第十届中国Stata用户大会（2026）

机器学习超参的贝叶斯优化与Stata应用

王群勇（南开大学经济学院）

吉林大学，长春，2026年7月5日

内容

贝叶斯优化简介

机器学习超参的贝叶斯优化

Stata应用

贝叶斯优化

在一些优化问题中，目标函数并不明确，无法计算目标函数的梯度向量，因此不能应用传统的拟牛顿法或梯度下降法。

贝叶斯优化是一种高斯过程的优化方法，用于解决黑盒优化问题（Bochu 等，2009; Frazier，2018 ）。

贝叶斯优化将未知函数用随机高斯过程来近似替代，从而将难以处理的原始问题转化为相对简单的优化问题。

格点搜索或随机搜索：格点搜索在错的区域耗费大量计算，而随机搜索不利用任何历史信息。

贝叶斯优化的优势来自两点：

- 平滑先验（GP核）让它在少量点上“插值/外推”出目标函数的趋势

在比较确定的区域“开发”（exploit），在不太确定的区域进行“探索”（explore）。

常用于超参调优（惩罚系数、学习率、树的深度、核函数、带宽等）

内容

简介

机器学习超参的贝叶斯优化

Stata应用

机器学习：

- 参数（parameters）
- 超参数（hyper-parameters）：事先设定

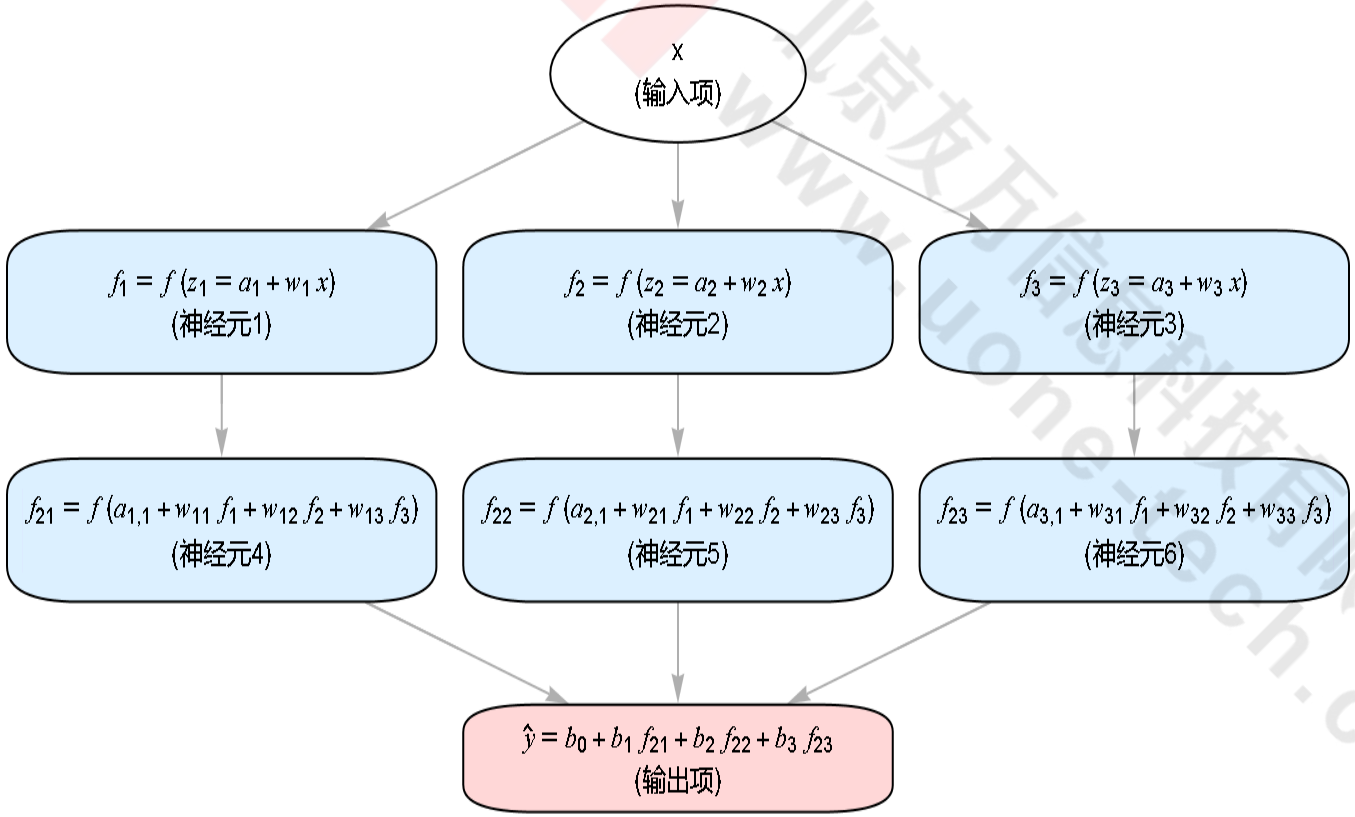
Lasso：

$$Obj = \frac{1}{2n}\sum_{i=1}^{n}(y_i - \boldsymbol{X}_i\beta)^2 + \left[\lambda\sum_{j=1}^{p}|\beta_j|\right].$$

随机森林：

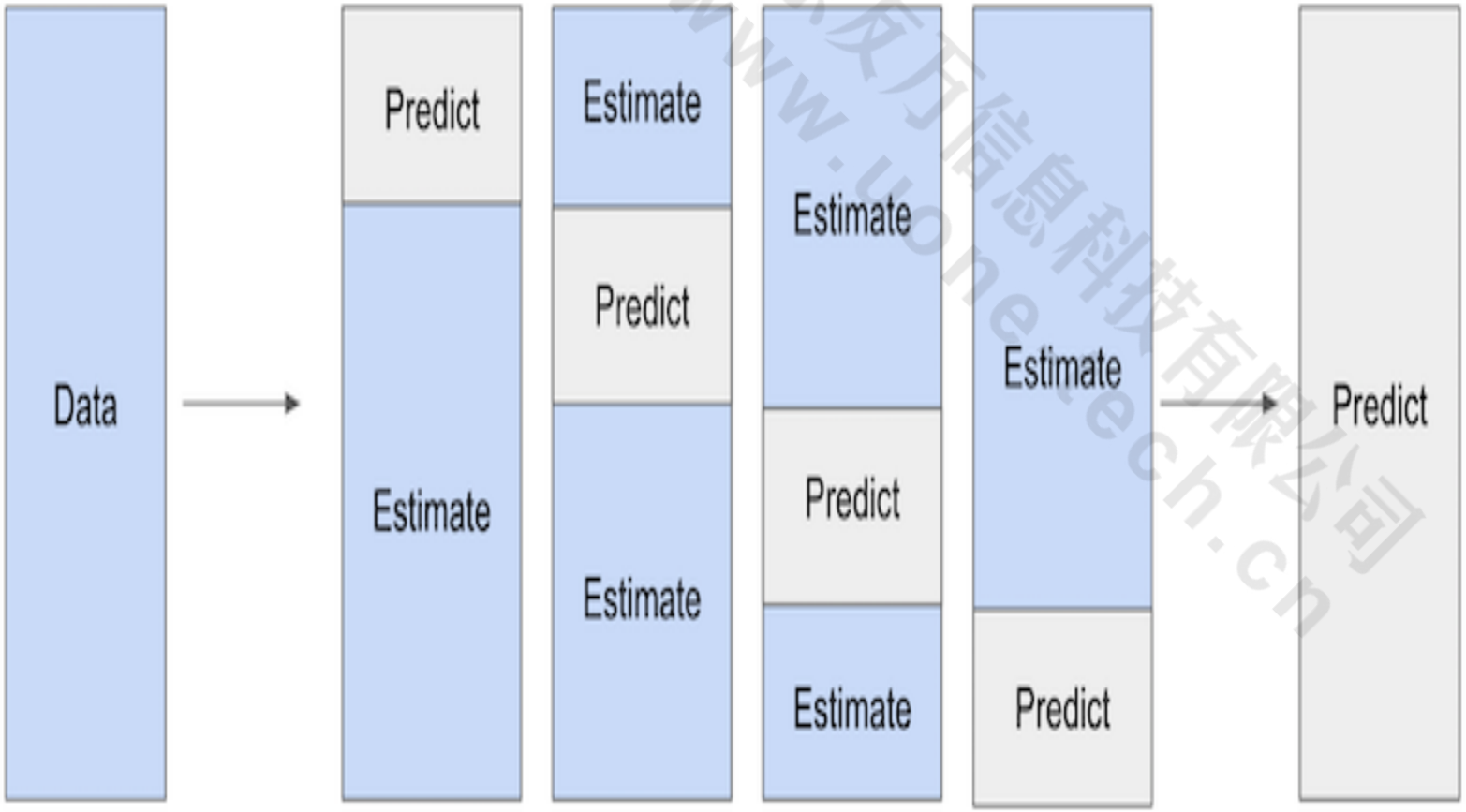
- 超参：树深度、每片叶子的最低样本量、每片叶子分割的最低样本量、随机抽样比例、随机抽取的特征比例。
- 参数：模型参数

