# Assignment 3

```
library(tidyverse)
```

1.The Titanic dataset records for each person on the ship the passenger class, age (child or adult), and sex, and whether they survived or not. In this assignment you will use logistic regression on a training set (ttrain) to develop a classification rule, and then this rule will be applied to the test set (ttest).

```
ttrain <- read.csv("data/ttrain.csv", header = TRUE, row.names = 1)
ttest <- read.csv("data/ttest.csv", header = TRUE, row.names = 1)
head(ttrain)
```

```
##      Class    Sex   Age Survived
## 633    3rd   Male Adult       No
## 1735  Crew   Male Adult      Yes
## 900   Crew   Male Adult       No
## 1941   1st Female Adult      Yes
## 2067   2nd Female Adult      Yes
## 101    1st   Male Adult       No
```

(a) Use logistic regression to build a model relating Survived to Class, Age and Sex for the training data ttrain.

```
model <- glm(Survived ~ Class + Age + Sex, data = ttrain,
             family = "binomial")
summary(model)
```

```
##
## Call:
## glm(formula = Survived ~ Class + Age + Sex, family = "binomial",
##     data = ttrain)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -2.1232  -0.7173  -0.4496   0.6768   2.1642
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)   2.1431     0.1922  11.153  < 2e-16 ***
## Class2nd     -1.0136     0.2219  -4.567 4.95e-06 ***
## Class3rd     -1.8467     0.1952  -9.462  < 2e-16 ***
## ClassCrew    -0.8321     0.1779  -4.678 2.90e-06 ***
## AgeChild      1.0606     0.2889   3.671 0.000242 ***
## SexMale      -2.5373     0.1606 -15.795  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 2214.5  on 1760  degrees of freedom
## Residual deviance: 1737.6  on 1755  degrees of freedom
## AIC: 1749.6
##
## Number of Fisher Scoring iterations: 4
```

(b) From the fitted model, calculate a vector prob of survival probabilities and a vector pred of predicted classes, for the training data. What proportion of survivors are missclassified? What proportion of those who died are missclassified? What proportion of the predicted survivors actually survived? What is the overall error rate for the training data?

```r
pred_prob <- predict(model, type = "response")
ttrain$pred_class <- ifelse(pred_prob < 0.5, "No", "Yes")

# table(ttrain$Survived, pred_class)

ttrain %>%
  group_by(pred_class, Survived) %>%
  count() %>%
  ungroup() %>%
  mutate(perc = scales::percent(n/sum(n)))
```

```
## # A tibble: 4 x 4
##   pred_class Survived     n perc
##   <chr>      <fct>    <int> <chr>
## 1 No         No        1097 62.3%
## 2 No         Yes        284 16.1%
## 3 Yes        No          96 5.5%
## 4 Yes        Yes        284 16.1%
```

```r
# Error = 5.5 + 16.1 = 21.6%
```

(c) From the fitted model, calculate a vector prob of survival probabilities and a vector pred of predicted classes, for the test data. What proportion of survivors are missclassified? What proportion of those who died are missclassified? What proportion of the predicted survivors actually survived? What is the overall error rate for the test data?

```r
# Probabilities
pred_prob_test <- predict(model, type = "response", newdata = ttest)
ttest$pred_class <- ifelse(pred_prob_test < 0.5, "No", "Yes")

ttest %>%
  group_by(pred_class, Survived) %>%
  count() %>%
  ungroup() %>%
  mutate(perc = scales::percent(n/sum(n)))
```

```
## # A tibble: 4 x 4
##   pred_class Survived     n perc
##   <chr>      <fct>    <int> <chr>
## 1 No         No         267 60.7%
## 2 No         Yes         78 17.7%
## 3 Yes        No          30 6.8%
## 4 Yes        Yes         65 14.8%
```

```
# Error = 6.8 + 17.7 = 24.5%
```

