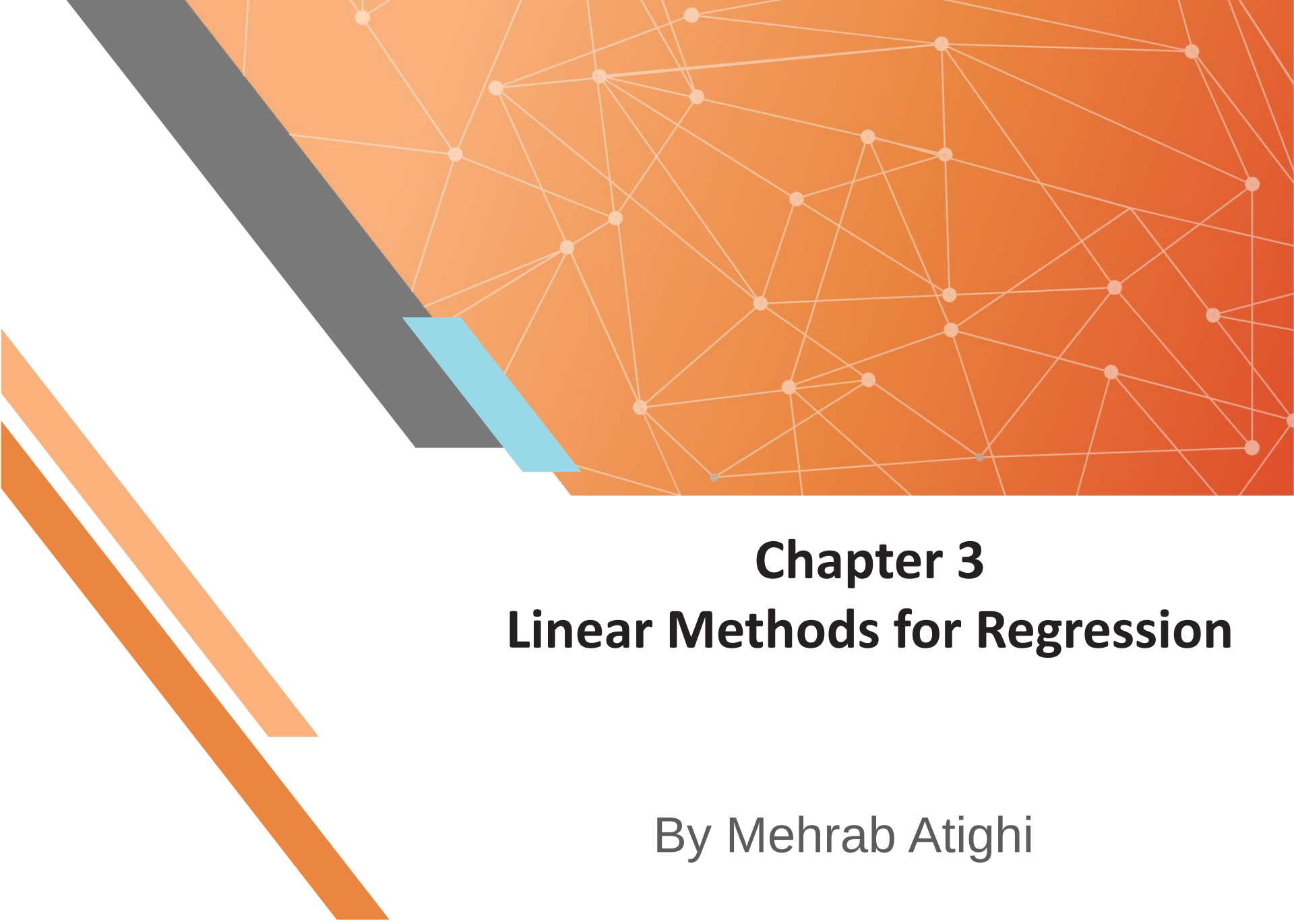

Chapter 3
Linear Methods for Regression

By Mehrab Atighi

Linear Regression Models and Least Squares

Linear Regression form

- $f(X) = \beta_0 + \sum_{j=1}^{p} X_j \beta_j$

Least Squares

- The most popular estimation method is *least squares*, in which we pick the coefficients $\beta = (\beta_0, \beta_1, \ldots, \beta_p)^T$ to minimize the residual sum of squares