神经网络模型：网络深度、神经元数量

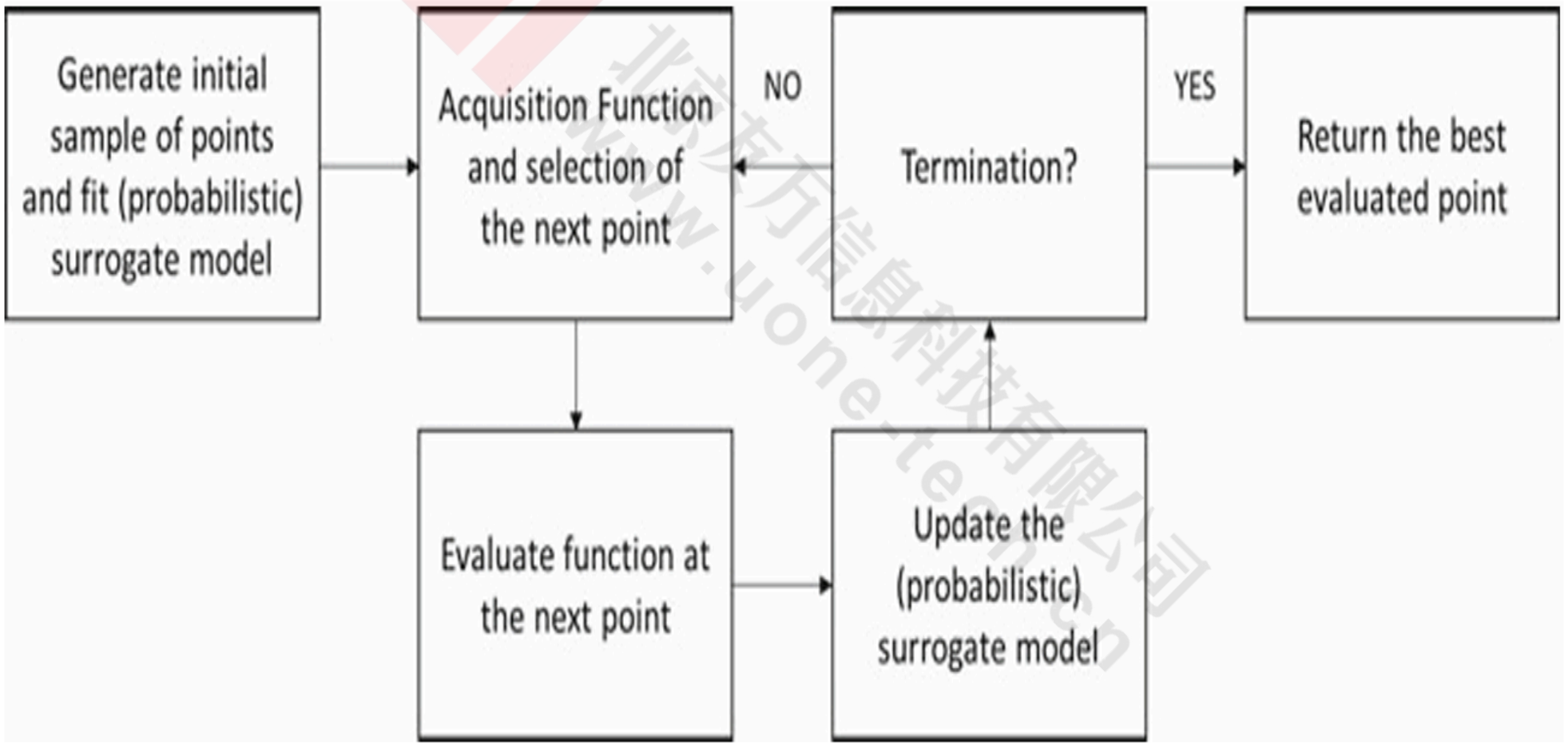


超参对于机器学习的预测至关重要。

交互校正法：



贝叶斯优化的基本步骤：



贝叶斯优化的三个核心要素：

（1）代理模型（Surrogate model）。最经典也最常用的为高斯过程（Gaussian Process, GP）。

（2）采集函数（Acquisition function），或者叫做utility function、infill criterion。常用的包括：EI（Expected Improvement）、UCB（Upper Confidence Bound）、PI（Probability of Improvement）。

（3）交替迭代（Alternation / Sequential rule）。这是BO核心算法，基本结构为：

第1步： 选初始点集$D = X_0$（随机/拉丁超立方），评估$f(x)$。

第2步：

- a. 用D拟合/更新代理模型（GP 后验）
- b. 计算采集函数$a(x)$（如 EI）
- c. 选$x_{t+1} = \mathrm{argmax}_{x \in X} a(x)$
- d. 评估 $y_{t+1} = f(x_{t+1}) + \epsilon_i$
- e. 更新$D = D \cup (x_{t+1}, y_{t+1})$

第3步：返回$\mathrm{argmin}_{x_t \in D} y_t$

高斯过程（Gaussian Process）：一组随机变量的任意集合服从高斯分布。GP完全由其均值函数$\mu(x)$和协方差函数$k(x,x')=Cov(f(x),f(x'))$来刻画。

$$f(x)\sim GP(\mu(x),k(x,x')).$$

高斯过程是对高斯随机向量的推广，可以将其视为基于一般连续函数的随机过程。之所以能这样处理，是因为高斯过程用核函数取代了传统的协方差矩阵，从而利用了核方法的优势。这种方法的核心在于计算条件分布。在贝叶斯框架下，这相当于从先验分布计算出后验分布。由于当随机变量服从高斯分布时，线性回归就是条件期望问题的解，因此很容易定义高斯过程回归模型。作为一种半参数机器学习模型（Rasmussen and Williams，2006），高斯过程已被应用于地理统计学（Cressie，1993 年）等很多领域。

