R \boldsymbol{z}, \tag{8}$$

$$\boldsymbol{z} = \frac{1}{N}\left(K \boldsymbol{\alpha}^{(i-1)} + R^{-1}\left(\boldsymbol{p} - \boldsymbol{t}\right)\right). \tag{9}$$

2.3 Support Vector Machines

The concept of the Support Vector Machine model is similar to that of the Kernel Logistic Regression. The algorithm finds an optimal nonlinear decision boundary in the original feature space by estimating a hyperplane in the transformed feature space. The objective function, which we minimize is given by

$$Q_{SVM} = \frac{1}{N} \sum_N \left[1 - y_n f(\boldsymbol{x}_n)\right]_+ + \frac{\lambda}{2} \|f\|^2 \tag{10}$$

with

$$f(\boldsymbol{x}_n) = \sum_N \alpha_n K(X, \boldsymbol{x}_n). \tag{11}$$

SVM are a binary classifier, with the decision rule given by the sign of $f(\boldsymbol{x}_n)$. Because of the type of the objective function some parameters can be zero, i. e.the model is sparse.

2.4 Relevance Vector Machines

The Relevance Vector Machines use the same model as the Kernel Logistic Regression given in Section 2.2. Consequently it can provide probabilistic output and can directly apply to multi-class problems.

The difference between both models is, that the RVM uses an ARD (Automatic Relevance Determination) prior over the model parameters as the regularization term, whereas the prior includes several regularization parameters, also called hyperparameters, which are determined during the optimization process. The difference between an ARD prior and a Gaussian prior used in the Kernel Logistic Regression is, that each parameter α_n has its own hyperparameter and not only one shared parameter λ.

The main disadvantage of the RVM is the long training time. However, because the hyperparameters are optimized simultaneously with the model parameters a cross-validation is not required. The optimization procedure is an Expectation Maximization (EM)-like learning method and can therefore suffer from local minima, whereas the optimization procedure of the SVM is guaranteed to find a global optimum and the greedy selection procedure in the IVM shows a good behavior to find the global optimum as demonstrated in our experiments.

3 Import Vector Machine

The Kernel Logistic Regression includes all training samples X to train the classifier, which is computationally expensive in data sets with many training samples. Similar to the SVM only a few feature vectors are necessary to define the decision boundaries. These feature vectors are called import vectors $\mathsf{X}_{\mathcal{S}}$. Using only these vectors we obtain a sparse solution of the Kernel Logistic Regression – the Import Vector Machines [37].

3.1 IVM Algorithm

Following Zhu and Hastie [37] we only choose a subset $\mathcal{S}$ out of the training set $\mathcal{T}$ with $S = |\mathcal{S}|$ samples and yield for (8) and (9)

$$\boldsymbol{\alpha}^{(i)} = \left(\frac{1}{N} \mathsf{K}_{\mathcal{S}}^{\mathsf{T}} \mathsf{R} \mathsf{K}_{\mathcal{S}} + \lambda \mathsf{K}_R \right)^{-1} \mathsf{K}_{\mathcal{S}}^{\mathsf{T}} \mathsf{R} \tilde{\boldsymbol{z}} \tag{12}$$

$$\tilde{\boldsymbol{z}} = \frac{1}{N}\left(\mathsf{K}_{\mathcal{S}}\boldsymbol{\alpha}^{(i-1)} + \mathsf{R}^{-1}\left(\boldsymbol{p} - \boldsymbol{t}\right)\right) \tag{13}$$

with an $(N \times S)$-dimensional kernel matrix $\mathsf{K}_S = \mathsf{K}(\mathsf{X}, \mathsf{X}_{\mathcal{S}})$, a $(S \times S)$-dimensional regularization matrix $\mathsf{K}_R = \mathsf{K}(\mathsf{X}_{\mathcal{S}}, \mathsf{X}_{\mathcal{S}})$ and the probabilities $p_n = 1/(1+\exp(-\boldsymbol{k}_{\mathcal{S},n}\boldsymbol{\alpha}))$ with $\boldsymbol{k}_{\mathcal{S},n}$ as the n-th row of the kernel matrix $\mathsf{K}_{\mathcal{S}}$.

The algorithm of the IVM is explained in Algorithm 1.

