Pruned Random Subspace Method for One-Class Classifiers

Veronika Cheplygina and David M. J. Tax

Pattern Recognition Lab, Delft University of Technology
v.cheplygina@tudelft.nl, d.m.j.tax@tudelft.nl

Abstract. The goal of one-class classification is to distinguish the target class from all the other classes using only training data from the target class. Because it is difficult for a single one-class classifier to capture all the characteristics of the target class, combining several one-class classifiers may be required. Previous research has shown that the Random Subspace Method (RSM), in which classifiers are trained on different subsets of the feature space, can be effective for one-class classifiers. In this paper we show that the performance by the RSM can be noisy, and that pruning inaccurate classifiers from the ensemble can be more effective than using all available classifiers. We propose to apply pruning to RSM of one-class classifiers using a supervised AUC criterion or an unsupervised consistency criterion. It appears that when the AUC criterion is used, the performance may be increased dramatically, while for the consistency criterion results do not improve, but only become more predictable.

Keywords: One-class classification, Random Subspace Method, Ensemble learning, Pruning Ensembles

1 Introduction

The goal of one-class classification is to create a description of one class of objects (called target objects) and distinguish this class from all other objects, not belong to this class (called outliers) [1]. One-class classification is particularly suitable for situations where the outliers not represented well in the training set. This is common in applications in which examples of one class are more difficult to obtain or expensive to sample, such as detecting credit card fraud, intrusions, or a rare disease [2].

For a single one-class classifier it may be hard to find a good model because of limited training data, high dimensionality of the feature space and/or the properties of the particular classifier. In one-class classification, a decision boundary should be fitted around the target class such that it distinguishes target objects from *all* potential outliers. That means that a decision boundary should be estimated in all directions in the feature space around the target class. Compared to the standard two-class classification problems, where we may expect objects from the other class predominantly in one direction, this requires more parameters to fit, and therefore more training data.

To avoid a too complex model and overfitting on the training target data, simpler models can be created that use less features. In the literature, approaches such as the Random Subspace Method (RSM) [3]) or feature bagging [4] are proposed. In RSM, several classifiers are trained on random feature subsets of the data and the decisions of these classifiers are combined. RSM is expected to benefit in problems suffering from the "curse of dimensionality" because of the improved object/feature ratio for each individual classifier. It has been demonstrated that combining classifiers can also be effective for one-class classifiers [1, 5]. RSM is successfully applied to a range of one-class classifiers in [4, 6, 7].

Although it was originally believed that larger ensembles are more effective, it has been demonstrated that using a subset of classifiers might be better than the using the whole set [8]. A simple approach is to evaluate L classifiers individually according to a specified criterion (such as accuracy on the training set) and select the L_s ($L_s < L$) best classifiers (i.e. prune the inaccurate classifiers). Pruning in RSM has been shown to be effective for traditional classifiers [9], however, to apply this to one-class classifiers, one faces the problem of choosing a good selection criterion. Most criteria require data from all classes, but in the case of one-class classifiers one assumes a (very) poorly sampled outlier class. This paper evaluates two criteria for the pruned RSM: the area under the ROC curve (AUC) [10], which uses both target and outlier examples, and the consistency criterion, using only target data [11].

In Sect. 2, some background concerning one-class classifiers and RSM is given. The pruned random subspace method (PRSM) is proposed in Sect. 3. In Sect. 4, experiments are performed to analyze the behavior of the PRSM compared to the basic RSM and other popular combining methods. Conclusion and suggestions for further research are presented in Sect. 5.

