HW3: Chapters 6 and 7

Marshall Tuck

8/2/2020

1. Chapter 6 Problem 1 - page 259 ISLR

- Three models using - forward stepwise, backwards stepwise, and best subset
  a. Smallest training RSS: Best subset
  b. Smallest test RSS: Validate using cross-validation, low training RSS does not guarantee low test RSS
  c.

  - True - k-parameter model using forward stepwise is a subset of k+1 parameter model using forward stepwise
  - True - k-parameter model using backwards stepwise is a subset of k+1 parameter model using backward stepwise
  - False - k-parameter model using backwards stepwise is *not* a subset of k+1 parameter model using forward stepwise.
  - False - k-parameter model using forwards stepwise is *not* a subset of k+1 parameter model using backwards stepwise.
  - False - k-parameter model using best subset is *not* a subset of k+1 parameter model using best subsets.

2. Chapter 6 Problem 3 a and b - page 260 ISLR

- Suppose we estimate the regression coefficients by minimizing an equation of RSS subject to sum of beta(j)<= s:
  a. As we increase s from 0, the training RSS will

  iv. Steadily decrease

  b. As we increase s from 0, the test RSS will

  ii. Decrease initially, then eventually start increasing in a U shape

3. Chapter 6 Problem 9, page 263 ISLR

a. Split data in training and test set

```
library(ISLR)

#Sample
set.seed(42)
trainindex<- sample(1:nrow(College),.8*nrow(College))

#Set aside 80 for training/20 for testing
college_train<- College[trainindex,]
college_test<- College[-trainindex,]
```

b. Fit linear model using least squares, report test error

```r
#Fit linear model, predict the test values
collegefit_linear<- lm(Apps~., data=college_train)
collegefit_app_predict<- predict(collegefit_linear, college_test)

#Mean squared error
mean((college_test$Apps-collegefit_app_predict)^2)
```

```
## [1] 1941715
```

Test MSE for Linear fit model, fit on training data, tested on test data, is 1941715.

c. Fit ridge regression on training data set, with lambda selected by cross validation. Report on test error.

```r
library(glmnet)
```

```
## Loading required package: Matrix

## Loaded glmnet 4.0-2
```

```r
# X model matrix and Y response
train.x<- model.matrix(Apps~., college_train)[,-1]
test.x<- model.matrix(Apps~., college_test)[,-1]

#Grid of possible lambdas
grid<- 10^seq(10,-2, length=100)

#Fit ridge model
collegefit_ridge<- glmnet(train.x,college_train$Apps,alpha=0, lambda=grid)

#Cross Validation function, output lambda associated with smallest train error
cv.out<- cv.glmnet(train.x,college_train$Apps,alpha=0)
bestlam<- cv.out$lambda.min
bestlam
```

```
## [1] 337.0816
```

```r
#Error Prediction

ridge.pred<- predict(collegefit_ridge, newx=test.x, s=bestlam)
mean((college_test$Apps-ridge.pred)^2)
```

```
## [1] 3830755
```

Train MSE for Ridge Regression Model, fit on training data, tested on test data, is 3830755, higher than the linear model. Our lambda value that minimizes training error is 337.0816.

d. Fit lasso model on training set, lambda chosen by cross validation, report on test error, along with number of non-zero coefficients

```r
#Fit Lasso model
collegefit_lasso<- glmnet(train.x,college_train$Apps, alpha=1, lambda=grid)

#Perform cross validation and output lambda that minimizes training error
cv.lasso<- cv.glmnet(train.x,college_train$Apps,alpha=1, lambda=grid)
bestlam.lasso<-cv.lasso$lambda.min
bestlam.lasso
```

```
## [1] 10.72267
```

```r
#Predict test values and return MSE
lasso.pred<- predict(collegefit_lasso, newx=test.x, s=bestlam.lasso)
mean((college_test$Apps-lasso.pred)^2)
```

```
## [1] 2053507
```

```r
#Output coefficients
predict(collegefit_lasso, type="coefficients", s=bestlam.lasso)
```

```
## 18 x 1 sparse Matrix of class "dgCMatrix"
##                        1
## (Intercept) -873.26931182
## PrivateYes  -541.43590228
## Accept         1.22693617
## Enroll         .
## Top10perc     40.86153716
## Top25perc     -9.49399088
## F.Undergrad    0.05301432
## P.Undergrad    0.02848991
## Outstate      -0.03734291
## Room.Board     0.19257958
## Books          .
## Personal       .
## PhD           -5.80372418
## Terminal      -5.62427604
## S.F.Ratio     15.41080695
## perc.alumni   -3.93523422
## Expend         0.07801583
## Grad.Rate      7.72816434
```

Test MSE for Lasso Model, fit on training data, tested on test data, is 2051567 - less than the Ridge Model. Lambda that minimizes training error is 10.72. There are 15 non-zero coefficients.

e. Fit a PCR model on the data set, with M chosen by cross validation. Report test error, along with value of M.

```r
library(pls)
```

