An experimental comparison of cross-validation techniques for estimating the area under the ROC curve

Antti Airola[a,b,*], Tapio Pahikkala[a,b], Willem Waegeman[c], Bernard De Baets[c], Tapio Salakoski[a,b]

[a] *Department of Information Technology, 20014, University of Turku, Finland*
[b] *Turku Centre for Computer Science (TUCS), Joukahaisenkatu 3-5 B, 20520 Turku, Finland*
[c] *KERMIT, Department of Applied Mathematics, Biometrics and Process Control, Coupure links 653, Ghent University, Belgium*

Abstract

Reliable estimation of the classification performance of inferred predictive models is difficult when working with small data sets. Cross-validation is in this case a typical strategy for estimating the performance. However, many standard approaches to cross-validation suffer from extensive bias or variance when the area under the ROC curve (AUC) is used as performance measure. This issue is explored through an extensive simulation study. Leave-pair-out cross-validation is proposed for conditional AUC-estimation, as it is almost unbiased, and its deviation variance is as low as that of the best alternative approaches. When using regularized least-squares based learners, efficient algorithms exist for calculating the leave-pair-out cross-validation estimate.

Keywords: area under the ROC curve, classifier performance estimation, conditional AUC estimation, cross-validation, leave-pair-out cross-validation

*Corresponding author. Tel.: +358 2 3338565; fax: +358 2 2410154
Email addresses: antti.airola@utu.fi (Antti Airola), tapio.pahikkala@utu.fi (Tapio Pahikkala), willem.waegeman@ugent.be (Willem Waegeman), bernard.debaets@ugent.be (Bernard De Baets), tapio.salakoski@utu.fi (Tapio Salakoski)

1. Introduction

The area under the ROC curve (AUC) (Hanley and McNeil, 1982) is a ranking-based measure of classification performance, which has gained substantial popularity in the machine learning community during recent years (Bradley, 1997; Fawcett and Flach, 2005; Huang and Ling, 2005; Provost et al., 1998; Waegeman et al., 2008; Vanderlooy and Hüllermeier, 2008). Its value can be interpreted as the probability that a classifier is able to distinguish a randomly chosen positive instance from a randomly chosen negative instance. In contrast to many alternative performance measures, the AUC is invariant to relative class distributions, and class-specific error costs. These properties have prompted the use of the AUC measure in medical decision making (Swets, 1988), microarray studies (Baker and Kramer, 2006; Gevaert et al., 2006), natural language processing (Pahikkala et al., 2009a), and in the evaluation of information extraction systems (Airola et al., 2008; Miwa et al., 2009), to name a few examples.

In the above domains, the available data often exhibits properties that make it challenging to reliably evaluate the quality of inferred predictive models. For example, it is typical for genomic studies to produce data containing thousands of features, measured from a small sample, possibly containing only tens of instances. Further, the relative distribution of the classes to be predicted is often highly imbalanced and their discriminability can be quite low.

When setting aside data for parameter estimation and validation of results cannot be afforded, cross-validation (CV) is typically used. However, many commonly used CV schemes suffer from substantial negative bias, when considering AUC in the small-sample setting. Of concern is also the variance in the quality of estimates.

In this work we explore the suitability of different types of CV estimators for measuring the conditional expected AUC performance of a classifier. In Section 2 we introduce the concept of conditional expected AUC, and discuss related work. In Section 3 we consider the pooled and averaged CV approaches, discuss their strengths and weaknesses, and introduce leave-pair-out CV. Next, in Section 4 we discuss the computational costs of different CV strategies thereby providing references to efficient methods. Finally, in Section 5 we present our main contribution, an extensive simulation study comparing the bias and variance of different CV strategies for conditional AUC estimation. We conclude in Section 6.

2. Preliminaries

Let D be a probability distribution over a sample space $\mathcal{Z} = \mathcal{X} \times \mathcal{Y}$, where the input space $\mathcal{X}$ is a set and the output space $\mathcal{Y} = \{-1, 1\}$. An instance $z = (x, y) \in \mathcal{Z}$ is thus a pair consisting of an input and an associated label, which describes whether the instance belongs to the positive or to the negative class. The conditional distribution of an input from $\mathcal{X}$, given that it belongs to the positive class is denoted by D_+, and given that it belongs to the negative class by D_-. Further, let the sequence $Z = ((x_1, y_1), \ldots, (x_m, y_m)) \in \mathcal{Z}^m$ drawn i.i.d. from D be a training set of m training instances, with $X = (x_1, \ldots, x_m) \in \mathcal{X}^m$ denoting the inputs and $Y = (y_1, \ldots, y_m) \in \mathcal{Y}^m$ the labels in the training set.

Now let us consider a prediction function $f_Z : \mathcal{X} \to \mathbb{R}$ returned by a learning algorithm based on a fixed training set Z. The values taken by f_Z on a set of inputs can be used to produce a ranking where inputs receiving the highest predicted values are considered as being the most likely to belong to the positive class. Alternatively, a decision threshold t can be chosen, so that for a new input x the positive class is predicted if $f_Z(x) > t$, and the negative class otherwise.

We are interested in the generalization performance of this function, that is, how well it will predict on unseen future data. The generalization performance of f_Z can be measured by its expected AUC $A(f_Z)$, sometimes also known as expected ranking accuracy (Agarwal et al., 2005), over all possible positive-negative instance pairs, that is

$$A(f_Z) = \mathrm{E}_{x_+ \sim D_+, x_- \sim D_-}[H(f_Z(x_+) - f_Z(x_-))],$$

where H is the Heaviside step function defined as

$$H(a) = \begin{cases} 1, & \text{if } a > 0 \\ 1/2, & \text{if } a = 0 \\ 0, & \text{if } a < 0 \end{cases}.$$

We call this measure the *conditional expected AUC* of the prediction function, as it is conditioned on a fixed training set Z.

In practice we almost never can directly access the probability distribution D to calculate A, but are rather limited to using some estimate $\hat{A}$ instead, such as one obtained from CV. To measure the quality of a conditional expected AUC estimator, we consider the deviation $B(Z) = \hat{A}(f_Z) - A(f_Z)$,

which measures the difference between the estimated and true conditional expected AUC of a prediction function (Braga-Neto and Dougherty, 2004).

In this work we consider the expected value $\mathrm{E}_{Z\sim D^m}[B(Z)]$ of the deviation distribution as a measure of the biasedness of the estimator. Further, we also consider the variance $\mathrm{Var}_{Z\sim D^m}[B(Z)]$ of the deviation distribution, which can be considered as a measure of the reliability of individual estimates. Preferably, an estimator would have both close to zero deviation mean and variance. The effects of bias and variance are sometimes combined by considering the mean squared error $\mathrm{E}[(B(Z))^2] = \mathrm{E}[B(Z)]^2 + \mathrm{Var}[B(Z)]$ or the root mean squared error.

In certain settings, instead of considering the quality of a single prediction function f_Z inferred from a fixed training set, one may need to consider the expectation taken over all possible training sets of size m. The *unconditional expected AUC* can be defined as

$$\mathrm{E}_{Z\sim D^m}[A(f_Z)]\,.$$

As discussed for example by Dietterich (1998), Schiavo and Hand (2000) and Hastie et al. (2009), these two measures correspond to two different questions of interest. The conditional expected performance corresponds to the question how well we expect that a prediction function inferred from a given training set will generalize to future instances. The question arises whenever a researcher is using a certain data set and wants to know how well a predictive model inferred from that particular data set will do on future instances. The unconditional expected performance measures the quality of the learning algorithm itself, that is, how well on average will a prediction function returned by the algorithm of interest based on a training set of a given size generalize to new data.

More often, machine learning related articles concentrate on the unconditional performance, as the goal usually is to measure the quality of learning algorithms, where training data is treated as a random variable. Among results found in this setting, we mention those of Kohavi (1995), which suggest that the use of 10-fold CV for unconditional error rate estimation achieves a good balance in trade-off between bias and variance of the estimate in this setting.

It has been common in machine learning and statistical literature to cast the problem of assessing the quality of a conditional performance estimator as that of measuring its ability to measure unconditional performance (see Krzanowski and Hand (1997); Dietterich (1998); Schiavo and

Hand (2000) for discussion). However, this approach can be considered problematic as the variance $\mathrm{Var}_{Z\sim D^m}[\hat{A}(f_Z)]$ of the unconditional performance estimate does not directly inform us about its quality as a conditional performance estimator. Indeed, if the performance of the inferred predictive models varies significantly when different training sets are sampled, it is to be expected that estimates of a high quality conditional performance estimator would vary accordingly. Rather, what is of interest in this setting is the variance in deviation. From bias point of view there is no difference in whether one considers the unconditional or conditional AUC estimation. The expected value of the deviation can be re-written as $\mathrm{E}_{\mathrm{Z}\sim\mathrm{D}^\mathrm{m}}[\hat{A}(f_Z) - A(f_Z)] = \mathrm{E}_{\mathrm{Z}\sim\mathrm{D}^\mathrm{m}}[\hat{A}(f_Z)] - \mathrm{E}_{\mathrm{Z}\sim\mathrm{D}^\mathrm{m}}[A(f_Z)]$, which is the same as the expected difference between estimated and true unconditional AUC. Thus, an unconditional AUC estimator with low bias will also have low expected deviance.

