# Compulsory exercise 2: Group 24

TMA4268 Statistical Learning V2019

Silje Anfindsen and Clara Panchaud

19 March, 2020

## Problem 1

### a)

Let's find the ridge regression estimator. Remember that $\hat{\beta}_{Ridge}$ minimizes $RSS + \lambda \sum_{j=1}^{p} \beta_j^2$. Let's rewrite this in matrix notation.

$$
\begin{aligned}
min_{\beta}\{(y - X\beta)^T(y - X\beta) + \lambda\beta^T\beta\} = & \quad \text{develop the expression} \\
min_{\beta}\{y^Ty - 2\beta^TX^Ty + \beta^TX^TX\beta + \lambda\beta^T\beta\} & \quad \text{take the derivative with respect to beta and set equal to 0} \\
-2X^Ty + 2X^TX\beta + 2\lambda\beta = 0 & \\
(X^TX + 2\lambda I)\beta = X^Ty & \\
\beta = (X^TX + \lambda I)^{-1}X^Ty &
\end{aligned}
$$

Therefore the estimator is $\hat{\beta}_{Ridge} = (X^TX + \lambda I)^{-1}X^Ty$.

### b)

To find the expected value and the variance-covariance matrix of $\hat{\beta}_{Ridge}$ we need to remember the distribution of y, $y \sim N(X\beta, \sigma^2 I)$. Therefore we get the expected value:

$$E(\hat{\beta}_{Ridge}) = E((X^TX + \lambda I)^{-1}X^Ty) = (X^TX + \lambda I)^{-1}X^TE(y) = (X^TX + \lambda I)^{-1}X^TX\beta$$

and the variance-covariance matrix:

$$
\begin{aligned}
Var(\hat{\beta}_{Ridge}) = Var((X^TX + \lambda I)^{-1}X^Ty) = & \quad \text{by property of the variance} \\
(X^TX + \lambda I)^{-1}X^TVar(y)((X^TX + \lambda I)^{-1}X^T)^T = & \quad \text{develop the expression} \\
\sigma^2(X^TX + \lambda I)^{-1}X^TX(X^TX + \lambda I)^{-1} &
\end{aligned}
$$

### c)

TRUE, FALSE, FALSE, TRUE

## d)

```r
library(ISLR)
library(leaps)
library(glmnet)
```

We want to work with the *College* data. First we split it into a training and a testing set.

```r
set.seed(1)

#make training and testing set
train.ind = sample(1:nrow(College), 0.5 * nrow(College))
college.train = College[train.ind, ]
college.test = College[-train.ind, ]

#the structure of the data
str(College)
```

```
## 'data.frame':    777 obs. of  18 variables:
##  $ Private    : Factor w/ 2 levels "No","Yes": 2 2 2 2 2 2 2 2 2 2 ...
##  $ Apps       : num  1660 2186 1428 417 193 ...
##  $ Accept     : num  1232 1924 1097 349 146 ...
##  $ Enroll     : num  721 512 336 137 55 158 103 489 227 172 ...
##  $ Top10perc  : num  23 16 22 60 16 38 17 37 30 21 ...
##  $ Top25perc  : num  52 29 50 89 44 62 45 68 63 44 ...
##  $ F.Undergrad: num  2885 2683 1036 510 249 ...
##  $ P.Undergrad: num  537 1227 99 63 869 ...
##  $ Outstate   : num  7440 12280 11250 12960 7560 ...
##  $ Room.Board : num  3300 6450 3750 5450 4120 ...
##  $ Books      : num  450 750 400 450 800 500 500 450 300 660 ...
##  $ Personal   : num  2200 1500 1165 875 1500 ...
##  $ PhD        : num  70 29 53 92 76 67 90 89 79 40 ...
##  $ Terminal   : num  78 30 66 97 72 73 93 100 84 41 ...
##  $ S.F.Ratio  : num  18.1 12.2 12.9 7.7 11.9 9.4 11.5 13.7 11.3 11.5 ...
##  $ perc.alumni: num  12 16 30 37 2 11 26 37 23 15 ...
##  $ Expend     : num  7041 10527 8735 19016 10922 ...
##  $ Grad.Rate  : num  60 56 54 59 15 55 63 73 80 52 ...
```

