Tutorial: OpenML with R and mlr

Bernd Bischl, Joaquin Vanschoren et al.

Reisensburg 2015

Section 1

The mlr Package

Motivation

The good news

- CRAN serves hundreds of packages for machine learning
- Many packages are compliant to the unwritten interface definition:

```
> model = fit(target ~ ., data = train.data, ...)
> predictions = predict(model, newdata = test.data, ...)
```

The bad news

- Some packages do not support the formula interface or their API is "just different"
- No meta-information available or buried in docs (sometimes not documented at all)
- Larger experiments lead to lengthy, tedious and error-prone code

Motivation: mlr

https://github.com/mlr-org/mlr

- Unified interface for the basic building blocks: tasks, learners, resampling, hyperparameters, ...
- Reflections: nearly all objects are queryable (i.e. you can ask them for their properties and program on them)
- Possibility to fit, predict, evaluate and resample models
- Different visualizations for e.g. ROC curves and predictions
- Benchmarking of learners for muliple data sets
- Parallelization is built-in
- ...

Task Abstractions

- Regression, classification, survival and cost-sensitive tasks
- Internally: data frame with annotations: target column(s), weights, misclassification costs, ...)

```
> data("Sonar", package = "mlbench")
> task = makeClassifTask(data = Sonar, target = "Class")
> print(task)

## Supervised task: Sonar
## Type: classif
## Target: Class
## Observations: 208
## Features:
## numerics  factors  ordered
##       60        0        0
## Missings: FALSE
## Has weights: FALSE
## Has blocking: FALSE
## Classes: 2
##   M   R
## 111  97
## Positive class: M
```

Learner Abstractions

- 56 classification, 46 regression, 10 survival
- Internally: functions to train and predict

```
> lrn = makeLearner("classif.rpart")
> print(lrn)


## Learner classif.rpart from package rpart
## Type: classif
## Name: Decision Tree; Short name: rpart
## Class: classif.rpart
## Properties: twoclass,multiclass,missings,numerics,factors,ordered,prob,weights
## Predict-Type: response
## Hyperparameters: xval=0


> lrn = makeLearner("classif.rpart", predict.type = "prob")
> print(lrn)


## Learner classif.rpart from package rpart
## Type: classif
## Name: Decision Tree; Short name: rpart
## Class: classif.rpart
## Properties: twoclass,multiclass,missings,numerics,factors,ordered,prob,weights
## Predict-Type: prob
## Hyperparameters: xval=0
```

Resampling

- Resampling techniques: CV, Bootstrap, Subsampling, ...

```
> cv3f = makeResampleDesc("CV", iters = 3, stratify = TRUE)
```

- 10-fold CV of rpart on iris

```
> lrn = makeLearner("classif.rpart", predict.type = "prob")
> cv10f = makeResampleDesc("CV", iters = 10)
> measures = list(acc, auc)
>
> resample(lrn, task, cv10f, measures)$aggr

## acc.test.mean auc.test.mean
##     0.7209524     0.7571493
```

