Multiple classifiers

JERZY STEFANOWSKI
Institute of Computing Sciences
Poznań University of Technology

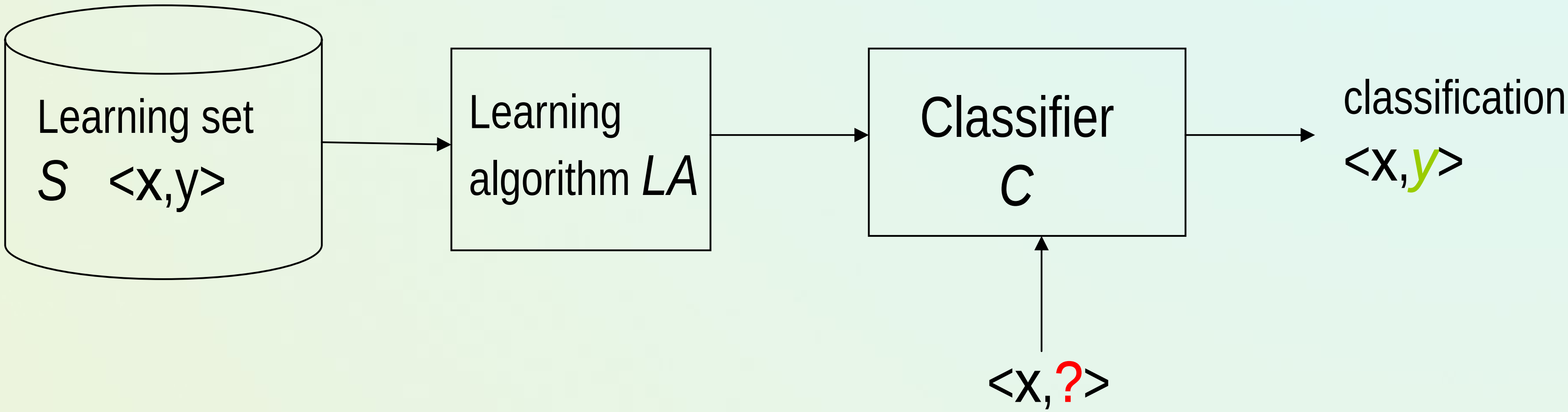
Zajęcia dla TPD - ZED 2009
Oparte na wykładzie dla
Doctoral School , Catania-Troina, April, 2008

Outline of the presentation

1. Introduction
2. Why do multiple classifiers work?
3. Stacked generalization – combiner.
4. Bagging approach
5. Boosting
6. Feature ensemble
7. Pairwise coupling $\rightarrow$ n^2 classifier for multi-class problems

Machine Learning and Classification

Classification - assigning a decision class label to a set of objects described by a set of attributes



Set of learning examples S = $\{\langle \mathbf{x}_1, y_1\rangle, \langle \mathbf{x}_2, y_2\rangle, \cdots, \langle \mathbf{x}_n, y_n\rangle\}$

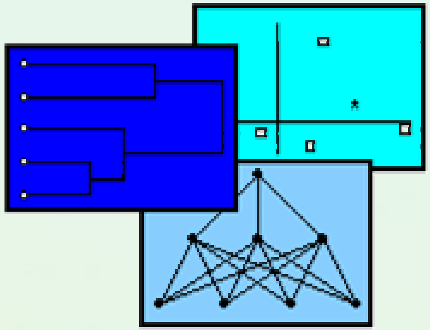
for some unknown classification function f : $y = f(\mathbf{x})$

$\mathbf{x}_i = \langle x_{i1}, x_{i2}, \ldots, x_{im} \rangle$ example described by m attributes

y – class label; value drawn from a discrete set of classes $\{Y_1, \ldots, Y_K\}$

Approaches to learn single classifiers

- Decision Trees
- Rule Approaches
- Logical statements (ILP)
- Bayesian Classifiers
- Neural Networks
- Discriminant Analysis
- Support Vector Machines
- k-nearest neighbor classifiers
- Logistic regression
- Artificial Neural Networks
- Genetic Classifiers
- …



Why could we integrate classifiers?

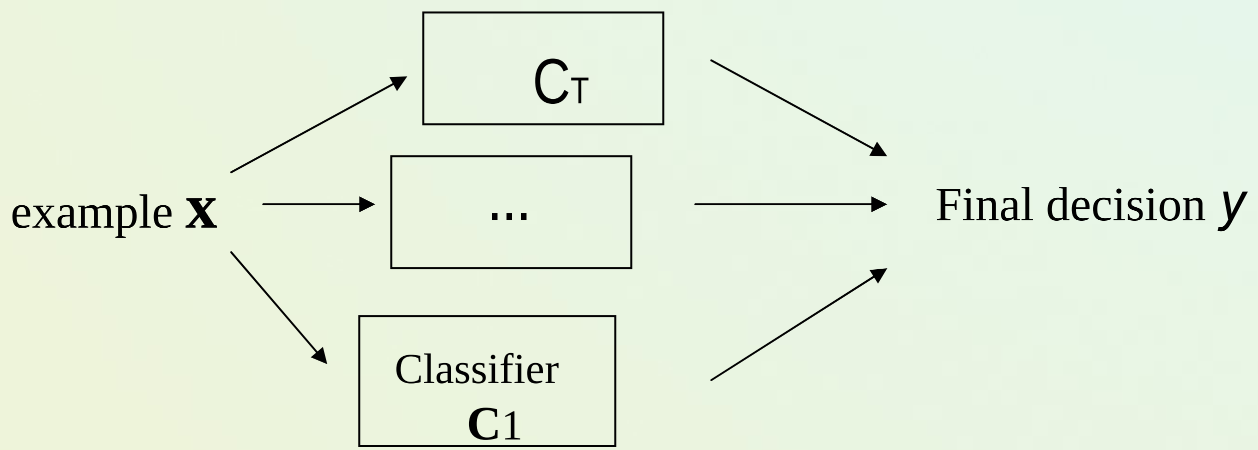
- Typical research → create and evaluate a single learning algorithm; compare performance of some algorithms.
- Empirical observations or applications → a given algorithm may outperform all others for a specific subset of problems
  - There is no one algorithm achieving the best accuracy for all situations! [No free lunch!]
- A complex problem can be decomposed into multiple sub-problems that are easier to be solved.
- Growing research interest in combining a set of learning algorithms / classifiers into one system

„Multiple learning systems try to exploit the local different behavior of the base learners to enhance the accuracy of the overall learning system"

- G. Valentini, F. Masulli

Multiple classifiers - definitions

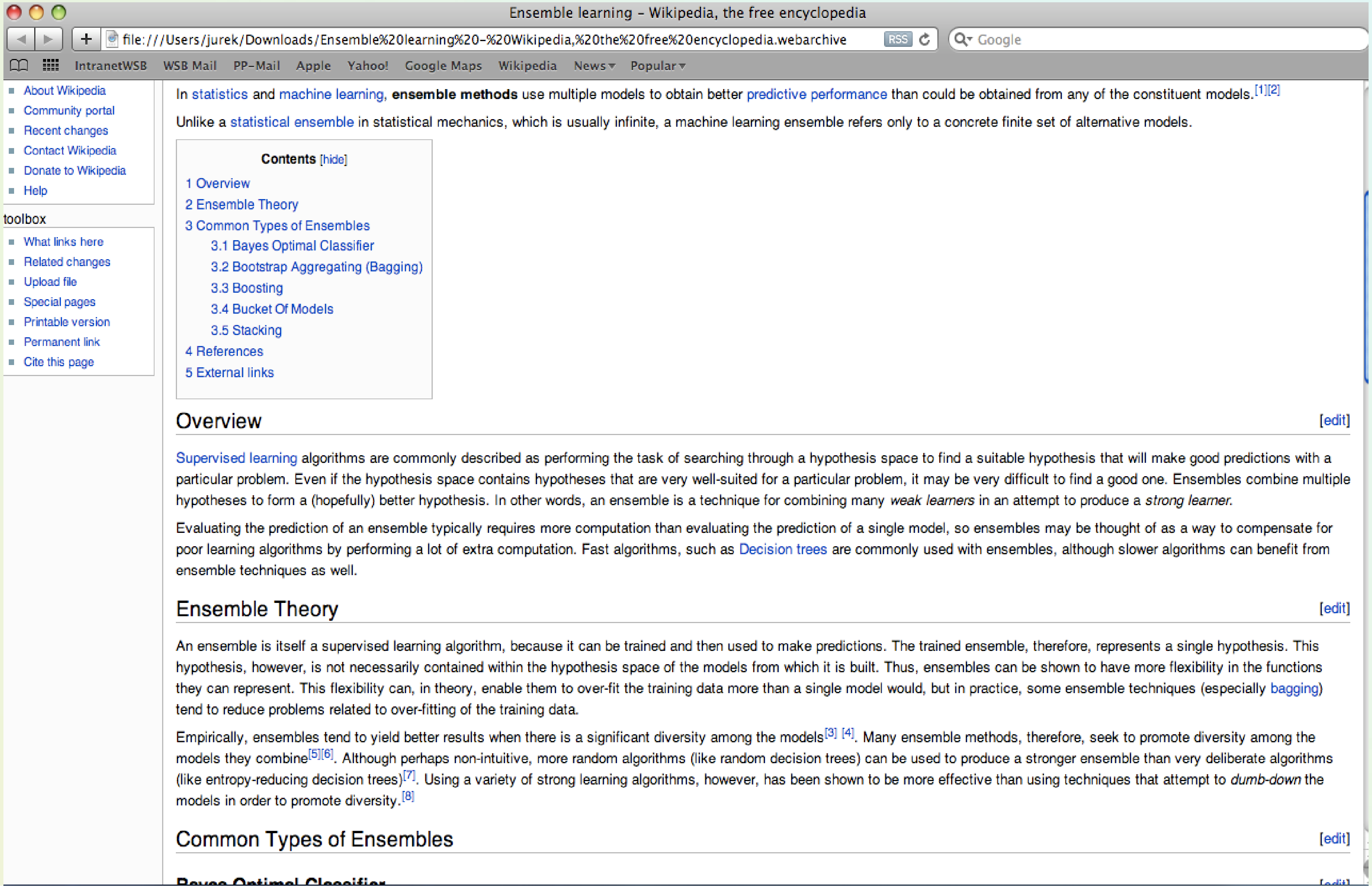
- Multiple classifier – a set of classifiers whose individual predictions are combined in some way to classify new examples.
- Various names: ensemble methods, committee, classifier fusion, combination, aggregation,…
- Integration should improve predictive accuracy.



