R 机器学习

第 06 讲　正则化回归

张敬信

2022 年 9 月 25 日

哈尔滨商业大学

一. 正则化回归模型

- 多元线性回归中，通常会在众多变量中选择对因变量显著性影响大的那些自变量。但常常会遇到一个问题：在某些情况下，增加或剔除一个自变量后，回归系数变化很大甚至改变符号，主要原因就是变量之间存在多重共线性，得到回归模型是 “伪回归”。
- 这时就需要岭回归、Lasso 回归、弹性网回归，它们都是基于一种正则化技术（可减少过拟合）。

1. 岭回归

- 岭回归是一种改良的最小二乘法，通过放弃最小二乘法的无偏性，以损失部分信息、降低精度为代价，获得回归系数更为符合实际、更可靠的回归方法。
- 这种改良是通过对损失函数添加 l_2 正则项（也称为惩罚项）来实现的：

$$
\begin{aligned}
J(\beta) &= \sum_{i=1}^{n}(y_i - X\beta)^2 + \lambda\|\beta\|_2^2 \\
&= \sum_{i=1}^{n}(y_i - X\beta)^2 + \lambda\sum_{i=1}^{n}|\beta_i|^2
\end{aligned}
$$

- 其中正则项系数 λ 为非负实数，用于平衡方差（回归系数的方差）和偏差
- 当 $\lambda = 0$ 时，损失函数退化为多元线性回归模型的损失函数
- 当 $\lambda \to \infty$ 时，会缩减回归系数 β 使其趋于 0
- β 为模型参数，λ 是需要调参的超参数。

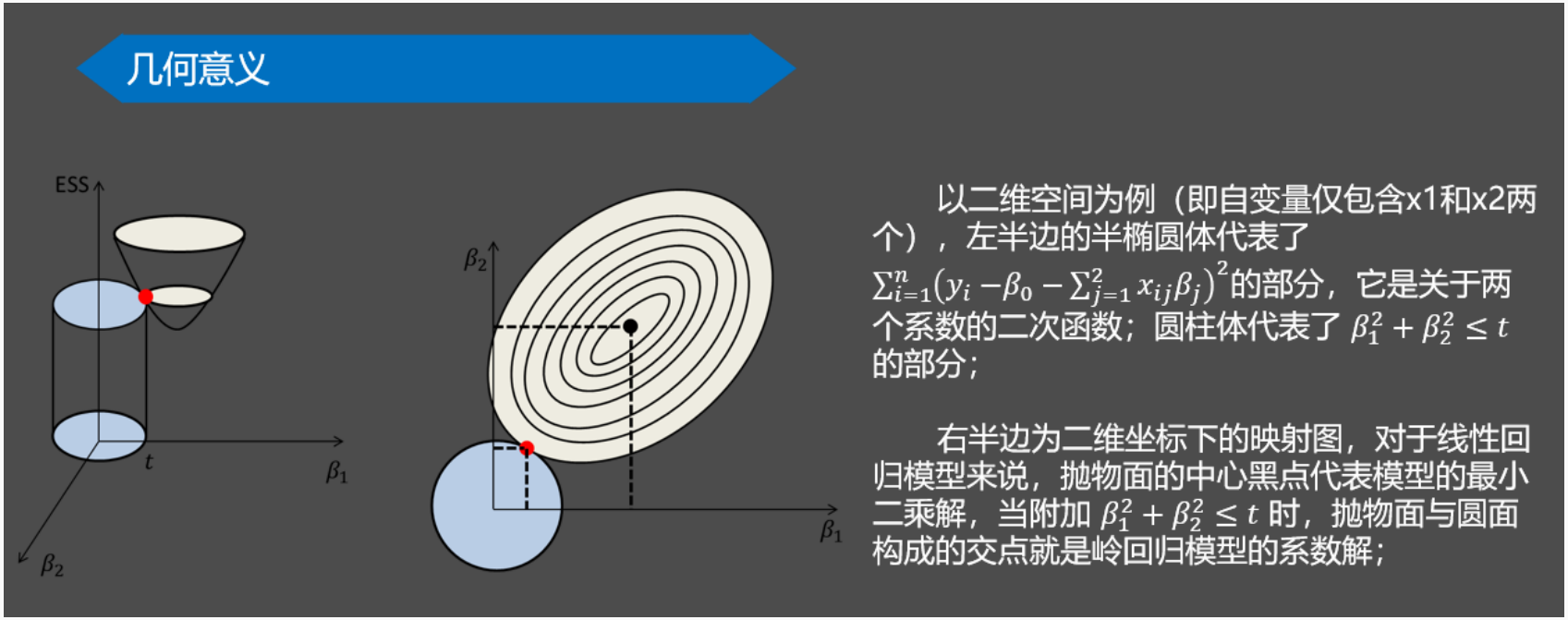
- 岭回归模型的等价表示（拉格朗日乘数法）：

$$J(\beta) = \sum (y - X\beta)^2 + \lambda \|\beta\|_2^2 = \sum (y - X\beta)^2 + \lambda \sum \beta^2$$

