HW 4

Elizabeth Hutton

September 20, 2018

1 7.9 Applied

a. Cubic Polynomial Regression of *Nox* vs. *Dis*

```
> library(MASS)
> attach(Boston)
> set.seed(1)
> #cubic polynomial regression
> fit = lm(nox~poly(dis,3), data = Boston)
> summary(fit)

Call:
lm(formula = nox ~ poly(dis, 3), data = Boston)

Residuals:
      Min        1Q    Median        3Q       Max
-0.121130 -0.040619 -0.009738  0.023385  0.194904

Coefficients:
               Estimate Std. Error t value Pr(>|t|)
(Intercept)    0.554695   0.002759 201.021  < 2e-16 ***
poly(dis, 3)1 -2.003096   0.062071 -32.271  < 2e-16 ***
poly(dis, 3)2  0.856330   0.062071  13.796  < 2e-16 ***
poly(dis, 3)3 -0.318049   0.062071  -5.124 4.27e-07 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 0.06207 on 502 degrees of freedom
Multiple R-squared:  0.7148,	Adjusted R-squared:  0.7131
F-statistic: 419.3 on 3 and 502 DF,  p-value: < 2.2e-16

> #make predictions for values of dis within range(dis)
> lims = range(dis)
> dis.grid = seq(from=lims[1],to=lims[2])
> preds=predict(fit,newdata=list(dis=dis.grid),se=TRUE)
```

```
> #get standard error bands
> se.bands=cbind(preds$fit+2*preds$se.fit,preds$fit-2*preds$se.fit)
> #plot data and regression line w/ error bands
> plot(dis,nox,xlim=lims ,cex=.5,col="darkgrey")
> lines(dis.grid,preds$fit,lwd=2,col="blue")
> matlines(dis.grid,se.bands,lwd=1,col="blue",lty=3)
>
```