Now we will apply forward selection, using *Outstate* as a response. We have 18 variables including the response so we will obtain a model including up to 17 variables.

```r
nb_predictors<-17
forward<-regsubsets(Outstate~.,college.train,nvmax=17,method="forward")
sum<-summary(forward)
```

In Figure 1 we can look at the RSS and the adjusted $R^2$ in order to pick the number of variables that gives the optimal result. Remember that if the difference is not very significant we would rather pick the simplest model. It seems like 5 variables would be good here.

```r
par(mfrow=c(1,2))
plot(sum$rss,xlab="Number of Variables",ylab="RSS",type="l")
plot(sum$adjr2,xlab="Number of Variables",ylab="Adjusted RSq",type="l")
```

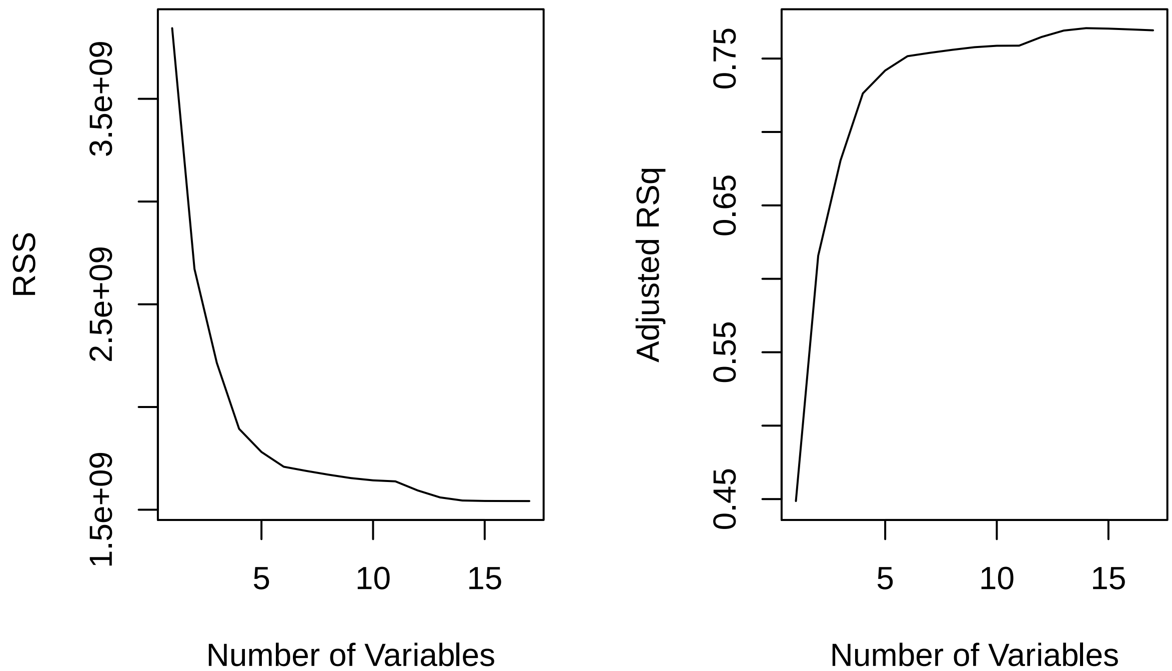



Figure 1: Comparison of models with different number of variables.

Below are the chosen variables when we decide to include 5 variables in the reduced model.

```r
nb_selected_pred<-5
variables<-names( coef( forward,id=nb_selected_pred ) )
variables
```

```
## [1] "(Intercept)" "PrivateYes"  "Room.Board"  "perc.alumni" "Expend"
## [6] "Grad.Rate"
```