In light of these considerations, several simulation studies have been undertaken during recent years to assess the relative quality of different conditional error estimators (Krzanowski and Hand, 1997; Braga-Neto and Dougherty, 2004; Hastie et al., 2009; Kim, 2009). Braga-Neto and Dougherty (2004) questioned the usefulness of CV estimates due to large variance observed in their experiments, unlike Kim (2009) who found CV more reliable. Hastie et al. (2009) noted the overall difficulty of conditional prediction error estimation. It is questionable to what extent the results observed for conditional error rate estimation can be generalized to conditional expected AUC estimation. Due to the differing nature of these two performance measures, their behavior can in certain conditions substantially differ (Cortes and Mohri, 2004).

There have not been many empirical studies about AUC estimation with CV. The simulation study of Parker et al. (2007) considered unconditional AUC estimation mainly on small low-dimensional data sets, and found that pooling-based CV techniques often had significant negative bias in this setting. Further experimental results about the bias of pooled CV techniques were presented by Forman and Scholz (2010). Recently, Hanczar et al. (2010) have studied conditional expected AUC estimation, demonstrating that in the small sample setting it is very difficult to accurately estimate the AUC performance. Preliminary results from the study presented in this paper were presented in (Airola et al., 2009).

3. Cross-validation

The AUC measure can be calculated using the following formula, also called the Wilcoxon-Mann-Whitney statistic:

$$\hat{A}(S, f_Z) = \frac{1}{|S_+||S_-|} \sum_{x_i \in S_+} \sum_{x_j \in S_-} H(f_Z(x_i) - f_Z(x_j)),$$

where S is a sequence of instances, and $S_+ \subset S$ and $S_- \subset S$ denote the positive and negative instances in S, respectively (Cortes and Mohri, 2004).

Perhaps the simplest performance estimator one can use is the resubstitution estimate $\hat{A}(Z, f_Z)$, that is, performance on the training set. It is a well-known fact that resubstitution often leads to highly overoptimistic evaluation of performance, though Braga-Neto and Dougherty (2004) note that in some very specific settings resubstitution can provide reasonable performance estimates with low variance. We do not further consider the resubstitution estimate in our study, as in our simulations the estimate typically yields an AUC of 1 whenever the number of dimensions exceeds the number of instances, even if there is no signal present.

When there is a large amount of data available, a standard approach is to set aside a test set T that is not used for learning the model, and use $\hat{A}(T, f_Z)$ to estimate the expected AUC of the learned model. This approach, however, cannot be used when there is not enough data available, such that a separate test set could be afforded.

In this paper, we consider CV, a performance evaluation technique commonly used in practice. Here, the data set is repeatedly partitioned into two non-overlapping parts, a training set and a hold-out set. For each partitioning, the hold-out set is used for testing while the remainder is used for training. The two most popular variants are *tenfold cross-validation* (10-fold CV), where the data is split into ten mutually disjoint folds, and *leave-one-out cross-validation* (LOO), where each training instance constitutes its own fold.

Stratification is commonly done to ensure that the hold-out sets share approximately the same class distributions. Further, for stratified CV on small data sets, Parker et al. (2007) have recently suggested a balancing strategy to ensure that all the training sets share the same number of positive and negative instances. When the sample size for a class is not a multiple of the number of folds, some folds will contain one extra instance from that class compared to the other folds. The balancing is done by randomly removing

members of overrepresented classes on each round of CV, so that all the training sets share the same number of instances from each class.

Two alternative strategies can be used to calculate the CV estimate over the folds, *pooling* and *averaging* (Bradley, 1997; Parker et al., 2007).

In pooling, the predictions made in each CV round are pooled into one set and one common AUC score is calculated from it. For LOO this is the only way to obtain the AUC score. The assumption made when using pooling is that classifiers produced on different CV rounds come from the same population. This assumption may make sense when using performance measures such as classification accuracy, but it is more dubious when computing AUC, since some of the positive-negative pairs are constructed using data instances from different folds. Indeed, Parker et al. (2007) show that this assumption is generally not valid for CV and can lead to large pessimistic biases. In their experiments with no-signal data sets, AUC values of less than 0.3 were observed instead of the expected 0.5. Similar results have been demonstrated by Forman and Scholz (2010).

An alternative approach, averaging, is to calculate the AUC score separately for each CV fold and average them to obtain one common performance estimate. Unlike with pooling, when using averaging predictions made for instances in different folds are never compared. While it was shown by Parker et al. (2007) that averaging eliminates the bias seen in pooling, we note it has one serious issue that pooling does not have. For N-fold CV, only a small subsample of all the possible positive-negative instance pairs present in the training set is considered, when calculating the AUC estimate. This can, as we will show in the following simulation study, lead to a high variance in the estimates when using small data sets. Also, as an extreme case, if there are more folds than observations for the minority class, some of the folds will not have instances from this class. For such folds, the AUC cannot be calculated.

To combine the strengths of the pooling and averaging strategies, we propose using leave-pair-out CV (LPO) for AUC estimation. In this approach each positive-negative instance pair constitutes its own fold, and the CV estimate is calculated by averaging over all these pairs. Thus, similarly to pooling, LPO makes maximal use of the available training data, while it also guarantees that only instances from the same round of CV are compared.

Previously, Krzanowski and Hand (1997) have introduced a leave-pair-out approach for error rate estimation as a means to achieve lower variance than that of the leave-one-out estimate. The reported improvements were however modest. Cortes et al. (2007) considered the LPO as an estimate

for pairwise ranking losses, and introduced an approximative algorithm for efficiently calculating LPO for learning algorithms that minimize the squared pairwise ranking loss. An exact version of the efficient LPO algorithm was derived recently by Pahikkala et al. (2008). For further discussion on efficient algorithmic implementations, see Section 4.

We formalize the CV approaches as follows. Let $I = \{1, \ldots, m\}$ denote the indices of the training instances. In CV, we have a set $\mathcal{U} = \{U_1, \ldots, U_N\}$ of hold-out sets, where $N \in \mathbb{N}$ and $U_i \subseteq I$.

By $f_{\overline{U}}$ we denote the prediction functions inferred from all the training instances except those indexed by U. Further, we use $U_+ \subseteq U$ to denote the set of positive, and $U_- \subseteq U$ the set of negative instances belonging to U.

The pooling approach can be written as

$$\frac{1}{C} \sum_{U,U' \in \mathcal{U}} \sum_{i \in U_+, j \in U'_-} H(f_{\overline{U}}(x_i) - f_{\overline{U'}}(x_j)),$$

where the normalizer $C = \sum_{U,U' \in \mathcal{U}} |U_+||U'_-|$ is the number of positive-negative pairs encountered in the summation. For pooling we assume that the folds are defined such that $U \cap U' = \emptyset$ for all $U, U' \in \mathcal{U}$.

The averaging estimate can be written as

$$\frac{1}{C} \sum_{U \in \mathcal{U}} \sum_{i \in U_+, j \in U_-} H(f_{\overline{U}}(x_i) - f_{\overline{U}}(x_j)),$$

where $C = \sum_{U \in \mathcal{U}} |U_+||U_-|$.

The AUC performance is calculated with LPO, which is a special case of averaging, as

$$\frac{1}{|I_+||I_-|} \sum_{i \in I_+} \sum_{j \in I_-} H(f_{\overline{\{i,j\}}}(x_i) - f_{\overline{\{i,j\}}}(x_j)),$$

where $f_{\overline{\{i,j\}}}$ denotes a classifier trained without the i-th and j-th training instance, and $I_+ \subset I$ and $I_- \subset I$ denote the indices of the positive and negative instances in the training set Z, respectively.

Next, we show that LPO is an almost unbiased estimator of unconditional expected AUC. The proof is analogous to the proof of Cortes et al. (2007) about the same property of LPO as an estimate for unconditional expected pairwise ranking error. As discussed in Section 2, the bias in unconditional AUC estimation and the expected deviance in conditional AUC estimation

coincide. This means that a method that produces close to unbiased estimates of the unconditional AUC also has the same property when estimating the conditional AUC.

Let $D^{p,n}$ denote the distribution of instance sequences containing p positive and n negative training instances. As before, we assume that the sets of p positive and n negative instances consist of an i.i.d. sample from their respective distributions. Let I_+ and I_- denote the indices of the positive and negative instances belonging to a training sequence sampled from $D^{p,n}$. Then, by definition $|I_+| = p$ and $|I_-| = n$. The expected value of LPO is

$$\begin{aligned}
& \mathrm{E}_{Z\sim D^{p,n}}\left[\frac{1}{pn}\sum_{i\in I_+}\sum_{j\in I_-} H\left(f_{\overline{\{i,j\}}}(x_i) - f_{\overline{\{i,j\}}}(x_j)\right)\right] \\
= \; & \mathrm{E}_{Z\sim D^{p,n}, i\in I_+, j\in I_-}\left[H\left(f_{\overline{\{i,j\}}}(x_i) - f_{\overline{\{i,j\}}}(x_j)\right)\right] \\
= \; & \mathrm{E}_{Z\sim D^{p-1,n-1}}\left[\mathrm{E}_{x^+\sim D_+, x^-\sim D_-} H\left(f_Z(x^+) - f_Z(x^-)\right)\right] \\
= \; & \mathrm{E}_{Z\sim D^{p-1,n-1}}\left[A(f_Z)\right],
\end{aligned}$$

which is the unconditional expected AUC, assuming a training set with $p-1$ positive and $n-1$ negative instances. By $i \in I_+$ and $j \in I_-$ we denote the selection of arbitrary members of the corresponding index sets. The first equality follows from the linearity of the expected value, and the fact that both the positive and negative instances are sampled i.i.d. (from their respective distributions). The final equality follows simply from the definition of unconditional expected AUC. Thus, LPO is an almost unbiased estimator of unconditional expected AUC. LOO is known to have an analogous property for univariate performance measures, such as the classification error rate or the mean squared error (Luntz and Brailovsky, 1969). From this perspective, LPO can be considered as an extension of LOO to pairwise performance measures.