2. Suppose we wish to predict whether a given stock will issue a dividend this year (yes or no) based on X, last year's percentage profit. We examine a large number of companies and discover that the mean value of X for companies that issued a dividend was 10, while the mean for those that didn't was 0. In addition, the variance of X for these two sets of companies was 36. Finally, 80% of companies issued dividends. Assuming that X follows a normal distribution, predict the probability that a company will issue a dividend this year given that its percentage profit was X = 4 last year.

```
# Probability of issuing dividend
p_div <- 0.8*exp(- (1/72) * (4 - 10)^2)
# Probability of non issuing dividend
p_ndiv <- 0.2*exp(- (1/72) * (4 - 0)^2)

# Result
p_div/(p_div + p_ndiv)
```

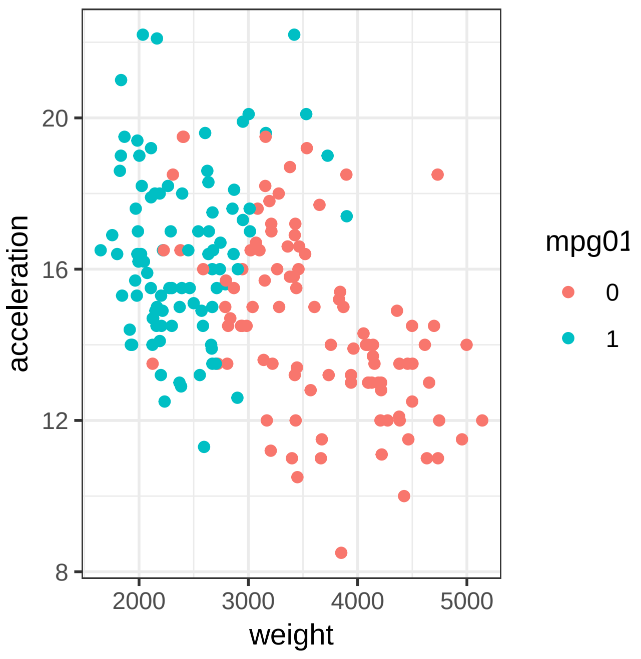
```
## [1] 0.7518525
```

3. In the Auto data, create a new variable that contains the value 1 for cars with above the median mpg, and 0 for other cars. Name this variable mpg01 Split the data into a test and training sets of size containing 50% and 50% of observations each.

```
library(MASS)
library(ISLR)
library(class)
m <- median(Auto$mpg)
Auto$mpg01 <- factor(ifelse(Auto$mpg <= m, 0, 1))
set.seed(1)
s <- sample(nrow(Auto), round(.5*nrow(Auto)))
Atrain <- Auto[s,]
Atest <- Auto[-s,]
```

(a) Plot the variables weight and acceleration using colour to show the two levels of mpg01 for the training set.

```
Atrain %>%
  ggplot(aes(weight, acceleration)) +
  geom_point(aes(colour = mpg01)) +
  theme_bw()
```



(b) Perform a linear discriminant analysis to predict mpg01, using variables weight and acceleration, on the training set. Use a plot to show the discriminant boundaries. What is the test error of the model obtained?

```
lda <- lda(mpg01 ~ weight + acceleration, data = Atrain)
lda
```