等价

$$\begin{cases} argmin\left\{\sum (y - X\beta)^2\right\} \\ \text{附加约束 } \sum \beta^2 \le t \end{cases}$$

- 岭回归模型正则项的几何意义：



2. Lasso 回归

- 岭回归添加 l_2 正则的惩罚项，但缺陷在于始终保留建模时的所有变量，无法降低模型的复杂度（减少变量）。
- 对于此，若改为添加 l_1 正则的惩罚项，就得到 Lasso 回归模型：

$$
\begin{aligned}
J(\beta) &= \sum_{i=1}^{n}(y_i - X\beta)^2 + \lambda \cdot \frac{1}{2}\|\beta\|_1^2 \\
&= \sum_{i=1}^{n}(y_i - X\beta)^2 + \lambda \cdot \frac{1}{2}\sum_{i=1}^{n}|\beta_i|
\end{aligned}
$$

- β 为模型参数, λ 是需要调参的超参数。

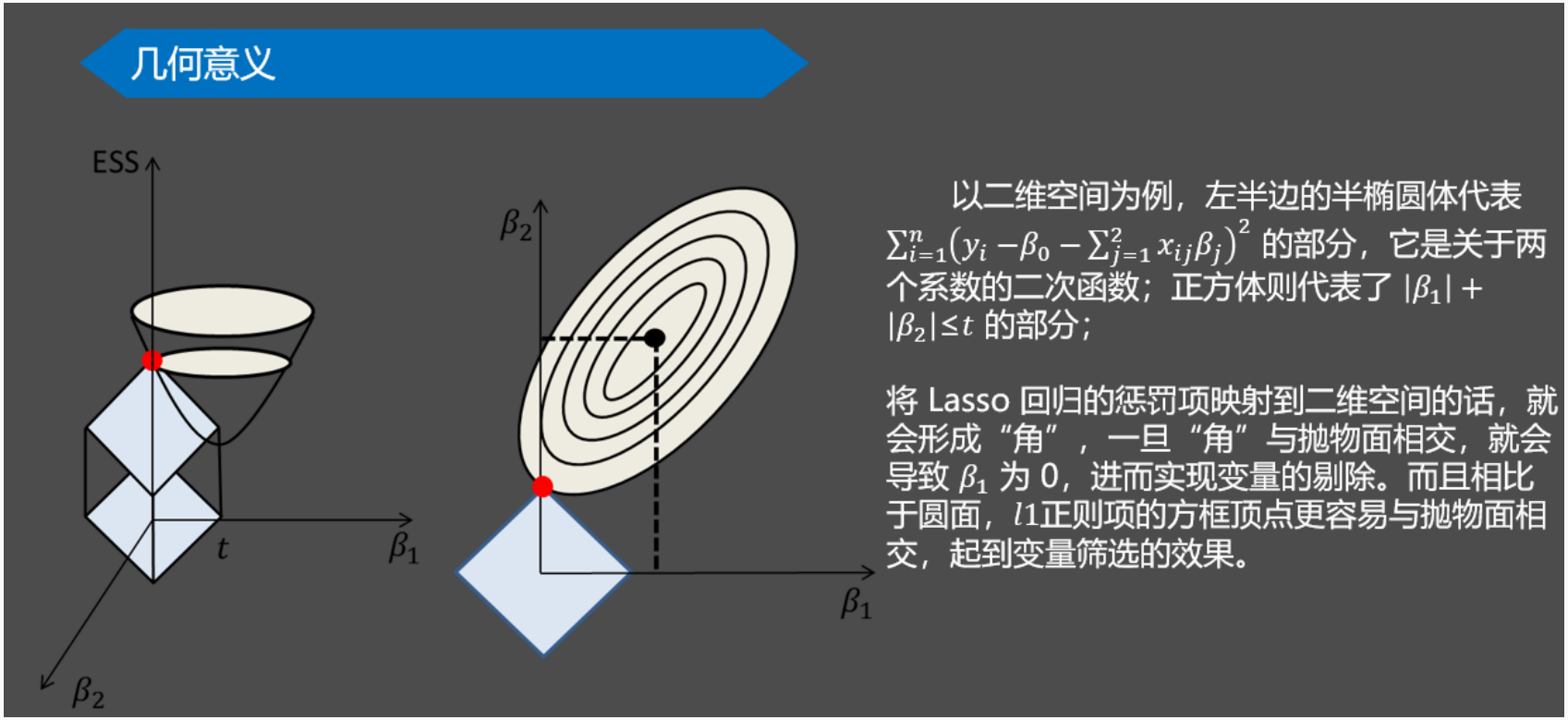
- Lasso 回归模型的等价表示（拉格朗日乘数法）：

$$J(\beta) = \sum (y - X\beta)^2 + \lambda \|\beta\|_1 = \sum (y - X\beta)^2 + \sum \lambda |\beta|$$

等价

$$\begin{cases} argmin\left\{\sum (y - X\beta)^2\right\} \\ \text{附加约束 } \sum |\beta| \le t \end{cases}$$