设目标函数$f(\theta) \sim N(\mu, K)$>，观测数据为y，假定$(f(\theta), y)$服从联合高斯分布：

$$\begin{bmatrix} f(\boldsymbol{\theta}) \\ y \end{bmatrix} \sim \mathcal{N}\left(\begin{bmatrix} \mu_f \\ \mu_y \end{bmatrix}, \begin{bmatrix} K_{ff} & K_{fy} \\ K'_{fy} & K_{yy} \end{bmatrix} \right)$$

根据贝叶斯定理，$f(\theta)$的后验分布仍为高斯分布：

$$f(\theta|y) \sim N(K_{yf}K_{ff}^{-1}y + \mu_f, K_{yy} - K_{yf}K_{ff}^{-1}K_{fy})$$

采集函数是贝叶斯优化中实现探索(exploration）与开发（exploitation）权衡的核心机制。各类采集函数的目标均是引导寻优过程指向目标函数值可能较低的区域，这种潜力既可能源于概率代理模型预测的函数值$f(x)$偏低，也可能源于同一模型的不确定性较高（或二者兼有)。其中，"开发"旨在关注当前代理模型下最有可能改进现有解的区域；而"探索"则是向搜索空间中采样较少、基于代理模型的预测不确定性更高（方差更大）的区域推进。

Probability of improvement (PI)(Kushner, 1964):

$$PI(x) = P(f(x) \leq f(x^+)) = \Phi\left(\frac{f(x^+) - \mu(x)}{\sigma(x)}\right)$$

PI的一个主要缺陷在于其倾向于过度偏向开发。为缓解这一问题，可引入参数ξ以调节探索与开发之间的平衡，改进后的表达式如下：

$$PI(x) = P(f(x) \leq f(x^+) + \xi) = \Phi\left(\frac{f(x^+) - \mu(x) - \xi}{\sigma(x)}\right)$$