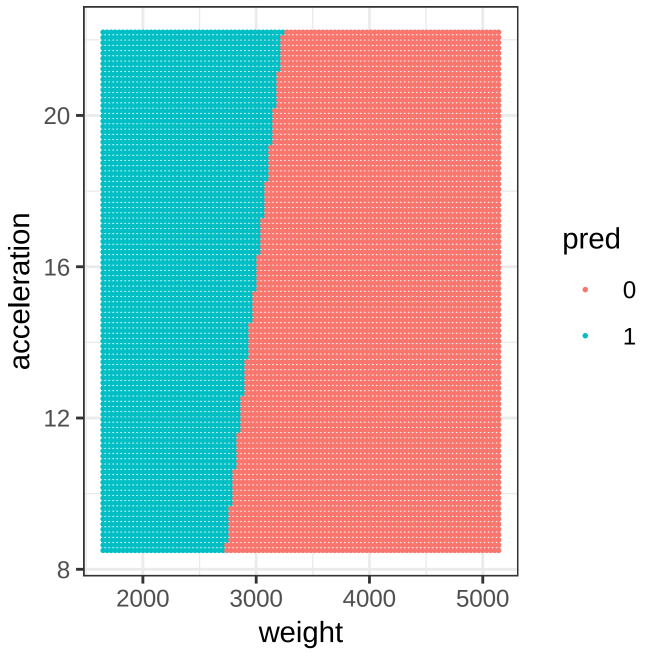
```
## Call:
## lda(mpg01 ~ weight + acceleration, data = Atrain)
##
## Prior probabilities of groups:
##         0         1
## 0.5255102 0.4744898
##
## Group means:
##     weight acceleration
## 0 3636.359     14.49806
## 1 2404.151     16.43226
##
## Coefficients of linear discriminants:
##                       LD1
## weight       -0.001635093
## acceleration  0.060260084
```

```
Atest$pred <- predict(lda, Atest)$class

Atest %>%
  group_by(pred, mpg01) %>%
  count() %>%
  ungroup() %>%
  mutate(perc = scales::percent(n/sum(n)))
```

```
## # A tibble: 4 x 4
##   pred  mpg01     n perc 
##   <fct> <fct> <int> <chr>
## 1 0     0        72 36.7%
## 2 0     1         4 2.0% 
## 3 1     0        21 10.7%
## 4 1     1        99 50.5%
```

```
# Error = 2 + 10.7 = 12.7%
```

```
grid <- expand.grid(
  weight = seq(min(Atrain$weight), max(Atrain$weight), length = 100),
  acceleration = seq(min(Atrain$acceleration), max(Atrain$acceleration),
                     length = 100)
)

grid$pred <- predict(lda, grid)$class
ggplot(aes(x = weight, y = acceleration, color = pred),
       data = grid) +
  geom_point(size=.3) +
  theme_bw()
```



(c) Repeat (b) using quadratic discriminant analysis. Which is better, LDA or QDA?

```
qda <- qda(mpg01 ~ weight + acceleration, data = Atrain)
qda
```

```
## Call:
## qda(mpg01 ~ weight + acceleration, data = Atrain)
## 
## Prior probabilities of groups:
```

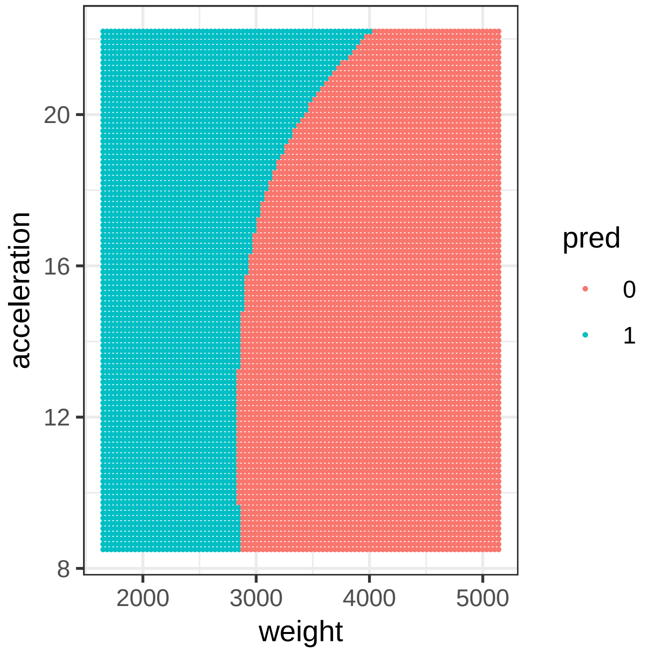
```
##         0         1
## 0.5255102 0.4744898
##
## Group means:
##     weight acceleration
## 0 3636.359     14.49806
## 1 2404.151     16.43226
```

```
Atest$pred <- predict(qda, Atest)$class

Atest %>%
  group_by(pred, mpg01) %>%
  count() %>%
  ungroup() %>%
  mutate(perc = scales::percent(n/sum(n)))
```

```
## # A tibble: 4 x 4
##   pred  mpg01     n perc
##   <fct> <fct> <int> <chr>
## 1 0     0        71 36.2%
## 2 0     1         2 1.0%
## 3 1     0        22 11.2%
## 4 1     1       101 51.5%
```

```
# Error = 1 + 12.2 = 12.2%
```

```
grid$pred <- predict(qda, grid)$class
ggplot(aes(x = weight, y = acceleration, color = pred),
       data = grid) +
  geom_point(size=.3) +
  theme_bw()
```