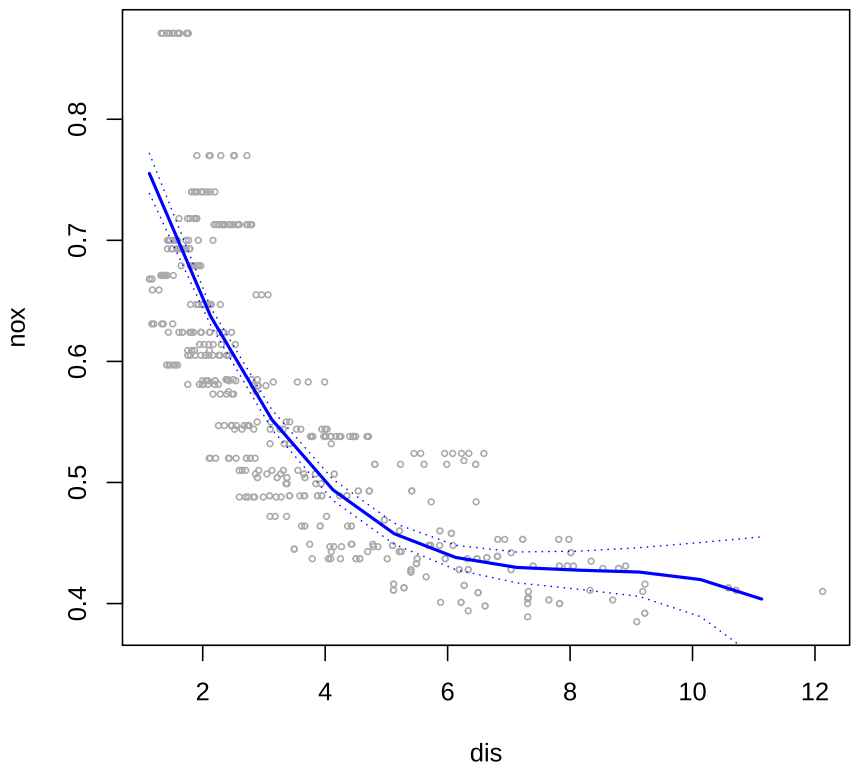


Figure 1: Cubic Regression Line for Nox vs. Distance

b. Regression polynomials degrees 1 through 10

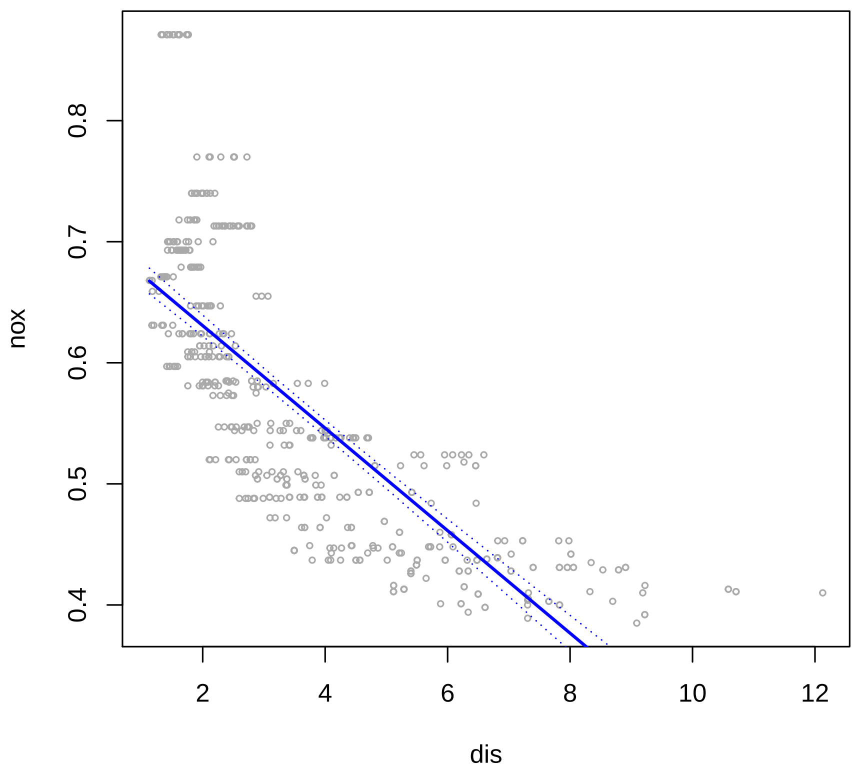


Figure 2: Degree 1 Polynomial Fit RSS = 2.769

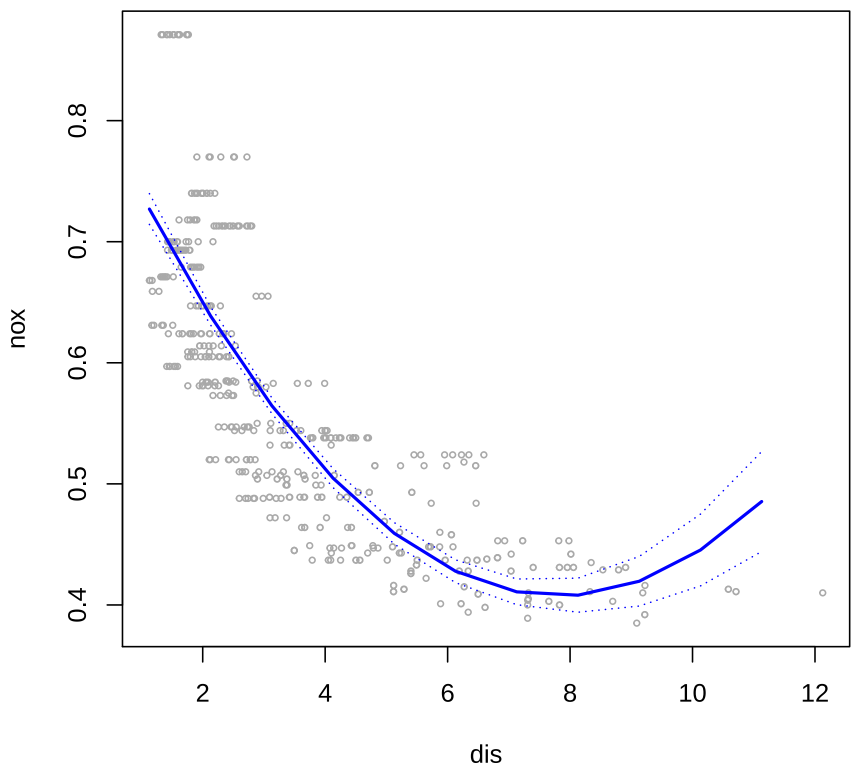


Figure 3: Degree 2 Polynomial Fit RSS = 2.035

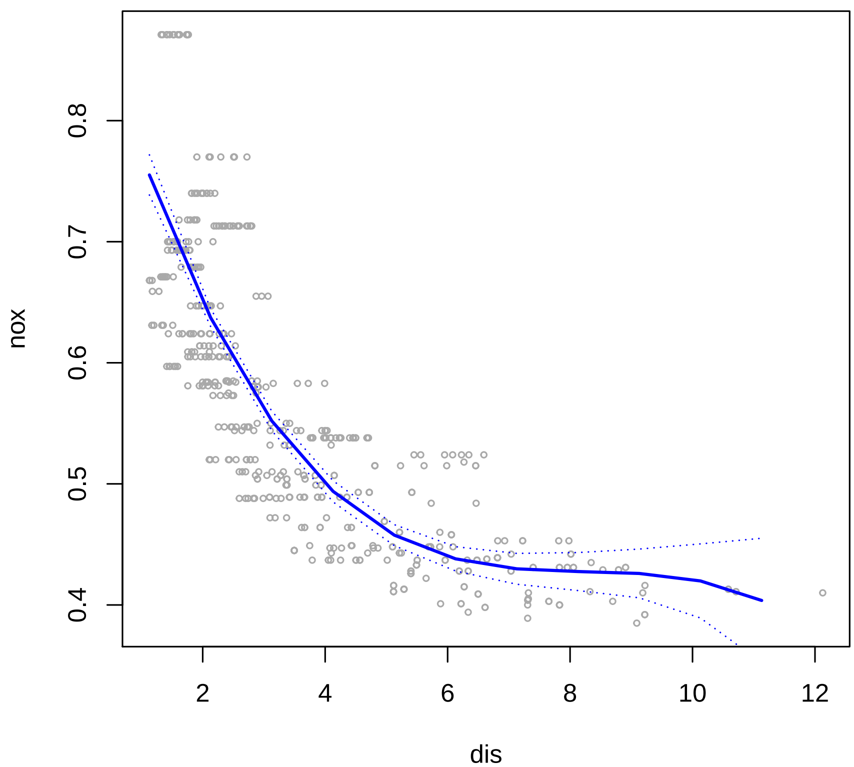


Figure 4: Degree 3 Polynomial Fit RSS = 1.934

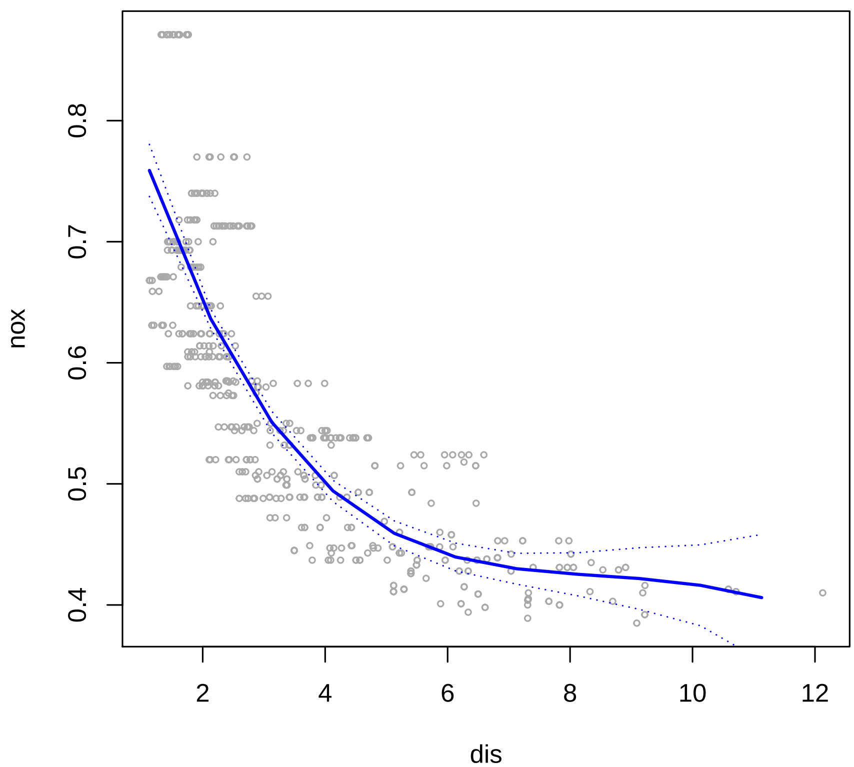


Figure 5: Degree 4 Polynomial Fit RSS = 1.933

