

Linear Model in Matrix Form

Lin Wang

Linear model

- Suppose we want to model the response Y in terms of three predictors, X_1, X_2 , and X_3 . One general form for the model would be

$$Y = f(X_1, X_2, X_3) + \varepsilon$$

- f is some unknown function and ε is the random error
- Even with just three predictors, we typically will not have enough data to try to estimate arbitrary f directly
- We usually have to assume that it has some more restricted form, for example, linear restrictions:

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \beta_3 X_3 + \varepsilon$$

- $\beta_i, i = 0, 1, 2, 3$ are unknown parameters to be estimated from observed data

Linear model

- In a linear model the parameters enter linearly — the predictors themselves do not have to be linear

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 \log X_2 + \beta_3 X_1 X_2 + \varepsilon$$



$$Y = \beta_0 + \beta_1 X_1^{\beta_2} + \varepsilon$$



Matrix representation of linear model

- Matrix representation is a powerful tool to represent the linear model when we have large and **high-dimensional (i.e., many predictors)** datasets
- Suppose we have a response Y and three predictors X_1, X_2 , and X_3
- The data can be presented in the matrix form

$$\begin{array}{cccc} y_1 & x_{11} & x_{12} & x_{13} \\ y_2 & x_{21} & x_{22} & x_{23} \\ \dots & & \dots & \\ y_n & x_{n1} & x_{n2} & x_{n3} \end{array}$$

- n is the number of observations
- p is the number of parameters (or dimension) and $p = 4$ in this example

Matrix representation of linear model

- We can plug in the n observations into the linear model

$$y_i = \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \beta_3 x_{i3} + \varepsilon_i \quad i = 1, \dots, n.$$

- We can furth write it in matrix form

$$y = X\beta + \varepsilon$$

- where $y = (y_1, \dots, y_n)^T$, $\varepsilon = (\varepsilon_1, \dots, \varepsilon_n)^T$, $\beta = (\beta_0, \dots, \beta_3)^T$ and

$$X = \begin{pmatrix} 1 & x_{11} & x_{12} & x_{13} \\ 1 & x_{21} & x_{22} & x_{23} \\ \dots & & \dots & \\ 1 & x_{n1} & x_{n2} & x_{n3} \end{pmatrix}$$

Least squares estimation

- The regression model $y = X\beta + \varepsilon$ partitions the response into a systematic component $X\beta$ and a random component ε
- We would like to choose β so that the systematic part **explains the response** as much as possible
- In other words, we want to minimize the error ε
- The best estimate of β should **minimize the sum of the squared errors**

$$\sum \varepsilon_i^2 = \varepsilon^T \varepsilon = (y - X\beta)^T (y - X\beta)$$

- The sum of the squared errors is a function of β
- To solve this optimization problem, we differentiate with respect to β and set to zero

$$\frac{\partial \varepsilon^T \varepsilon}{\partial \beta} = \frac{\partial (y - X\beta)^T (y - X\beta)}{\partial \beta} = 0$$

Least squares estimation

- The rule of matrix derivative
 - Similar to scalar case
 - $X\beta \approx ax$
 - $X^T X \approx x^2$
 - But need to match the dimension
- $\frac{\partial \varepsilon^T \varepsilon}{\partial \beta}$ is $p \times 1$
- $\frac{\partial (y - X\beta)^T (y - X\beta)}{\partial \beta} = -2(y - X\beta)X$
where $y - X\beta$ is $n \times 1$ matrix, X is $n \times p$ matrix
- To match the final dimension $p \times 1$, we need to let
- $\frac{\partial (y - X\beta)^T (y - X\beta)}{\partial \beta} = -2X^T(y - X\beta)$
- where
 - X^T is $p \times n$ and $y - X\beta$ is $n \times 1$
 - $-2X^T(y - X\beta)$ is $p \times 1$

Least squares estimation

- Now we have

$$\frac{\partial \varepsilon^T \varepsilon}{\partial \beta} = \frac{\partial (y - X\beta)^T (y - X\beta)}{\partial \beta} = -2X^T(y - X\beta) = -2X^T y + 2X^T X\beta = 0$$