Multiple classifiers – review studies

- Relatively young research area – since the 90's
- A number of different proposals or application studies
- Some review papers or book:
  - L.Kuncheva, Combining Pattern Classifiers: Methods and Algorithms, 2004 (large review + list of bibliography).
  - T.Dietterich, Ensemble methods in machine learning, 2000.
  - J.Gama, Combining classification algorithms, 1999.
  - G.Valentini, F.Masulli, Ensemble of learning machines, 2001 [exhaustive list of bibliography].
  - J.Kittler et al., On combining classifiers, 1998.
  - J.Kittler et al. (eds), Multiple classifier systems, Proc. of MCS Workshops, 2000, … ,2003.
  - See also many papers by L.Breiman, J.Friedman, Y.Freund, R.Schapire, T.Hastie, R.Tibshirani,

Other less reputable resources



Ensemble learning - Wikipedia, the free encyclopedia

file:///Users/jurek/Downloads/Ensemble%20learning%20-%20Wikipedia,%20the%20free%20encyclopedia.webarchive

IntranetWSB WSB Mail PP-Mail Apple Yahoo! Google Maps Wikipedia News Popular

- About Wikipedia
- Community portal
- Recent changes
- Contact Wikipedia
- Donate to Wikipedia
- Help

toolbox

- What links here
- Related changes
- Upload file
- Special pages
- Printable version
- Permanent link
- Cite this page

In statistics and machine learning, **ensemble methods** use multiple models to obtain better predictive performance than could be obtained from any of the constituent models.[1][2]

Unlike a statistical ensemble in statistical mechanics, which is usually infinite, a machine learning ensemble refers only to a concrete finite set of alternative models.

Contents [hide]

1 Overview
2 Ensemble Theory
3 Common Types of Ensembles
3.1 Bayes Optimal Classifier
3.2 Bootstrap Aggregating (Bagging)
3.3 Boosting
3.4 Bucket Of Models
3.5 Stacking
4 References
5 External links

Overview [edit]

Supervised learning algorithms are commonly described as performing the task of searching through a hypothesis space to find a suitable hypothesis that will make good predictions with a particular problem. Even if the hypothesis space contains hypotheses that are very well-suited for a particular problem, it may be very difficult to find a good one. Ensembles combine multiple hypotheses to form a (hopefully) better hypothesis. In other words, an ensemble is a technique for combining many *weak learners* in an attempt to produce a *strong learner*.

Evaluating the prediction of an ensemble typically requires more computation than evaluating the prediction of a single model, so ensembles may be thought of as a way to compensate for poor learning algorithms by performing a lot of extra computation. Fast algorithms, such as Decision trees are commonly used with ensembles, although slower algorithms can benefit from ensemble techniques as well.

Ensemble Theory [edit]

An ensemble is itself a supervised learning algorithm, because it can be trained and then used to make predictions. The trained ensemble, therefore, represents a single hypothesis. This hypothesis, however, is not necessarily contained within the hypothesis space of the models from which it is built. Thus, ensembles can be shown to have more flexibility in the functions they can represent. This flexibility can, in theory, enable them to over-fit the training data more than a single model would, but in practice, some ensemble techniques (especially bagging) tend to reduce problems related to over-fitting of the training data.

Empirically, ensembles tend to yield better results when there is a significant diversity among the models[3] [4]. Many ensemble methods, therefore, seek to promote diversity among the models they combine[5][6]. Although perhaps non-intuitive, more random algorithms (like random decision trees) can be used to produce a stronger ensemble than very deliberate algorithms (like entropy-reducing decision trees)[7]. Using a variety of strong learning algorithms, however, has been shown to be more effective than using techniques that attempt to *dumb-down* the models in order to promote diversity.[8]

Common Types of Ensembles [edit]

Multiple classifiers – why do they work?

- How to create such systems and when they may perform better than their components used independently?
- Combining identical classifiers is useless!

A necessary condition for the approach to be useful is that member classifiers should have a substantial level of disagreement, i.e., they make error independently with respect to one another

- Conclusions from some studies (e.g. Hansen&Salamon90, Ali&Pazzani96):
  Member classifiers should **make uncorrelated errors** with respect to one another; each classifier should perform better than a random guess.

Improving performance with respect to a single classifier

- An example of binary classification (50% each class), classifiers have the same error rate and make errors independently; final classification by uniform voting → the expected error of the system should decrease with the number of classifiers

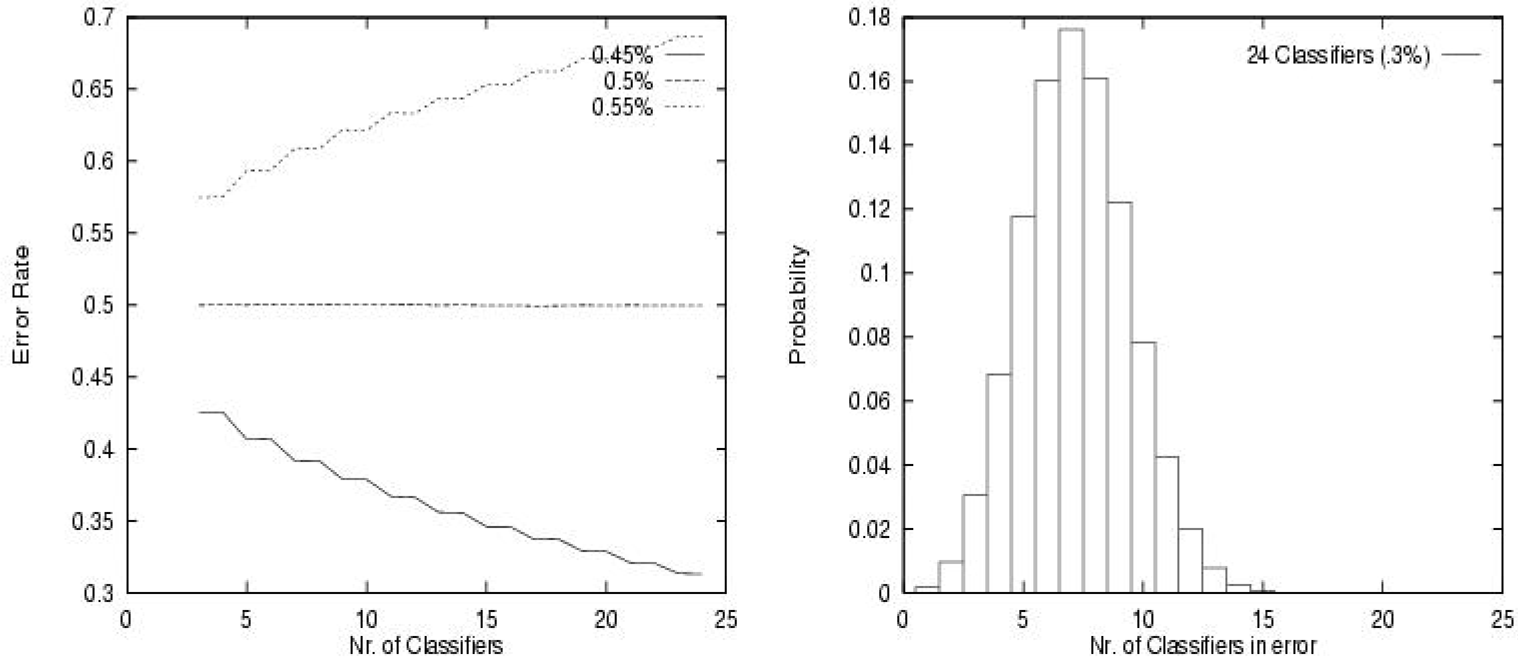


Figure 5.1: (a) Error rate versus nr. of classifiers in an ensemble. (b) Probability that exactly n of 24 classifiers will make an error.

Diversification of classifiers - intuition

Two classifiers are diverse, if they make different errors on a new object

Assume a set of three classifiers $\{h_1,h_2,h_3\}$ and a new object $\mathbf{x}$

- If all are identical, then when $h_1(\mathbf{x})$ is wrong, $h_2(\mathbf{x})$ and $h_3(\mathbf{x})$ will be also wrong (making the same decision)
- If the classifier errors are uncorrelated, then when $h_1(\mathbf{x})$ is wrong, $h_2(\mathbf{x})$ and $h_3(\mathbf{x})$ may be correct $\rightarrow$ a majority vote will correctly classify $\mathbf{x}$!

Dietterich's reasons why multiple classifier may work better…

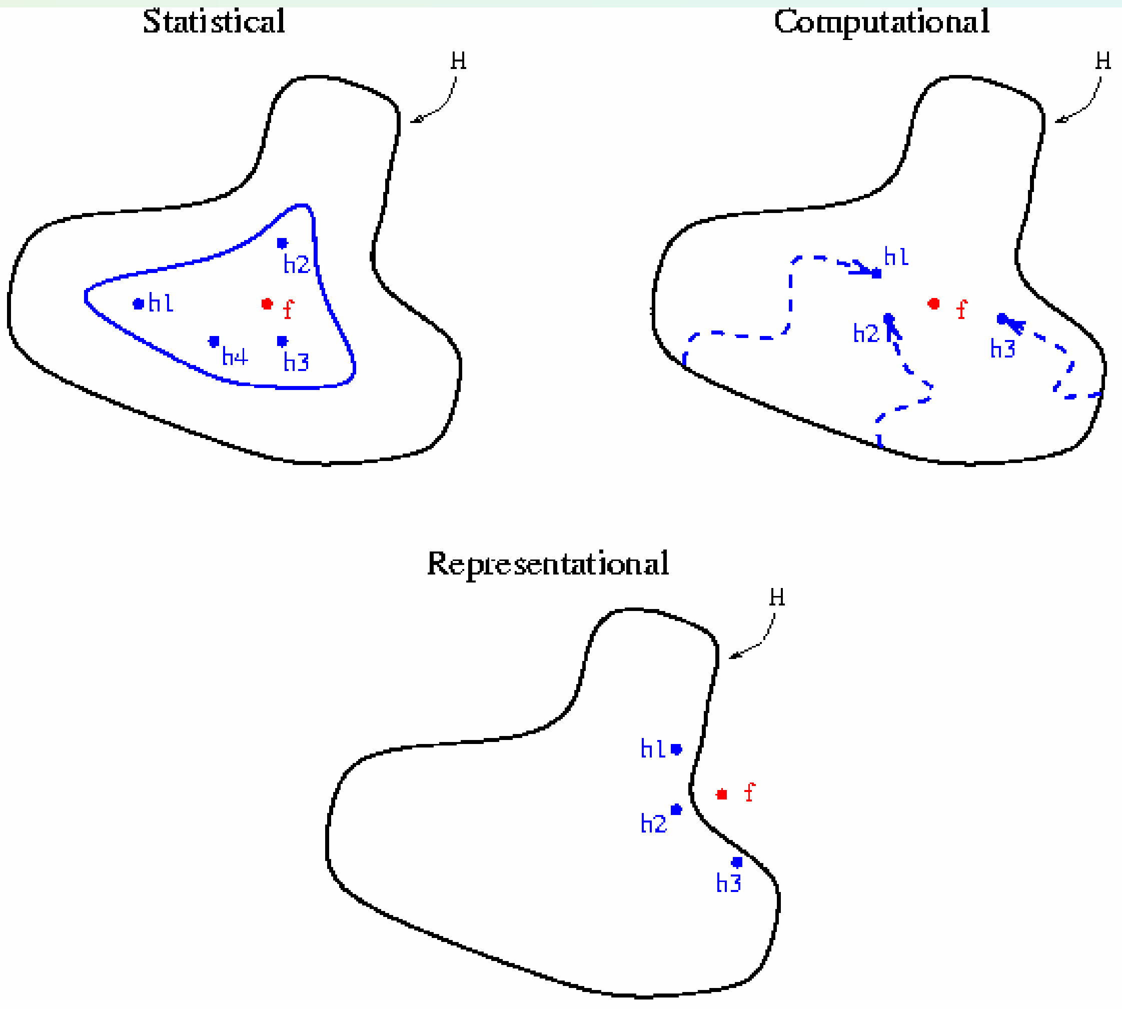


Fig. 2. Three fundamental reasons why an ensemble may work better than a single classifier

