DPA Assignment 3

Girish Rajani

2023-03-05

#Recitation Problems

#Chapter 6

#1.

(a) The best subset selection will have the lowest training RSS because it fits all Pk models that contain exactly k predictors. All combinations of p are taken into consideration here. This is not the case with forward and backward stepwise selection because they are greedy and don't consider all predictors.

(b) For smallest test RSS, it is difficult to say based on the given information because best subset selection considers all predictors and has lowest training RSS so it may also have lowest testing RSS, however, it may overfit. The other two models, since they avoid overfitting, may also find a model that gives the lowest testing RSS.

#(c)

i - True - In forward stepwise selection, we add one additional parameter in Mk and so we select the best model within the predictors of the (k+1) variable

ii - True - In backward stepwise selection, we use the subset of the predictors in the (k+1) variable model and remove one predictor

iii - False - The predictors in backward stepwise selection are only identified by a subset of predictors in the k+1 variable model by the backward stepwise selection because the model will contain all but one predictor, therefore both models end up selecting different predictors

iv - False - The predictors in forward stepwise selection are only identified by a subset of predictors in the k+1 variable model by the forward stepwise selection because the model will contain one additional predictor, therefore both models end up selecting different predictors

v - False - The best subset selection predictors are not identified by a subset from the k+1 variable. It fits all k variables and contains exactly k predictors

#2.

(a)

iii. Since Lasso yields sparse models meaning only a subset of variables is used so the model is less flexible. Additionally, as our lambda increases and we perform shrinkage and variable selection, the variance significantly decreases more than the increase in bias so we have a bias/variance trade off which gives improved prediction accuracy.

(b)

iii. Ridge regression follows the same concept as above but we only perform shrinkage so our features shrink close to 0 but not exactly to 0. This means that as p goes closer to n, the increase in bias is less than the decrease in variance.

(c)

ii. Non-linear models have high flexibility which means that the variance increases and so for improved predictions, we will have a significant decrease in bias and a slight increase in variance.

#3. (a) iv. The training RSS will steadily decrease as we increase s from 0 because as s increases, the least square solution falls within this budget and will become less restrictive. Therefore, the training RSS will steadily decrease until it yields this solution.

(b)

ii. The testing RSS will initially decrease but as it fits the model, flexibility increases which will cause overfitting to occur and result in an increase in RSS.

(c)

iii. The variance will steadily increase as our model flexibilty increases

(d)

iv. The bias will steadily decrease as our model flexibility increases

(e)

v. The irreducible error is not dependent on s therefore it remains constant

#4. (a) iii. The training RSS will steadily increase because an increase in lambda will also increase the penalty term

(b)

ii. The testing RSS will initially decrease but as it fits the model, flexibility decreases which will cause overfitting to occur and result in an increase in RSS.

(c)

iv. The variance will steadily decrease as our model flexibilty decreases

(d)

iii. The bias will steadily increase as our model flexibility decreases

(e)

v. The irreducible error is not dependent on lambda therefore it remains constant

5) a Ridge Regression Optimization:

$$\underset{\hat{\beta}}{\text{minimize}} \left\{ \sum_{i=1}^{n} \left(y_i - \hat{\beta}_0 - \sum_{j=1}^{p} \hat{\beta}_j x_{ij} \right)^2 + \lambda \sum_{j=1}^{p} \hat{\beta}_j^2 \right\}$$

When $\hat{\beta}_0 = 0$, $n=2$, $p=2$, in this setting, we get:

$$\text{minimize} \left\{ \left(y_1 - \hat{\beta}_1 x_{11} - \hat{\beta}_2 x_{12}\right)^2 + \left(y_2 - \hat{\beta}_1 x_{21} - \hat{\beta}_2 x_{22}\right)^2 + \lambda\left(\hat{\beta}_1^2 + \hat{\beta}_2^2\right) \right\}$$

b We can assume that since $x_{11} = x_{12}$, we can sub them as x_1,
" " " " " $x_{21} = x_{22}$, " " " " x_2

