CSCI447: Analysis of Traditional Machine-Learning Algorithms

Christopher R. Barbour, Brandon Fenton, John Sherrill

September 11, 2015

Abstract

Hello, this is our abstract.

1 Introduction

The purpose of this study is to assess the classification performance of 10 machine-learning algorithms on 10 datasets from the UCI Machine Learning Repository. The algorithms, described in Section 2.2, cover a broad spectrum of classification techniques with varying degrees of computational intensity and inductive bias. The latter of which will be the focus of the performance hypothesis for each dataset and of the interpretation of results.

2 Experimental Design

For each of the selected datasets and algorithms, we will randomly partition the examples into a training dataset which will be used to construct the classifier and a test dataset which will be used to test the classifier's ability to generalize, i.e. correctly classify similar examples from the same population of interest. Each algorithm will be heuristically tuned to each dataset prior to the experiment using a single run 10-fold cross-validation scheme.

Three measures were chosen to assess different capabilities of each algorithm:

1. weighted average F-measure: The F-measure for a given class is a function of the precision and recall of a test (the harmonic mean of the precision and recall). The weighted average F-measure is the sum of the F-measures for all the classes in the data weighted by the sample sizes of the examples **(instances?)** in each class.
2. one minus the misclassification rate: Misclassification rates measure the proportion of incorrectly classified observations, so one minus this quantity represents the proportion of correctly classified observations.
3. weighted average Area Under the ROC curve: For a given class, area under the ROC (AUROC) assesses the classifier's sensitivity (probability of predicting the correct class, also referred to as precision) and specificity (probability of not predicting the incorrect class). The weighted AUROC is the sum of the AUROCs over the classes in the data weighted by the sample sizes of the observations **(number of instances in each class?)**.

2.1 Datasets

Ten datasets were obtained from the UCI Machine Learning Repository (Lichman 2013) **(need to organize references)**. Features found to have unique values for each instance (e.g. identification numbers or the name of an animal) were removed. **Not sure if we should include this:** Table 1 displays the name of our datasets with brief descriptions, the types of attributes present, the number of classes for prediction, and the number of total examples.

2.2 Algorithm Descriptions

2.2.1 Instance-Based Methods (IB1 and IBk)

The nearest neighbor algorithm [1] classifies instances by way of referencing a set X of pre-classified instances. For a given instance x of unknown class, the algorithm searches X for an element y that most closely resembles x in attribute. Various metrics may be used for defining "resemblance" although Euclidean distance is most often used. The class of x is then deteremined to be the class of y.

The k-Nearest Neighbor (k-NN) classifier is a generalization of the simple nearest neighbor algorithm. For instance x to be classified, the k-NN algorithm searches a reference set X for the $k \in \mathbb{N}$ elements most closely resembling x. The class of x is assigned as be the most frequent class of the k elements inspected. Modifications to this algorithm include weighting the neighbors based on proximity to the instance being classified, as well as a Parzen windows implementation. We chose to tune this algorithm in the cross-validation tuning procedure over the possible combinations of $k \in \{3, \ldots, 8\}$ and whether to use no distance weighting or the reciprocal of distance as the weighting. We did not utilize the Parzen windows option.

The inductive bias of these instance-based methods lies in the assumption that observations that are close to one another are similar in there class assignment.

2.2.2 Naive Bayes

Naive Bayes estimation [2] assumes conditional independence of the covariates conditioned on the class and applies Bayes rule to get the maximum class probability conditioned on the covariates.

For categorical predictors, we can simply estimate the probability of X conditioned on C is simply the number of examples that have both predictor X and class C divided by the total examples in class C. However, this can result in a probability of 0 if there are no examples in class C with covariate Xi, so we typically assume a dirichlet prior on the distribution on the classes and produce a maximum a posterior (MAP) estimator, which effectively "smooth" these probabilities away from 0. The estimated probability of class C is simply the total number of examples in class C divided by the total number of examples in the data. For continuous input, we can either assume that X_i conditioned on class C_i is a Gaussian random variable and use probability theory to estimate the conditional mean and variance, or we can use a kernel density estimation instead. This reference show that kernel estimation can improve prediction performance when covariates are not conditionally Gaussian and does not hinder performance much when they are. Therefore, we will use kernel density estimation in our computations involving continuous predictors.

The inductive bias of naive bayes estimators is that the classes are linearly separable and that preditors are conditional independent which is often violated in practice.

2.2.3 Logistic Regression with Regularization

Logistic regression (reference) is a form of generalized linear modeling, where we use a logit link function and assume a binomial (or more generally, a multinomial) likelihood. The parameters of logistic regression are estimated using a gradient ascent search method to maximize the conditional log-likelihood from the data. Since this is a gradient search method, it is possible to get stuck at a local maxima instead of the global maximum. This algorithm assumes linear separation of the groups, as well as independence of the covariates. Note: in practice we could attempt to model interaction and quadratic effects of covariates in order to improve prediction performance. Since this option is not readily available in Weka, we will ignore this issue. We can also choose to maximize the log-likelihood plus a penalty, where the penalty corresponds to a constant multiplied by the squared L_2 norm of the model parameters. This corresponds to a "ridge" penalty and will have the effect of shrinking out unimportant parameters in the model. Ridge estimators have been shown to often improve predictive performance in the estimators (reference). For the tuning procedure, the grid of values selected for λ, the ridge penalty, were 0.00001, 0.0001, 0.001, 0.01, 0.1, and 1.

The inductive bias of the logistic regression algorithm comes from its assumption of linear seperability between the classes.

2.2.4 C4.5 Decision Tree (J48)

Decision trees are a type of classifier that creates sets of rules to classify new examples. They begin by finding an initial split that maximally decreases the entropy gain ratio.

After the tree is constructed, it is typically pruned (for a fixed confidence factor, which we will call c).

The inductive bias of decision trees lie in the prefernces for simpler trees versus complicated ones. Decision trees do not need linearly seperable groups in order to be succesful classifiers.

john-These algorithms classify instances by forming a cascading decision making mechanism in the form of a "descision tree". The tree is constructed such that attributes found to carry more information about the class of the instances are used earlier in the mechanism than those attributes found to carry less information.