Combing classifier predictions

- Intuitions:
  - Utility of combining diverse, independent opinions in human decision-making
- Voting vs. non-voting methods
  - Counts of each classifier are used to classify a new object
  - The vote of each classifier may be weighted, e.g., by measure of its performance on the training data. (Bayesian learning interpretation).
- Non-voting $\rightarrow$ output classifiers (class-probabilities or fuzzy supports instead of single class decision)
  - Class probabilities of all models are aggregated by specific rule (product, sum, min, max, median,…)
  - More complicated $\rightarrow$ extra meta-learner

Group or specialized decision making

- Group (static) – all base classifiers are consulted to classify a new object.
- Specialized / dynamic integration – some base classifiers performs poorly in some regions of the instance space
  - So, select only these classifiers whose are „expertised” (more accurate) for the new object

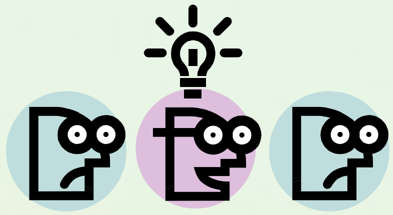
Dynamic voting of sub-classifiers

Change the way of aggregating predictions from sub-classifiers!

- Standard $\rightarrow$ equal weight voting.

Dynamic voting:

- For a new object to be classified:
  - Find its *h*-nearest neighbors in the original learning set.
  - Reclassify them by all sub-classifiers.
  - Use weighted voting, where a sub-classifier weight corresponds to its accuracy on the *h*-nearest neighbors.



Diversification of classifiers

- Different training sets (different samples or splitting,..)
- Different classifiers (trained for the same data)
- Different attributes sets

  (e.g., identification of speech or images)
- Different parameter choices

  (e.g., amount of tree pruning, BP parameters, number of neighbors in KNN,…)
- Different architectures (like topology of ANN)
- Different initializations

Different approaches to create multiple systems

- **Homogeneous classifiers** – use of the same algorithm over diversified data sets
  - Bagging (Breiman)
  - Boosting (Freund, Schapire)
  - Multiple partitioned data
  - Multi-class specialized systems, (e.g. ECOC pairwise classification)
- **Heterogeneous classifiers** – different learning algorithms over the same data
  - Voting or rule-fixed aggregation
  - Stacked generalization or meta-learning

Stacked generalization [Wolpert 1992]

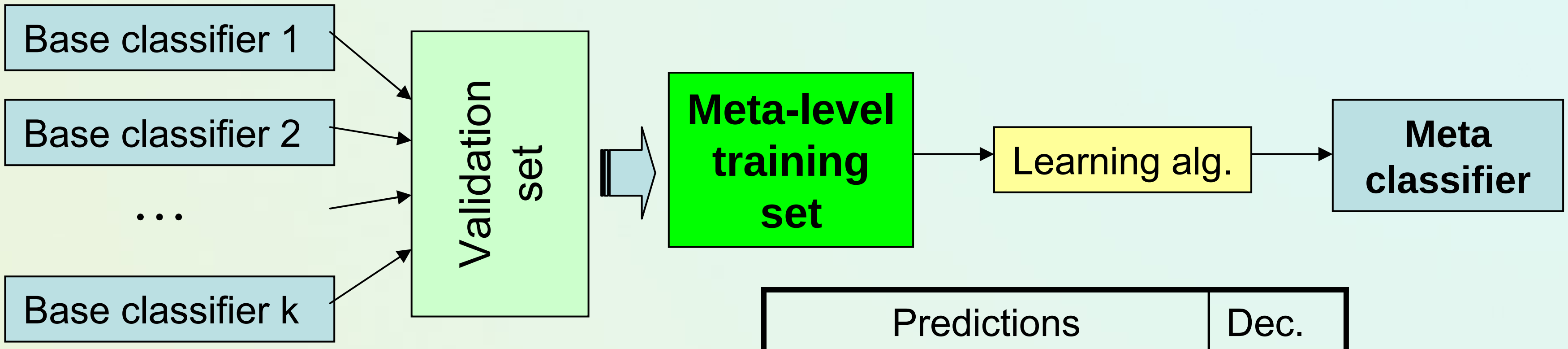
- Use meta learner instead of averaging to combine predictions of base classifiers.
    - Predictions of base learners (*level-0 models*) are used as input for meta learner (*level-1 model)*
- Method for generating base classifiers usually apply different learning schemes.
- Hard to analyze theoretically.

The Combiner - 1

Chan & Stolfo : *Meta-learning*.

- Two-layered architecture:
  - 1-level – base classifiers.
  - 2-level – meta-classifier.
- Base classifiers created by applying the different learning algorithms to the same data.

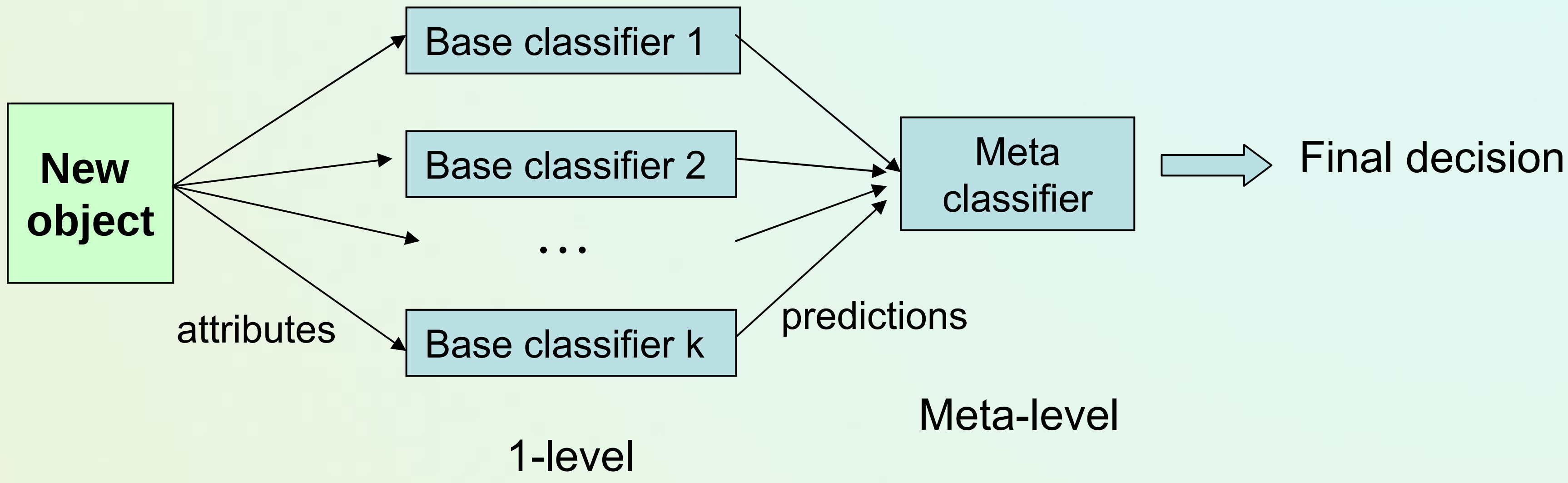
Learning the meta-classifier



Predictions				Dec. class
Cl.1	Cl.2	…	Cl.K	
A	A	…	B	**A**
A	B	…	C	**B**

- Predictions of base classifiers on an extra validation set (not directly training set – apply „internal” cross validation) with correct class decisions → a meta-level training set.
- An extra learning algorithm is used to construct a meta-classifiers.
- The idea → a meta-classifier attempts to learn relationships between predictions and the final decision;
  It may correct some mistakes of the base classifiers.

The Combiner - 2



Classification of a new instance by the combiner

- Chan & Stolfo [95/97] : experiments that their combiner ({CART,ID3,K-NN}→NBayes) is better than equal voting.

More on stacking

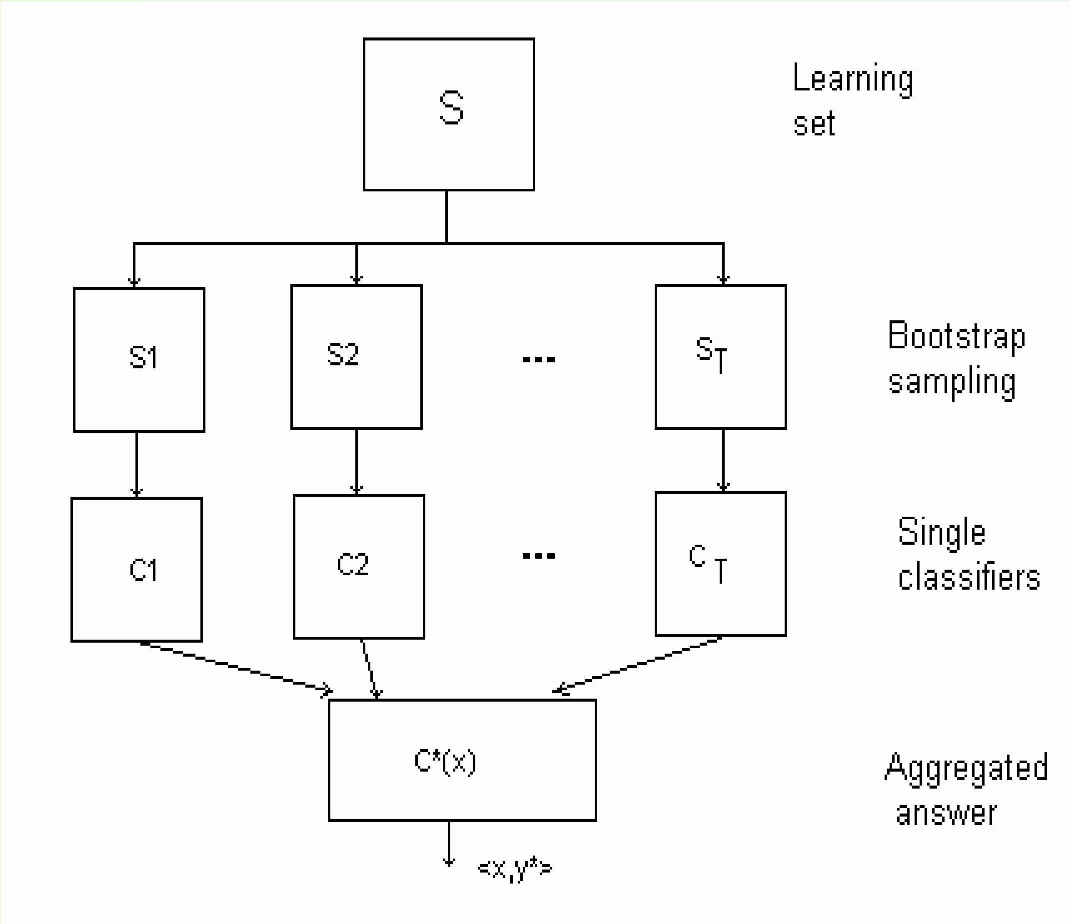
- Other 1-level solutions: use additional attribute descriptions, introduce an arbiter instead of simple meta-combiner.
- If base learners can output probabilities it’s better to use those as input to meta learner
- Which algorithm to use to generate meta learner?
  - In principle, any learning scheme can be applied
  - David Wolpert:
    - Base learners do most of the work
    - Reduces risk of overfitting
- Relationship to more complex approaches: SCANN [Mertz] create a new attribute space for the metalearning.