Rewrite a →
$$\text{minimize} \left\{ \left(y_1 - \hat{\beta}_1 x_1 - \hat{\beta}_2 x_1\right)^2 + \left(y_2 - \hat{\beta}_1 x_2 - \hat{\beta}_2 x_2\right)^2 + \lambda\left(\hat{\beta}_1^2 + \hat{\beta}_2^2\right) \right)$$

~~the~~ We can tak partial derivative w.r.t β_1 and equate to 0:

$$\hat{\beta}_1 (x_1^2 + x_2^2 + \lambda) + \hat{\beta}_2 (x_1^2 + x_2^2) - y_1 x_1 - y_2 x_2 = 0$$

We can take partial derivative w.r.t β_2 and eqate to 0:

$$\hat{\beta}_1 (x_1^2 + x_2^2) + \hat{\beta}_2 (x_1^2 + x_2^2 + \lambda) - y_1 x_1 - y_2 x_2 = 0$$

$\therefore$

$$\hat{\beta}_1 (x_1^2 + x_2^2 + \lambda) + \hat{\beta}_2 (x_1^2 + x_2^2) \cancel{- y_1 x_1 - y_2 x_2} = \hat{\beta}_1 (x_1^2 + x_2^2) + \hat{\beta}_2 (x_1^2 + x_2^2 + \lambda) \cancel{- y_1 x_1 - y_2 x_2}$$

Simplify:
$$\hat{\beta}_1 \lambda + \cancel{\hat{\beta}_2 (x_1^2 + x_2^2)} = \cancel{\hat{\beta}_2 (x_1^2 + x_2^2)} + \hat{\beta}_2 \lambda$$
$$\hat{\beta}_1 \cancel{\lambda} = \hat{\beta}_2 \cancel{\lambda}$$
$$\hat{\beta}_1 = \hat{\beta}_2$$

#5.

c) Lasso regression Optimization:

$$\text{minimize} \left\{ \sum_{i=1}^{n} \left(y_i - \hat{\beta}_0 - \sum_{j=1}^{p} \hat{\beta}_j x_{ij} \right)^2 + \lambda \sum_{i=1}^{p} |\beta_i| \right\}$$

When $\hat{\beta}_0 = 0$, $n = 2$, $p = 2$, in this setting we get:

$$\text{minimize} \left\{ (y_1 - \hat{\beta}_1 x_{11} - \hat{\beta}_2 x_{12})^2 + (y_2 - \hat{\beta}_1 x_{21} - \hat{\beta}_2 x_{22})^2 + \lambda\left(|\hat{\beta}_1| + |\hat{\beta}_2|\right) \right\}$$

d) Replace the penalty term from Lasso ~~Regression~~ and derivate w.r.t β

$$\partial/\partial\beta \, (\lambda|\beta|) = \frac{\lambda|\beta|}{\beta}$$

We ~~also~~ expand this for β_1 & β_2:

$$\frac{\lambda|\beta_1|}{\beta_1} = \frac{\lambda|\beta_2|}{\beta_2}$$

Assuming that both β_1 and β_2 are either positive or negative.

5c and 5d

#Chapter 7

#2. (a) As lambda goes to infinity, the penalty term is significant, g(x) = 0 and hence g_hat =0

