

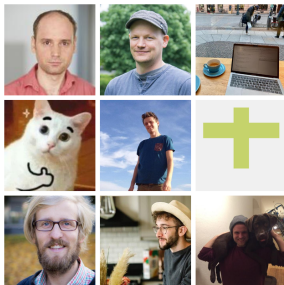
# Modern Machine Learning in R

---



<https://mlr-org.com/>

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



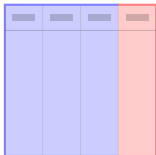
---

**Bernd Bischl, Michel Lang, Martin Binder, Florian Pfisterer, Jakob Richter,  
Patrick Schratz, Lennart Schneider, Raphael Sonabend, Marc Becker**

October 22, 2021

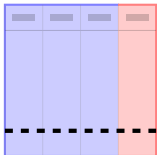
# Resampling

# RESAMPLING



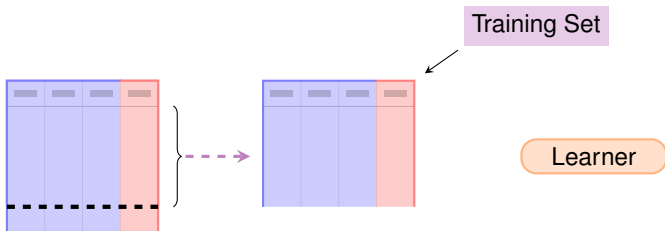
Learner

# RESAMPLING

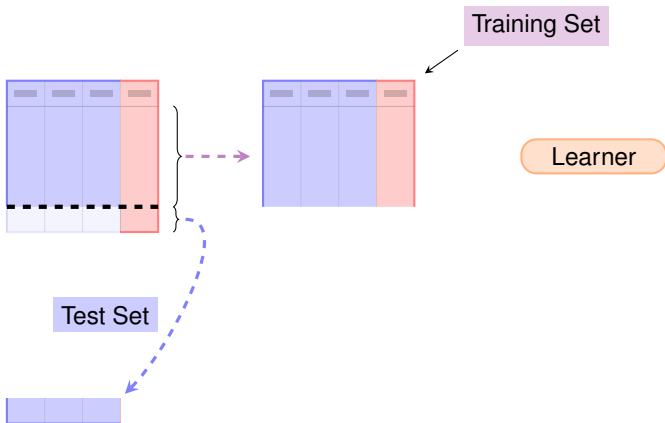


Learner

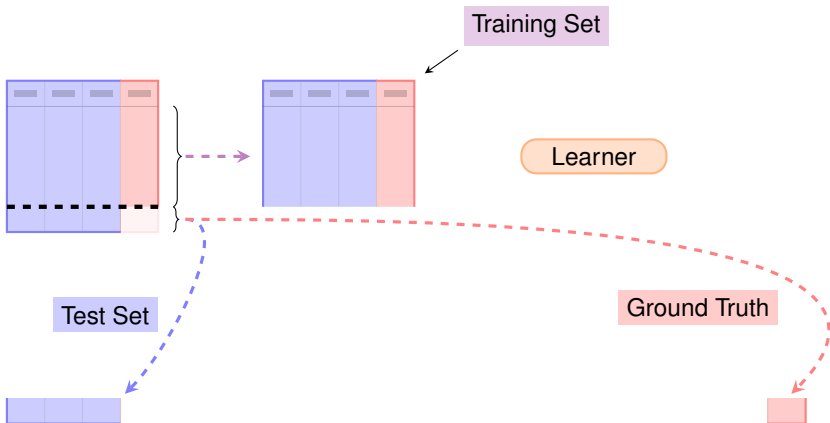