2 Combining One-class Classifiers

Supervised classification consists of approximating the true classification function $y = h(\mathbf{x})$ using a collection of example object-label pairs $X = \{(\mathbf{x}_1, y_1)..., (\mathbf{x}_N, y_N)\}$ ($\mathbf{x} \in \mathbb{R}^d$) with an hypothesis h, and then using h to assign y values to previously unseen objects $\mathbf{x}$. Let us assume there are two classes, i.e. $y \in \{T, O\}$. A traditional classifier needs labeled objects of both classes in order to create h, and its performance suffers if one of the classes is absent from the training data. In these situations one-class classifiers can be used. A one-class classifier only needs objects of one class to create h. It is thus trained to accept objects of one class (target class) and reject objects of the other (outlier class).

A one-class classifier consists of two elements: the "similarity" of an object to the target class, expressed as a posterior probability $p(y = T|\mathbf{x})$ or a distance (from which $p(y = T|\mathbf{x})$ can be estimated), and a threshold θ on this measure, which is used to determine whether a new object belongs to the target class or not:

$$h(\mathbf{x}) = \delta(p(y = T|\mathbf{x}) > \theta) = \begin{cases} +1, & \text{when } p(y = T|\mathbf{x}) > \theta, \\ -1, & \text{otherwise,} \end{cases} \tag{1}$$

where δ is the indicator function. In practice, θ is set such that the classifier rejects a fraction f of the target class.

Estimating $p(y = T|\mathbf{x})$ is hard, in particular for high-dimensional feature spaces and low sample size situations. By making several (lower dimensional) approximations of h and combining these, a more robust model can be obtained. Assume that $\mathbf{s}(\mathbf{x})$ maps the original d-dimensional data to a d_s-dimensional subspace:

$$\mathbf{s}(\mathbf{x}; \mathbf{I}) = [x_{I_1}, x_{I_2}, ..., x_{I_{d_s}}]^T \tag{2}$$

where $\mathbf{I} = [I_1, I_2, ..., I_{d_s}]$ is the index vector indicating the features that are selected. Typically, the features are selected at random, resulting in RSM [3].

When in each of the subspaces a model $h_i(\mathbf{s}(\mathbf{x}; \mathbf{I}_i))$ (or actually $p_i(y = T|\mathbf{s}(\mathbf{x}; \mathbf{I}_i))$) is estimated, several combining methods can be used [12] to combine the subspace results. Two standard approaches are averaging of the posterior probabilities (also called mean combining):

$$\tilde{p}(y = T|\mathbf{x}) = \frac{1}{L} \sum_{i=1}^{L} p_i(y = T|\mathbf{s}(\mathbf{x}; \mathbf{I}_i)) \tag{3}$$

or voting, i.e.:

$$\tilde{p}(y = T|\mathbf{x}) = \frac{1}{L} \sum_{i=1}^{L} I(p_i(y = T|\mathbf{s}(\mathbf{x}; \mathbf{I}_i)) > \theta_i) \tag{4}$$

In Table 1, the AUC performances are shown of a combination of Gaussian models (such that $p(y = T|\mathbf{x})$ in (1) is modelled as a normal distribution) for three datasets, which are described in more detail in Sect.4. The number of subspaces L and the combining rules are varied. The subspace dimensionality is fixed at 25% of the original feature space dimensionality.

As can be observed, RSM is able to significantly outperform the base classifiers, but this improvement is quite unpredictable, i.e., it is difficult to say what settings need to be chosen to achieve the best performance. The optimal number of subspaces differs across the datasets ($L = 25$ for Concordia, $L = 10$ for Imports, $L = 50$ for Sonar), and also different combining rules are preferred for different datasets. On the other hand, there seems to be a trend in the combining methods: mean produces better results than voting in most situations.

Furthermore, the subspace dimensionality d_s also has an influence [13]. In literature, dimensionalities between 20% and 80% of the original feature space are being used [7, 6]. In our experiments, 25% gave reasonable performances overall, so that is being fixed in this paper.