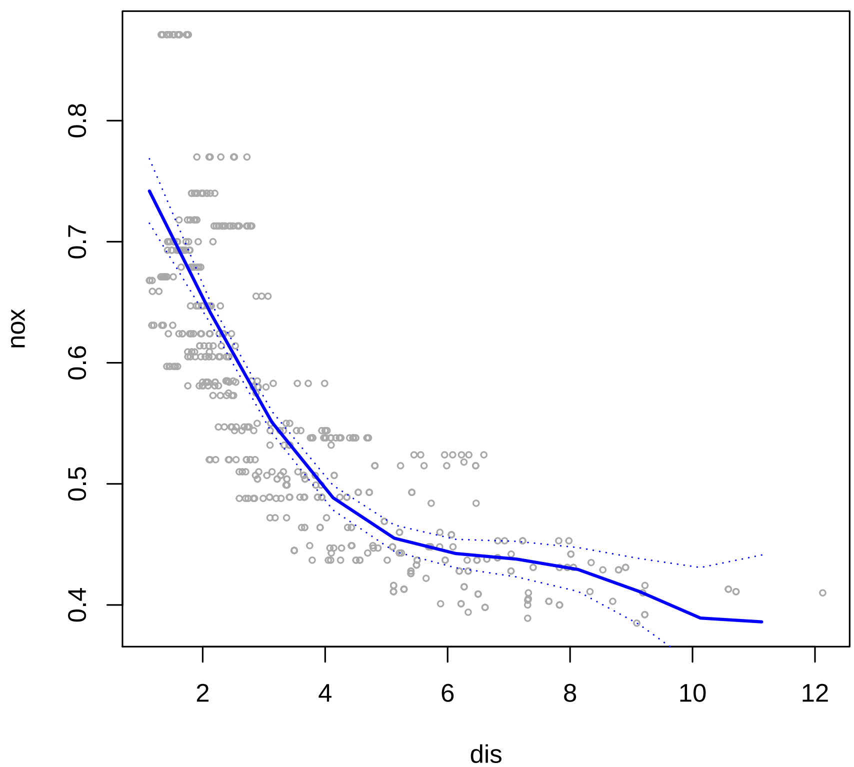


Figure 6: Degree 5 Polynomial Fit RSS = 1.915

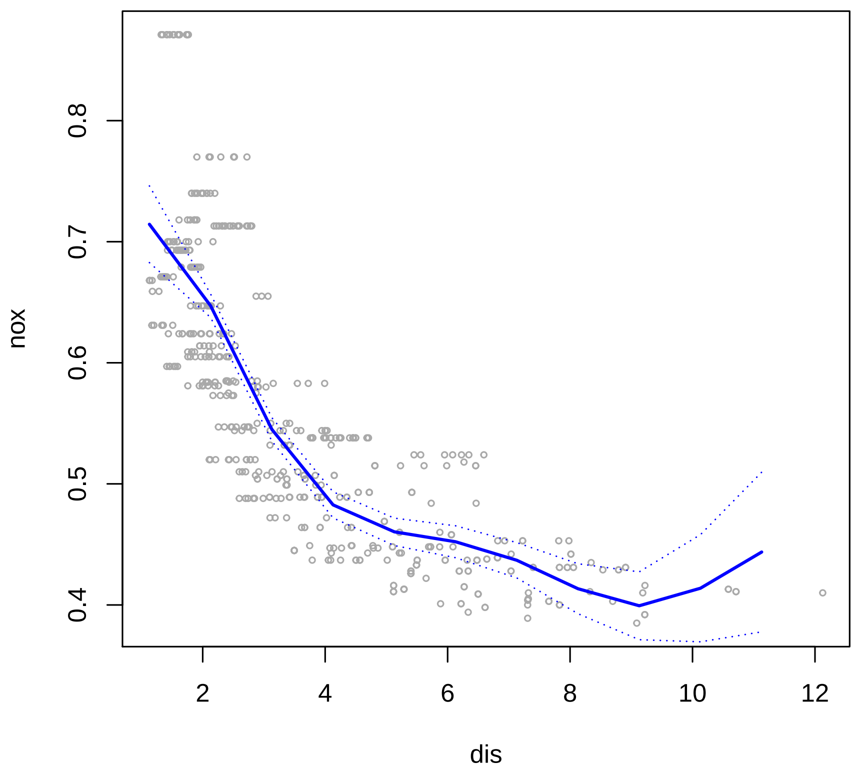


Figure 7: Degree 6 Polynomial Fit RSS = 1.878

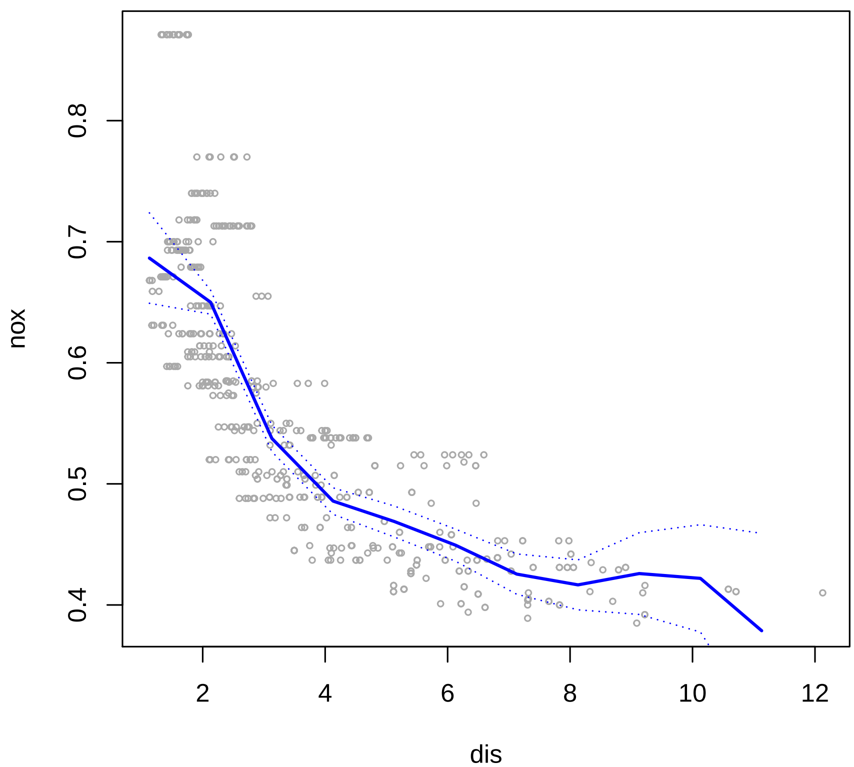


Figure 8: Degree 7 Polynomial Fit RSS = 1.849

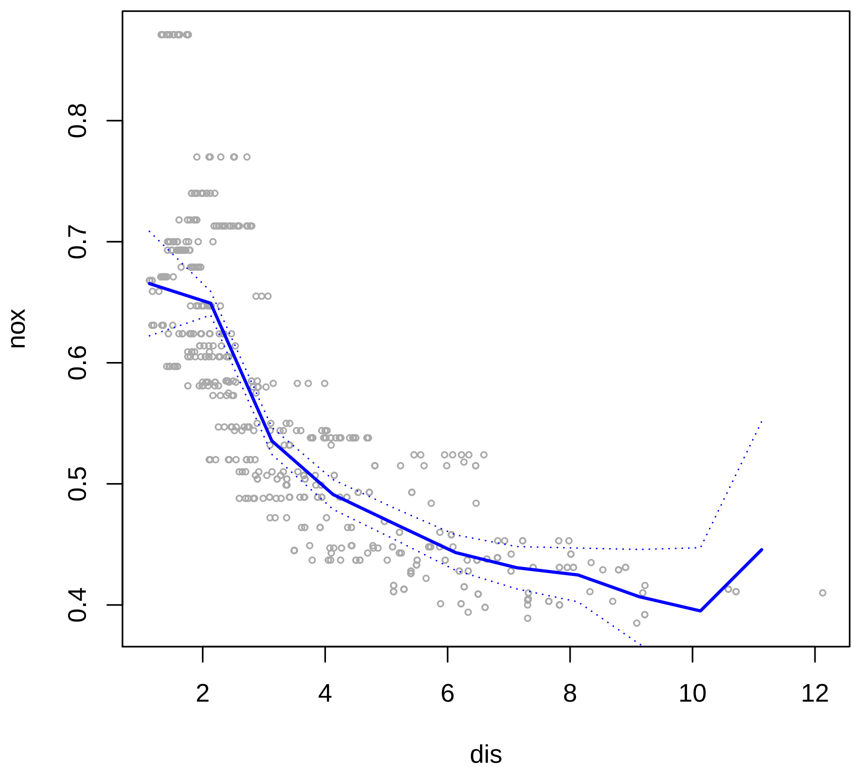


Figure 9: Degree 8 Polynomial Fit RSS = 1.836

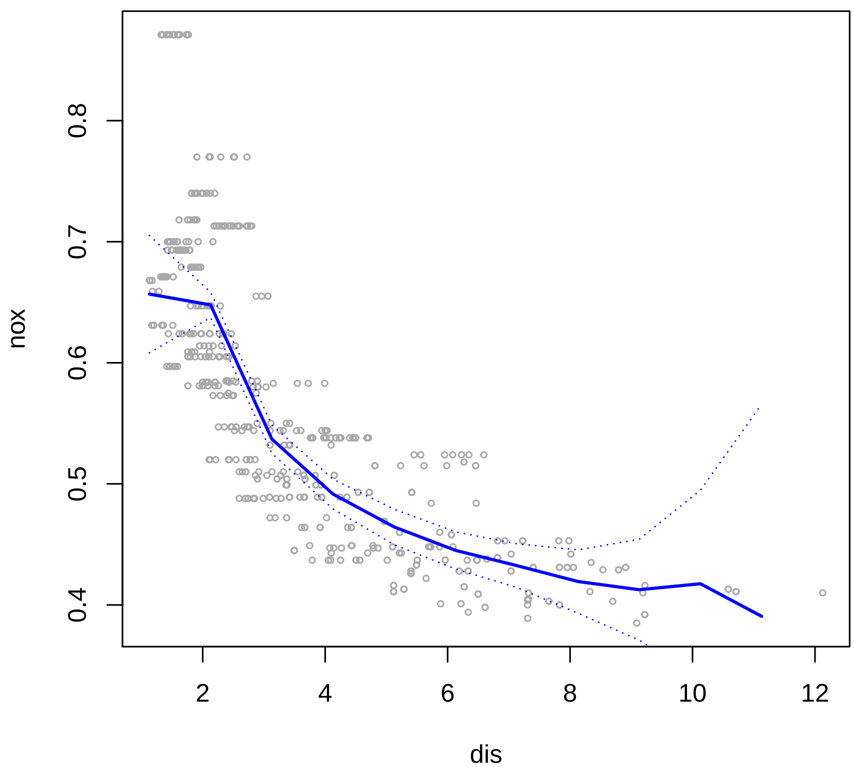


Figure 10: Degree 9 Polynomial Fit RSS = 1.833

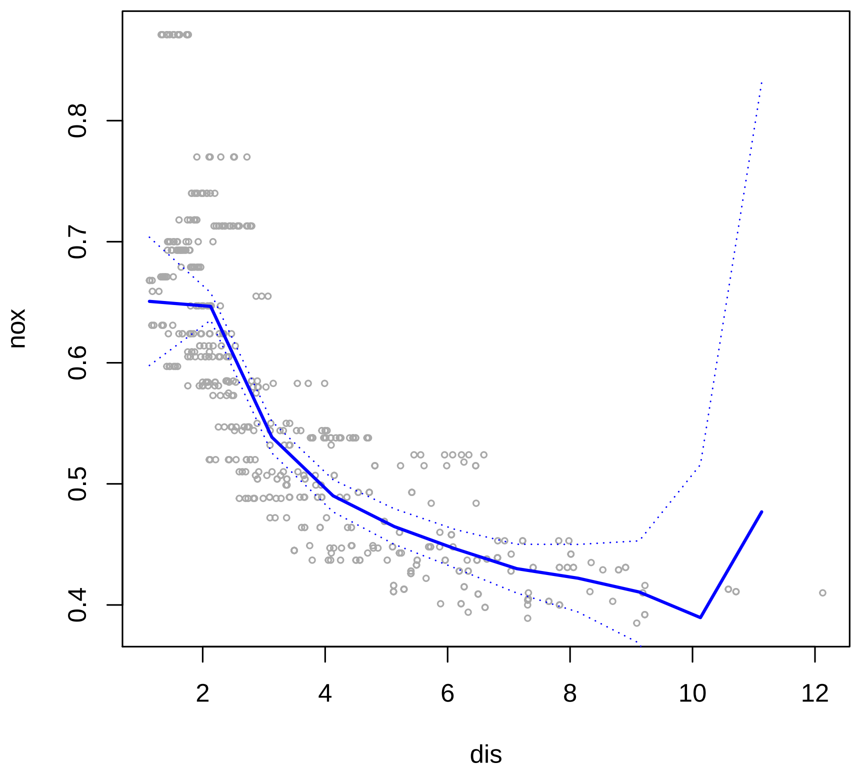


