Homework3

Sohaib Syed

2023-03-05

Rectitation Exercises

1.1 Chapter 6

Exercise 1

a The smallest training RSS is likely to be the best subset model because it takes into account all of the combinations of predictors to get the smallest RSS values.

b I believe it can not be said for certain, but a better test RSS will be associated with a model that was less flexible during the training set. It is possible that best subset still has the better training RSS as it has taken into consideration every possible model, but forward/backward stepwise selection could have chosen a less flexible model that performs better on out-of-sample data by simply taking the greedy choice for predictors.

c

i. True, because if the first model has k variables by forward selection by adding those predictors, then doing another forward selection should add another predictor to get k+1 predictors

ii. True, because if one model has k predictors chosen by backwards selection by subtracting a predictor at a time, then it had to be a subset of a model that first had K+1 predictors as the first model should have one predictor less

iii. Generally this is false. As backwards selection and forward selection start at different null models, getting to the point where one is a subset of another is unlikely because they possibly took different paths and have different predictors based on the criteria that was best at earlier steps

iv. Again generally false. This is because of the same reasons as part iii as the models are likely to take different paths in earlier steps when adding, or subtracting predictors.

v. False, because with best subset selection we are not simply adding and subtracting predictors. If model 1 has k+1 predictors, we can not say that model 2 with k predictors is just model 1 with 1 of the predictors taken away because the different combnination of k variables that best subset computes to choose the best model

Exercise 2

a iii is correct because first we know lasso has a constraint in the formula making it less flexible, and also we trade variance for higher bias, so as option 3 states when the penalty increases we reduce the variance and increase bias, thus when the opposite happens and we want to minimize penalty the bias increases less than the variance decreases leading to a better model.

b Again, iii is correct for similar reason as lasso because they have similar penalty terms. When penalty is increased for ridge, the variance is decreased and bias is increased. Ridge only differs in the sense that coefficients can not be 0.

c A non-linear model is expected to certainly be more flexible as the model is not restricted to assume a certain structure. The prediction accuracy is better when the decrease in bias is more than increase in variance.

Exercise 3

a iv as model becomes more flexible (s increases), the model will continually decrease RSS and eventually overfit

b ii At first the greater flexibility will allow for the model to start explaining features and use that information to predict correctly, until the point where the model is overfit to the training data and starts to miss predict on out of sample data

c iii variance should increase steadily because a more flexible model is susceptible to a greater change if a new training point is added to data set

d iv with more flexibility the model can closer fit data, and the squared bias will decrease steadily

e v The irreducible error is independent of the model, so there will always be a constant unexplained error

Exercise 4

a iv because the penalty term will increase and regardless of how small the RSS gets the shrinkage term will become massive and result in ridge training RSS to increase

b ii as the goal of the shrinkage term is to find best model for test accuracy so initially the hope is (and reality is likely) that test RSS decreases while shrinkage increases. Eventually the penalty will be large and the increase in bias will be greater resulting in rise in test RSS

c iv as shrinkage is increased the penalty will start to decrease Betas towards 0 thus leading to less flexible models.

d iii similar reasoning to part c, where as betas go towards 0, less flexible model, and the squared bias is higher

e v because the model and unexplained error are independent regardless of flexibility or number of predictors

Exercise 5

a n=2, p=2, x11=x12, x21= x22

y1=-y2, x11=-x21 x12=-x22

Bhat_0=0

from ridge equation

we have to optimize beta on this equation:

(y1-B1x11-B2x12)^2 + (y2 - B1x21 - B2x22)^2 + lambda*(B1^2 +B2^2)

b argue Bhat1=Bhat2

from part a, take partial derivatives with respect to Bhat1 and Bhat2

partial of B1hat = 2(-x11)(y1-Bhat2x11-Bhat2x11)+2(-x21)(y1-Bhat2x21-Bhat2x21)+ (2lambda*Bhat1)=0

partial of B2hat = 2(-x11)(y1-Bhat2x11-Bhat2x11)+2(-x11)(y1-Bhat2x21-Bhat2x21)+ (2lambda*Bhat2)=0

lambda*Bhat1= x11y1+ x21y2 +2 B1hat x11 x22 + 2 B2hat x11 x22

lambda*Bhat2= x11y1+ x21y2 +2 B1hat x11 x22 + 2 B2hat x11 x22

as we see if we divide over the lambda we get Bhat1 =Bhat2

c lasso equation need to optimize Betas for this equation:

(y1-Bhat1x11-Bhat2x12)^2 + (y2-Bhat1x21-Bhat2x22)^2 + lambda (|Bhat1|+|Bhat2|)

d we can take a similar approach as in part b by taking derivatives and setting equal to 0 to try to obtain equal solutions:

from c we can further subsitute and get:

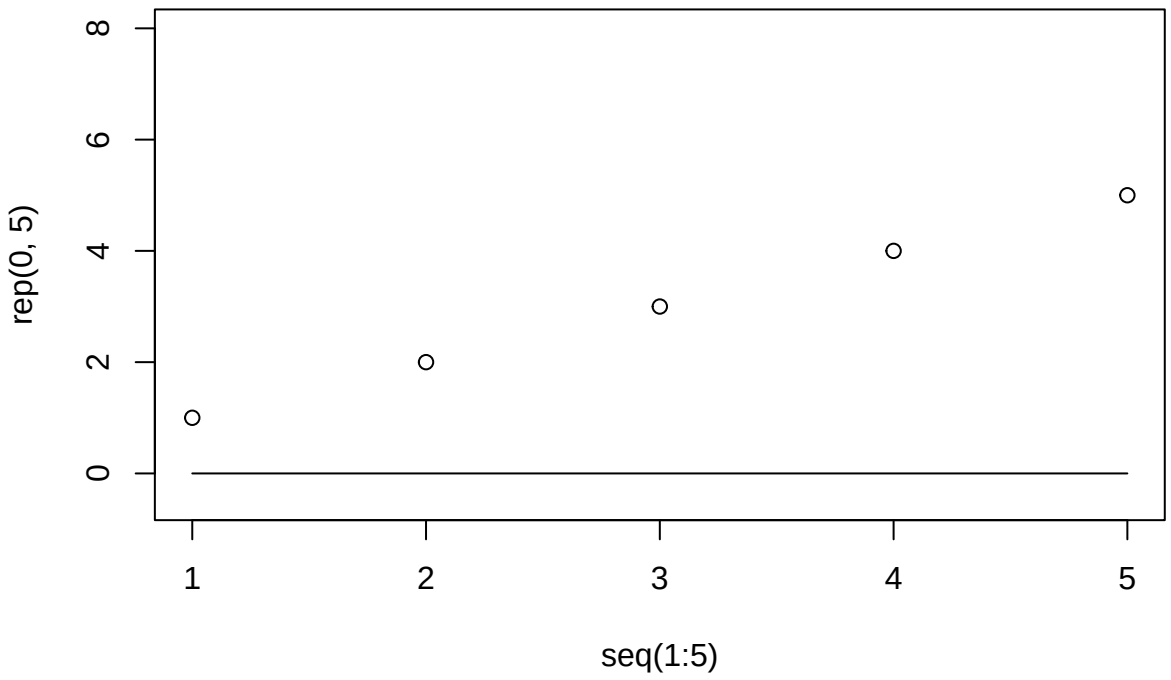
2(y1-Bhat1x11-Bhat2x11)^2+ lambda (|Bhat1|+|Bhat2|)

we can already see a problem with trying to take derivatives with absolute value sign and this should be an indication that Bhat1 and Bhat2 cant be unique as therw will be +/- sign associated with their derivatives

1.2 Chapter 7

Exercise 2

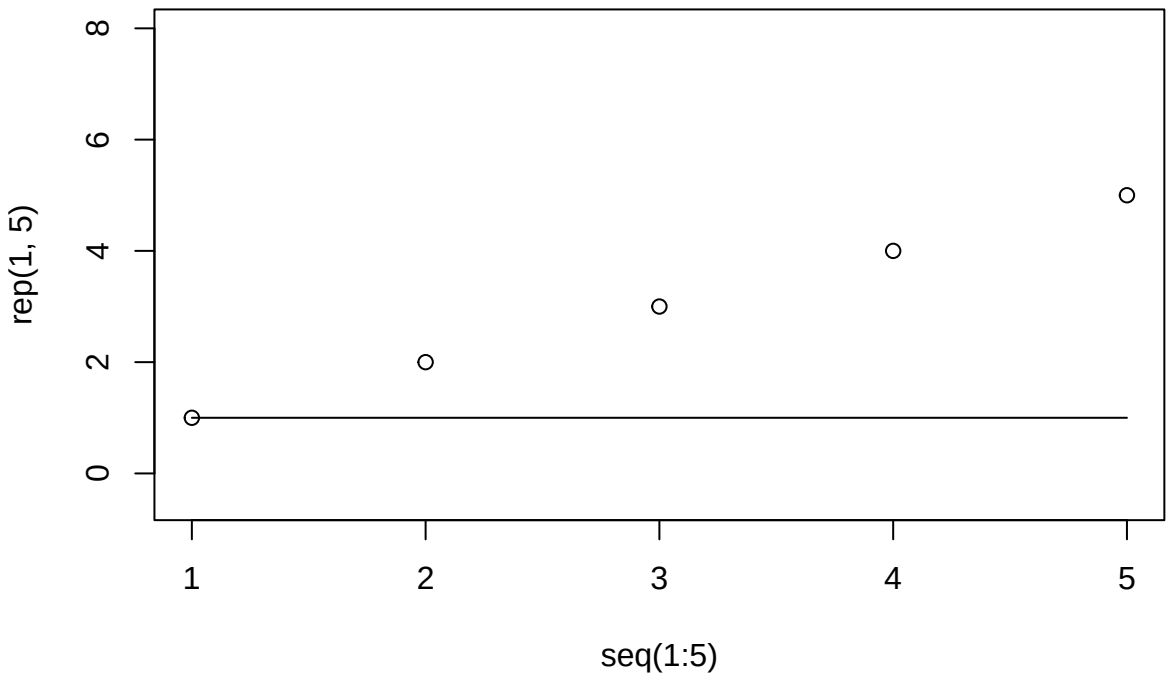
```
plot(seq(1:5),rep(0,5),type='l',ylim=c(-.5,8))
points(c(1,2,3,4,5),c(1,2,3,4,5))
```