Bagging [L.Breiman, 1996]

- Bagging = **B**ootstrap **agg**regation
  - Generates individual classifiers on bootstrap samples of the training set
- As a result of the sampling-with-replacement procedure, each classifier is trained on the average of 63.2% of the training examples.
  - For a dataset with N examples, each example has a probability of $1-(1-1/N)^N$ of being selected at least once in the N samples. For $N \to \infty$, this number converges to (1-1/e) or 0.632 [Bauer and Kohavi, 1999]
- Bagging traditionally uses component classifiers of the same type (e.g., decision trees), and combines prediction by a simple majority voting across.

More about „Bagging”

- Bootstrap aggregating – L.Breiman [1996]



input S – learning set, T – no. of bootstrap samples, LA – learning algorithm

output C* - multiple classifier

for i=1 **to** T **do**

begin

S_i:=bootstrap sample from S;

C_i:=$LA(S_i)$;

end;

$$C^*(x) = \mathrm{argmax}_y \sum_{i=1}^{T} (C_i(x) = y)$$

Bagging Empirical Results

Misclassification error rates [%] $\rightarrow$ CART trees

Data	Single	Bagging	Decrease
waveform	29.0	19.4	33%
heart	10.0	5.3	47%
breast cancer	6.0	4.2	30%
ionosphere	11.2	8.6	23%
diabetes	23.4	18.8	20%
glass	32.0	24.9	22%
soybean	14.5	10.6	27%

Breiman “Bagging Predictors” Berkeley Statistics Department TR#421, 1994

Bagging – how does it work?

- Related works – experiments Breiman [96], Quinlan [96], Bauer&Kohavi [99]; Conclusion – bagging improves accuracy for decision trees.
- The perturbation in the training set due to the bootstrap re+sampling causes different base classifiers to be built, particularly if the classifier is unstable
- Breiman says that this approach works well for unstable algorithms:
  - Whose major output classifier undergoes major changes in response to small changes in learning data.
- Bagging can be expected to improve accuracy if the induced classifiers are uncorrelated!

Experiments with rules

- The single use of the MODLEM induced classifier is compared against bagging classifier (composed of rule sub-classifiers - also induced by MODLEM)
- Comparative studies on 18 datasets. Predictive accuracy evaluated by 10-fold cross-validation (stratified or random)
- An analysis of the change parameter *T* (number of sub-classifiers) on the performance of the bagging classifier

Comparing classifiers

Dataset	Single	Bagging - with different T			
	MODLEM	3	5	7	10
bank	93.81 ± 0.94	95.05 ± 0.91	94.95 ± 0.84	95.22 ± 1.02	93.95* ± 0.9
buses	97.20 ± 0.94	98.05* ± 0.97	99.54 ± 1.09	97.02* ± 1.15	97.45* ± 1.1
zoo	94.64 ± 0.67	93.82* ± 0.68	93.89* ± 0.71	93.47 ± 0.73	93.68 ± 0.70
hsv	54.52 ± 1.05	64.75 ± 1.21	65.94 ± 0.69	64.78 ± 0.57	64.53 ± 0.55
hepatitis	78.62 ± 0.93	82.00 ± 1.14	84.05 ± 1.1	81.05 ± 0.97	84.0 ± 0.49
iris	94.93 ± 0.5	95.13* ± 0.46	94.86* ± 0.54	95.06* ± 0.53	94.33* ± 0.5
auto	85.23 ± 1.1	82. 98 ± 0.86	83.0 ±0.99	82.74 ±0.9	81.39 ± 0.84
segmentation	85.71 ± 0.71	86.19* ± 0.82	87.62 ± 0.55	87.61 ± 0.46	87.14 ± 0.9
glass	72.41 ± 1.23	68.5 ± 1.15	74.81 ± 0.94	74.25 ± 0.89	76.09 ± 0.68
bricks	90.32* ± 0.82	90.3 * ± 0.54	89.84* ± 0.65	91.21* ± 0.48	90.77* ± 0.7
vote	92.67 ± 0.38	93.33* ± 0.5	94.34 ± 0.34	95.01 ± 0.44	96.01 ± 0.29
bupa	65.77 ± 0.6	64.98* ± 0.76	76.28 ± 0.44	70.74 ± 0.96	75.69 ± 0.7
election	88.96 ± 0.54	90.3 ± 0.36	91.2 ± 0.47	91.66 ± 0.34	90.75 ± 0.55
urolog1	62.4 ± 0.51	65.2 ± 0.25	63.1* ± 0.5	65.8 ± 0.35	65.2 ± 0.34
urolog2	63.80 ± 0.73	64.8 ± 0.83	65.0 ± 0.43	67.40 ± 0.46	67.0 ± 0.67
german	72.16 ± 0.27	73.07* ± 0.39	76.2 ± 0.34	75.62 ± 0.34	75.75 ± 0.35
crx	84.64 ± 0.35	84.74* ± 0.38	86.24 ± 0.39	87.1 ± 0.46	89.42 ± 0.44
pima	73.57 ± 0.67	75.78* ± 0.6	74.35* ± 0.64	74.88 ± 0.44	77.87 ± 0.39

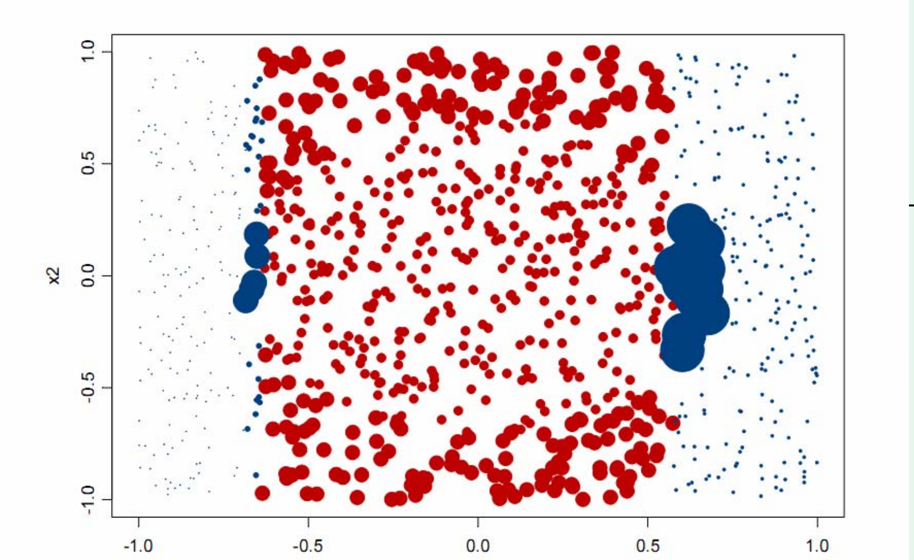
Classification accuracy [%] – average over 10 f-c-v with standard deviations; Asterik – difference is not significant α =0.05

Boosting [Schapire 1990; Freund & Schapire 1996]

- In general takes a different weighting schema of resampling than bagging.
- Freund & Schapire: theory for “weak learners” in late 80’s
- **Weak Learner**: performance on ***any*** train set is slightly better than chance prediction
  - Schapire has shown that a weak learner can be converted into a strong learner by changing the distribution of training examples
- Iterative procedure:
  - The component classifiers are built sequentially, and examples that are misclassified by previous components are chosen more often than those that are correctly classified!
  - So, new classifiers are influenced by performance of previously built ones. New classifier is encouraged to become expert for instances classified incorrectly by earlier classifier.
- There are several variants of this algorithm – **AdaBoost** the most popular (see also arcing).

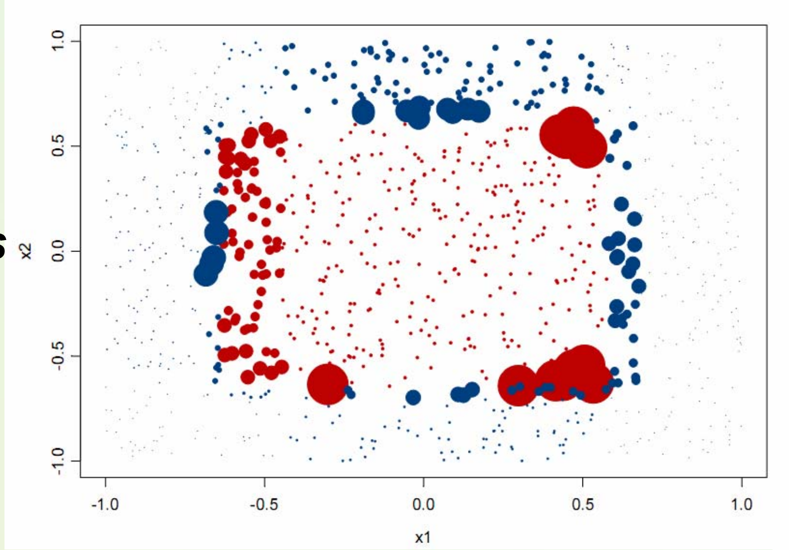
AdaBoost

- Weight all training examples equally (1/n)
- Train model (classifier) on train sample D_i
- Compute error e_i of model on train sample D_i
- A new training sample D_{i+1} is produced by decreasing the weight of those examples that were correctly classified (multiple by $e_i/(1-e_i)$)), and increasing the weight of the misclassified examples.
- Normalize weights of all instances.
- Train new model on re-weighted train set
- Re-compute errors on weighted train set
- The process is repeated until (# iterations or error stopping)
- Final model: weighted prediction of each classifier
  - Weight of class predicted by component classifier $\log(e_i/(1-e_i))$

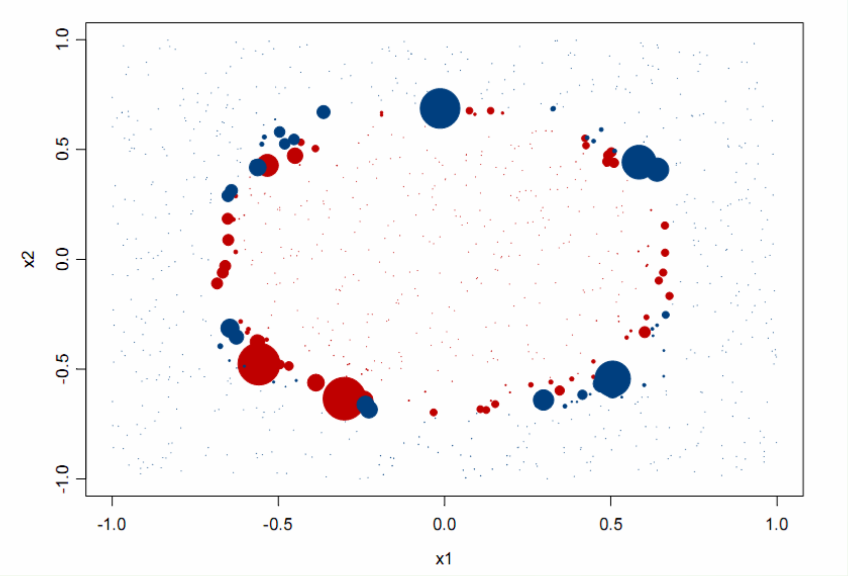


Classifications (colors) and Weights (size) after *1 iteration* Of AdaBoost

3 iterations



20 iterations



from Elder, John. From Trees to Forests and Rule Sets - A Unified Overview of Ensemble Methods. 2007.