(d) Perform a linear discriminant analysis to predict mpg01, using variables displacement, horsepower, weight and acceleration on the training set. What is the test error of the model obtained?

```r
lda <- lda(mpg01 ~ displacement + horsepower +
           weight + acceleration, data = Atrain)
lda
```

```
## Call:
## lda(mpg01 ~ displacement + horsepower + weight + acceleration,
##     data = Atrain)
##
## Prior probabilities of groups:
##         0         1
## 0.5255102 0.4744898
##
## Group means:
##   displacement horsepower   weight acceleration
## 0     272.6214  131.69903 3636.359     14.49806
## 1     123.9140   80.11828 2404.151     16.43226
##
## Coefficients of linear discriminants:
##                        LD1
## displacement -0.0055437546
## horsepower   -0.0039716637
## weight       -0.0009141395
## acceleration  0.0086776698
```

```r
Atest$pred <- predict(lda, Atest)$class

Atest %>%
  group_by(pred, mpg01) %>%
  count() %>%
  ungroup() %>%
  mutate(perc = scales::percent(n/sum(n)))
```

```
## # A tibble: 3 x 4
##   pred  mpg01     n perc
##   <fct> <fct> <int> <chr>
## 1 0     0        73 37.2%
## 2 1     0        20 10.2%
## 3 1     1       103 52.6%
```

```r
# Error = 10.2%
```

(e) Repeat (d) using quadratic discriminant analysis. Which is better, LDA or QDA?

```r
qda <- qda(mpg01 ~ displacement + horsepower +
           weight + acceleration, data = Atrain)
qda
```

```
## Call:
```

```
## qda(mpg01 ~ displacement + horsepower + weight + acceleration,
##     data = Atrain)
##
## Prior probabilities of groups:
##         0         1
## 0.5255102 0.4744898
##
## Group means:
##   displacement horsepower   weight acceleration
## 0     272.6214  131.69903 3636.359     14.49806
## 1     123.9140   80.11828 2404.151     16.43226
```

```
Atest$pred <- predict(qda, Atest)$class

Atest %>%
  group_by(pred, mpg01) %>%
  count() %>%
  ungroup() %>%
  mutate(perc = scales::percent(n/sum(n)))
```

```
## # A tibble: 4 x 4
##   pred  mpg01     n perc
##   <fct> <fct> <int> <chr>
## 1 0     0        75 38.3%
## 2 0     1         6 3.1%
## 3 1     0        18 9.2%
## 4 1     1        97 49.5%
```

```
# Error = 3.1 + 9.2 = 12.3 %
```

(f) Perform KNN with response of mpg01, and the four predictors displacement, horsepower, weight and acceleration. Remember to scale the predictors. Use k = 5 and k = 30. Which value of k gives the best result on the test set?

```
scaled_train <-   Atrain %>%
    dplyr::select(displacement, horsepower, weight, acceleration) %>%
    mutate_all(scale)

scaled_test <-   Atest %>%
    dplyr::select(displacement, horsepower, weight, acceleration) %>%
    mutate_all(scale)

knn_5 <- knn(
  scaled_train,
  scaled_test,
  cl = Atrain$mpg01,
  k = 5)

knn_30 <- knn(
  scaled_train,
  scaled_test,
  cl = Atrain$mpg01,
```

```r
  k = 30)

table(Atest$mpg01, knn_5)
```

```
##    knn_5
##      0  1
##   0 81 12
##   1  7 96
```

```r
table(Atest$mpg01, knn_30)
```

```
##    knn_30
##      0  1
##   0 81 12
##   1  8 95
```