Figure 11: Degree 10 Polynomial Fit RSS = 1.832

c. K-Fold Cross Validation

```
> set.seed(1)
> n_folds = 5 # k folds
> deg = 10 # up to degree 5 polynomials considered
> fold = sample(n_folds, nrow(Boston), replace = TRUE)
> mse = vector(length=n_folds) #store mse for each k
> cv = vector(length=deg) #cross validated mse for each degree
> for (d in 1:deg){
+   for (k in 1:n_folds){
+
+     #fit degree d on all but kth fold, test on kth fold predictions
+     test = (fold==k)
+     train = !test
+     fit = lm(nox~poly(dis,d), data = Boston, subset = train)
+     pred=predict(fit,Boston[test,],type ="response")
+     mse[k] = mean((pred-nox[test])^2) #test mse for kth fold
+   }
+   cv[d] = mean(mse) #mean mse over all k folds
+ }
> plot(cv,xlab = "Degree",ylab = "Test MSE")
```

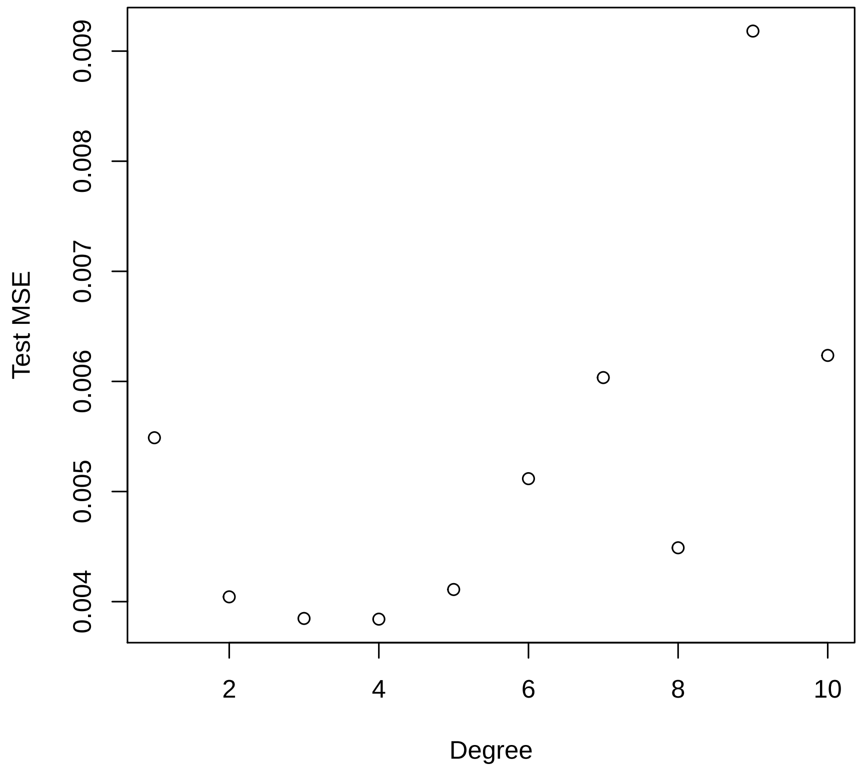


Figure 12: Cross Validated MSE vs. Polynomial Degree

K-fold cross validation reveals that a 4-degree polynomial provides the best fit to the data, with a test MSE of 0.0038, as shown in the above plot.