- $RSS(\beta) = \sum_{i=1}^{N} (y_i - f(x_i))^2 = \sum_{i=1}^{N} \left(y_i - \beta_0 - \sum_{j=1}^{p} X_{ij}\beta_j\right)^2$

مدل رگرسیونی p متغیره

یک مدل رگرسیونی با p متغیر $x_i(x_{i1}, \dots, x_{ip})$ و خطای ε_i به صورت زیر است :

$$y_i = f(x_i) + \varepsilon_i$$

که f در آن یک تابع ثابت از متغیرهای $x_1, \dots, x_p$ است و اطلاعات سیستماتیکی که p تا x، درمورد y به ما می‌دهد را مشخص می‌کند.

چرا تابع f را برآورد بکنیم؟

1) برای انجام پیش‌گویی

2) برای انجام استنباط

Prediction
پیش‌گویی

یک برآورد از تابع اصلی ما که مجهول است می‌توان پیشگویی یا prediction برای y باشد .

$$\hat{y} = \widehat{f(x)}$$

میزان دقت $\hat{y}$ به دو کمیت وابسته است:

$$y_i = \overbrace{f(x_i)}^{1} + \overbrace{\hat{\varepsilon}_i}^{2}$$

۱) خطای تحویل پذیر (Reducibble error):

- می‌دانیم که $\hat{f}$ یک برآورد دقیق برای f نخواهد بود و مهم نیست که تعداد متغیرهای ما چندتا باشد، و این یعنی خطایی که بین چیزی که واقعیت است و چیزی که ما داریم بدست می‌آوریم، خطای تحویل پذیر می‌گوییم.

۲) خطای تحویل ناپذیر(IrReducible error):

- حتی اگر ما یک پیش‌بینی کاملا دقیق و بدون خطای تحویل پذیر داشته باشیم، باز هم y ما خطا دارد زیرا ε_i ها وجود دارند و چون y تابعی از آنهاست و این ε_i ها مستقل از x ها هستند، پس مهم نیست که ما چقدر f را خوب برآورد کرده باشیم، ما نمی‌توانیم خطای ε_i ها را کاهش بدهیم.

Inference
استنباط

در این روش ما به اینکه نوع رابطه‌ای که بین x,y است می‌پردازیم و یا به عبارتی اگر $\bar{x} = (x_1, \dots, x_p)$ تغییر بکند چه تاثیری روی y می‌گذارد؟

1. کدام یک از متغییرهای پیشگو ما تاثیر بیشتری دارند؟ باید بتوانیم از بین آنها، مفیدترها و موثرترها را پیدا بکنیم.
2. رابطه بین متغیرپاسخ و متغیرهای پیشگو ما چگونه است؟ وابسته به پیچیدگی f است.
3. آیا رابطه بین متغیر پاسخ و هریک از متغیرهای پیشگو را می‌توانیم بصورت خطی در بیاریم؟ بهترین مدل، مدلی ساده می‌باشد که بازدهی خوبی داشته باشد و دقیق تر باشد.

انواع روش‌های برآورد کردن f

روش‌های پارامتری:

- در این روش، مسله‌ی برآوردکردن f، که p بعدی نیز می‌باشد به برآوردکردن یک مجموعه از پارامترها تنزل پیدا می‌کند. یعنی بجای برآوردکردن f با p بعد، مسله را به برآوردکردن یک تعداد پارامتر کاهش می‌دهیم.

روش‌های ناپارامتری:

- در این روش ما درمورد تابع f، هیچ فرضی را قرار نمی‌هیم یعنی فرم یا شکل خاصی قرار نمی‌دهیم. دراین صورت دنبال یک برآوردی از f می‌گردیم که تاجایی که ممکن است به نقاطی که ما داریم نزدیک و هموار باشند.