4. A classifier gives the following result. In the table below, Group gives the true class, and Prob gives the estimated probability of Group A (positive) using the classifier.

```r
groups <- data.frame(
  group = c(rep("A", each = 6), rep("B", each = 4)),
  p = c(0.206, 0.177, 0.687, 0.384, 0.770, 0.498, 0.718,
        0.992, 0.380, 0.777)
)
groups %>% knitr::kable()
```

| group | p |
|---|---:|
| A | 0.206 |
| A | 0.177 |
| A | 0.687 |
| A | 0.384 |
| A | 0.770 |
| A | 0.498 |
| B | 0.718 |
| B | 0.992 |
| B | 0.380 |
| B | 0.777 |

(a) What are the predicted classes? Use a threshold of 0.5.

```r
groups <- groups %>%
  mutate(pred = ifelse(p > 0.5, "A", "B"))
```

(b) What is the error rate? What is the false positive rate? The true positive rate?

```r
groups %>%
  group_by(pred, group) %>%
  count() %>%
  ungroup() %>%
  mutate(perc = scales::percent(n/sum(n)))
```

```
## # A tibble: 4 x 4
##   pred  group     n perc 
##   <chr> <fct> <int> <chr>
## 1 A     A         2 20.0%
## 2 A     B         3 30.0%
## 3 B     A         4 40.0%
## 4 B     B         1 10.0%
```

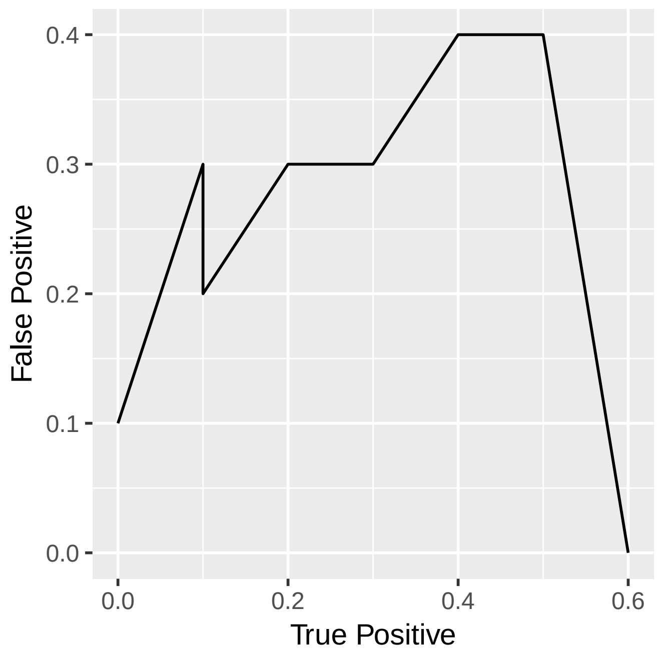
```r
# Error = 30 + 40 = 70%
```

(c) Now let the threshold take values 0, .2, .4,.6,.8,1. For each threshold calculate the false positive rate, and the true positive rate. (If doing this in R use more thresholds.)

```r
trs <- seq(0, 1, by = 0.05)
fc <- function(trs){
  res <- groups %>%
    mutate(pred = ifelse(p > trs, "A", "B"))
  tab <- c(prop.table(table(res$pred, res$group)))

  while(length(tab) < 4){
    tab <- c(tab, 0)
  }
  tab <- matrix(tab, ncol = 2, nrow = 2, byrow = TRUE)
  tp <- tab[1,1]
  fp <-  tab[2, 1]
  return(list(tp = tp, fp = fp))

}

res <-  trs %>% purrr::map(fc)
df <- data.frame(fp = res %>% map_dbl("fp"),
                 tp = res %>% map_dbl("tp"))
```

(d) Plot the true positive rate versus the false positive rate. This is the ROC curve.

```r
df %>%
  ggplot(aes(tp, fp)) +
  geom_line() +
  labs(y = "False Positive", x = "True Positive")
```



(e) (Optional, if doing in R) Another classifier just assigns class probabilities randomly, ie the estimated probabilities are: Plot the ROC curve for this classifier.

