Stacking for Prediction

川田恵介

Table of contents

1 Stacking

- **異なるアルゴリズム**が生み出すモデルを集計する
  - **同じアルゴリズム (決定木)** よりも、より多様なモデルを集計できる

1.1 予測モデルの性能

- 一貫して優れた予測を生み出せる Algorithm は、現状、存在しない
- 母集団の構造に応じて、適したアルゴリズムは異なる
  - 母集団の構造は BlackBox なので、事前にはわからない
- 例: Liner VS Tree Model
  - X と Y の間に" なめらかな" 関係性があれば、Linear Model の方が性能が良い

1.2 実践

- どれを使えば良いか?

1. モデルを試作 & 検証し、最も性能が良いものを使う
   - アルゴリズム選択
2. モデルを集計する
   - アルゴリズム選択 や Random Forest の一般化

- ともに (Training) data をランダムに分割することで、実装できる

1.3 アルゴリズム選択

1. (Training) data を 2 分割し、Training/Validation data に分割
2. Training data のみを使用し、異なる Algorithm を用いて、予測モデル群を試作
3. Validation data で予測性能を評価し、最善の Algorithm を選ぶ
4. Training/Validation data を合体し、最善の Algorithm を用いて、最終予測モデルを推定

1.4 Example: Data

```
TempData
```

```
# A tibble: 6 x 4
  Price  Size Distance Group
  <dbl> <dbl>    <dbl> <dbl>
1    11    15        3     1
2    32    30        3     1
3    85    45        2     1
4   130    80        0     2
5    45    40        4     2
6    28    20        4     2
```

1.5 Example: モデルの試作

```
TempData$OLS = lm(
  Price ~ Size + Distance,TempData,
  subset = Group == 1) |>
  predict(TempData)

TempData$LASSO = hdm::rlasso(
  Price ~ Size + Distance,TempData,
  subset = Group == 1,
  post = FALSE) |>
  predict(TempData)

TempData
```

```
# A tibble: 6 x 6
  Price  Size Distance Group   OLS LASSO[,1]
  <dbl> <dbl>    <dbl> <dbl> <dbl>     <dbl>
1    11    15        3     1  11        19.8
2    32    30        3     1  32.0      41.4
3    85    45        2     1  85        68.3
4   130    80        0     2 198       129.
5    45    40        4     2  14.0      50.5
6    28    20        4     2 -14.0      21.7
```

1.6 Example: 中間評価

```
TempData |>
  mutate(
    Error_OLS = (OLS - Price)^2 |> round(),
    Error_LASSO = (LASSO - Price)^2 |> round()) |>
  mutate(
    MSE_OLS = Error_OLS |> mean() |> sqrt(),
    MSE_LASSO = Error_LASSO |> mean() |> sqrt(),
    .by = c(Group))
```

```
# A tibble: 6 x 10
  Price  Size Distance Group   OLS LASSO[,1] Error_OLS Error_LASSO[,1] MSE_OLS
  <dbl> <dbl>    <dbl> <dbl> <dbl>     <dbl>     <dbl>           <dbl>   <dbl>
1    11    15        3     1  11        19.8         0              78     0
2    32    30        3     1  32.0      41.4         0              89     0
3    85    45        2     1  85        68.3         0             279     0
4   130    80        0     2 198       129.       4624               1    49.5
5    45    40        4     2  14.0      50.5        961              30    49.5
6    28    20        4     2 -14.0      21.7       1764              40    49.5
# i 1 more variable: MSE_LASSO <dbl>
```

1.7 Example: 最終モデル

- LASSO が性能が良かったので

```
hdm::rlasso(
  Price ~ Size + Distance,
  TempData,
  post = FALSE
)
```

```
Call:
rlasso.formula(formula = Price ~ Size + Distance, data = TempData,
    post = FALSE)

Coefficients:
(Intercept)         Size     Distance
```

```
14.234        1.438       -5.321
```