J48 is an open source Java implementation of Quinlan's C4.5 algorithm. The reference for that particular flavor of Decision tree algorithms is Quinlan, J. R. C4.5: Programs for Machine Learning. Morgan Kaufmann Publishers, 1993.

Inductive biases may include: a preference for shorter trees, trees that place high information gain attributes close to the root are preferred over those that do not, selection of the first functioning tree (the learning algorithm is a greedy algorithm). Linear separability of domain (reference for this statement?)

2.2.5 RIPPER (JRip)

The inductive bias of Ripper K is similar to that of Decision trees, we prefer shorter simpler rules rather than complicated ones. This method does not assume that the groups are linearly seperable.

2.2.6 Support vector machine (LibSVM or SMO)

Support Vector classifiers attempt to find a plane (or hyperplane in more than 3 dimensions) that maximizes the margins separating the groups. Formally...

In cases, where groups are not linearly seperable, a kernal transformation can be used to project the problem into a higher dimensional space where the classes are linearly seperable.

To generalize to 3 or more classes, support vector machines are constructed for each pairwise-class comparison, and the majority winner of each of the pairwise comparison is the class prediction for a new example.

The inductive bias of the support vector machine lies in the assumption that the classes are seperated by a margin (not necessarily in the original dimension).

2.2.7 Feedforward neural networks (Multilayer Perceptron)

These algorithms classify instances by forming a directed graph that contains no cycles, that is, the information passes through the constructed network in a forward only direction. The graph consists of connected nodes (neurons) that either engage or don't based upon signals presented to them from other neurons upstream. The signals are added in a weighted manner and the neuron fires if a specified threshold is met.

Back-propagation is the most common learning method for multi-layer perceptrons. Neurons feed other neurons by manner of weights and ideal weights are found by utilizing gradient descent on a specified loss function where the loss function is minimized over the space of possible weights.

Inductive biases include: starting point for gradient descent optimization, require linear separability of domain.

Backpropagation: D. E. Rumelhart, C. E. Hiton, and R. J. Williams. Nature 323, 533-536, 9 October 1986.

2.2.8 Kernel neural network (RBFNetwork)

An RBF Network is essentially a neural network with that uses radial basis functions as the activation function for neurons.

Broomhead, D. S.; Lowe, David (1988). Radial basis functions, multi-variable functional interpolation and adaptive networks (Technical report). RSRE. 4148.

2.2.9 Ensemble Algorithm (AdaboostM1)

AdaboostM1 is an algorithm built from an ensemble of classifiers. The algorithm is described in detail in (**2 references**). The general idea of boosting is that using an ensemble of different classifiers and taking a majority vote will often produce a more accurate classifier. At each iteration of the algorithm, larger weights w_i are applied to observations who were incorrectly classified by the previous step. This effectively forces the algorithm to put more effort into classifying these observations. A boosting weight is computed at each step which quantifies how well each iteration classifies correctly. The final output takes a "weighted-majority" vote to classify the observations. We selected J48 as our algorithm for boosting. We tuned the number of boosting iterations (using 10, 20, 30, and 40 iterations) and the pruning weights (weights of 0, 50, and 100).

The inductive bias of Adaboost comes from the assumption that a collection of classifiers will perform better than a single classifier, which could be false if we are creating a collection of poor performing classifiers.

2.3 Hypothesis

Which algorithm do we think will work best with each dataset. There are natural coparisons between certain algorithms (e.g. JRip and C4.5).

2.3.1 Abalone

An initial plot of the abalone predictors yeilds a few peices of information. The first being that the covariates share much information with one another. When their is a lot of shared information in the predictors, some methods may have less stable solutions due to multicollinearity. The second peice is that this may not be a completely linearly seperable problem. The plots show fairly distinct separation between infants and adults, but their is not a clear distinction between the males and females. So the algorithms with a linearly seperable inductive bias may do well at separating infants form adults, but not as well from seperating among adults. Based on these observations, we predict that....x

Of course, we are only looking at the pairwise relationships in 2 dimensions, and their may be linear seperation in higher-dimensions.

2.3.2 Car Evaluation

2.3.3 Contraceptive Method

2.3.4 Ecoli

Initial plots of the ecoli data show that while groups can be visually distinugeshed in some dimensions, other are more problematic. We also have a relatively small amount of training data. The linearly separable assumption is more appropriate than the yeast dataset; we see distinct differences in some attributes across the different localization sites. We also have very few examples in some of the classes, making the classificaion of these to be difficult. Given that linear seperability does not seem feasible, we propose that IBk, J48, Ripper, and Adaboost will acheive the best performance amongst the candidates. Since boosting generally increases performance, we predict that his will performa the best among the candidates.

2.3.5 Flag

2.3.6 Tic Tac Toe

2.3.7 Wine

2.3.8 Wine Quality

2.3.9 Yeast

Initial plots of the yeast data show that certain classes can be visually distingueshed in some of the dimensions. Although the linear seperable assumption does not seem to hold, it does appear that observations close to one another seem to have similar classes overall. Although it may not be as efficient as the instance based methods, they rule or tree based methods could potentially separate these groups if appropriately grown and pruned. We postulate that IBK will perform the best amongst the candidates, followed by J48, Ripper, and Adaboost. Since boosting generally performs better than a single instance of a classifier, we predict that the ensamble will have improved performance compard to a single decision tree.

2.3.10 Zoo

When examining the raw data, it seems that we would be able to perform well in this problem. Most of the classes that we are differentiating are almost completely separable in a handful of the attributes present. So it seems to reason that combining them together, we should be able to classify these groups very well. This is also some evidence of a linearly separable problem. Due to this, we hypothesize that our loss measures will be greatest overall for Naive Bayes, Logistic regression, and MultilayerPerceptron. Due to the relatively small amount of training data for this problem, we hypothesize that Naive Bayes will perform the best of these three.