3 Pruned Random Subspaces for One-class Classification

A successful classifier should have good performance *and* be predictable in terms of how its parameters influence its performance. It should be possible to choose

Table 1. AUC performances (×100) of the RSM combining Gaussian models on the Concordia digit 3, Imports and Sonar datasets. Horizontally the number of subspaces is varied, vertically the combining rule. The subspace dimensionality is fixed to 25% of the total feature size. Results are averaged over 5 times 10-fold stratified cross-validation. Results in bold indicate the best performance over the combining rules (or a performance not significantly worse than the best at a $\alpha = 0.05$ significance level).

Comb.rule	Number of subspaces L					
	10	25	50	75	100	Base
Concordia digit 3 dataset, 256 features						
mean	**91.2 (2.6)**	**94.5 (1.9)**	**93.9 (1.9)**	**94.0 (2.0)**	**93.8 (2.1)**	**92.1 (3.2)**
prod	90.9 (2.8)	94.0 (2.3)	93.5 (2.4)	93.4 (2.5)	92.6 (2.4)	**92.1 (3.2)**
vote	89.9 (2.4)	93.7 (2.0)	93.4 (2.1)	93.6 (2.0)	93.4 (2.0)	**92.1 (3.2)**
max	90.2 (3.2)	93.7 (2.5)	93.5 (2.4)	93.4 (2.6)	93.4 (2.6)	**92.1 (3.2)**
Imports dataset, 25 features						
mean	**78.0 (11.4)**	73.7 (12.2)	71.2 (12.4)	69.6 (12.8)	68.8 (12.6)	**74.0 (13.2)**
prod	**78.3 (10.8)**	73.5 (11.8)	71.6 (12.7)	67.5 (11.4)	65.6 (10.4)	**74.0 (13.2)**
vote	**78.7 (12.6)**	**77.0 (13.4)**	**74.5 (14.2)**	72.8 (14.0)	72.7 (14.0)	**74.0 (13.2)**
max	75.6 (12.1)	**76.3 (11.2)**	**76.0 (11.3)**	**75.2 (11.2)**	**74.3 (11.7)**	**74.0 (13.2)**
Sonar rocks dataset, 60 features						
mean	**65.8 (13.3)**	**65.1 (13.1)**	**65.9 (13.0)**	**65.6 (13.0)**	**65.5 (12.9)**	**63.0 (12.5)**
prod	**65.8 (13.2)**	**65.1 (13.1)**	64.5 (13.0)	62.6 (11.5)	61.7 (10.0)	**63.0 (12.5)**
vote	62.6 (12.3)	63.2 (12.1)	64.5 (12.0)	**65.1 (12.3)**	64.7 (12.4)	**63.0 (12.5)**
max	64.1 (13.9)	**65.1 (13.3)**	64.6 (13.3)	64.0 (12.4)	64.4 (11.8)	**63.0 (12.5)**

parameters based on knowledge of the data and classifier, rather than trying all possible parameter combinations, as in Table 1. Based on our observations, the only parameter that can be chosen with some certainty is the combining rule. In most situations, the mean combining performs the best (although in some situations other combining methods might be comparable). This still leaves the "difficult" parameter, the number of subspaces L.

It appears that RSM produces subspace classifiers of different quality. In some subspaces, the distinction between the target and outlier objects is much clearer than in others. Averaging is sensitive to such variations [14], thereby causing the performance of the whole ensemble to vary significantly. By excluding the less accurate classifiers, the performance of the ensemble should stabilize. In this case, the choice for the number of subspaces is less critical and less prone to noise in the subspace selection.

Therefore we propose PRSM, a pruned version of RSM. The implementation of PRSM is shown in Algorithm 1. The essential difference with the RSM is visible in the second last line. Instead of combining all subspace results, PRSM only uses the best L_s outcomes. In order to find the best subspaces, the subspaces should be ranked according to an evaluation measure C.