We will now find the reduced model as well as the MSE (mean squared error) on the test set.

```r
#fit the reduced model
reduced.model<-lm(Outstate~Private+Room.Board+Grad.Rate+perc.alumni+Expend, data =college.train)
summary(reduced.model)
```

```
##
## Call:
```

```
## lm(formula = Outstate ~ Private + Room.Board + Grad.Rate + perc.alumni +
##     Expend, data = college.train)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -7293.1 -1537.5  -159.9  1286.7  9254.8
##
## Coefficients:
##               Estimate Std. Error t value Pr(>|t|)
## (Intercept) -2.711e+03  5.155e+02  -5.259 2.41e-07 ***
## PrivateYes   2.250e+03  2.787e+02   8.075 8.92e-15 ***
## Room.Board   1.241e+00  1.205e-01  10.296  < 2e-16 ***
## Grad.Rate    3.855e+01  7.850e+00   4.910 1.35e-06 ***
## perc.alumni  6.446e+01  1.113e+01   5.792 1.45e-08 ***
## Expend       2.182e-01  2.317e-02   9.417  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 2159 on 382 degrees of freedom
## Multiple R-squared:  0.7452, Adjusted R-squared:  0.7419
## F-statistic: 223.4 on 5 and 382 DF,  p-value: < 2.2e-16
```

The reduced model is

$$\begin{aligned}\textbf{Outstate} = &-2711.4329907 + 2250.1100562\textbf{Private} + 1.2410466\textbf{Room.Board}\\ &+ 38.5491289\textbf{Grad.Rate} + 64.4580901\textbf{perc.alumni} + 0.218216\textbf{Expend},\end{aligned}$$

```
#find test MSE
p<-predict(reduced.model,newdata=college.test)
error1 <- mean(((college.test$Outstate)-p)^2)
error1
```

```
## [1] 4010675
```

The test MSE is

$$MSE = \frac{1}{n}\sum_{i=1}^{n}(y_i - f(\hat{x_i}))^2 = 4.0106747 \times 10^6$$

### e)

We will now select a model for the same dataset as in (d) but this time with the Lasso method. Again, we use both a training and testing set for the data.

```
#Make a x matrix and y vector for both the training and testing set
x_train<-model.matrix(Outstate~.,college.train)[,-1]
y_train<-college.train$Outstate
x_test<-model.matrix(Outstate~.,college.test)[,-1]
y_test<-college.test$Outstate
```

In order to select the best value for the tuning parameter $\lambda$ we will use cross validation.

```r
set.seed(5555)

#perform the Lasso method and choose the best model using CV
lasso.mod = glmnet(x_train,y_train,alpha=1) #lasso method on train set
cv.lasso = cv.glmnet(x_train,y_train,alpha=1) #CV on train set
lambda.best = cv.lasso$lambda.min #select best lambda
lambda.best
```

```
## [1] 5.093201
```

```r
#find the test MSE
predictions<-predict(lasso.mod,s=lambda.best,newx=x_test)
error2 <- mean((predictions-y_test)^2) #test MSE
error2
```

```
## [1] 3717020
```

%%Check if log scale!! From cross validation we can observe that the optimal tuning parameter is $\lambda = 5.0932009$ as this is the parameter that minimizes the MSE for the training set.

The test MSE is now $3.7170197 \times 10^6$, which is lower than what we found for the reduced model using forward selection in d).

The lasso yields sparse models which involves only a subset of variables. Lasso performs variable selection by forcing some of the coefficient estimates to be exactly zero. The selected variables that was not put to zero are displayed below.

```r
c<-coef(lasso.mod,s=lambda.best,exact=TRUE)
inds<-which(c!=0)
variables<-row.names(c)[inds]
variables
```

```
##  [1] "(Intercept)" "PrivateYes"  "Apps"        "Accept"      "Enroll"
##  [6] "Top10perc"   "Top25perc"   "F.Undergrad" "P.Undergrad" "Room.Board"
## [11] "Books"       "Personal"    "PhD"         "Terminal"    "S.F.Ratio"
## [16] "perc.alumni" "Expend"      "Grad.Rate"
```

# Problem 2

## a)

FALSE, FALSE, TRUE, FALSE

## b)

The basis functions for a cubic spline with knots at each quartile, of variable $X$ are,

