Variable Selection
and others

Hao Zheng
November 6, 2009
haozheng@stat.wisc.edu
Outline

- What’s the role of intercept?
- Bias-Variance Tradeoff
- Which model is better?
- Simulation about under-fitting and over-fitting
- Cross Validation
What’s the role of intercept?

How to prove \( \frac{1}{n} \sum_{i=1}^{n} \hat{\varepsilon}_i = 0 \) ?

\[
\frac{1}{n} \sum_{i=1}^{n} \hat{\varepsilon}_i = 0 \iff 1^T (Y - \hat{Y}) = 0 \\
\iff X^T (Y - \hat{Y}) = 0 \\
\iff X^T Y = X^T \hat{Y} \\
\iff X^T Y = X^T X (X^T X)^{-1} X^T Y = X^T \hat{Y}
\]
\[ \frac{1}{n} \sum_{i=1}^{n} \hat{e}_i = 0 \iff \frac{1}{n} \sum_{i=1}^{n} (y_i - \hat{y}_i) = 0 \]

\[ \iff \frac{1}{n} \sum_{i=1}^{n} y_i = \frac{1}{n} \sum_{i=1}^{n} \hat{y}_i = \frac{1}{n} \sum_{i=1}^{n} (\hat{\beta}_0 + \hat{\beta}_1 x_{i1} + \ldots + \hat{\beta}_p x_{ip}) \]

\[ \iff \bar{y} = \hat{\beta}_0 + \hat{\beta}_1 \bar{x}_1 + \ldots + \hat{\beta}_p \bar{x}_p \]

\[ \iff \hat{\beta}_0 = \bar{y} - \hat{\beta}_1 \bar{x}_1 - \ldots - \hat{\beta}_p \bar{x}_p \]
Bias-Variance Tradeoff

\[ \text{MSE} = E(\hat{\beta} - \beta)^2 \]
\[ E(\hat{\beta} - \beta)^2 = E(\hat{\beta} - E\hat{\beta} + E\hat{\beta} - \beta)^2 \]
\[ = E\left( (\hat{\beta} - E\hat{\beta})^2 + 2(\hat{\beta} - E\hat{\beta})(E\hat{\beta} - \beta) + (E\hat{\beta} - \beta)^2 \right) \]
\[ = E(\hat{\beta} - E\hat{\beta})^2 + 2E\left( (\hat{\beta} - E\hat{\beta}) (E\hat{\beta} - \beta) + (E\hat{\beta} - \beta)^2 \right) \]
\[ = E(\hat{\beta} - E\hat{\beta})^2 + (E\hat{\beta} - \beta)^2 \]
\[ = \text{Variance} + \text{Bias}^2 \]
Bias-Variance Tradeoff

\[ Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \varepsilon \]

\[ \varepsilon \text{ follows } \mathcal{N}_n(0, \sigma^2 I_n) \]

We know LSE is BLUE, which means it is the best among unbiased estimates, but

\[ \text{var}(\hat{\beta}_1) = \frac{\sigma^2}{\sum_{i=1}^{n} (x_{i1} - \bar{x}_1)^2} \times \frac{1}{1 - r_{12}^2} \]
More generally,

\[ \text{var}(\hat{\beta}) = \sigma^2 (X^T X)^{-1} \]

If \((X^T X)\) is almost singular, the estimate would have large variance, not very stable.

Sometimes, we can consider some biased estimate if its variance is small while biasness is acceptable.
Bias-Variance Tradeoff

Under what kind of conditions that \( (X^TX) \) will be (almost) singular?

- n<p
  - Choose only some of variables to build model.
- Some variables are highly correlated
  - Choose not so close variables to build model.

Choose variable? Variable Selection!
Reasons of doing variable selection

- Too many variables!
- To deal with collinearity (predictors are highly correlated)
- Reasonably smaller size of predictors can simplify result.
- ...
Which model is better?

- This example is given by Mantel, 1970.

```R
> dat
   Y X1 X2 X3
[1,] 5  1 1004  6.0
[2,] 6  200 806  7.3
[3,] 8 -50 1058 11.0
[4,] 9 909 100 13.0
[5,] 11 506 505 13.1

> cor(dat)
     Y   X1    X2    X3
Y 1.000000 0.6145611 -0.6108095 0.9525563
X1 0.6145611 1.0000000 -0.9999887 0.6858141
X2 -0.6108095 -0.9999887 1.0000000 -0.6826107
X3 0.9525563 0.6858141 -0.6826107 1.0000000
```
Which model is better?