4. Computational Efficiency

The computational cost can be seen as a limitation of CV techniques in general, and in particular for the LOO and LPO methods. For a training set of m instances, a straightforward implementation of LOO requires training the learning algorithm m times, while for LPO the required number of training rounds is of the order $O(m^2)$. Even though these computational costs

may be affordable on small training sets, they can become a limiting factor as the training set size increases.

Regularized least-squares (RLS) (Rifkin, 2002), also known as the least-squares SVM (Suykens and Vandewalle, 1999) or kernel ridge regression (Hoerl and Kennard, 1970; Saunders et al., 1998), is a state-of-the-art classification method, which we consider in our experiments. RLS is a kernel method based on regularized risk minimization, where a least-squares approximation of the classification error is minimized together with a quadratic regularizer. The method is closely related to the SVM classifier (Vapnik, 1995), one of the most successful classification methods in machine learning, the difference being that the SVM optimizes the hinge loss rather than the least-squares loss. The similar behavior of the two methods has been demonstrated both theoretically and empirically in the literature (see e.g. (Suykens and Vandewalle, 1999; Fung and Mangasarian, 2001; Rifkin, 2002; Zhang and Peng, 2004)). An analogous relationship holds between the RankRLS (Pahikkala et al., 2007; Cortes et al., 2007; Pahikkala et al., 2009b) and the RankSVM (Herbrich et al., 1999) methods, which extend the RLS and SVM respectively to minimize the pairwise least-squares and pairwise hinge losses, which can be considered convex approximations for AUC (for further discussion and comparisons, see (Pahikkala et al., 2009b)).

A main benefit in using RLS based learning algorithms is that CV procedures can be implemented very efficiently without having to sacrifice exactness by approximating. The existence of an efficient LOO procedure is a classical result (Vapnik, 1979). This result has been recently extended to repeated hold-out and CV with arbitrary sized, possibly overlapping holdout sets by Pahikkala et al. (2006) and An et al. (2007), independently of each other. Analogously, for RankRLS, an efficient and exact leave-pair-out procedure has been derived by Pahikkala et al. (2008, 2009b). The CV algorithms, which are based on matrix calculus, utilize the fact that the results of the most expensive matrix decomposition operations required in training can be re-used in the repeated hold-out procedure. The computational complexities of the efficient LOO and LPO methods both equal $O(m^2)$, assuming $O(m^3)$-complexity operations, necessary for training the RLS or RankRLS methods, have been previously computed. Since these algorithms combine state-of-the-art classification performance with efficient computational shortcuts for CV, they are a natural fit to settings where CV is to be used.

method	type	abbreviation
leave-pair-out	averaged	LPO
leave-one-out	pooled	LOO
balanced leave-one-out	pooled	BLOO
averaged 5-fold	averaged	A 5-F
averaged 10-fold	averaged	A 10-F
pooled 10-fold	pooled	P 10-F

Table 1: Cross-validation strategies considered in the simulation study

5. Experimental Study

In the simulation study, we measure the mean and variance of the deviation distribution of several CV estimators. The considered approaches are summarized in Table 1. We consider three pooled strategies, LOO, balanced LOO (BLOO) (Parker et al., 2007) and pooled 10-fold CV, as well as the averaged 5-fold CV, averaged 10-fold CV and LPO. Stratification is used where possible.

5.1. Basic setup

Two learning algorithms are used in the experiments, RLS and RankRLS. RLS minimizes an approximation of the classification error rate, like most machine learning algorithms. RankRLS in contrast optimizes an approximation of the AUC. Since the RLS algorithm behaves very similarly to the SVM, and RankRLS to the RankSVM algorithm, the results of the simulations should be also quite directly applicable to these algorithms. To test this assumption, we perform an additional experiment with SVMs (see Section 5.4). We use the linear kernel in all the experiments.

The considered algorithms have a regularization parameter, which controls the trade-off between model complexity and fit to the training data. We noticed the considered simulation setting to be fairly robust with respect to the choice of this parameter. In preliminary experiments conducted on both low and high-dimensional, and signal and non-signal data, shifting the parameter value by several orders of magnitude did not greatly affect the resulting deviations. Therefore, to simplify the experimental setup, we fixed the regularization parameter to the value of 1. In the absence of any prior knowledge about a suitable value for the parameter, its value can be selected by combining grid search with CV. However, using the CV estimates both for

parameter selection and estimation of generalization performance at the same time can be expected to result in overoptimistic evaluation of performance.

The prediction function returned by the RLS method with linear kernel can be written as $f(x) = \mathbf{w}^{\mathrm{T}}\mathbf{x} + b$, where $\mathbf{w} \in \mathbb{R}^n$ contains the coefficients of the linear model, $\mathbf{x} \in \mathbb{R}^n$ contains the feature representation of an instance, and b is an intercept term. Following the approach discussed for example in (Zhang, 2002) we introduce the intercept term by adding a feature with constant value 1 to each instance (note that for $\hat{\mathbf{x}} = [\mathbf{x}^{\mathrm{T}}, 1]^{\mathrm{T}}$, $\hat{\mathbf{w}} = [\mathbf{w}^{\mathrm{T}}, b]^{\mathrm{T}}$, $f(x) = \hat{\mathbf{w}}^{\mathrm{T}}\hat{\mathbf{x}}$). The intercept considerably increases the representational power of the RLS classifiers if the dimensionality of the data is low. However, adding an intercept feature has no effect to the RankRLS optimization problem, as the minimizer of RankRLS loss will always set the corresponding coefficient to zero.

The used learning and CV algorithms are from the RLScore software package, available at `http://www.tucs.fi/rlscore`. The software is based on the matrix libraries of NumPy and SciPy. Data generation was implemented using the R statistical environment.

Our setting is similar to that of Parker et al. (2007), who compared the bias of pooling and averaging approaches mainly on low-dimensional data. We consider synthetic data, as this allows estimating the conditional expected AUC of the inferred prediction functions. The training set size is 30 instances in the simulations, unless otherwise explicitly stated. All the results are calculated over 10000 repetitions, where in each repetition a new training set is sampled.

In the no-signal simulations, where no signal occurs in the data, instances from both classes are drawn from normal distributions with zero mean, unit variance and no covariance between the features. In the signal experiments, the means of a number of discriminating features are shifted to 0.5 for the positive, and to -0.5 for the negative class. The conditional expected AUC of a prediction function is for the no-signal experiments always equal to 0.5, as no model can do either better or worse than random in this setting, in terms of AUC. For the signal experiments, generated test sets with 10000 instances are used to estimate the conditional expected AUC of the inferred prediction functions. The test set estimates thus represent the true generalization performance $A(f_Z)$, to which the CV estimates $\hat{A}(f_Z)$ are compared to.

One question we explored in the simulations was whether the almost unbiased LPO empirically differs significantly in terms of bias from the other

estimators. We assess the significance of the difference between the deviation of the LPO estimate and the alternative estimates using the Wilcoxon signed-rank test (Wilcoxon, 1945), with $p = 0.05$, applying the conservative Bonferroni correction for multiple hypothesis testing.

Previously, a rather surprising result has been reported by Hastie et al. (2009), who observed in a simulation study in many cases negative correlations between conditional performance estimates and the true conditional performance. Thus, in their simulation, when sampling a more informative training set than on average, the CV estimates had a tendency to be lower than on average, and vice versa. The correlations were always calculated for a fixed learning method and fixed data distribution. We also calculated the correlations between the true conditional performance and the estimates for all experiments with signal data. In our simulations, the correlations were in all cases found to be positive. We note that our experimental setup is quite different from the one used by Hastie et al. (2009), and that the subject of their study was conditional error rate rather than conditional AUC.

We do not consider squared error separately, but note that its behavior is in these experiments dominated by the contribution made by the variance. This means that any conclusion drawn about the variability of the estimators, based on these simulation results, holds also for the squared error of the estimators. Results for the averaged N-fold CV strategies are provided only when at least N instances from both classes are present, as AUC can not be calculated for folds that have no instances from both classes. We also performed limited experiments with the averaged N-times repeated holdout method, in which the holdout sets are allowed to overlap. The method showed a much higher variance than any of the approaches considered in this study.