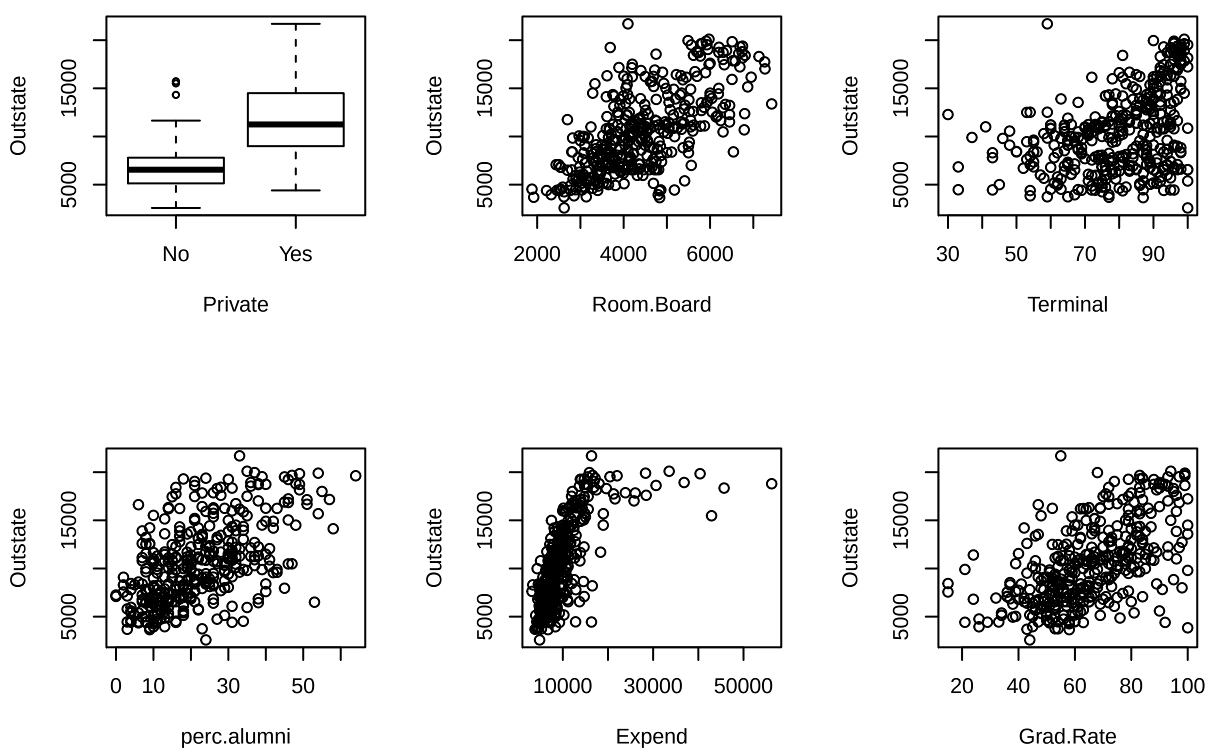
$$
\begin{aligned}
b_0(X) &= 1 & b_4(X) &= (X - q_1)^3_+ \\
b_1(X) &= x & b_5(x) &= (X - q_2)^3_+ \\
b_2(X) &= x^2 & b_6(X) &= (X - q_3)^3_+ \\
b_3(X) &= x^3
\end{aligned}
$$

## c)

We will now investigate the realtionship between *Outstate* and the 6 of the predictors, *Private*, *Room.Board*, *Terminal*, *perc.alumni*, *Expend*, and *Grad.Rate*.

```
ds1 = college.train[c("Private", "Outstate")] #binary variable
ds2 = college.train[c("Room.Board", "Outstate")]
ds3 = college.train[c("Terminal", "Outstate")]
ds4 = college.train[c("perc.alumni", "Outstate")]
ds5 = college.train[c("Expend", "Outstate")]
ds6 = college.train[c("Grad.Rate", "Outstate")]

par(mfrow=c(2,3))
plot(ds1)
plot(ds2)
plot(ds3)
plot(ds4)
plot(ds5)
plot(ds6)
```




From each of the plots above we can conclude that at least *Terminal* and *Expend* seems to have a non-linear relationship with *Outstate*. These two variables therefore might benefit from a non-linear transformation. The others variables, *Room.Board*, *perc.alumni* and *Grad.Rate* seem to have a linear realtionship with the response variable. Check out the binary variable *Private*???

## d)

We will now fit several polynomial regression models for *Outstate* with *Terminal* as the only covariate. Each polynomial will have a degree from $d = 1, ..10$.

