

# A Consistency-Based Model Selection for One-class Classification

David M.J.Tax

Delft University of Technology  
2600 GA Delft, The Netherlands  
D.M.J.Tax@ewi.tudelft.nl

Klaus-Robert Müller

Fraunhofer FIRST.IDA, Kekuléstr.7,  
12489 Berlin, Germany  
Klaus-Robert.Mueller@first.fhg.de

## Abstract

*Model selection in unsupervised learning is a hard problem. In this paper a simple selection criterion for hyperparameters in one-class classifiers (OCCs) is proposed. It makes use of the particular structure of the one-class problem. The main idea is that the complexity of the classifier is increased until the classifier becomes inconsistent on the target class. This defines the most complex classifier which can still reliably be trained on the data. Experiments indicated the usefulness of the approach.*

## 1. Introduction

When classifiers are optimized using a training set, it is almost always assumed that this data is a good reflection of what will be encountered in future data. An example is machine diagnostics where faulty operating machines should be distinguished from normally operating machines. It is very hard, expensive and sometimes impossible, to sample sufficiently from the faulty operating mode; for each example a machine (e.g. an engine) has to sacrificed [11, 3].

To solve these classification problems, one-class classification (OCC) has proved to be successful. Here a *target* class of objects has to be distinguished from all other objects, which will be considered the *outliers* [8]. It is assumed that a well-sampled training set of the target class is available, while for the outlier class, no (or very few) examples are present. One-class classifiers are then trained to minimize the error on both the target as the outlier class.

Any one-class classifier contains hyperparameters which determine the classifier complexity (basically their flexibility). In a nutshell: the more complex the model, the more flexible the boundary and the tighter it will fit the target data. This, however, comes at the price of overfitting, where the flexible boundary mainly adapts to noise, and the generalization on the target class deteriorates.

Unlike in supervised classification problems, in OCC the standard model selection methods (minimizing the er-

ror on a cross-validation set) will not work. For unsupervised learning, model selection criteria (the number of clusters in the data, the errorbars for ICA) based on stability have been introduced [6, 1]. For OCC, the main interest is to obtain a stable and reliable boundary around the target class, which is as complex as the data allows. Unfortunately, because neither a stable clustering, nor stable model parameters are directly required for stable boundaries the mentioned procedures are not directly applicable.

In this paper we propose to evaluate the consistency of the OC classifier, using *only* the error on the target class. For this, the complexity of the OC classifier is increased until the classifier becomes inconsistent, which means that the performance on the target class on a validation set deviates more than can be expected from the stochastic nature of the training data. This results in the most complex classifier which can still be estimated reliably on the target data.

## 2. Theory

Assume a training set  $\mathcal{X}^{tr} = \{\mathbf{x}_i, i = 1, \dots, n\}$ ,  $\mathbf{x}_i \in \mathbb{R}^d$  is drawn from a target distribution  $p(\mathbf{x})$ , for which the support should be characterized by a classifier. One-class classifiers all have the following form:

$$f(\mathbf{z}|K) = \mathbf{1}(h_\alpha(\mathbf{z}|K) > \theta) = \begin{cases} 1 & : \mathbf{z} \text{ is a target object,} \\ 0 & : \mathbf{z} \text{ is an outlier,} \end{cases} \quad (1)$$

where function  $h$  is modeling the similarity to the (training) target data  $\mathcal{X}^{tr}$ ,  $\alpha$  are its free parameters,  $\theta$  is a threshold on this similarity and  $\mathbf{1}(\cdot)$  is the indicator function. Furthermore,  $K$  is the parameter of the model  $h$  indicating the complexity<sup>1</sup>. Parameter  $\theta$  is optimized to reject a certain (user specified) fraction  $\nu$  of the target class, say  $\nu = 1\%$ . This specifies the error on the target class  $\varepsilon^{\dagger,2}$ . Given this constraint, function  $h_\alpha$  should be optimized such that it mini-

1 It appears that almost all OC classifiers have some parameter, which influence the flexibility, and thus the complexity of the classifier.