Benchmarking

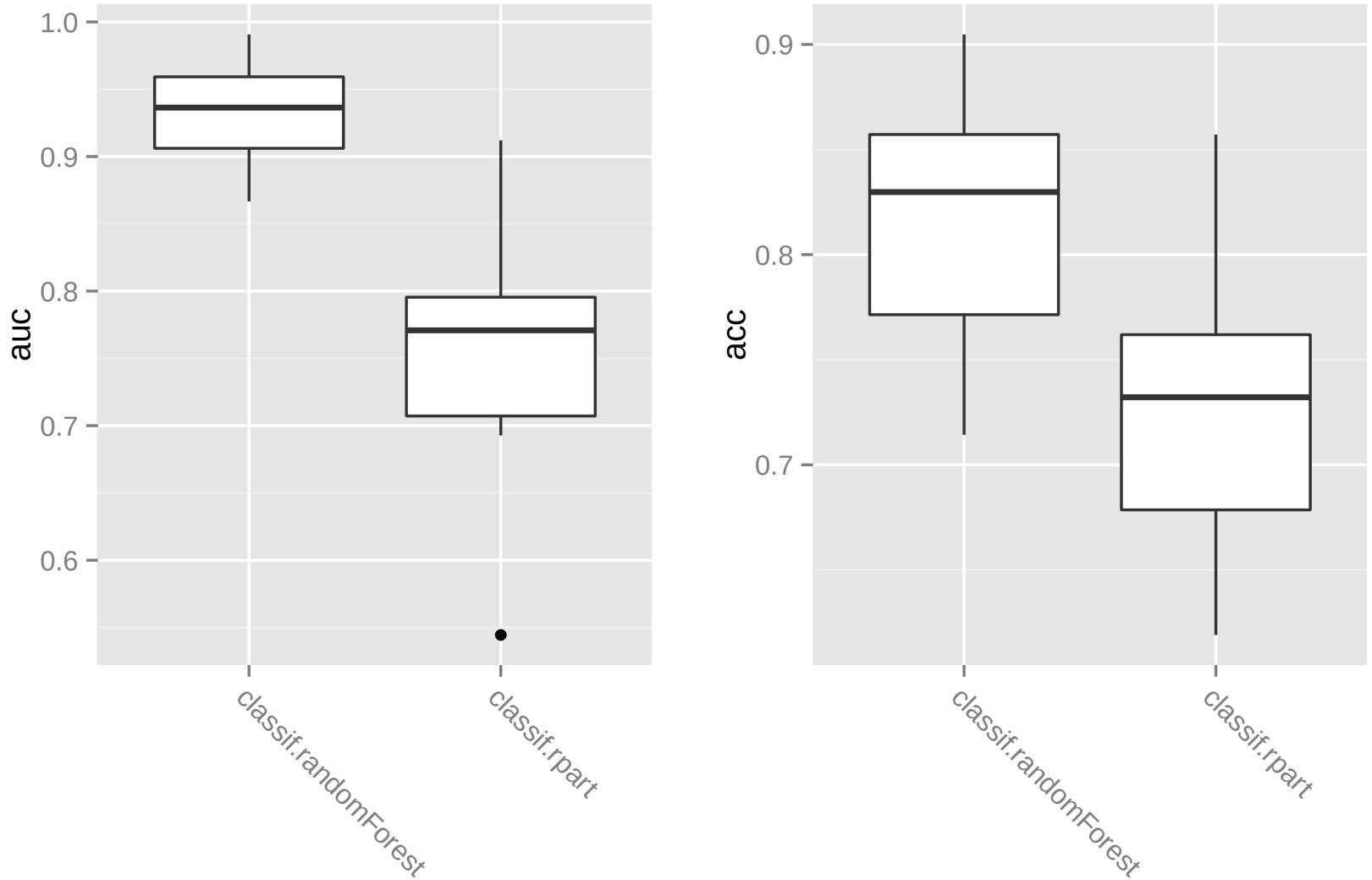
- Compare multiple learners on multiple tasks
- Fair comparisons: same training and test sets for each learner

```
> sonar.task = makeClassifTask(data = Sonar, target = "Class")
> cv10f = makeResampleDesc("CV", iters = 10)
> measures = list(acc, auc)
> learners = list(
+   makeLearner("classif.randomForest", predict.type = "prob"),
+   makeLearner("classif.rpart", predict.type = "prob")
+ )
>
> (res = benchmark(learners, sonar.task, cv10f, measures))

##   task.id           learner.id acc.test.mean auc.test.mean
## 1   Sonar classif.randomForest     0.8173810     0.9311755
## 2   Sonar        classif.rpart     0.7352381     0.7563062
```

Benchmarking

```
> library(gridExtra)
> grid.arrange(plotBenchmarkResult(res, auc, pretty.names = F),
+              plotBenchmarkResult(res, acc, pretty.names = F), ncol=2)
```



Section 2

OpenML R-Package

OpenML R-Package

`https://github.com/openml/r`

Current API in R

- Explore data and tasks
- Download data and tasks
- Register learners
- Upload runs
- Explore your own and other people's results

Already nicely connected to `mlr`!