a

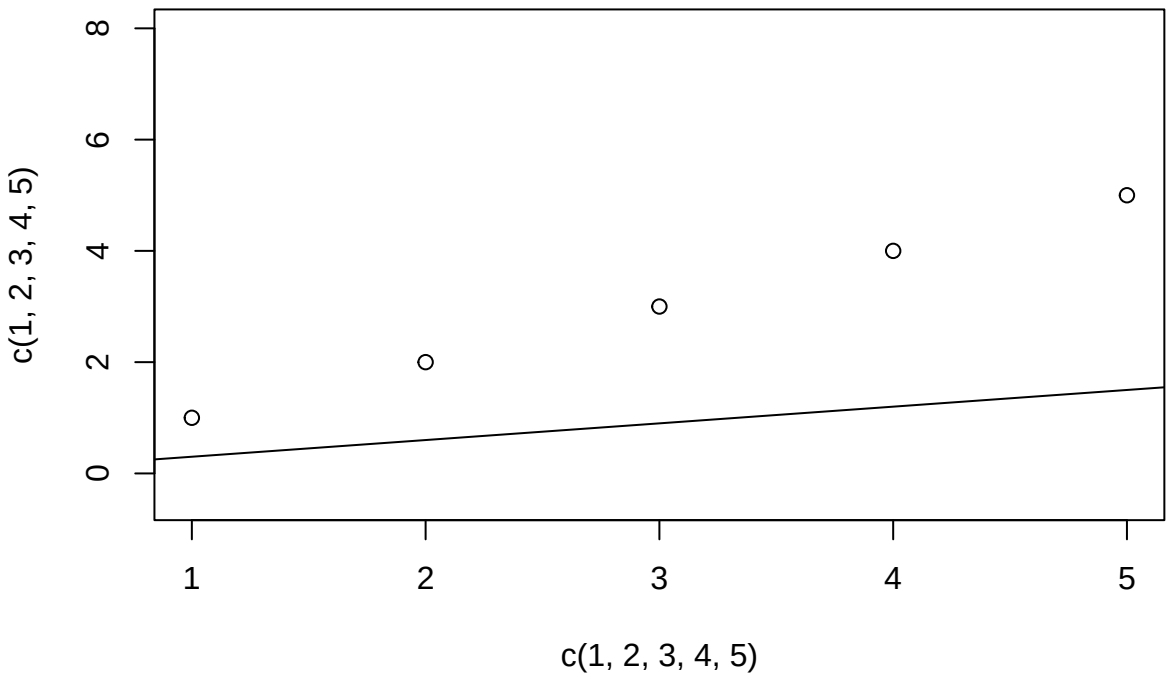
b

```
plot(seq(1:5),rep(1,5),type='l',ylim=c(-.5,8))
points(c(1,2,3,4,5),c(1,2,3,4,5))
```

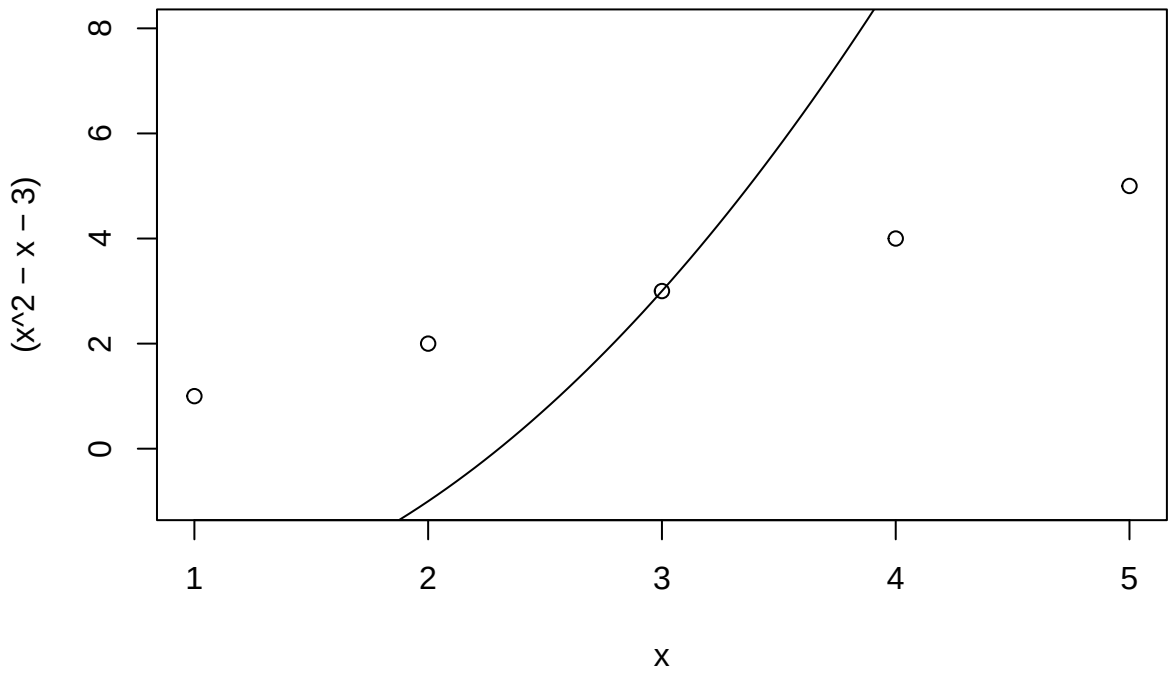


c

```
plot(c(1,2,3,4,5),c(1,2,3,4,5),ylim=c(-.5,8))
abline(a=0,b=.3)
```

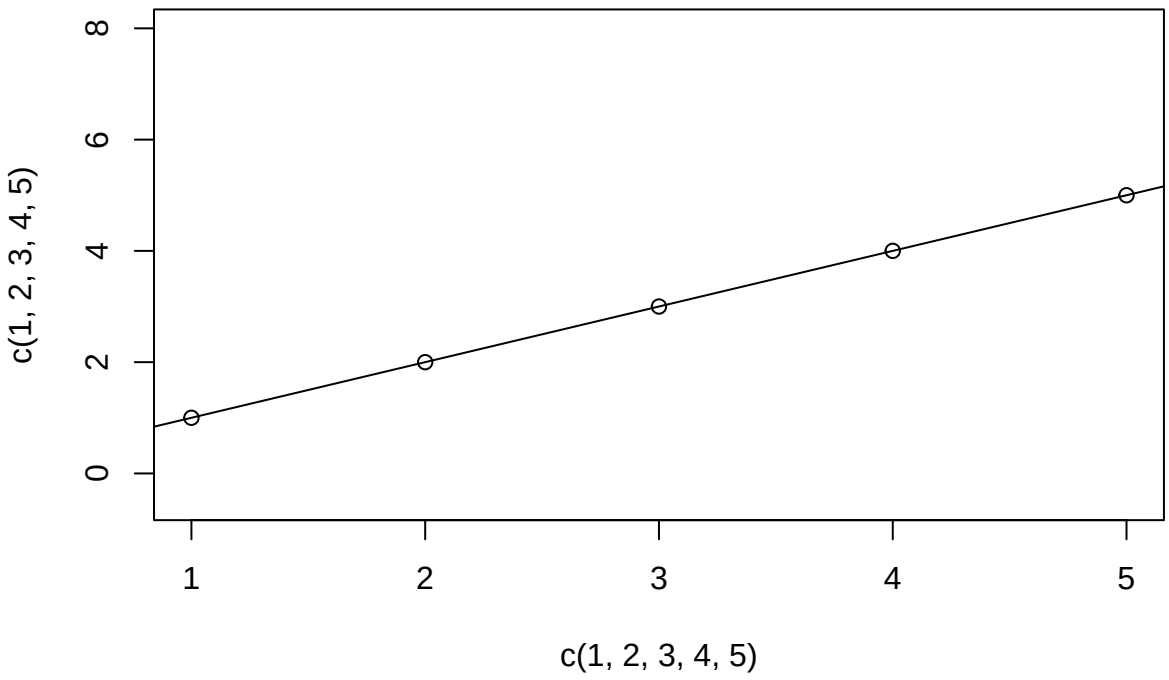



```
curve((x^2-x-3),from=1,to=5,ylim=c(-1,8))
points(c(1,2,3,4,5),c(1,2,3,4,5))
```