5.2. First simulation study

In the initial simulation study we studied the effect of the presence or absence of signal in the data and the relative class distributions of positive/negative pairs on the reliability of different AUC estimators. The relative distribution of positive instances was varied between 10% and 50% in steps of 10%. Results for the relative distributions between 60% and 90% were the same as those between 40% and 10%, as is to be expected. Therefore, they will not be separately considered in the following. Both low-dimensional data with 10, and high-dimensional data with 1000 features were considered. We considered both non-signal and signal data. For the signal data with

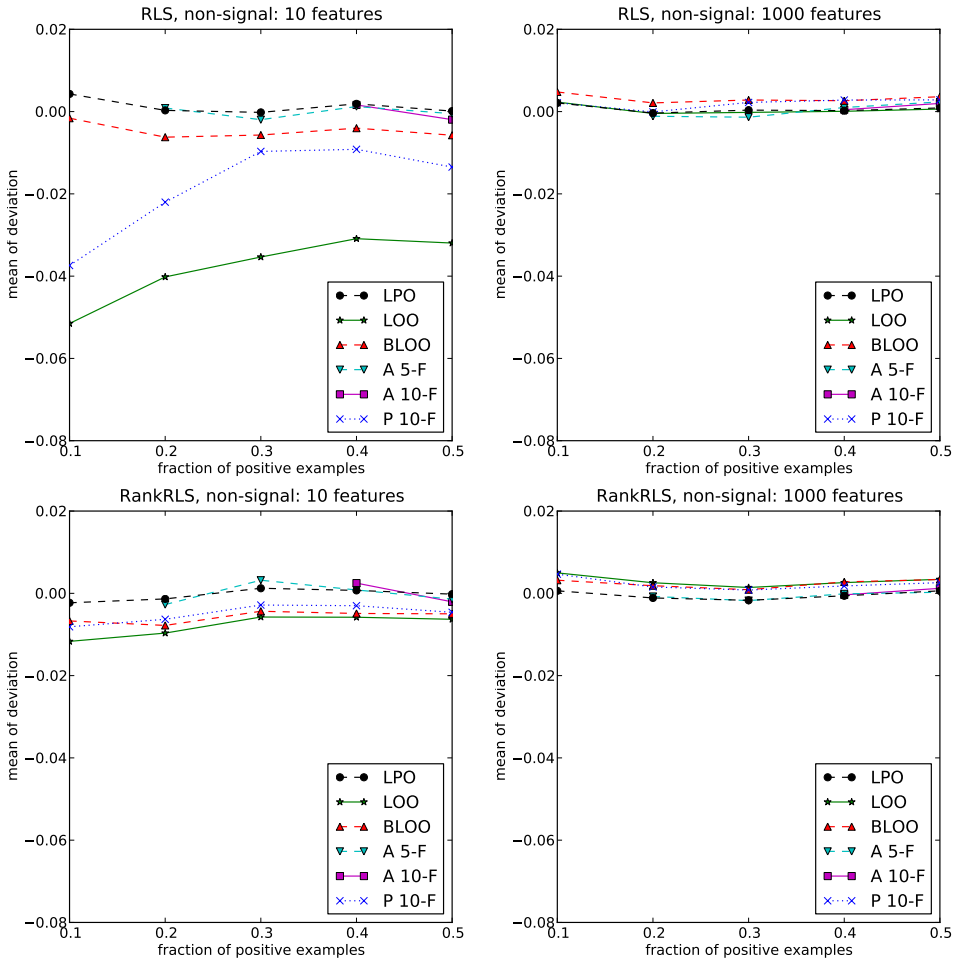

Figure 1: Deviation means with non-signal data.

10 features, 1 feature contains signal, and with 1000 features, 10 features contain signal.

Figure 1 displays the deviation results for non-signal data. When using the RLS algorithm on low-dimensional data, we observe a substantial negative bias for the pooled estimators, with BLOO being the least biased of them. The averaging strategies work better, with LPO showing significantly less bias than all of the pooled strategies. These results are consistent with those reported in (Parker et al., 2007). With RankRLS and low-dimensional data, the pessimistic bias of the pooled strategies is much smaller, but nonetheless significant differences compared to the LPO are observed. LPO and the

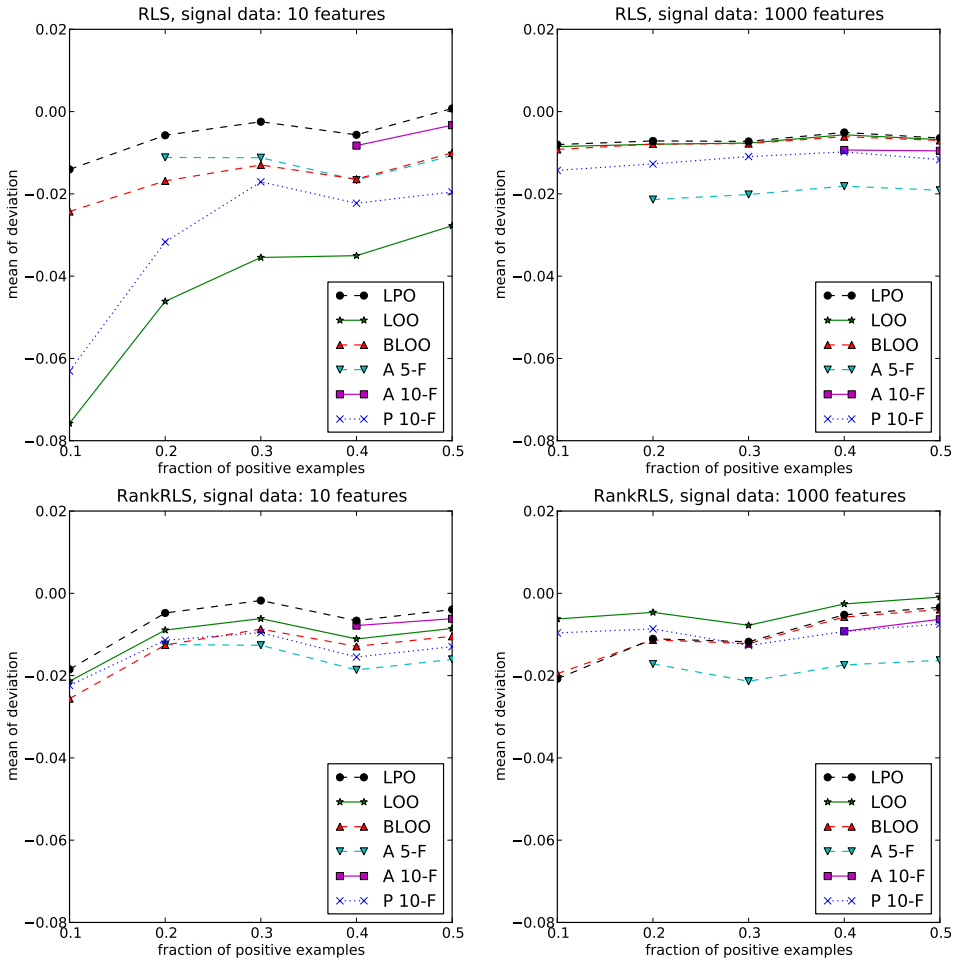

Figure 2: Deviation means with signal data.

other averaging strategies behave similarly. On high-dimensional data the pessimistic bias disappears.

Figure 2 displays the results for signal data. Compared to the non-signal results, the negative bias in pooled estimators is larger. Unlike with non-signal data, some negative bias is now present also in the averaged estimation strategies. This is to be expected, as there is an inherent negative bias in the CV procedure due to the fact that some instances are left unused on each round. This does not effect the non-signal simulation results as there is no information to be learned from the held out instances anyway. On signal results the effect is however clear. Comparing the averaged strategies

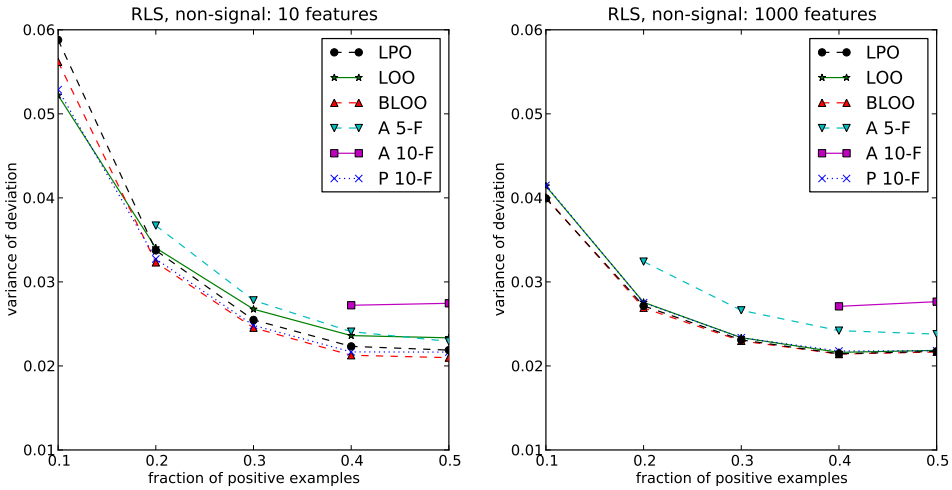

Figure 3: Variances of the deviation on non-signal data and RLS.

we see that the larger the folds, the greater the bias. LPO is less biased than averaged 10-Fold CV, which in turn is less biased than averaged 5-fold CV. Again for RankRLS the negative bias of the pooled estimators is smaller on the low-dimensional setting than the bias with RLS. In case of high-dimensional data, the excess pessimistic bias again disappears from the pooled estimates, making them competitive with the averaged strategies in that setting. In fact with RankRLS and high dimensional data, LOO provides significantly less biased estimates than LPO.

In Figure 3 we present the deviation variances for RLS on non-signal data. The results for signal data were essentially the same. The RankRLS results were also very similar with the exception that in the most imbalanced setting (only 10% of positive instances) BLOO and LPO always had higher variance than the LOO and pooled 10-fold CV. In all the experiments, the averaged N-fold CV strategies have a larger variance than the pooled strategies and LPO. The more imbalanced the relative class distributions, the higher the variance becomes.

