Case Study 9: Model selection: Predicting prostate cancer

Richard Wilkinson

The data

The data for this example come from a study by Stamey et al. (1989) that examined the correlation between the level of prostate specific antigen (PSA) and a number of clinical measures, in 97 men who were about to receive a radical prostatectomy. PSA is a protein that is produced by the prostate gland. The higher a man's PSA level, the more likely it is that he has prostate cancer.

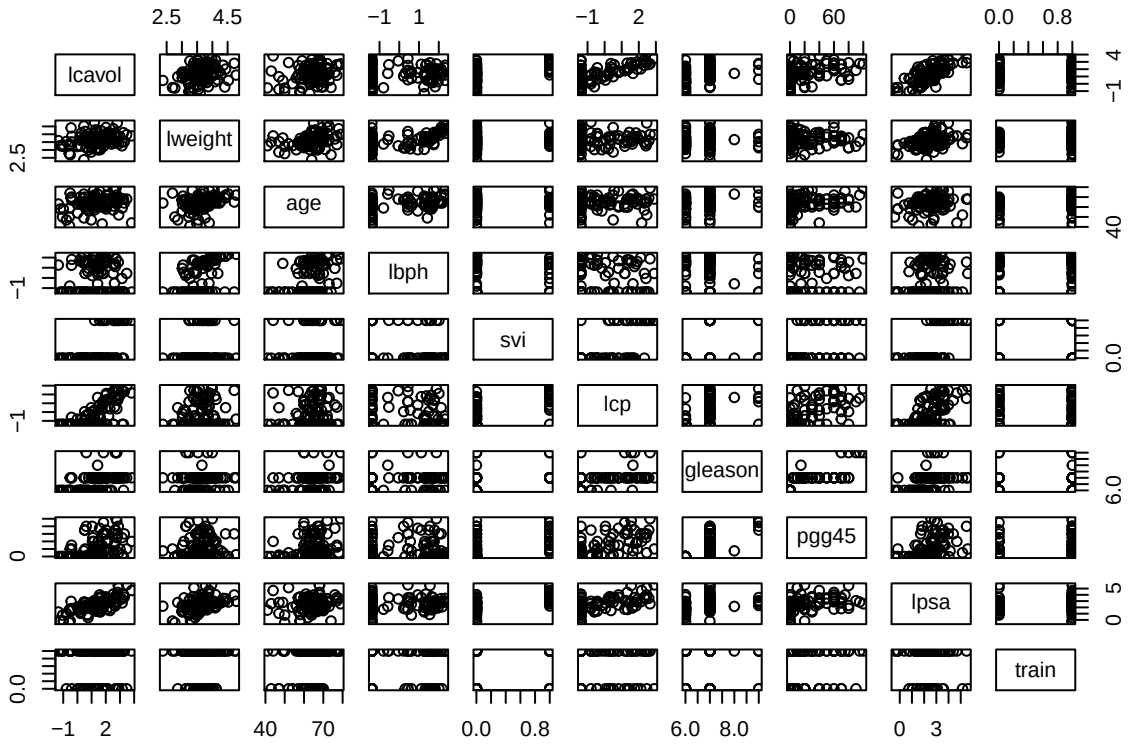
The goal is to predict the log of PSA (lpsa) from a number of measurements including log cancer volume (lcavol), log prostate weight (lweight), age, log of benign prostatic hyperplasia amount (lbph), seminal vesicle invasion (svi), log of capsular penetration (lcp), Gleason score (gleason), and percent of Gleason scores 4 or 5 (pgg45).