3 Tuning

As stated earlier, individual tuning to each dataset was done prior to the experiment, and these results are displayed in the supplementary information **should this be included?**. For each dataset, the algorithms were tuned using a course grid of tuning parameters and 10-fold cross-validation, selecting the value which minimized the average root mean-squared error of prediction. Some options remained fixed during all of the experiment, and these are discussed below along with certain examples of how tuning was performed. Unless, otherwise stated, the options for the algorithms in Weka were set to the default setting.

4 Results

4.1 Abalone

Since the abalone dataset was not used originally for classifying sex, it makes sense that all classifiers performated rather poolry. The clear winner across all threee measures was logistic regression. In terms of classification percentage, the differences between the algorithms were quite small. For the weighted F-measure, many of the algorithms performed similarly as well. It is interesting to note that the Multilater Perceptron had a very wide variability on this loss scale. The clearest differences in the algorithms are manifested in the weighted AUC measures, with Logistic and Multilayer Perceptron performing the best. Ripper and J48 performed similarly on this dataeset. We see a big increase in using IBk over IB1. Interstingly, Adaboost performed poorly compared to the other algorithms, so the ensemble of trees actually decreased the classification performance in this example.

4.2 Cars

4.3 Contraceptive Use

4.4 Ecoli

The algorithms performed better on the Ecoli dataset compared to the yeast dataset. We see overall higher classifications, F-measures, and AUROCs. We see that, once again, Logistic regression, Naive Bayes, and K nearest neighbors perform at the top of the list. We see a decrease in performance using the rule based methods and the ensamble. The wide variation in these measures could be a result of the overfitting occuring, or the methods inability to easily pick identify classes in linearly seperable problems, something that the top methods can do well.

4.5 Flags

4.6 Tic-tac-toe

4.7 Wine

4.8 Wine Quality

4.9 Yeast

For misclassifcation rate and the Average F-measure, we see small differences between the algorithms, with IB1 performing noticably worse. An interesting result is noticed when examining Adaboost results. We see that Adaboost performs slightly worse than J48 in terms of misclassification rate and average F-measure, but performs slightly better in terms of the AUC. Two of the classes compose 60% of the examples in this dataset, and when looking at individual runs of the algorithm, we see that they perform well at classifying these sites and poorly on the others. In terms of weighted AUC, we see the top performances come from Logistic regression, Naive Bayes, and MultilayerPerceptron, followed closely by the IBK and RBFNetworks.

4.10 Zoo

All algorithms performed quite well on the zoo dataset. This seems to follow our logic; given the amount of information present on the different species it seems reasonable that we would able to classify them into similar groups of species. The classificaiton rate and the weighted F-measure provide practically the same information, and we do not see much differences in the algorithms other than the slight decrease in performance using Ripper and J48, with J48 using performing slightly better. In the weighted AUC, it is not visually clear which algorithm performs best overall, although we see that IBK, NaiveBayes, Logistic, and MultilayerPerceptron all have a median value very close to 1. There was also good performance from RBFNetwork and SMO, but these algorithms had slightly more variation in the observed values, making them slightly less desirable than those stated earlier. The ensamble of trees in Adaboost generally performed slightly better than the single tree produced by J48.

5 Discussion

6 References

1. Hastie, T., Tibshirani, R., Friedman, J. (2009) *The Elements of Statistical Learning: Data Mining, Inference, and Prediction.*. Springer.
2. Kuhn, M., Johnson, K. (2013). *Applied Predictive Modeling.* Springer.
3. Cohen, W. (1995) "Fast Effective Rule Induction". *Twelfth International Conference on Machine Learning.* 115-123.

4. John, G., Langley, P. (1995). "Estimating Continuous Distributions in Bayesian Classifiers". *Eleventh Conference on Uncertainty in Artificial Intelligence*. 338-345.
5. le Cessie, S., van Houwelingen, J.C. (1992). "Ridge Estimators in Logistic Regression." *Applied Statistics*. 41(1), 191-201.
6. Aha, D., Kibler, D. (1991). "Instance-based Learning Algorithms". *Machine Learning*. 6:37-66.
7. Freund, Y., Schapire, R. (1996). "Experiments with a new boosting algorithm". *Thirteenth International Conference on Machine Learning*, 148-156.
8. Warner, B., Misra, M. (1996). "Understanding Neural Networks as Statistical Tools". *The American Statistician*. 50(4):284-293.
9. Lichman, M. (2013). *UCI Machine Learning Repository* [http://archive.ics.uci.edu/ml]. Irvine, CA: University of California, School of Information and Computer Science.
10. D. Aha, D. Kibler (1991). "Instance-based learning algorithms. Machine Learning." 6:37-66.