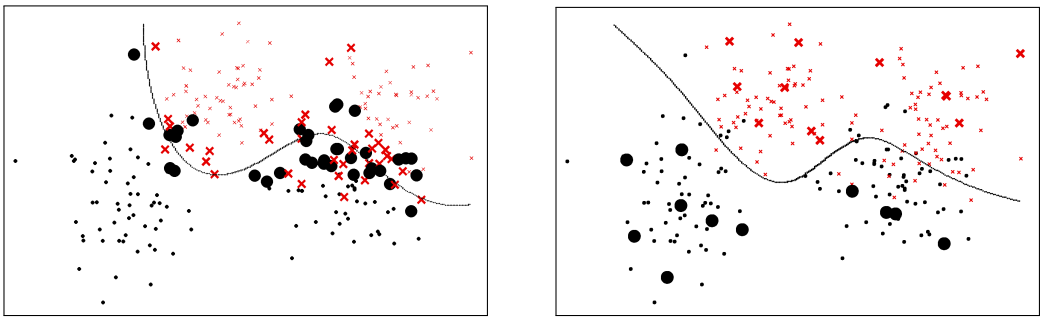
Input: $\mathcal{S} = \{\}$, $\mathcal{T} = \{\boldsymbol{x}_1, \ldots, \boldsymbol{x}_N\}$, $i = 1$, λ, σ
repeat
  Compute $\tilde{\boldsymbol{z}}$ from the current set $\mathcal{S}$;
  foreach $\boldsymbol{x}_n \in \mathcal{T}$ **do**
    Let $\mathcal{S}_n = \mathcal{S} \cup \boldsymbol{x}_n$;
    Compute $\boldsymbol{\alpha}_n^{(i)}$ with $\mathcal{S}_n$ in a one-step iteration;
    Compute $\mathcal{Q}_n^{(i)}$;
  end
  Find $\boldsymbol{x}_{best} \cup \mathcal{S} = \text{argmin}\, \mathcal{Q}_n^{(i)}$;
  Update $\mathcal{S} = \mathcal{S} \cup \boldsymbol{x}_{best}$ and $\mathcal{T} = \mathcal{T} \setminus \boldsymbol{x}_{best}$;
  Recompute $\boldsymbol{\alpha}$, $\boldsymbol{p}$ and $\mathcal{Q}^{(i)}$;
until $\mathcal{Q}$ *converges* ;

Algorithm 1: Import Vector Machines [37]: The algorithm starts with an empty subset $\mathcal{S}$ of import vectors. From the current set $\mathcal{S}$, $\tilde{\boldsymbol{z}}$ (13) is computed. In the next step each point from the training set $\mathcal{T}$ is tested to be in the subset $\mathcal{S}$. From the subset the parameters $\boldsymbol{\alpha}^{(i)}$ (12) are estimated in a one-step iteration, that means the parameters are only updated once. Then for each possible subset $\mathcal{S}_n = \boldsymbol{x}_n \cup \mathcal{S}$ the error function $\mathcal{Q}_n^{(i)}$ is evaluated, which depends on the probabilities of all given training points. The tested point $\boldsymbol{x}_{best} \cup \mathcal{S}$ yielding the lowest error is moved from set $\mathcal{T}$ to subset $\mathcal{S}$. The iteration stops as soon as any convergence criterion is satisfied.

The convergence criterion is proposed by the ratio $\epsilon = |\mathcal{Q}^{(i)} - \mathcal{Q}^{(i-\Delta i)}|/|\mathcal{Q}^{(i)}|$ with a small integer $\Delta i = 3$ and chosen to be $\epsilon = 0.001$.

To save computational cost we do not test every sample to be in the subset $\mathcal{S}$ in each iteration but only a representative part of them. We select these tested samples randomly, because they do not tend to lie near the boundary but spread over the whole feature space. Figure 1 underlines this aspect by comparing IVM with SVM using the Ripley [26] toy example.

Both the IVM and the SVM are discriminative models with a similar objective function. The advantages of the Support Vector Machines are the sparsity and the fast training because of the simple decision rule. On the other



(a) Support Vectors (b) Import Vectors

Abbildung 1: Decision boundaries of SVM and IVM. The bold points are the support vectors and the import vectors respectively.

hand all Logistic Regression models have a probabilistic output, which can be used for example as input in a graphical model [27]. The optimization can be done with the Newton-Raphson procedure, which can be very slow if the feature dimension is very high. The IVM are sparse because of their greedy selection algorithm and is therefore computationally and memory effective.

3.2 Choice of the Kernel and Regularization Parameter

For the algorithm both the kernel parameter σ and the regularization parameter λ are assumed to be fixed.

Different from the determination by a grid search by testing all combinations of the both parameters, Zhu and Hastie proposed a simultaneous selection of the regularization parameter λ and the subset $\mathcal{S}$. First a small tuning set is split off from the training set. The algorithm starts with a high regularization parameter and decreases the parameter each time the algorithm converges until a stopping criterion is reached, e. g., a lower bound for the regularization parameter. In each iteration the error on the tuning set is reported, so that the regularization parameter with the lowest error rate is chosen to be the optimal one.

The kernel parameter is determined by a n-fold cross-validation.

So the IVM need only $2N_\sigma$ runs instead of $N_\lambda N_\sigma$ for the Support Vector Machines yielding an optimal kernel and regularization parameter, whereby N_σ is the number of needed runs for the determination of σ and N_λ the number of needed run for the determination of λ depending on the choice of the type of cross-validation.