d

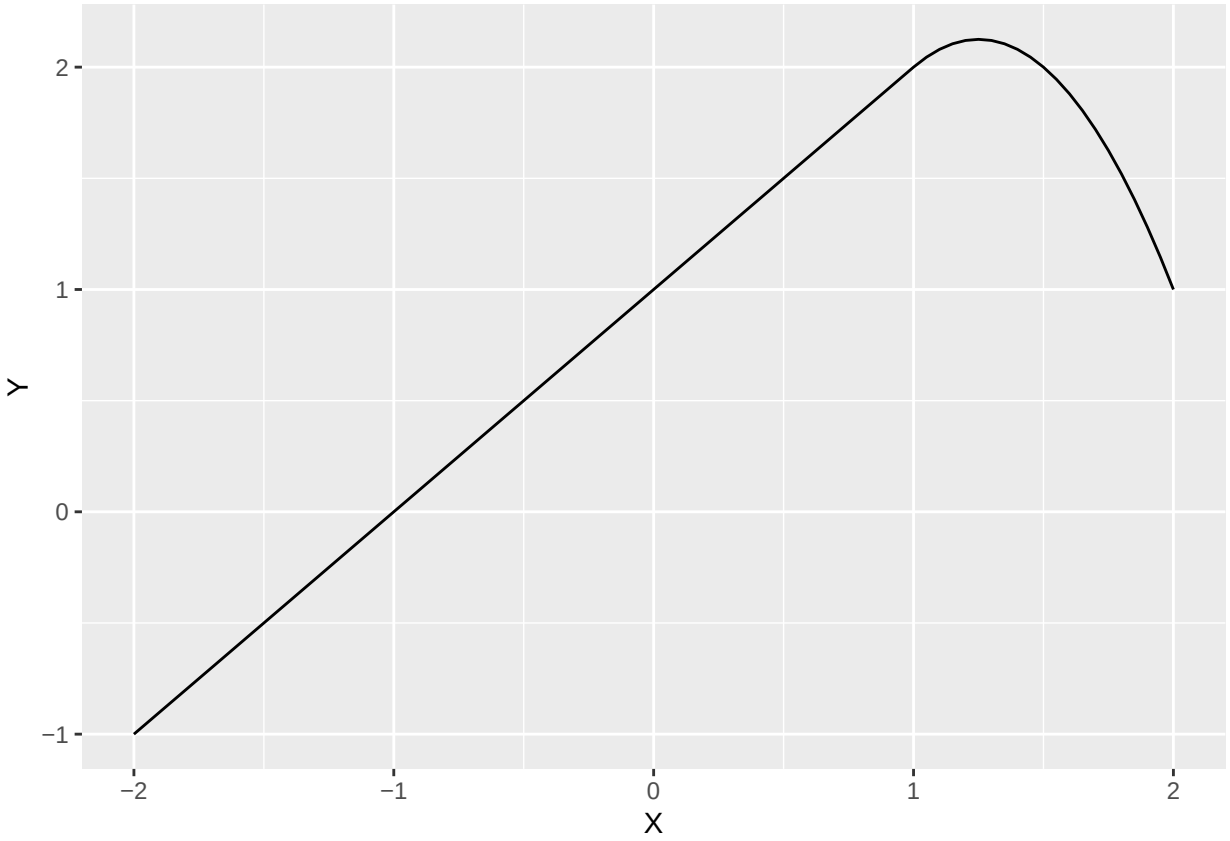
```
plot(c(1,2,3,4,5),c(1,2,3,4,5),ylim=c(-.5,8))
abline(a=0,b=1)
```



e

Exercise 3

```
X = seq(-2, 2,.05)
Y = 1 + X + -2 * (X - 1)^2 * (X >= 1)
df <- data.frame(X, Y)

ggplot(df, aes(x = X, y = Y)) + geom_line()
```



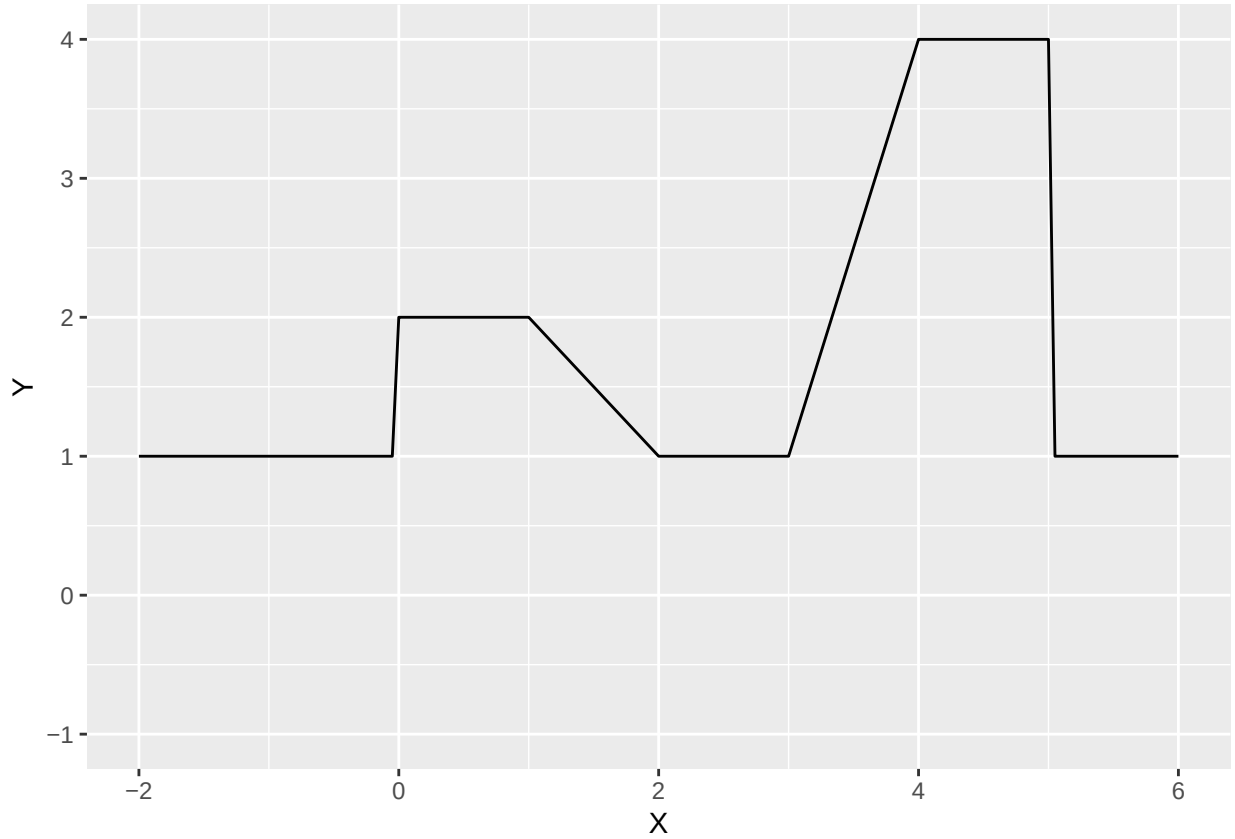
For X<1 the slope is 1, the y-intercept is at 1.

The function changes slope at x=1 since the indicator function activates b2, and that portion has a negative slope because of the negative coefficient Bhat2

So this is a 'piece wise function' where it is smooth and continuous at (1,2)

Exercise 4

```
X = seq(-2, 6, 0.05)
Y = 1 + (X >= 0 & X <= 2) - (X - 1)*(X >= 1 & X <= 2)+
  3*(X - 3)*(X >= 3 & X <= 4) + 3*(X > 4 & X <= 5)
df <- data.frame(X, Y)

ggplot(df, aes(x = X, y = Y)) +
  geom_line()+ylim(-1,4)
```



From -2 to 0 y=1 since none of the other basis functions are factring in, from 0 to 1 only first indicator function of b1 is contributing a 1 so y=1+1, then from 1 to 2, all of basis 1 (b1)is activated so the (x-1) term also contributes to y. Then from 3 to 4 only (x-3) from b2 is activated and since it is multiplied by coefficient B2=3 it is 3 times that factor, then from 4 to 5, only second half of b2 is contributing with coefficient 3 so we would get 3 from that half which + from B0 is 4. After 5, all indicator functions are 0 so none of the basis functions contribute and thus the coefficients are 0 and only y=1

Exercise 5

a g2 will have smaller training RSS as g2 will have a more flexible model to begin with since the penalty term uses 4th deriative.

when lambda infinity gˆ(m) will go to infinity, and thus we need to find what type of function would have 3 derivatives until 0 and 4 derivatives until 0, we see that a cubic polynomial would need 4 derivatives to go to 0 and that is more flexoble than a quadratic hence why g2 will be more flexible and lead to smaller training RSS

b We can assume that if we see that in part a g2 will overfit, then g1 must be a little better at predicitng testing data so g1 could possibly have lower test RSS

c if lambda=0 then there is no penalty term and both functions are the exact same which is just the RSS equation, and equal training and test RSS for both g1 and g2.