2 Due to estimation problems, the true  $\varepsilon_K^{\dagger}$  for a given  $K$  might be much different from the user-supplied  $\nu$ .

mizes the error on the outliers  $\varepsilon_K^o$ . Assuming a uniform outlier distribution, it means that volume captured by the classifier (for a given  $K$ ) should be minimized:

$$\min_{\alpha} \varepsilon_K^o = \min_{\alpha} \int_{\mathbb{R}^d} \mathbf{1}(h_{\alpha}(\mathbf{z}|K) > \theta) d\mathbf{z}, \quad (2)$$

under the constraint:  $\int_{\mathbb{R}^d} \mathbf{1}(h_{\alpha}(\mathbf{z}|K) > \theta) p(\mathbf{z}) d\mathbf{z} = 1 - \nu$ .

When the optimization of  $h(\mathbf{z})$  is independent of the determination of  $\theta$ ,<sup>3</sup> the constraint changes into:

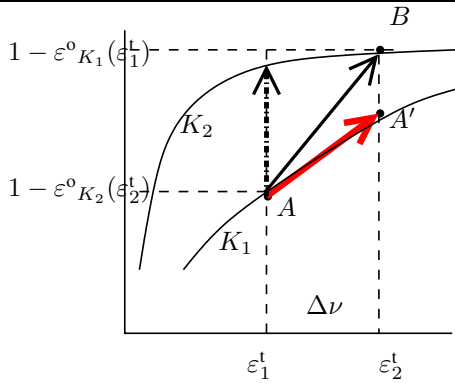
$$\int_0^{\theta} p(h_{\alpha}(\mathbf{z})) dh_{\alpha}(\mathbf{z}|K) = \nu \quad (3)$$

The optimal solution for (2) (with constraint) is  $h_{\alpha}(\mathbf{x}) = p(\mathbf{x})$  with  $\theta$  determined as in (3). This defines the minimum volume set for this distribution and this  $\nu$  [9].

## 2.1. Area under the ROC curve

To optimize  $K$ , the total error of the classifier has to be discussed. A robust way of evaluating one-class classifiers, is to compute the Receiver-Operating-Characteristic (ROC) curve [7], basically given by  $1 - \varepsilon_K^o$  for each value of  $\varepsilon_K^t$ . To compare different curves, the area under the ROC curve (AUC) is computed [2]:

$$\text{AUC}_K = 1 - \int_0^1 \varepsilon_K^o(\varepsilon_K^t) d\varepsilon_K^t \quad (4)$$



**Figure 1. The ROC curve, i.e.  $1 - \varepsilon^o$  as function of  $\varepsilon^t$ , for increasing the complexity  $K_1 \rightarrow K_2$  for a given  $\varepsilon^t$ .**

In general, the more complex the classifier is, the more flexible and the higher the resolution at which it can follow the boundary of the dataset. Because the volume will

<sup>3</sup> This is not always the case; the support vector data description, and the linear-programming classifiers optimize both  $\alpha$  and  $\theta$ .

be minimized by Equation (2), using a more flexible boundary means that a lower outlier error  $\varepsilon^o$  will be obtained. We therefore assume that when the complexity is increased from  $K_1$  to  $K_2$ :

$$\varepsilon_{K_2}^o(\varepsilon_1^t) \leq \varepsilon_{K_1}^o(\varepsilon_1^t) \quad (5)$$

and therefore  $\text{AUC}_{K_2} \geq \text{AUC}_{K_1}$ .

On the other hand increasing the  $K$  will make  $h$  more sensitive to overfitting, which means that the error on the target class  $\varepsilon^t$  will be larger,  $\varepsilon_2^t > \varepsilon_1^t$ . In figure 1 both effects are shown: for fixed  $\nu$  the  $\varepsilon^o$  should decrease (dash-dotted arrow), but due to overfitting on the target data,  $\varepsilon^t$  will also increase (solid arrow). So we are facing the classical bias-variance trade-off [5]: to minimize the covered volume, we should increase the classifier complexity, but to avoid overfitting on the target class, we should reduce it.