- Lasso 回归模型正则项的几何意义：



3. 弹性网模型

- 弹性网模型是将岭回归和 Lasso 回归结合起来，混合 l_1 正则项和 l_2 正则项，用新参数 α 控制二者所占比重：

$$
\begin{aligned}
J(\beta) &= \sum_{i=1}^{n}(y_i - X\beta)^2 + \lambda\left[(1-\alpha)\|\beta\|_2^2 + \alpha\|\beta\|_1^2\right] \\
&= \sum_{i=1}^{n}(y_i - X\beta)^2 + \lambda\left[(1-\alpha)\cdot\frac{1}{2}\sum_{i=1}^{n}|\beta_i|^2 + \alpha\sum_{i=1}^{n}|\beta_i|\right]
\end{aligned}
$$

- $\alpha = 0$ 退化为岭回归, $\alpha = 1$ 退化为 Lasso 回归
- β 为模型参数, α 和 λ 都是需要调参的超参数

二. 正则化回归案例

- `mlr3` 做正则化回归是调用的 `glmnet` 包实现岭回归、Lasso 回归、弹性网回归，提供了两个学习器：
  - “regr.glmnet”: 调用 `glmnet::glmnet()`
  - “regr.cv_glmnet”: 调用 `glmnet::cv.glmnet()`

- 两个原始函数的基本格式为：

```
glmnet(x, y, family=..., alpha=1, lambda=NULL, ...)
cv.glmnet(x, y, family=..., alpha=1, lambda=NULL, ...)
```

- 其中，
  - x 为自变量数据，y 为因变量数据；
  - family 指定因变量的类型，默认为 “gaussian”（适合一般连续变量），还可以是 “binomial”, “poisson”, “multinomial”, “cox”, “mgaussian”；
  - alpha 为弹性网混合系数，0 为岭回归，1 为 Lasso 回归，介于 0 和 1 之间为弹性网回归；
  - lambda 设定惩罚程度的超参数
- **二者区别**：cv.glmnet() 已经带有交叉验证对 lambda 调参，根据预测错误率或 RMSE 最小选择最优 lambda

1. 准备数据

- 本例使用自带任务 boston_housing（波士顿房价数据），先加载包

```
library(tidyverse)
library(mlr3verse)
library(glmnet)          # 正则化广义回归模型
```

- 从 mlr3 自带任务 boston_housing 中提取 Boston 房价数据

```
boston = tsk("boston_housing")$data()
glimpse(boston)
#> Rows: 506
#> Columns: 19
#> $ medv    <dbl> 24.0, 21.6, 34.7, 33.4, 36.2, 28.7, 22.9,
#> $ age     <dbl> 65.2, 78.9, 61.1, 45.8, 54.2, 58.7, 66.6,
#> $ b       <dbl> 397, 397, 393, 395, 397, 394, 396, 397, 38
#> $ chas    <fct> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
#> $ cmedv   <dbl> 24.0, 21.6, 34.7, 33.4, 36.2, 28.7, 22.9,
#> $ crim    <dbl> 0.00632, 0.02731, 0.02729, 0.03237, 0.0690
#> $ dis     <dbl> 4.09, 4.97, 4.97, 6.06, 6.06, 6.06, 5.56,
#> $ indus   <dbl> 2.31, 7.07, 7.07, 2.18, 2.18, 2.18, 7.87,
#> $ lat     <dbl> 42.3, 42.3, 42.3, 42.3, 42.3, 42.3, 42.3,
#> $ lon     <dbl> -71.0, -71.0, -70.9, -70.9, -70.9, -70.9,
#> $ lstat   <dbl> 4.98, 9.14, 4.03, 2.94, 5.33, 5.21, 12.43,
#> $ nox     <dbl> 0.538, 0.469, 0.469, 0.458, 0.458, 0.458,
```

额外步骤

- glmnet 包比较特殊，不能直接处理因子型变量，所以需要做额外的数据准备：
  - town 水平值过于分散，为了简单直接从特征变量中移除它
  - chas 是两水平因子，处理成虚拟变量
- 函数 model_matrix() 根据回归公式，将自变量数据转换为模型数据，因子型变量将处理成虚拟变量，但多了一列都是 1 的截距列，需移除它

```
boston = boston[,-"town"]
x = modelr::model_matrix(boston, medv ~ .)[,-1]
boston = bind_cols(x, boston["medv"])
```

2. 创建任务

```
bh_task = as_task_regr(boston, target = "medv")
bh_task
#> <TaskRegr:boston> (506 x 18)
#> * Target: medv
#> * Properties: -
#> * Features (17):
#>   - dbl (17): age, b, chas1, cmedv, crim, dis, indus, lat,
#>     nox, ptratio, rad, rm, tax, tract, zn
```

3. 划分训练集和测试集

- 做留出 (holdout) 重抽样，80% 作为训练集，其余 20% 作为测试集
- 为了保持训练集、测试集的因变量数据具有相似的分布，采用分层抽样方法
- 用 partition( ) 函数对任务做划分，默认按因变量分层，取出训练集索引和测试集索引

```
set.seed(123)
split = partition(bh_task, ratio = 0.8)
# 默认 stratify = TRUE
```