当 $\xi > 0$时，相当于提高了目标阈值，算法需要更大的提升才能满足条件，这促使算法去探索不确定性更高的区域（增加探索）。

当 $\xi < 0$时，降低目标阈值，算法更容易满足提升条件，从而更专注于当前看起来最优的区域（增加开发）。

$$x_{t+1} = \text{argmax}_{x \in X} PI(x)$$

$$\mathrm{EI}(x) = \begin{cases} (f(x^+) - \mu(x) - \xi)\Phi(Z) + \sigma(x)\phi(Z) & \text{if } \sigma(x) > 0 \\ 0 & \text{if } \sigma(x) = 0 \end{cases}$$

$$Z = \begin{cases} \dfrac{f(x^+) - \mu(x) - \xi}{\sigma(x)} & \text{if } \sigma(x) > 0 \\ 0 & \text{if } \sigma(x) = 0 \end{cases}$$

目前尚无针对采集函数选择的通用指导原则。此外，同一种采集函数未必在整个优化过程的各个阶段都是最佳选择。

混合策略（Brochu et al., 2010）：在每一次迭代中，都从一个“函数组合”里自适应地选取采集函数。该选择机制可以表述为一个在线多臂赌博机问题；经研究对比多种策略后发现，分层对冲方法表现最优。其核心算法被称为 GP-Hedge。

Stata mlhpbo

syntax：

```
mlhpbo varlist, [model(string) mltype(string) params(string) configs(string) kfold(integer 5) crit(string) method(string) seed(integer 123) iters(integer 100) deter
```

`model` 设定机器学习模型：

- 回归问题：
  - `lasso`，`ridge`，`enet`（`LinearRegression`）
  - `rf`（`RandomForest`）
  - `gbt`（`GradientBoostedTrees`），`nn`（`NearestNeighbors`），`ann`（`NeuralNetwork`），`Markov`，`NaiveBayes`，`svm`（`SupportVectorMachine`），`class`（`ClassDistribution`）
- 分类问题：`logit`（`LogisticRegression`），`dt`（`DecisionTree`），`rf`（`RandomForest`），`gbt`（`GradientBoostedTrees`），`nn`（`NearestNeighbors`），

不同的 `model` 具有自己的特定的参数: `params()`：

`lasso`, `ridge`, `enet`：

- `L1Regularization`,
- `L2Regularization`

`dt`：

- `DistributionSmoothing`: 1 (default)
- `FeatureFraction`: 1 (default)

`ann`：

- `NetworkDepth`: 5 (default)
- `MaxTrainingRounds`: 50 (default)

`svm`：

注：`mlhybo`的目标函数为

$$\operatorname{argmin} \frac{1}{2}\sum_{i=1}^{n}(y_i - f(\theta, x_i))^2 + \lambda_1 \sum_{k=1}^{K}|\theta_k| + \frac{\lambda_2}{2}\sum_{k=1}^{K}\theta_k^2$$

Stata中，`lasso`，`elasticnet`的目标函数为

$$\operatorname{argmin} \frac{1}{2N}\sum_{i=1}^{n}(y_i - f(\theta, x_i))^2 + \lambda\left(\sum_{k=1}^{K}\alpha|\theta_k| + \frac{1-\alpha}{2}\sum_{k=1}^{K}\theta_k^2\right)$$

二者具有关系：$\lambda_1 = N\lambda\alpha,\ \lambda_2 = N\lambda(1-\alpha)$.

`Fit[]`：$\frac{1}{2\sqrt{n}}RSS + \lambda_2\text{Penalty}$

`Predict[]`：$\frac{1}{2}RSS + \lambda_1\text{Penalty}$

`lasso`：$\frac{1}{2n}RSS + \lambda_1\text{Penalty}$

`mltype`：机器学习的类型： `predict` ， `classify`

不同模型的参数设置 `configs` 。比如，

- `model(rf) params(FeatureFraction) configs({0.4, 0.6, 0.8})`：参数为{0.4,0.6,0.8}的特定元素
- `model(lasso) params(L1Regularization) configs(Interval[{0,10}]`：参数落在[0,10]区间
- `model(enet) params(L1Regularization L2Regularization) configs(Rectangle[{0,10},{0,10}])`：参数落在{{-3,-3},{3,3}}直角区域