Practicum Problems

2.1 Problem 1

fors this portion I used chapter 6 lab on ridge and lasso

```
library(glmnet)
```

```
## Loading required package: Matrix
```

```
## Loaded glmnet 4.1-6
```

```
library(caret)
```

```
## Loading required package: lattice
```

```
set.seed(2)

cars_x<-model.matrix (mpg~., mtcars)[, -1]
cars_y<-mtcars$mpg

carsdf<-data.frame(cars_x,cars_y)
carstrainIdx <- createDataPartition(
  1:dim(carsdf)[1],
  times = 1,
  p = 0.8,
  list = T)$Resample1
carsTrain <- carsdf[carstrainIdx,]
carsTest <- carsdf[-carstrainIdx,]

carslinfit<-lm(cars_y~.,data=carsTrain)
summary(carslinfit)
```

```
##
## Call:
## lm(formula = cars_y ~ ., data = carsTrain)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.5609 -1.5735 -0.2537  0.9528  4.2843
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 20.79873   19.66233   1.058    0.305
## cyl         -0.32620    1.09140  -0.299    0.769
## disp         0.01546    0.02121   0.729    0.476
## hp          -0.02907    0.02559  -1.136    0.272
## drat         0.58640    1.77883   0.330    0.746
## wt          -3.30594    2.25006  -1.469    0.160
## qsec         0.46657    0.81665   0.571    0.575
```

```
## vs           0.68785    2.16676   0.317    0.755
## am           2.75122    2.21818   1.240    0.232
## gear         0.61945    1.51988   0.408    0.689
## carb        -0.52471    1.00173  -0.524    0.607
##
## Residual standard error: 2.669 on 17 degrees of freedom
## Multiple R-squared:  0.8851, Adjusted R-squared:  0.8176
## F-statistic:  13.1 on 10 and 17 DF,  p-value: 3.978e-06
```

hp, wt and am are the three features with greatest magnitude of t, |t|>1.Coefficients of features: hp= -0.02907, wt= -3.30594, am= 2.75122

```
carsrtrain_x<-model.matrix (cars_y~., carsTrain)[, -1]
carstrain_y<-carsTrain$cars_y

grid <- 10^ seq (10, -2, length = 100)
carsridgefit_cv<-cv.glmnet(carsrtrain_x,carstrain_y,alpha=0,lambda = grid,parallel = T)
```

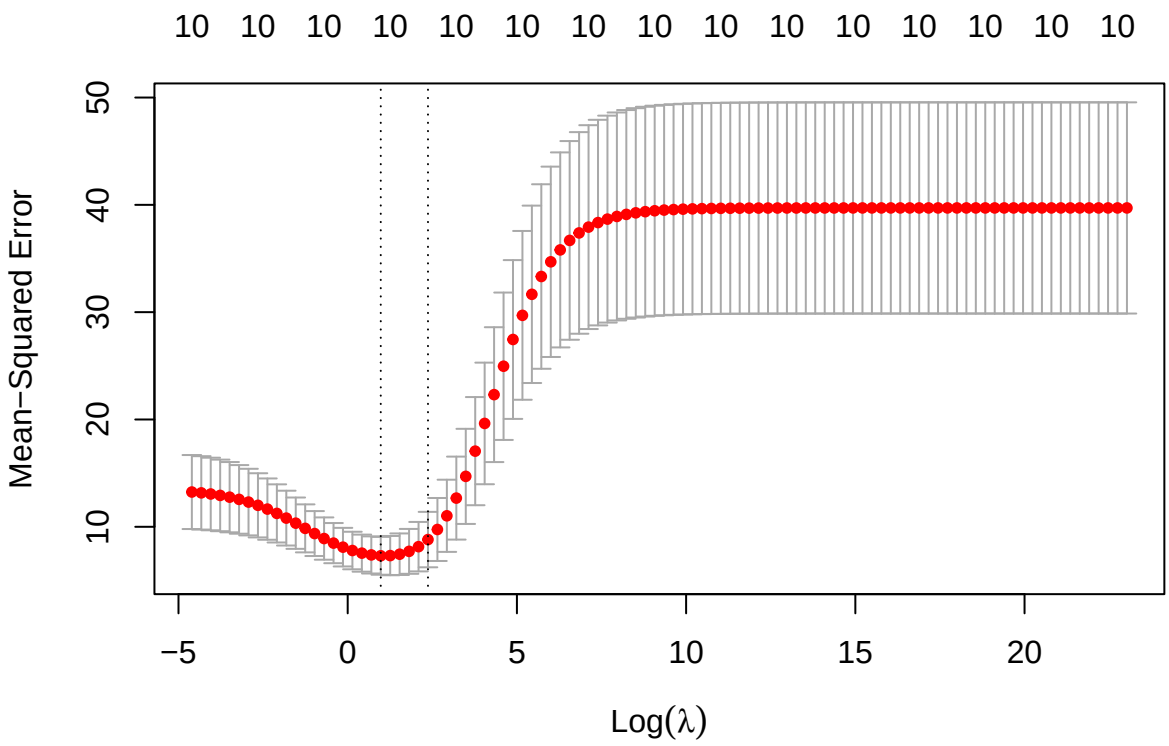
```
## Warning: executing %dopar% sequentially: no parallel backend registered
```

```
## Warning: Option grouped=FALSE enforced in cv.glmnet, since < 3 observations per
## fold
```

```
cat("\n min lambda is",carsridgefit_cv$lambda.min)
```

```
##
##  min lambda is 2.656088
```

```
plot(carsridgefit_cv)
```




```
carstest_x<-model.matrix (cars_y~., carsTest)[, -1]
carstest_y<-carsTest$cars_y
ridge.pred <- predict (carsridgefit_cv , s = carsridgefit_cv$lambda.min ,
newx = carstest_x)

mean((ridge.pred -carstest_y)^2)
```

```
## [1] 6.042079
```

```
coef(carsridgefit_cv,s=carsridgefit_cv$lambda.min)
```

```
## 11 x 1 sparse Matrix of class "dgCMatrix"
##                       s1
## (Intercept) 23.462180242
## cyl         -0.387145186
## disp        -0.004712688
## hp          -0.014173688
## drat         0.959179008
## wt          -1.166982468
## qsec         0.094891983
## vs           0.740023839
## am           1.885198178
## gear         0.578721583
## carb        -0.830666921
```

The MSE on the test set was 6.04. The coefficients certainly differ and the noticeable one is that the ridge coefficients are smaller in magnitude. It looks like the method did a little bit of shrinkage and variable selection because in the example of drat, vs, and carb the coefficients got bigger in magnitude and were not pushed towards 0.

2.2 Problem 2

```
swiss_x<-model.matrix (Fertility ~., swiss)[, -1]
swiss_y<-swiss$Fertility

swissdf<-data.frame(swiss_x,swiss_y)
swisstrainIdx <- createDataPartition(
  1:dim(swissdf)[1],
  times = 1,
  p = 0.8,
  list = T)$Resample1
swissTrain <- swissdf[swisstrainIdx,]
swissTest <- swissdf[-swisstrainIdx,]

swisslinfit<-lm(swiss_y~.,data=swissTrain)
summary(swisslinfit)
```

```
##
## Call:
## lm(formula = swiss_y ~ ., data = swissTrain)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -14.9059  -5.5062   0.3274   4.3465  15.1988
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)      67.83906   12.67504   5.352 6.52e-06 ***
## Agriculture      -0.18968    0.08159  -2.325 0.026376 *
## Examination      -0.26554    0.30107  -0.882 0.384157
## Education        -0.89905    0.23624  -3.806 0.000582 ***
## Catholic          0.11693    0.04356   2.685 0.011269 *
## Infant.Mortality  1.04739    0.43746   2.394 0.022493 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 7.766 on 33 degrees of freedom
## Multiple R-squared:  0.6426, Adjusted R-squared:  0.5885
## F-statistic: 11.87 on 5 and 33 DF,  p-value: 1.285e-06
```

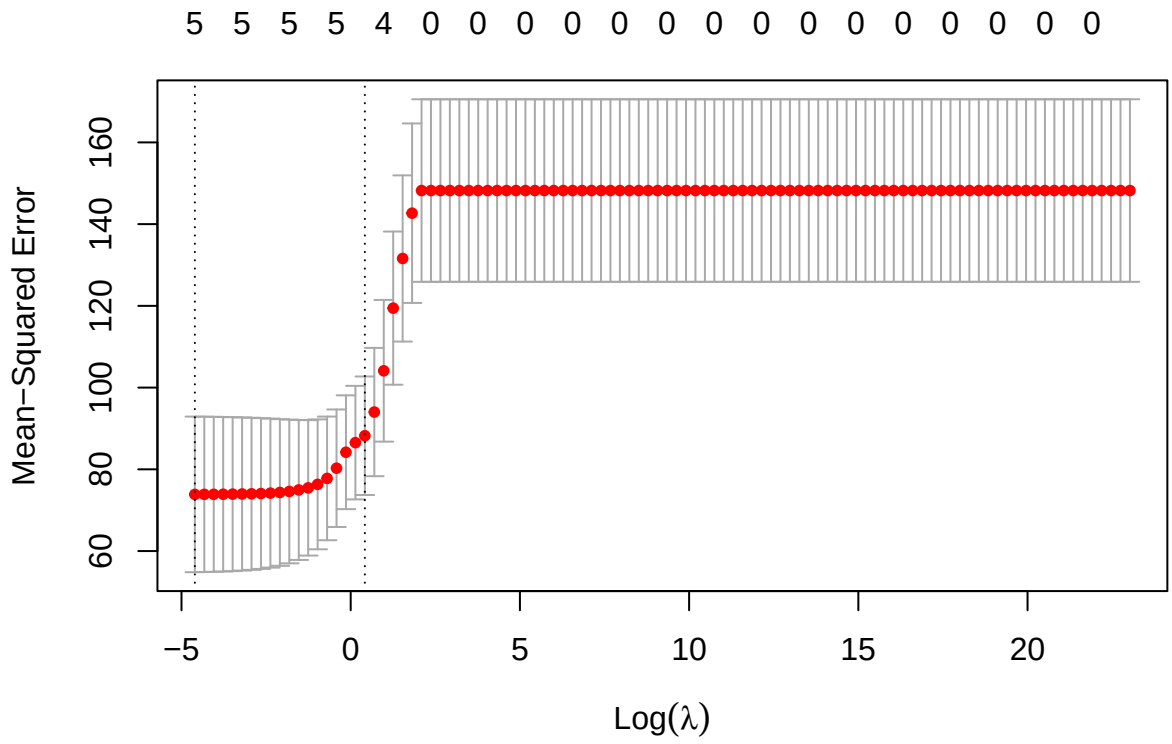
The important features seem to be Agriculture, Education, Catholic, and infant mortality

