Variable Selection for Big Data

Tadao Hoshino （星野匡郎）

Econometrics II: ver. 2019 Spring Semester

Big Data

Big Data

What is Big Data? How big is big?

- There is no precise definition. But it is often characterized by the three **V**'s:

  Volume: The amount of data is extremely large.

  Variety: A large variety of data types is available.

  Velocity: The data is generated and updated automatically and almost instantaneously.

- As we are focusing on "static" analytical methods (rather than "dynamic" ones), only the first two **V**'s, Volume and Variety, matter.
- In the language of statistics, the availability of huge data volume is equivalent to that the sample size is extremely large.
- Similarly, the availability of huge data variety implies the existence of so many variables. *High-Dimensional Data*.
- More specifically, for a dataset $\{(Y_i, X_{1i}, \ldots, X_{pi}) : 1 \leq i \leq n\}$, where Y is a response variable, and $(X_1, \ldots, X_p)$ is a set of input variables,
  - **Volume problem** $\iff$ n is so large.
  - **Variety problem** $\iff$ p is so large.

Big Data

- Recall:
  - The law of large numbers:

$$\frac{1}{n}\sum_{i=1}^{n} X_i \xrightarrow{p} E[X] \; \underline{\text{as } n \text{ increases to infinity}}$$

  - The central limit theorem: the probability distribution of $\frac{1}{n}\sum_{i=1}^{n} X_i$ can be approximated by a normal distribution <u>as n increases to infinity</u>.
- Thus, the Volume problem is not necessarily a problem; rather, it is theoretically desirable. (quite computationally heavy, though)
- Theoretically speaking, the Variety problem is a more serious issue than the Volume problem.

Big Data

- When the number of input variables p is extremely huge, there occur several "theoretical" problems:
  - Overfitting problem: a particular dataset is fit too closely (or exactly) by a statistical model / ML algorithm, resulting in poor prediction accuracy on other datasets:

    $$\text{Too small training error} \Rightarrow \text{Large test error}$$

  - Non-identifiability of model parameters: when p is larger than n, standard regression analysis is infeasible.[1]

[1] When the number of unknowns is larger than the number of observations, then the system will have infinitely many solutions.

Overfitting Problem

Overfitting Problem

- Consider a multiple regression model:

$$
\begin{aligned}
Y &= \beta_0 + X_1\beta_1 + \cdots + X_p\beta_p + error \\
&= \mathbf{X}_1^\top \beta_1 + \mathbf{X}_2^\top \beta_2 + error,
\end{aligned}
$$

where

$$
\begin{aligned}
\mathbf{X}_1 &= (1, X_1, ..., X_k)^\top, \;\; \beta_1 = (\beta_0, ..., \beta_k)^\top, \\
\mathbf{X}_2 &= (X_{k+1}, ..., X_p)^\top, \;\; \beta_2 = (\beta_{k+1}, ..., \beta_p)^\top.
\end{aligned}
$$

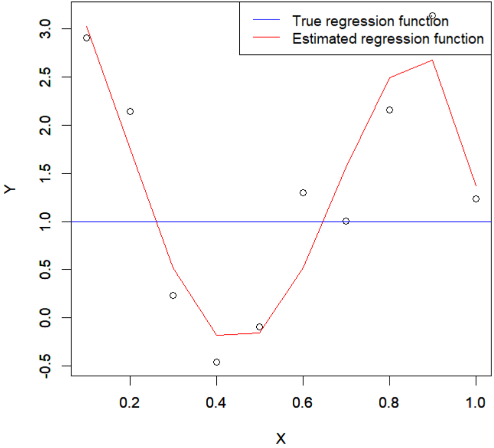
- Suppose that $\beta_2 = \mathbf{0}$, i.e., $\mathbf{X}_2 = (X_{k+1}, ..., X_p)^\top$ are redundant variables.
- Note that, even in this case, including $\mathbf{X}_2$ in the regression model can improve the Goodness-of-Fit (e.g., R-squared value).