(1) 选择学习器

```
learner = lrn("regr.glmnet", alpha = 0)   # 需要 glmnet 包
learner
#> <LearnerRegrGlmnet:regr.glmnet>
#> * Model: -
#> * Parameters: family=gaussian, alpha=0
#> * Packages: mlr3, mlr3learners, glmnet
#> * Predict Types:  [response]
#> * Feature Types: logical, integer, numeric
#> * Properties: weights
```

注：可用 `lrns()` 查看所有可用的学习器，更多学习器在 `mlr3extralearners` 包。

(2) 超参数调参

- 查看模型的所有超参数及默认值

```
learner$param_set
#> <ParamSet>
#>                     id    class lower upper nlevels
#>  1:             alignment ParamFct    NA    NA       2
#>  2:                 alpha ParamDbl     0     1     Inf
#>  3:                   big ParamDbl  -Inf   Inf     Inf
#>  4:                devmax ParamDbl     0     1     Inf
#>  5:                 dfmax ParamInt     0   Inf     Inf <NoD
#>  6:                   eps ParamDbl     0     1     Inf
#>  7:                epsnr ParamDbl     0     1     Inf
#>  8:                exact ParamLgl    NA    NA       2
#>  9:              exclude ParamInt     1   Inf     Inf <NoD
#> 10:                 exmx ParamDbl  -Inf   Inf     Inf
#> 11:               family ParamFct    NA    NA       2
```

- 需要对模型中的惩罚超参数 λ 做调参，对应超参数中的 s
- 使用自动调参器，需要设置学习器、重抽样方法、模型评估指标、搜索空间、终止条件、搜索方法

```
library(paradox)
search_space = ps(
  s = p_dbl(lower = 0.001, upper = 10))

at = AutoTuner$new(
  learner = learner,
  resampling = rsmp("cv", folds = 3L),      # 3 折交叉验证
  measure = msr("regr.rmse"),               # 评估指标选 rmse
  search_space = search_space,
  terminator = trm("evals", n_evals = 10), # 计算 10 次终止
  tuner = tnr("random_search"))             # 随机搜索
```

- 启动自动调参过程

```
at$train(bh_task, row_ids = split$train)
#> INFO  [21:25:16.612] [bbotk] Starting to optimize 1 parame
#> INFO  [21:25:16.649] [bbotk] Evaluating 1 configuration(s)
#> INFO  [21:25:16.688] [mlr3] Running benchmark with 3 resam
#> INFO  [21:25:16.727] [mlr3] Applying learner 'regr.glmnet'
#> INFO  [21:25:16.779] [mlr3] Applying learner 'regr.glmnet'
#> INFO  [21:25:16.804] [mlr3] Applying learner 'regr.glmnet'
#> INFO  [21:25:16.827] [mlr3] Finished benchmark
#> INFO  [21:25:16.861] [bbotk] Result of batch 1:
#> INFO  [21:25:16.863] [bbotk]     s regr.rmse warnings erro
#> INFO  [21:25:16.863] [bbotk]  2.87      2.37        0
#> INFO  [21:25:16.863] [bbotk]
#> INFO  [21:25:16.863] [bbotk]  bf8bb37c-500a-4d53-b076-4ec9
#> INFO  [21:25:16.865] [bbotk] Evaluating 1 configuration(s)
#> INFO  [21:25:16.896] [mlr3] Running benchmark with 3 resam
#> INFO  [21:25:16.901] [mlr3] Applying learner 'regr.glmnet'
```

- 查看最优超参数

```
at$tuning_result
#>         s learner_param_vals  x_domain regr.rmse
#> 1: 0.262          <list[3]> <list[1]>       1.3
```

(3) 训练模型

- 用调出的最优参数更新学习器的参数集

```
learner$param_set$values =
  at$tuning_result$learner_param_vals[[1]]

learner$train(bh_task, row_ids = split$train)
```