The coefficients of features are: Agriculture = -0.18968, Education =-0.89905, Catholic=0.11693, and infant mortality=1.04739

```
swisstrain_x<-model.matrix (swiss_y~., swissTrain)[, -1]
swisstrain_y<-swissTrain$swiss_y

swisslassofit_cv<-cv.glmnet(swisstrain_x,swisstrain_y,alpha=1,lambda = grid,parallel = T)

cat("\n min lambda is",swisslassofit_cv$lambda.min)
```

```
##
##  min lambda is 0.01
```

```
plot(swisslassofit_cv)
```




```
swisstest_x<-model.matrix (swiss_y~., swissTest)[, -1]
swisstest_y<-swissTest$swiss_y
lasso.pred <- predict(swisslassofit_cv,s=swisslassofit_cv$lambda.min,newx = swisstest_x)

mean((lasso.pred -swisstest_y)^2)
```

```
## [1] 16.94832
```

```
coef(swisslassofit_cv,s=swisslassofit_cv$lambda.min)
```

```
## 6 x 1 sparse Matrix of class "dgCMatrix"
```

```
##                          s1
## (Intercept)      67.7293098
## Agriculture      -0.1878494
## Examination      -0.2647225
## Education        -0.8956230
## Catholic          0.1164538
## Infant.Mortality  1.0468197
```

MSE on test is 16.94. The coefficients are smaller, but not by a lot. This lasso method only used shrinkage, but didn't variable select which is odd.

2.3 Problem 3

```
library(mgcv)
```

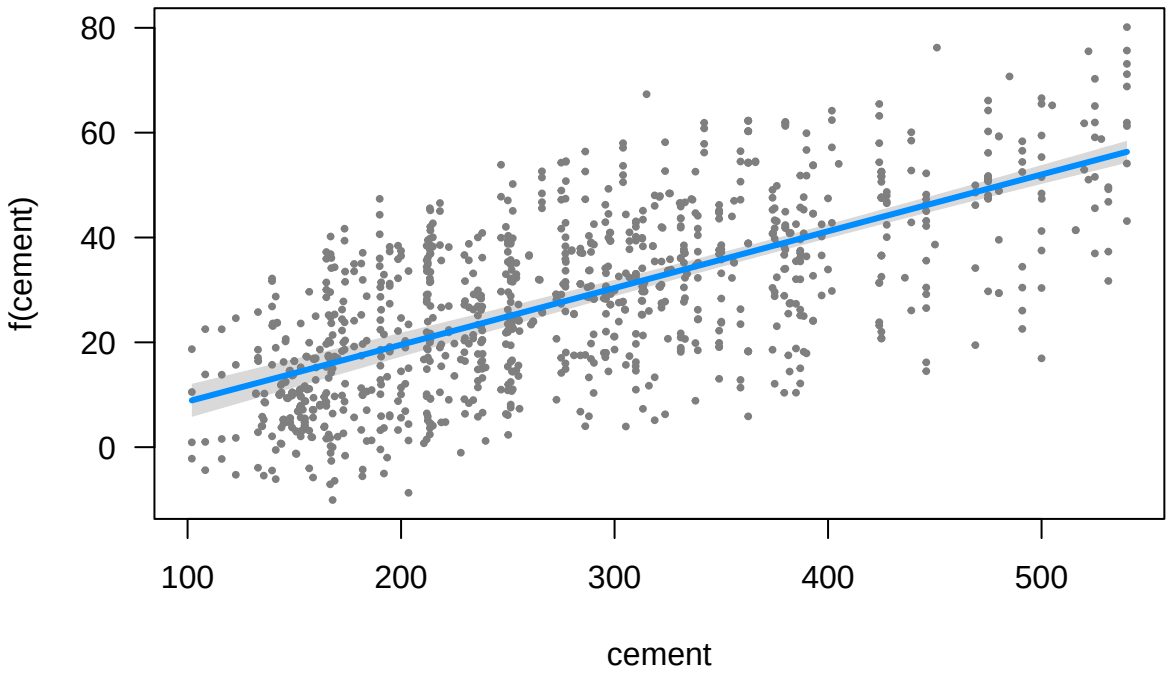
```
## Loading required package: nlme
```

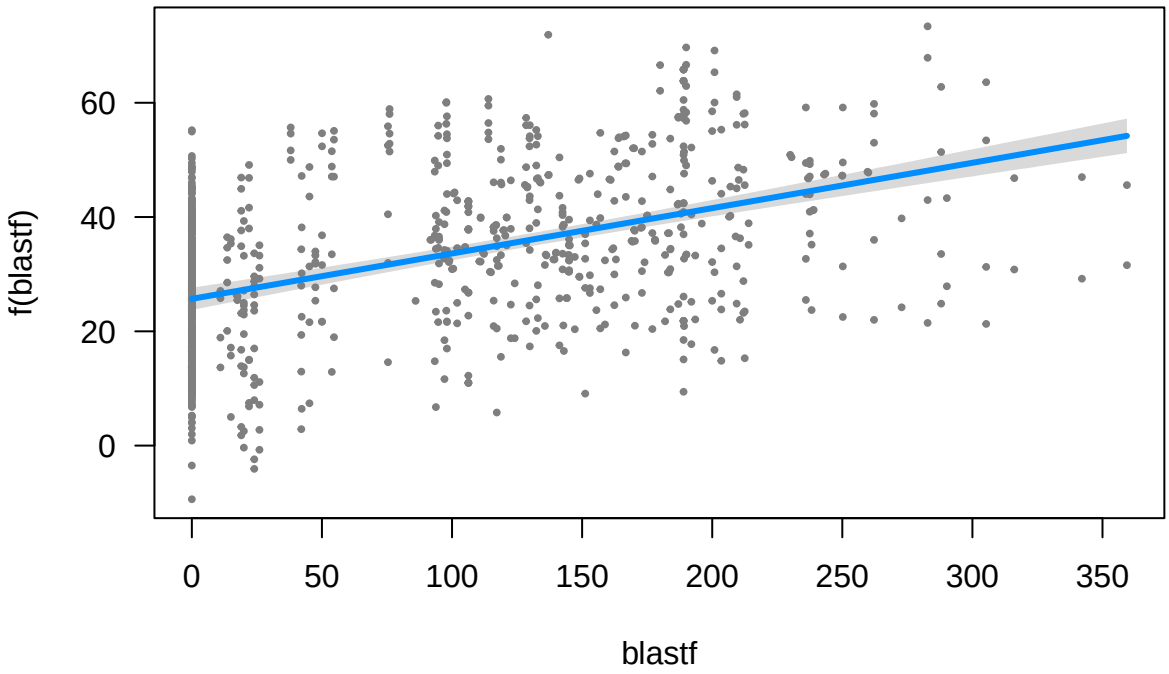
```
## This is mgcv 1.8-41. For overview type 'help("mgcv-package")'.
```

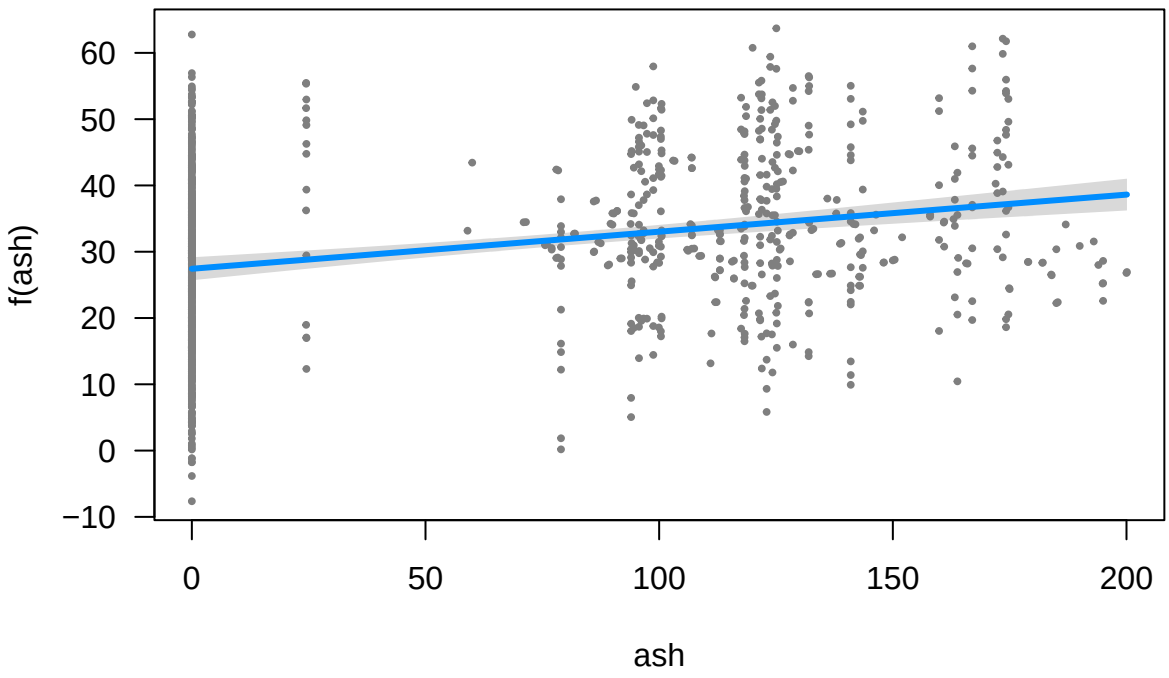
```
library(visreg)
library("readxl")
concretedf<-as.data.frame(read_excel("Concrete_Data.xls"))

colnames(concretedf) <- c("cement", "blastf", "ash", "water",
                          "super", "coarse", "fine", "age", "strength")

gamfitlinear<-gam(strength~ cement+blastf+ash+water+super+coarse,data=concretedf)
summary(gamfitlinear)
```