常用的参数空间设置：

设置	例子
`Interval[{min,max}]`	`Interval[{10, 20}]`
`Triangle[]`	`Triangle[{{0, 0}, {1, 1}, {2, 0}}]`
`Rectangle[{xmin,ymin},{xmax,ymax}]`	
`Disk[{x0,y0},radius]`	
`Disk[{x0,y0}, {xradius, yradius}]`	
`Triangle[p1,p2,p3]`	
`Polygon[p1,p2,...]`	

`crit`：模型评估指标。回归模型的常用指标：

`crit`	含义
"RSquared"	coefficient of determination
"MeanSquare"	mean square of the residuals
"StandardDeviation"	root mean square of the residuals
"MeanDeviation"	mean absolute value of the residuals
"LogLikelihood"	log-likelihood of the model given the test data
"StandardDeviationBaseline"	standard deviation of test set values
"MeanCrossEntropy"	mean cross entropy over test examples
"FractionVarianceUnexplained"	
"Perplexity"	exponential of the mean cross entropy

分类模型的常用指标：

`crit`	含义
"Accuracy"	fraction of correctly classified examples
"Accuracy"->n	top-n accuracy
"AccuracyBaseline"	accuracy if predicting the commonest class
"CohenKappa"	Cohen's kappa coefficient
"Error"	fraction of incorrectly classified examples
"GeometricMeanProbability"	geometric mean of the actual-class probabilities
"LogLikelihood"	log-likelihood of the model given the test set
"MeanCrossEntropy"	mean cross entropy over test examples
"MeanDecisionUtility"	mean utility over test example

`method` 包括:

- `mei` : maximize expected improvement (default)
- `mip` : maximize improvement probability

`seed()` : random seed

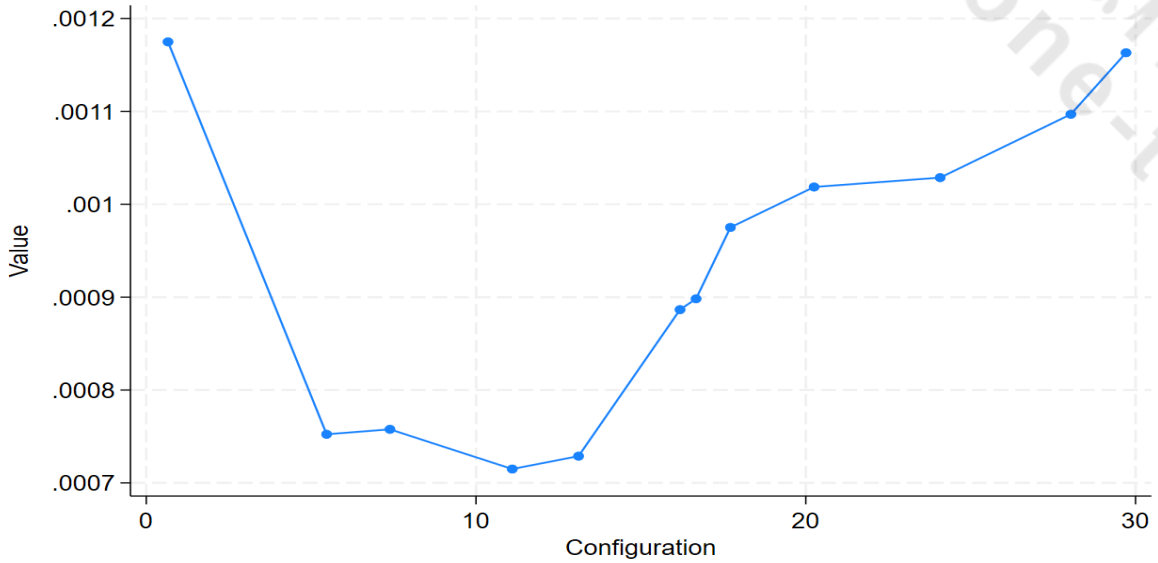
`mlhpbo` 的结果保存在 `boresults.xlsx` 文件，有两个工作表： `bohistory` 和 `coef` 。