To conclude, LPO seems to be often the least biased of the alternatives and it has a very similar variance as the pooled strategies. Averaged 10-fold CV is also very competitive in terms of bias. For averaged 5-fold CV, a large pessimistic bias appears in the signal experiment, as a fifth of the training data is held out of the already very small training set in each round. Both of the averaged N-fold CV strategies suffer from a large variance. Pooling leads

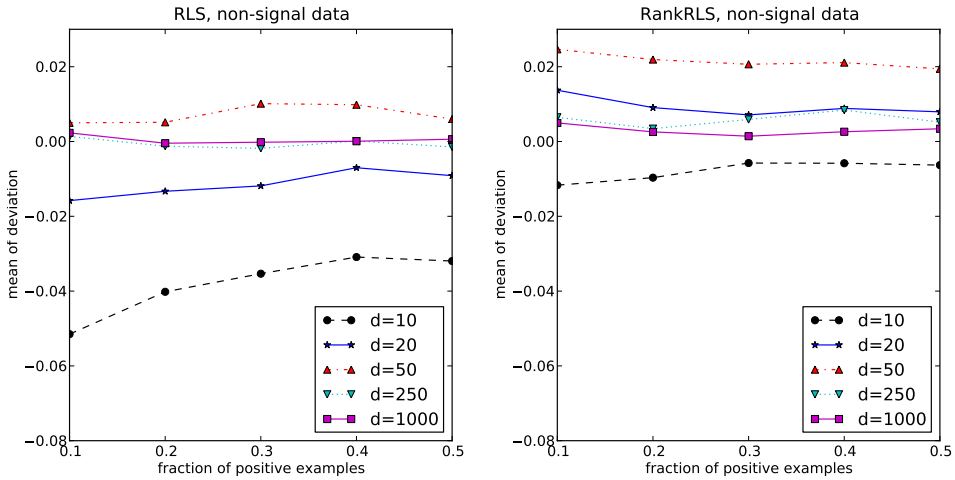

Figure 4: Mean deviation of LOO with varying feature space dimensionality.

to a lower variance, but can also result in a large bias, as seen especially with RLS on low-dimensional data. Balancing can help to reduce this bias, as demonstrated by the BLOO results. For RankRLS the pooled CV estimates were on low-dimensional data much less biased than for RLS.

5.3. Second simulation study

Based on the results of the initial study, we designed a further suite of experiments, to explore some of the issues encountered. As the dimensionality of the feature space was found to be one of the most important factors influencing the amount of bias in the estimators, we chose to explore this further. In addition, we studied the effect of varying the amount of signal in the data, and the sample size.

First, we consider results for non-signal data and varying feature space dimensionality. In this experiment, we run the non-signal experiment with 10, 20, 50, 250 and 1000 dimensional data. We noticed that the deviation variance was not greatly affected by changing the feature space dimensionality. Similarly to results presented in Figure 3, averaged 10-fold and 5-fold CV always had a higher variance than pooled strategies and LPO. Also, averaged estimators again did not show any systematic bias in this experiment. They are therefore not considered further here.

In Figure 4 we present the mean deviations for LOO, as it is representative of how the pooled estimators in general behaved. Increasing dimensionality

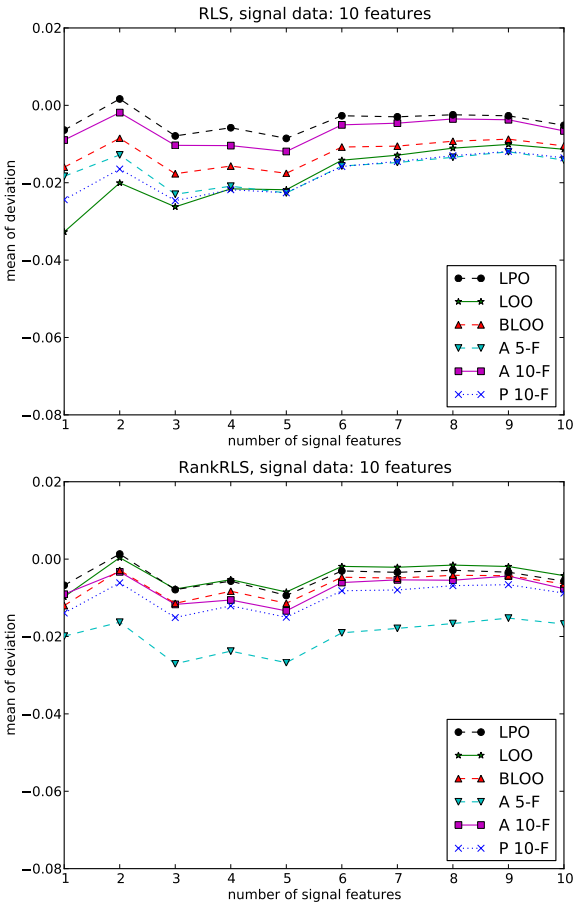

Figure 5: Mean deviations for signal experiments with 10 features

was found to lead to a positive bias in the pooled estimators. At 20 dimensions much of the negative bias has disappeared, and at 50 dimensions we can see a clear positive bias. At 250 dimensions the positive bias has mostly disappeared. To conclude, the pooled strategies show a clear bias in several different settings on non-signal data, though the magnitude and even the direction of the bias does change with feature space dimension.

Additional analysis about the behavior of LOO in low and high dimensions can be found in the Appendix. While the analysis does not fully explain the observed phenomena, it yields some insight into why LOO with RLS has such strong negative bias in low dimensions, while the effect disappears with high enough dimensionality.

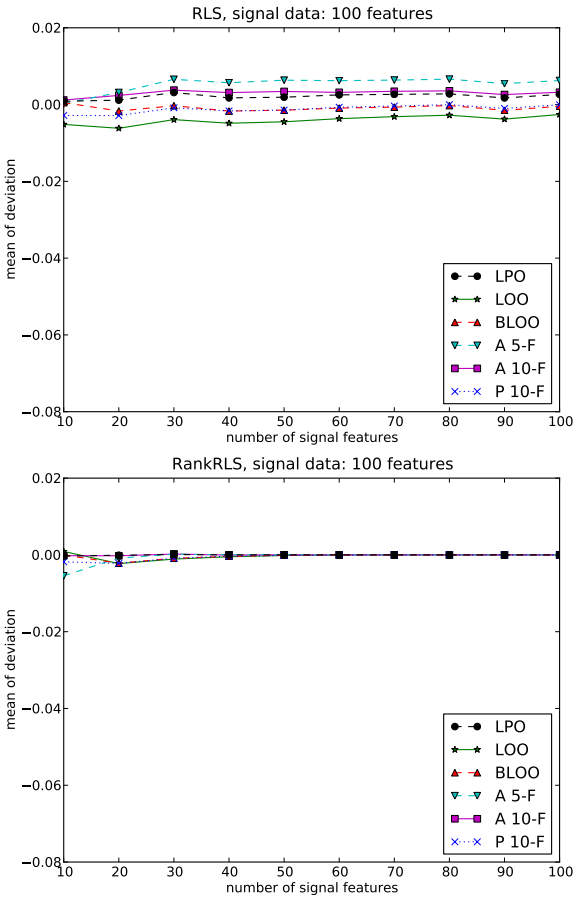

Figure 6: Mean deviations for signal experiments with 100 features

In the signal experiment we studied the behavior of the deviation bias and variance as a function of signal strength. In each experiment the dimensionality of the feature space is fixed, and signal strength is varied from 10% of the features being informative to all of them being informative. As the number of informative features grows, so does the true conditional expected AUC of the inferred models. The results are presented for the setting where the class distribution is balanced. We also run the experiments in an imbalanced setting where only 10% of the instances belonged to the positive class. The observed trends were similar, though the magnitude of the variance was about twice as high.

Figure 5 shows the deviation means for 10-dimensional data; we observe

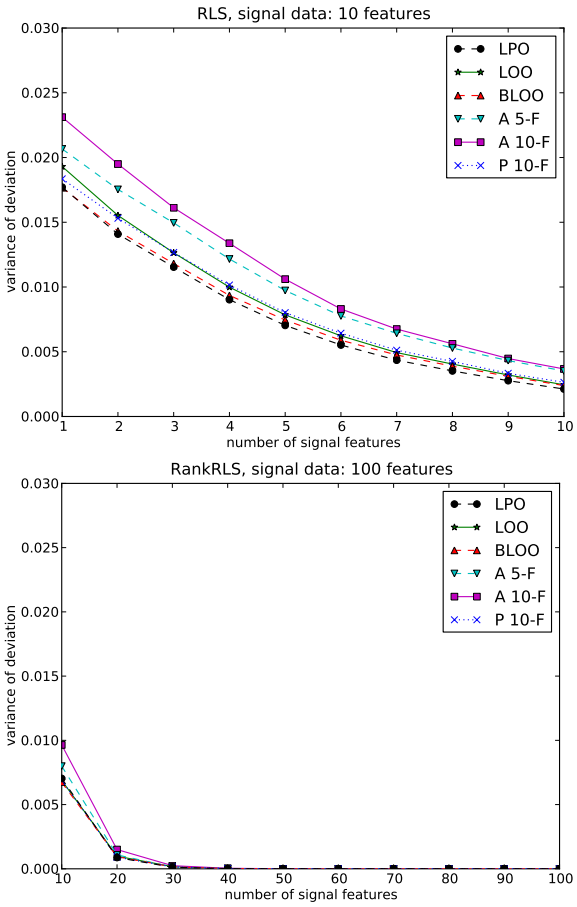

Figure 7: Deviation variance plotted against signal strength

again a pessimistic bias in the pooled estimators, though for LOO most of the bias disappears for RankRLS. As before, 5-fold CV also has a negative bias. With 100-dimensional data and RLS (see Figure 6), all the pooled estimators show some negative bias, and averaged fivefold has a slight positive bias. For RankRLS the signal in the data has now become strong enough that it can learn to separate the classes almost perfectly, achieving unconditional expected AUCs close to 1 from 30 signal features onwards. All of the estimators predict this correctly with their mean deviations reaching almost 0 at this point. The same could be achieved for RLS when the number of features was increased to 1000 and the number of signal features to 100.

