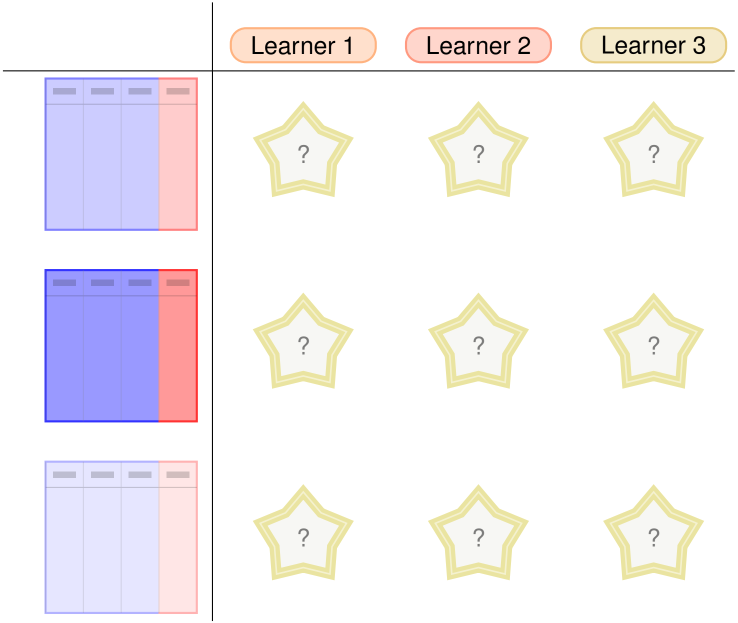
Applied Machine Learning

Machine Learning in R:
MLR3 Benchmarking & Summary

Learning goals

- Setting up benchmarking experiments
- Working with ‘BenchmarkResult‘ object
- Parallelization
- MLR3 resources and help

PERFORMANCE COMPARISON



PERFORMANCE COMPARISON

- Multiple Learners, multiple Tasks:

```r
library("mlr3learners")
learners = list(lrn("classif.rpart"), lrn("classif.kknn"))
tasks = list(tsk("iris"), tsk("sonar"), tsk("wine"))
```

PERFORMANCE COMPARISON

- Multiple Learners, multiple Tasks:

```r
library("mlr3learners")
learners = list(lrn("classif.rpart"), lrn("classif.kknn"))
tasks = list(tsk("iris"), tsk("sonar"), tsk("wine"))
```

- Set up the *design* and execute benchmark:

```r
design = benchmark_grid(tasks, learners, cv5)
bmr = benchmark(design)
```

PERFORMANCE COMPARISON

- Multiple Learners, multiple Tasks:

```
library("mlr3learners")
learners = list(lrn("classif.rpart"), lrn("classif.kknn"))
tasks = list(tsk("iris"), tsk("sonar"), tsk("wine"))
```

- Set up the *design* and execute benchmark:

```
design = benchmark_grid(tasks, learners, cv5)
bmr = benchmark(design)
```

- We get a `BenchmarkResult` object which shows that `kknn` outperforms `rpart`:

```
bmr_ag = bmr$aggregate()
bmr_ag[, c("task_id", "learner_id", "classif.ce")]

#>    task_id    learner_id classif.ce
#>     <char>        <char>      <num>
#> 1:    iris classif.rpart      0.053
#> 2:    iris  classif.kknn      0.040
#> 3:   sonar classif.rpart      0.274
#> 4:   sonar  classif.kknn      0.130
#> 5:    wine classif.rpart      0.157
#> 6:    wine  classif.kknn      0.039
```

BENCHMARK RESULT

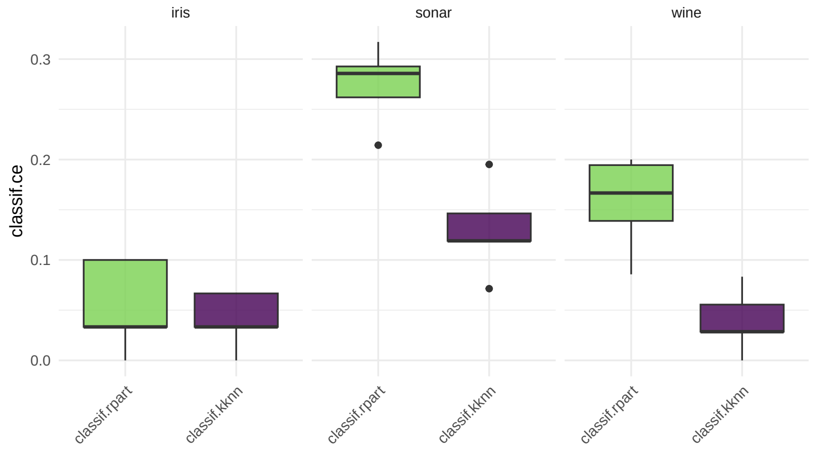
What exactly is a `BenchmarkResult` object?
Just like `Prediction` and `ResamplingResult`!

- Table representation using `as.data.table()`
- Active bindings and functions that make information easily accessible

BENCHMARK RESULT

The `mlr3viz` package contains `autoplot()` functions for many mlr3 objects

```
library(mlr3viz)
autoplot(bmr)
```



CONTROL OF EXECUTION

Parallelization

```r
future::plan("multicore")
```

- runs each resampling iteration as a job
- also allows nested resampling (although not needed here)

Encapsulation

```r
learner$encapsulate = c(train = "callr", predict = "callr")
```

- Spawns a separate R process to train the learner
- Learner may segfault without tearing down the session
- Logs are captured
- Possibilty to have a fallback to create predictions

Help and Summary

HOW TO GET HELP

- Where to start?
  - Check these slides
  - **Check the mlr3book https://mlr3book.mlr-org.com**

HOW TO GET HELP

- Where to start?
  - Check these slides
  - **Check the mlr3book https://mlr3book.mlr-org.com**
- Get help for R6 objects?
  1. Find out what kind of R6 object you have:

     ```
     class(bmr)
     #> [1] "BenchmarkResult" "R6"
     ```

  2. Go to the corresponding help page:

     ```
     ?BenchmarkResult
     ```

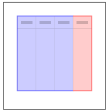
  New: open the corresponding man page with

```
learner$help()
```

OVERVIEW

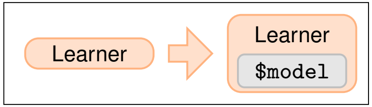
Ingredients:

Data



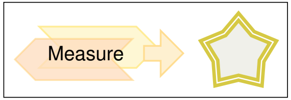
```
TaskClassif,
TaskRegr,
tsk()
```

Learning Algorithms



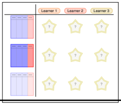
```
lrn() ⇒ Learner,
↪Learner$train(),
↪Learner$predict() ⇒ Prediction
```

Performance Evaluation



```
rsmp() ⇒ Resampling,
msr() ⇒ Measure,
resample() ⇒ ResamplingResult,
↪ ResamplingResult$score(),
↪ ResamplingResult$aggregate()
```

Performance Comparison



```
benchmark_grid(),
benchmark() ⇒ BenchmarkResult
```

- Machine Learning in R Using mlr3 - 18/18