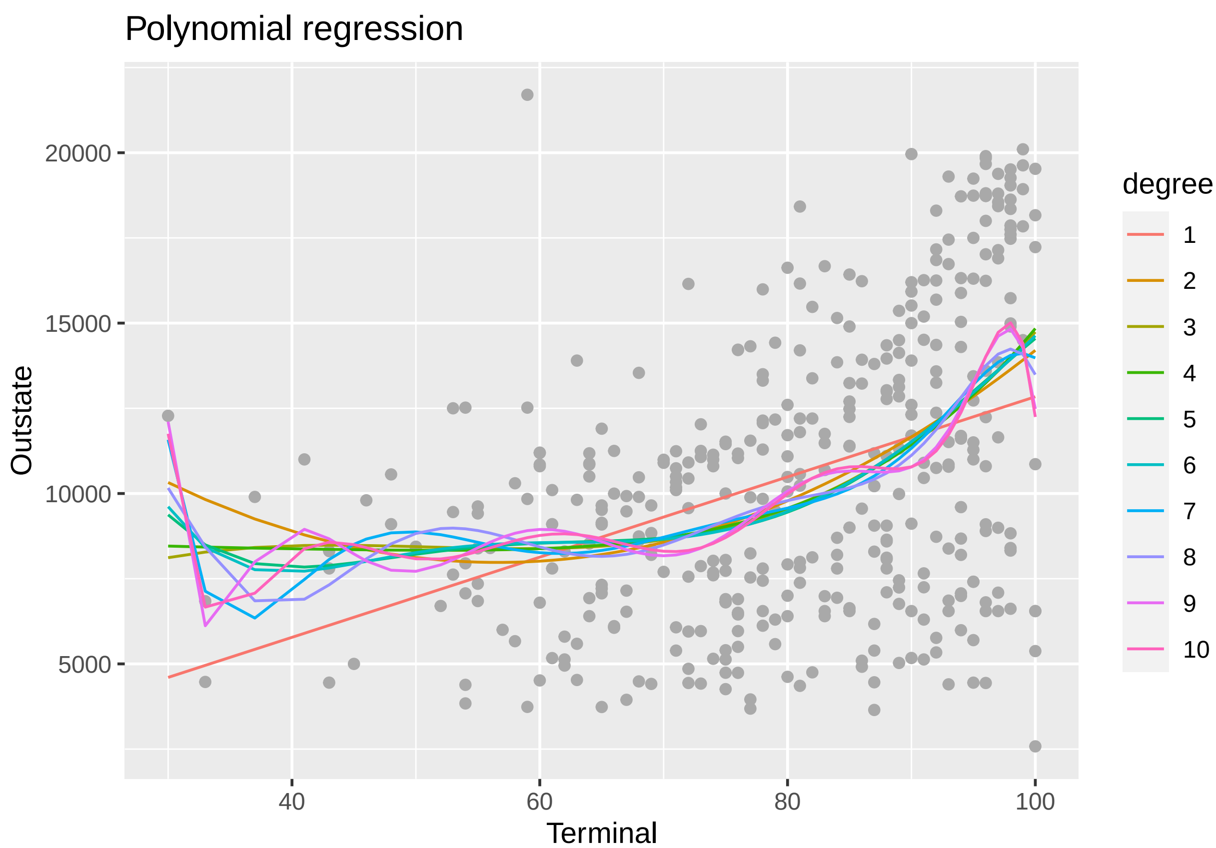
```
library(ggplot2)

#make a dataframe
ds = College[c("Terminal", "Outstate")]
n = nrow(ds)

# chosen degrees
deg = 1:10

#now iterate over each degree d
dat = c()  #make a empty variable to store predicted values for each degree
MSE = list(rep(0, 10))  #make a empty variable to store MSE for each degree
for (d in deg) {
    # fit model with this degree
    mod = lm(Outstate ~ poly(Terminal, d), ds[train.ind, ])
    #datagram for Terminal and Outstate showing result for each degree over all samples
    dat = rbind(dat, data.frame(Terminal = ds[train.ind, 1], Outstate = mod$fit,
                               degree = as.factor(rep(d,length(mod$fit)))))
    # training MSE
    MSE[d] = mean((predict(mod, ds[-train.ind, ]) - ds[-train.ind, 2])^2)
}

# plot fitted values for different degrees
ggplot(data = ds[train.ind, ], aes(x = Terminal, y = Outstate)) +
    geom_point(color = "darkgrey") + labs(title = "Polynomial regression")+
    geom_line(data = dat, aes(x = Terminal, y = Outstate, color = degree))
```