Figure 7 plots the deviation variance against the amount of informative

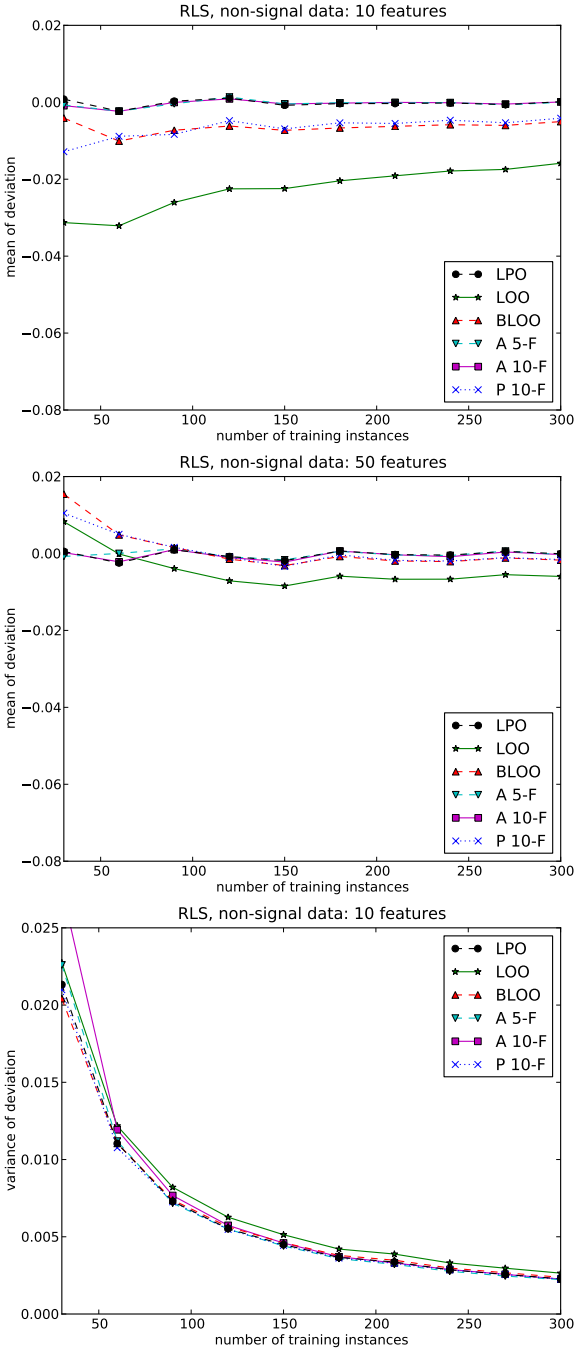

Figure 8: Sample size experiments for non-signal data and RLS with 10 and 50 features

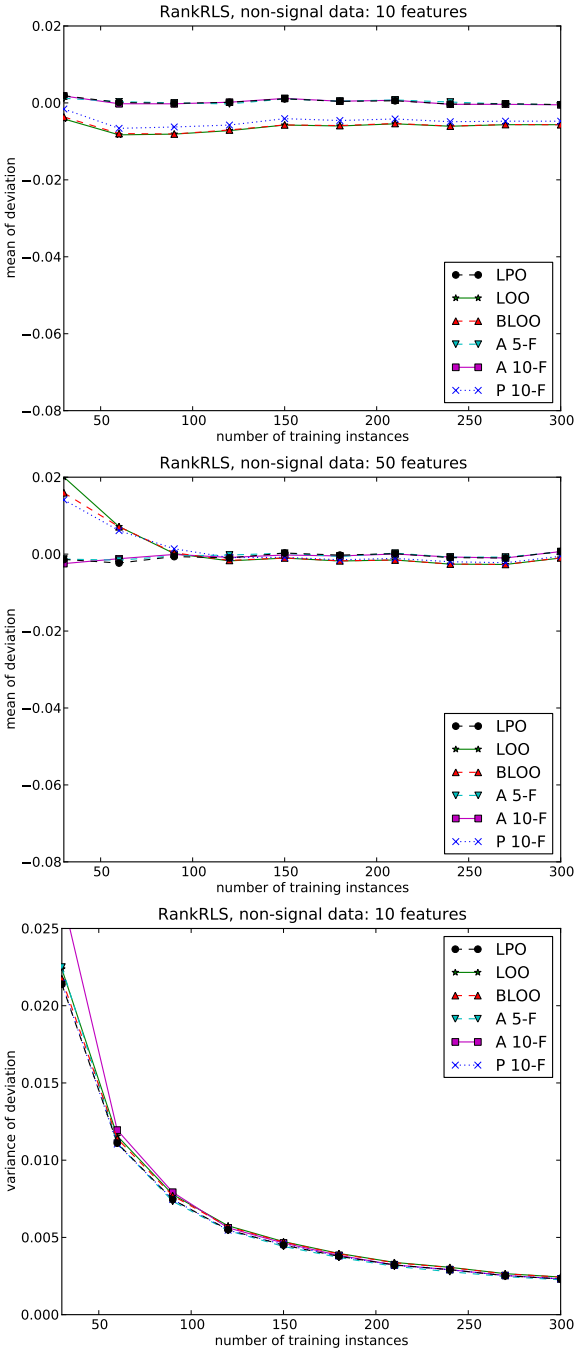

Figure 9: Sample size experiments for non-signal data and RankRLS with 10 and 50 features

features. For 10-dimensional data, the variances of RLS and RankRLS were essentially identical. It can be clearly seen from the plot that increasing the signal strength effects the variance; the stronger the signal the smaller the variance. In the lower picture we can see that for RankRLS on 100-dimensional data and strong enough signal, the deviation variance finally reaches 0. The same effect occurs for RLS in 1000 dimensions.

To conclude, while for non-signal data the pooled estimators were negatively biased in low-dimensional spaces, this bias was seen to shift to a positive one, and finally disappear, as the dimensionality of the feature space increased. For the averaged estimators neither a negative nor a positive bias was observed for non-signal data. In the signal experiment, increasing the strength of signal in the data did not remove the negative bias of pooled estimators encountered on 10-dimensional data. In high dimensions, with a strong enough signal, the negative bias disappeared first from averaged N-fold CV strategies, and finally from pooled CV strategies.

All experiments considered, LPO and averaged 10-fold CV are again the least biased. An increase in signal strength leads to a smaller variance in the estimates. However, the relative differences in variance always remain, with averaged N-fold CV strategies typically having a much larger deviation variance than the other CV estimators.

Finally, we consider the effect of the sample size. On non-signal data we incrementally increase the sample size from 30 instances to 300, and examine the effect on the deviation. Figure 8 summarizes the results on 10 and 50-dimensional data for RLS, and Figure 9 for RankRLS. For 10-dimensional data a small decrease in the magnitude of the pessimistic bias is seen as the sample size grows. Still, the bias is quite noticeable even for a sample size of 300, indicating that the phenomenon is not only limited to extremely small sample sizes. With 50-dimensional data we notice that, unlike the negative bias, the positive bias shown by the pooled estimators disappears as the sample size increases. This suggests that the positive bias of pooled estimators may appear only in certain quite specific circumstances. As shown for 10-dimensional data, the variance of the estimators becomes much smaller when the sample size grows. The same trend was observed in experiments on higher-dimensional data. Moreover, it is interesting that averaged N-fold CV estimators become competitive in terms of variance when the sample size grows.

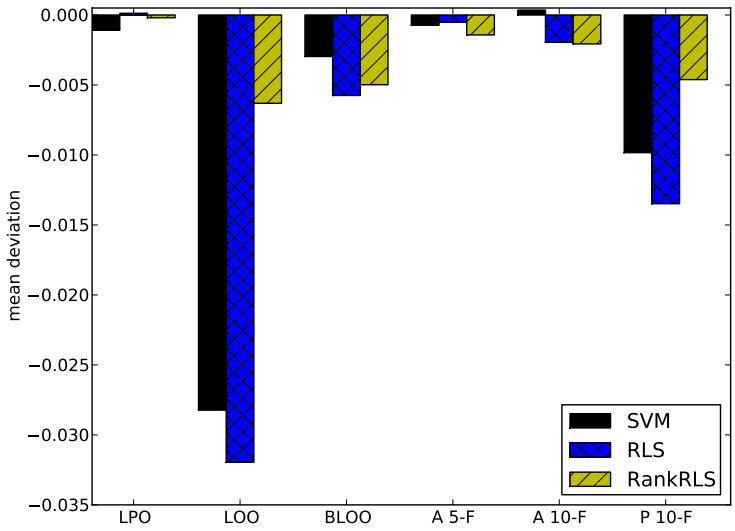

Figure 10: Mean deviations with SVM, RLS and RankRLS on 10-dimensional non-signal data with balanced class distribution.