A numerical simulation ($n = 10$, $p = 6$):

```
Y   <- 1 + rnorm(10)
X   <- (1:10)/10
XX  <- cbind(X, X^2, X^3, X^4, X^5)
reg <- lm(Y ~ XX)
```

- In this DGP, the true regression function is a constant function: $Y = g(\mathbf{X}) + error$, where $g(\mathbf{X}) = 1$.
- Thus, the regressors (X, X^2, X^3, X^4, X^5) should contain no information at all that can be used to predict the value of Y.

Overfitting Problem



Overfitting Problem

- Large gap between the true model and the fitted model;
  $\Rightarrow$ the estimated model adapts to the training data too well, resulting in poor performance in out-of-sample predictions.
- This problem is called overfitting.
- The overfitting problem easily occurs when the sample size is relatively small to the number of estimation parameters.

- In an extreme case, when $n = p$, it is possible to achieve an R-squared of 1; zero in-sample error but huge out-of-sample error.

```
Y  <- 1 + rnorm(5)
X1 <- rnorm(5)
X2 <- rnorm(5)
X3 <- rnorm(5)
X4 <- rnorm(5)

reg <- lm(Y ~ X1 + X2 + X3 + X4)
summary(reg)
```

- The regressors $(X_1, \ldots, X_4)$ are totally independent of Y.

Overfitting Problem

```
> reg <- lm(Y ~ X1 + X2 + X3 + X4)
> summary(reg)

Call:
lm(formula = Y ~ X1 + X2 + X3 + X4)

Residuals:
ALL 5 residuals are 0: no residual degrees of freedom!

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  -0.9420         NA      NA       NA
X1           -3.4975         NA      NA       NA
X2           -0.2114         NA      NA       NA
X3            2.7895         NA      NA       NA
X4            1.8960         NA      NA       NA

Residual standard error: NaN on 0 degrees of freedom
Multiple R-squared:      1,     Adjusted R-squared:    NaN
F-statistic:   NaN on 4 and 0 DF,  p-value: NA
```

Overfitting Problem

- In order to mitigate the overfitting problem, we need to remove less important variables from the model.
- Which variables are (not) important?
  $\Rightarrow$ We need some criterion.

Information Criterion

Information Criterion

- The information criterion consists of two parts:

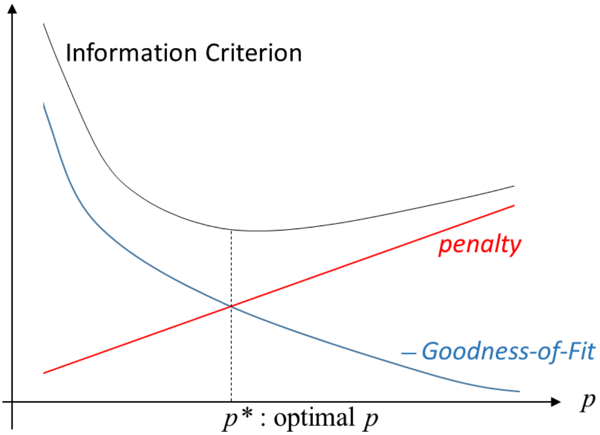
$$\text{Information Criterion}(p) = -\textit{Goodness-of-Fit}(p) + \textit{penalty}(p)$$

  where p is the number of estimated parameters (# input variables).
- Both *Goodness-of-Fit*(p) (such as the values of R-squared and log-likelihood) and *penalty*(p) are monotonically increasing in p.
- The penalty term often takes the form of

$$\textit{penalty}(p) = kp$$

  for some positive number $k > 0$.
- There is a trade-off between the first term and the second term.
- The model that has the smallest information criterion value is "preferred".