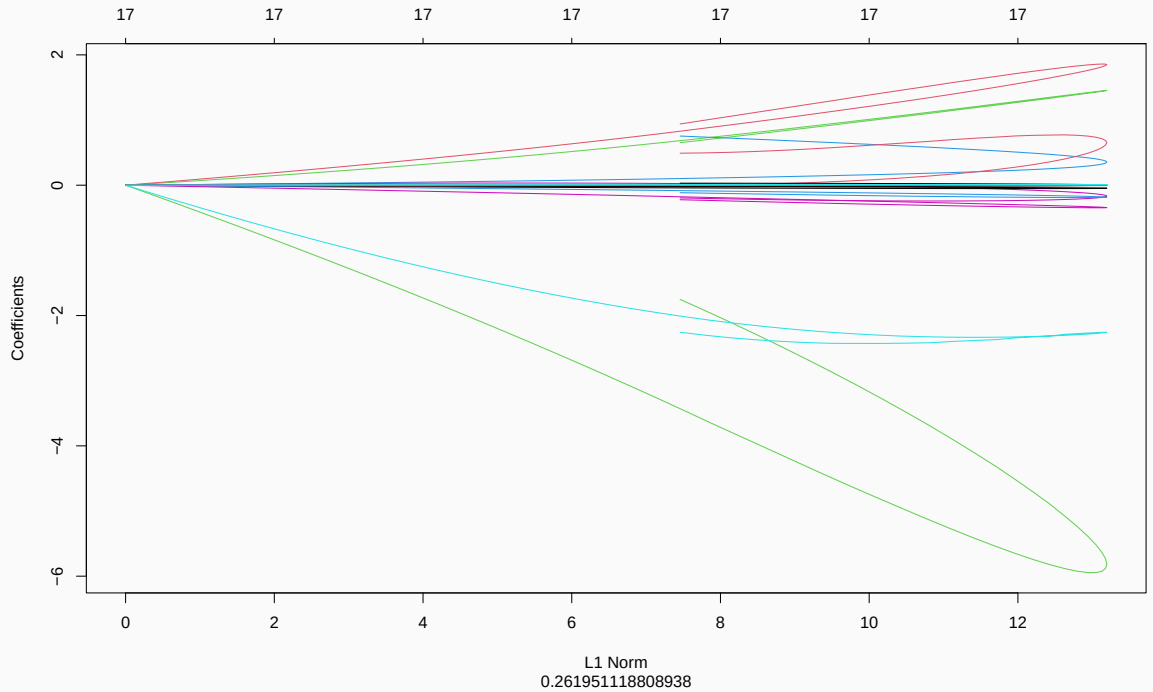
注：也可以用最优的超参数重新构建学习器

```
learner = lrn("regr.glmnet", alpha = 0,
              s = at$tuning_result$s)
```

- 提取岭回归模型系数：

```
coef(learner$model, s = at$tuning_result$s)
#> 18 x 1 sparse Matrix of class "dgCMatrix"
#>                       s1
#> (Intercept) -1.38e+02
#> age          5.02e-04
#> b            1.93e-03
#> chas1        6.53e-01
#> cmedv        7.54e-01
#> crim        -2.39e-02
#> dis         -1.95e-01
#> indus       -1.37e-02
#> lat          4.91e-01
#> lon         -1.75e+00
#> lstat       -1.15e-01
#> nox         -2.26e+00
#> ptratio     -2.23e-01
```

```
plot(learner$model, s = at$tuning_result$s)
```



(4) 模型预测

```
predictions = learner$predict(bh_task, row_ids = split$test)
predictions
#> <PredictionRegr> for 102 observations:
#>     row_ids truth response
#>           2  21.6    22.45
#>           3  34.7    33.60
#>          30  21.0    21.10
#> ---
#>         476  13.3    13.87
#>         487  19.1    19.19
#>         490   7.0     7.87
```

(5) 模型评估

```
tibble(RMSE = predictions$score(msr("regr.rmse")),
       RSquare = predictions$score(msr("regr.rsq")))
#> # A tibble: 1 x 2
#>    RMSE RSquare
#>   <dbl>   <dbl>
#> 1  1.29   0.980
```

注：用 msrs() 可查看所有可用的模型评估指标。

(6) 预测新数据

```
newdata = boston[1:5,-18]
learner$predict_newdata(newdata)
#> <PredictionRegr> for 5 observations:
#>  row_ids truth response
#>        1    NA     25.5
#>        2    NA     22.5
#>        3    NA     33.6
#>        4    NA     32.3
#>        5    NA     34.3
```

(1) 选择学习器

```
learner = lrn("regr.glmnet", alpha = 1)   # 需要 glmnet 包
learner
#> <LearnerRegrGlmnet:regr.glmnet>
#> * Model: -
#> * Parameters: family=gaussian, alpha=1
#> * Packages: mlr3, mlr3learners, glmnet
#> * Predict Types:  [response]
#> * Feature Types: logical, integer, numeric
#> * Properties: weights
```

注：可用 `lrns()` 查看所有可用的学习器，更多学习器在 `mlr3extralearners` 包。

(2) 超参数调参

- 查看模型的所有超参数及默认值

```
learner$param_set
#> <ParamSet>
#>                     id    class lower upper nlevels
#>  1:             alignment ParamFct    NA    NA       2
#>  2:                 alpha ParamDbl     0     1     Inf
#>  3:                   big ParamDbl  -Inf   Inf     Inf
#>  4:                devmax ParamDbl     0     1     Inf
#>  5:                 dfmax ParamInt     0   Inf     Inf <NoD
#>  6:                   eps ParamDbl     0     1     Inf
#>  7:                epsnr ParamDbl     0     1     Inf
#>  8:                 exact ParamLgl    NA    NA       2
#>  9:               exclude ParamInt     1   Inf     Inf <NoD
#> 10:                  exmx ParamDbl  -Inf   Inf     Inf
#> 11:                family ParamFct    NA    NA       2
```