هدف نهایی ما از برآوردکردن:
مدلی را انتخاب بکنیم که هم دقت پیشگویی آن و هم تفسیرپذیری مدل مدنظر ما می‌باشد و باید یک توازنی بین این دو پارامتر برقرار بکنیم ولذا نه خیلی پیچیده باشد و نه خیلی ساده و بی‌دقت.

Subset selection methods

- **این روش شامل ۳ بخش مختلف می‌شود:**

1. انتخاب بهترین مدل $Best\ subset\ selection$
2. انتخاب بهترین مدل گام به گام$Step\ wise\ selection$
3. انتخاب مدل بهینه $Choosing\ the\ Optimal\ Model$

Best Subset selection

در این روش ما می‌خواهیم با انتخاب یک زیرمجموعه‌ای از p متغیر پیشگویی که داریم، به ساختن مدل بپردازیم و براساس یکسری معیارها آنها را ارزیابی بکنیم.

در این روش ابتدا رگرسیون حداقل مربعات را با تمام ترکیب‌های مختلف pتا متغیر پیشگو خود انجام می‌دهیم و گام‌های این روش به صورت زیراست.

گام اول: در گام اول مدل M_0 خود را یا همان مدلی که هیچ متغیری داخل آن نیست را می‌سازیم(میانگین نمونه‌ای داده‌های ما)

گام دوم: برای $k = 1,2, \dots ,p$ بیاییم و تمام ترکیب‌های $\binom{p}{k}$ که دقیقا شامل k، متغیرپیشگو هست را برازش بدهیم و بهترین این مدل‌هایی که تشکیل داده‌ایم را براساس معیارمورد نظر انتخاب کرده و M_k می‌نامیم و لذا ما p+1 مدل رگرسیونی خواهیم داشت.

گام سوم: دراین مرحله از بین p+1 مدلی که داریم یعنی $M_0, \dots ,M_p$ براساس معیارمورد نظر بهترین مدل را انتخاب می‌کنیم.

نکته: در این روش حتما می‌بایست $n > p$ باشد.

نکته: معیارهای گام دوم و سوم نیز می‌تواند متفاوت باشد.

نکته: تا زمانی که $p \leq 8$ باشد، می‌توانیم از این روش استفاده بکنیم درغیر این صورت روش‌های دیگری وجود دارد.

Best Subset selection

$$\#R\ codes\ for\ Best\ subset\ selection\ method$$

$$library(\text{"leaps"})$$

$$best - subset < -regsubsets(y \sim x, data, nvmax = 19)$$

$$summary(best - subset)$$

Step Wise selection

زمانی که تعداد متغیرهای ما بیشتر از ۸ تا بود این روش بهتر جواب خواهد داد.

روش انتخاب مدل گام به گام نیز خود شامل ۳ روش مختلف می‌باشد.

1. روش گام به گام پیش‌رو $Forward\ step\ wise$
2. روش گام به گام پس‌رو $Backward\ step\ wise$
3. روش ترکیبی $Hybrid\ Approaches$

Forward Step Wise

در روش گام‌به‌گام پیش‌رو، درهرگام یک متغیرپیشگو به مدل قبلی اضافه می‌کنیم و آن متغیر باید بهترین تاثیر را داشته باشد.

گام اول: M_0 ما یک مدل خالی و بدون متغییر پیشگو می‌باشد.

گام دوم: به ازای مقادیر مختلف k ، $K = 0,1,2, \dots, P-1$ می‌توانیم p-k متغییر را به مدل خود اضافه بکنیم. و آن مدل جدید را M_k می‌نامیم. یعنی مرحله اول p تا متغییر به M_0 اضافه می‌کنیم و طبق معیار بهترین آنها را M_k می‌نامیم. و در مرحله بعدی p-k تا متغییر را می‌توانیم به M_k اضافه بکنیم و به همین ترتیب...

گام سوم: حال از بین $p+1$ تا مدلی که داریم، $M_0, M_1, \dots, M_{p-1}$ می‌آییم و طبق معیار مورد نظر بهترین مدل را انتخاب می‌کنیم.