3.3 Extension to the Multi-class Case

We can generalize the two class model to the multi-class model. Then the objective function is

$$\mathcal{Q} = -\frac{1}{N}\sum_{N} \boldsymbol{t}_n^{\mathsf{T}} \log \boldsymbol{p}_n + \frac{\lambda}{2}\sum_{k} \boldsymbol{\alpha}_k^{\mathsf{T}} K_R \boldsymbol{\alpha}_k \tag{14}$$

with the probabilities $P = [p_1, \ldots, p_N]$ obtained by

$$p_{nk} = \frac{\exp(\boldsymbol{k}_{\mathcal{S},n}\boldsymbol{\alpha}_k)}{\sum_j \exp(\boldsymbol{k}_{\mathcal{S},n}\boldsymbol{\alpha}_j)}. \tag{15}$$

The binary target vector $\boldsymbol{t}_n$ of length K uses the 1-of-K coding scheme so that all components but t_{nk} are 0 if the point $\boldsymbol{x}_n$ is from class $\mathcal{C}_k$.

In the Newton-Raphson procedure in (12) and (13) we have to use one R_k and one $\tilde{\boldsymbol{z}}$ for each class. In consequence of the over-determined system we need to apply the pseudo-inverse $(\cdot)^+$ instead of the normal inverse $(\cdot)^{-1}$.

4 Experimental Setup and Results

4.1 Data sets

Two hyperspectral data sets from study sites with different environmental setting were used in this study. Both data sets have been used in a multitude of studies, e. g., [18, 24, 35] and thus, regarded as kind of benchmark data sets for comparison with new approaches.

The first image was acquired by ROSIS-3 sensor in 2003. The spatial resolution of the image is 1.3 meter per pixel. The data cover the range from 0.43 m to 0.86 m of the electromagnetic spectrum. However, some bands have been removed due to noise and finally 102 channels have been used in the classification. The image strip, with 1096×492 pixels in size, lies around the center of Pavia. The classification is aiming on 9 land cover classes, ranging from 2152 and 103551 samples in size (Table 1).

The second data set was acquired by the AVIRIS instrument in 1992. The study site lies in a predominately agricultural region in NW Indiana, USA. AVIRIS operates from the visible to the short-wave infrared region of the electromagnetic spectrum, ranging from 0.4 m to 2.4 m. The data set covers 145×145 pixels, with a spatial resolution of 20 m per pixel. The experiments are aiming on the classification of 14 classes, ranging from 54 to 2466 samples in size (Table 2).

Tabelle 1: Pavia data set. Number of training and test samples.

class	# training samples	# test samples
Asphalt	678	6907
Bare Soil	820	5729
Bitumen	808	6479
Meadows	797	2108
Bricks	485	1667
Shadow	195	1970
Tiles	223	2899
Trees	785	5723
Water	745	64533
Total	5536	98015

4.2 Experimental Setup

For both images ground truth information was used for generating training and validation sets, using an equalized random sampling. In doing so, it is guaranteed that all classes are equally included in the training sets, given that a sufficient number of samples is available for each class. Training sets with different size were generated to assess the possible impact of the number of training samples on the performance of IVM, containing, e. g., 25, 50, and 100 training samples per class, respectively (from now on referred to as tr#25, tr#50, and tr#100). The same training sample sets were used for both methods, IVM and SVM. In addition, an independent test set was used that was fixed in all experiments.

In the presented study the performance of IVM is compared to SVM. SVM are perhaps the most popular approach in more recent applications and seems particularly advantageous when classifying high-dimensional data sets. Thus, the method is regarded as a kind of benchmark classifier for comparison with new approaches.

For both methods and each training set size (e. g., tr#50) the training and classification was performed 50 times to reduce the impact of the random generation of the training sample sets. The final results were averaged and standard deviations were derived.

Accuracy assessment was performed giving kappa and confusion matrices that were used to derive the producers and users accuracies. In addition, the sparsity of IVM and SVM was compared.

For the SVM and the IVM we transform the features into kernel space with a radial basis function kernel. The kernel parameters are determined by a 3-fold cross-validation. The SVM classification is performed in MATLAB,

Tabelle 2: Indian Pines data set. Number of training and test samples.

class	# training samples	# test samples
Alalfa	27	27
Corn-notil	717	717
Corn-min	417	417
Corn	117	117
Grass-pasture	248	249
Pasture-trees	373	374
Hay	244	245
Soy-notil	484	484
Soy-mid	1234	1234
Soy-clean	307	307
Wheat	106	106
Woods	647	647
Bldg-grass	190	190
Stone	47	48
Total	5158	5162

using the LIBSVM approach by Chang and Lin [6]. The IVM algorithm is based on our own implementation.

In our experiments we do not consider the RVM in detail, because first results have shown that RVM generate relatively instable results and require a long training time. We use a multi-class implementation in MATLAB [1]. In some cases of the 50 iteration (e. g., using tr#25, tr#50) the RVM can achieve the same accuracies as the SVM and the IVM on the Pavia data. However, other accuracies are significantly lower, resulting in a high standard deviation and lower average accuracy. In addition the training time is significantly increasing with an increasing number of samples. Similar findings can be reported for the Indian Pines data set. Especially for tr#10 and tr#25 the results are instable. Thus, RVM are not further considered in our studies.