The data are modelled in depth in the beautiful (but somewhat advanced) book *The Elements of Statistical Learning*, which is available for free at http://statweb.stanford.edu/~tibs/ElemStatLearn/

```
library(ElemStatLearn)
data(prostate)
str(prostate)
```

```
## 'data.frame':    97 obs. of  10 variables:
##  $ lcavol : num  -0.58 -0.994 -0.511 -1.204 0.751 ...
##  $ lweight: num  2.77 3.32 2.69 3.28 3.43 ...
##  $ age    : int  50 58 74 58 62 50 64 58 47 63 ...
##  $ lbph   : num  -1.39 -1.39 -1.39 -1.39 -1.39 ...
##  $ svi    : int  0 0 0 0 0 0 0 0 0 0 ...
##  $ lcp    : num  -1.39 -1.39 -1.39 -1.39 -1.39 ...
##  $ gleason: int  6 6 7 6 6 6 6 6 6 6 ...
##  $ pgg45  : int  0 0 20 0 0 0 0 0 0 0 ...
##  $ lpsa   : num  -0.431 -0.163 -0.163 -0.163 0.372 ...
##  $ train  : logi  TRUE TRUE TRUE TRUE TRUE TRUE ...
```

```
plot(prostate)
```



The scatter plot matrix suggests some correlation between the covariates and lpsa. The data have been randomly split into a training set and a test set. Lets create two seperate datasets for ease.

```
library(dplyr)
prostTrain <- filter(prostate, train)
prostTest <- filter(prostate, !train)
prostTrain <-select(prostTrain, -train)
prostTest <-select(prostTest, -train)
```

We will use the training set to build the model, and then try to predict the test set. By comparing different models on the test set, we can compare model perfomance. We could repeat this with different assignment of data points to training or test set (a process that is then known as cross validation.)

Lets start by fitting a linear model.

```
fit <- lm(lpsa~., data=prostTrain)
coef(fit)
```

```
##  (Intercept)       lcavol      lweight          age         lbph
##  0.429170133  0.576543185  0.614020004 -0.019001022  0.144848082
##          svi          lcp      gleason        pgg45
##  0.737208645 -0.206324227 -0.029502884  0.009465162
```

```
predictions <- predict(fit, newdata=select(prostTest, -lpsa))
mse_MSE <- mean((predictions - select(prostTest, lpsa))^2)
mse_MSE
```

```
## [1] 0.521274
```

Here I've calculated the mean square prediction error on the test data.

Best subsets regression

Lets now look at best subsets regression. Here, every possible model in the model hierarchy is tried (all 2^8 models), and the criterion calculated for each.