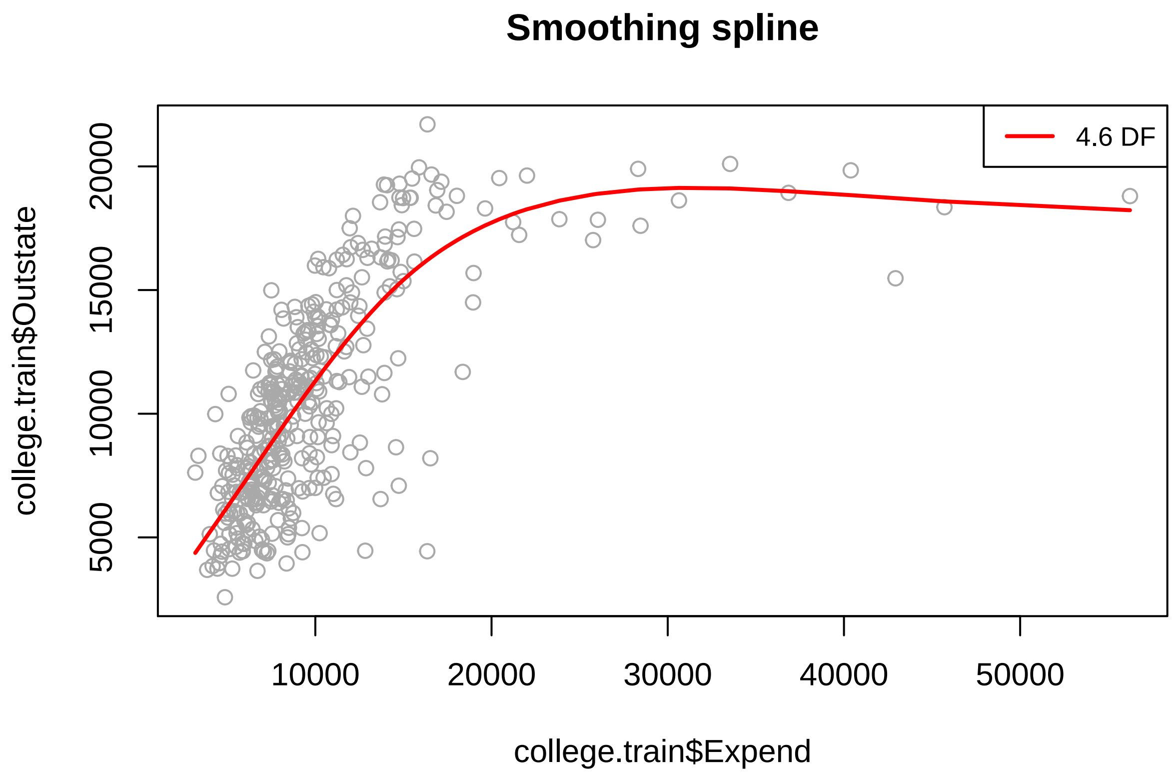



```r
library(splines)

plot(college.train$Expend, college.train$Outstate, col = "darkgrey", main="Smoothing spline")
fit = smooth.spline(college.train$Expend, college.train$Outstate, cv=TRUE)
fit$df #choose df from CV
```

```
## [1] 4.660711
```

```r
lines(fit, col="red", lwd=2)
legend("topright", legend=c("4.6 DF"), col="red", lty=1, lwd=2, cex=.8)
```




We found the degrees of freedom by doing cross validation.

```
#MSE for polynomial regression models (1-10)
MSE
```

```
##  [1] 11892400 11103379 10937428 10936549 11005681 11105882 16429293 10914136
##  [9] 81326931 24310822
```

```
#MSE for smoothing spline
?predict.smooth.spline
pred = predict(fit, newdata=college.train)
#MSE2 = mean( (college.train$Outstate - pred)^2 )
#MSE2
```