We use two different evaluation measures particularly suitable for OCCs: AUC [10] and consistency [11]. The AUC is obtained by integrating the area under the Receiver-Operating-Characteristic curve, given by $(\varepsilon^t, 1 - \varepsilon^o)$ pairs,

Algorithm 1 PRSM Algorithm

Input: Training set X, base classifier h, number of classifiers L, number of selected classifiers L_s, subspace dim. d_s, subspace criterion C, combining method M
Output: Ensemble E
  for $i = 1$ to L **do**
    $I_i \leftarrow RandomSelectFeatures(X, d_s)$
    $score_i \leftarrow CrossValidate(X, h, I_i, L_s, C)$
  end for
  $\mathbf{I} \leftarrow rankByScore(\mathbf{I}, score)$
  $\mathbf{h} \leftarrow Train(X, h, \mathbf{I}_{1:L_s})$
  return $E \leftarrow Combine(\mathbf{h}, M)$

where ε^t is the error on the target class and ε^o is the error on the outlier class, as shown in (5). Because the true values of ε^t and ε^o are unknown, in practice, the AUC is obtained by varying f, estimating the $(\hat{\varepsilon}^t, 1-\hat{\varepsilon}^o)$ pairs using a validation set, and integrating under this estimated curve.

$$AUC = 1 - \int_0^1 \varepsilon^o(\varepsilon^t)\,\mathrm{d}\varepsilon^t. \tag{5}$$

The consistency measure indicates how consistent a classifier is in rejecting fraction f of the target data. It is obtained by comparing f with $\hat{\varepsilon^t}$, an estimate of the error on the target class:

$$Consistency = |\hat{\varepsilon}^t - f| \tag{6}$$

The AUC measure is chosen for its insensitiveness to class imbalance, which may often be a problem in one-class classification tasks. However, to obtain the AUC both $\hat{\varepsilon}^t$ and $\hat{\varepsilon}^o$ are needed, which means that the validation set must contain outliers. Therefore, pruning using AUC is not strictly a one-class method and the performance estimates might be more optimistic than in the pure one-class setting. On the other hand, to obtain the consistency measure only $\hat{\varepsilon}^t$ is required, which means no outlier information is used.

Using a validation set has another implication: there may be too little data to use a separate validation set. An alternative is to do the evaluation using the training set as in [9]. However, this is not possible for classifiers which have 100% performance on the training set, such as 1-NN. In our implementation, we perform 10-fold cross-validation using the training set to evaluate the subspace classifiers. After ranking the classifiers, they are retrained using the complete training set.

4 Experiments

In this section, experiments are performed to analyze the behavior of the PRSM compared to the basic RSM and other combining methods. First, the PRSM is compared with RSM with respect to the absolute performance and the stability.

Next, the PRSM is compared to other classifiers across a range of datasets. We use the base classifier and RSM, Bagging [15], and AdaBoost [16] ensembles of the same base classifier. A separate comparison is performed for each base classifier. For the comparison, we use the Friedman test [17] and the post-hoc Nemenyi test [18], as recommended in [19].

Table 2. List of datasets. N_T and N_O represent the numbers of target and outlier objects, d stands for dimensionality.

Dataset	N_T	N_O	d	Dataset	N_T	N_O	d
Arrhythmia	237	183	278	Prime Vision	50	150	72
Cancer non-ret	151	47	33	Pump 2×2 noisy	64	176	64
Cancer ret	47	151	33	Pump 1×3 noisy	41	139	64
Concordia 2	400	3600	256	Sonar mines	111	97	60
Concordia 3	400	3600	256	Sonar rocks	97	111	60
Glass	17	197	9	Spambase	79	121	57
Housing	48	458	13	Spectf normal	95	254	44
Imports	71	88	25	Vowel 4	48	480	10
Ionosphere	225	126	34	Wine 2	71	107	13

All experiments are performed in Matlab using PRTools [20] and the Data Description toolbox [21]. In [1], several strategies for deriving one-class classifiers are described. In this paper we consider three typical examples: the Gaussian (Gauss), Nearest Neighbor (1-NN), and k-Means one-class classifiers. Gauss is a *density* method, which fits a normal distribution to the data and rejects objects on the tails of this distribution. 1-NN is a *boundary* method, which uses distances between objects to calculate the local densities of objects, and rejects new objects with a local density lower than that of its nearest neighbor. k-Means is a *reconstruction* method, which assumes that the data is clustered in k groups, finds k prototype objects for these groups, and creates a boundary out of the hyperspheres placed at these objects.