- 需要对模型中的惩罚超参数 λ 做调参，对应超参数中的 s
- 使用自动调参器，需要设置学习器、重抽样方法、模型评估指标、搜索空间、终止条件、搜索方法

```
search_space = ps(
  s = p_dbl(lower = 0.001, upper = 10))

at = AutoTuner$new(
  learner = learner,
  resampling = rsmp("cv", folds = 3L),      # 3 折交叉验证
  measure = msr("regr.rmse"),               # 评估指标选 rmse
  search_space = search_space,
  terminator = trm("evals", n_evals = 10), # 计算 10 次终止
  tuner = tnr("random_search"))             # 随机搜索
```

- 启动自动调参过程

```
at$train(bh_task, row_ids = split$train)
#> INFO  [21:25:18.667] [bbotk] Starting to optimize 1 parame
#> INFO  [21:25:18.675] [bbotk] Evaluating 1 configuration(s)
#> INFO  [21:25:18.712] [mlr3] Running benchmark with 3 resam
#> INFO  [21:25:18.718] [mlr3] Applying learner 'regr.glmnet'
#> INFO  [21:25:18.741] [mlr3] Applying learner 'regr.glmnet'
#> INFO  [21:25:18.765] [mlr3] Applying learner 'regr.glmnet'
#> INFO  [21:25:18.785] [mlr3] Finished benchmark
#> INFO  [21:25:18.815] [bbotk] Result of batch 1:
#> INFO  [21:25:18.817] [bbotk]     s regr.rmse warnings erro
#> INFO  [21:25:18.817] [bbotk]  6.54      6.58        0
#> INFO  [21:25:18.817] [bbotk]
#> INFO  [21:25:18.817] [bbotk]  91eed8de-a314-4855-8489-2273
#> INFO  [21:25:18.820] [bbotk] Evaluating 1 configuration(s)
#> INFO  [21:25:18.907] [mlr3] Running benchmark with 3 resam
#> INFO  [21:25:18.913] [mlr3] Applying learner 'regr.glmnet'
```

- 查看最优超参数

```
at$tuning_result
#>         s learner_param_vals  x_domain regr.rmse
#> 1: 0.501          <list[3]> <list[1]>     0.674
```

(3) 训练模型

- 用调出的最优参数更新学习器的参数集

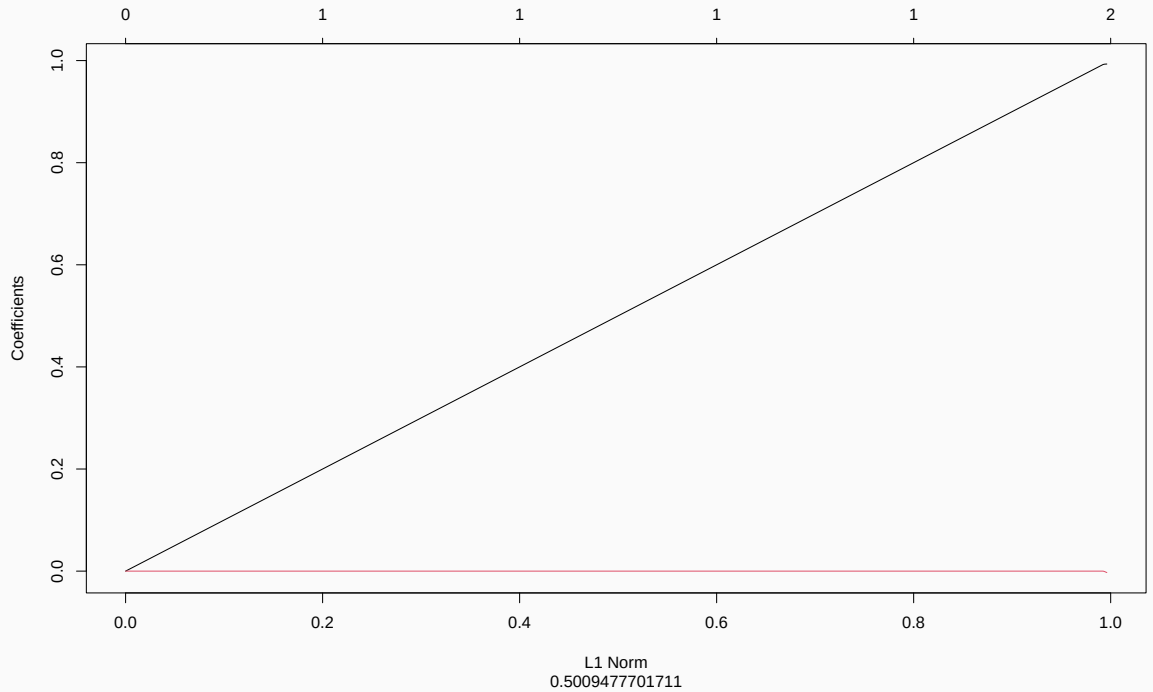
```
learner$param_set$values =
  at$tuning_result$learner_param_vals[[1]]

learner$train(bh_task, row_ids = split$train)
```