# RESAMPLING



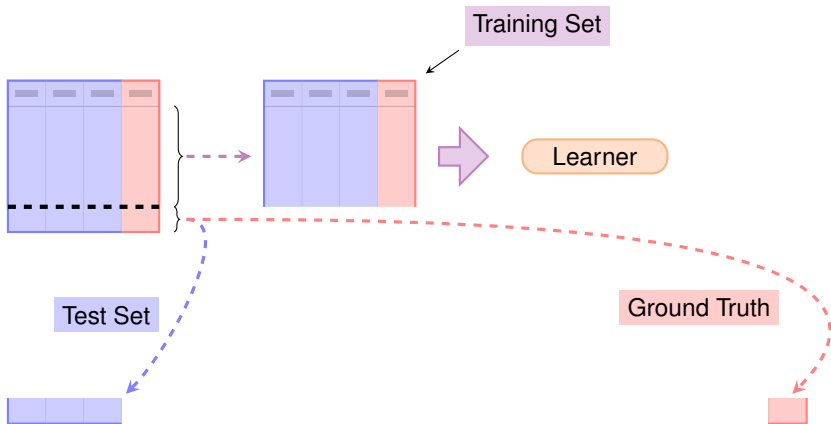
# RESAMPLING



# RESAMPLING

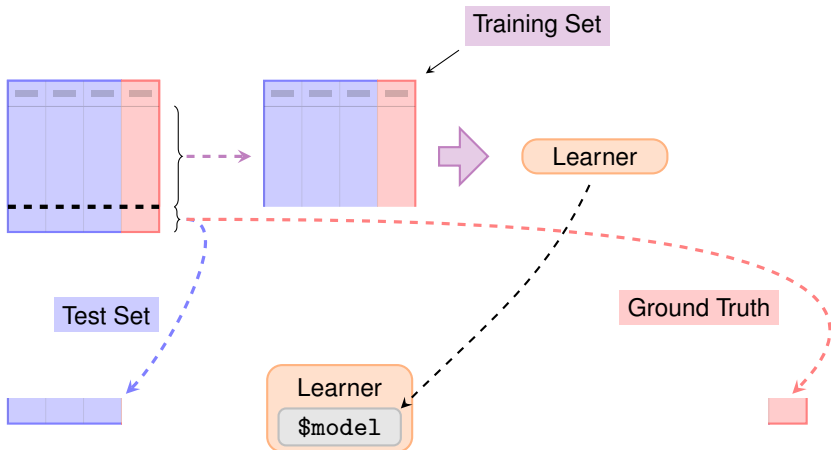


# RESAMPLING

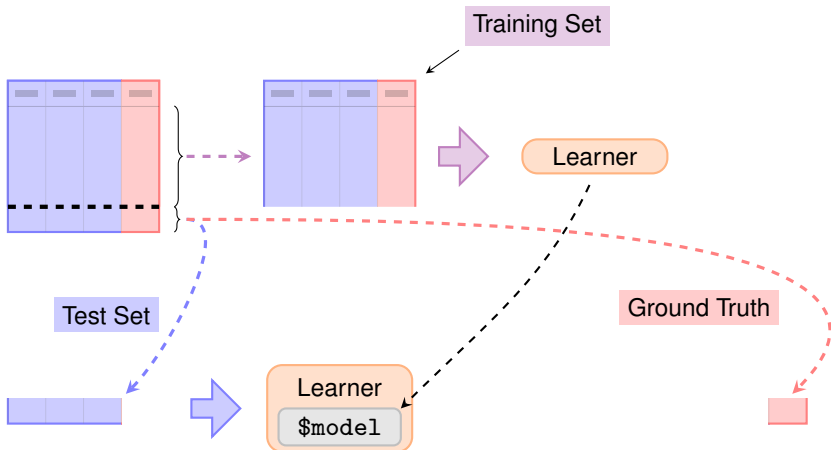




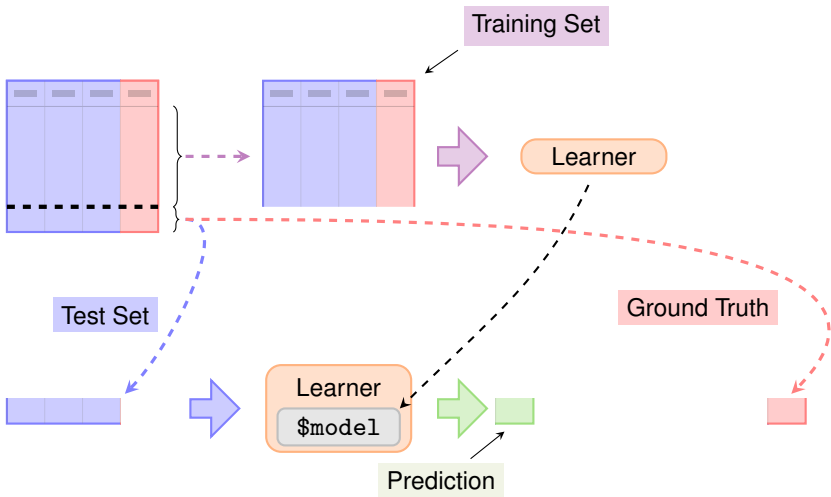
# RESAMPLING



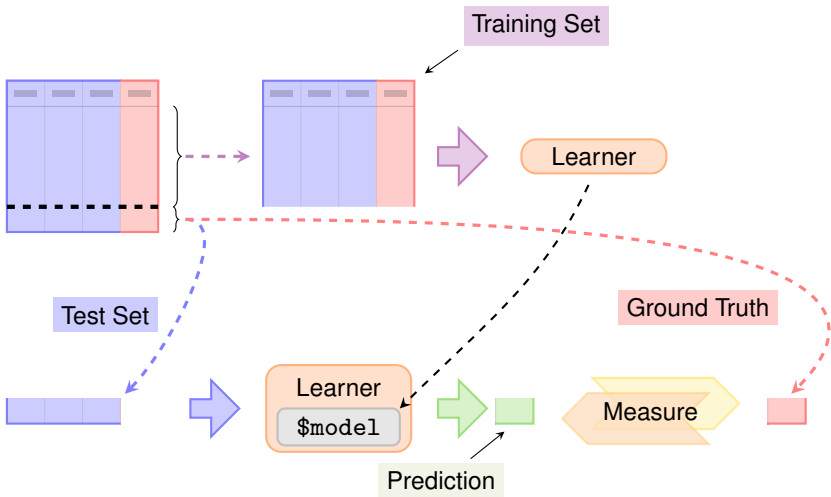
# RESAMPLING



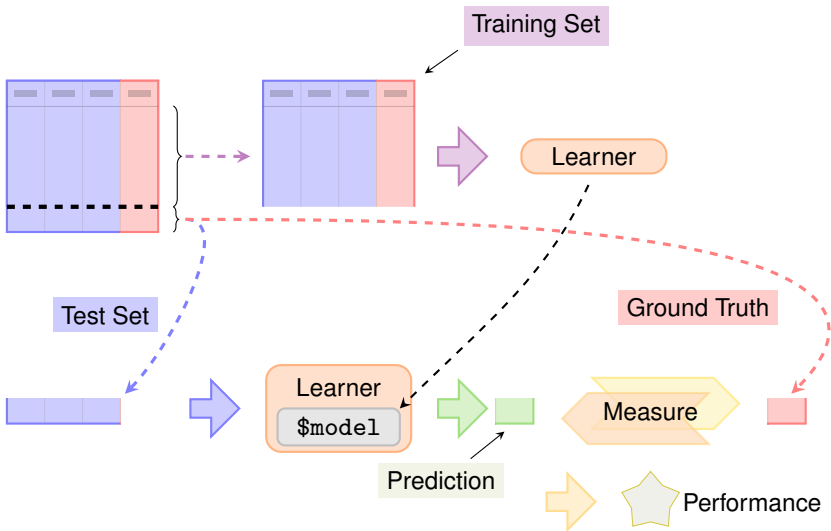
# RESAMPLING



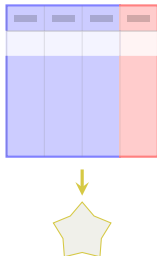
# RESAMPLING



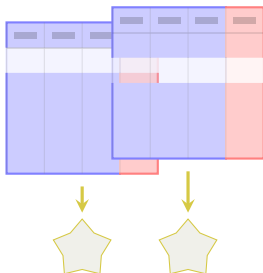
# RESAMPLING



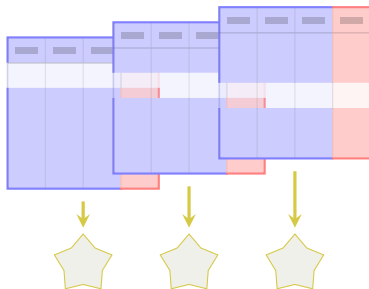
# RESAMPLING



# RESAMPLING

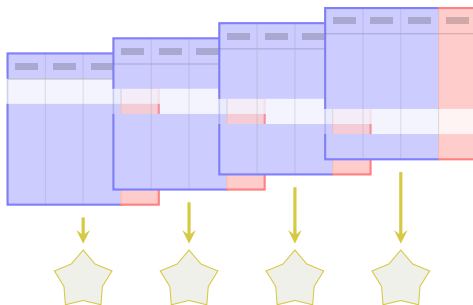