d. Regression spline with 4 degrees of freedom: for a cubic spline, use 1 knot at the halfway point for 4 DF.

```
> library(splines)
> fit=lm(nox ~ bs(dis,df=4),data=Boston)
> pred=predict(fit,newdata=list(dis=dis.grid),se=T)
> #plot spline with std error
> plot(dis,nox,col="gray")
> lines(dis.grid,pred$fit,lwd=2)
> lines(dis.grid,pred$fit+2*pred$se ,lty="dashed")
```

```
> lines(dis.grid,pred$fit-2*pred$se ,lty="dashed")
```

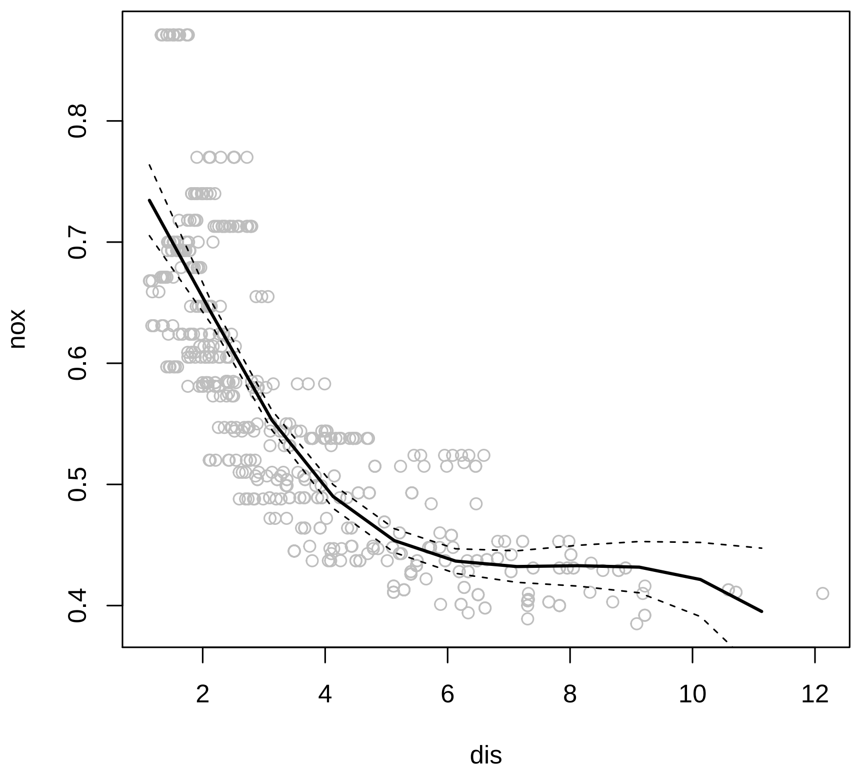
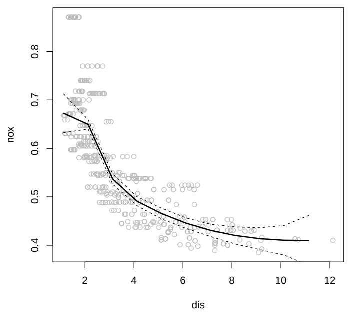
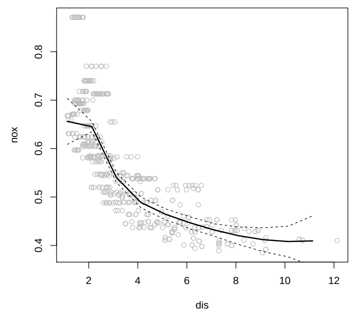


Figure 13: Cubic Spline with 4 DF, 1 Knot

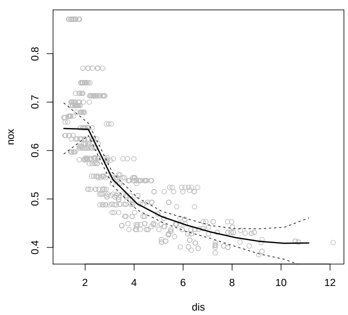
e. Cubic splines with degrees of freedom ranging from 5 to 10:



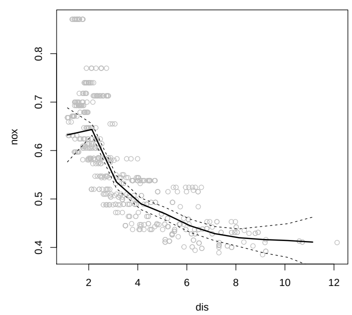
(a) 5 DF Spline, RSS = 1.84



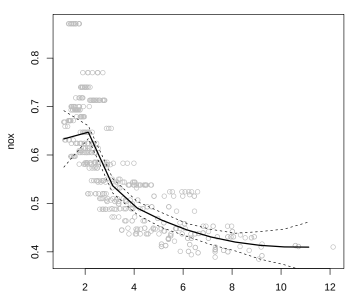
(b) 6 DF Spline, RSS = 1.834



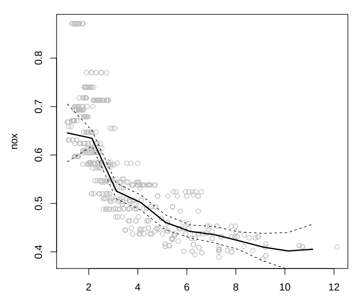
(c) 7 DF Spline, RSS = 1.83



(d) 8 DF Spline, RSS = 1.817



(e) 9 DF Spline, RSS = 1.826



(f) 10 DF Spline, RSS = 1.793

Figure 14: Cubic Splines with Various Degrees Freedom

f. K-Fold Cross Validation to Select DF

```
> set.seed(1)
> n_folds = 5 # k folds
> deg = 15 # up to 15 degrees of freedom considered (starting from 4 DF)
> fold = sample(n_folds, nrow(Boston), replace = TRUE)
> mse = vector(length=n_folds) #store mse for each k
> cv = vector(length=deg-4+1) #cross validated mse for each DF
> for (d in 4:deg){
+   for (k in 1:n_folds){
+
+     #fit degree d on all but kth fold, test on kth fold predictions
+     test = (fold==k)
+     train = !test
+     fit = lm(nox~bs(dis,df=d), data = Boston, subset = train)
+     pred=predict(fit,Boston[test,],type ="response")
+     mse[k] = mean((pred-nox[test])^2) #test mse for kth fold
+   }
+   cv[d-4+1] = mean(mse) #mean mse over all k folds
+ }
> DF = seq(4,deg, by=1)
> plot(DF,cv,xlab = "Degree Freedom",ylab = "Test MSE")
```