- 提取 Lasso 回归模型系数：

```
coef(learner$model, s = at$tuning_result$s)
#> 18 x 1 sparse Matrix of class "dgCMatrix"
#>                    s1
#> (Intercept) 1.242
#> age            .
#> b              .
#> chas1          .
#> cmedv          0.945
#> crim           .
#> dis            .
#> indus          .
#> lat            .
#> lon            .
#> lstat          .
#> nox            .
#> ptratio
```

```
plot(learner$model, s = at$tuning_result$s)
```



(4) 模型预测

```
predictions = learner$predict(bh_task, row_ids = split$test)
predictions
#> <PredictionRegr> for 102 observations:
#>     row_ids truth response
#>           2  21.6    21.66
#>           3  34.7    34.04
#>          30  21.0    21.09
#> ---
#>         476  13.3    13.81
#>         487  19.1    19.29
#>         490   7.0     7.86
```

(5) 模型评估

```
tibble(RMSE = predictions$score(msr("regr.rmse")),
       RSquare = predictions$score(msr("regr.rsq")))
#> # A tibble: 1 x 2
#>    RMSE RSquare
#>   <dbl>   <dbl>
#> 1 0.817   0.992
```

注：用 msrs() 可查看所有可用的模型评估指标。

(6) 预测新数据

```
newdata = boston[1:5,-18]
learner$predict_newdata(newdata)
#> <PredictionRegr> for 5 observations:
#>  row_ids truth response
#>        1    NA     23.9
#>        2    NA     21.7
#>        3    NA     34.0
#>        4    NA     32.8
#>        5    NA     35.5
```

(1) 选择学习器

```
learner = lrn("regr.glmnet")   # 需要 glmnet 包
learner
#> <LearnerRegrGlmnet:regr.glmnet>
#> * Model: -
#> * Parameters: family=gaussian
#> * Packages: mlr3, mlr3learners, glmnet
#> * Predict Types:  [response]
#> * Feature Types: logical, integer, numeric
#> * Properties: weights
```

注：可用 `lrns()` 查看所有可用的学习器，更多学习器在 `mlr3extralearners` 包。

(2) 超参数调参

- 查看模型的所有超参数及默认值

```
learner$param_set
#> <ParamSet>
#>                     id    class lower upper nlevels
#>  1:             alignment ParamFct    NA    NA       2
#>  2:                 alpha ParamDbl     0     1     Inf
#>  3:                   big ParamDbl  -Inf   Inf     Inf
#>  4:                devmax ParamDbl     0     1     Inf
#>  5:                 dfmax ParamInt     0   Inf     Inf <NoD
#>  6:                   eps ParamDbl     0     1     Inf
#>  7:                epsnr ParamDbl     0     1     Inf
#>  8:                 exact ParamLgl    NA    NA       2
#>  9:               exclude ParamInt     1   Inf     Inf <NoD
#> 10:                  exmx ParamDbl  -Inf   Inf     Inf
#> 11:                family ParamFct    NA    NA       2
```

- 需要对模型中的惩罚超参数 λ 以及权重参数 α 做调参，分别对应超参数中的 s 和 alpha
- 使用自动调参器，需要设置学习器、重抽样方法、模型评估指标、搜索空间、终止条件、搜索方法

```
search_space = ps(
  s = p_dbl(lower = 0.001, upper = 10),
  alpha = p_dbl(lower = 0, upper = 1))

at = AutoTuner$new(
  learner = learner,
  resampling = rsmp("cv", folds = 3L),       # 3 折交叉验证
  measure = msr("regr.rmse"),                # 评估指标选 rmse
  search_space = search_space,
  terminator = trm("evals", n_evals = 10), # 计算 10 次终止
  tuner = tnr("random_search"))              # 随机搜索
```

- 启动自动调参过程

```
at$train(bh_task, row_ids = split$train)
#> INFO  [21:25:20.643] [bbotk] Starting to optimize 2 parame
#> INFO  [21:25:20.654] [bbotk] Evaluating 1 configuration(s)
#> INFO  [21:25:20.685] [mlr3] Running benchmark with 3 resam
#> INFO  [21:25:20.692] [mlr3] Applying learner 'regr.glmnet'
#> INFO  [21:25:20.717] [mlr3] Applying learner 'regr.glmnet'
#> INFO  [21:25:20.740] [mlr3] Applying learner 'regr.glmnet'
#> INFO  [21:25:20.768] [mlr3] Finished benchmark
#> INFO  [21:25:20.800] [bbotk] Result of batch 1:
#> INFO  [21:25:20.801] [bbotk]      s  alpha regr.rmse warni
#> INFO  [21:25:20.801] [bbotk]  0.664 0.0376      1.03
#> INFO  [21:25:20.801] [bbotk]
#> INFO  [21:25:20.801] [bbotk]  94f0880a-de8f-479e-8166-fb31
#> INFO  [21:25:20.804] [bbotk] Evaluating 1 configuration(s)
#> INFO  [21:25:20.833] [mlr3] Running benchmark with 3 resam
#> INFO  [21:25:20.839] [mlr3] Applying learner 'regr.glmnet'
```