4.3 Results for Pavia

The classifications in this paper were conducted with IVM and SVM, using training sets with different number of samples. The results demonstrate that the IVM perform at least equally well than the SVM in terms of accuracy, irrespectively from the number of training samples. The IVM achieved average kappa coefficients between 0.93 and 0.95, a regular SVM on the other hand

[1] available under `http://mi.eng.cam.ac.uk/~at315/MVRVM`

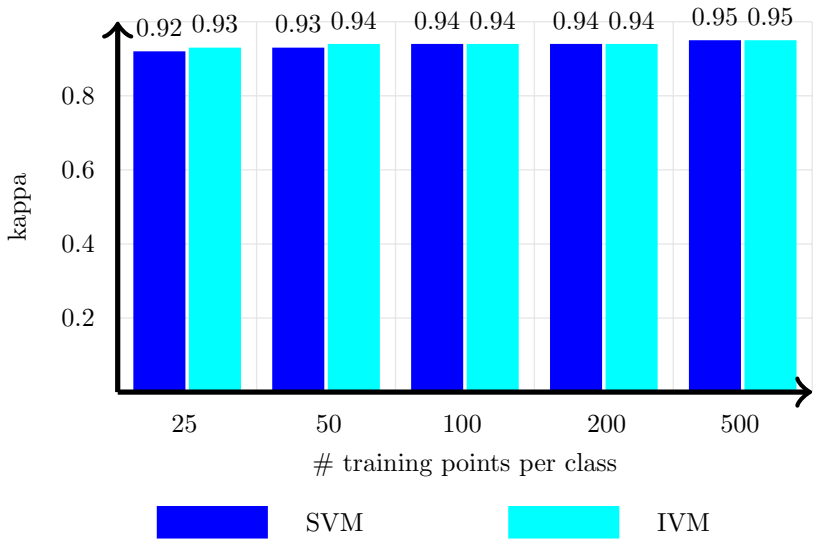


Abbildung 2: Pavia data set. Average kappa coefficient, using SVM and IVM with different number of training samples per class.

achieved average accuracies between 0.92 and 0.95 (Figure 2). The standard deviation of kappa varies between 0.003 and 0.01 for both methods.

Figure 3 shows the ground truth and a classification result with tr#150 of the IVM algorithm.

The analysis of the producer and user accuracies confirms the aforementioned findings. With the exception of bricks (e. g., SVM and tr#500), the results demonstrate that neither IVM nor SVM outperform the other method in terms of accuracy (Table 3). Whereas in some cases the IVM achieve higher user and producer accuracies (e. g., asphalt, using tr#25), other classes are classified more accurately by the SVM (e. g., bricks using tr#25) or both methods generate comparable results (e. g., bare soil). Irrespectively of the method, the experimental results show a positive effect of additional training samples and the class accuracies are increased (e. g., bricks and trees).

Comparing the number of used support vectors and import vectors, respectively, the results confirm that the number of support vectors clearly increases with an increasing number of training samples (Figure 4). Whereas the SVM training with the sample set tr#25 results on average in 101 support vectors, approximately 518 are used when the training is performed with the large sample set tr#500. The number of import vectors on the other hand, does not clearly increase with the number of training samples and varies approximately between 81 and 86.

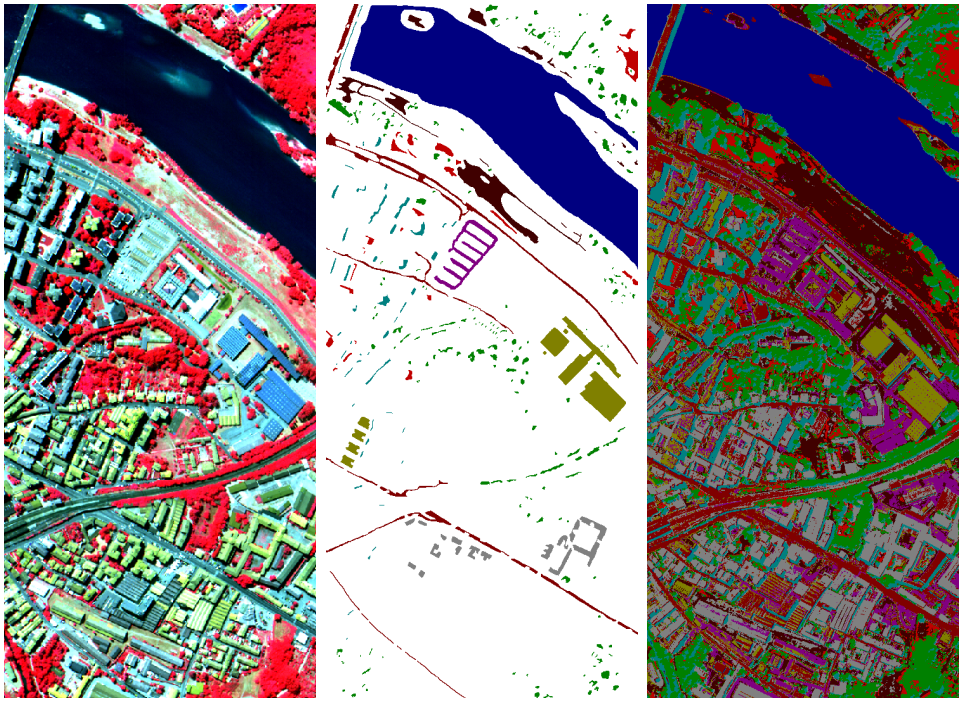


Abbildung 3: Pavia data (left), ground truth (middle) and classification result (right), using IVM and tr#150.