(b) As lambda goes to infinity, and g^1(x) = 0, then g_hat will be a constant

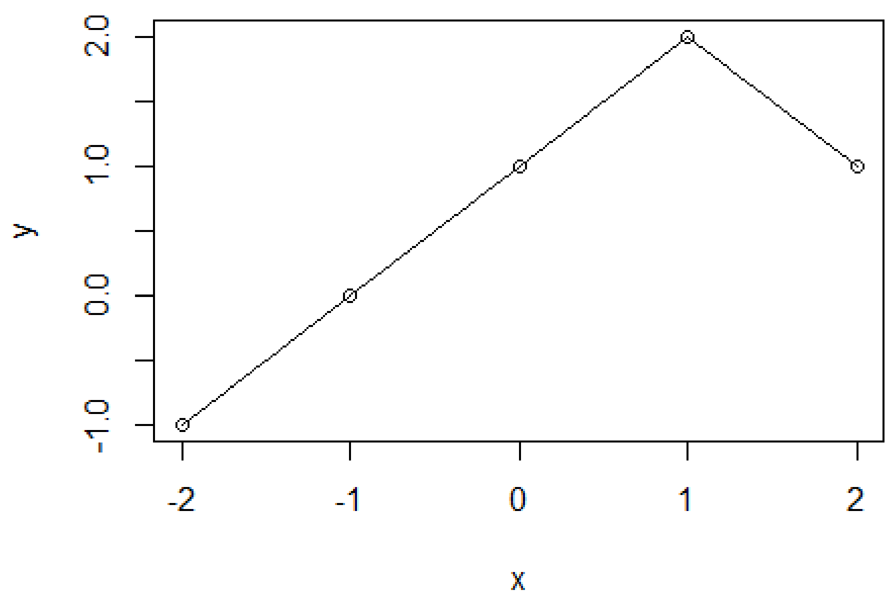
(c) As lambda goes to infinity and m = 2, g^2(x) = 0 which will result in a straight line, a function ax+b

(d) As lambda goes to infinity and m = 3, g^3(x) = 0 which will result in a quadratic plot, a function ax^2 + bx + c

(e) As lambda is 0, there will be no penalty term applied to g_hat. This means that the straight line will be drawn through all points perfectly

#3.

```
library(ggplot2)

x <- -2:2

y <- 1+x+-2*(x-1)^2*I(x>1)

plot(x, y, type="o")
```

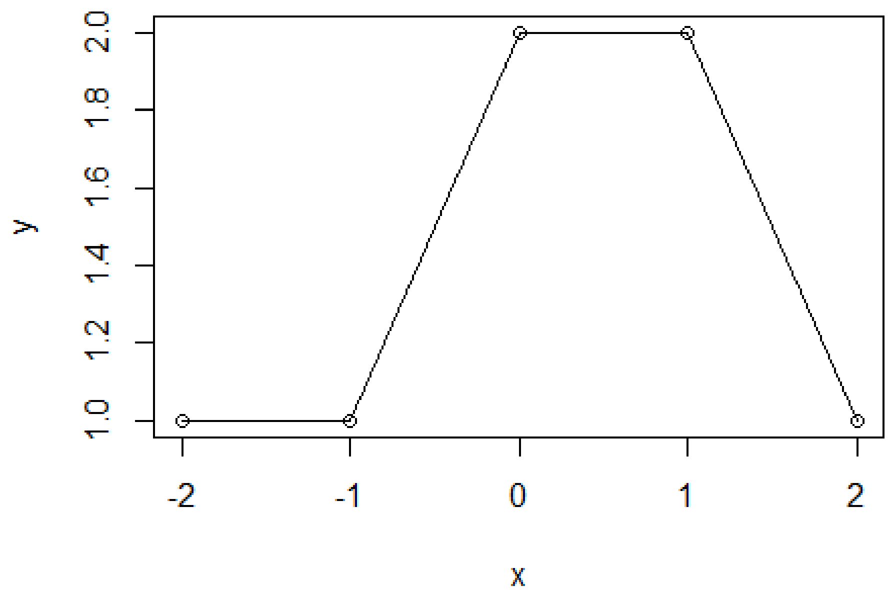


From above, we can see that when X goes from -2 to 1, the estimated curve is linear which results in a slope and y intercept of 1. However, when X is greater than one, we can observe a quadratic estimated curve.