Figure 1 shows what happens when the complexity parameter is changed for a fixed user-supplied  $\nu$ . Using complexity  $K_1$  a classifier  $A$  is obtained. Increasing the complexity to  $K_2$  a classifier  $B$  is found. In this case  $\varepsilon^o$  decreases ( $1 - \varepsilon^o$  increases), but  $\varepsilon^t$  increases. It can be shown that the total AUC will improve when for all  $\nu$  the target error of  $B$  is smaller than that of  $A'$  (see Figure). Using an approximation for the derivative  $\frac{\partial \varepsilon_{K_1}^o}{\partial \varepsilon^t}$  this can be written as:

$$\varepsilon_{K_2}^o(\varepsilon_2^t) < \varepsilon_{K_1}^o(\varepsilon_1^t + \Delta\varepsilon^t) \quad (6)$$

When no overfitting on the target data occurs,  $\Delta\nu = 0$ , (6) reduces to our assumption (5), and the AUC will decrease. For large  $\Delta\nu$  (heavy overfitting), (6) might not hold, as is already suggested in the extreme right of figure 1 where  $\varepsilon_{K_2}^o$  and  $\varepsilon_{K_1}^o$  will eventually cross.

## 2.2. Consistency

Increasing the complexity  $K$  will therefore increase the AUC error, provided that the  $\Delta\varepsilon^t$  will be (close to) zero.<sup>4</sup> To check if  $\Delta\varepsilon^t$  stays close to zero, the classifier should be validated on some validation data. Assume that a one-class classifier  $h$  is trained to reject a fraction  $\varepsilon^t$  of the target objects, and assume that a validation set  $\{\mathbf{z}_i, i = 1 \dots M\}$  is available. We can estimate the error  $\hat{\varepsilon}^t$  on this set:

$$\hat{\varepsilon}^t = \sum_{i=1}^M (1 - \mathbf{1}(h_{\alpha}(\mathbf{z}_i|K) > \theta)) \quad (7)$$

This can be modeled as  $M$  binomial experiments. The expected number of rejected objects, and the variance, is

$$E[\hat{\varepsilon}^t] = M\varepsilon^t, \quad V[\hat{\varepsilon}^t] = M\varepsilon^t(1 - \varepsilon^t) \quad (8)$$

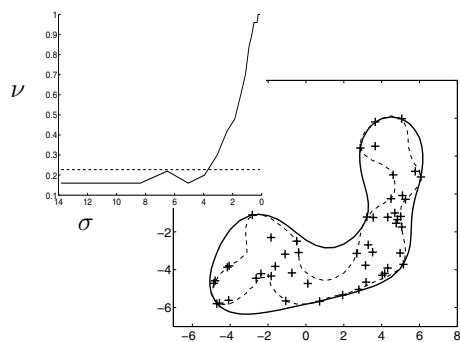
<sup>4</sup> In principle, this should hold for all values of  $\varepsilon^t$ . In practical applications, one is often interested in a small range of  $\varepsilon^t$ , for instance, the error on the target class is expected between 1% and 15%. To keep the notation simple, we will use a single, user-supplied value for  $\varepsilon^t$ . In practice an average  $\varepsilon^t$  can be used.

When the estimated number of rejected target objects exceeds (say) the  $2\sigma$  bounds around this average, the classifier will be called *inconsistent*:

$$\text{Classifier } f(\mathbf{x}|K) \text{ is inconsistent at level } \varepsilon^t \text{ if} \\ \hat{\varepsilon}^t \geq \varepsilon^t + 2\sqrt{M\varepsilon^t(1-\varepsilon^t)} \quad (9)$$

To use this inconsistency criterion, the classifier is trained first with the hyperparameter which yields the most simple classifier. The complexity parameter is then increased until the classifier becomes inconsistent. The hyperparameter for which the boundary could still be estimated reliably, will then be used. In order to make this possible, an ordering of the complexity parameters  $K_i$  should be given such that the complexity increases with increasing  $i$ . A further advantage is, that the classifier does not have to be trained for the more complex hyperparameter settings.