Information Criterion



Information Criterion

- When maximum likelihood is used for fitting the model, we can use the Akaike information criterion (AIC) or the Bayes information criterion (BIC) for variable selection:

$$\text{AIC} = -2\widehat{\ell}_n + 2p$$
$$\text{BIC} = -2\widehat{\ell}_n + \ln(n)p,$$

  where $\widehat{\ell}_n$ is the log-likelihood value, and n is the sample size.
- The BIC criterion chooses more parsimonious models compared to the AIC criterion.

- For a dataset with a response variable Y and p explanatory variables $(X_1, \ldots, X_p)$, there are in total 2^p possible distinct regression models:

$$\text{(Full model)} \quad Y = \beta_0 + X_1\beta_1 + \cdots + X_p\beta_p + error$$
$$\text{(Minimum model)} \quad Y = \beta_0 + error$$

- In principle, we need to calculate the AIC or BIC for all possible models, and choose the one that achieves the minimum among them.
- However, when p is large, this approach is extremely computationally heavy and often infeasible.
- Fortunately, **R** has a built-in algorithm called `step()` that can find a "pseudo" optimal model computationally quickly.

Information Criterion

- Practice datasets: training data **train.csv**, test data **test.csv**.
  - This is an artificial dataset I created.
  - The true model is

    $$Y = \beta_0 + X_1\beta_1 + \cdots + X_{30}\beta_{30} + error,$$

    where I set the coefficients

    $$(\beta_0, \beta_2, \beta_4, \beta_8, \beta_{14}, \beta_{15}, \beta_{17}, \beta_{22}, \beta_{25}, \beta_{28}) \neq 0,$$

    and the other coefficients are set to almost zero.
- The data csv files are available from my website or from **Course Navi**.
- Set your working directory appropriately and import the csv files.

```
> train <- read.csv("train.csv")
> test  <- read.csv("test.csv")
>
> head(train)
          Y         X1         X2         X3         X4
1 -4.895293  0.4420104 -0.1069674 -1.9800604  1.3536843
2 -2.338560 -0.5427156  1.4861381  0.3461045  0.5049104
3 -1.863893  0.6747334 -0.6382183  0.8383224  0.4203452
4  1.886575  0.1042394 -0.6573750 -0.2384937 -0.5089250
5 -4.207808 -1.1303783  1.9822079  0.5271462  0.6019664
6 -1.051983 -0.1859916 -0.7183498 -0.4481307  1.2072333

         X22        X23        X24         X25        X2
1 -1.1865095 -0.3790765 -0.4907028 -0.73106378  0.680168
2 -1.1753293 -1.6785635  0.2147433  1.05391159 -0.444677
3 -0.4378268 -1.4468853 -0.5527107 -0.04828141  0.589865
4 -0.6410097 -0.2018785  0.1514414  0.68378757  0.405962
5  0.4902416 -0.2931813 -0.7991648 -1.40568472 -0.518499
6 -0.2492836 -0.4800650 -0.7864809 -0.61576504  0.710233
> dim(train)
[1] 300  31
> dim(test)
[1] 500  31
```

Information Criterion

- Before running the variable selection procedure, we estimate the "full" model, where all 30 input variables are in the model:

```
full <- lm(Y ~., data = train)
summary(full)
```

  To perform a regression using all of the variables in the dataset as predictors, we can use the shorthand: "`Y ~.`".