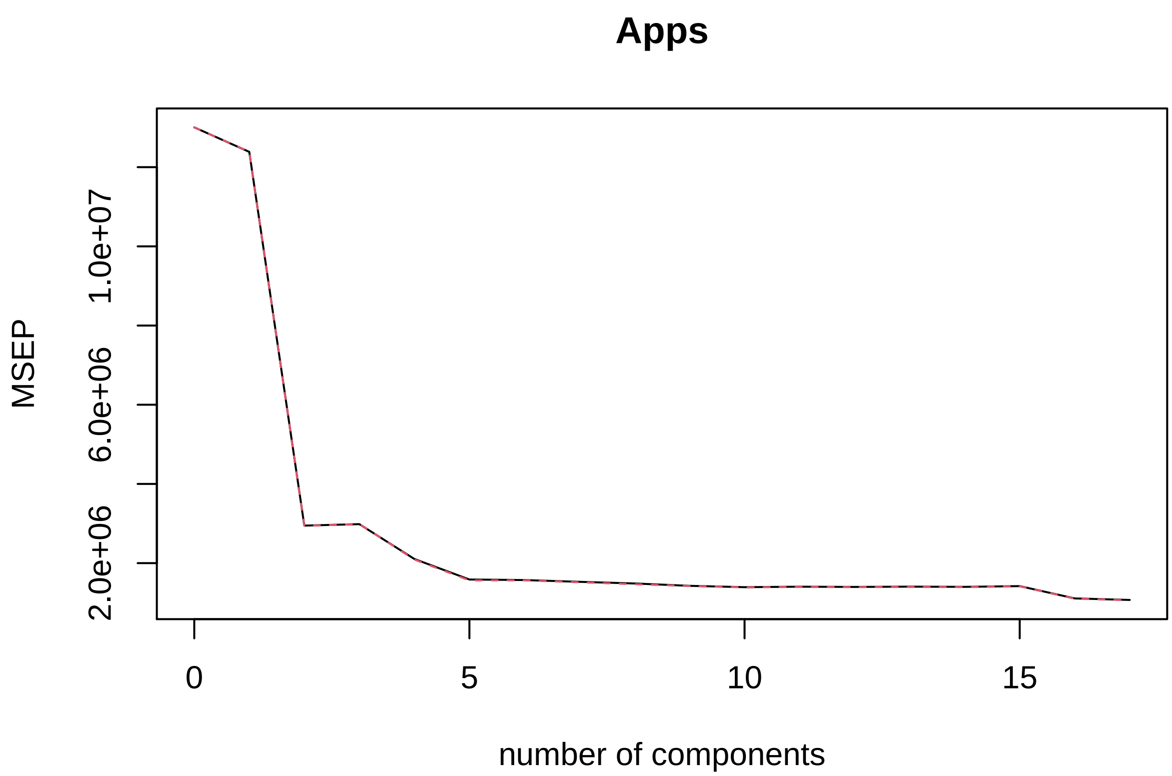
```
##
## Attaching package: 'pls'
```

```
## The following object is masked from 'package:stats':
##
##     loadings
```

```r
#Fit PCR Model
pcr.fit<- pcr(Apps~., data=college_train,scale=TRUE, validation="CV")

#Output plot of M
#We find an M value that minimizes MSE to be 17
summary(pcr.fit)
```

```
## Data:    X dimension: 621 17
##  Y dimension: 621 1
## Fit method: svdpc
## Number of components considered: 17
##
## VALIDATION: RMSEP
## Cross-validated using 10 random segments.
##        (Intercept)  1 comps  2 comps  3 comps  4 comps  5 comps  6 comps
## CV            3606     3519     1717     1728     1451     1259     1254
## adjCV         3606     3520     1714     1729     1446     1251     1251
##        7 comps  8 comps  9 comps  10 comps  11 comps  12 comps  13 comps
## CV        1236     1219     1194      1180      1185      1183      1186
## adjCV     1231     1212     1192      1178      1183      1180      1183
##        14 comps  15 comps  16 comps  17 comps
## CV         1183      1191      1053      1034
## adjCV      1181      1189      1050      1031
##
## TRAINING: % variance explained
##       1 comps  2 comps  3 comps  4 comps  5 comps  6 comps  7 comps  8 comps
## X      31.872    57.63    64.85    70.51    75.83    80.81    84.41    87.82
## Apps    6.168    77.92    77.92    84.76    88.39    88.39    88.95    89.44
##       9 comps  10 comps  11 comps  12 comps  13 comps  14 comps  15 comps
## X       90.84     93.24     95.34     97.11     98.14     98.92     99.44
## Apps    89.66     89.91     89.97     90.07     90.08     90.12     90.13
##       16 comps  17 comps
## X        99.82    100.00
## Apps     92.34     92.71
```

```r
validationplot(pcr.fit, val.type = "MSEP")
```




```
#Predict test values using model, calculate error
predict.pcr<- predict(pcr.fit, college_test, ncomp=17)
mean((college_test$Apps-predict.pcr)^2)
```

```
## [1] 1941715
```

Test MSE for PCR Model is 1941714 - equal to the linear model, with an optimal value of 17 for M.

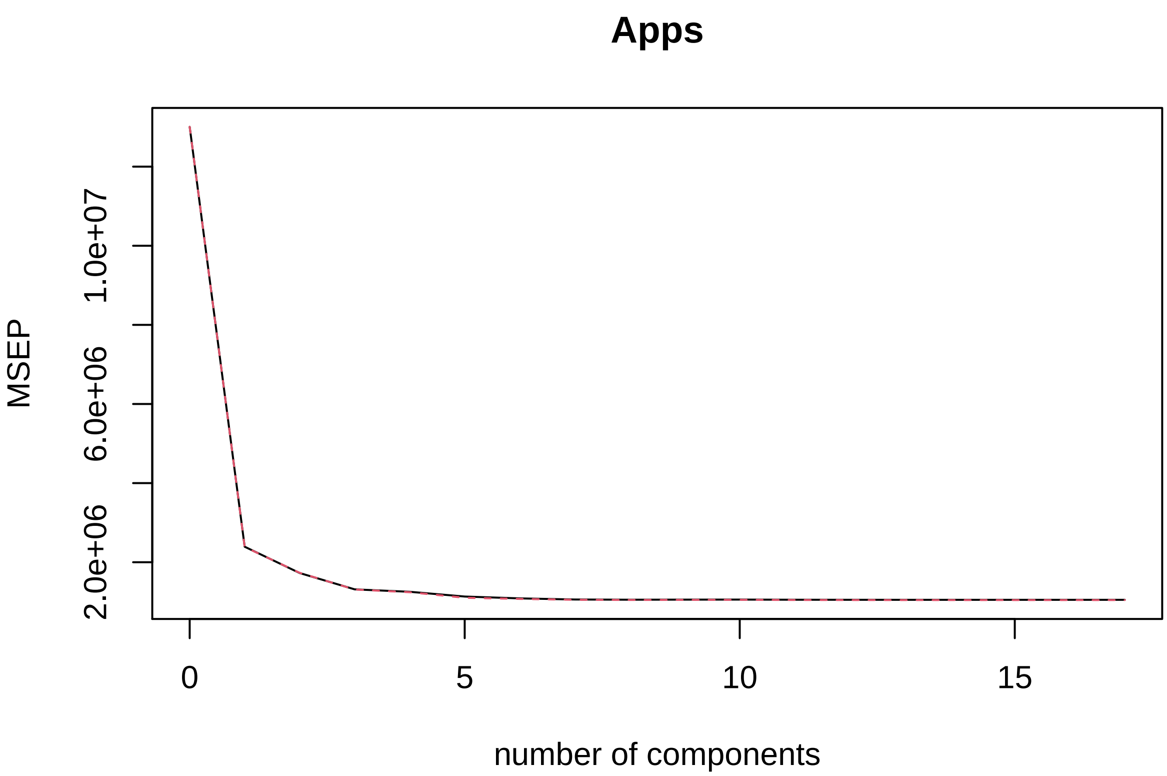
f. Fit a PLS model on the training set, with M chosen by cross validation, Report on Test error and M.

```
#Fit Partial Least Squares Model
plsr.fit<- plsr(Apps~., data=college_train, scale=TRUE, validation="CV")

# We see a minimum train error at M=11
summary(plsr.fit)
```

```
## Data:    X dimension: 621 17
##  Y dimension: 621 1
## Fit method: kernelpls
## Number of components considered: 17
##
## VALIDATION: RMSEP
## Cross-validated using 10 random segments.
##        (Intercept)  1 comps  2 comps  3 comps  4 comps  5 comps  6 comps
## CV            3606     1547     1314     1146     1119     1064     1042
## adjCV         3606     1545     1316     1145     1114     1050     1036
##        7 comps  8 comps  9 comps  10 comps  11 comps  12 comps  13 comps
```

```
## CV         1029     1026     1027      1028      1026      1025      1025
## adjCV      1025     1023     1024      1025      1023      1022      1022
##        14 comps  15 comps  16 comps  17 comps
## CV         1025      1025      1025      1025
## adjCV      1022      1022      1022      1022
##
## TRAINING: % variance explained
##        1 comps  2 comps  3 comps  4 comps  5 comps  6 comps  7 comps  8 comps
## X        26.34    46.91    62.99    66.01    68.24    71.98    75.40    81.19
## Apps     82.05    87.06    90.33    91.21    92.20    92.54    92.63    92.64
##        9 comps  10 comps  11 comps  12 comps  13 comps  14 comps  15 comps
## X        83.51     85.23     87.18     88.80     91.34     93.31     97.11
## Apps     92.67     92.69     92.70     92.71     92.71     92.71     92.71
##        16 comps  17 comps
## X         99.34    100.00
## Apps      92.71     92.71
```

```r
validationplot(plsr.fit, val.type = "MSEP")
```




```r
#Predict test values
predict.plsr<- predict(plsr.fit, college_test, ncomp=11)
mean((college_test$Apps-predict.plsr)^2)
```

```
## [1] 1944806
```