To obtain an independent estimation of  $\varepsilon^t$ , several standard methods can be applied. For methods having a random initialization (e.g. for  $k$ -means or neural network models), training several times will yield different results. But this will fail for deterministic methods. When sufficient data is available, a separate validation set can be used. Or when an appropriate noise level in the data can be assumed, noise injection is possible. In this paper we use cross-validation, because this can be applied for all datasets. Unfortunately this will introduce some extra noise, and this requires an extra offset in Equation (9). For sufficiently large training sets its effect vanishes, and due to space constraints we will not go deeper into this.

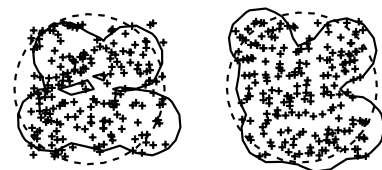


**Figure 2. Scatterplot of a simple 2D banana-shaped dataset. Left) the fraction of rejected target objects is shown for a varying parameter  $\sigma$  in the SVDD,  $\nu = 0.1$ . Right) scatterplot with the optimized decision boundary.**

In figure 2 the results for a simple 2D banana-shaped dataset is shown for the SVDD. The training set contains 50 objects, and the validation set is generated using 5-fold cross-validation. The hyperparameter  $\sigma$  is changed for the classifier, and the  $\nu$ 's are depicted in the smaller upper-left picture. The consistency threshold (from (9)) is indicated by the dashed line. For the SVDD  $\sigma = 4$  is still consistent. The corresponding classifier is shown in the scatterplot. An overfitted SVDD with  $\sigma = 2$  is shown by the dashed line.

### 2.3. Consistency is not stability

As already mentioned, stability criteria for model selection in clustering problems have been used before [6]. Some



one-class classifiers, like the  $k$ -means method, use clustering algorithms to characterize the target class. Unfortunately, stability criteria cannot be used for one-class classification. A stable clustering means that results by different methods give identical results (up to permutation of the cluster labels). In one-class classification just the boundary of the dataset is of interest, and the stability of the clusters as such is irrelevant.

In figure 2.3 the difference between the results of the stability and the consistency criterion is shown. The target data consists of two square uniform distributions, containing 100 objects. On this data a  $k$ -means clustering is trained, where the  $k$  is optimized to maximize the stability as presented in [6]. The most stable clustering solution is  $k = 2$  clusters, shown by the dashed circles. For the OC classifier, the square boundaries can be described much better by 4 to 6 prototypes per cluster. Again a  $k$ -means clustering is trained (with  $\nu = 0.1$ ), but now  $k$  is maximized under the constraint of a consistent data description. Here  $k = 11$  is optimal and the resulting decision boundaries are shown by the thick solid lines.

### 3. Experiments

To evaluate the optimization of the hyper-parameters, example outliers should be available. Therefore a standard multiclass dataset, the Concordia handwritten digits dataset [4] (400 training objects in  $16 \times 16 = 256$  dimensional feature space), will be transformed into a one-class dataset. Each single class of the dataset will be used the target class once, and the nine other classes will be considered outlier objects. During training of the OC classifier, only the target objects will be used. The consistency procedure contains two parameters to set. The number of cross-validations is not very critical. The range of hyper-parameters  $K_i$  is im-

portant, but is easily be defined for each OC method: for the clustering methods,  $k$  ranges from 1 to the number of training objects, for the methods using a scale parameter,  $\sigma$  ranges from the largest distance in the dataset to the smallest nearest neighbor distance.

class	true AUC	art. outliers	consistency
0	20 (99.1)	4 (98.3)	6 (98.4)
1	17 (99.8)	3 (98.1)	5 (98.3)
2	20 (93.0)	5 (90.4)	5 (91.2)
3	16 (96.0)	3 (92.5)	6 (92.5)
4	20 (94.5)	3 (93.6)	6 (94.4)
5	18 (95.3)	3 (91.7)	5 (93.6)
6	16 (98.5)	3 (96.8)	7 (97.9)
7	16 (95.2)	3 (94.0)	5 (94.5)
8	15 (89.0)	4 (88.3)	5 (86.6)
9	19 (90.6)	2 (83.1)	4 (89.0)