5.4. SVM experiment

To demonstrate that effects similar to those observed in the previous experiments do appear also when using other learning methods than RLS or RankRLS. we repeat one of the experiments using the soft margin SVM classifier (Vapnik, 1995). We implement the SVM based on the BMRM training algorithm (Teo et al., 2010), using the CVXOPT open source optimizer for quadratic programming. We use the linear kernel and the value 1 for the regularization parameter, as was done in the previous experiments. Using 30 training instances, 10-features and a balanced data distribution, we measure the mean deviations of the different estimators for SVM. Due to computational costs, we limit the experiment to 2000 repetitions instead of the 10000 repetitions used in all the previous experiments. In Figure 10 are the mean deviations. The results are very close to those observed with RLS. The pooled estimators have a clear negative bias, with LOO being the most biased, while the averaged estimators have a close to zero deviation. Similarly to earlier experiments, averaged 10-fold had the highest deviation variance also for SVM. The main drawback of using SVMs is the lack of fast exact CV methods, which makes especially computing LOO, BLOO, and LPO computationally inefficient, as discussed in Section 4.

6. Conclusions

In this work we have considered the merits and drawbacks of different CV estimators for the conditional expected AUC. In line with earlier results of Parker et al. (2007), we observed a large negative bias in the pooled CV estimators on low-dimensional data. This bias was seen to persist even when the signal strength or the sample size was increased, suggesting that the pooled estimators can systematically fail in a wide variety of settings when using low-dimensional data. Increasing the feature space dimensionality caused a positive bias to appear on small data sets with no signal or a weak signal, but this phenomenon disappeared with an increased signal or an increased sample size. On high-dimensional data, the pooled estimators did not show substantial bias. Averaged strategies did not suffer from the same bias effect as the pooled ones. However, averaged N-fold estimates had larger deviation variance on small data sets than the pooled approaches. LPO performed significantly better than the pooled estimators in terms of bias in most of the experiments, and had competitive variance. The main difference between the RLS and RankRLS results was that the negative bias in pooled estimates on low dimensional data was much stronger with RLS than with RankRLS.

A natural question that arises is, to which extent can our results be generalized to other learning methods? The differences between the RLS and RankRLS results demonstrate that the underlying learning method does affect the behavior of the CV estimators. Still, some general learner-independent principles can be recognized behind the experimental results. The pooling approach can result in bias, as the predictions made by different models are combined together before calculating the AUC, whereas this is not the case for averaging. Pooled approaches and LPO allow making use of all the positive-negative pairs in the training set, which allows lower variance compared to averaged N-fold estimates. Finally, CV with large folds can lead to a negative bias with signal data, since a large part of the training set is not used for learning in each round.

LPO was observed to be the most robust of the considered approaches, as it provided in all our experiments close to unbiased performance estimates, and showed a variance that was competitive with that of the pooled estimators. Therefore, we suggest that LPO is the preferred method for conditional AUC estimation. However, due to its computational costs, it may not always be a practical choice when using learners that do not have fast algorithms for CV. Of the alternative approaches, we suggest that averaged N-fold es-

	Dimensionality	
Sample size	low	high
small (e.g. 30 instances)	LPO Balanced pooling	LPO Balanced pooling
medium (e.g. 300 instances)	LPO Averaging	LPO Averaging

Table 2: Recommendations about preferred cross-validation strategies in different settings

timates are the most reliable, as long as one has at least several hundreds of training instances available to achieve an acceptable level of variance. Pooled estimators may be useful on very small data sets due to their variance being lower than that of averaged N-fold estimates in this setting. However, their bias seems to be a serious problem especially on low-dimensional data. As previously recommended by Parker et al. (2007), balancing should be done when applying pooling, as it can reduce this bias. While in our experiments pooling did not show a strong bias in high dimensions, care should be exercised in generalizing this result. The basic problem of making comparisons between predictions made in different CV rounds remains, meaning that bias might appear also in high dimensions for other types of data distributions or learning methods than those considered in our simulations. We summarize our recommendations in Table 2.

We note the existence of particularly efficient algorithms for calculating the CV estimates for RLS and RankRLS. Possible future directions of study include exploring ways to reliably and efficiently assess the statistical significance of achieved results, for example by applying permutation tests. The efficient CV algorithms and learning methods used in this study are made publicly available under open source license as part of the RLScore machine learning software package, distributed at `http://www.tucs.fi/rlscore`.

Acknowledgments

This work has been supported by the Academy of Finland. W.W. was supported by a research visit grant from the Research Foundation - Flanders. We would like to thank CSC, the Finnish IT center for science, for providing us with extensive computational resources.

Appendix

Here, we analyze the effect the data dimensionality has on the amount of bias in pooled estimation. We restrict our considerations to LOO with non-signal data, because the largest negative bias was seen in this setting. The bias was strong with low-dimensional non-signal data, but it was seen to disappear with large enough dimensionality. Instead of analyzing RLS directly, we consider the Parzen window classifier, which is easier to analyze and is known to be an approximation of both RLS (Zhang et al., 2005) and SVM (Hastie et al., 2004). Here, we denote by $\langle \mathbf{x}_1, \mathbf{x}_2 \rangle$ the inner product between two vectors $\mathbf{x}_1$ and $\mathbf{x}_2$.

The prediction function for the linear Parzen window classifier is:

$$f_Z(\mathbf{x}) = \sum_{(\mathbf{x}_i, y_i) \in Z} y_i \langle \mathbf{x}_i, \mathbf{x} \rangle.$$

Let $\hat{\mathbf{x}}^+ = \mathbf{x}^+ + \mu$ and $\hat{\mathbf{x}}^- = \mathbf{x}^- + \mu$ be feature vectors of a positive and a negative training instance, where μ is the mean vector of the training set. Now let $z^+ = (\hat{\mathbf{x}}^+, 1) \in Z$ and $z^- = (\hat{\mathbf{x}}^-, -1) \in Z$. Further, let

$$\mathbf{a} = \sum_{(\hat{\mathbf{x}}_i, y_i) \in Z \setminus \{z^+, z^-\}} y_i \hat{\mathbf{x}}_i.$$

The difference of LOO hold-out predictions for a positive $\mathbf{x}^+$ and negative $\mathbf{x}^-$ training instance is:

$$\begin{aligned} e &= f_{Z \setminus \{z^+\}}(\hat{\mathbf{x}}^+) - f_{Z \setminus \{z^-\}}(\hat{\mathbf{x}}^-) \\ &= f_{Z \setminus \{z^+, z^-\}}(\hat{\mathbf{x}}^+) - \langle \hat{\mathbf{x}}^-, \hat{\mathbf{x}}^+ \rangle - (f_{Z \setminus \{z+, z-\}}(\hat{\mathbf{x}}^-) + \langle \hat{\mathbf{x}}^+, \hat{\mathbf{x}}^- \rangle) \\ &= f_{Z \setminus \{z^+, z^-\}}(\hat{\mathbf{x}}^+ - \hat{\mathbf{x}}^-) - 2\langle \hat{\mathbf{x}}^+, \hat{\mathbf{x}}^- \rangle \\ &= \langle \mathbf{a}, \mathbf{x}^+ - \mathbf{x}^- \rangle - 2\langle \mathbf{x}^+, \mathbf{x}^- \rangle - 2\langle \mathbf{x}^+, \mu \rangle - 2\langle \mu, \mathbf{x}^- \rangle - 2\|\mu\|^2. \end{aligned} \quad (1)$$

Recall that if $e > 0$, the holdout instances are correctly ranked. As discussed before, AUC is the relative frequency of correctly ranked positive-negative pairs, with 0.5 being the expected value for non-signal data.

For simplicity, we now assume that the data having no signal follows a normal distribution as is the case in our experiments. For other distributions, the analysis is more complicated. The probability of the first four terms of (1) being positive is 0.5, while the last term $-2\|\mu\|^2$ is always negative. As a consequence, if the mean of the data is not zero, a negative bias is introduced

into the LOO estimate, since the probability of correctly ordering the two holdout instances becomes less than 0.5.

In our experiments with data having no signal, the data follows a normal distribution with zero mean. Of course, the finite training samples have a nonzero mean. Nevertheless, the norm of the mean vector is usually negligible compared to the other four terms of (1), and hence it is only able to cause a minor bias in the LOO results. However, as discussed in Section 5.1, the feature vectors of the training instance were appended with a constant valued feature in order to increase the representational power of the learning algorithms in low dimensions. This operation directly increases the norm of the mean vector of the training set by the absolute value of the constant feature. This increase, in proportion to the other four terms of 1, is large in low dimensional data, while it again becomes negligible as the dimensionality of the data increases.

The analysis suggest that the existence of the intercept term can introduce substantial bias to LOO AUC estimates with the Parzen window classifier, and by extension to the RLS and SVM classifiers. However if the value of this term is held constant, the effect lessens with increase in dimensionality. The results do not apply to RankRLS, which ignores the intercept term if present. The analysis is consistent with results observed in the experiments, where LOO estimates for RLS are extremely biased on low-dimensions, but the effect disappears in higher dimensions, whereas RankRLS is much less biased. Further, based on this analysis we performed an experiment on low-dimensional non-signal data both for RLS and SVM, where we removed the intercept term from the optimization problem. This reduced the negative bias considerably, suggesting a possible way to trade off the expressive power of these learning algorithms for lessened bias in pooled estimation. We do not however recommend this approach, as with low dimensional signal data the intercept term can be crucial for allowing the inferred model to separate the classes.

Thus, we conjecture that this analysis partly explains the phenomenon observed in our experiments that the bias of LOO is for RLS much more severe in low-dimensions than in high dimensions.