# RESAMPLING

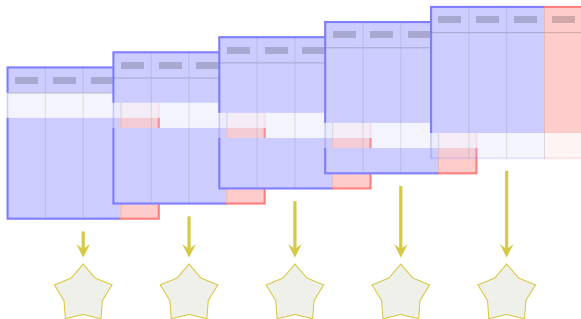




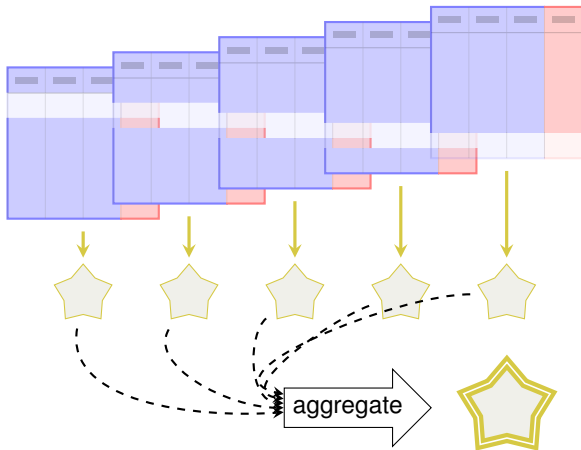
# RESAMPLING



# RESAMPLING

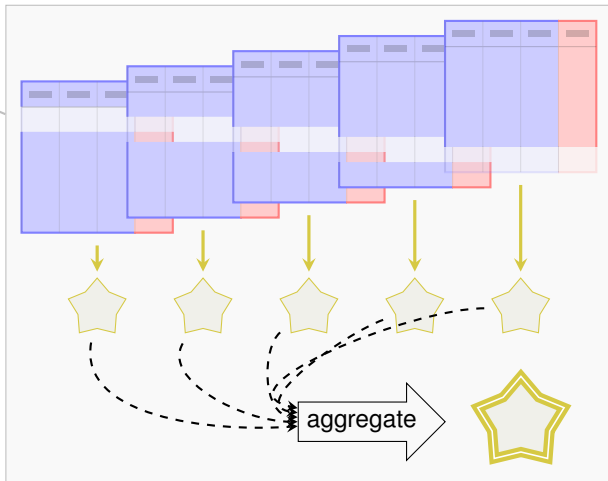


# RESAMPLING



# RESAMPLING

`resample()`



# RESAMPLING

- Resample description: How to split the data

```
cv5 = rsmp("cv", folds = 5)
```

# RESAMPLING

- Resample description: How to split the data

```
cv5 = rsmp("cv", folds = 5)
```

- Use the `resample()` function for resampling:

```
task = TaskClassif$new("iris", iris, "Species")  
learner = lrn("classif.rpart")  
rr = resample(task, learner, cv5)
```

# RESAMPLING

- Resample description: How to split the data

```
cv5 = rsmp("cv", folds = 5)
```

- Use the `resample()` function for resampling:

```
task = TaskClassif$new("iris", iris, "Species")  
learner = lrn("classif.rpart")  
rr = resample(task, learner, cv5)
```

- We get a `ResamplingResult` object:

```
print(rr)  
#> <ResampleResult> of 5 iterations  
#> * Task: iris  
#> * Learner: classif.rpart  
#> * Warnings: 0 in 0 iterations  
#> * Errors: 0 in 0 iterations
```

# RESAMPLING RESULTS

What exactly is a `ResamplingResult` object?