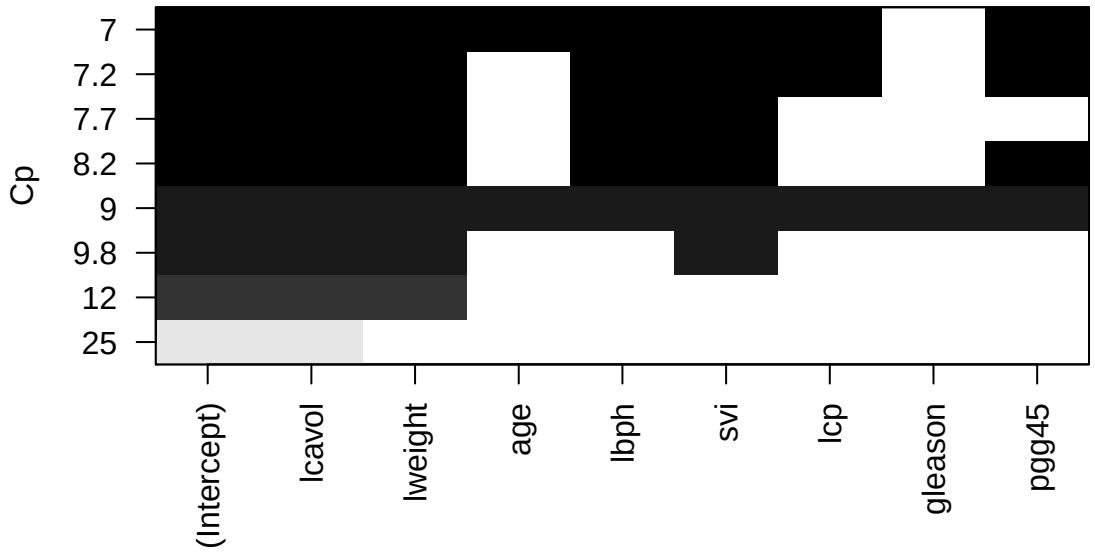
```
require(leaps)
a<-regsubsets(lpsa~., data=prostTrain)
summary.out <-summary(a)
summary.out
```

```
## Subset selection object
## Call: regsubsets.formula(lpsa ~ ., data = prostTrain)
## 8 Variables  (and intercept)
##         Forced in Forced out
## lcavol      FALSE      FALSE
## lweight     FALSE      FALSE
## age         FALSE      FALSE
## lbph        FALSE      FALSE
## svi         FALSE      FALSE
## lcp         FALSE      FALSE
## gleason     FALSE      FALSE
## pgg45       FALSE      FALSE
## 1 subsets of each size up to 8
## Selection Algorithm: exhaustive
##          lcavol lweight age lbph svi lcp gleason pgg45
## 1  ( 1 ) "*"    " "     " " " "  " " " " " " "     " "
## 2  ( 1 ) "*"    "*"     " " " "  " " " " " " "     " "
## 3  ( 1 ) "*"    "*"     " " " "  "*" " " " " "     " "
## 4  ( 1 ) "*"    "*"     " " "*"  "*" " " " " "     " "
## 5  ( 1 ) "*"    "*"     " " "*"  "*" " " " " "     "*"
## 6  ( 1 ) "*"    "*"     " " "*"  "*" "*" " " "     "*"
## 7  ( 1 ) "*"    "*"     "*" "*"  "*" "*" " " "     "*"
## 8  ( 1 ) "*"    "*"     "*" "*"  "*" "*" "*"     "*"
```

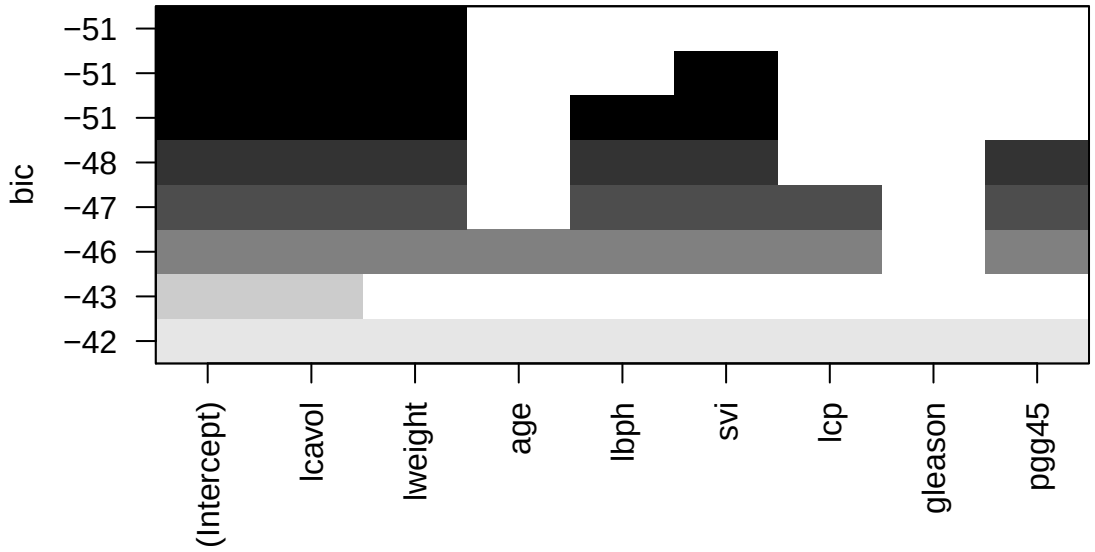
```
summary.out$cp
```

```
## [1] 24.766739 12.108779  9.803880  7.679020  8.209530  7.194521  7.021515
## [8]  9.000000
```

```
plot(a, scale='Cp')
```




```
plot(a, scale='bic')
```