Seventeen datasets are used from the UCI Machine Learning Repository [22], and have been modified in order to contain a target and an outlier class [23]. Furthermore, an additional dataset, the Prime Vision dataset, provided by a company called Prime Vision[1] in Delft, The Netherlands, is used. A summary of the datasets is included in Table 2. All the datasets have low object-to-feature ratios, either originally or after downsampling (in case of Prime Vision and Spambase datasets).

In Fig. 1, a comparison between PRSM+AUC (PRSM using the AUC criterion), PRSM+Con (PRSM using the consistency criterion) and RSM for the Concordia digit 3 dataset is shown. For varying number of subspaces L the AUC performance of the combined classifier is presented. We choose $d_s = 64$ (that is

[1] http://www.primevision.com

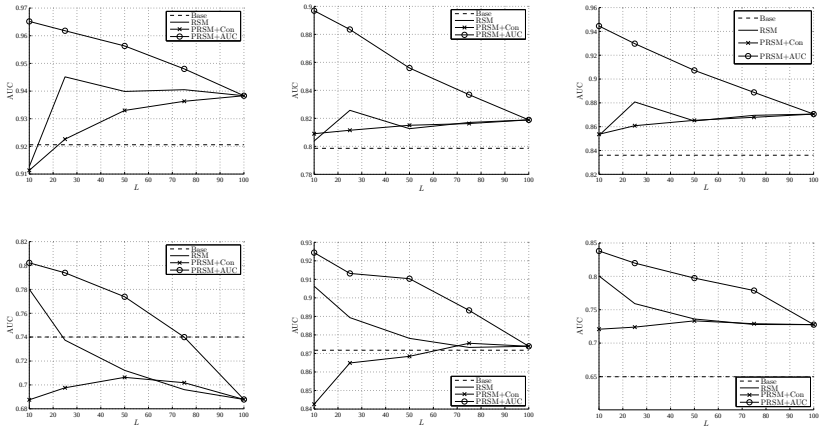

Fig. 1. The AUC (on test set) for varying number of subspaces for RSM, PRSM+AUC and PRSM+Con for the Concordia digit 3 dataset (top row) and for the Imports dataset (bottom row). From left to right: Gauss, 1-NN and k-Means base classifiers.

25% of the features) and compare the performances of RSM with L_s classifiers to PRSM with L_s classifiers out of $L = 100$.

The results show that both the RSM and the PRSM improve over the baseline performance. The results also show that the PRSM+Cons often has a lower performance than the standard RSM, but the outcomes are less noisy and more smooth over varying number of subspaces. The optimal performance using the consistency criterion is often around 50% of the total of $L = 100$ subspaces. Using PRSM+AUC, on the other hand, generally significantly improves the results. In most situations PRSM+AUC obtains the best performance by using just a very small subset of $L_s = 10$. Based on these experiments, L_s is fixed to 50% for the consistency criterion, and 10% for the AUC criterion.

Next, the performance of PRSM+Con, PRSM+AUC and RSM are compared to the baseline performance, and a few standard approaches, namely Bagging [15] and AdaBoost [16]. In Bagging, $L = 100$ bootstrapped versions of the training set X are generated, a base classifier is trained on these bootstrapped training sets, and the combined classifier is the average (as in (3)) of the L trained base classifiers. In AdaBoost, new training sets are iteratively generated by reweighing objects. Objects that are often misclassified get a higher weight, and are thus more likely to be selected in a training set.