**Table 1. Results for the Mixture of Gaussians on the Concordia digit classes. The left column shows the optimized  $k$  (on a validation set), the middle using artificial outlier objects, the right the consistency on the target class. Values in brackets indicate the AUC ( $\times 100$ ).**

In table 1 the optimal number of clusters  $k$  is estimated using three methods. The first uses a validation set containing 'real' outliers (the other digits), the second uses a validation set using artificially created outliers [10], and the third uses the consistency criterion. The values in brackets show the AUC-errors on the 'real' outliers. To minimize the error on the artificial outliers, very large numbers of objects have to be created ( $5 \cdot 10^6$ ) to have a relatively reliable estimate. Still the variance of the AUC is relatively high (around 1.0). There is a substantial difference in AUC between the first method and the other two. The 'real' outliers appear to be close to the target objects, thus forcing a higher complexity classifier. Although this increases the error on the target class, this is balanced by a decrease in error on the outlier objects. When the outliers are uniformly distributed, on the other hand, a low complexity solution suffices. The consistency method slightly overestimates the complexity, but without significant decrease in performance.

## 4. Conclusions

This paper shows a automatic optimization procedure for selecting hyper parameters for one-class classifiers. It is shown that, when the error on the target class is kept constant and the classifier stays consistent, increasing the complexity of the classifier will yield a lower error. In practice

the complexity of the classifier cannot be increased without overfitting on the target class. It is therefore proposed to increase the complexity until the error on the target class exceeds the stochastic variation due to sampling.

This method gives very satisfactory results, under the assumption that the outlier objects are uniformly distributed in the input space. When the outliers are very close to the target class, the consistency algorithm will produce classifiers which are not complex enough. In that case, it will be beneficial to the use these outliers in the training of a supervised classifier.

**Acknowledgments** This research was supported through a European Community Marie Curie Fellowship. I would like to thank Klaus Müller for the brainstorming and the discussions, and Julian Laub for the relabeling software.

## References

- [1] A. Ben-Hur, A. Elisseeff, and I. Guyon. A stability based method for discovering structure in clustered data. In *Pacific symposium on biocomputing*, 2002.
- [2] A. Bradley. The use of the area under the ROC curve in the evaluation of machine learning algorithms. *Pattern Recognition*, 30(7):1145–1159, 1997.
- [3] C. Campbell and K. Bennett. A linear programming approach to novelty detection. In *Neural Information Processing Systems*, pages 395–401, 2000.
- [4] S.-B. Cho. Neural-network classifiers for recognizing totally unconstrained handwritten numerals. *IEEE Transactions on Neural Networks*, 8(1), 1997.
- [5] S. Geman, E. Bienenstock, and R. Doursat. Neural networks and the bias/variance dilemma. *Neural Computation*, 4(1):1–58, 1992.
- [6] T. Lange, M. Braun, V. Roth, and J. Buhmann. Stability-based model selection. In S. Becker, S. Thrun, and K. Obermayer, editors, *Advances in Neural Information Processing Systems*, volume 15. MIT Press: Cambridge, MA, 2003.
- [7] C. Metz. Basic principles of ROC analysis. *Seminars in Nuclear Medicine*, VIII(4), October 1978.
- [8] M. Moya, M. Koch, and L. Hostetler. One-class classifier networks for target recognition applications. In *Proceedings world congress on neural networks*, pages 797–801, Portland, OR, 1993. International Neural Network Society.
- [9] W. Polonik. Minimum volume sets and generalized quantile processes. *Stochastic processes and their applications*, 69:1–24, 1997.
- [10] D. Tax. *One-class classification*. PhD thesis, Delft University of Technology, <http://www.ph.tn.tudelft.nl/~davidt/thesis.pdf>, June 2001.
- [11] A. Ypma and P. Pajunen. Rotating machine vibration analysis with second-order independent component analysis. In *Proceedings of the First International Workshop on Independent Component Analysis and Signal Separation, ICA'99*, pages 37 – 42, January 1999.