Information Criterion

```
>  full <- lm(Y ~., data = train)
>  summary(full)
Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept)  1.031427   0.092902  11.102   <2e-16 ***
X1          -0.012504   0.093734  -0.133   0.8940
X2          -1.184742   0.090041 -13.158   <2e-16 ***
X3          -0.161716   0.090725  -1.783   0.0758 .
X4          -3.780112   0.093949 -40.236   <2e-16 ***
X5           0.089574   0.092914   0.964   0.3359
X6           0.013031   0.096538   0.135   0.8927
X7          -0.073502   0.092867  -0.791   0.4294
X8           1.243137   0.089188  13.938   <2e-16 ***
X9           0.143851   0.097073   1.482   0.1395
X10          0.026790   0.090603   0.296   0.7677
X11         -0.089505   0.098487  -0.909   0.3643
X12         -0.036002   0.092849  -0.388   0.6985
X13          0.004741   0.098388   0.048   0.9616
X14          1.422275   0.091887  15.478   <2e-16 ***
X15          0.915547   0.086498  10.585   <2e-16 ***
X16         -0.075762   0.093888  -0.807   0.4204
X17          2.518779   0.096865  26.003   <2e-16 ***
X18         -0.058914   0.092639  -0.636   0.5253
X19         -0.038191   0.094228  -0.405   0.6856
X20         -0.096178   0.091410  -1.052   0.2937
X21          0.005685   0.085687   0.066   0.9471
X22         -1.501290   0.090940 -16.509   <2e-16 ***
X23         -0.009095   0.096360  -0.094   0.9249
X24          0.076590   0.097485   0.786   0.4328
X25         -1.457047   0.092606 -15.734   <2e-16 ***
X26         -0.014211   0.094471  -0.150   0.8805
X27          0.129472   0.088360   1.465   0.1440
X28         -1.243803   0.092998 -13.374   <2e-16 ***
X29          0.069613   0.094107   0.740   0.4601
X30         -0.061407   0.093674  -0.656   0.5127
```

- Variable selection based on AIC:

```
AIC <- step(full, k = 2)
summary(AIC)
```

- Variable selection based on BIC:

```
BIC <- step(full, k = log(nrow(train)))
summary(BIC)
```

(`nrow(train)` = the size of the the training data.)

Information Criterion

```
>  summary(AIC)

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  1.01318    0.08772  11.550   <2e-16 ***
X2          -1.18985    0.08613 -13.814   <2e-16 ***
X3          -0.12256    0.08368  -1.465    0.144
X4          -3.78322    0.08845 -42.773   <2e-16 ***
X8           1.25139    0.08511  14.704   <2e-16 ***
X9           0.14514    0.09307   1.560    0.120
X14          1.44349    0.08766  16.467   <2e-16 ***
X15          0.91735    0.08287  11.069   <2e-16 ***
X17          2.52874    0.08854  28.560   <2e-16 ***
X22         -1.53298    0.08581 -17.866   <2e-16 ***
X25         -1.45089    0.08762 -16.558   <2e-16 ***
X27          0.12785    0.08402   1.522    0.129
X28         -1.23511    0.08771 -14.083   <2e-16 ***
```

Recall: $(\beta_0, \beta_2, \beta_4, \beta_8, \beta_{14}, \beta_{15}, \beta_{17}, \beta_{22}, \beta_{25}, \beta_{28}) \neq 0$

```
>  summary(BIC)

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  1.02198    0.08819   11.59   <2e-16 ***
X2          -1.21413    0.08615  -14.09   <2e-16 ***
X4          -3.78896    0.08817  -42.98   <2e-16 ***
X8           1.24583    0.08566   14.54   <2e-16 ***
X14          1.43768    0.08810   16.32   <2e-16 ***
X15          0.91253    0.08328   10.96   <2e-16 ***
X17          2.52301    0.08907   28.32   <2e-16 ***
X22         -1.54740    0.08623  -17.95   <2e-16 ***
X25         -1.45312    0.08815  -16.48   <2e-16 ***
X28         -1.23478    0.08761  -14.10   <2e-16 ***
```

Recall: $(\beta_0, \beta_2, \beta_4, \beta_8, \beta_{14}, \beta_{15}, \beta_{17}, \beta_{22}, \beta_{25}, \beta_{28}) \neq 0$

- For each model, the predicted values for `Y`'s in the test dataset can be easily obtained by

```
pred.f <- predict(full, newdata = test)
pred.A <- predict(AIC, newdata = test)
pred.B <- predict(BIC, newdata = test)
```