If we focus on the *BIC*, this suggests we try the three models

```
fit1 <- lm(lpsa ~ lcavol + lweight, data=prostTrain)
fit2 <- lm(lpsa ~ lcavol + lweight+svi, data=prostTrain)
fit3 <- lm(lpsa ~ lcavol + lweight+svi+lbph, data=prostTrain)

predictions <- predict(fit1, newdata=select(prostTest, -lpsa))
mse_bestsubets1 <- mean((predictions - select(prostTest, lpsa))^2)

predictions <- predict(fit2, newdata=select(prostTest, -lpsa))
mse_bestsubets2 <- mean((predictions - select(prostTest, lpsa))^2)

predictions <- predict(fit3, newdata=select(prostTest, -lpsa))
mse_bestsubets3 <- mean((predictions - select(prostTest, lpsa))^2)

mse_bestsubets1
```

```
## [1] 0.4924823
```

```
mse_bestsubets2
```

```
## [1] 0.4005308
```

```
mse_bestsubets3
```

```
## [1] 0.4563321
```

Stepwise regression

In this case, $n = 67$ and there are only 256 possible models, and so an exhaustive search is possible. But suppose it were not, then we could resort to stepwise regression instead.

Here, we use the AIC as the criterion to select the model. R uses a slightly different version of the AIC to that given in the notes, which differs by an additive constant to the standard definition of AIC. However, as only relative differences matter, this makes no difference to the end result.

```
fit_step1 <- step(fit)
```

```
## Start:  AIC=-37.13
## lpsa ~ lcavol + lweight + age + lbph + svi + lcp + gleason +
##     pgg45
##
##           Df Sum of Sq    RSS     AIC
## - gleason  1    0.0109 29.437 -39.103
## <none>                 29.426 -37.128
## - age      1    0.9886 30.415 -36.914
## - pgg45    1    1.5322 30.959 -35.727
## - lcp      1    1.7683 31.195 -35.218
## - lbph     1    2.1443 31.571 -34.415
## - svi      1    3.0934 32.520 -32.430
## - lweight  1    3.8390 33.265 -30.912
## - lcavol   1   14.6102 44.037 -12.118
##
## Step:  AIC=-39.1
## lpsa ~ lcavol + lweight + age + lbph + svi + lcp + pgg45
##
##           Df Sum of Sq    RSS     AIC
## <none>                 29.437 -39.103
## - age      1    1.1025 30.540 -38.639
## - lcp      1    1.7583 31.196 -37.216
## - lbph     1    2.1354 31.573 -36.411
## - pgg45    1    2.3755 31.813 -35.903
## - svi      1    3.1665 32.604 -34.258
## - lweight  1    4.0048 33.442 -32.557
## - lcavol   1   14.8873 44.325 -13.681
```

```
fit0 <- lm(lpsa ~1, data=prostTrain)
fit_step2 <- step(fit0, scope = lpsa~lcavol + lweight + age +lbph + svi + lcp + gleason + pgg45)
```

```