References

Agarwal, S., Graepel, T., Herbrich, R., Har-Peled, S., Roth, D., 2005. Generalization bounds for the area under the ROC curve. Journal of Machine

Learning Research 6, 393–425.

Airola, A., Pahikkala, T., Waegeman, W., De Baets, B., Salakoski, T., 2009. A comparison of AUC estimators in small-sample studies, in: Džeroski, S., Geurts, P., Rousu, J. (Eds.), Proceedings of the Third International Workshop on Machine Learning in Systems Biology (MLSB'09). Helsinki University Printing House, Helsinki, Finland, pp. 15–23.

Airola, A., Pyysalo, S., Björne, J., Pahikkala, T., Ginter, F., Salakoski, T., 2008. All-paths graph kernel for protein-protein interaction extraction with evaluation of cross-corpus learning. BMC Bioinformatics 9, S2.

An, S., Liu, W., Venkatesh, S., 2007. Fast cross-validation algorithms for least squares support vector machine and kernel ridge regression. Pattern Recognition 40, 2154–2162.

Baker, S., Kramer, B., 2006. Identifying genes that contribute most to good classification in microarrays. BMC Bioinformatics 7, 407.

Bradley, A.P., 1997. The use of the area under the ROC curve in the evaluation of machine learning algorithms. Pattern Recognition 30, 1145–1159.

Braga-Neto, U.M., Dougherty, E.R., 2004. Is cross-validation valid for small-sample microarray classification? Bioinformatics 20, 374–380.

Cortes, C., Mohri, M., 2004. AUC optimization vs. error rate minimization, in: Thrun, S., Saul, L., Schölkopf, B. (Eds.), Advances in Neural Information Processing Systems 16. MIT Press, Cambridge, Massachusetts, USA.

Cortes, C., Mohri, M., Rastogi, A., 2007. An alternative ranking problem for search engines, in: Demetrescu, C. (Ed.), Proceedings of the 6th Workshop on Experimental Algorithms. Springer, Berlin / Heidelberg, Germany. volume 4525 of *Lecture Notes in Computer Science*, pp. 1–21.

Dietterich, T.G., 1998. Approximate statistical tests for comparing supervised classification learning algorithms. Neural Computation 10, 1895–1923.

Fawcett, T., Flach, P.A., 2005. A response to Webb and Ting's "On the application of ROC analysis to predict classification performance under varying class distributions". Machine Learning 58, 33–38.

Forman, G., Scholz, M., 2010. Apples-to-apples in cross-validation studies: Pitfalls in classifier performance measurement. SIGKDD Explorations 12, 49–57.

Fung, G., Mangasarian, O.L., 2001. Proximal support vector machine classifiers, in: KDD '01: Proceedings of the seventh ACM SIGKDD international conference on Knowledge discovery and data mining, ACM, New York, NY, USA. pp. 77–86. doi:http://doi.acm.org/10.1145/502512.502527.

Gevaert, O., De Smet, F., Timmerman, D., Moreau, Y., De Moor, B., 2006. Predicting the prognosis of breast cancer by integrating clinical and microarray data with bayesian networks. Bioinformatics 22, 184–190.

Hanczar, B., Hua, J., Sima, C., Weinstein, J., Bittner, M., Dougherty, E.R., 2010. Small-sample precision of ROC-related estimates. Bioinformatics 26, 822–830.

Hanley, J.A., McNeil, B.J., 1982. The meaning and use of the area under a receiver operating characteristic (ROC) curve. Radiology 143, 29–36.

Hastie, T., Rosset, S., Tibshirani, R., Zhu, J., 2004. The entire regularization path for the support vector machine. Journal of Machine Learning Research 5, 1391–1415.

Hastie, T., Tibshirani, R., Friedman, J., 2009. The Elements of Statistical Learning: Data Mining, Inference and Prediction, Second Edition. Springer Series in Statistics, Springer.

Herbrich, R., Graepel, T., Obermayer, K., 1999. Support vector learning for ordinal regression, in: Proceedings of the Ninth International Conference on Articial Neural Networks (ICANN'99), Institute of Electrical Engineers, London. pp. 97–102.

Hoerl, A.E., Kennard, R.W., 1970. Ridge regression: Biased estimation for nonorthogonal problems. Technometrics 12, 55–67.

Huang, J., Ling, C.X., 2005. Using AUC and accuracy in evaluating learning algorithms. IEEE Transactions on Knowledge and Data Engineering 17, 299–310.

Kim, J.H., 2009. Estimating classification error rate: Repeated cross-validation, repeated hold-out and bootstrap. Computational Statistics & Data Analysis 53, 3735–3745.

Kohavi, R., 1995. A study of cross-validation and bootstrap for accuracy estimation and model selection, in: Mellish, C. (Ed.), Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence, Morgan Kaufmann, San Mateo, California, USA. pp. 1137–1143.

Krzanowski, W., Hand, D., 1997. Assessing error rate estimators: the leave-one-out method reconsidered. Australian Journal of Statistics 39, 35–46.

Luntz, A., Brailovsky, V., 1969. On estimation of characters obtained in statistical procedure of recognition. Technicheskaya Kibernetica 3, 563–575. [in Russian].

Miwa, M., Sætre, R., Miyao, Y., Tsujii, J., 2009. Protein-protein interaction extraction by leveraging multiple kernels and parsers. International Journal of Medical Informatics 78, e39–e46.

Pahikkala, T., Airola, A., Boberg, J., Salakoski, T., 2008. Exact and efficient leave-pair-out cross-validation for ranking RLS, in: Honkela, T., Pöllä, M., Paukkeri, M.S., Simula, O. (Eds.), Proceedings of the 2nd International and Interdisciplinary Conference on Adaptive Knowledge Representation and Reasoning, Helsinki University of Technology. pp. 1–8.

Pahikkala, T., Boberg, J., Salakoski, T., 2006. Fast n-fold cross-validation for regularized least-squares, in: Honkela, T., Raiko, T., Kortela, J., Valpola, H. (Eds.), Proceedings of the Ninth Scandinavian Conference on Artificial Intelligence, Otamedia, Espoo, Finland. pp. 83–90.

Pahikkala, T., Pyysalo, S., Boberg, J., Järvinen, J., Salakoski, T., 2009a. Matrix representations, linear transformations, and kernels for disambiguation in natural language. Machine Learning 74, 133–158.

Pahikkala, T., Tsivtsivadze, E., Airola, A., Boberg, J., Järvinen, J., 2009b. An efficient algorithm for learning to rank from preference graphs. Machine Learning 75, 129–165.

Pahikkala, T., Tsivtsivadze, E., Airola, A., Boberg, J., Salakoski, T., 2007. Learning to rank with pairwise regularized least-squares, in: Joachims, T.,

Li, H., Liu, T.Y., Zhai, C. (Eds.), SIGIR 2007 Workshop on Learning to Rank for Information Retrieval, pp. 27–33.

Parker, B.J., Gunter, S., Bedo, J., 2007. Stratification bias in low signal microarray studies. BMC Bioinformatics 8, 326.

Provost, F.J., Fawcett, T., Kohavi, R., 1998. The case against accuracy estimation for comparing induction algorithms, in: Shavlik, J. (Ed.), Proceedings of the Fifteenth International Conference on Machine Learning, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA. pp. 445–453.

Rifkin, R., 2002. Everything Old Is New Again: A Fresh Look at Historical Approaches in Machine Learning. Ph.D. thesis. Massachusetts Institute of Technology.

Saunders, C., Gammerman, A., Vovk, V., 1998. Ridge regression learning algorithm in dual variables, in: Proceedings of the Fifteenth International Conference on Machine Learning, Morgan Kaufmann Publishers Inc., San Francisco, California, USA. pp. 515–521.

Schiavo, R.A., Hand, D.J., 2000. Ten more years of error rate research. International Statistical Review 68, 295–310.

Suykens, J.A.K., Vandewalle, J., 1999. Least squares support vector machine classifiers. Neural Processing Letters 9, 293–300.

Swets, J.A., 1988. Measuring the accuracy of diagnostic systems. Science 240, 1285–1293.

Teo, C.H., Vishwanathan, S.V., Smola, A., Le, Q.V., 2010. Bundle methods for regularized risk minimization. Journal of Machine Learning Research 11, 311–365.

Vanderlooy, S., Hüllermeier, E., 2008. A critical analysis of variants of the AUC. Machine Learning 72, 247–262.

Vapnik, V., 1979. Estimation of Dependences Based on Empirical Data [in Russian]. Nauka, Moscow. (English translation: Springer, New York, 1982).

Vapnik, V.N., 1995. The nature of statistical learning theory. Springer-Verlag New York, Inc., New York, NY, USA.

Waegeman, W., De Baets, B., Boullart, L., 2008. ROC analysis in ordinal regression learning. Pattern Recognition Letters 29, 1–9.

Wilcoxon, F., 1945. Individual comparisons by ranking methods. Biometrics 1, 80–83.

Zhang, P., Peng, J., 2004. SVM vs regularized least squares classification, in: Kittler, J., Petrou, M., Nixon, M. (Eds.), Proceedings of the 17th International Conference on Pattern Recognition, IEEE Computer Society, Washington, DC, USA. pp. 176–179.

Zhang, P., Peng, J., Riedel, N., 2005. Finite sample error bound for Parzen windows, in: AAAI'05: Proceedings of the 20th national conference on Artificial intelligence, AAAI Press. pp. 925–930.

Zhang, T., 2002. On the dual formulation of regularized linear systems with convex risks. Machine Learning 46, 91–129.