- Not surprisingly, different approaches lead to different results.
- How to measure??
Which model is better? - R command

- `#library(leaps)`
  - `> leaps(X, Y)`
- `# library(wle)`
  - `> mle.stepwise(Y~X, type="")`
  - `> step(lm(Y~X), direction="")`
Which model is better? - Example

- `data(mtcars)`
- `attach(mtcars)`
- `leaps(cbind(cyl, hp, wt, am), mpg)`
- `mle.stepwise(mpg~cyl+hp+wt+am)`
- `step(lm(mpg~cyl+hp+wt+am))`
Simulation about under-fitting and over-fitting

> S = 100  # S is the number of simulations
> n = 50   # sample size
>
> beta0 = 2
> beta1 = 1.2
> beta2 = -1.4
> beta3 = -0.5
>
> under = matrix(0, 3, 0)
> over = matrix(0, 6, 0)
> sigma2_under = rep(0, S)
> sigma2_over = rep(0, S)
>
> for (s in 1:S) {
+  X1 = rnorm(n, 1, 1)
+  X2 = rnorm(n, 2, 2)
+  X3 = rnorm(n, 3, 3)
+  X4 = rnorm(n, 2, 2)
+  X5 = rnorm(n, 3, 3)
+  E = rnorm(n, 0, 1)
+  Y = beta0 + beta1*X1 + beta2*X2 + beta3*X3 + E
+  lm_under = lm(Y~X1+X2)
+  beta_under = as.matrix(lm_under$coefficients)
+  under = cbind(under, beta_under)
+  sigma2_under[s] = sum((lm_under$residuals)^2)/(n-3)
+  lm_over = lm(Y~X1+X2+X3+X4+X5)
+  beta_over = as.matrix(lm_over$coefficients)
+  over = cbind(over, beta_over)
+  sigma2_over[s] = sum((lm_over$residuals)^2)/(n-5)
+ }
>
> Beta_under = matrix(rep(c(beta0, beta1, beta2, beta3), S), 4, S, byrow=F)
> Under = rbind(under, rep(0, S))
> Bias_under = apply(Under-Beta_under, 1, mean)
> Error_under = apply((Under-Beta_under)^2, 1, mean)
>
> Over = over[1:4,]
> Beta_over = matrix(rep(c(beta0, beta1, beta2, beta3), S), 4, S, byrow=F)
> Bias_over = apply(Over-Beta_over, 1, mean)
> Error_over = apply((Over-Beta_over)^2, 1, mean)
Cross Validation

- Cross validation mainly used in setting where the goal is prediction.
- Cross validation involves partitioning a sample into training set and validation (test) set.
2-fold cross-validation for *mtcars* data set

```r
> rm(list=ls())
> data(mtcars)
> attach(mtcars)
> MPG = mpg
> fuel = data.frame(mpg, cyl, hp, wt, am)
> detach()
> index = sample(32,16)
> set1 = fuel[index,]
> set2 = fuel[-index,]
>
> fit_full_1 = lm(mpg~cyl+hp+wt+am, data=set1)
> fit_am_1 = lm(mpg~cyl+hp+wt, data=set1)
> full_mpg_1 = predict(fit_full_1, set2)
> res_full_1 = sum((MPG[-index]-full_mpg_1)^2)
> am_mpg_1 = predict(fit_am_1, set2)
> res_am_1 = sum((MPG[-index]-am_mpg_1)^2)
>
> fit_full_2 = lm(mpg~cyl+hp+wt+am, data=set2)
> fit_am_2 = lm(mpg~cyl+hp+wt, data=set2)
> full_mpg_2 = predict(fit_full_2, set1)
> res_full_2 = sum((MPG[index]-full_mpg_2)^2)
> am_mpg_2 = predict(fit_am_2, set1)
> res_am_2 = sum((MPG[index]-am_mpg_2)^2)
>
> am = sum(res_am_1, res_am_2)
> full = sum(res_full_1, res_full_2)
> print(am)
```
Thank you!

Have a nice weekend!