1.8 モデル集計

- ここでは Linear Model として集計する
- $$g(X) = \beta_0 + \beta_{OLS} \times \underbrace{g_{OLS}(X)}_{OLS} + \beta_{RF} \times \underbrace{g_{RF}(X)}_{RandomForest} + \beta_{LASSO} \times \underbrace{g_{LASSO}(X)}_{LASSO}$$
- $\beta =$ 予測結果への Weight
  - 独立したデータへの適合度を最大にするように決定

1.9 モデル集計

1. (Training) data を 2 分割し、Training/Validation data に分ける
2. Training data のみを使用し、異なる Algorithm を用いて、予測モデル群を試作する
3. **Validation data で、Y と予測値を回帰し、β を決定**
4. **Training/Validation data を合体し、$g(X)$ を推定し、最終予測モデルを算出**

1.10 Example: Weight 推定

```
TempData
```

```
# A tibble: 6 x 6
  Price  Size Distance Group   OLS LASSO[,1]
  <dbl> <dbl>    <dbl> <dbl> <dbl>     <dbl>
1    11    15        3     1  11        19.8
2    32    30        3     1  32.0      41.4
3    85    45        2     1  85        68.3
4   130    80        0     2 198       129.
5    45    40        4     2  14.0      50.5
6    28    20        4     2 -14.0      21.7
```

```
Weight = hdm::rlasso(
  Price ~ OLS + LASSO,
  TempData,
```

```
  subset = Group == 2,
  post = FALSE)

Weight$coefficients
```

```
(Intercept)         OLS       LASSO
 2.50912027  0.05075589  0.90452827
```

1.11 Example: 最終モデル

```
OLS = lm(Price ~ Size + Distance, TempData)
LASSO = hdm::rlasso(Price ~ Size + Distance, TempData, post = FALSE)

Test = tibble(Size = 80,Distance = 15)

predict(OLS,Test)
```

```
       1
46.60214
```

```
predict(LASSO,Test)
```

```
         [,1]
[1,] 49.45247
```

```
Weight$coefficients[1] +
  Weight$coefficients[2]*predict(OLS,Test) +
  Weight$coefficients[3]*predict(LASSO,Test)
```

```
         [,1]
[1,] 49.60561
```

1.12 交差推定の活用

- 交差推定を活用し、Weight 推定の精度を高めることもできる

1.13 Example: Data

```
TempData
```

```
# A tibble: 6 x 4
```

```
  Price  Size Distance Group
  <dbl> <dbl>    <dbl> <dbl>
1    11    15        3     1
2    32    30        3     1
3    85    45        2     2
4   130    80        0     2
5    45    40        4     3
6    28    20        4     3
```

1.14 Example

```
Target = 1
PredOLS = lm(Price ~ Size + Distance, TempData,
         subset = Group != Target) |> predict(TempData)

PredLASSO = hdm::rlasso(Price ~ Size + Distance, TempData, post = FALSE,
                    subset = Group != Target) |> predict(TempData)

TempData = TempData |>
  mutate(
    OLS = case_when(Group == Target ~ PredOLS, .default = 0),
    LASSO = case_when(Group == Target ~ PredLASSO, .default = 0))

TempData
```

```
# A tibble: 6 x 6
  Price  Size Distance Group   OLS LASSO[,1]
  <dbl> <dbl>    <dbl> <dbl> <dbl>     <dbl>
1    11    15        3     1  45.7      19.8
2    32    30        3     1  54.4      41.4
3    85    45        2     2   0         0
4   130    80        0     2   0         0
5    45    40        4     3   0         0
6    28    20        4     3   0         0
```

1.15 Example

```
Target = 2
PredOLS = lm(Price ~ Size + Distance, TempData,
```

```
         subset = Group != Target) |> predict(TempData)

PredLASSO = hdm::rlasso(Price ~ Size + Distance, TempData, post = FALSE,
                    subset = Group != Target) |> predict(TempData)

TempData = TempData |>
  mutate(
    OLS = case_when(Group == Target ~ PredOLS, .default = OLS),
    LASSO = case_when(Group == Target ~ PredLASSO, .default = LASSO))

TempData
```

```
# A tibble: 6 x 6
  Price  Size Distance Group   OLS LASSO[,1]
  <dbl> <dbl>    <dbl> <dbl> <dbl>     <dbl>
1    11    15        3     1  45.7      19.8
2    32    30        3     1  54.4      41.4
3    85    45        2     2  37.9      68.3
4   130    80        0     2  60.3     129.
5    45    40        4     3   0         0
6    28    20        4     3   0         0
```

