MostSimple Prediction Task with mlr3

川田恵介

Purpose

- 予測問題についての包括的な作業工程を例示
  - ただし Stacking は除く
- TuningParameter の推定と本推定を同時に行う AutoTuner を活用
- モデル比較を簡便かつ柔軟に行える benchmark を活用

RoadMap

1. Define Learner, Measurement, ReSampling, TerminalCondition
2. Define Learner with Parameter Turning
3. Split sample into Training and Test
4. Choose Best Model by CrossValidation
5. Construct Final Model

SetUp

```r
library(mlr3verse)
library(tidyverse)

set.seed(1)

Data <- read_csv("ExampleData/Example.csv")

Task <- as_task_regr(Data,
                     "Price") # Define Task
```

```r
Subgroup <- partition(Task, ratio = 0.8)

R2 <- msr("regr.rsq") # Define Evaluation with R2

Mean <- lrn("regr.featureless") # Define SimpleMean
OLS <- lrn("regr.lm") # Define OLS
Tree <- lrn("regr.rpart") # Define AdaptiveTree
RandomForest <- lrn("regr.ranger") # Define Random Forest
LASSO <- lrn("regr.cv_glmnet") # Define LASSO
```

Tuning

- TuningParameter の推定を行うアルゴリズムを定義

```r
CV <- rsmp("cv",folds = 2) # Define CrossValidation with 2 folds

Tuner <- tnr("grid_search") # Define search method

Terminator <- trm("evals", n_evals = 100) # Define Terminal condition
```

PruneTree

- Prune Tree の推定

```r
AutoTree <- AutoTuner$new(
  learner = Tree,
  resampling = CV,
  measure = R2,
  search_space =  ps(
  cp = p_dbl(lower = 0, upper = 0.1),
  minsplit = p_int(lower = 5,upper = 20)
  ), # Define hyperparameter space
  tuner = Tuner,
  terminator = Terminator,
  store_models = TRUE
)

AutoTree$id <- "regr.PruneTree"
```

ElasticNet

- ElasticNet を推定

```r
ElasNet <- AutoTuner$new(
  learner = lrn("regr.glmnet"),
  resampling = CV,
  measure = R2,
  search_space =  ps(
  lambda = p_dbl(lower = 0, upper = 1),
  alpha = p_dbl(lower = 0 , upper = 1)
  ), # Define hyperparameter space
  tuner = Tuner,
  terminator = Terminator,
  store_models = TRUE
)

ElasNet$id <- "regr.ElasticNet"
```

BenchMaking

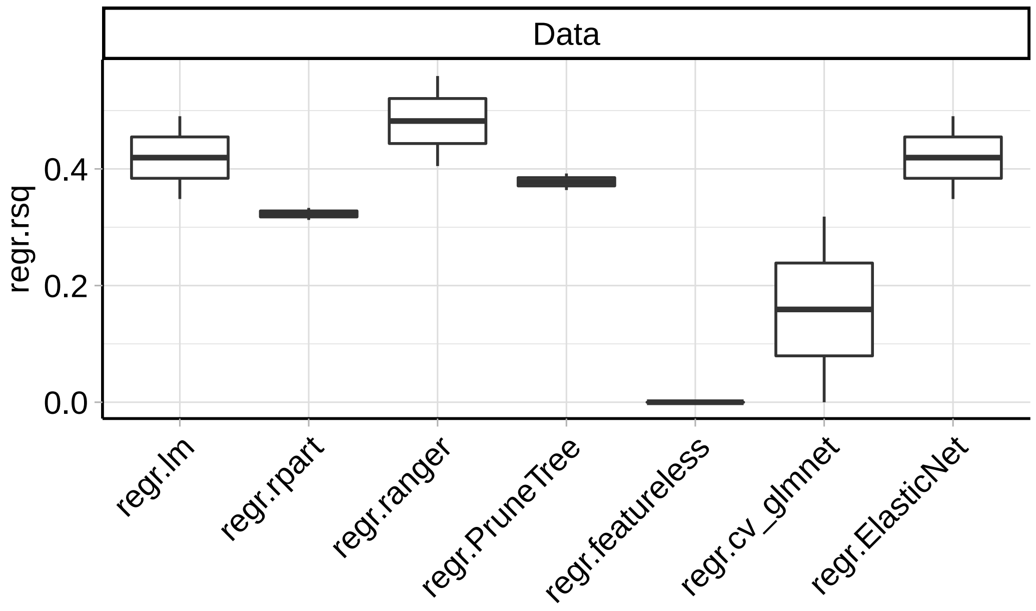
- 訓練データのみを用いて、アルゴリズムを比較 (TrainingData 内の CrossValidation)

```r
Design <- benchmark_grid(
  tasks = Task$clone(deep=TRUE)$filter(Subgroup$train),
  learners = list(OLS,Tree,RandomForest,AutoTree,Mean,LASSO,ElasNet),
  resamplings = CV
)

Result <- benchmark(Design)
```

可視化

```r
autoplot(Result,
         measure = R2)
```



Final Model

- 最善のアルゴリムであった RandomForest、及び全訓練データを用いて最終予測モデルを推定

```
RandomForest$train(Task,Subgroup$train)

RandomForest$predict(Task,Subgroup$test)$score(R2) # Peformance in TestData
```

```
 regr.rsq
0.6043542
```