Remarks on Boosting

- Boosting can be applied without weights using re-sampling with probability determined by weights;
  - Example weights might be harder to deal with some algorithms or packages.
  - Draw a bootstrap sample from the data with the probability of drawing each example is proportional to it's weight
- Boosting should decrease exponentially the training error in the number of iterations;
- Boosting works well if base classifiers are not too complex and their error doesn't become too large too quickly!

Boosting vs. Bagging with C4.5 [Quinlan 96]

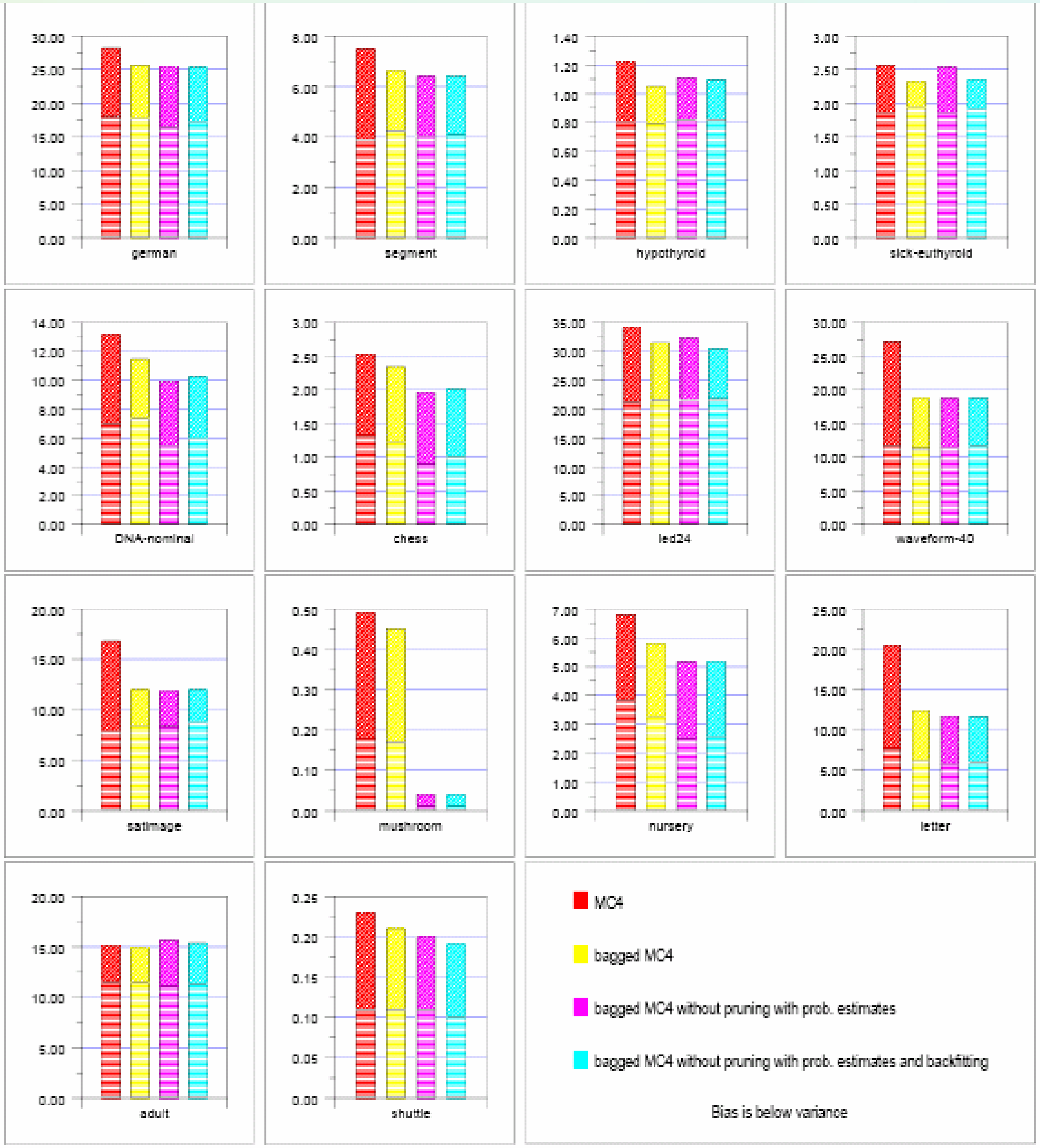
	C4.5	Bagged C4.5 vs C4.5			Boosted C4.5 vs C4.5			Boosting vs Bagging	
	err (%)	err (%)	w-l	ratio	err (%)	w-l	ratio	w-l	ratio
anneal	7.67	6.25	10-0	.814	4.73	10-0	.617	10-0	.758
audiology	22.12	19.29	9-0	.872	15.71	10-0	.710	10-0	.814
auto	17.66	19.66	2-8	1.113	15.22	9-1	.862	9-1	.774
breast-w	5.28	4.23	9-0	.802	4.09	9-0	.775	7-2	.966
chess	8.55	8.33	6-2	.975	4.59	10-0	.537	10-0	.551
colic	14.92	15.19	0-6	1.018	18.83	0-10	1.262	0-10	1.240
credit-a	14.70	14.13	8-2	.962	15.64	1-9	1.064	0-10	1.107
credit-g	28.44	25.81	10-0	.908	29.14	2-8	1.025	0-10	1.129
diabetes	25.39	23.63	9-1	.931	28.18	0-10	1.110	0-10	1.192
glass	32.48	27.01	10-0	.832	23.55	10-0	.725	9-1	.872
heart-c	22.94	21.52	7-2	.938	21.39	8-0	.932	5-4	.994
heart-h	21.53	20.31	8-1	.943	21.05	5-4	.978	3-6	1.037
hepatitis	20.39	18.52	9-0	.908	17.68	10-0	.867	6-1	.955
hypo	.48	.45	7-2	.928	.36	9-1	.746	9-1	.804
iris	4.80	5.13	2-6	1.069	6.53	0-10	1.361	0-8	1.273
labor	19.12	14.39	10-0	.752	13.86	9-1	.725	5-3	.963
letter	11.99	7.51	10-0	.626	4.66	10-0	.389	10-0	.621
lymphography	21.69	20.41	8-2	.941	17.43	10-0	.804	10-0	.854
phoneme	19.44	18.73	10-0	.964	16.36	10-0	.842	10-0	.873
segment	3.21	2.74	9-1	.853	1.87	10-0	.583	10-0	.684
sick	1.34	1.22	7-1	.907	1.05	10-0	.781	9-1	.861
sonar	25.62	23.80	7-1	.929	19.62	10-0	.766	10-0	.824
soybean	7.73	7.58	6-3	.981	7.16	8-2	.926	8-1	.944
splice	5.91	5.58	9-1	.943	5.43	9-0	.919	6-4	.974
vehicle	27.09	25.54	10-0	.943	22.72	10-0	.839	10-0	.889
vote	5.06	4.37	9-0	.864	5.29	3-6	1.046	1-9	1.211
waveform	27.33	19.77	10-0	.723	18.53	10-0	.678	8-2	.938
average	*15.66*	*14.11*		*.905*	*13.36*		*.847*		*.930*

Table 1: Comparison of C4.5 and its bagged and boosted versions.

Bias-variance decomposition

- Theoretical tool for analyzing how much *specific* training set affects performance of a classifier
  - Total expected error of the prediction: bias + variance
  - The *bias* of a classifier is the expected error of the classifier due to the fact that the classifier is not perfect
  - The *variance* of a classifier is the expected error due to the particular training set used
- Often (trade off):
  - low bias => high variance
  - low variance => high bias

