The *uci621* benchmark experiment

R supplement of

"Exploratory and Inferential Analysis of Benchmark Experiments"

Manuel J. A. Eugster

`benchmark` version 0.01

The *uci621* benchmark experiment is the exemplar study we used in our article "Exploratory and Inferential Analysis of Benchmark Experiments" [Eugster et al., 2008]. The primary goal of this example is to illustrate general interesting aspects of benchmark experiments; not necessarily using up-to-date classifiers and newly data sets.

1 The Setup

The candidate algorithms are:

linear discriminant analysis: available through the function `lda` in package `MASS`.

naive bayes classifier: available through the function `naiveBayes` in package `e1071`; sometimes referenced with `nb`.

k-nearest neighbour classifier: available through the function `knn` in package `class`. The hyperparameter k (the number of neighours) is determined with cross-validation between 1 and $\sqrt{n}$, n the number of observations.

classification trees: available through the function `rpart` in package `rpart`. The fulled tree is pruned according to the 1-SE rule [e.g., Venables and Ripley, 2002, Hastie et al., 2001].

support vector machines: available through the function `svm` in package `e1071`. We use the *C-classification* machine, which has two hyperparameters γ (the cost of constraints violation) and c (the kernel parameter). Following Meyer et al. [2003] the best choices are determined with a grid search over the two-dimensional parameter space (γ, c), γ ranges from 2^{-5} to 2^{12} and c from 2^{-10} to 2^{5}.

neural networks: available through the function `nnet` in package `nnet`. The hyperparameter is the number of hidden units. The best value is searched with cross-validation between 1 and $\log(n)$, n the number of observations [following Meyer et al., 2003].

The set of data set consists 21 binary classification problems originated from the UCI Machine Learning repository [Asuncion and Newman, 2007] – cleaned before used:

Problem	#Attributes		#Samples		Class
	nominal	continuous	complete	incomplete	distribution (%)

promotergene	A	57		106		50.00/50.00
hepatitis	B	13	6	80	75	20.65/79.35
Sonar	C		60	208		53.37/46.63
Heart1	D	8	5	296	7	54.46/45.54
liver	E		6	345		42.03/57.97
Ionosphere	F	1	32	351		35.90/64.10
HouseVotes84	G	16		232	203	61.38/38.62
musk	H		166	476		56.51/43.49
monks3	I	6		554		48.01/51.99
Cards	J	9	6	653	37	44.49/55.51
BreastCancer	K	9		683	16	65.52/34.48
PimaIndiansDiabetes	L		8	768		65.10/34.90
tictactoe	M	9		958		34.66/65.34
credit	N		24	1000		70.00/30.00
Circle (*)	O		2	1200		50.67/49.33
ringnorm (*)	P		20	1200		50.00/50.00
Spirals (*)	Q		2	1200		50.00/50.00
threenorm (*)	R		20	1200		50.00/50.00
twonorm (*)	S		20	1200		50.00/50.00
titanic	T	3		2201		67.70/32.30
chess	U	36		3196		47.78/52.22

Resampling strategy is bootstrapping; we draw 250 samples per data set as learning samples and use the out-of-bootstrap samples as test samples. Performance measures are the misclassification and the computation time.

2 The Execution

The execution was done by some simple hand-knitted `for`-loops, hyperparameter tuning was done using the `tune` functions from the `e1071` package. The execution results in a four dimensional array with first dimension the sampling, second dimension the algorithms, third dimension the performance measures and fourth dimension the data sets.

```
> load('uci621raw.RData')
> str(uci621raw)
```

```
 num [1:250, 1:6, 1:2, 1:21]     NA 0.0618 0.0445 0.0413 0.0685 ...
 - attr(*, "dimnames")=List of 4
  ..$ samp: NULL
  ..$ alg : chr [1:6] "lda" "naiveBayes" "knn" "rpart" ...
  ..$ perf: chr [1:2] "Misclassification" "Time"
  ..$ ds  : chr [1:21] "BreastCancer" "Cards" "chess" "Circle" ...
```

For analyses of the raw benchmark results we use the package `benchmark`.

```
> library(benchmark)
```

```
> uci621 <- as.bench(uci621raw)
```

```
Benchmark experiment
```

```
     samples   algorithms performances    data sets
         250            6            2           21
```

A variety of methods is especially for the most popular benchmark scenairo with one performance measure and one data set. For this purpose we use the *misclassification* × *monks3* subset from the *uci621* experiment as exemplar benchmark experiment.

```
> monks3 <- uci621[,,'Misclassification', 'monks3']
```

```
Benchmark experiment

     samples   algorithms performances    data sets
         250            6            1            1
```

For visualizations we define colors for each algorithm; "normal" and "light" versions:

```
> source('sixcolors.R')
> par(mar=c(0,0,0,0))
> pal(c(sixcols, sixcols.light))
```

The assignment is simply according to the chronology of the appearance:

```
> names(sixcols) <- colnames(uci621raw)
> names(sixcols.light) <- colnames(uci621raw)
```

References

A. Asuncion and D.J. Newman. UCI machine learning repository, 2007. URL `http://www.ics.uci.edu/ mlearn/MLRepository.html`.

Manuel J. A. Eugster, Torsten Hothorn, and Friedrich Leisch. Exploratory and inferential analysis of benchmark experiments. Technical Report 30, Institut für Statistik, Ludwig-Maximilians-Universität München, Germany, 2008. URL `http://epub.ub.uni-muenchen.de/4134/`.

Trevor Hastie, Robert Tibshirani, and Jerome Friedman. *The Elements of Statistical Learning: Data Mining, Inference, and Prediction.* Springer-Verlag, 2001.

David Meyer, Friedrich Leisch, and Kurt Hornik. The support vector machine under test. *Neurocomputing*, 55:169–186, September 2003.

William Venables and Brian Ripley. *Modern Applied Statistics with S.* Springer-Verlag, fourth edition, 2002.