Test MSE for PLSR Model is 1944806, slightly larger than the PCR Model and the linear model. M was found to be 11 to minimize CV value.

g. Comment on the results obtained. How accurately can we predict the number of college applications received? Is there much difference in error among the 5 approaches?

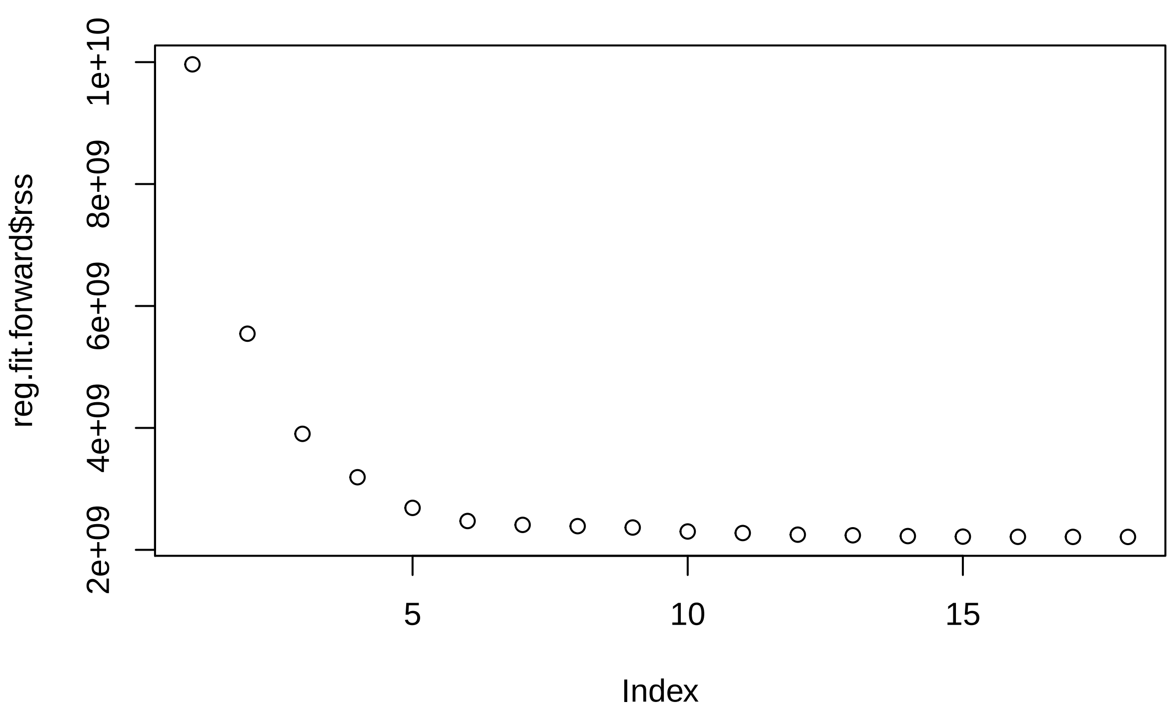
- The model with the lowest test MSE is the Linear Model and the Partial Least Squares Model indicating more predictive ability, with test MSE of 1941715.
- The model with the largest test MSE indicating less predictive ability is Ridge Regression model = 3830755.