# RESAMPLING RESULTS

What exactly is a `ResamplingResult` object?

Remember Prediction:

# RESAMPLING RESULTS

What exactly is a ResamplingResult object?

Remember Prediction:

- Get a table representation using `as.data.table()`

```
rr_table = as.data.table(rr)

print(rr_table)
```

#	task	learner	resampling
# 1:	<TaskClassif[47]>	<LearnerClassifRpart[36]>	<ResamplingCV[19]>
# 2:	<TaskClassif[47]>	<LearnerClassifRpart[36]>	<ResamplingCV[19]>
# 3:	<TaskClassif[47]>	<LearnerClassifRpart[36]>	<ResamplingCV[19]>
# 4:	<TaskClassif[47]>	<LearnerClassifRpart[36]>	<ResamplingCV[19]>
# 5:	<TaskClassif[47]>	<LearnerClassifRpart[36]>	<ResamplingCV[19]>
#	iteration	prediction	
# 1:	1	<PredictionClassif[19]>	
# 2:	2	<PredictionClassif[19]>	
# 3:	3	<PredictionClassif[19]>	
# 4:	4	<PredictionClassif[19]>	
# 5:	5	<PredictionClassif[19]>	

# RESAMPLING RESULTS

What exactly is a `ResamplingResult` object?

Remember Prediction:

- Get a table representation using `as.data.table()`

```
rr_table = as.data.table(rr)

print(rr_table)
```

#	task	learner	resampling
# 1:	<TaskClassif[47]>	<LearnerClassifRpart[36]>	<ResamplingCV[19]>
# 2:	<TaskClassif[47]>	<LearnerClassifRpart[36]>	<ResamplingCV[19]>
# 3:	<TaskClassif[47]>	<LearnerClassifRpart[36]>	<ResamplingCV[19]>
# 4:	<TaskClassif[47]>	<LearnerClassifRpart[36]>	<ResamplingCV[19]>
# 5:	<TaskClassif[47]>	<LearnerClassifRpart[36]>	<ResamplingCV[19]>
#	iteration	prediction	
# 1:	1	<PredictionClassif[19]>	
# 2:	2	<PredictionClassif[19]>	
# 3:	3	<PredictionClassif[19]>	
# 4:	4	<PredictionClassif[19]>	
# 5:	5	<PredictionClassif[19]>	

- Active bindings and functions that make information easily accessible

# RESAMPLING RESULTS

- Calculate performance:

```
rr$aggregate(msr("classif.ce"))  
#> classif.ce  
#>          0.06
```

# RESAMPLING RESULTS

- Calculate performance:

```
rr$aggregate(msr("classif.ce"))  
#> classif.ce  
#>          0.06
```

- Get predictions

```
rr$prediction()  
#> <PredictionClassif> for 150 observations:  
#>      row_ids      truth  response  
#>          11      setosa    setosa  
#>          13      setosa    setosa  
#>          14      setosa    setosa  
#> ---  
#>        142 virginica virginica  
#>        144 virginica virginica  
#>        147 virginica virginica
```

# RESAMPLING

- Predictions of individual folds

```
predictions = rr$predictions()
predictions[[1]]

#> <PredictionClassif> for 30 observations:
#>      row_ids      truth  response
#>         11      setosa    setosa
#>         13      setosa    setosa
#>         14      setosa    setosa
#> ---
#>        136 virginica virginica
#>        138 virginica virginica
#>        150 virginica virginica
```

# RESAMPLING

- Predictions of individual folds

```
predictions = rr$predictions()
predictions[[1]]

#> <PredictionClassif> for 30 observations:
#>      row_ids      truth  response
#>      11      setosa    setosa
#>      13      setosa    setosa
#>      14      setosa    setosa
#> ---
#>      136 virginica virginica
#>      138 virginica virginica
#>      150 virginica virginica
```