در این روش ما از بین 2^p تا مدلی که داریم، تضمینی وجود ندارد که بهترین مدل را انتخاب کرده باشیم.

```
#R codes
forward <-regsubsets(y~x , data , nvmax = 19 , method = "forward")
summary(forward)
```

Back Ward Step Wise

در این روش ما برعکس روش گام‌به‌گام پیش‌رو، یک مدل کامل داریم و درهرمرحله متغییری که اهمیت و تاثیرکمتری را دارد از مدل خود حذف می‌کنیم.

گام اول: ما با M_p که نشان‌دهنده‌ی یک مدل کامل با p متغییراست شروع می‌کنیم.

گام دوم: به ازای مقادیر $K = p,p-1,p-2,\dots,1$، می‌آییم p-k تا متغییر را از مدل M_p خود حذف می‌کنیم. و اسم آنرا M_{k-1}قرار می‌دهیم.

گام سوم: مجدد از بین $M_0,\dots,M_{p-1},M_p$ تا مدلی که داریم، بر اساس معیارهای خودمان بهترین مدل را انتخاب می‌کنیم. در این روش حتما باید $n > p$ باشد.

```
#R codes
backward <-regsubsets(y~x , data, nvmax = 19, metho = ("backward")
summary(backwrd)
```

Hybrid Approaches

این روش یک روش ترکیبی از روش‌های گام‌به‌گام پیش‌رو و پس‌رو می‌باشد.

در این روش می‌آید درکنار افزایش متغییر، به کاهش متغییر هم نگاه می‌کند و بهترین مدل را انتخاب می‌کند تا به بهترین انتخاب ممکن برسیم.

Criterions for Step3

برای انتخاب کردن معیارهای خود درگام سوم روش‌های بالا ۲ روش کلی وجود دارد.

1. بصورت غیرمستقیم خطای آزمون را محاسبه بکنیم و برآورد بکنیم.
2. بصورت مستقیم به بررسی خطای آزمون بپردازیم.

روش غیرمستقیم

در این روش ما بصورت غیر مستقیم خطای آزمون را محاسبه و برآورد می‌کنیم، به این طریق که خطا را تعدیل می‌کنیم و اریبی آنرا نیز برطرف می‌کنیم.

از معیارهای برآورد غیرمستقیم نیز می‌توانیم به C_p, AIC, BIC, $adjR^2$ اشاره بکنیم.

اگر ما یک مدل داشته باشیم که d متغییر داشته باشد دراین صورت

برای مدل با pتا متغییر

$\varepsilon_i \sim N(0, \sigma^2)$

$$C_p = \frac{1}{n}\left(Rss + 2d\,\widehat{\sigma^2}\right)$$

و لذا $2d\,\widehat{\sigma^2}$ برای حذف اریبی به Rss ما اضافه گردیده است.

هرچه d بیشتر شود، C_p ما بیشتر خواهد شد و کمترین مقدار این معیار، بهترین مدل را نشان می‌دهد یعنی هرچه کمتر باشد بهتر است.

روش غیر مستقیم

با توجه به C_p که در اسلاید قبلی درمورد آن بحث شد، می‌خواهیم معیارهای دیگر را معرفی بکنیم:

$$AIC = \frac{1}{\widehat{\sigma^2}} C_p$$

این معیار نیز، هرچه مقدار کمتری داشته باشد نشانه بهتر بودن مدل ما خواهد بود.

$$BIC = \frac{1}{n}\left(Rss + \log(\mathrm{n})\mathrm{d}\widehat{\sigma^2}\right)$$

این معیار نیز، هرچه مقدار کمتری داشته باشد نشانه بهتر بودن مدل ما خواهد بود، و اگر $\log(n) > 2$ یا $n > 7$ باشد، ضریب $d\widehat{\sigma^2}$ نسبت به ضریب C_p بیشتر خواهد بود.

$$R^2 = 1 - \left(\frac{Rss}{SST}\right) \quad , Adj.R^2 = 1 - \left(\frac{MSE}{MST}\right)$$

این معیار نیز، هرچه مقدار بیشتری داشته باشد نشانه بهتر بودن مدل ما خواهد بود.

$$\#R\ codes\ For\ AIC, BIC, C_p, Adjusted\ R\ squred:$$
$$result < -summary(best - subset)$$
$$Cp = result\$adjr2$$
$$BIC = result\$bic$$
$$which.\max(adj - R2)\ ; which.\min(Cp)\ ; which.\min(BIC)$$ اندیس را به ما می‌دهد.