The results of the comparison are shown in Table 3. A key observation is that overall, the one-class classifiers have comparable performance, however, there is no overall winning classifier. For some datasets, such as Glass, it is often best to just use the base classifier as no ensemble is able to improve on this performance. This can be explained because this dataset is very small (only 17 training target objects), and fitting a more complex classifier than the base classifier just overfits on the training data.

Surprisingly, also AdaBoost did not perform very well. AdaBoost is originally developed to boost two-class classifiers by reweighing training objects. In these experiments, the base classifier is a one-class classifier, trained to describe the target data. Errors on the target class are generally weighted more heavily than errors on the outlier class. This results in a heavy imbalance in the sampling of the two classes, and consequently in poorly trained base classifiers.

In most situations, PRSM+AUC outperforms the other ensemble methods, and the base classifiers. The improvement is not always very large, but for some datasets it is quite significant, for instance, for the Arrhythmia, Imports, Spambase, and the Concordia datasets. These are often the high-dimensional datasets. A noticeable exception is the Ionosphere dataset. Here, the base classifiers already perform well, and the improvement achieved by the ensembles is modest.

The results of the statistical tests are shown in Table 4. For each base classifier, the F-statistic value indicates there are any significant differences between the classification methods. Significant differences are found if F is larger than a critical value (2.33 in this case) which depends on the number of datasets (18) and classifiers (6) in the experiment. If this is the case, the Nemenyi test can be used to compare any two classifiers. A significant difference between two classifiers occurs when the difference in classifier ranks is larger than the critical difference for the Nemenyi test, 1.77 in this case.

RSM is only significantly better than the base classifier for k-Means and even worse than the base classifier for Gauss. PRSM+Con produces worse (but not significantly worse) results than RSM for all three base classifiers. On the other hand, PRSM+AUC is significantly better than RSM for Gauss and 1-NN. Bagging and AdaBoost perform poorly in general, except for Gauss where Bagging is slightly better than RSM.

5 Conclusions

In this paper, we investigated the effect of pruning on Random Subspace ensembles on the Gaussian, Nearest Neighbor, and k-Means one-class classifiers. Experiments show that pruning the random subspaces for one-class classifiers improves the stability of the outcomes, but does not directly improve the classification performance. In order to improve the performance, additional information, in the form of extra outlier objects, has to be used. The pruned random subspace method in which subspaces are selected according to the Area under the ROC curve, therefore, shows a significant improvement over the standard random subspace method. Furthermore, the number of subspaces that is required for good performance is often very low: 10 out of 100 subspaces is often already showing optimal performance. These results suggest that combining a few accurate classifiers may be more beneficial than combining all available classifiers. Furthermore, the more stable performance of PRSM reduces the number of parameters that need to be set for RSM, making it a more applicable classifier. The effect of the subspace dimensionality parameter requires further investigation.