- Score of individual folds

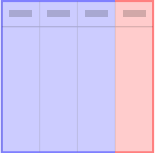



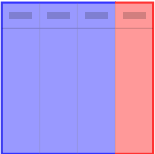



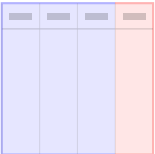



```
scores = rr$score()
scores[1:3, c("iteration", "classif.ce")]

#>      iteration classif.ce
#> 1:           1      0.067
#> 2:           2      0.100
#> 3:           3      0.033
```

# Benchmark



# PERFORMANCE COMPARISON

	Learner 1	Learner 2	Learner 3
			
			
			

# PERFORMANCE COMPARISON

- Multiple Learners, multiple Tasks:

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

# 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)
```

# 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
#> 1:   iris classif.rpart    0.060
#> 2:   iris classif.kknn    0.040
#> 3: sonar classif.rpart    0.283
#> 4: sonar classif.kknn    0.144
#> 5:  wine classif.rpart    0.107
#> 6:  wine classif.kknn    0.051
```

# BENCHMARK RESULT

What exactly is a `BenchmarkResult` object?

# BENCHMARK RESULT

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

# BENCHMARK RESULT

What exactly is a `BenchmarkResult` object?

Just like `Prediction` and `ResamplingResult`!

- Table representation using `as.data.table()`

# BENCHMARK RESULT

What exactly is a `BenchmarkResult` object?

Just like `Prediction` and `ResamplingResult`!

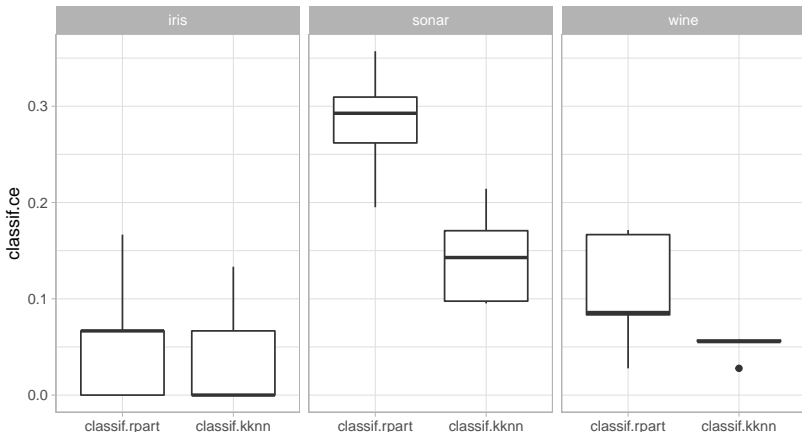
- Table representation using `as.data.table()`
- Active bindings and functions that make information easily accessible



# BENCHMARK RESULT

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

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



# Control of Execution

# CONTROL OF EXECUTION

## Parallelization

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

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

## Encapsulation

```
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
- Possibility to have a fallback to create predictions

# How to get Help

# 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?
  - ❶ Find out what kind of R6 object you have:

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

- ❷ Go to the corresponding help page:

```
?BenchmarkResult
```

New: open the corresponding man page with

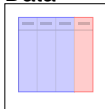
```
learner$help()
```

# Outro

# OVERVIEW

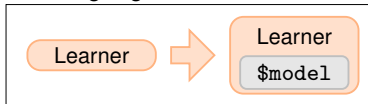
Ingredients:

Data



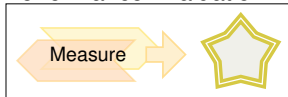
`TaskClassif,`  
`TaskRegr,`  
`tsk()`

Learning Algorithms



`lrn()`  $\Rightarrow$  Learner,  
 $\hookrightarrow$  `Learner$train()`,  
 $\hookrightarrow$  `Learner$predict()`  $\Rightarrow$  Prediction

Performance Evaluation



`rsmp()`  $\Rightarrow$  Resampling,  
`msr()`  $\Rightarrow$  Measure,  
`resample()`  $\Rightarrow$  ResamplingResult,  
 $\hookrightarrow$  `ResamplingResult$score()`,  
 $\hookrightarrow$  `ResamplingResult$aggregate()`

Performance Comparison



`benchmark_grid()`,  
`benchmark()`  $\Rightarrow$  BenchmarkResult