4. Chapter 7 Problem 10 b, c, and d - page 300 ISLR
   b. Fit a Gam on College training data set, using out of state-tuition as the response and features forward subset regressors.

```
library(leaps)
library(gam)
```

```
## Warning: package 'gam' was built under R version 4.0.2

## Loading required package: splines

## Loading required package: foreach

## Loaded gam 1.20
```

```
#Use reg subsets to find the forward selection steps
reg.fit.forward<- regsubsets(Outstate~., data=college_train, nvmax=18, method="forward")

#Plot the forward selected RSS, find a good model has 7 regressors
plot(reg.fit.forward$rss)
```

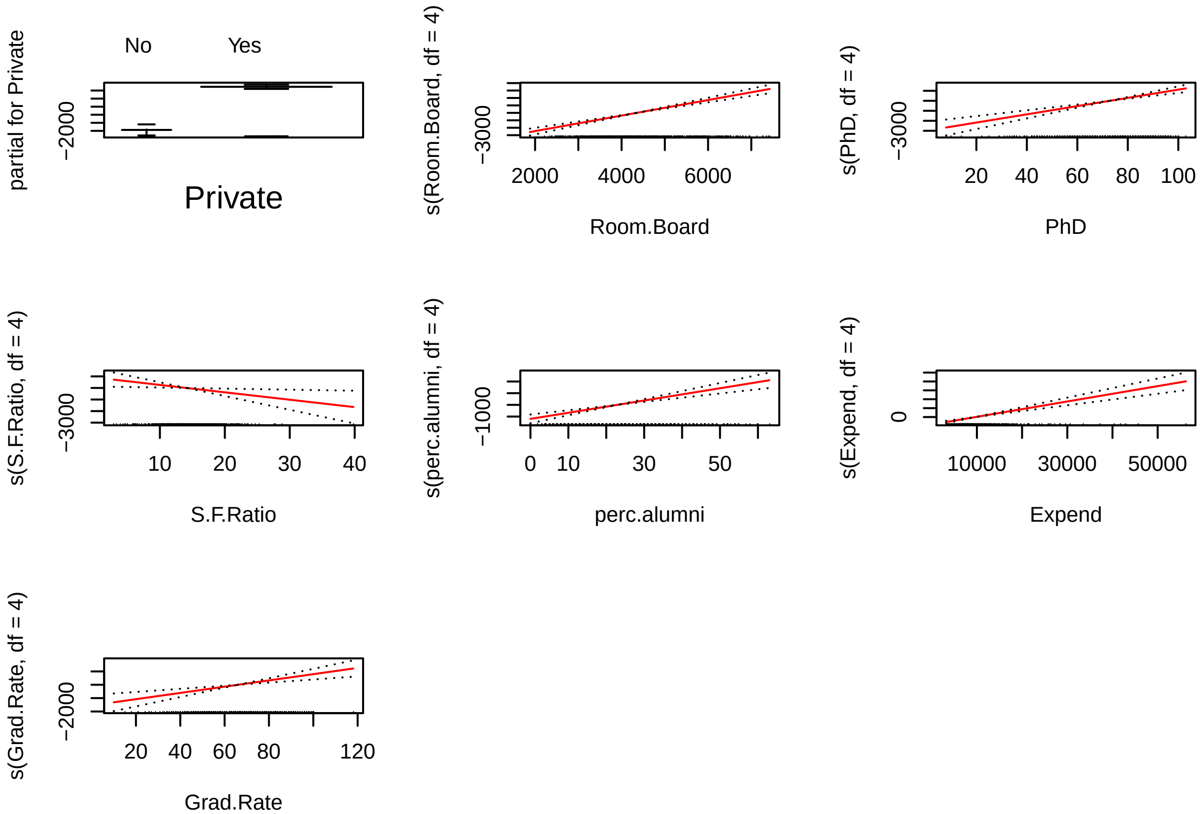


```r
#Output the 7 included regressors PrivateYes, RoomBoard, PhD, S.F.Ratio, perc.alumni, Expend, Grad.Rate
summary(reg.fit.forward)
```

```
## Subset selection object
## Call: regsubsets.formula(Outstate ~ ., data = college_train, nvmax = 18,
##     method = "forward")
## 17 Variables  (and intercept)
##             Forced in Forced out
## PrivateYes      FALSE      FALSE
## Apps            FALSE      FALSE
## Accept          FALSE      FALSE
## Enroll          FALSE      FALSE
## Top10perc       FALSE      FALSE
## Top25perc       FALSE      FALSE
## F.Undergrad     FALSE      FALSE
## P.Undergrad     FALSE      FALSE
## Room.Board      FALSE      FALSE
## Books           FALSE      FALSE
## Personal        FALSE      FALSE
## PhD             FALSE      FALSE
## Terminal        FALSE      FALSE
## S.F.Ratio       FALSE      FALSE
## perc.alumni     FALSE      FALSE
## Expend          FALSE      FALSE
## Grad.Rate       FALSE      FALSE
## 1 subsets of each size up to 17
## Selection Algorithm: forward
##           PrivateYes Apps Accept Enroll Top10perc Top25perc F.Undergrad
## 1  ( 1 )  " "        " "  " "    " "    " "       " "       " "
## 2  ( 1 )  " "        " "  " "    " "    " "       " "       " "
## 3  ( 1 )  " "        " "  " "    " "    " "       " "       " "
## 4  ( 1 )  "*"        " "  " "    " "    " "       " "       " "
## 5  ( 1 )  "*"        " "  " "    " "    " "       " "       " "
## 6  ( 1 )  "*"        " "  " "    " "    " "       " "       " "
## 7  ( 1 )  "*"        " "  " "    " "    " "       " "       " "
## 8  ( 1 )  "*"        " "  "*"    " "    " "       " "       " "
## 9  ( 1 )  "*"        " "  "*"    "*"    " "       " "       " "
## 10  ( 1 ) "*"        " "  "*"    "*"    "*"       " "       " "
## 11  ( 1 ) "*"        "*"  "*"    "*"    "*"       " "       " "
## 12  ( 1 ) "*"        "*"  "*"    "*"    "*"       " "       " "
## 13  ( 1 ) "*"        "*"  "*"    "*"    "*"       " "       " "
## 14  ( 1 ) "*"        "*"  "*"    "*"    "*"       " "       "*"
## 15  ( 1 ) "*"        "*"  "*"    "*"    "*"       " "       "*"
## 16  ( 1 ) "*"        "*"  "*"    "*"    "*"       "*"       "*"
## 17  ( 1 ) "*"        "*"  "*"    "*"    "*"       "*"       "*"
##           P.Undergrad Room.Board Books Personal PhD Terminal S.F.Ratio
## 1  ( 1 )  " "         "*"        " "   " "      " " " "      " "
## 2  ( 1 )  " "         "*"        " "   " "      " " " "      " "
## 3  ( 1 )  " "         "*"        " "   " "      " " " "      " "
## 4  ( 1 )  " "         "*"        " "   " "      " " " "      " "
## 5  ( 1 )  " "         "*"        " "   " "      "*" " "      " "
## 6  ( 1 )  " "         "*"        " "   " "      "*" " "      " "
## 7  ( 1 )  " "         "*"        " "   " "      "*" " "      "*"
```

```
## 8  ( 1 )  " "          "*"         " "   " "      "*" " "       "*"
## 9  ( 1 )  " "          "*"         " "   " "      "*" " "       "*"
## 10  ( 1 ) " "          "*"         " "   " "      "*" " "       "*"
## 11  ( 1 ) " "          "*"         " "   " "      "*" " "       "*"
## 12  ( 1 ) " "          "*"         "*"   " "      "*" " "       "*"
## 13  ( 1 ) " "          "*"         "*"   " "      "*" "*"       "*"
## 14  ( 1 ) " "          "*"         "*"   " "      "*" "*"       "*"
## 15  ( 1 ) " "          "*"         "*"   "*"      "*" "*"       "*"
## 16  ( 1 ) " "          "*"         "*"   "*"      "*" "*"       "*"
## 17  ( 1 ) "*"          "*"         "*"   "*"      "*" "*"       "*"
##           perc.alumni Expend Grad.Rate
## 1  ( 1 )  " "         " "    " "
## 2  ( 1 )  "*"         " "    " "
## 3  ( 1 )  "*"         "*"    " "
## 4  ( 1 )  "*"         "*"    " "
## 5  ( 1 )  "*"         "*"    " "
## 6  ( 1 )  "*"         "*"    "*"
## 7  ( 1 )  "*"         "*"    "*"
## 8  ( 1 )  "*"         "*"    "*"
## 9  ( 1 )  "*"         "*"    "*"
## 10  ( 1 ) "*"         "*"    "*"
## 11  ( 1 ) "*"         "*"    "*"
## 12  ( 1 ) "*"         "*"    "*"
## 13  ( 1 ) "*"         "*"    "*"
## 14  ( 1 ) "*"         "*"    "*"
## 15  ( 1 ) "*"         "*"    "*"
## 16  ( 1 ) "*"         "*"    "*"
## 17  ( 1 ) "*"         "*"    "*"
```

```r
#Fit GAM model
gam.fit<- lm(Outstate~Private+s(Room.Board, df=4)+s(PhD, df=4)+s(S.F.Ratio, df=4)+s(perc.alumni, df=4)+s

par(mfrow=c(3,3))
plot.Gam(gam.fit, se=TRUE, col="red")
```