OpenML: Explore and Select Data I

```
> library(OpenML)
> # You can get your own account at openml.org
> authenticateUser(username = "openml.rteam@gmail.com",
+                  password = "testpassword")

## Authenticating user at server:  openml.rteam@gmail.com
## Retrieved session hash.  Valid until:  2015-07-20 22:19:04

> listOMLDataSets()[1:3, 1:5]

## Downloading 'http://www.openml.org/api/?f=openml.data' to '<mem>'

##   did status        name NumberOfClasses NumberOfFeatures
## 1   1 active      anneal               6               39
## 2   2 active anneal.ORIG               6               39
## 3   3 active    kr-vs-kp               2               37
```

OpenML: Explore and Select Data II

```
> listOMLTasks()[1:3, 1:7]

## Downloading
'http://www.openml.org/api/?f=openml.tasks&task_type_id=1' to '<mem>'

##   task_id                task_type did status        name
## 1       1 Supervised Classification   1 active      anneal
## 2       2 Supervised Classification   2 active anneal.ORIG
## 3       3 Supervised Classification   3 active    kr-vs-kp
##      estimation_procedure evaluation_measures
## 1 10-fold Crossvalidation predictive_accuracy
## 2 10-fold Crossvalidation predictive_accuracy
## 3 10-fold Crossvalidation predictive_accuracy
```

OpenML: Download a Data Set

```
> # uses built in caching from disk
> getOMLDataSet(1)


##
## Data Set "anneal" :: (Version = 2, OpenML ID = 1)
##   Default Target Attribute: class
```

OpenML: Download a Task I

```
> # uses built in caching from disk
> oml.task = getOMLTask(task.id = 1)
> print(oml.task)

##
## OpenML Task 1 :: (Data ID = 1)
##   Task Type             : Supervised Classification
##   Data Set              : anneal :: (Version = 2, OpenML ID = 1)
##   Target Feature(s)     : class
##   Estimation Procedure  : Stratified crossvalidation (1 x 10 folds)
```

OpenML: Download a Task II

```
> oml.task$input$data.set

##
## Data Set "anneal" :: (Version = 2, OpenML ID = 1)
##   Default Target Attribute: class

> oml.task$input$estimation.procedure

##
## Estimation Method :: crossvalidation
##  Parameters:
##  number_repeats = 1
##  number_folds = 10
##  stratified_sampling = true

> oml.task$input$evaluation.measures

## [1] "predictive_accuracy"
```

OpenML: Run several Learners on one Task

```
> lrn1 = makeLearner("classif.rpart")
> lrn2 = makeLearner("classif.randomForest")
> res = runMultipleLearnersOnTask(oml.task, list(lrn1, lrn2))
> res$benchmark

##   task.id           learner.id acc.test.mean
## 1    data        classif.rpart     0.9765918
## 2    data classif.randomForest     0.9922097
```

OpenML R-Package

Let's have a look at the R-Package

Thanks!

References I


Van Rijn, J. N., Bischl, B., Torgo, L., Gao, B., Umaashankar, V., Fischer, S., Winter, P., Wiswedel, B., Berthold, M. R., and Vanschoren, J. (2013).
Openml: A collaborative science platform.
In *Machine learning and knowledge discovery in databases*, pages 645–649. Springer.


Vanschoren, J., van Rijn, J. N., Bischl, B., and Torgo, L. (2014).
Openml: networked science in machine learning.
ACM SIGKDD Explorations Newsletter, 15(2):49–60.