روش مستقیم

در این روش ما بصورت مستقیم به بررسی خطای آزمون می‌پردازیم: در این روش ما باید خطای ارزیابی متقابل خود را برای مدل‌ها حساب کرده و بهترین را انتخاب بکنیم. ما دراینجا برای مدل‌های مختلف با تعداد متغییرهای پیشگو مختلف یک خطای استاندارد(one standard error) داریم. و برای هریک از این مدل‌های p متغییره، می‌آییم و مدلی را که کمترین میانگین انحراف استاندارد را دارد انتخاب می‌کنیم و آنرا با درنمودار خود + و − sd رسم می‌کنیم و بعد هریک از مدل‌هایی که داخل این بازه قرار داشتند، ساده ترین آنها بهترین مدل ما خواهد بود.

$$X = \begin{matrix} obs1 \\ \vdots \\ obsn \end{matrix} \begin{bmatrix} 1x_1 & \dots & x_p \\ 1 \vdots & \dots & \vdots \\ 1 \vdots & \dots & \vdots \end{bmatrix}_{n*p+1}$$

ما بعد از انتخاب بهترین مدل p متغیره خود، ضرایب آنرا استخراج می‌کنیم که آنها p+1 تا خواهند بود و بعد بصورت زیر مقادیر را پیش‌بینی خواهیم کرد.

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

روش مستقیم

```
#R codes for Cross Validaion set Approches Method:
Validation - erros < -vector("double",lenght=19)
for(i in 1:19){coef - x < -coef(best - subset, id = i)
pred - x < -test - m[, names(coef - x)]% * %coef - x
validation - errors[i] = mean((test$y - pred - x)^2)}
Plot(Validation - errors, type = "b")
```

روش مستقیم

```
#R codes for K Fold Cross Validation
predict.regsubset = function(object, newdata, id, ...){
    form <-as.formula(object$call[[2]])
    mat <-model.matrix(form, newdata)
    coefi <- coef(object, id = id)
    xvars <- names(coefi)
    mat[, xvars] %*% coefi
    }
    K <-10
    Folds <-sample(1:K , nrow(data), replace = TRUE)
    Cv.errors <-matrix (NA, k, 19, dimnames = list(NULL, paste(1:19)))
    for(j in 1:k){
    nest - subset <-regsubset(y~x, data[folds == j, ], nvmax = 19)
    for(i in 1:19){
    pred - x = predict.regsubset(best - subset, data[folds == j, ], id = i)
    Cv - errors[i, j] <-mean(data$y[folds == j] - pred - x)^2)
    }}
```

Regularization Regression

در روش‌های قبلی $n > p$ و برای بدست آوردن خطای آزمون $Min(Rss)$ را بدست می‌آوردیم.

اما حال اگر $p > n$ باشد و هرچه p افزایش پیدا بکند، شرایط ols نقض می‌شود یعنی با افزایش تعداد متغییرهای ما، احتمال وجود داشتن هم‌خطی بین متغییرها زیاد می‌شود و افزایش واریانس را خواهیم دید.

رگرسیون تابانیده: ***Ridge Regression***:

در این روش نیز هدف ما $minimize\{Rss + p(penalty)\}$ می‌باشد.

$$minimize\left\{\sum_{i=1}^{n}\left(y_i - \beta_0 - \sum_{j=1}^{p}\beta_j x_{ij}\right)^2 + \lambda\sum_{j=1}^{p}\beta_j^2\right\}$$