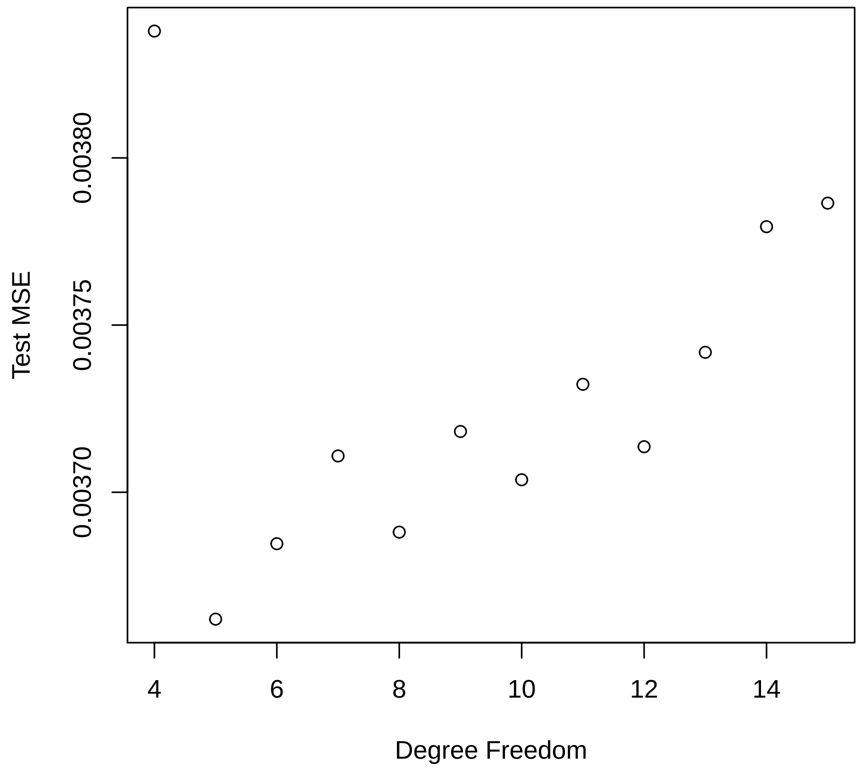


Figure 15: 5-Fold Cross Validation Results

The above plot of test MSE vs. DF from k-fold cross-validation shows a steep drop from $DF = 4$ to $DF = 5$ and increases gradually after that. Thus, 5 degrees of freedom for a cubic spline provides the best fit, with a test MSE of 0.004.

2 7.10 Applied

a. Forward selection to choose predictors for *Outstate* from the *College* data.

```
> library(ISLR)
> library(leaps)
> set.seed(1)
> # train and test subsets
```

```
> train = sample(c(TRUE,FALSE), nrow(College), rep=TRUE)
> test = (!train)
> n = length(College)-1 #num predictors
> #forward selection of predictor subset
> regfit.fwd = regsubsets(Outstate ~.,data=College[train,], nvmax=n, method ="forward")
> #cross validation
> test.mat=model.matrix(Outstate~.,data=College[test,])
> val.errors=rep(NA,n)
> for(i in 1:n){
+   coefi=coef(regfit.fwd,id=i)
+   pred=test.mat[,names(coefi)]%*%coefi
+   val.errors[i]=mean((College$Outstate[test]-pred)^2)
+ }
> #best model has min test mse
> best = which.min(val.errors)
> coef(regfit.fwd,id=best)
```

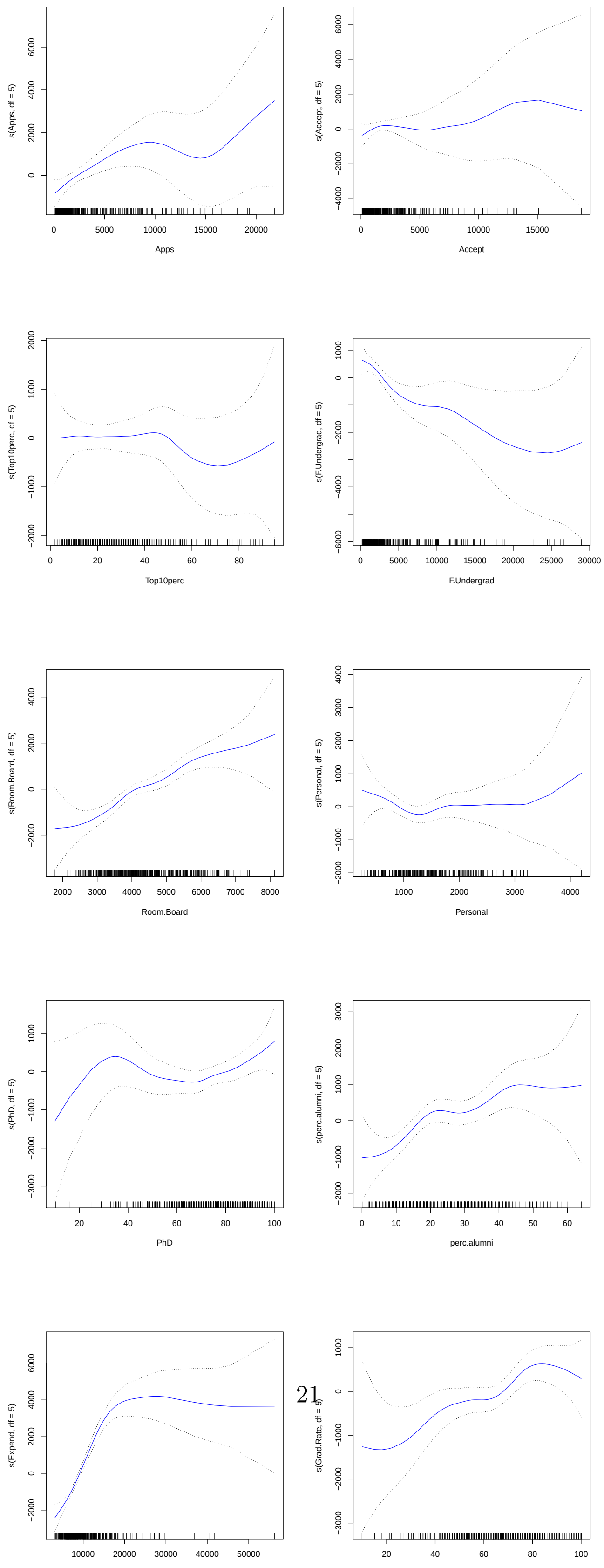
```
  (Intercept)    PrivateYes          Apps        Accept        Enroll
-1.276848e+03  2.378837e+03 -8.357838e-02  5.032838e-01 -7.321260e-01
    Top10perc     Top25perc   F.Undergrad   P.Undergrad    Room.Board
 1.916876e+01 -2.825120e+00 -3.375354e-03 -9.525220e-02  1.102276e+00
        Books      Personal           PhD      Terminal     S.F.Ratio
-5.925794e-01 -5.758528e-02  1.797836e+00  2.612394e+01 -5.693272e+01
  perc.alumni        Expend     Grad.Rate
 4.979977e+01  1.451966e-01  1.910491e+01
```