From our 7 included variables, we see negative relationships between our response Outstate and S.F. Ratio, and positive relationships for all other included variables.
c. Evaluate the model on the test set

```
gam.pred<-predict(gam.fit, college_test)
mean((college_test$Outstate-gam.pred)^2)
```

```
## [1] 5014766
```

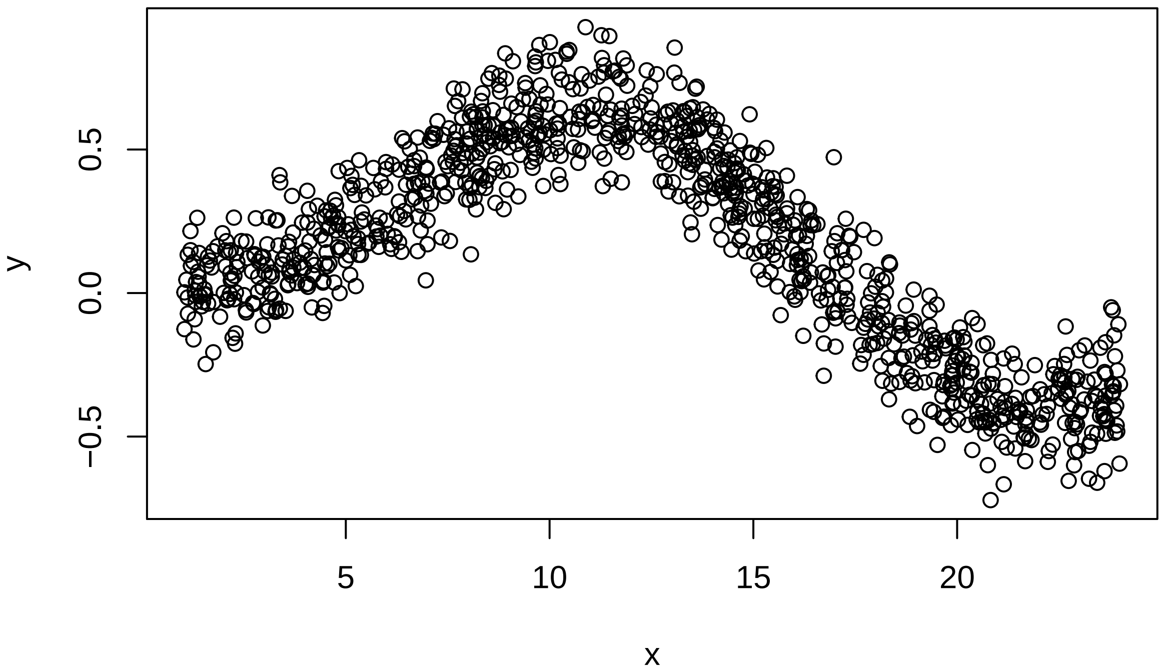
We see a MSE of 5014766 when evaluated on our test set.

```
d. For which variables is there evidence of non-linear relationship?
```

I do not see evidence of a non-linear relationship in the middle of the observed values of our data. Any non-linearity will be at the tails based on the standard errors given. This indicates that most likely the effect is linear and any non-linearity may not be significant if it exists at all.