4.4 Results for Indian Pines

As for the Pavia data set, results achieved with the IVM show comparable accuracies than those from the regular SVM for all five training sample sets. The kappa coefficient varies between 0.59 and 0.82, using IVM, while the results that are achieved with the SVM vary between 0.58 and 0.82 (Figure 5). The 50 classifications that were generated for each training set size (e. g., tr#10) show only a few variations and the standard deviation for kappa varies between 0.01 and 0.04 for both methods.

Figure 3 shows the ground truth and a classification result with tr#150 of the IVM algorithm.

The results demonstrate the usually dependency of the overall and class accuracies on the number of available training samples. Whereas a small training sample set generates relatively low class accuracies, which are sometimes remain below 60% (e. g., corn-notil, soy-clean), the producer as well as user accuracies are clearly increased by a larger number of training samples. However, the analysis of the producer and user accuracies also confirms the previous findings that IVM and SVM perform comparable in terms of the class accuracies (Table 4).

classes	25 training points per class				500 training points per class			
	User acc. [%]		Prod. acc. [%]		User acc. [%]		Prod. acc. [%]	
	SVM	IVM	SVM	IVM	SVM	IVM	SVM	IVM
Asphalt	85.4	88.6	93.2	94.5	88.7	89.2	95.7	96.3
Bare Soil	93.2	93.2	93.9	94.7	95.2	94.7	94.3	96.5
Bitumen	89.1	92.8	81.4	80.3	87.5	96.6	95.3	86.0
Meadows	70.2	71.5	92.0	91.9	80.0	77.9	95.1	94.8
Bricks	55.4	54.6	76.5	75.1	84.9	63.0	83.0	81.1
Shadow	99.9	99.6	100.0	99.9	100.0	99.5	100.0	99.9
Tiles	97.2	95.2	99.1	98.1	98.4	98.8	99.8	97.9
Trees	96.4	97.3	86.5	86.1	98.5	98.4	91.6	90.0
Water	100.0	100.0	98.5	99.4	100.0	100.0	98.3	99.4

Tabelle 3: Pavia data set. Average user's and producer's accuracies, using 25 training and 500 training samples per class.

As before, the number of support vectors strongly depends on the size of the training sample set and varies between 115 and 926. Also the number of import vectors used by the IVM increases with the number of training samples. However, the number is significantly lower when compared to the number of support vectors and varies on average between 52 and 164 (Figure 7).

5 Discussion

The potential of IVM for classifying hyperspectral imagery was discussed. The proposed method was shown to be generally positive for experimental results and usually perform at least equally well. This general trend exists independent from the study site, the number of training samples and specific classes, as shown by the kappa coefficient and the producers and users accuracies.

Experimental results showed some impact of training sample size. Whereas the Pavia data set was classified very accurately by a small number of training samples, and thus, accuracies only slightly increased by additional training samples, the accuracies of the second data set significantly depends on the number of available training samples.

This is in accordance with results in previous studies dealing with SVM. Although the method performs efficiently with small training sets, even when classifying high dimensional imagery, the accuracy is affected by the number

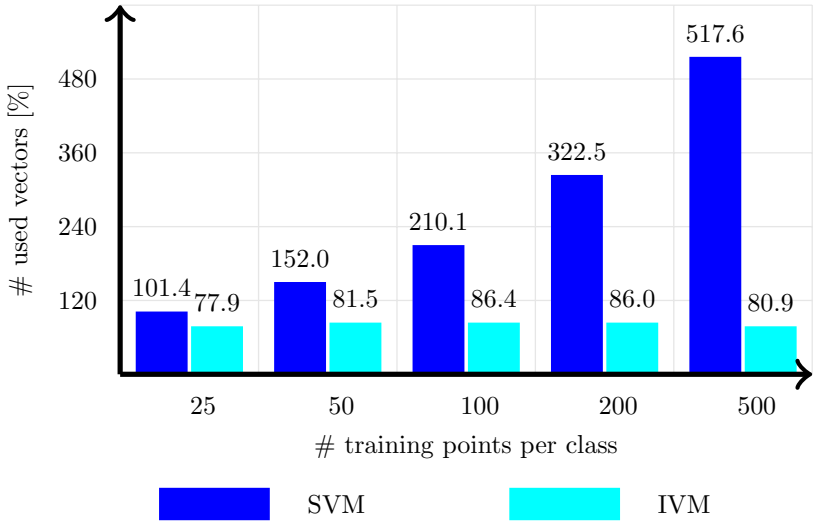


Abbildung 4: Pavia data. Number of training points per class versus average number of import vectors and support vectors.

of features (i.e., the Hughes phenomenon) and available training samples [22, 34]. Therefore, the use of an adequate number of training samples is recommended, also in context of SVM and IVM.