7 Appendices

Supplementary Information

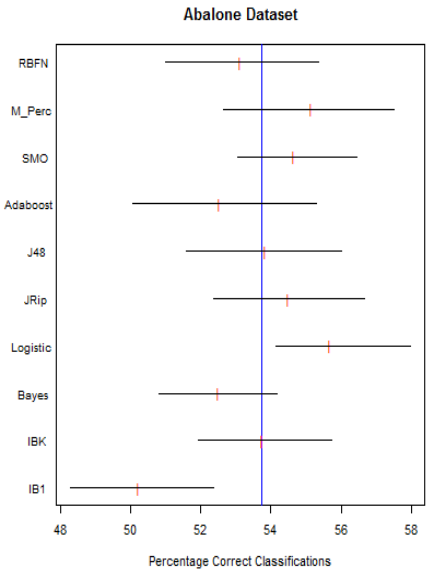
Tuning Parameter Information

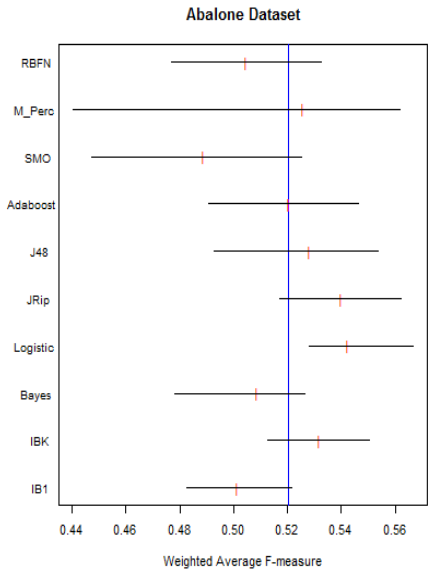
Supplemental Plots of Results

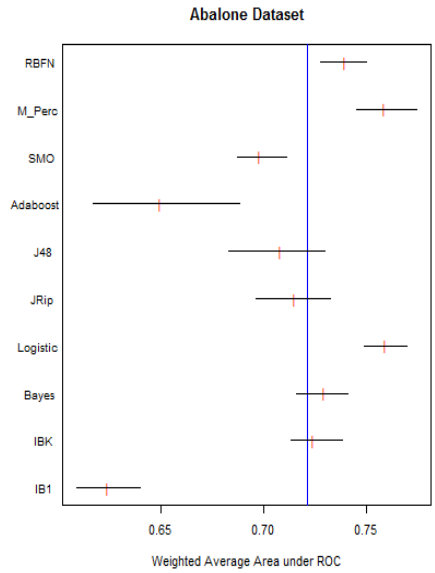
References

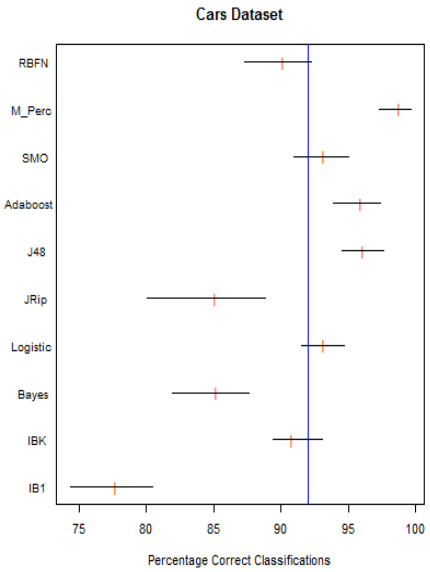
[1] D. Aha, D. Kibler (1991). "Instance-based Learning Algorithms." *Machine Learning*. 6:37-66.

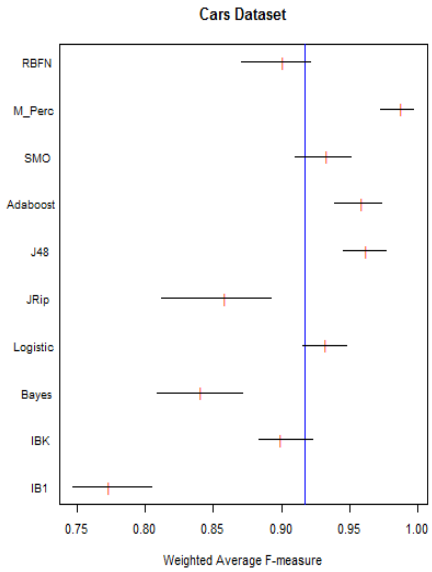
[2] George H. John, Pat Langley (1995). "Estimating Continuous Distributions in Bayesian Classifiers." Eleventh Conference on Uncertainty in Artificial Intelligence, San Mateo, 338-345.

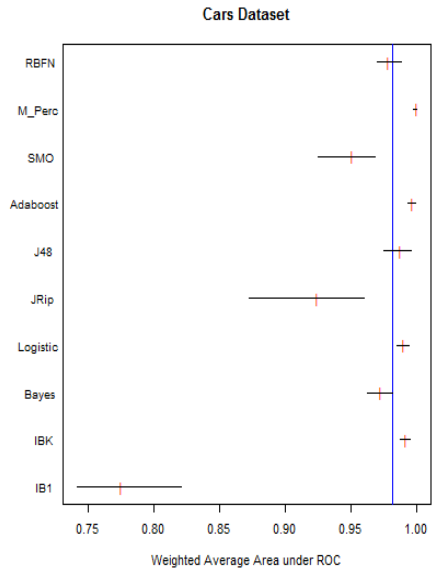

Abalone Dataset
RBFN
M_Perc
SMO
Adaboost
J48
JRip
Logistic
Bayes
IBK
IB1
48
50
52
54
56
58
Percentage Correct Classifications

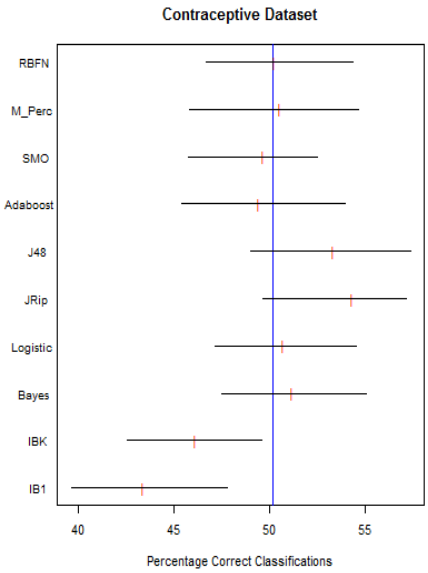

Abalone Dataset
RBFN
M_Perc
SMO
Adaboost
J48
JRip
Logistic
Bayes
IBK
IB1
0.44
0.46
0.48
0.50
0.52
0.54
0.56
Weighted Average F-measure

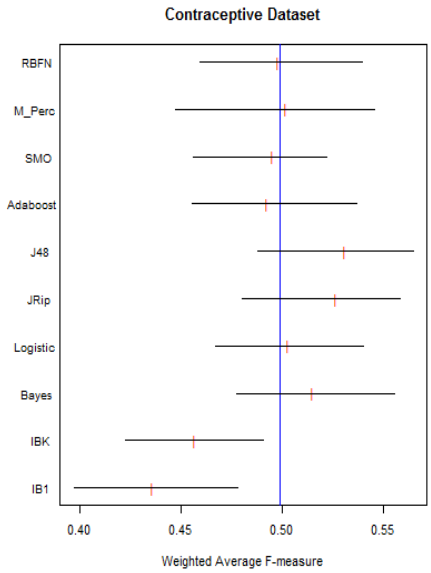

Abalone Dataset
RBFN
M_Perc
SMO
Adaboost
J48
JRip
Logistic
Bayes
IBK
IB1
0.65
0.70
0.75
Weighted Average Area under ROC