1.16 Example

```
Target = 3
PredOLS = lm(Price ~ Size + Distance, TempData,
         subset = Group != Target) |> predict(TempData)

PredLASSO = hdm::rlasso(Price ~ Size + Distance, TempData, post = FALSE,
                    subset = Group != Target) |> predict(TempData)

TempData = TempData |>
  mutate(
    OLS = case_when(Group == Target ~ PredOLS, .default = OLS),
    LASSO = case_when(Group == Target ~ PredLASSO, .default = LASSO))

TempData
```

```
# A tibble: 6 x 6
```

```
  Price  Size Distance Group   OLS LASSO[,1]
  <dbl> <dbl>    <dbl> <dbl> <dbl>     <dbl>
1    11    15        3     1  45.7      19.8
2    32    30        3     1  54.4      41.4
3    85    45        2     2  37.9      68.3
4   130    80        0     2  60.3     129.
5    45    40        4     3  59.1      50.5
6    28    20        4     3  21.8      21.7
```

1.17 Example

```
Weight = lm(
  Price ~ OLS + LASSO,
  TempData)

Weight$coefficients
```

```
(Intercept)         OLS       LASSO
 20.4418801  -0.6694697   1.1943079
```

1.18 Example: 最終モデル

```
OLS = lm(Price ~ Size + Distance, TempData)
LASSO = hdm::rlasso(Price ~ Size + Distance, TempData, post = FALSE)

Test = tibble(Size = 80,Distance = 15)

predict(OLS,Test)
```

```
       1
46.60214
```

```
predict(LASSO,Test)
```

```
         [,1]
[1,] 49.45247
```

```
Weight$coefficients[1] +
  Weight$coefficients[2]*predict(OLS,Test) +
  Weight$coefficients[3]*predict(LASSO,Test)
```

```
          [,1]
[1,] 48.30463
```

1.19 まとめ

- Stacking により予測性能が劇的に改善することは少ない
  - ”良い予測を生み出す Algorithm を見逃す” リスクを減らせる
  - 一致性およびある程度の収束性能を確保することが重要な比較研究の応用においても重要
- 実証家向けのガイド: Phillips et al. (2023)
  - 理論的性質 (Chen, Klusowski, and Tan 2023 とその引用)
    - Stacking に使った最善の Algorithm よりも、(漸近的に) 予測性能が高い
- Chap 9 in CausalML 参照

2 実例: 価格予測

2.1 Algorithm

- データ: Data.csv の 2022 年第三四半期 (2877 事例)
  - Y = 取引価格
  - X = 広さ、築年数、駅からの距離、容積率、改築の有無、立地 (23 区)、最寄駅
- Algorithm
  - OLS: Linear Model = 47 変数
  - LASSO: Interaction + 2nd order polynomial = 747 変数
  - Random Forest
  - Stacking

2.2 Peformance

```
# A tibble: 4 x 2
     R2 Algorithm
  <dbl> <chr>
1 0.752 OLS
2 0.868 LASSO
```

```
3 0.827 RF
4 0.879 Stacking
```

2.3 Weight in Stacking

```
# A tibble: 4 x 2
  Term           Weight
  <chr>           <dbl>
1 (Intercept)    -2.97
2 OLS.response    0.01
3 RF.response     0.301
4 LASSO.response  0.761
```

Reference

Chen, Xin, Jason M Klusowski, and Yan Shuo Tan. 2023. "Error Reduction from Stacked Regressions." *arXiv Preprint arXiv:2309.09880*.

Phillips, Rachael V, Mark J Van Der Laan, Hana Lee, and Susan Gruber. 2023. "Practical Considerations for Specifying a Super Learner." *International Journal of Epidemiology* 52 (4): 1276–85.