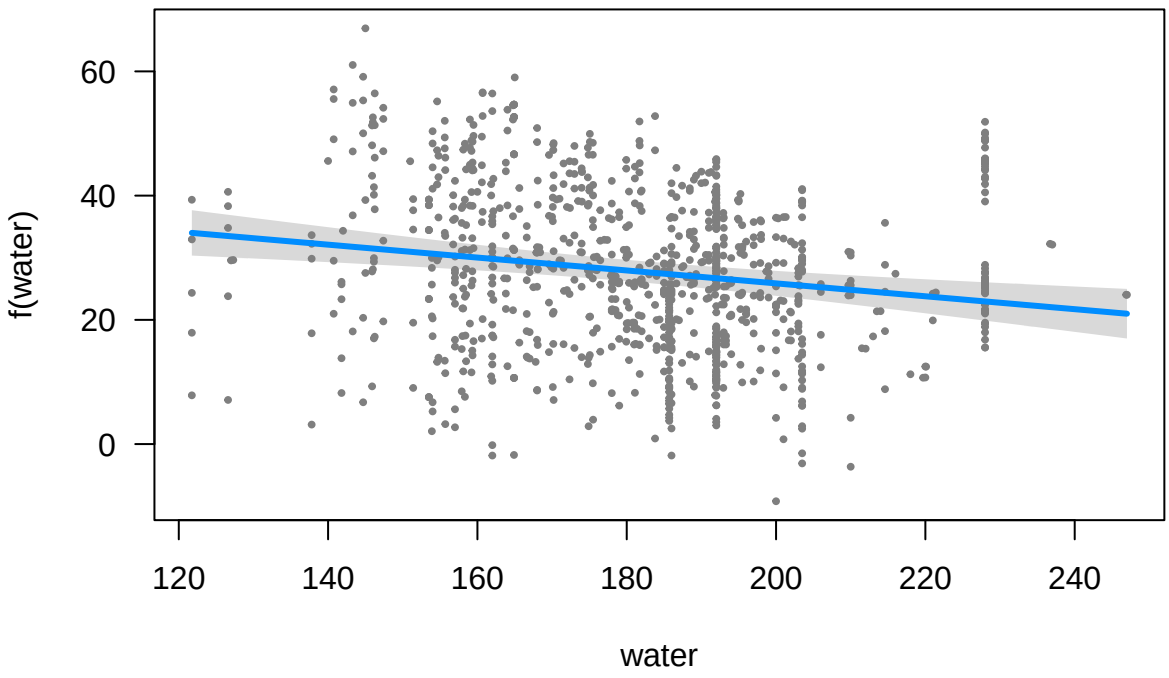
```
##
## Family: gaussian
## Link function: identity
##
## Formula:
## strength ~ cement + blastf + ash + water + super + coarse
##
## Parametric coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  5.326997  10.510518   0.507 0.612387
## cement       0.108256   0.005214  20.761  < 2e-16 ***
## blastf       0.079357   0.006193  12.814  < 2e-16 ***
## ash          0.055928   0.009287   6.022  2.4e-09 ***
## water       -0.103871   0.027796  -3.737 0.000197 ***
## super        0.356016   0.110251   3.229 0.001281 **
## coarse       0.008027   0.006272   1.280 0.200940
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##
## R-sq.(adj) =  0.445   Deviance explained = 44.9%
## GCV = 155.83  Scale est. = 154.77    n = 1030
```