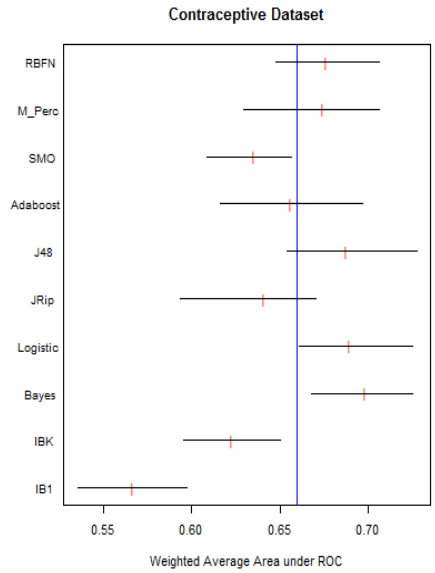

Cars Dataset
RBFN
M_Perc
SMO
Adaboost
J48
JRip
Logistic
Bayes
IBK
IB1
75
80
85
90
95
100
Percentage Correct Classifications

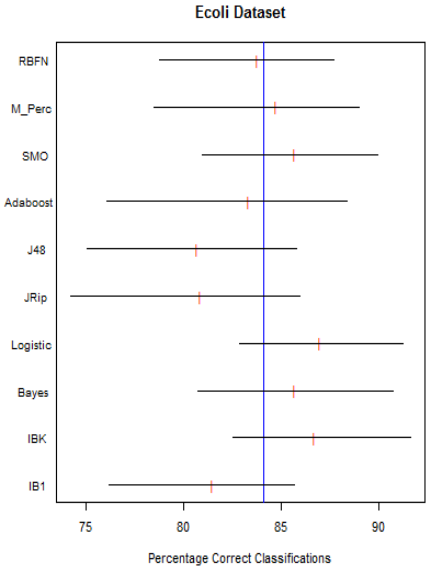

Cars Dataset
RBFN
M_Perc
SMO
Adaboost
J48
JRip
Logistic
Bayes
IBK
IB1
0.75
0.80
0.85
0.90
0.95
1.00
Weighted Average F-measure

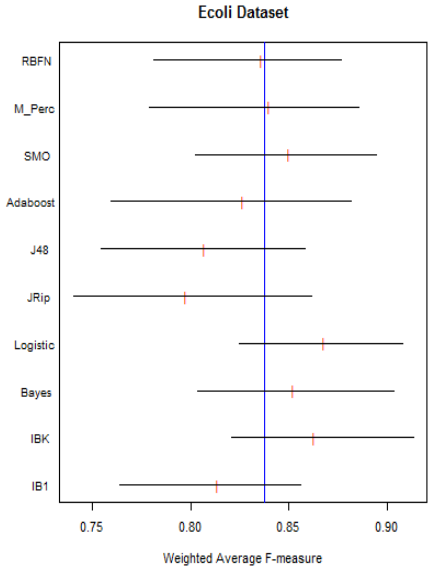

Cars Dataset
RBFN
M_Perc
SMO
Adaboost
J48
JRip
Logistic
Bayes
IBK
IB1
0.75
0.80
0.85
0.90
0.95
1.00
Weighted Average Area under ROC

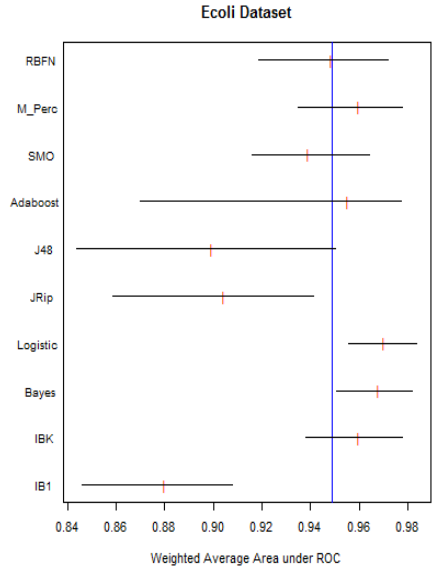

Contraceptive Dataset
RBFN
M_Perc
SMO
Adaboost
J48
JRip
Logistic
Bayes
IBK
IB1
40
45
50
55
Percentage Correct Classifications

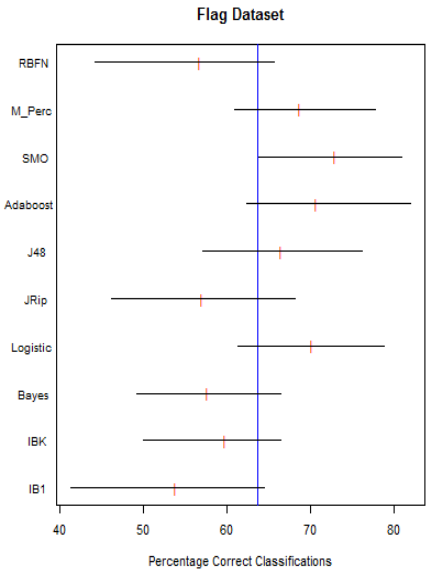

Contraceptive Dataset
RBFN
M_Perc
SMO
Adaboost
J48
JRip
Logistic
Bayes
IBK
IB1
0.40
0.45
0.50
0.55
Weighted Average F-measure

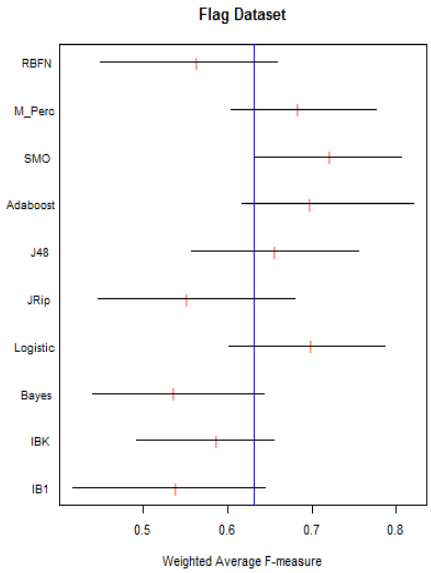

Contraceptive Dataset
RBFN
M_Perc
SMO
Adaboost
J48
JRip
Logistic
Bayes
IBK
IB1
0.55
0.60
0.65
0.70
Weighted Average Area under ROC