## Problem 3

### a)

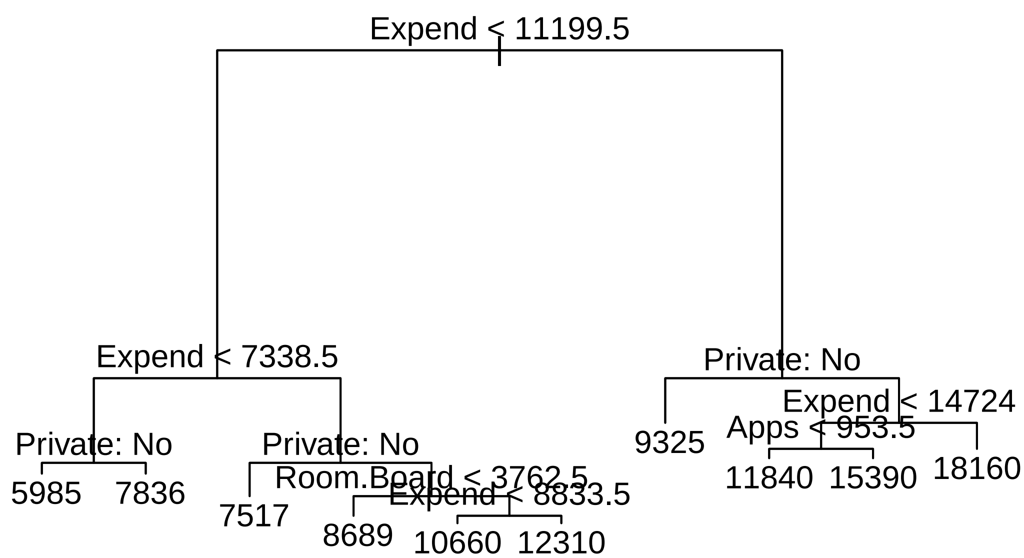
False, Not sure ( usually True but not for large trees?), TRUE, FALSE

### b)

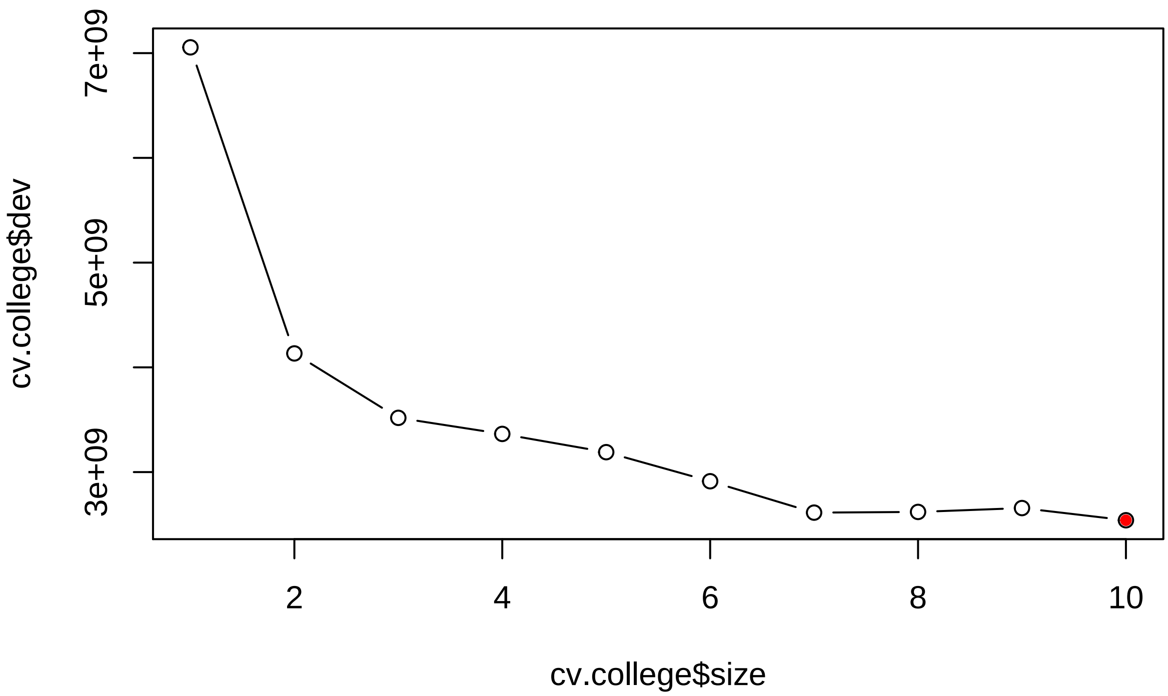
```r
library(tree)
tree.mod = tree(Outstate ~ .,college.train)
summary(tree.mod)
```

```
##
## Regression tree:
## tree(formula = Outstate ~ ., data = college.train)
## Variables actually used in tree construction:
## [1] "Expend"     "Private"    "Room.Board" "Apps"
## Number of terminal nodes:  10
## Residual mean deviance:  3971000 = 1.501e+09 / 378
## Distribution of residuals:
##     Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
## -7594.00 -1236.00    71.59     0.00  1263.00  6803.00
```

```r
plot(tree.mod)
text(tree.mod, pretty = 0)
```