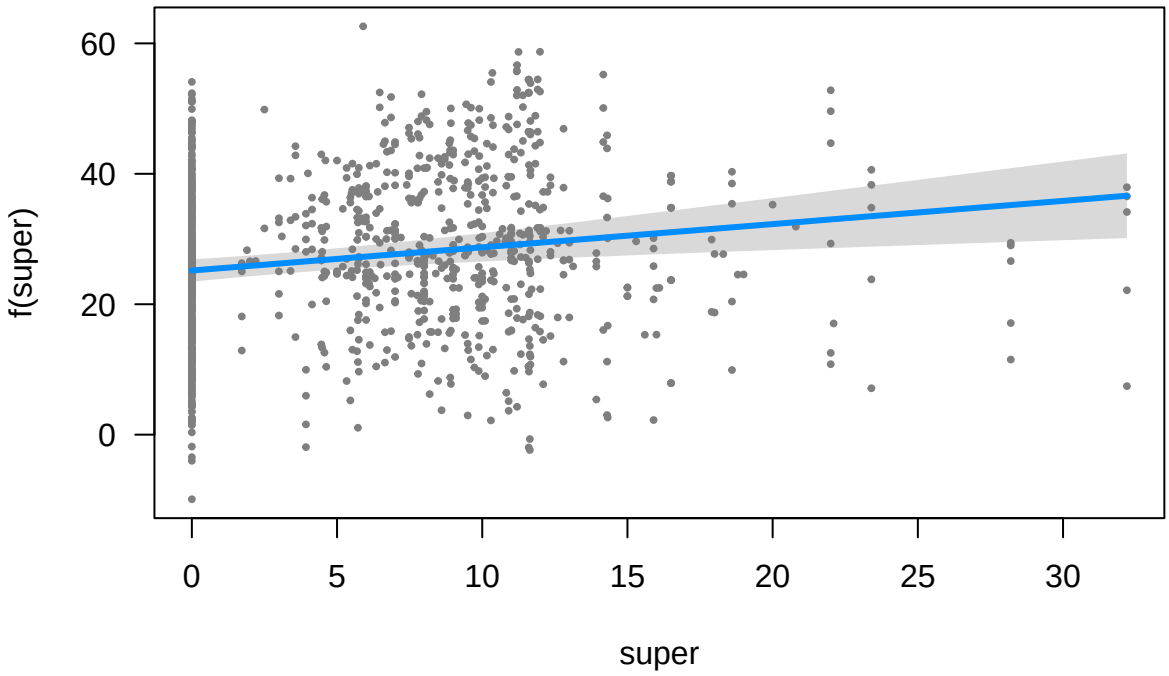
```
visreg(gamfitlinear)
```

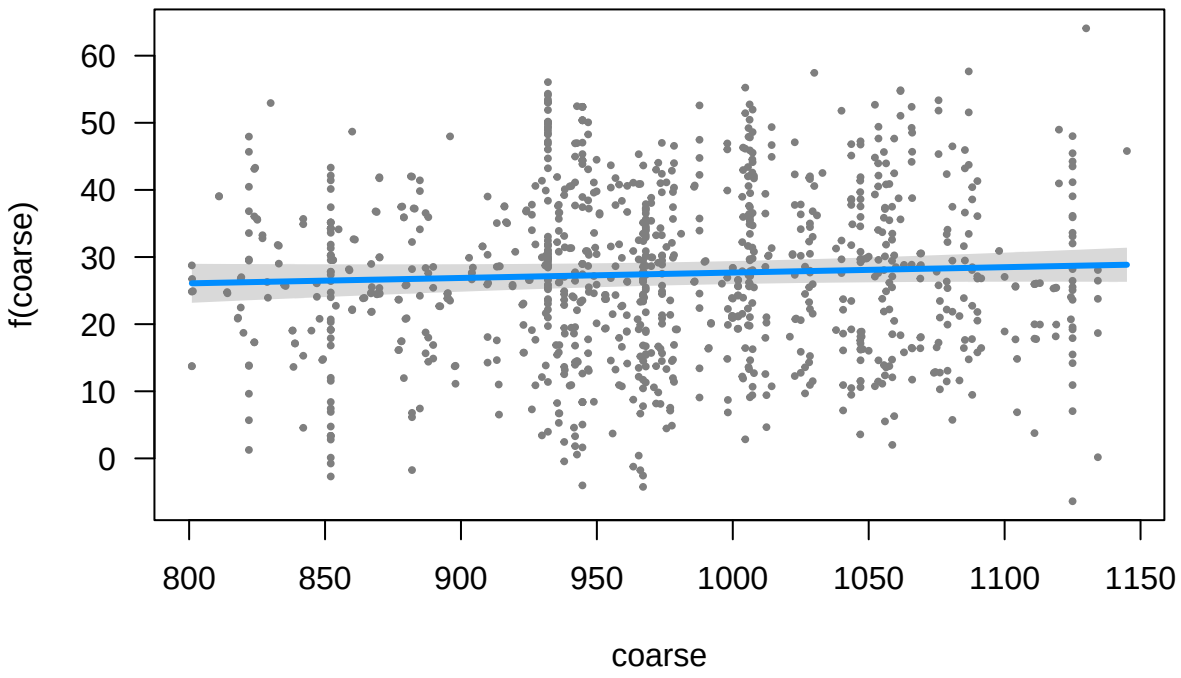



f(blastf)
60
40
20
0
0
50
100
150
200
250
300
350
blastf


60
50
40
30
20
10
0
−10
f(ash)
0
50
100
150
200
ash


60
40
20
0
f(water)
120
140
160
180
200
220
240
water


60
40
20
0
f(super)
0
5
10
15
20
25
30
super



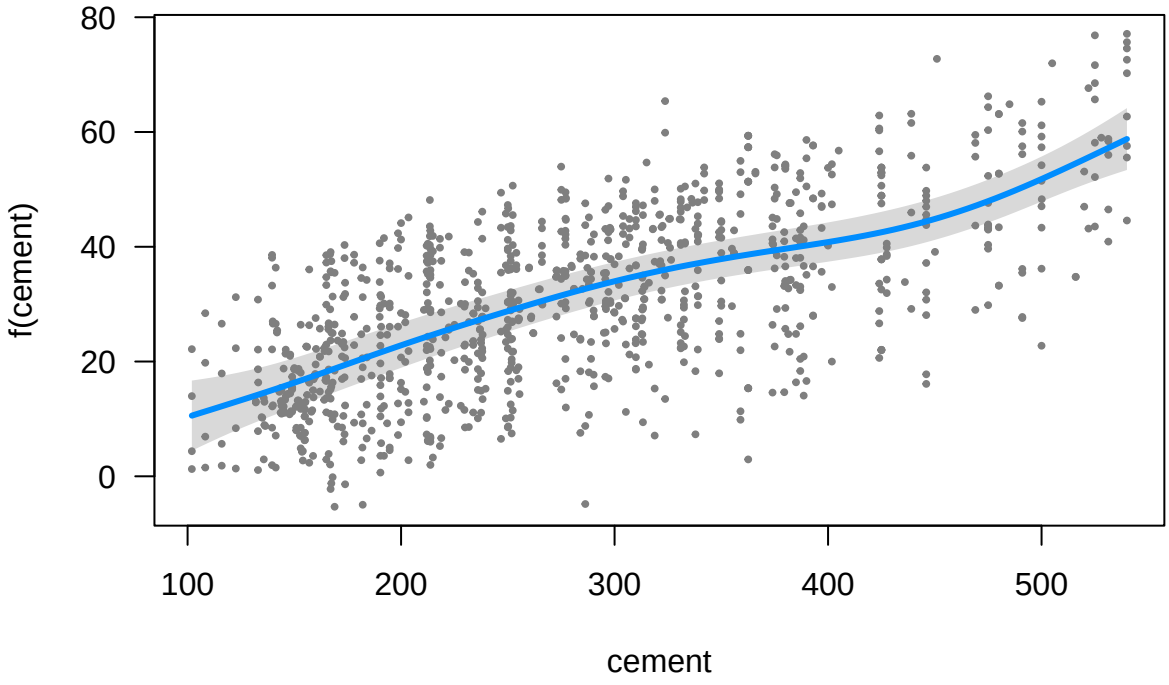
R^2 is 0.445

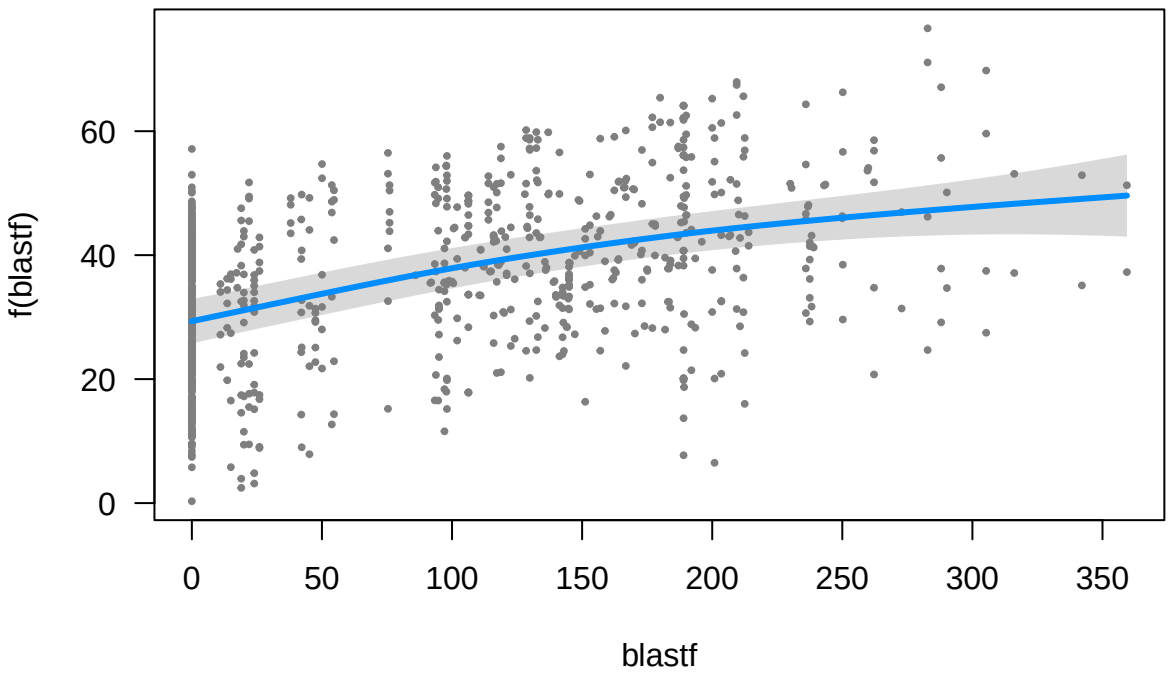
```
gamfitsmooth<-gam(strength~ s(cement)+s(blastf)+s(ash)+s(water)+s(super)
                  +s(coarse),data=concretedf)
summary(gamfitsmooth)
```

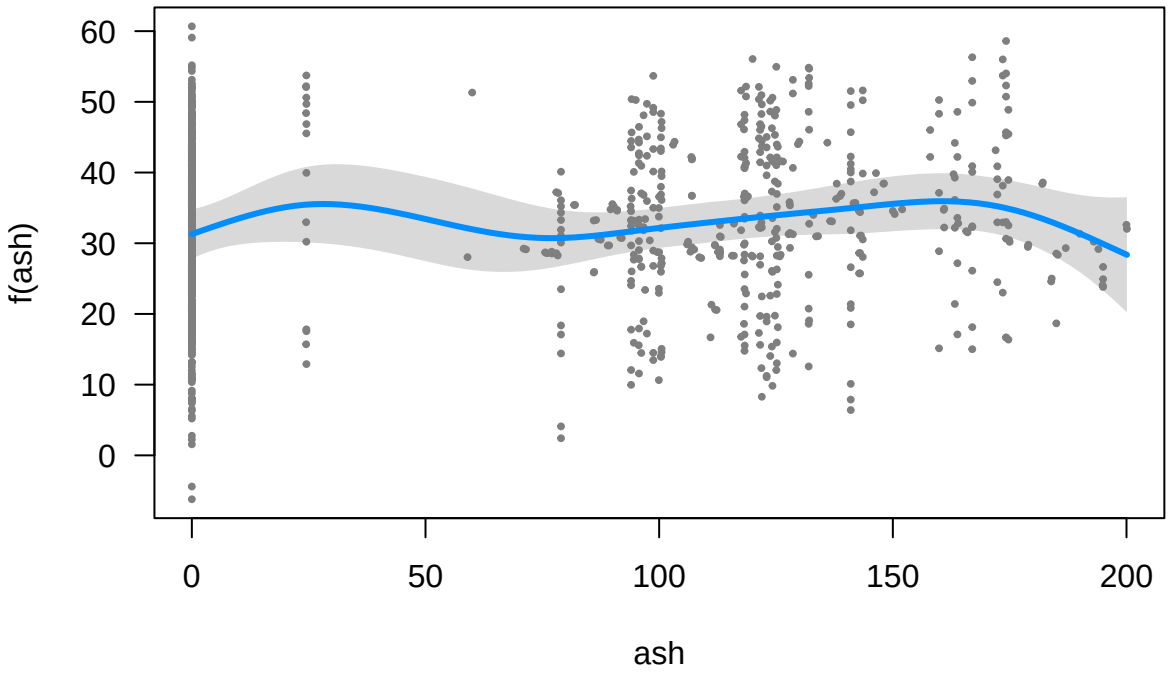
```
##
## Family: gaussian
## Link function: identity
##
## Formula:
## strength ~ s(cement) + s(blastf) + s(ash) + s(water) + s(super) +
##     s(coarse)
##
## Parametric coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  35.8178     0.3566   100.4   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Approximate significance of smooth terms:
##             edf Ref.df      F  p-value
## s(cement) 4.464  5.513 69.530  < 2e-16 ***
## s(blastf) 2.088  2.578 48.091  < 2e-16 ***
## s(ash)    5.332  6.404  1.784    0.101
## s(water)  8.567  8.936 13.504  < 2e-16 ***
```