- Cancel constant 2 and move one item to the other side of the equation, we have

$$X^T X \hat{\beta} = X^T y$$

- We use “hat” to indicate that $\hat{\beta}$ is our solution or estimated parameter
- Suppose $X^T X$ is invertible, multiply $(X^T X)^{-1}$ on both side

$$\hat{\beta} = (X^T X)^{-1} X^T y$$

Prediction and Residual

- The predicted values of the response at X are

$$\hat{y} = X\hat{\beta} = X(X^T X)^{-1}X^T y = Hy$$

where $H = X(X^T X)^{-1}X^T$ is called hat matrix

- Residuals are the difference between model fit \hat{y} and true response y

$$\hat{\varepsilon} = y - \hat{y} = y - X\hat{\beta} = y - Hy = (I - H)y$$

One example

- The dataset `gala` contains the number of species found on the various Galápagos Islands
- There are 30 cases (Islands) and seven variables in the dataset
- We only use six of them

```
> data(gala, package="faraway")
> head(gala[,-2])
  Species   Area Elevation Nearest Scruz Adjacent
Baltra      58  25.09       346     0.6    0.6    1.84
Bartolome   31   1.24       109     0.6   26.3  572.33
Caldwell     3   0.21       114     2.8   58.7    0.78
Champion    25   0.10        46     1.9   47.4    0.18
Coamano     2   0.05        77     1.9    1.9  903.82
Daphne.Major 18   0.34       119     8.0    8.0    1.84
```

One example

- Let's fit a linear model by using lm() function

```
> lmod <- lm(Species ~ Area + Elevation + Nearest + Scruz + Adjacent,  
+ data=gala)  
> summary(lmod)  
  
Call:  
lm(formula = Species ~ Area + Elevation + Nearest + Scruz +  
+ Adjacent, data = gala)  
  
Residuals:  
    Min      1Q  Median      3Q     Max  
-111.68 -34.90   -7.86   33.46  182.58  
  
Coefficients:  
            Estimate Std. Error t value Pr(>|t|)  
(Intercept) 7.06822  19.15420   0.37  0.7154  
Area        -0.02394  0.02242  -1.07  0.2963  
Elevation    0.31946  0.05366   5.95 0.0000038 ***  
Nearest      0.00914  1.05414   0.01  0.9932  
Scruz        -0.24052  0.21540  -1.12  0.2752  
Adjacent     -0.07480  0.01770  -4.23  0.0003 ***  
---  
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1  
  
Residual standard error: 61 on 24 degrees of freedom  
Multiple R-squared:  0.766,    Adjusted R-squared:  0.717  
F-statistic: 15.7 on 5 and 24 DF,  p-value: 6.84e-07
```

One example

- Now Let's fit a linear model by using $\hat{\beta} = (X^T X)^{-1} X^T y$
- First, construct a design matrix X and response variable y

```
x <- model.matrix(~ Area + Elevation + Nearest + Scruz + Adjacent,  
gala)  
y <- gala$Species
```

- Second, calculate $(X^T X)^{-1}$

```
xtxi <- solve(t(x) %*% x)
```

- Third, calculate $(X^T X)^{-1} X^T y$

```
xtxi %*% t(x) %*% y
```

	(Intercept)	Area	Elevation	Nearest	Scruz	Adjacent
Baltra	1	25.09	346	0.6	0.6	1.84
Bartolome	1	1.24	109	0.6	26.3	572.33
Caldwell	1	0.21	114	2.8	58.7	0.78
Champion	1	0.10	46	1.9	47.4	0.18
Coamano	1	0.05	77	1.9	1.9	903.82
Daphne.Major	1	0.34	119	8.0	8.0	1.84
Daphne.Minor	1	0.08	93	6.0	12.0	0.34

One example

Fit by $(X^T X)^{-1} X^T y$

xtxi	%*%	t(x)	%*%	y
			[, 1]	
1		7.068221		
Area		-0.023938		
Elevation		0.319465		
Nearest		0.009144		
Scruz		-0.240524		
Adjacent		-0.074805		

Fit by lm() function

Coefficients:	
	Estimate
(Intercept)	7.06822
Area	-0.02394
Elevation	0.31946
Nearest	0.00914
Scruz	-0.24052
Adjacent	-0.07480

Analysis of Variance: Goodness of fit

- We use R^2 to measure how well the model fits the data

$$R^2 = 1 - \frac{\sum(\hat{y}_i - y_i)^2}{\sum(y_i - \bar{y})^2} = 1 - \frac{SS_{Error}}{SS_{Total}}$$

- $0 \leq R^2 \leq 1$: values closer to one indicates better fits
- Intuitively, better fitting \rightarrow smaller SS_{Error} \rightarrow larger R^2
- $R^2 = 1$: perfect fitting
- $R^2 = 0$: worst fit (e.g., using \bar{y} for all fitted values)