## Start:  AIC=26.29
## lpsa ~ 1
##
##           Df Sum of Sq    RSS      AIC
## + lcavol   1    51.753 44.529 -23.3736
## + svi      1    29.859 66.422   3.4199
## + lcp      1    23.042 73.239   9.9657
## + lweight  1    22.668 73.614  10.3071
## + pgg45    1    19.328 76.953  13.2799
## + gleason  1    11.290 84.992  19.9368
## + lbph     1     6.657 89.625  23.4930
## + age      1     4.989 91.292  24.7279
## <none>                96.281  26.2931
##
## Step:  AIC=-23.37
## lpsa ~ lcavol
##
##           Df Sum of Sq    RSS     AIC
## + lweight  1     7.437 37.092 -33.617
## + lbph     1     4.536 39.992 -28.572
## + svi      1     2.216 42.313 -24.794
## <none>                44.529 -23.374
## + pgg45    1     1.105 43.423 -23.058
## + gleason  1     0.105 44.424 -21.531
## + lcp      1     0.061 44.468 -21.465
## + age      1     0.033 44.496 -21.423
## - lcavol   1    51.753 96.281  26.293
##
## Step:  AIC=-33.62
## lpsa ~ lcavol + lweight
##
##           Df Sum of Sq    RSS     AIC
## + svi      1     2.184 34.908 -35.683
## + pgg45    1     1.658 35.434 -34.680
## <none>                37.092 -33.617
## + lbph     1     1.077 36.015 -33.590
## + gleason  1     0.433 36.658 -32.404
## + age      1     0.275 36.817 -32.115
## + lcp      1     0.002 37.090 -31.621
## - lweight  1     7.437 44.529 -23.374
## - lcavol   1    36.522 73.614  10.307
##
## Step:  AIC=-35.68
## lpsa ~ lcavol + lweight + svi
##
##           Df Sum of Sq    RSS     AIC
## + lbph     1    2.0928 32.815 -37.825
## <none>                34.908 -35.683
## + pgg45    1    0.8336 34.074 -35.302
## + lcp      1    0.6341 34.274 -34.911
## + gleason  1    0.3011 34.607 -34.263
## + age      1    0.1956 34.712 -34.059
## - svi      1    2.1841 37.092 -33.617
## - lweight  1    7.4048 42.313 -24.794
```

```
## - lcavol   1   16.8065 51.714 -11.350
##
## Step:  AIC=-37.83
## lpsa ~ lcavol + lweight + svi + lbph
##
##           Df Sum of Sq    RSS     AIC
## <none>                  32.815 -37.825
## + pgg45    1    0.7455 32.069 -37.365
## + age      1    0.5306 32.284 -36.917
## + lcp      1    0.4922 32.323 -36.838
## + gleason  1    0.1781 32.637 -36.190
## - lbph     1    2.0928 34.908 -35.683
## - lweight  1    3.1545 35.969 -33.675
## - svi      1    3.2002 36.015 -33.590
## - lcavol   1   15.7863 48.601 -13.510
```

```
coef(fit_step1)
```

```
##  (Intercept)       lcavol      lweight          age         lbph
##  0.259061747  0.573930391  0.619208833 -0.019479879  0.144426474
##          svi          lcp        pgg45
##  0.741781258 -0.205416986  0.008944996
```

```
coef(fit_step2)
```

```
## (Intercept)      lcavol     lweight         svi        lbph
##  -0.3259212   0.5055209   0.5388292   0.6718487   0.1400111
```

```
predictions <- predict(fit_step1, newdata=select(prostTest, -lpsa))
mse_step1 <- mean((predictions - select(prostTest, lpsa))^2)

predictions <- predict(fit_step2, newdata=select(prostTest, -lpsa))
mse_step2 <- mean((predictions - select(prostTest, lpsa))^2)
mse_step1
```

```
## [1] 0.5165135
```

```
mse_step2
```

```
## [1] 0.4563321
```

Note the very different answers found depending upon where we start the searches from.

Ridge regression

Finally, ridge regression.