例：Lasso

```
. insheet using "d:\course202601\growth.csv",clear
. mlhpbo gr6096 abslatit - ztropics, model(lasso) configs(Interval[{0.001, 30}]) crit(MeanSquare) seed(126) iters(200)

Configurations: 11.1002
Objective: 0.000715697
Results stored in boresults.xlsx with evaluation history and coefficients.

. import excel using "boresults.xlsx", firstrow sheet("bohistory") clear
. twoway (connected Value Configuration, sort(Configuration))
```



```
. import excel using "boresults.xlsx", firstrow sheet("coef") clear
(2 vars, 24 obs)

. list

     +-----------------------+
     |         x             b |
     |-----------------------|
  1. |     avelf   -.01804897 |
  2. |    buddha    .10509897 |
  3. |     civ72   -.00050562 |
  4. |    confuc    .15375537 |
  5. |   dens65c    .07877238 |
     |-----------------------|
  6. |      east    .17194348 |
  7. |  gdpch60l   -.05551927 |
  8. |   govnom1   -.00334344 |
  9. |     gvr61   -.05948969 |
 10. |   hindu00    .01542942 |
     |-----------------------|
```

弹性网（L1+L2）：

```
. insheet using "d:\course202601\growth.csv",clear
. mlhpbo gr6096 abslatit - ztropics, model(enet) configs(Rectangle[{0.001, 0.001},{10,10}]) crit(MeanSquare) seed(126) iters(200)

Configurations: {6.41272, 9.11644}
Objective: 0.000741037
Results stored in boresults.xlsx with evaluation history and coefficients.
```

随机森林（样本比例+树深）：

```
. mlhpbo gr6096 abslatit - ztropics, model(rf) configs({{0.3,0.5,0.7},{5,7,9}}) crit(MeanSquare)

Configurations: {0.7, 5}
Objective: 0.000778365
```

扩展：在多个机器学习方法中选择最优的。

Wolfram文件：mmprog.wls

```
{path, file} = Import["d:/mlci/mlboinfo.txt", "Lines"];
SetDirectory[path];
Needs["MLBO`"]
data = Import[file, {"Tabular"}];
If[FileExistsQ["mlboparams.do"], DeleteFile["mlboparams.do"], Nothing]
bo = mlBoHp[data, "enet", Rectangle[{0.001, 0.001},{10,10}], 5, "MeanSquare", AssumeDeterministic->True, RandomSeeding->126];
hp = Association[Thread[{"boconfig", "bobj"} -> bo[[1 ;; 2]]]];
str1 = StringRiffle[KeyValueMap["global " <> #1 <> "    " <> ToString[#2] &, hp], "\n"];
str2 = "global knum " <> ToString[bo[[4]]] <> "\n";
Export["mlboparams.do", StringRiffle[{str1, str2}, "\n"], "Text"];
If[FileExistsQ["boresults.xlsx"], DeleteFile["boresults.xlsx"], Nothing]
Export["boresults.xlsx", "Sheets" -> {"bohistory" -> bo[[3]],  "coef" -> bo[[5]]}, "Rules"];
```

Wolfram文件: "D:/MLCI/mm optimization.nb"

附录：wolframscript.exe安装方法

第1步：安装WolframEngine。如下两种方式均可：

- wolframengine.exe是Mathematica的一部分，可以直接安装Mathematica。
- 单独安装wolframengine.exe: https://www.wolfram.com/engine/ (根据提示注册账号，免费下载)

第2步：在Stata指定wolframscript.exe（不是mathematica.exe）的路径。指令为（实际目录因电脑而异）：

```
. whereis wolframscript "C:\Program Files\Wolfram Research\WolframScript\wolframscript.exe"
```

参考文献

Klein, A., Falkner, S., Bartels, S., Hennig, P. and Hutter, F., 2017.Fast Bayesian Optimization of Machine Learning Hyperparameters on Large Datasets. Proceedings of the 20th International Conference on Artificial Intelligence and Statistics, Vol 54, pp. 528--536. PMLR.

Joan Gonzalvez, Edmond Lezmi, Thierry Roncalli, Jiali Xu. Financial Applications of Gaussian Processes and Bayesian Optimization. https://arxiv.org/abs/1903.04841

Carl Edward Rasmussen and Christopher K. I. Williams. Gaussian Processes for Machine Learning. MIT Press, 2006. http://gaussianprocess.org/gpml/

谢 谢！