- 查看最优超参数

```
at$tuning_result
#>         s alpha learner_param_vals  x_domain regr.rmse
#> 1: 0.102 0.257          <list[3]> <list[2]>     0.396
```

(3) 训练模型

- 用调出的最优参数更新学习器的参数集

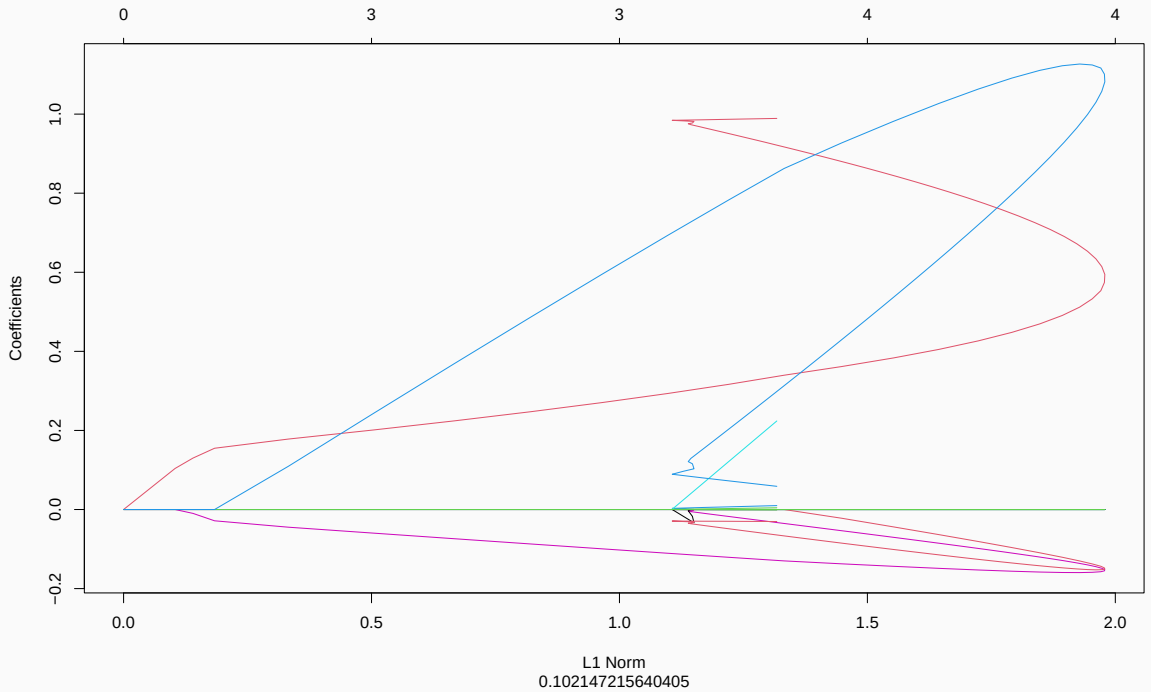
```
learner$param_set$values =
  at$tuning_result$learner_param_vals[[1]]

learner$train(bh_task, row_ids = split$train)
```

- 提取弹性网回归模型系数：

```
coef(learner$model, s = at$tuning_result$s)
#> 18 x 1 sparse Matrix of class "dgCMatrix"
#>                       s1
#> (Intercept)  4.78e-01
#> age           .
#> b             3.87e-05
#> chas1         .
#> cmedv         9.76e-01
#> crim         -4.59e-05
#> dis           .
#> indus         .
#> lat           .
#> lon           .
#> lstat        -4.11e-03
#> nox          -3.01e-03
#> ptratio      -3.47e-02
```

```
plot(learner$model, s = at$tuning_result$s)
```



(4) 模型预测

```
predictions = learner$predict(bh_task, row_ids = split$test)
predictions
#> <PredictionRegr> for 102 observations:
#>     row_ids truth response
#>           2  21.6    21.69
#>           3  34.7    34.58
#>          30  21.0    21.01
#> ---
#>         476  13.3    13.41
#>         487  19.1    19.11
#>         490   7.0     7.18
```

(5) 模型评估

```
tibble(RMSE = predictions$score(msr("regr.rmse")),
       RSquare = predictions$score(msr("regr.rsq")))
#> # A tibble: 1 x 2
#>    RMSE RSquare
#>   <dbl>   <dbl>
#> 1 0.653   0.995
```

注：用 msrs() 可查看所有可用的模型评估指标。

(6) 预测新数据

```
newdata = boston[1:5,-18]
learner$predict_newdata(newdata)
#> <PredictionRegr> for 5 observations:
#>  row_ids truth response
#>        1    NA     24.2
#>        2    NA     21.7
#>        3    NA     34.6
#>        4    NA     33.3
#>        5    NA     36.0
```

主要参考文献

[1] mlr3book. 2021. https://mlr3book.mlr-org.com/

[2] A. Kassambara, Machine Learning Essential: Practiale Guide in R. 2017. http://www.sthda.com

[3] 刘顺祥. 从零开始学 Python–数据分析与挖掘. 清华大学出版社, 2018.