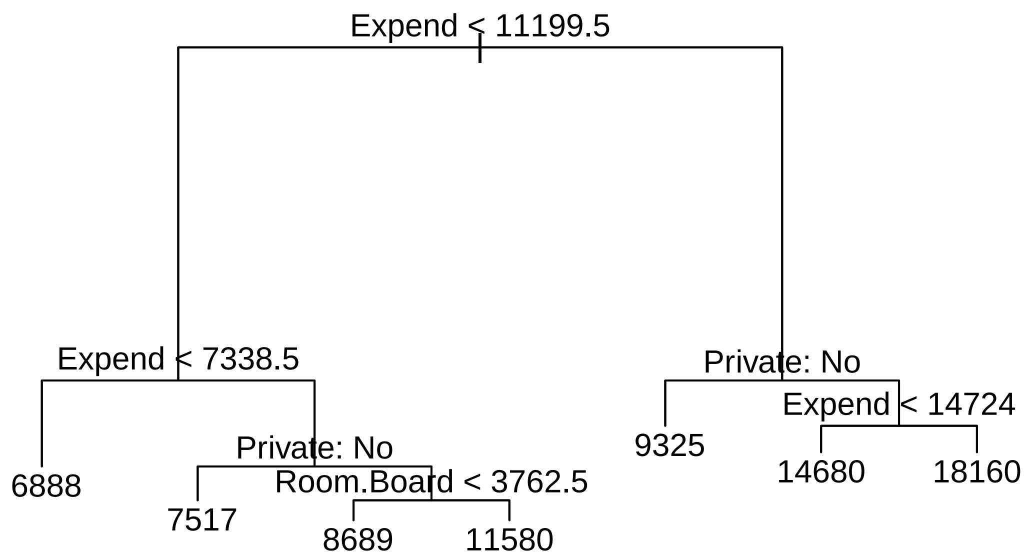



```r
set.seed(4268)
cv.college = cv.tree(tree.mod)
tree.min = which.min(cv.college$dev)
best = cv.college$size[tree.min]
plot(cv.college$size, cv.college$dev, type = "b")
points(cv.college$size[tree.min], cv.college$dev[tree.min], col = "red", pch = 20)
```




We see that 7 is on the same level of deviance as for size 10, since this is the smallest number for the size we pick 7. Let us now prune the tree to make it size 7.

```
pr.tree = prune.tree(tree.mod, best = 7)
plot(pr.tree)
text(pr.tree, pretty = 0)
```


Expend < 11199.5
Expend < 7338.5
Private: No
Expend < 14724
Private: No
Room.Board < 3762.5
6888
7517
8689
11580
9325
14680
18160

### c)

## Problem 4

### a)

### b)

### c)

### d)

## Problem 5

### a)

### b)

### c)

### d)

### e)

### f)

## References

James, G., D. Witten, T. Hastie, and R. Tibshirani. 2013. An Introduction to Statistical Learning with Applications in R. New York: Springer.