References

1. Tax, D.: One-class classification; Concept-learning in the absence of counter-examples. PhD thesis, Delft University of Technology (2001)
2. Chandola, V., Banerjee, A., Kumar, V.: Anomaly detection: A survey. ACM Computing Surveys (CSUR) **41**(3) (2009) 1–58
3. Ho, T.: The random subspace method for constructing decision forests. IEEE Transactions on Pattern Analysis and Machine Intelligence **20**(8) (1998) 832–844
4. Lazarevic, A., Kumar, V.: Feature bagging for outlier detection. In: Proc. of the 11th ACM SIGKDD Int. Conf. on Knowledge Discovery in Data Mining. (2005) 157–166
5. Tax, D., Duin, R.: Combining one-class classifiers. Multiple Classifier Systems (2001) 299–308
6. Nanni, L.: Experimental comparison of one-class classifiers for online signature verification. Neurocomputing **69**(7-9) (2006) 869–873
7. Biggio, B., Fumera, G., Roli, F.: Multiple classifier systems under attack. Multiple Classifier Systems (2010) 74–83
8. Zhou, Z., Wu, J., Tang, W.: Ensembling neural networks: Many could be better than all. Artificial Intelligence **137**(1-2) (2002) 239–263
9. Bryll, R., Gutierrez-Osuna, R., Quek, F.: Attribute bagging: improving accuracy of classifier ensembles by using random feature subsets. Pattern Recognition **36**(6) (2003) 1291–1302
10. Bradley, A.: The use of the area under the ROC curve in the evaluation of machine learning algorithms. Pattern Recognition **30**(7) (1997) 1145–1159
11. Tax, D., Muller, K.: A consistency-based model selection for one-class classification. In: Proc. of the 17th Int. Conf. on Pattern Recognition. (2004) 363–366
12. Kittler, J., Hatef, M., Duin, R., Matas, J.: On combining classifiers. IEEE Transactions on Pattern Analysis and Machine Intelligence **20**(3) (2002) 226–239
13. Cheplygina, V.: Random subspace method for one-class classifiers. Master's thesis, Delft University of Technology (2010)
14. Oza, N., Tumer, K.: Classifier ensembles: Select real-world applications. Information Fusion **9**(1) (2008) 4–20
15. Breiman, L.: Bagging predictors. Machine Learning **24**(2) (1996) 123–140
16. Schapire, R., Freund, Y.: Experiments with a new boosting algorithm. In: Proc. of the 13th Int. Conf. on Machine Learning. (1996) 148
17. Friedman, M.: The use of ranks to avoid the assumption of normality implicit in the analysis of variance. Journal of the American Statistical Association **32**(200) (1937) 675–701
18. Nemenyi, P.: Distribution-free multiple comparisons. PhD thesis, Princeton (1963)
19. Demšar, J.: Statistical comparisons of classifiers over multiple data sets. The Journal of Machine Learning Research **7** (2006) 1–30
20. Duin, R.: PRtools. Version 4.1.5. `http://www.prtools.org` (2009)
21. Tax, D.: DD_tools, the Data Description toolbox for Matlab. Version 1.7.4. `http://prlab.tudelft.nl/david-tax/dd_tools.html` (2010)
22. Asuncion, A. and Newman, D.J.: UCI Machine Learning Repository (2007)
23. Tax, D.: OC classifier results. `http://homepage.tudelft.nl/n9d04/occ/index.html` (2010)

Table 3. AUC performances (×100) of the base classifier and ensemble methods for the Gauss, 1-NN and k-Means base classifiers. PRSc indicates the PRSM using the consistency criterion, PRSa indicates the PRSM using the AUC criterion. Bag means bagging, and AB means AdaBoost.