However, the number of support vectors significantly increases with the number of available training samples and clearly exceeds the number of import vectors in all cases. In contrast to this, the number of import vectors remains almost constant on the Pavia data set and show a small increase on the second data set. Consequently, the computation time of the IVM during the classification is much faster when compared to SVM, because the number of mathematical operations to perform depends on the number of support and import vectors.

Moreover, IVM directly provide probability outputs, which can be used for further processing like Discriminative Random Fields [27, 15], and can apply for multi-class problems without specific multi-class strategies.

Despite these advantages, the matter of computational complexity during the IVM training process might be the main drawback of the approach. The proposed IVM technique is based on the Newton-Raphson optimization scheme, and thus, results in a longer training time when compared to SVM. However, this fact should be discussed against the background of possible methods to reduce computation times, e. g., due to GPU implementations and incremental learning strategies, which was recently discussed in the context of SVM [4, 21].

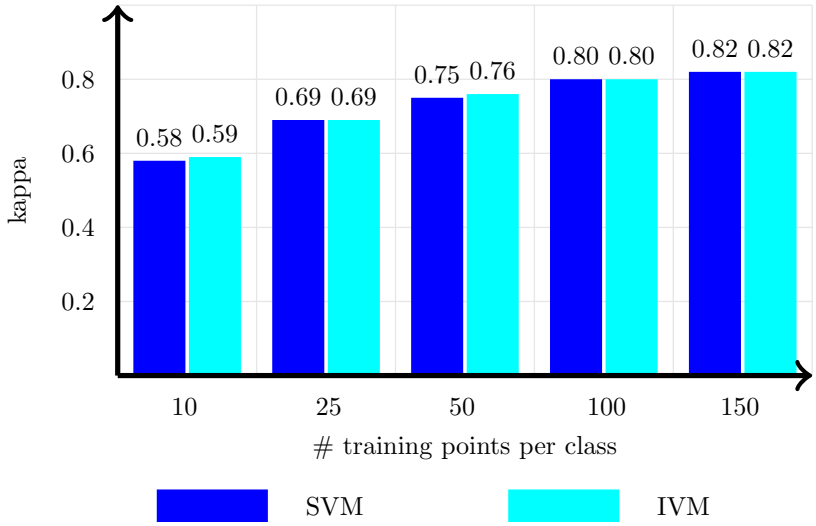


Abbildung 5: Indian Pines data. Average kappa coefficient, using SVM and IVM with different number of training points per class.

Abbildung 6: Indian Pines datas (left), ground truth (middle) and classification result (right), using IVM and tr#150.

6 Conclusion and Outlook

We tested and evaluated the performance of IVM in the context of classifying hyperspectral imagery. Regarding the three research questions stated in the Section I, it can be assessed that the proposed IVM method performs similar when compared to SVM in terms of accuracy. This finding is independent from the study site, the number of training samples and specific classes. As expected, the classification accuracies are enhanced by additional training samples, and thus, the use of large training sample set can be advantages in terms of accuracy.

However, our experimental results underline the strong dependency of the

classes	10 training points per class				150 training points per class			
	User acc. [%]		Prod. acc. [%]		User acc. [%]		Prod. acc. [%]	
	SVM	IVM	SVM	IVM	SVM	IVM	SVM	IVM
Alalfa	40.3	45.0	88.6	89.8	92.1	93.7	89.6	85.6
corn-notil	58.1	60.4	45.6	53.0	81.4	80.9	78.2	79.4
corn-min	42.6	44.3	54.7	52.7	70.2	70.1	81.6	81.6
corn	26.9	30.9	70.8	69.3	59.1	63.0	92.7	85.7
grass-pasture	72.8	72.8	83.0	78.3	90.5	91.1	94.2	94.3
pasture-trees	86.3	86.8	87.4	90.3	94.9	94.8	97.3	97.9
hay	99.2	98.7	87.4	89.9	99.7	99.1	98.8	99.0
soy-notil	52.1	48.1	61.2	62.2	77.5	73.5	82.5	84.7
soy-mid	69.8	68.9	48.3	46.8	88.5	89.3	72.6	71.2
soy-clean	43.9	52.3	50.0	57.0	78.1	78.9	92.0	91.7
wheat	80.8	79.2	98.1	98.4	99.1	99.8	99.1	99.1
woods	93.8	93.0	80.0	77.5	96.6	96.2	90.3	91.3
bldg-grass	42.9	42.4	48.0	51.2	66.3	68.2	83.4	82.8
stone	92.9	87.8	93.3	92.6	95.9	93.5	97.2	96.0

Tabelle 4: Indian Pines data. Average user's and producer's accuracies, using 25 training and 500 training samples per class.

number of support vectors on the number of available training samples. In contrast to this, the number of import vectors is significantly lower when compared to the number of support vectors. Moreover, the number of import vectors shows only a slight increase, with an increasing number of training samples.

Overall, the proposed IVM approach appears worthwhile and efficient implementation strategies and further modifications should be investigated. Particularly for hyperspectral data sets, which require a sufficient large number of training samples to ensure an adequate accuracy, IVM constitute a feasible approach and an useful alternative for classification. Another important advantage of the IVM, is the provision of a probabilistic output. These probabilities can be used, for example, as indicator for classification uncertainty and input in Markov Random Fields. Thus, a more detailed analysis of the IVM outputs is foreseen.