Bauer & Kohavi bias variance decomposition



Bagging vs. boosting

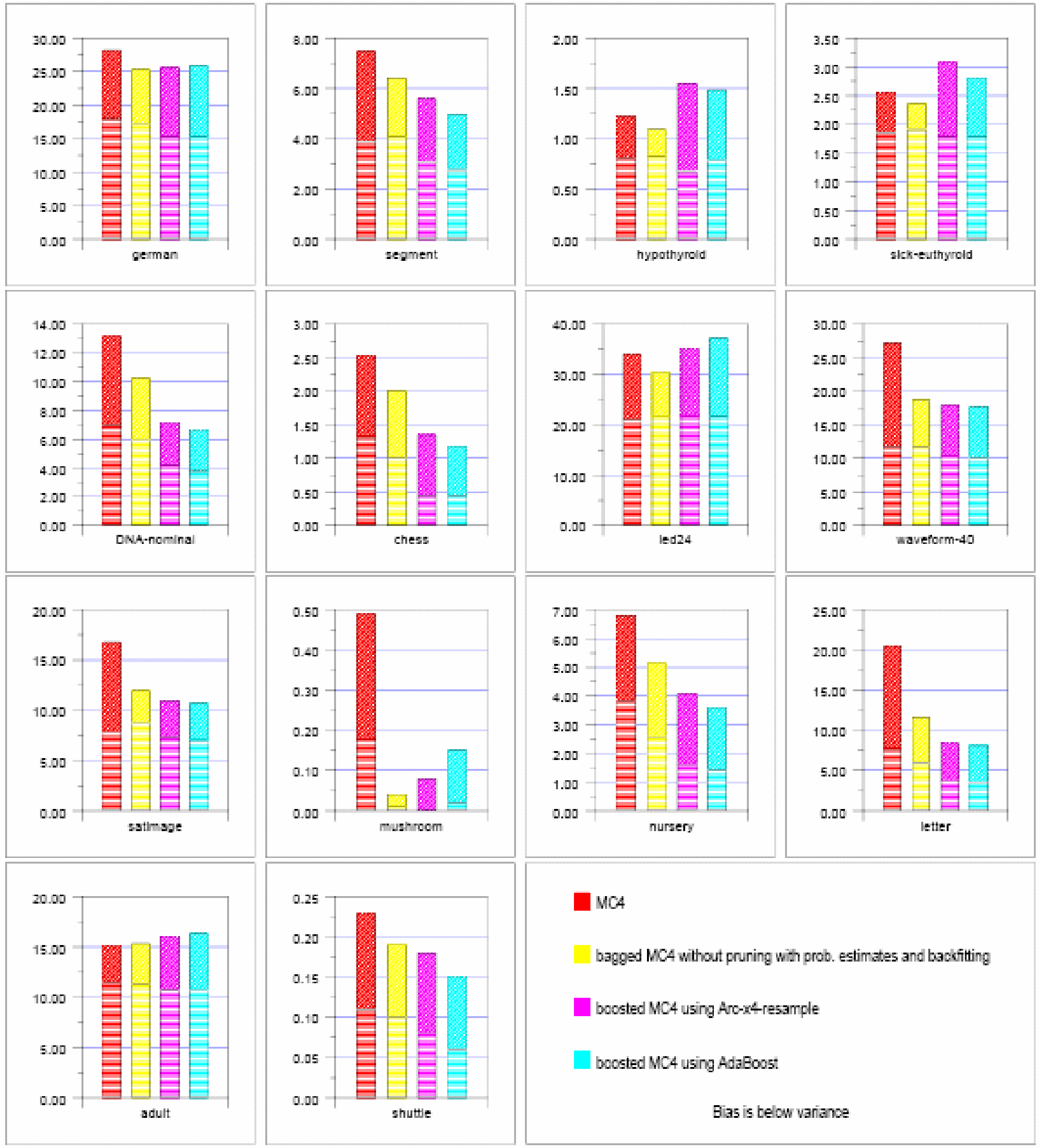


Figure 9. The bias and variance decomposition for MC4, backfit-p-Bagging, Arc-x4-resample, and AdaBoost. The boosting methods (Arc-x4 and AdaBoost) are able to reduce the bias over Bagging in some cases (e.g., DNA, chess, nursery, letter, shuttle). However, they also increase the variance (e.g., hypothyroid, sick-euthyroid, LED-24, mushroom, and adult).

Boosting vs. Bagging

- Bagging doesn’t work so well with stable models. Boosting might still help.
- Boosting might hurt performance on noisy datasets. Bagging doesn’t have this problem.
- On average, boosting helps more than bagging, but it is also more common for boosting to hurt performance.
- In practice bagging almost always helps.
- Bagging is easier to parallelize.

Feature-Selection Ensembles

• ***Key idea:*** Provide a different subset of the input features in each call of the learning algorithm.

• ***Example:*** Venus&Cherkauer (1996) trained an ensemble with 32 neural networks. The 32 networks were based on 8 different subsets of 119 available features and 4 different algorithms. The ensemble was significantly better than any of the neural networks!

•See also Random Subspace Methods by Ho.

Random forests [Breiman]

- At every level, choose a random subset of the attributes (not examples) and choose the best split among those attributes.
- Combined with selecting examples like basic bagging.
- Doesn’t overfit.

Data set	Adaboost	Selection	Forest-RI single input	One tree
Glass	22.0	20.6	21.2	36.9
Breast cancer	3.2	2.9	2.7	6.3
Diabetes	26.6	24.2	24.3	33.1
Sonar	15.6	15.9	18.0	31.7
Vowel	4.1	3.4	3.3	30.4
Ionosphere	6.4	7.1	7.5	12.7
Vehicle	23.2	25.8	26.4	33.1
German credit	23.5	24.4	26.2	33.3
Image	1.6	2.1	2.7	6.4
Ecoli	14.8	12.8	13.0	24.5
Votes	4.8	4.1	4.6	7.4
Liver	30.7	25.1	24.7	40.6
Letters	3.4	3.5	4.7	19.8
Sat-images	8.8	8.6	10.5	17.2
Zip-code	6.2	6.3	7.8	20.6
Waveform	17.8	17.2	17.3	34.0
Twonorm	4.9	3.9	3.9	24.7
Threenorm	18.8	17.5	17.5	38.4
Ringnorm	6.9	4.9	4.9	25.7

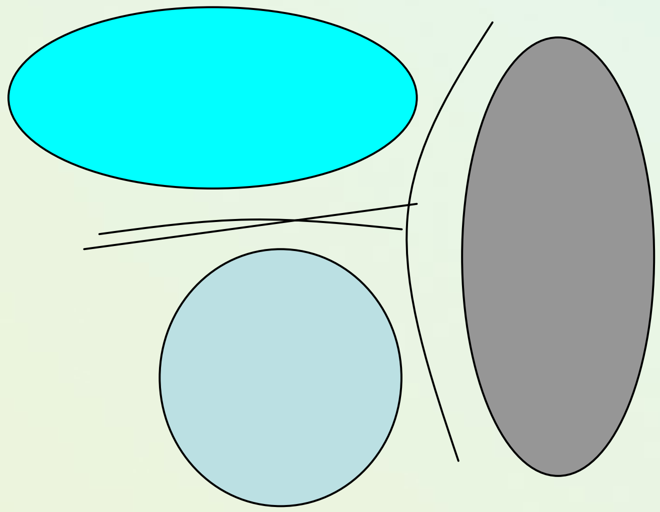
Breiman, Leo (2001). "Random Forests". Machine Learning 45 (1), 5-32

The n^2 classifier for multi-class problems

- Specialized approach for multi-class difficult problems.
  - Decompose a multi-class problem into a set of two-class sub-problems.
  - Combine them to obtain the final classification decision
- The idea based on pairwise coupling by Hastie T., Tibshirani R [NIPS 97] and J.Friedman 96.
- The n^2 version proposed by Jacek Jelonek and Jerzy Stefanowski [ECML 98].
- Other specialized approaches:
  - One-per-class,
  - Error-correcting output codes.

Solving multi-class problems

- The problem is to classify objects into a set of *n* decision classes (*n*>2)
- Some problems may be difficult to be learned (complex target concepts with non-linear decision boundaries).
- An example of three-class problem, where pairwise decision boundaries between each pairs of classes are simpler.

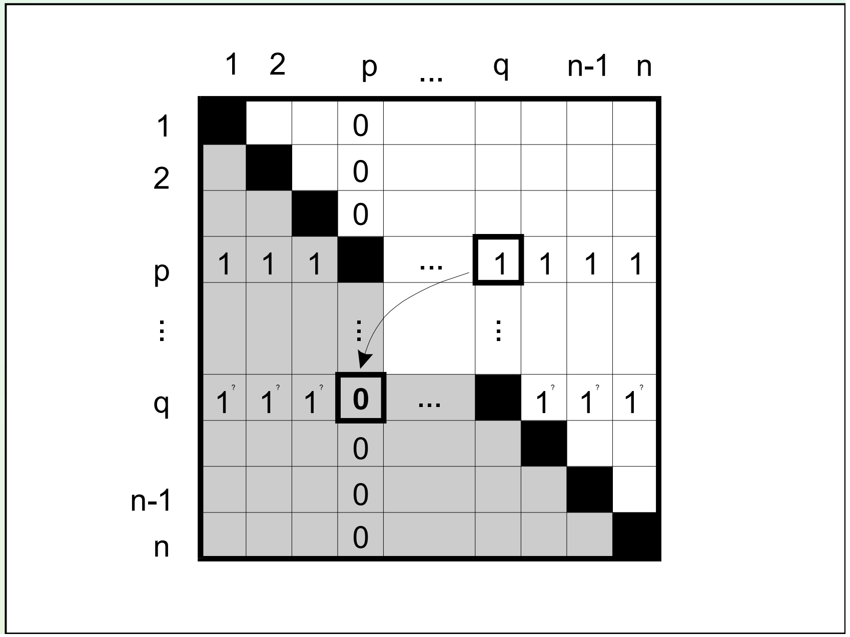


The n2-classifier

It is composed of $(n^2-n)/2$ *base binary classifiers* (all combinations of pairs of n classes).

- discrimination of each pair of the classes (i,j), where $i,j \in [1..n]$, $i \neq j$, by an independent binary classifier C_{ij}
- The specificity of training binary classifier C_{ij} - only examples from two classes i,j.
- classifier C_{ij} yields binary classification (1 or 0), classifiers C_{ij} and C_{ji} are equivalent

$$C_{ji}(\mathbf{x}) = 1 - C_{ij}(\mathbf{x})$$



Final classification decision of the n^2-classifier

- For an unseen example **x,** a final classification of the n^2-classifier is a proper aggregation of predictions of all base classifiers $C_{ij}(\mathbf{x})$
- Simplest aggregation - find a class that wins the most pairwise comparison
- The aggregation could be extended by estimating credibility of each base classifier (during learning phase) P_{ij}
- Final classification decision - a weighted majority rule:
  - choose such a decision class „*i*" that maximizes:

$$\sum_{j=1, i \neq j}^{n} P_{ij} \cdot C_{ij}(x)$$

Conditions of experiments

- We examine an influence of the learning algorithm on the classification performance of n^2-classifier:
  - Decision trees
  - Decision rules (MODLEM)
  - Artificial neural network (feed forward multi-layer network trained by Back-Propagation)
  - Instance based learning (k-nn, k=1, Euclidean distance)
- Computations on MLR-UCI benchmark data sets and our medical ones.
- The classification accuracy estimated by stratified 10-fold cross validation

Performance of n^2 classifier based on decision trees

Data set	Classification accuracy *DT* (%)	Classification accuracy n^2 (%)	Improvement n^2 vs. *DT* (%)
Automobile	85.5 ± 1.9	87.0 ± 1.9	**1.5***
Cooc	54.0 ± 2.0	59.0 ± 1.7	**5.0**
Ecoli	79.7 ± 0.8	81.0 ± 1.7	**1.3**
Glass	70.7 ± 2.1	74.0 ± 1.1	**3.3**
Hist	71.3 ± 2.3	73.0 ± 1.8	**1.7**
Meta-data	47.2 ± 1.4	49.8 ± 1.4	**2.6**
Primary Tumor	40.2 ± 1.5	45.1 ± 1.2	**4.9**
Soybean-large	91.9 ± 0.7	92.4 ± 0.5	**0.5***
Vowel	81.1 ± 1.1	83.7 ± 0.5	**2.6**
Yeast	49.1 ± 2.1	52.8 ± 1.8	**3.7**