5. Read in the dataset below. Take a random sample of 400 as the test set.

```
#Set seed
set.seed(42)

#Read in data, output plot
df3<- read.table("hump1000.csv", header=TRUE, sep=',')
plot(df3$x, df3$y, xlab="x", ylab="y")
```




```r
testindex<-sample(1:nrow(df3), 400)
df3_test<-df3[testindex,]
df3_train<-df3[-testindex,]
```

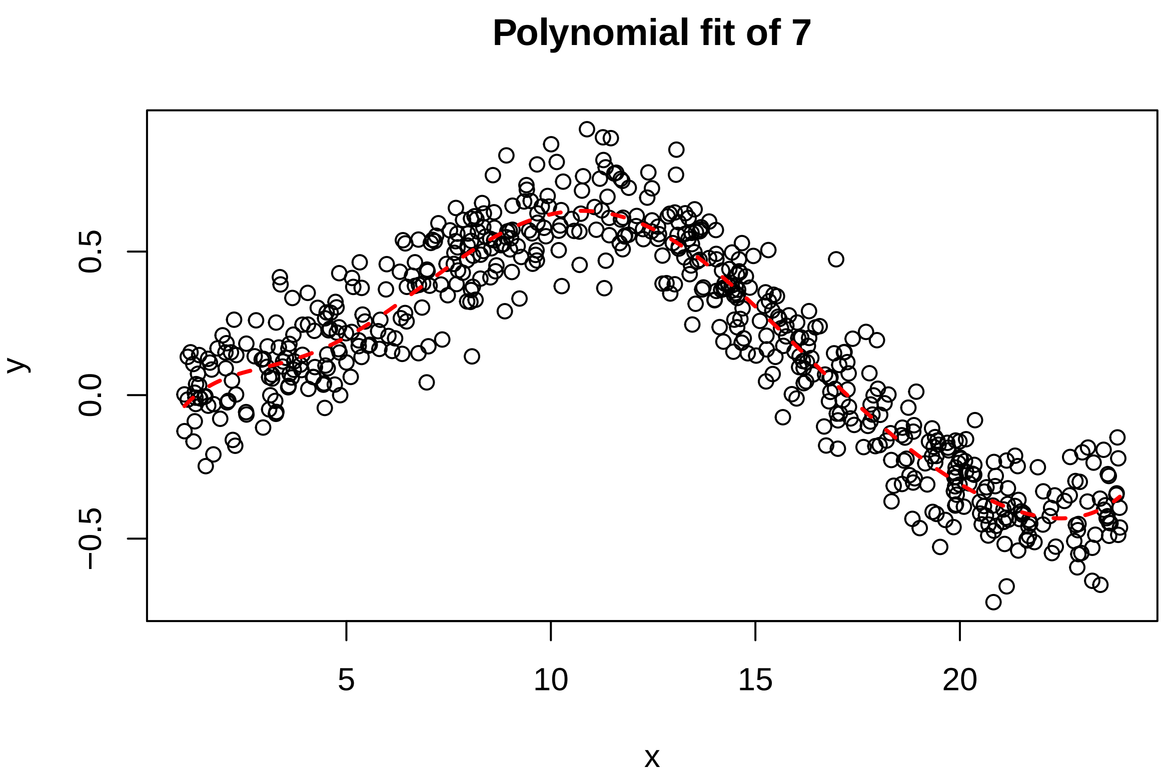
a. Use R to fit a polynomial model to this data. Plot the data and fitted model. What is estimated RMSE on the test data?

```r
#Fit Polynomial to the eight to see where significance ends, ends at power of 7
# That will be our chosen model (to the seventh power)
polynom<- lm(y~poly(x,8), data=df3_train)
summary(polynom)
```

```
##
## Call:
## lm(formula = y ~ poly(x, 8), data = df3_train)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -0.36125 -0.07460 -0.00232  0.08363  0.44465
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.154596   0.004877  31.701  < 2e-16 ***
## poly(x, 8)1 -4.748795   0.119453 -39.755  < 2e-16 ***
## poly(x, 8)2 -6.317080   0.119453 -52.883  < 2e-16 ***
## poly(x, 8)3  1.719236   0.119453  14.393  < 2e-16 ***
## poly(x, 8)4  2.421154   0.119453  20.269  < 2e-16 ***
## poly(x, 8)5 -0.280770   0.119453  -2.350 0.019078 *
## poly(x, 8)6 -0.441518   0.119453  -3.696 0.000239 ***
## poly(x, 8)7  0.323517   0.119453   2.708 0.006958 **
## poly(x, 8)8  0.167786   0.119453   1.405 0.160660
```

```
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.1195 on 591 degrees of freedom
## Multiple R-squared:  0.8947, Adjusted R-squared:  0.8933
## F-statistic: 627.9 on 8 and 591 DF,  p-value: < 2.2e-16
```

```
best_poly<- lm(y~poly(x,7), data=df3_train)

#Make some predictions
pred_df<-data.frame(x=seq(min(df3_train$x), max(df3_train$x),by=.01))

#Fit the best poly model to those predictions
pred_poly_train<-predict(best_poly, newdata=pred_df, se=TRUE)


#Plot the raw data and the predicted values of x and their corresponding fits.
plot(df3_train)
lines(pred_df$x, pred_poly_train$fit, col='red', lty=2, lwd=2)
title (" Polynomial fit of 7 ")
```




```
#Predict test values
pred_test<- predict(best_poly, newdata=df3_test)
#Output RMSE
sqrt(mean((df3_test$y-pred_test)^2))
```

```
## [1] 0.1283048
```