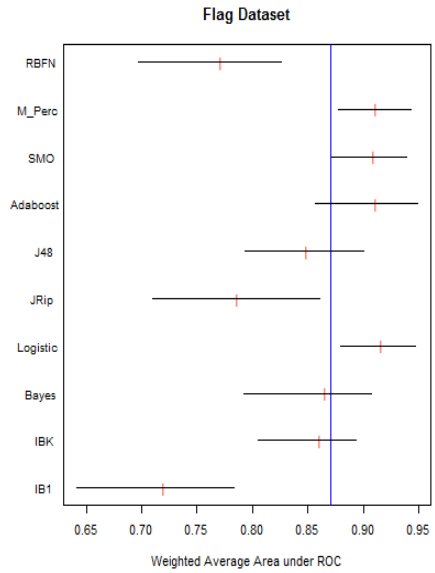

Ecoli Dataset
RBFN
M_Perc
SMO
Adaboost
J48
JRip
Logistic
Bayes
IBK
IB1
75
80
85
90
Percentage Correct Classifications

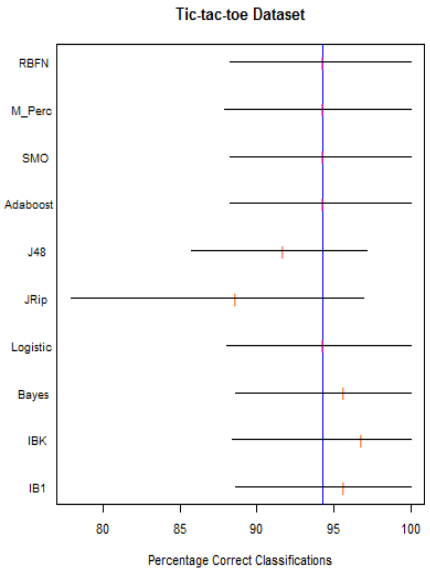

Ecoli Dataset
RBFN
M_Perc
SMO
Adaboost
J48
JRip
Logistic
Bayes
IBK
IB1
0.75
0.80
0.85
0.90
Weighted Average F-measure

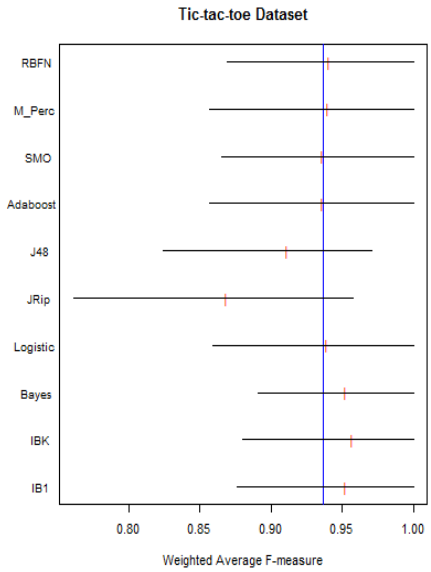

Ecoli Dataset
RBFN
M_Perc
SMO
Adaboost
J48
JRip
Logistic
Bayes
IBK
IB1
0.84
0.86
0.88
0.90
0.92
0.94
0.96
0.98
Weighted Average Area under ROC

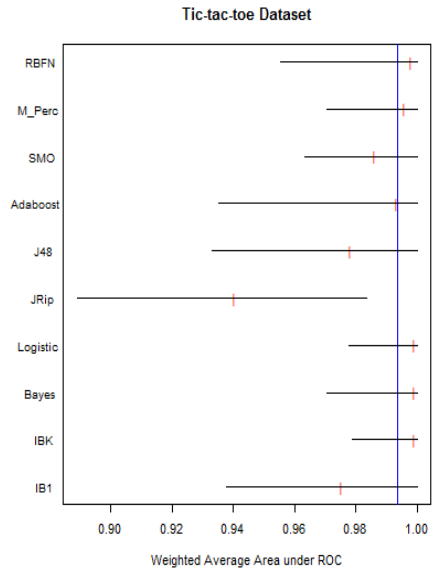

Flag Dataset
RBFN
M_Perc
SMO
Adaboost
J48
JRip
Logistic
Bayes
IBK
IB1
40
50
60
70
80
Percentage Correct Classifications

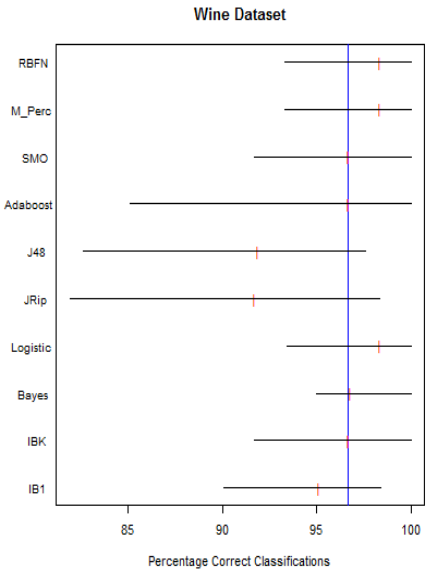

Flag Dataset
RBFN
M_Perc
SMO
Adaboost
J48
JRip
Logistic
Bayes
IBK
IB1
0.5
0.6
0.7
0.8
Weighted Average F-measure