b. GAM model of *Oustate* from the 11 variables selected in previous step:

```
> library(gam)
> gam2 = gam(Outstate~ Private + s(Apps,df=5)+ s(Accept,df=5) + s(Top10perc,df=5)
+           + s(F.Undergrad,df=5) + s(Room.Board,df=5) + s(Personal,df=5)
+           + s(PhD,df=5) + s(perc.alumni,df=5) + s(Expend,df=5)
+           + s(Grad.Rate,df=5), data=College[train,])
> summary(gam2)
```

```
Call: gam(formula = Outstate ~ Private + s(Apps, df = 5) + s(Accept,
    df = 5) + s(Top10perc, df = 5) + s(F.Undergrad, df = 5) +
    s(Room.Board, df = 5) + s(Personal, df = 5) + s(PhD, df = 5) +
    s(perc.alumni, df = 5) + s(Expend, df = 5) + s(Grad.Rate,
    df = 5), data = College[train, ])
Deviance Residuals:
     Min       1Q   Median       3Q      Max
-5820.10  -988.52    55.74  1126.18  6807.63

(Dispersion Parameter for gaussian family taken to be 3079723)

    Null Deviance: 6334941086 on 399 degrees of freedom
```

```
Residual Deviance: 1071744105 on 348.0001 degrees of freedom
AIC: 7161.586

Number of Local Scoring Iterations: 3

Anova for Parametric Effects
                      Df     Sum Sq    Mean Sq  F value    Pr(>F)
Private                1 1446319741 1446319741 469.6265 < 2.2e-16 ***
s(Apps, df = 5)        1  924722790  924722790 300.2616 < 2.2e-16 ***
s(Accept, df = 5)      1  228197438  228197438  74.0967 2.607e-16 ***
s(Top10perc, df = 5)   1  345800541  345800541 112.2830 < 2.2e-16 ***
s(F.Undergrad, df = 5) 1  164659699  164659699  53.4657 1.823e-12 ***
s(Room.Board, df = 5)  1  516398355  516398355 167.6769 < 2.2e-16 ***
s(Personal, df = 5)    1    2857178    2857178   0.9277 0.3361200
s(PhD, df = 5)         1   34377751   34377751  11.1626 0.0009252 ***
s(perc.alumni, df = 5) 1  141856333  141856333  46.0614 4.951e-11 ***
s(Expend, df = 5)      1  309162918  309162918 100.3866 < 2.2e-16 ***
s(Grad.Rate, df = 5)   1   36906044   36906044  11.9836 0.0006036 ***
Residuals            348 1071744105    3079723
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Anova for Nonparametric Effects
                     Npar Df  Npar F     Pr(F)
(Intercept)
Private
s(Apps, df = 5)            4  3.8054  0.004826 **
s(Accept, df = 5)          4  1.5203  0.195750
s(Top10perc, df = 5)       4  0.6553  0.623485
s(F.Undergrad, df = 5)     4  1.9339  0.104329
s(Room.Board, df = 5)      4  1.3601  0.247404
s(Personal, df = 5)        4  1.5101  0.198734
s(PhD, df = 5)             4  2.4289  0.047535 *
s(perc.alumni, df = 5)     4  2.2971  0.058762 .
s(Expend, df = 5)          4 20.9886 1.554e-15 ***
s(Grad.Rate, df = 5)       4  1.7419  0.140252
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1
```


s(Apps, df = 5)
Apps
s(Accept, df = 5)
Accept
s(Top10perc, df = 5)
Top10perc
s(F.Undergrad, df = 5)
F.Undergrad
s(Room.Board, df = 5)
Room.Board
s(Personal, df = 5)
Personal
s(PhD, df = 5)
PhD
s(perc.alumni, df = 5)
perc.alumni
s(Expend, df = 5)
Expend
s(Grad.Rate, df = 5)
Grad.Rate

From the above plots, you can see how the smoothing splines capture the relationship between each predictor and the response. None of the relationships appear particularly linear.

c./d.
```
> preds=predict(gam2,newdata=College[test,])
> mse = mean((College$Outstate[test]-preds)^2)
```

With a test MSE of 4032569.48, it seems like this model is not a very good predictor of the response variable. More steps would be needed to cross validate the model against alternative combinations of the variables and their basis functions.