- We evaluate the prediction accuracy of the models by the mean absolute error (MAE):

```
MAE.f <- mean(abs(test$Y - pred.f))
MAE.A <- mean(abs(test$Y - pred.A))
MAE.B <- mean(abs(test$Y - pred.B))
```

Information Criterion

```
> MAE.f
[1] 1.258406
> MAE.A
[1] 1.235173
> MAE.B
[1] 1.212522
```

- Thus, the BIC-based model achieves the best prediction accuracy, while the full model has the lowest accuracy.
- The `step()` function can be applied to `glm` objects as well. The same variable selection procedure can be used for logistic regression.

LASSO

LASSO

- A drawback of the above mentioned variable selection method is that, when the number of input variables p is larger than the sample size n, the full model is not estimable.
- A popular variable selection method that works even for such high-dimensional data is the LASSO (least absolute shrinkage and selection operator).
- Recall that, for a given dataset $\{(Y_i, X_{i1}, \ldots X_{ip}) : 1 \leq i \leq n\}$, the OLS estimator of $\beta = (\beta_0, \beta_1, \ldots, \beta_p)^\top$ is defined as

$$\widehat{\beta}_n^{ols} = \underset{\beta}{\operatorname{argmin}} \sum_{i=1}^{n} \left(Y_i - \beta_0 - \sum_{j=1}^{p} X_{ij}\beta_j \right)^2$$

- However, if $p > n$, $\widehat{\beta}_n^{ols}$ is not available.
- The LASSO estimator is defined by modifying the above as follows

$$\widehat{\beta}_n^{lasso} = \underset{\beta}{\operatorname{argmin}} \left\{ \sum_{i=1}^{n} \left(Y_i - \beta_0 - \sum_{j=1}^{p} X_{ij}\beta_j \right)^2 + \underbrace{\lambda \sum_{j=1}^{p} |\beta_j|}_{\text{penalty term}} \right\},$$

  where $\lambda > 0$ is a pre-specified constant term.[2]
- The LASSO estimator has a nice property that, for a large λ, it restricts the coefficients of less important variables to exactly zero.
  $\Rightarrow$ The LASSO can be used for variable selection.

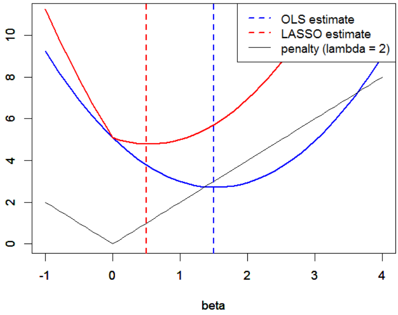
[2] How to choose λ is a crucial problem. Many methodologies have been proposed in the literature, but the details are omitted here.

- A simple linear regression model without intercept:

$$Y = 1.6X + \varepsilon,$$

where $X \sim N(0,1)$ and $\varepsilon \sim N(0, 1.2^2)$. A dataset of size 500 is generated.

- Blue curve = OLS obj func; red curve = LASSO obj func with $\lambda = 2$.



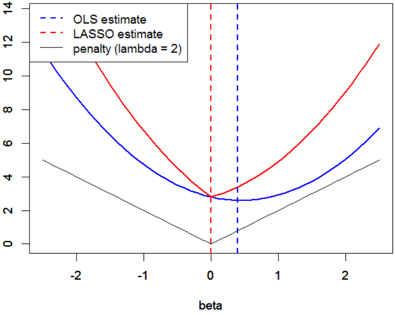
LASSO

- A simple linear regression model without intercept:

$$Y = 0.5X + \varepsilon,$$

where $X \sim N(0,1)$ and $\varepsilon \sim N(0, 1.2^2)$. A dataset of size 500 is generated.

- Blue curve = OLS obj func; red curve = LASSO obj func with $\lambda = 2$.