5. Dataset on diabetes in Pima Indian Women in library(MASS). For a description of the data see ?Pima.tr.

Use any supervised classification technique to predict diabetes from the 7 available features. Train your algorithms on Pima.tr and present the overall error rate for the test data Pima.te.

6. Generate some fake data using the following code:

```
set.seed(1)
x <- rnorm(100)
y <- 1 + .2*x+3*x^2+.6*x^3 + rnorm(100)
d <- data.frame(x = x, y = y)

d <- purrr::map(2:10,
                ~{ d$x^.x }
) %>% bind_cols() %>%
  bind_cols(d)

# or

# for(i in 2:10){
#   d[ , paste("var", i)] <- d$x^i
#
# }
```

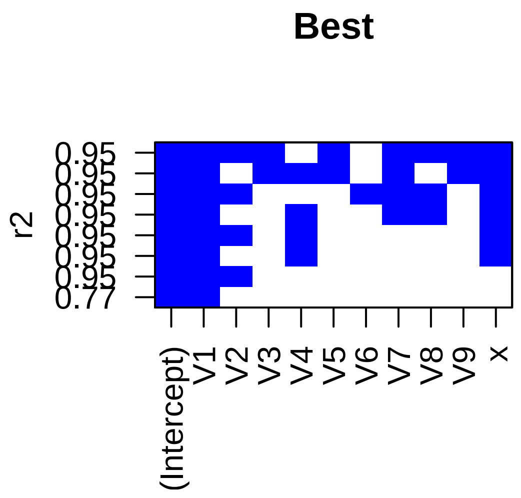
Use best subset selection to choose the best model containing predictors $X, X^2, \ldots, X^{10}$. Which terms are included in the best 3 variable model?

```
library(leaps)
allfits <- regsubsets(y ~ ., data = d)
summary(allfits)$which
```

```
##   (Intercept)   V1    V2    V3    V4    V5    V6    V7    V8    V9     x
## 1        TRUE TRUE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## 2        TRUE TRUE  TRUE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## 3        TRUE TRUE FALSE FALSE  TRUE FALSE FALSE FALSE FALSE FALSE  TRUE
## 4        TRUE TRUE  TRUE FALSE  TRUE FALSE FALSE FALSE FALSE FALSE  TRUE
## 5        TRUE TRUE FALSE FALSE  TRUE FALSE FALSE  TRUE  TRUE FALSE  TRUE
## 6        TRUE TRUE  TRUE FALSE FALSE FALSE  TRUE  TRUE  TRUE FALSE  TRUE
## 7        TRUE TRUE FALSE  TRUE  TRUE  TRUE FALSE  TRUE FALSE  TRUE  TRUE
## 8        TRUE TRUE  TRUE  TRUE FALSE  TRUE FALSE  TRUE  TRUE  TRUE  TRUE
```

(b) Make a plot of $C^p$ versus number of predictors for the models in all fits. Which model has the lowest $C^p$? What are its predictors?

```
par(mar = c(3,3,0,0))
plot(allfits, scale = "r2", col = "blue", main = "Best")
```