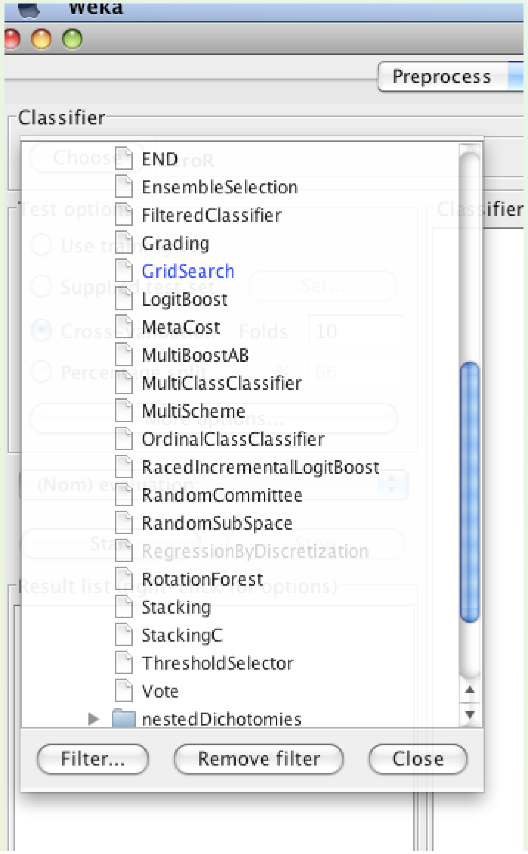
Discussion of experiments with various algorithms

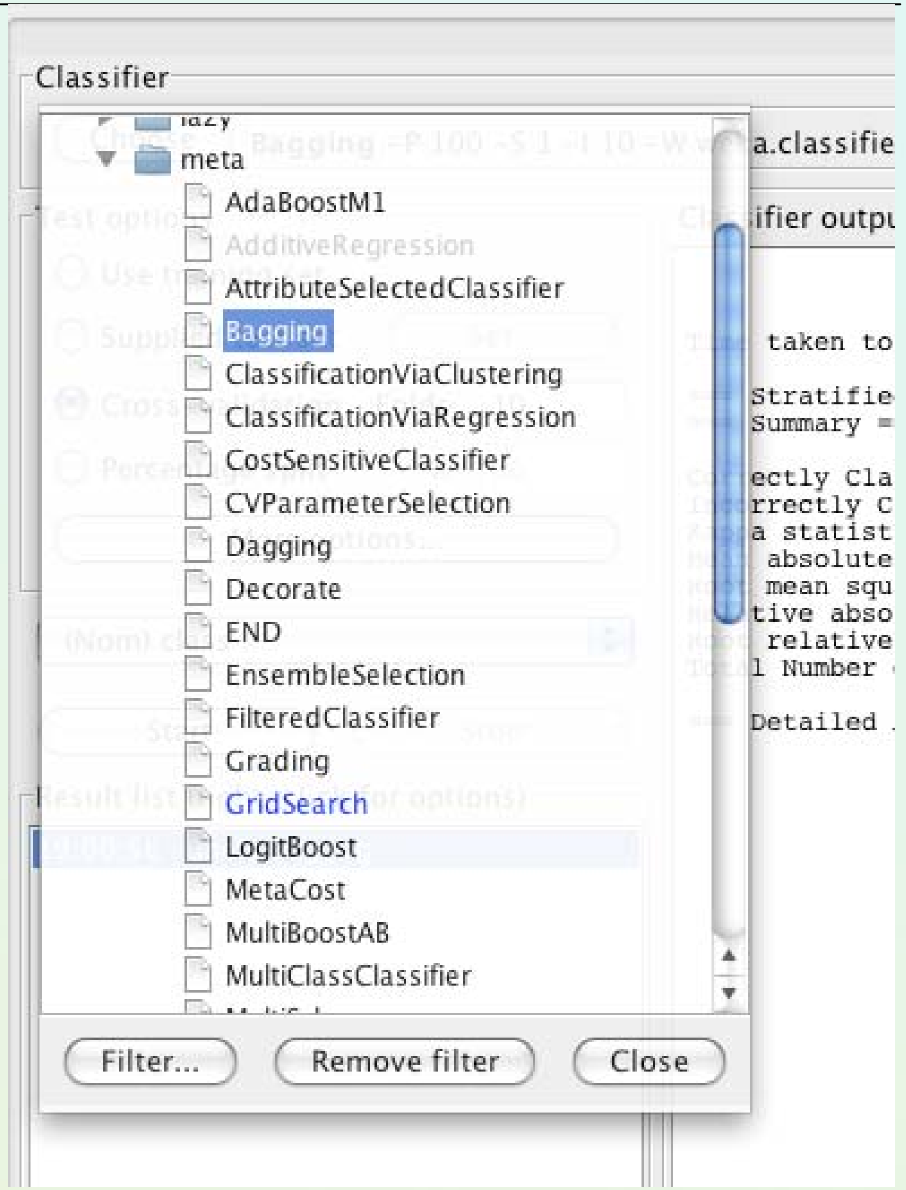
- Decision trees $\rightarrow$ significant better classification for 8 of all data sets; other differences non-significant
- Comparable results for decision rules
  Artificial neural networks $\rightarrow$ generally better classification for 9 of all data sets; some of highest improvements but difficulties in constructing networks
- However, k-nn does not result in improving classification performance of the n^2-classier with respect to single multi-class instance-based learner!
  - We proposed an approach to select attribute subsets discriminating each pair of classes $\rightarrow$ it improved a k-nn constructed classifier.