- To implement LASSO in **R**, we can use the **glmnet** package.
- Install and load the package by

```
install.packages("glmnet")
library(glmnet)
```

and import the same simulated data used in the previous exercise:

```
train <- read.csv("train.csv")
```

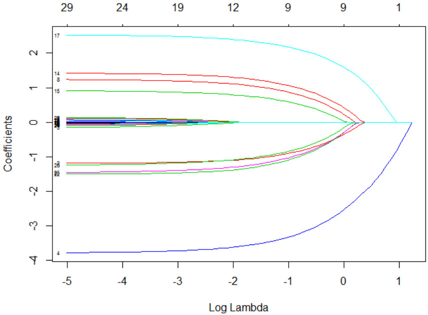
- Run the following code:

```
Y <- as.matrix(train[,1])
X <- as.matrix(train[,-1])
```

LASSO

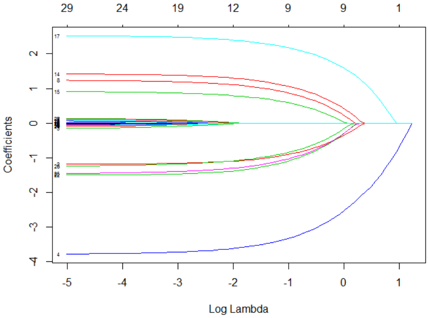
- We first check which coefficients have non-zero values for each different value of λ.

```
fit <- glmnet(X, Y)
plot(fit, xvar = "lambda", label = TRUE)
```



LASSO

- This figures shows that more than half of the variables drop out of the model when $\log \lambda$ reaches to -2. $=$ less important variables
- On the other hand, the rest of the variables remain in the model until $\log \lambda$ becomes larger than 0. $=$ important variables
- Thus, the optimal λ may lie between $\exp(-2)$ and $\exp(0)$.



LASSO

Variable selection by LASSO:

```
LASSO <- glmnet(X, Y, lambda = exp(-1))
LASSO$beta
```

```
> LASSO <- glmnet(X, Y, lambda = exp(-1))
> LASSO$beta
30 x 1 sparse Matrix of class "dgCMatrix"
            s0
X1    .                X16   .
X2   -0.8985855        X17   2.1787197
X3    .                X18   .
X4   -3.3230457        X19   .
X5    .                X20   .
X6    .                X21   .
X7    .                X22  -1.0899186
X8    0.8711417        X23   .
X9    .                X24   .
X10   .                X25  -1.0277567
X11   .                X26   .
X12   .                X27   .
X13   .                X28  -0.8406376
X14   1.0623966        X29   .
X15   0.5878282        X30   .
```

The LASSO obtained the same variable selection result as the BIC method.

LASSO

- It is known that if the LASSO variable selection correctly includes all components of the "true" regression model, applying the OLS to the selected model can yield better estimates, so-called the post-LASSO estimator.
- Post-LASSO:

```
Xused <- X[,which(LASSO$beta != 0)]
reg <- lm(Y ~ Xused)
summary(reg)
```

```
> reg <- lm(Y ~ Xused)
> summary(reg)

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)  1.02198    0.08819   11.59   <2e-16 ***
XusedX2     -1.21413    0.08615  -14.09   <2e-16 ***
XusedX4     -3.78896    0.08817  -42.98   <2e-16 ***
XusedX8      1.24583    0.08566   14.54   <2e-16 ***
XusedX14     1.43768    0.08810   16.32   <2e-16 ***
XusedX15     0.91253    0.08328   10.96   <2e-16 ***
XusedX17     2.52301    0.08907   28.32   <2e-16 ***
XusedX22    -1.54740    0.08623  -17.95   <2e-16 ***
XusedX25    -1.45312    0.08815  -16.48   <2e-16 ***
XusedX28    -1.23478    0.08761  -14.10   <2e-16 ***
```

- The regression result is the same as the one in p.26, of course.
- The `glmnet()` function can be applied to logistic regression as well, with a slight modification.