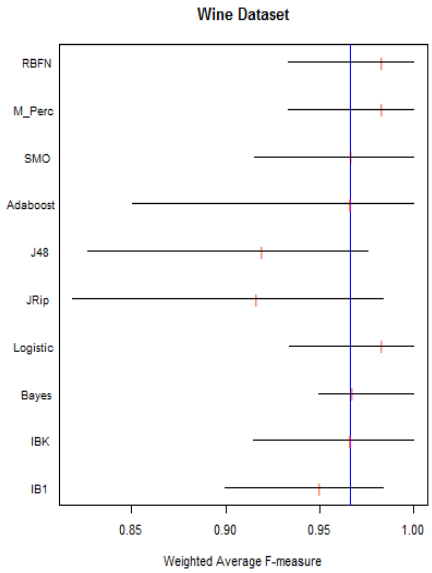

Flag Dataset
RBFN
M_Perc
SMO
Adaboost
J48
JRip
Logistic
Bayes
IBK
IB1
0.65
0.70
0.75
0.80
0.85
0.90
0.95
Weighted Average Area under ROC

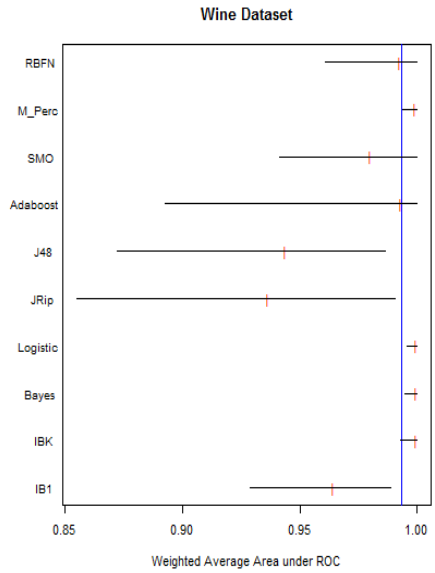

Tic-tac-toe Dataset
RBFN
M_Perc
SMO
Adaboost
J48
JRip
Logistic
Bayes
IBK
IB1
80
85
90
95
100
Percentage Correct Classifications

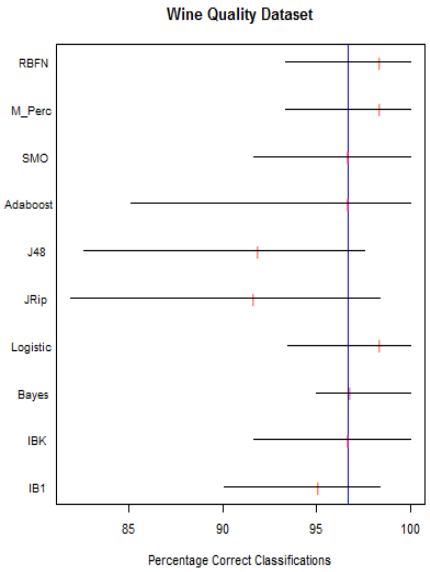

Tic-tac-toe Dataset
RBFN
M_Perc
SMO
Adaboost
J48
JRip
Logistic
Bayes
IBK
IB1
0.80
0.85
0.90
0.95
1.00
Weighted Average F-measure

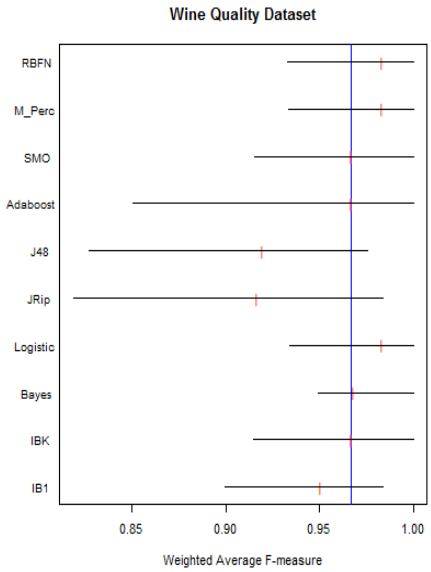

Tic-tac-toe Dataset
RBFN
M_Perc
SMO
Adaboost
J48
JRip
Logistic
Bayes
IBK
IB1
0.90
0.92
0.94
0.96
0.98
1.00
Weighted Average Area under ROC

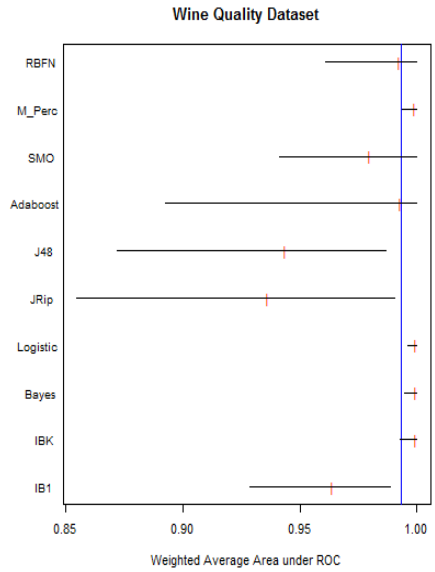

Wine Dataset
RBFN
M_Perc
SMO
Adaboost
J48
JRip
Logistic
Bayes
IBK
IB1
85
90
95
100
Percentage Correct Classifications

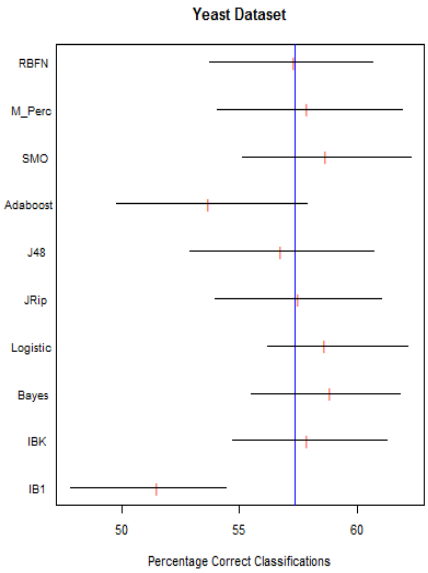

Wine Dataset
RBFN
M_Perc
SMO
Adaboost
J48
JRip
Logistic
Bayes
IBK
IB1
0.85
0.90
0.95
1.00
Weighted Average F-measure