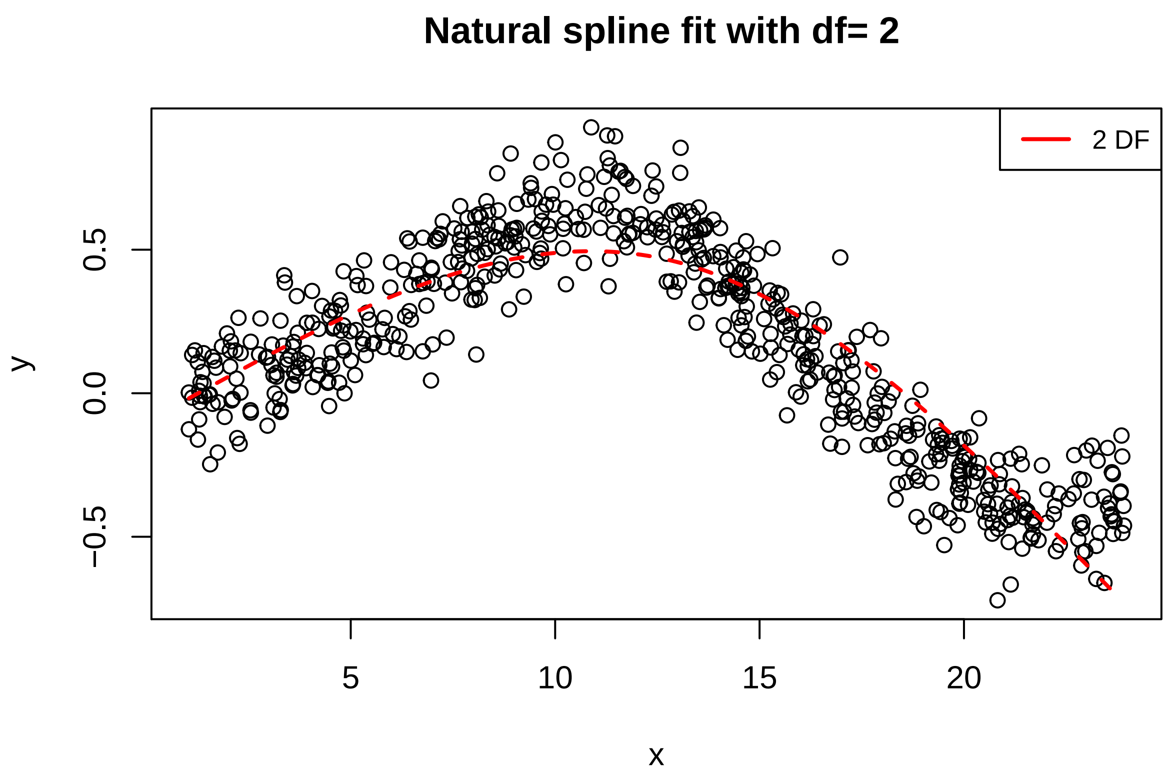
The RMSE of the fitted model on the test data is .1283.

b. Use R to fit a natural spline, consider models up to 10 knots. What is knot number that gives the best fit to the training data set? What is RMSE?

```
library(tidyverse)
```

```
## -- Attaching packages ------------------------------------------------------------------------------------

## v ggplot2 3.3.2     v purrr   0.3.4
## v tibble  3.0.1     v dplyr   1.0.0
## v tidyr   1.1.0     v stringr 1.4.0
## v readr   1.3.1     v forcats 0.5.0

## -- Conflicts ---------------------------------------------------------------------------------------------
## x purrr::accumulate() masks foreach::accumulate()
## x tidyr::expand()     masks Matrix::expand()
## x dplyr::filter()     masks stats::filter()
## x dplyr::lag()        masks stats::lag()
## x tidyr::pack()       masks Matrix::pack()
## x tidyr::unpack()     masks Matrix::unpack()
## x purrr::when()       masks foreach::when()
```

```
#Fit spline model to training data set using df=2
spline_fit<- lm(y~ns(x, df=2), data=df3_train)
#Predict output from predicted x values above
spline_fit_df<- predict(spline_fit, newdata=pred_df)

#Plot outcome
plot(df3_train)
lines(pred_df$x, spline_fit_df, col='red', lty=2, lwd=2)
title (" Natural spline fit with df= 2")
legend("topright",legend=c("2 DF"),
        col=c("red"),lty=1,lwd=2,cex=.8)
```



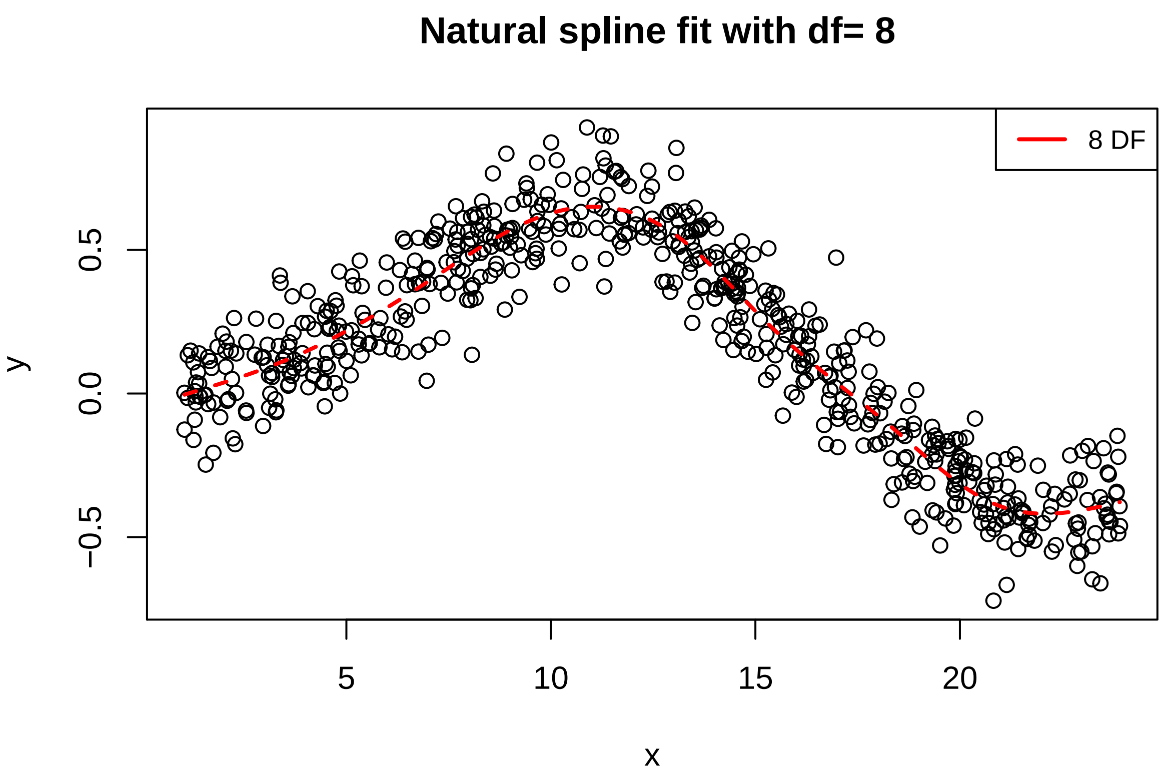

```
#Write function to find smallest RMSE from knot count 2 to 10
errors<-tibble()
for (i in 2:10){
  knot.count<-i
  spline_fit_knot<- lm(y~ns(x, df=c(i)), data=df3_train)
  pred_spline<-predict(spline_fit_knot, newdata=df3_test)
  RMSE<-sqrt(mean((df3_test$y-pred_spline)^2))%>%as_tibble()
  error.i<-tibble(RMSE, knot.count)
  errors<-errors%>%
    bind_rows(error.i)
}

errors
```

```
## # A tibble: 9 x 2
##   value knot.count
##   <dbl>      <int>
## 1 0.178          2
## 2 0.160          3
## 3 0.130          4
## 4 0.129          5
## 5 0.128          6
## 6 0.129          7
## 7 0.128          8
## 8 0.129          9
## 9 0.129         10
```

```r
#Return min RMSE and corresponding knot amount
errors%>%
  filter(value==min(value))
```

```
## # A tibble: 1 x 2
##   value knot.count
##   <dbl>      <int>
## 1 0.128          8
```

```r
#Fit spline model
spline_fit_knot<- lm(y~ns(x, df=c(8)), data=df3_train)
spline_fit_knot_df<- predict(spline_fit_knot, newdata = pred_df)

#Plot values
plot(df3_train)
lines(pred_df$x, predict(spline_fit_knot, newdata=pred_df), col='red', lty=2, lwd=2)
title (" Natural spline fit with df= 8")
legend("topright",legend=c("8 DF"),
         col=c("red"),lty=1,lwd=2,cex=.8)
```