Data	Base	RSM	PRSc	PRSa	Bag	AB	Data	Base	RSM	PRSc	PRSa	Bag	AB
							Gauss						
Arr.	75.6	78.0	77.8	**79.4**	75.5	50.0	PV	89.4	88.3	88.1	**89.8**	**90.2**	50.6
Canc1	50.2	**53.6**	**53.4**	**52.8**	51.1	**52.8**	Pump1	**94.8**	83.1	83.8	92.3	88.5	50.0
Canc2	62.4	60.9	59.6	**65.7**	61.0	52.8	Pump2	91.8	84.0	86.2	**98.2**	82.6	50.0
Conc.2	83.6	87.0	86.3	**91.3**	82.5	51.5	Sonar1	65.7	64.1	63.5	**70.4**	65.2	54.6
Conc.3	92.1	93.8	93.2	**96.4**	91.2	51.8	Sonar2	62.2	64.7	65.0	**69.1**	62.4	54.1
Glass	**86.2**	74.7	73.3	78.3	85.1	70.0	Spam	81.5	79.1	78.7	**85.5**	77.1	61.1
Hous.	88.1	84.1	84.3	**91.3**	88.0	73.7	Spect	94.5	88.3	88.4	88.8	**94.9**	88.5
Impor.	72.6	68.5	71.0	**80.3**	73.8	70.0	Vow.4	**99.1**	95.6	94.5	96.2	**99.1**	96.7
Ion.	96.5	96.9	97.0	**97.3**	96.4	87.4	Wine	94.4	92.9	90.4	**96.4**	94.3	90.1
							1-NN						
Arr.	73.8	75.5	75.3	**78.1**	73.8	69.4	PV	88.0	88.7	88.5	**90.9**	87.9	86.2
Canc1	**52.6**	**53.2**	**53.1**	**53.5**	**53.4**	**51.8**	Pump1	77.3	76.0	76.3	**82.0**	76.9	77.2
Canc2	55.8	57.6	58.0	**68.3**	56.2	57.5	Pump2	78.5	77.2	76.9	**84.4**	77.9	72.1
Conc.2	69.7	80.3	80.4	**88.9**	68.7	64.0	Sonar1	67.8	68.8	68.2	**78.5**	66.5	59.2
Conc.3	83.7	87.2	86.6	**94.6**	82.8	78.1	Sonar2	72.2	72.3	72.4	**74.6**	70.7	63.3
Glass	**72.8**	**73.7**	**72.5**	**74.5**	69.9	62.5	Spam	56.7	60.5	60.8	**74.7**	53.4	53.2
Hous.	87.5	**94.7**	**94.8**	**94.7**	83.7	83.4	Spect	**95.4**	**95.6**	**95.8**	**95.9**	95.2	87.3
Impor.	85.6	86.5	86.3	**92.4**	80.2	77.3	Vow.4	**99.4**	99.2	99.1	**99.5**	99.1	97.6
Ion.	95.9	96.1	96.1	**97.3**	96.3	90.2	Wine	87.1	91.7	90.5	**94.0**	88.2	92.6
							k-means						
Arr.	74.0	75.2	75.3	**77.1**	74.4	65.3	PV	86.5	88.2	88.1	**89.6**	86.8	84.4
Canc1	**51.1**	**51.7**	**51.9**	**51.0**	**52.0**	46.4	Pump1	71.6	72.6	72.3	**79.3**	71.0	63.6
Canc2	56.2	58.6	57.6	**63.9**	56.5	56.8	Pump2	66.8	69.5	69.4	**79.2**	68.8	66.2
Conc.2	60.8	69.8	69.5	**80.9**	59.6	50.0	Sonar1	61.8	63.6	63.8	**67.3**	61.0	55.1
Conc.3	79.9	81.9	81.5	**89.5**	79.9	69.8	Sonar2	65.9	67.4	67.4	**71.1**	65.7	61.2
Glass	70.1	**73.9**	**72.5**	**74.9**	69.6	69.0	Spam	50.6	53.7	54.0	**71.5**	50.2	52.7
Hous.	83.4	**93.5**	92.5	**94.2**	82.5	87.2	Spect	84.5	87.2	**87.4**	**87.8**	85.8	84.1
Impor.	65.0	72.3	73.1	**83.4**	63.2	68.2	Vow.4	95.7	**98.5**	98.2	97.9	**98.3**	**98.1**
Ion.	96.3	97.3	97.3	**97.7**	97.3	97.3	Wine	86.6	92.5	91.7	**94.7**	87.4	89.8

Table 4. Results of the Friedman/Nemenyi test. F stands for F-statistic, Signif. for any significant differences, CV for critical value and CD for critical difference.

Base clasf.	Tests				Ranks					
	F	CV	Signif.?	CD	Base	RSM	PRSc	PRSa	Bag	AB
Gauss	16.12	2.32	Yes	1.78	2.78	3.72	3.89	1.61	3.44	5.56
1-NN	32.99	2.32	Yes	1.78	3.94	2.89	3.17	1.06	4.44	5.50
k-Means	40.38	2.32	Yes	1.78	4.83	2.33	2.61	1.44	4.50	5.28