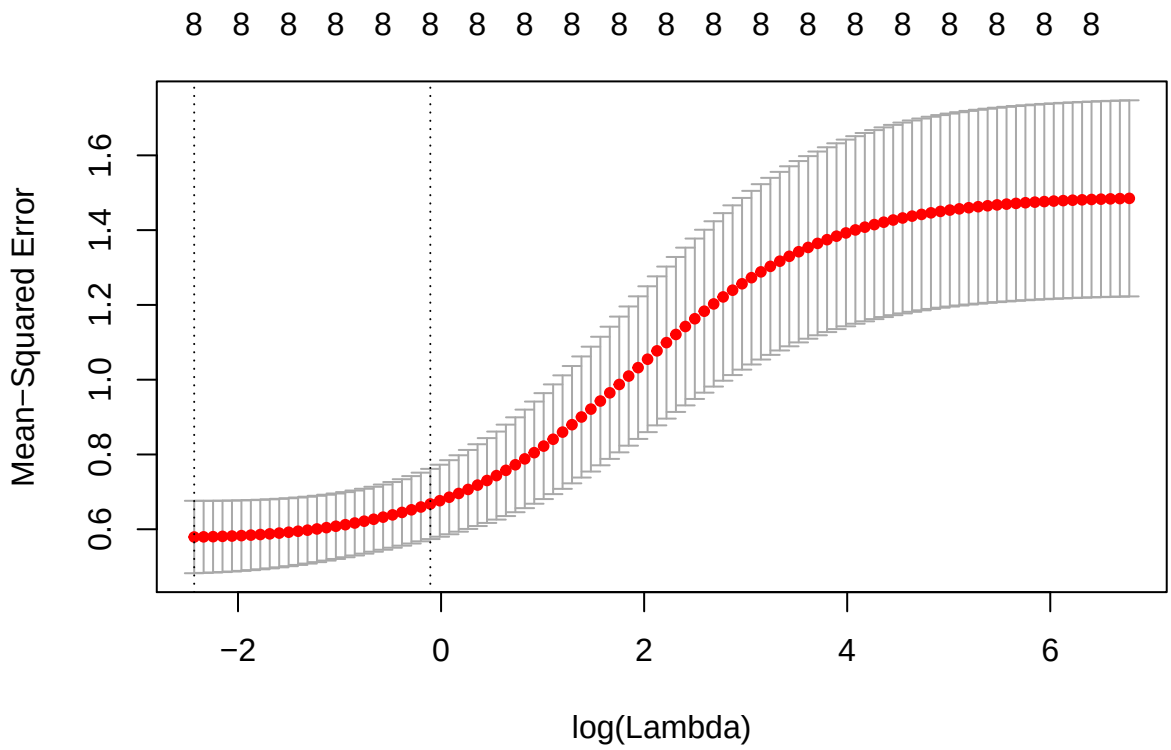
```
library(glmnet)
x =   select(prostTrain, c(lcavol, lweight, age, lbph, svi, lcp, gleason, pgg45))
x=as.matrix(x)
y =   select(prostTrain, lpsa)
y=as.matrix(y)
ridge=glmnet(x,y, alpha=0)
ridge.cv=cv.glmnet(x,y, alpha=0)
ridge.cv$lambda.min
```

```
## [1] 0.08788804
```

```
ridge.cv$lambda.1se
```

```
## [1] 0.8995614
```

```
plot(ridge.cv)
```




```
xnew = select(prostTest,  c(lcavol, lweight, age, lbph, svi, lcp, gleason, pgg45))
xnew = as.matrix(xnew)
ridge.prediction1 = predict(ridge.cv, xnew, s = "lambda.1se")
ridge.prediction2 = predict(ridge.cv, xnew, s = "lambda.min")
mse_ridge1 <-  mean((ridge.prediction1 - select(prostTest, lpsa))^2)
mse_ridge2 <-  mean((ridge.prediction2 - select(prostTest, lpsa))^2)

mse_ridge1
```

```
## [1] 0.5161568
```

```
mse_ridge2
```

```
## [1] 0.4943793
```

So for this test set, the model

$$lpsa = a + b \times lcavol + c \times lweight + d \times svi + \epsilon$$

gives the best performance on the test set. If we split the data into test and training sets in a different way, we may find a different model performs better. Taking many such splits into test and training set, and finding the average prediction error on the test set, is a process called cross-validation, and is a key method for assessing the predictive performance of a model.