It appears a model with df=8 is the best fit to the data, represented by the red dotted line. The RMSE of our 8 knots model against our test data is .1283. This corresponds to our RMSE from using a polynomial fit to the eighth power.

c. Use R to fit a local regression model. Consider a range of spans for the loess call. Estimate the span needed to give the correct model. What is RMSE for fitted model?

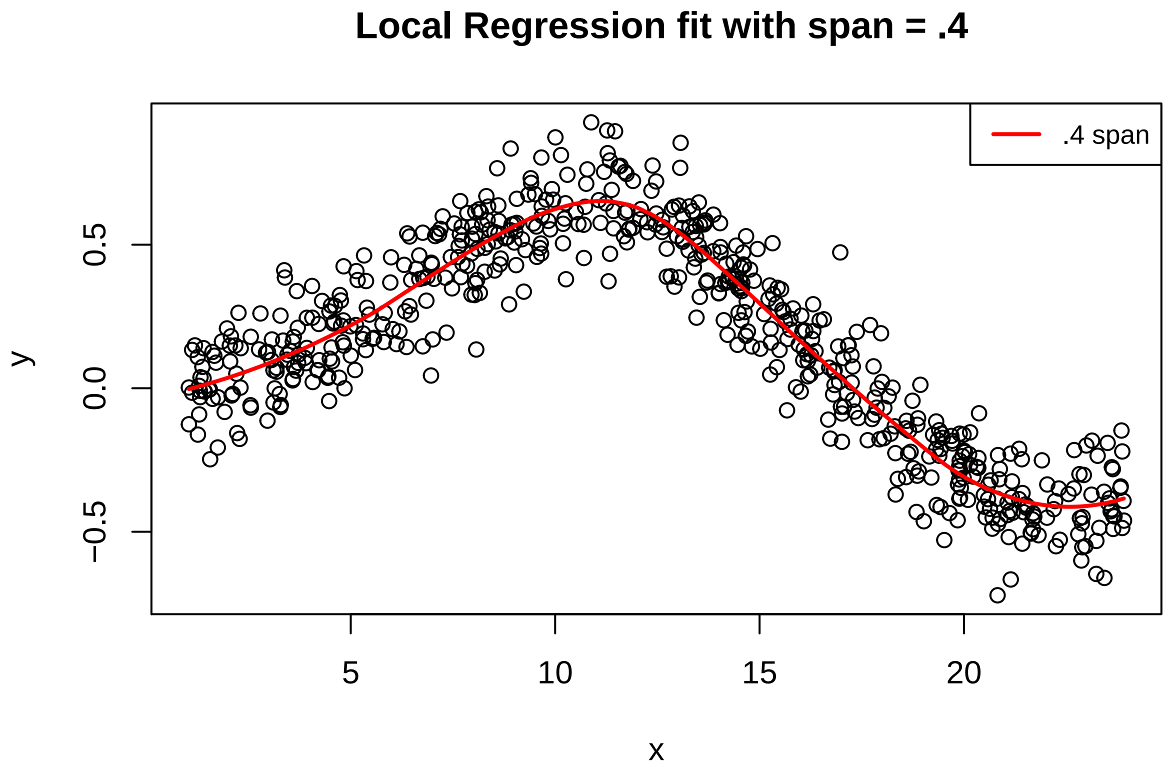
```r
#Write function to find smallest RMSE from span of .1 to 10
errors<-tibble()
for (i in 1:100){
  span<-i/10
  loess_fit<- loess(y~x,span=span,data=df3_train, control=loess.control(surface="direct"))
  pred_loess<-predict(loess_fit, newdata=df3_test)
  RMSE<-sqrt(mean((df3_test$y-pred_loess)^2))%>%as_tibble()
  error.i<-tibble(RMSE, span)
  errors<-errors%>%
    bind_rows(error.i)
}

errors
```

```
## # A tibble: 100 x 2
##    value  span
##    <dbl> <dbl>
##  1 0.130   0.1
##  2 0.129   0.2
##  3 0.129   0.3
##  4 0.129   0.4
##  5 0.129   0.5
##  6 0.131   0.6
##  7 0.135   0.7
##  8 0.146   0.8
##  9 0.156   0.9
## 10 0.166   1
## # ... with 90 more rows
```

```r
#Return span amount associated with smallest test RMSE
errors%>%
  filter(value==min(value))
```

```
## # A tibble: 1 x 2
##   value  span
##   <dbl> <dbl>
## 1 0.129   0.4
```

```r
#Set local regression models with optimum span values
loess_fit<- loess(y~x,span=.4,data=df3_train, control=loess.control(surface="direct"))


#Plot fit
plot(df3_train)
lines(pred_df$x,predict(loess_fit, newdata= pred_df), col="red",lwd=2)
title (" Local Regression fit with span = .4")
legend("topright",legend=c(".4 span"),
         col=c("red"),lty=1,lwd=2,cex=.8)
```



A span=.4, corresponding to the red line, is the span value that best fits the data. It has an RMSE of .1286 compared to the testing data, slightly greater than the spline and Polynomial Model RMSE.