Ensembles in WEKA – see Meta in Classifiers

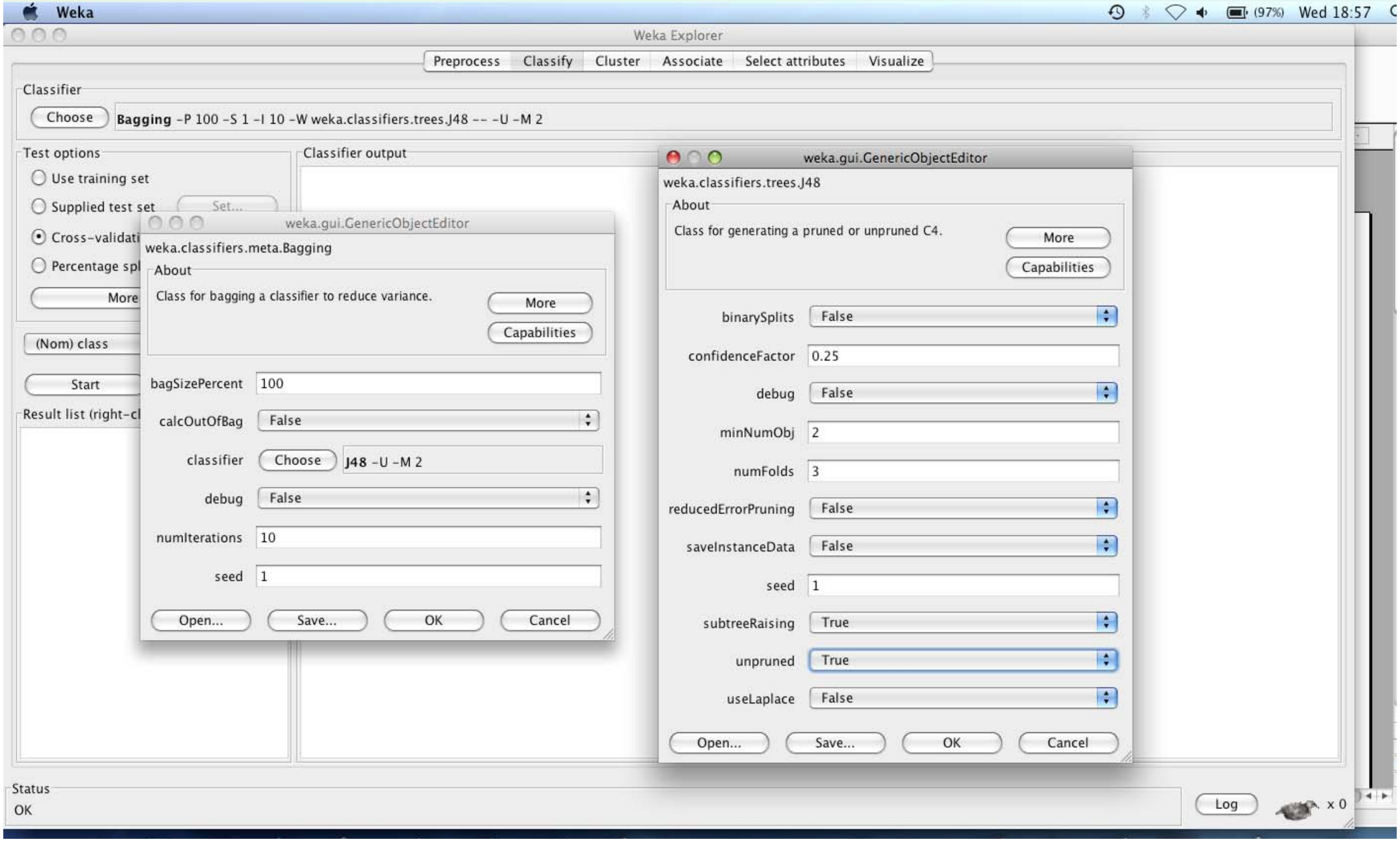
Classifiers

- Meta
- There are many techniques …



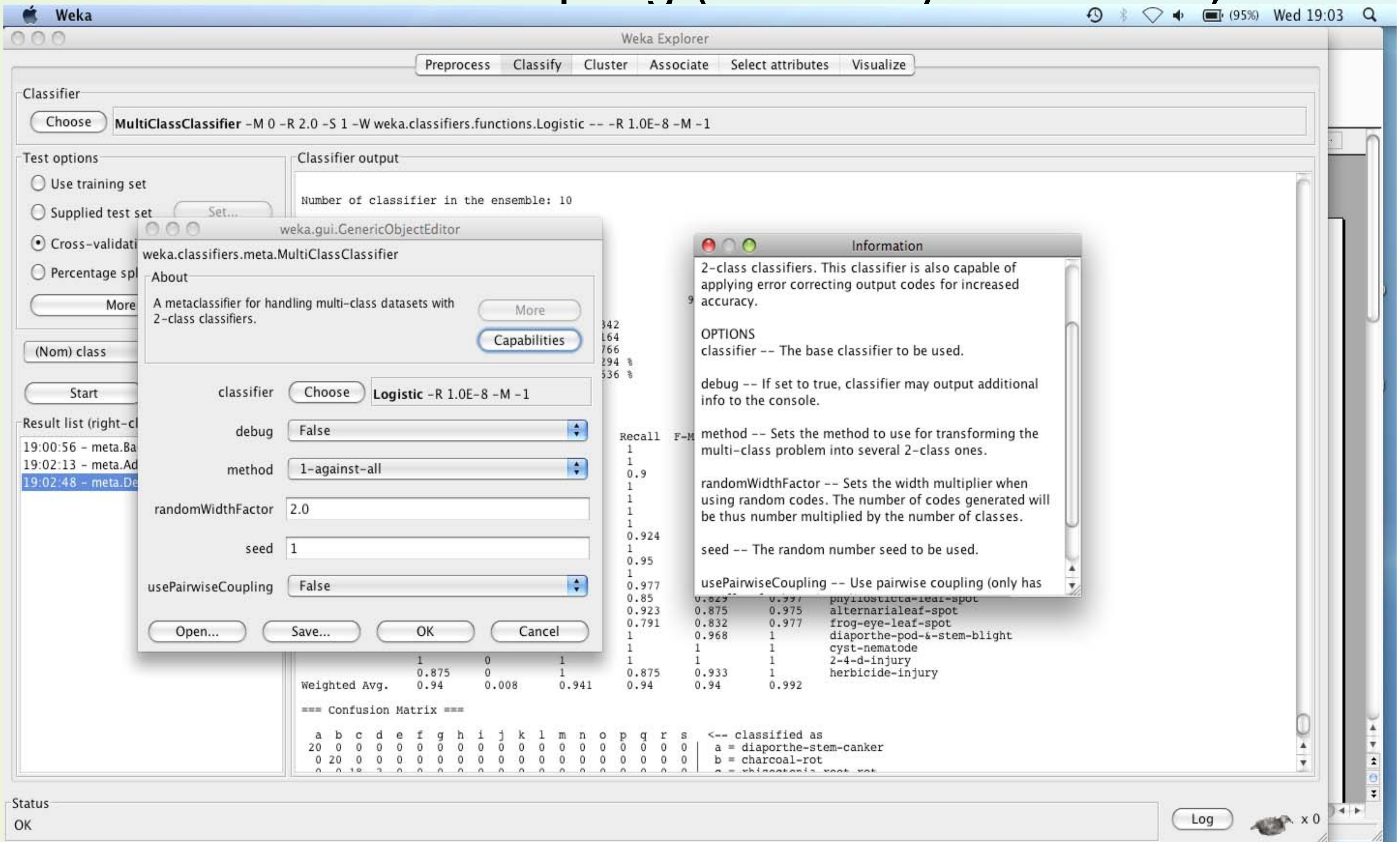


Experience with WEKA - bagging

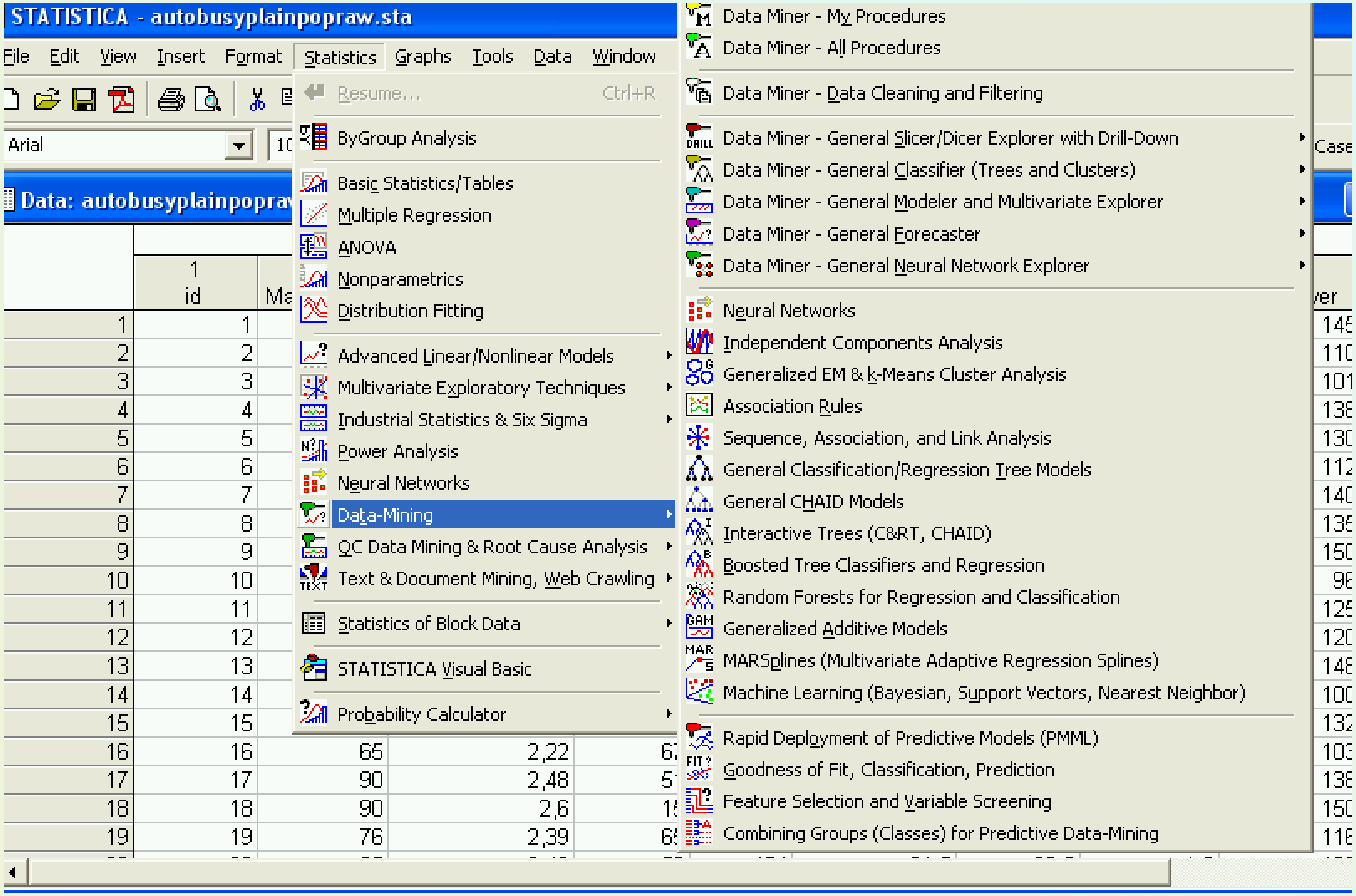


Other multi-classifiers

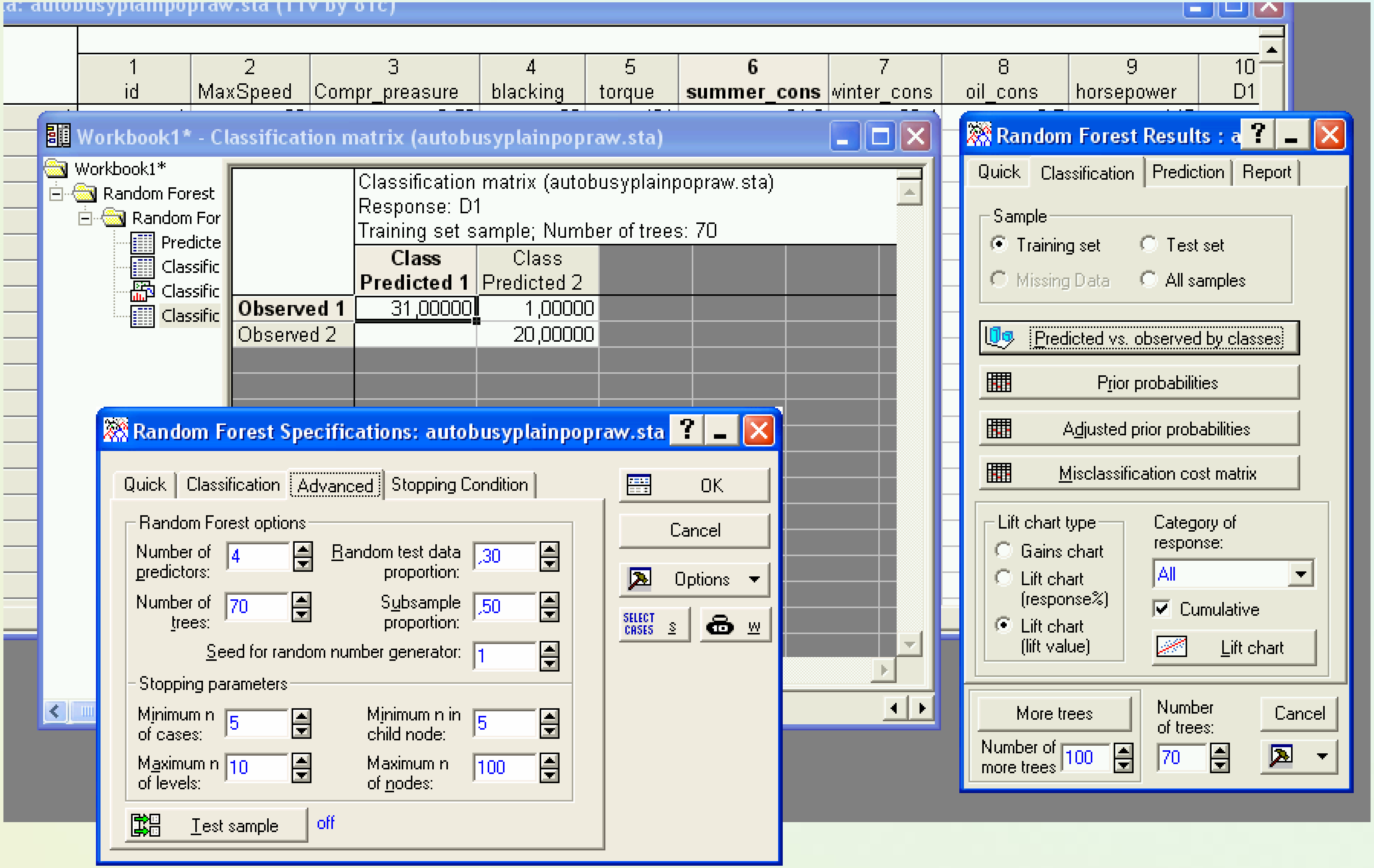
- Pair-wise coupling (n2 binary classifiers)



Multiple classifiers in Statistica



Random Forest (CART)



Some Practical Advices [Smirnov]

If the classifier is unstable (i.e, decision trees) then apply bagging!

If the classifier is stable and simple (e.g. Naïve Bayes) then apply boosting!

If the classifier is stable and very complex (e.g. Neural Network) then apply randomization injection!

If you have many classes and a binary classifier then try error-correcting codes! If it does not work then use a complex binary classifier!

Any questions, remarks?

?

Other Sources

- David Mease. Statistical Aspects of Data Mining. Lecture. http://video.google.com/videoplay?docid=-4669216290304603251&q=stats+202+engEDU&total=13&start=0&num=10&so=0&type=search&plindex=8
- Dietterich, T. G. Ensemble Learning. In The Handbook of Brain Theory and Neural Networks, Second edition, (M.A. Arbib, Ed.), Cambridge, MA: The MIT Press, 2002. http://www.cs.orst.edu/~tgd/publications/hbtnn-ensemble-learning.ps.gz
- Elder, John and Seni Giovanni. From Trees to Forests and Rule Sets - A Unified Overview of Ensemble Methods. KDD 2007 http://Tutorial. videolectures.net/kdd07_elder_ftfr/
- Christopher M. Bishop. Neural Networks for Pattern Recognition. Oxford University Press.