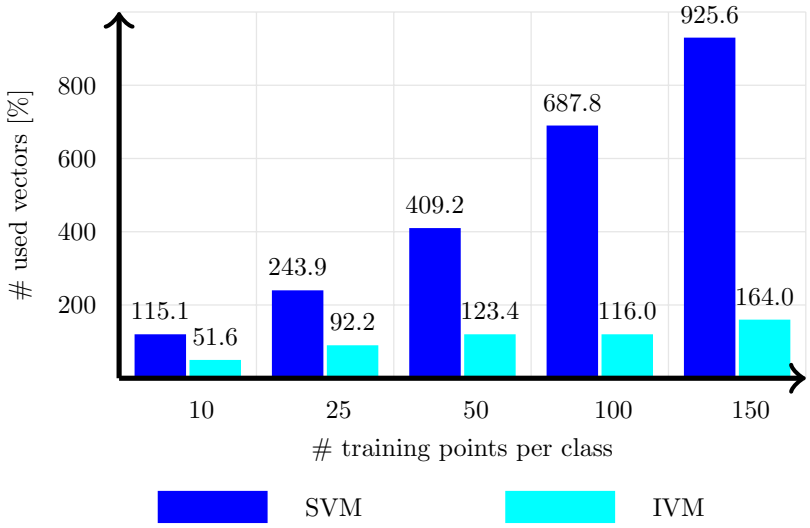


Abbildung 7: Indian Pines data. Number of training points per class versus average number of import vectors and support vectors.

Acknowledgment

The authors would like to thank D. Landgrebe and L. Biehl (Purdue University, USA) for providing the Indian Pines data[2] and P. Gamba (University of Pavia, Italy) for providing the Pavia dataset.

Literatur

[1] M.A. Aizerman, E.M. Braverman, and L. Rozonoèr. Theoretical Foundations of the Potential Function Method in Pattern Recognition Learning. *Automation and Remote Control*, 25(6):821–837, 1964.

[2] J.A. Benediktsson, J.A. Palmason, and J.R. Sveinsson. Classification of Hyperspectral Data from Urban Areas based on Extended Morphological Profiles. *IEEE Trans. Geosci. Remote Sens.*, 43(3):480–491, 2005.

[3] G. Camps-Valls, N. Shervashidze, and K. M. Borgwardt. Spatio-Spectral Remote Sensing Image Classification with Graph Kernels. *IEEE Geosci. Remote Sens. Lett.*, 7(4):741–745, 2010.

[2]available on: `http://cobweb.ecn.purdue.edu/~biehl/MultiSpec/`

[4] G. Cauwenberghs and T. Poggio. Incremental and Decremental Support Vector Machine Learning. In *Advances in Neural Information Processing Systems*, pages 409–415, 2001.

[5] G.C. Cawley and N.L.C. Talbot. Efficient Model Selection for Kernel Logistic Regression. *Pattern Recogn.*, 2:439–442, 2004.

[6] C.C. Chang and C.J. Lin. LIBSVM: A Library for Support Vector Machines, 2001.

[7] X. Chen, T.A. Warner, and D.J. Campagna. Integrating Visible, Near-Infrared and Short-Wave Infrared Hyperspectral and Multispectral Thermal Imagery for Geological Mapping at Cuprite, Nevada. *Remote Sens. Environ.*, 110(3):344–56, 2007.

[8] Q. Cheng, P.K. Varshney, and M.K. Arora. Logistic regression for feature selection and soft classification of remote sensing data. *IEEE Geosci. Remote Sens. Lett.*, 3(4):491–494, 2006.

[9] M. Chi and L. Bruzzone. Semisupervised Classification of Hyperspectral Images by SVMs Optimized in the Primal. *IEEE Trans. Geosci. Remote Sens.*, 45(6):1870–1880, 2007.

[10] B. Demir and S. Erturk. Hyperspectral Image Classification Using Relevance Vector Machines. *IEEE Geosci. Remote Sens. Lett.*, 4:586–590, 2007.

[11] G.M. Foody. RVM-Based Multi-Class Classification of Remotely Sensed Data. *Int. J. Remote Sens.*, 29(6):1817–1823, 2008.

[12] A.F.H. Goetz. Three Decades of Hyperspectral Remote Sensing of the Earth: A Personal View. *Remote Sens. Environ.*, 113:5–16, 2009.

[13] L. Guanter, K. Segl, and H. Kaufmann. Simulation of Optical Remote-Sensing Scenes With Application to the EnMAP Hyperspectral Mission. *IEEE Trans. Geosci. Remote Sens.*, 47(7):2340–2351, 2009.

[14] S.S. Keerthi, K.B. Duan, S.K. Shevade, and A.N. Poo. A Fast Dual Algorithm for Kernel Logistic Regression. *Mach. Learn.*, 61(1):151–165, 2005.

[15] S. Kumar and M. Hebert. Discriminative Random Fields. *Int. J. Comput. Vision*, 68(2):179–201, 2006.