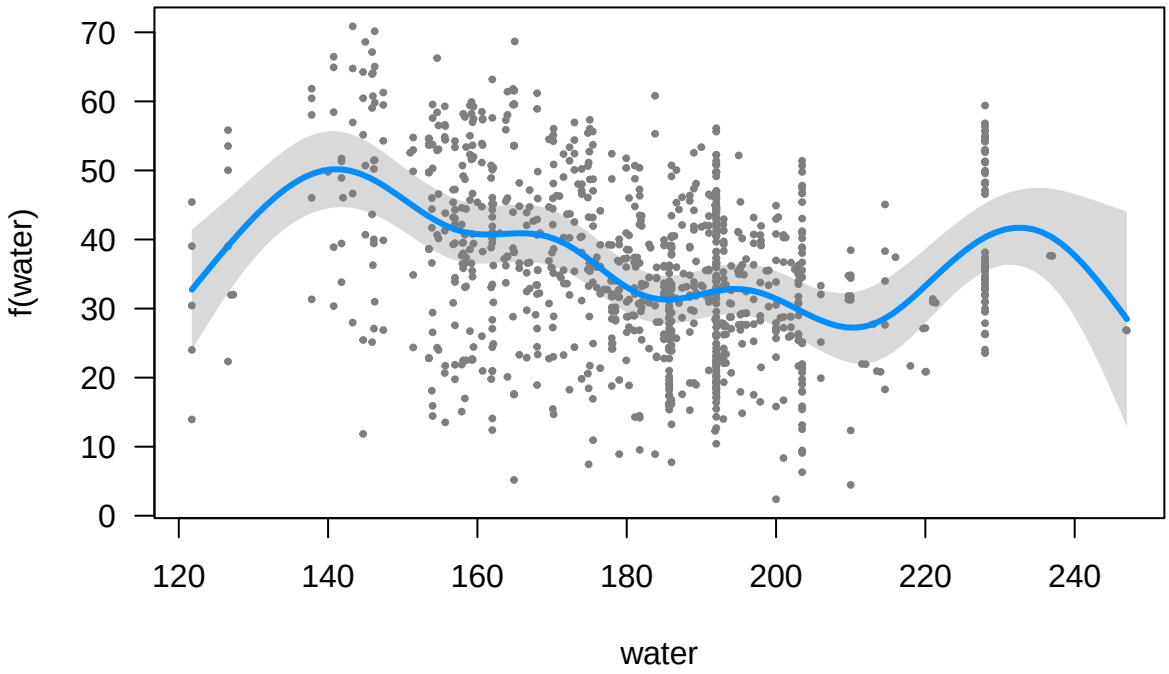
```
## s(super)  7.133  8.143  5.498 1.22e-06 ***
## s(coarse) 1.000  1.000  0.018    0.892
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## R-sq.(adj) =  0.531   Deviance explained = 54.4%
## GCV = 134.84  Scale est. = 130.96    n = 1030
```

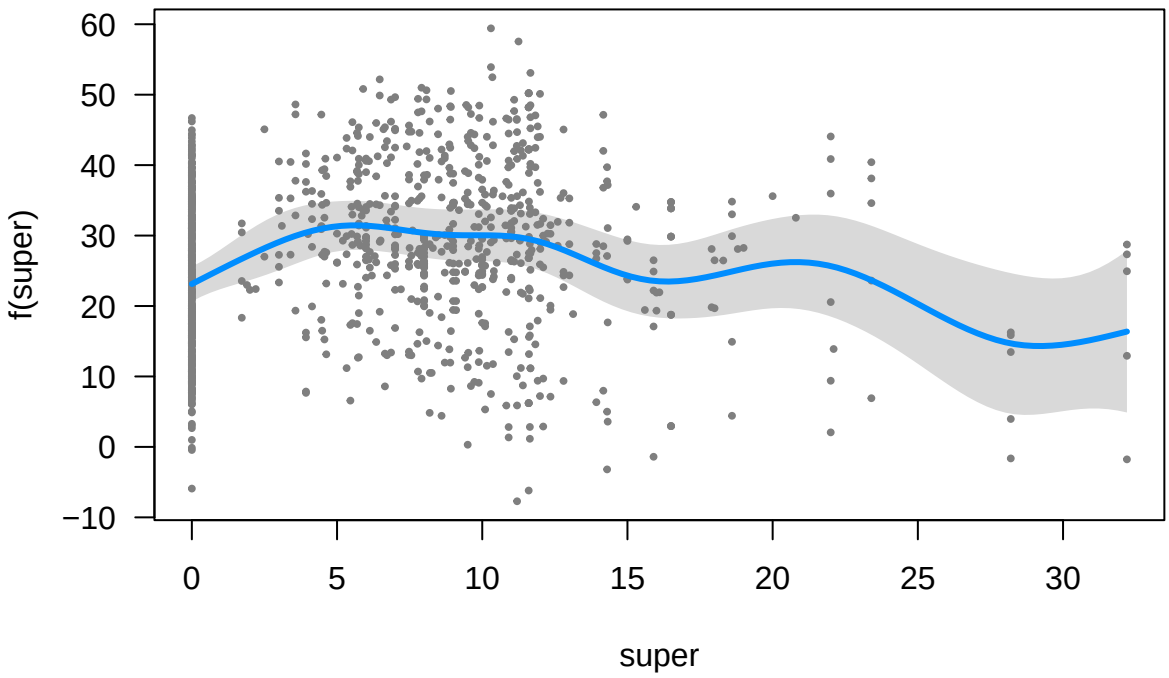
```
visreg(gamfitsmooth)
```

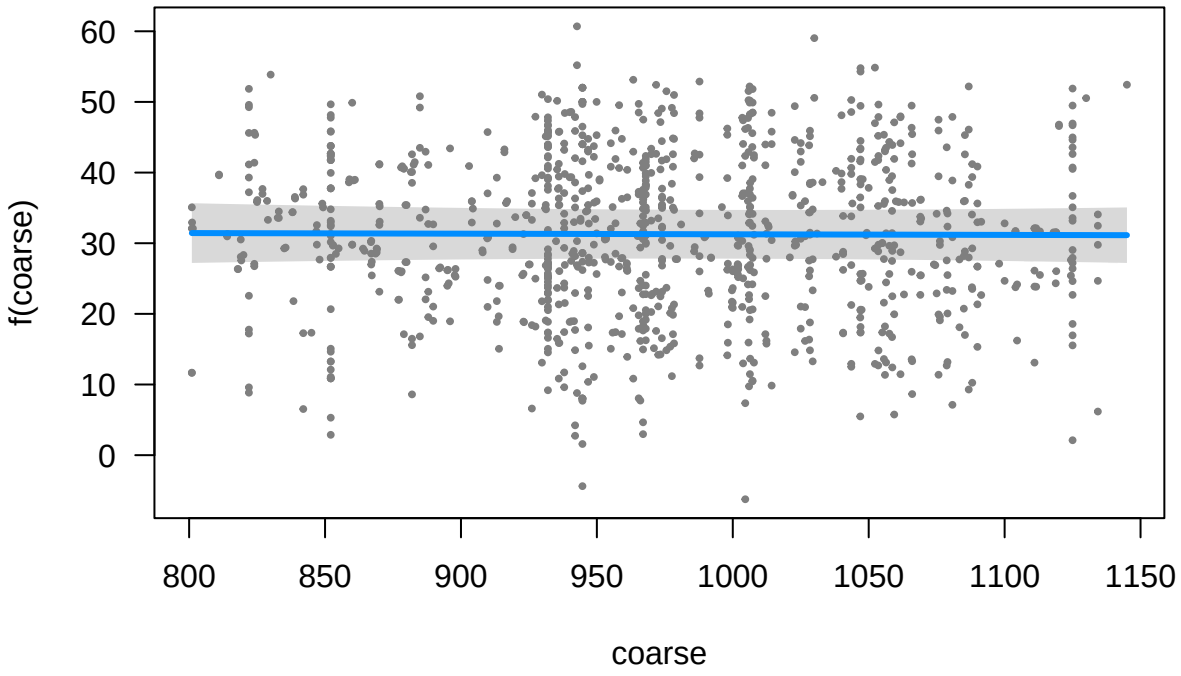



f(blastf)
0
20
40
60
0
50
100
150
200
250
300
350
blastf


60
50
40
30
20
10
0
f(ash)
0
50
100
150
200
ash


f(water)
70
60
50
40
30
20
10
0
120
140
160
180
200
220
240
water


60
50
40
30
20
10
0
−10
f(super)
0
5
10
15
20
25
30
super



R^2 is 0.531 which is an improvement from non-smoothed gam model

First talking about the non smoothed model, the quality of the fit at extreme values were poor especially for the Blast Furnace Slag, Fly Ash, and superplasticizer features as most points were stacked on the 0 value, and as the value of the feature increased there were less points and the lines were not good fits for the extreme values

For the smoothed model, the same problem was present as the extreme values were not fitted well. For the same features as well.

The confidence interval however for the smoothed model certainly is better as it widens at the upper extreme where there is less data like for the superplasticizer feature, but it also narrows down a lot for the lower extreme.

The fitted lines aren't as high quality but the confidence intervals are better for smoothed.