$\lambda \geq 0$ در اینجا پارامتر تنظیم کننده است، در این روش ما هم به دنبال کمترین مقدار Rss هستیم و هم به دنبال یک پنالتی انقباضی هستیم و زمانی این پنالتی ما کوچک می‌شود که $\beta_1, \ldots, \beta_p$ همگی به صفر نزدیک بشوند. λ اثر نسبی بین دوجمله را کنترل می‌کند یعنی اگر برابر با ۰ باشد که همان RSS را داریم و اگر به سمت بینهایت میل بکند، برای اینکه تابع هدف ما کمترین مقدار خود را بگیرد باید تمامی ضرایب رگرسیونی که داریم به ۰ میل بکنند. ما در این روش، به ازای هر مقدار λ یک مجموعه‌ای از ضرایب را داریم و آنها را به صورت $\widehat{\beta_\lambda}R$ یا $\widehat{\beta_\lambda}Ridge$ نشان می‌دهیم.

نکته قابل توجه، این هست که ما با مقادیر خود λ ها کاری نداریم و با مقدار لگاریتم آنها یعنی $\log(\lambda)$ کار داریم.

Ridge Regression

- $\#R\ codes\ for\ Ridge\ Regression$

$$data - y - train = \log(data_{train}\$y)$$

$$data - x - train = model.matrix(y \sim x\ , data_{train})[, -1]$$

$$data - x - test = model.matrix(y \sim x, data_{test})[, -1]$$

$$data - y - test = \log(data_{test}\$y)$$

در تابع glm.net باید توجه داشته باشیم که x,y ما باید از داده های اصلی جدا باشند و اینکه اگر متغیرهای ما کیفی بودند خودش خودکار طبقه‌بندی می‌کند و اگر داده های ما استاندارد شده بودند،آرگمان standadize را برابر با FALSE قرار بدهیم.

$$data - ridge = glm.net(x = data - x - train, y = data - y - train, alpha = 0)$$

$$plot(data - ridge, xvars = \text{"lambda"})$$

$$data - ridge\$lambda$$

برای نشان داده شدن به صورت نزولی: $ames - ridge\$lambda\% > \%head()$

برای‌دیدن کوچترین ضرایب لاندا: $coef(data - ridge)\ [\ c(\text{"predictor1"}, \text{"}predictor2, \dots)\ , 100]$

برای‌دیدن بزرگترین ضرایب لاندا: $coef(data - ridge)\ [\ c(\text{"predictor1"}, \text{"}predictor2, \dots)\ , 1]$

Ridge Regression with Cross Validation

$$\#R\ codes\ for\ Ridge\ Regression\ with\ Cross\ Validtaion:$$

$$fit$$
$$< -cv.glmnet(x = data - x - train\ , y$$
$$= data - y - train\ , alpha = 0)$$

$$Plot(fit)$$

$$\min(fit\$cvm)\ \#\ \min MSE$$

$$fit\$lambda.\min \quad \#\lambda for\ \min MSE$$

$$fit\$cvm[fit\$lambda == fit\$lambda.1se] \quad \#for\ \lambda\ , MSE + Sd$$

$$coef(fit, s = \text{"lambda.1se"})$$

Lasso Regression

در این روش نسبت به روش قبلی، پنالتی ما متفاوت خواهد بود و نه تنها نتایج را بهبود می‌بخشد، حتی یک انتخاب مدل نیز برای ما انجام می‌دهد . از دیگر مزایای این روش می‌توانیم به این اشاره بکنیم که دیگر ضرایب ما به ۰ میل نمیکنند بلکه دقیقا ۰ خواهند شد. و از معایب این روش می‌توانیم به کاهش دقت مدل اشاره بکنیم که از حذف متغیرها منشع دارد.

و تنها تفاوت از نظر کدهای زبان برنامه نویسی R این هست که مقدار $\alpha = 1$ باید باشد.

$$\#R\ Codes\ for\ lasso\ Regression\ with\ Cross\ Validation:$$
$$fit$$
$$< -cv.glmnet(x = data - x - train\ , y$$
$$= data - y - train\ , alpha = 1)$$
$$Fit\$lambda.\min$$
$$Fit.lambda.1se$$
$$pred$$
$$= predict(fit, s = cv - lasso\$lambda.min, data - x - test)$$
$$mean(((data - y - test) - pred)^2)$$

End

Thanks For Youre Attention