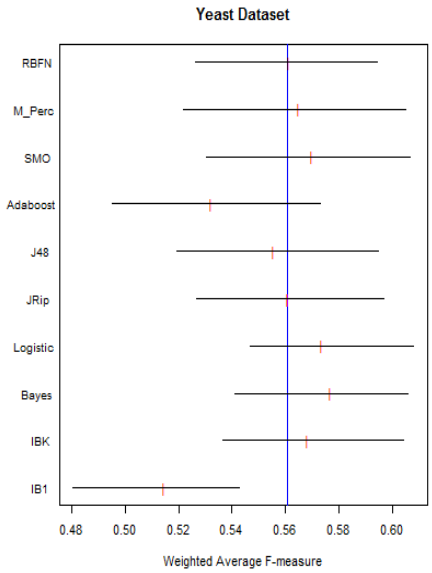

Wine Dataset
RBFN
M_Perc
SMO
Adaboost
J48
JRip
Logistic
Bayes
IBK
IB1
0.85
0.90
0.95
1.00
Weighted Average Area under ROC

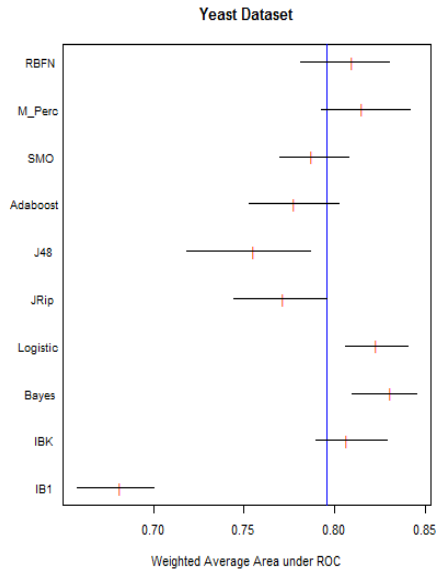

Wine Quality Dataset
RBFN
M_Perc
SMO
Adaboost
J48
JRip
Logistic
Bayes
IBK
IB1
85
90
95
100
Percentage Correct Classifications

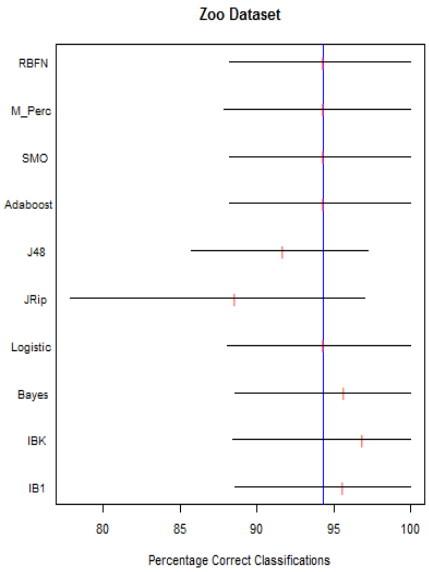

Wine Quality Dataset
RBFN
M_Perc
SMO
Adaboost
J48
JRip
Logistic
Bayes
IBK
IB1
0.85
0.90
0.95
1.00
Weighted Average F-measure

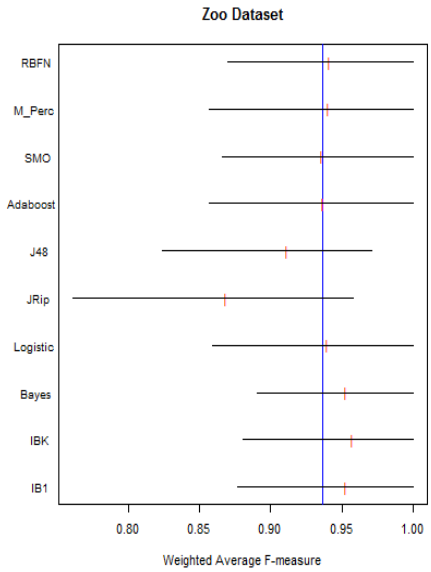

Wine Quality Dataset
RBFN
M_Perc
SMO
Adaboost
J48
JRip
Logistic
Bayes
IBK
IB1
0.85
0.90
0.95
1.00
Weighted Average Area under ROC

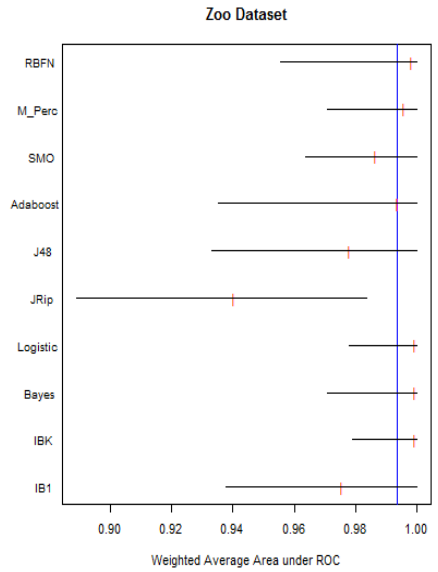

Yeast Dataset
RBFN
M_Perc
SMO
Adaboost
J48
JRip
Logistic
Bayes
IBK
IB1
50
55
60
Percentage Correct Classifications


Yeast Dataset
RBFN
M_Perc
SMO
Adaboost
J48
JRip
Logistic
Bayes
IBK
IB1
0.48
0.50
0.52
0.54
0.56
0.58
0.60
Weighted Average F-measure


Yeast Dataset
RBFN
M_Perc
SMO
Adaboost
J48
JRip
Logistic
Bayes
IBK
IB1
0.70
0.75
0.80
0.85
Weighted Average Area under ROC


Zoo Dataset
RBFN
M_Perc
SMO
Adaboost
J48
JRip
Logistic
Bayes
IBK
IB1
80
85
90
95
100
Percentage Correct Classifications


Zoo Dataset
RBFN
M_Perc
SMO
Adaboost
J48
JRip
Logistic
Bayes
IBK
IB1
0.80
0.85
0.90
0.95
1.00
Weighted Average F-measure


Zoo Dataset
RBFN
M_Perc
SMO
Adaboost
J48
JRip
Logistic
Bayes
IBK
IB1
0.90
0.92
0.94
0.96
0.98
1.00
Weighted Average Area under ROC