[16] J. Lafferty, X. Zhu, and Y. Liu. Kernel Conditional Random Fields: Representation and Clique Selection. page 64, 2004.

[17] A. Mathur and GM Foody. Multiclass and binary SVM classification: implications for training and classification users. *IEEE Geosci. Remote Sens. Lett.*, 5(2):241–245, 2008.

[18] F. Melgani and L. Bruzzone. Classification of Hyperspectral Remote Sensing Images with Support Vector Machines. *IEEE Trans. Geosci. Remote Sens.*, 42(8):1778–1790, 2004.

[19] G.H. Mitri and I.Z. Gitas. Mapping Postfire Vegetation Recovery Using EO-1 Hyperion Imagery. *IEEE Trans. Geosci. Remote Sens.*, 48(3):1613–1618, 2010.

[20] J. Muñoz-Marí, F. Bovolo, L. Gómez-Chova, L. Bruzzone, and G. Camp-Valls. Semisupervised One-Class Support Vector Machines for Classification of Remote Sensing Data. *IEEE Trans. Geosci. Remote Sens.*, 48(8):3188–3197, 2010.

[21] J. Muñoz-María, A.J. Plazab, J.A. Gualtieric, and G. Camps-Vallsa. *Parallel Implementations of SVM for Earth Observation.* IOS Press, Amsterdam, The Netherlands, The Netherlands, 2009.

[22] M. Pal and G.M. Foody. Feature Selection for Classification of Hyperspectral Data by SVM. *IEEE Trans. Geosci. Remote Sens.*, 48(5):2297–2307, 2010.

[23] M. Pal and P.M. Mather. Some Issues in the Classification of DAIS Hyperspectral Data. *Int. J. Remote Sens.*, 27(14):2895–2916, 2006.

[24] A. Plaza, J.A. Benediktsson, J.W. Boardman, J. Brazile, L. Bruzzone, G. Camps-Valls, J. Chanussot, M. Fauvel, P. Gamba, A. Gualtieri, M. Marconcini, J.C. Tilton, and G. Trianni. Recent Advances in Techniques for Hyperspectral Image Processing. *Remote Sens. Environ.*, 113:110–122, 2009.

[25] J.A. Richards. Analysis of Remotely Sensed Data: The Formative Decades and the Future. *IEEE Trans. Geosci. Remote Sens.*, 43(3):422–432, 2005.

[26] B.D. Ripley. *Pattern Recognition and Neural Networks.* Cambridge University Press, 2008.

[27] R. Roscher, B. Waske, and W. Förstner. Kernel Discriminative Random Fields for Land Cover Classification. In *Proc. IAPR Workshop on Pattern Recognition and Remote Sensing*, 2010.

[28] V. Roth. Probabilistic Discriminative Kernel Classifiers for Multi-class Problems. In *Pattern Recognition: 23rd DAGM Symposium (Lecture Notes in Comput. Sci.)*, pages 246–253, 2001.

[29] B. Somers, S. Delalieux, W.W. Verstraeten, J. Verbesselt, S. Lhermitte, and P. Coppin. Magnitude- and Shape-Related Feature Integration in Hyperspectral Mixture Analysis to Monitor Weeds in Citrus Orchards. *IEEE Trans. Geosci. Remote Sens.*, 47(11):3630–3642, 2009.

[30] M.E. Tipping. Sparse Bayesian Learning and the Relevance Vector Machine. *J. of Mach. Learn. Research*, 1:211–244, 2001.

[31] S. Van der Linden, A. Janz, B. Waske, M. Eiden, and P. Hostert. Classifying Segmented Hyperspectral Data from a Heterogeneous Urban Environment using Support Vector Machines. *J. Appl. Remote Sens.*, 1(1), 2007.

[32] V.N. Vapnik. *The Nature of Statistical Learning Theory*. Springer, 2000.

[33] B. Waske, J.A. Benediktsson, K. Arnason, and J.R. Sveinsson. Mapping of Hyperspectral AVIRIS Data using Machine-Learning Algorithms. *Can. J. Remote Sensing*, 35:106–116, 2009.

[34] B. Waske, S. van der Linden, J.A. Benediktsson, A. Rabe, and P. Hostert. Sensitivity of Support Vector Machines to Random Feature Selection in Classification of Hyperspectral Data. *IEEE Trans. Geosci. Remote Sens.*, 48(7):2880–2889, 2010.

[35] Jinn-Min Yang, Bor-Chen Kuo, Pao-Ta Yu, and Chun-Hsiang Chuang. A Dynamic Subspace Method for Hyperspectral Image Classification. *IEEE Trans. Geosci. Remote Sens.*, 48(7):2840–2853, 2010.

[36] Ping Zhong, Peng Zhang, and Runsheng Wang. Dynamic Learning of SMLR for Feature Selection and Classification of Hyperspectral Data. *IEEE Geosci. Remote Sens. Lett.*, 5(2):280–284, APR 2008.

[37] J. Zhu and T. Hastie. Kernel Logistic Regression and the Import Vector Machine. *J. Comput. Graph. Stat.*, 14(1):185–205, 2005.