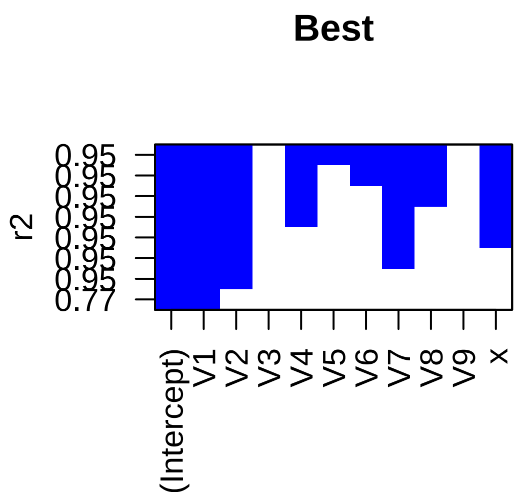


(c) Reconstruct all fits with option method = ”forward”. Which model has the lowest $C^p$? What are its predictors?

```
forw <- regsubsets(y ~ ., data = d, method="forward")
summary(forw)$which
```

```
##   (Intercept)   V1    V2    V3    V4    V5    V6    V7    V8    V9     x
## 1        TRUE TRUE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## 2        TRUE TRUE  TRUE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## 3        TRUE TRUE  TRUE FALSE FALSE FALSE FALSE  TRUE FALSE FALSE FALSE
## 4        TRUE TRUE  TRUE FALSE FALSE FALSE FALSE  TRUE FALSE FALSE  TRUE
## 5        TRUE TRUE  TRUE FALSE  TRUE FALSE FALSE  TRUE FALSE FALSE  TRUE
## 6        TRUE TRUE  TRUE FALSE  TRUE FALSE FALSE  TRUE  TRUE FALSE  TRUE
## 7        TRUE TRUE  TRUE FALSE  TRUE FALSE  TRUE  TRUE  TRUE FALSE  TRUE
## 8        TRUE TRUE  TRUE FALSE  TRUE  TRUE  TRUE  TRUE  TRUE FALSE  TRUE
```

```r
plot(forw, scale = "r2", col = "blue", main = "Best")
```

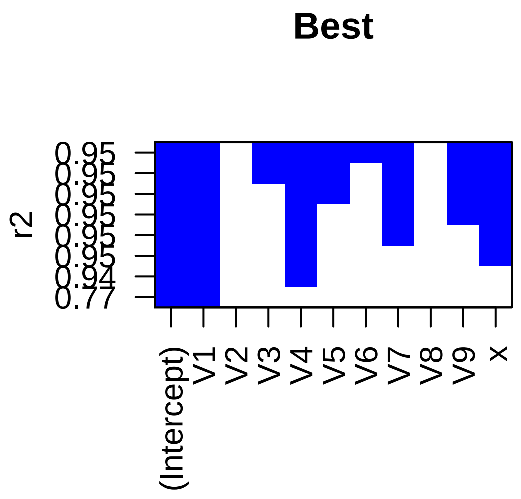


(d) Reconstruct allfits with option method = "backward". Which model has the lowest $C^p$? What are its predictors?

```r
back <- regsubsets(y ~ ., data = d, method="backward")
summary(back)$which
```

```
##   (Intercept)   V1    V2    V3    V4    V5    V6    V7    V8    V9     x
## 1        TRUE TRUE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE FALSE
## 2        TRUE TRUE FALSE FALSE  TRUE FALSE FALSE FALSE FALSE FALSE FALSE
## 3        TRUE TRUE FALSE FALSE  TRUE FALSE FALSE FALSE FALSE FALSE  TRUE
## 4        TRUE TRUE FALSE FALSE  TRUE FALSE FALSE  TRUE FALSE FALSE  TRUE
## 5        TRUE TRUE FALSE FALSE  TRUE FALSE FALSE  TRUE FALSE  TRUE  TRUE
## 6        TRUE TRUE FALSE FALSE  TRUE  TRUE FALSE  TRUE FALSE  TRUE  TRUE
## 7        TRUE TRUE FALSE  TRUE  TRUE  TRUE FALSE  TRUE FALSE  TRUE  TRUE
## 8        TRUE TRUE FALSE  TRUE  TRUE  TRUE  TRUE  TRUE FALSE  TRUE  TRUE
```

```r
plot(back, scale = "r2", col = "blue", main = "Best")
```


Best
r2
(Intercept)
V1
V2
V3
V4
V5
V6
V7
V8
V9
x