Doug Woodward

CS613 HW 9

|  |  |
| --- | --- |
| # | Answer |
| 1 | 1. Best subset gives the smallest RSS as the other models have a path dependency, iterating to the kth model. 2. Best subset Selection    1. True    2. True    3. False    4. False    5. False |
| 8 | The best model is a 3 variable model according to majority    (Intercept) poly(x, 10, raw = T)1 poly(x, 10, raw = T)2 poly(x, 10, raw = T)3   2.094409 -1.445153 3.766142 1.756419  (Intercept) poly(x, 10, raw = T)1 poly(x, 10, raw = T)2 poly(x, 10, raw = T)3   2.094409 -1.445153 3.766142 1.756419  [1] 4  [1] 5  [1] 4  [1] 5  [1] 7  [1] 5  The models are split between 4 and 5 variable models, with one 7 as well. There is not a real clear winner here, 5 would be most likely.  library(leaps)    set.seed(4)  x = rnorm(100)  eps = rnorm(100)  b0 = 2  b1 = -1  b2 = 4  b3 = 1.5  y = b0 + b1\*x+b2\*x^2+b3\*x^3+err  data.full = data.frame(y=y, x=x)  regsubs = regsubsets(y ~ poly(x, 10, raw = T), data = data.full, nvmax = 10)  reg.summary = summary(regsubs)  which.min(reg.summary$cp)  which.min(reg.summary$bic)  which.min(reg.summary$adjr2)  par(mfrow=c(2,2))  plot(reg.summary$cp, xlab = "Subset Size", ylab = "Cp", pch = 20, type = "l")  points(3, reg.summary$cp[3], pch = 4, col = "red", lwd = 7)  plot(reg.summary$bic, xlab = "Subset Size", ylab = "BIC", pch = 20, type = "l")  points(3, reg.summary$bic[3], pch = 4, col = "red", lwd = 7)  plot(reg.summary$adjr2, xlab = "Subset Size", ylab = "Adjusted R2", pch = 20, type = "l")  points(3, reg.summary$adjr2[3], pch = 4, col = "red", lwd = 7)  coefficients(regsubs, id = 3)  mod.fwd = regsubsets(y ~ poly(x, 10, raw = T), data = data.full, nvmax = 10, method = "forward")  mod.bwd = regsubsets(y ~ poly(x, 10, raw = T), data = data.full, nvmax = 10, method = "backward")  fwd.summary = summary(mod.fwd)  bwd.summary = summary(mod.bwd)  which.min(fwd.summary$cp)  which.min(bwd.summary$cp)  which.min(fwd.summary$bic)  which.min(bwd.summary$bic)  which.max(fwd.summary$adjr2)  which.max(bwd.summary$adjr2) |
|  | library(leaps)  set.seed(4)  p = 20  n = 1000  x = matrix(rnorm(n \* p), n, p)  B = rnorm(p)  B[3] = 0  B[4] = 0  B[9] = 0  B[19] = 0  B[10] = 0  eps = rnorm(p)  y = x %\*% B + eps  train = sample(seq(1000), 100, replace = FALSE)  y.train = y[train,]  y.test = y[-train,]  x.train = x[train,]  x.test = x[-train,]  regfit.full = regsubsets(y ~ ., data = data.frame(x = x.train, y = y.train), nvmax = p)  val.errors = rep(NA, p)  x\_cols = colnames(x, do.NULL = FALSE, prefix = "x.")  for (i in 1:p) {  coefi = coef(regfit.full, id = i)  pred = as.matrix(x.train[, x\_cols %in% names(coefi)]) %\*% coefi[names(coefi) %in%  x\_cols]  val.errors[i] = mean((y.train - pred)^2)  }  plot(val.errors, ylab = "Training MSE", pch = 19, type = "b")  which.min(val.errors)  coef(regfit.full, id = 16)  val.errors = rep(NA, p)  a = rep(NA, p)  b = rep(NA, p)  for (i in 1:p) {  coefi = coef(regfit.full, id = i)  a[i] = length(coefi) - 1  b[i] = sqrt(sum((B[x\_cols %in% names(coefi)] - coefi[names(coefi) %in% x\_cols])^2) +  sum(B[!(x\_cols %in% names(coefi))])^2)  }  plot(x = a, y = b, xlab = "number of coefficients", ylab = "error between estimated and true coefficients")  which.min(b)  [1] 20  (Intercept) x.1 x.2 x.5 x.6 x.7 x.8  x.9 x.11 x.12  0.09672144 0.89232674 0.21590685 0.17031629 -0.37508422 1.22016947 0.99899589 0.122  46537 -1.54124911 -1.34373352  x.13 x.14 x.15 x.16 x.17 x.18 x.20  1.07424956 -0.12875703 1.41695605 0.49461087 1.67981743 -0.53264561 2.03548429  [1] 12 |