#4.

```
x <- -2:2
y <- c(1 + 0 + 0, # x = -2
 1 + 0 + 0, # x = -1
```

```
 1 + 1 + 0, # x = 0
 1 + (1-0) + 0, # x = 1
 1 + (1-1) + 0 # x =2
)
plot(x,y, type = "o")
```



The estimated curve from above shows a slope of -2

#5. (a) As lambda goes to infinity, g_2_hat will have a smaller training RSS because it is a more flexible model. It is more flexible because it has a higher penalty term of the fourth derivative order when compared to g_1_hat which has a lower penalty term.

(b) As lambda goes to infinity, g_1_hat is more likely to have a smaller training RSS because it is less likely to overfit due to it's lower derivative penalty term. Since g_2_hat is more flexible and has lower training RSS, it may overfit on test RSS.

(c) Since lambda is 0 here, there will be no penalty term applied in neither models. This means that both models will have the same train/test RSS.

```
#Import necessary libraries
data("mtcars")
data("swiss")
library(glmnet)

## Loading required package: Matrix

## Loaded glmnet 4.1-6
```

```
library("readxl")
library(mgcv)

## Loading required package: nlme

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

library(visreg)

#Problem 1

#Load mtcars sample dataset into dataframe
mtcars <- data.frame(mtcars)
set.seed(5)

#Perform a basic 80/20 test-train split
split <- sample(c(rep(0, 0.8 * nrow(mtcars)), rep(1, 0.2 * nrow(mtcars))))
train  <- mtcars[split == 0, ]
test   <- mtcars[split == 1, ]

summary(mtcars)

##       mpg             cyl             disp             hp       
##  Min.   :10.40   Min.   :4.000   Min.   : 71.1   Min.   : 52.0  
##  1st Qu.:15.43   1st Qu.:4.000   1st Qu.:120.8   1st Qu.: 96.5  
##  Median :19.20   Median :6.000   Median :196.3   Median :123.0  
##  Mean   :20.09   Mean   :6.188   Mean   :230.7   Mean   :146.7  
##  3rd Qu.:22.80   3rd Qu.:8.000   3rd Qu.:326.0   3rd Qu.:180.0  
##  Max.   :33.90   Max.   :8.000   Max.   :472.0   Max.   :335.0  
##       drat             wt             qsec             vs        
##  Min.   :2.760   Min.   :1.513   Min.   :14.50   Min.   :0.0000  
##  1st Qu.:3.080   1st Qu.:2.581   1st Qu.:16.89   1st Qu.:0.0000  
##  Median :3.695   Median :3.325   Median :17.71   Median :0.0000  
##  Mean   :3.597   Mean   :3.217   Mean   :17.85   Mean   :0.4375  
##  3rd Qu.:3.920   3rd Qu.:3.610   3rd Qu.:18.90   3rd Qu.:1.0000  
##  Max.   :4.930   Max.   :5.424   Max.   :22.90   Max.   :1.0000  
##        am              gear            carb      
##  Min.   :0.0000   Min.   :3.000   Min.   :1.000  
##  1st Qu.:0.0000   1st Qu.:3.000   1st Qu.:2.000  
##  Median :0.0000   Median :4.000   Median :2.000  
##  Mean   :0.4062   Mean   :3.688   Mean   :2.812  
##  3rd Qu.:1.0000   3rd Qu.:4.000   3rd Qu.:4.000  
##  Max.   :1.0000   Max.   :5.000   Max.   :8.000  

head(mtcars)

##                    mpg cyl disp  hp drat    wt  qsec vs am gear carb
## Mazda RX4         21.0   6  160 110 3.90 2.620 16.46  0  1    4    4
## Mazda RX4 Wag     21.0   6  160 110 3.90 2.875 17.02  0  1    4    4
## Datsun 710        22.8   4  108  93 3.85 2.320 18.61  1  1    4    1
## Hornet 4 Drive    21.4   6  258 110 3.08 3.215 19.44  1  0    3    1
```

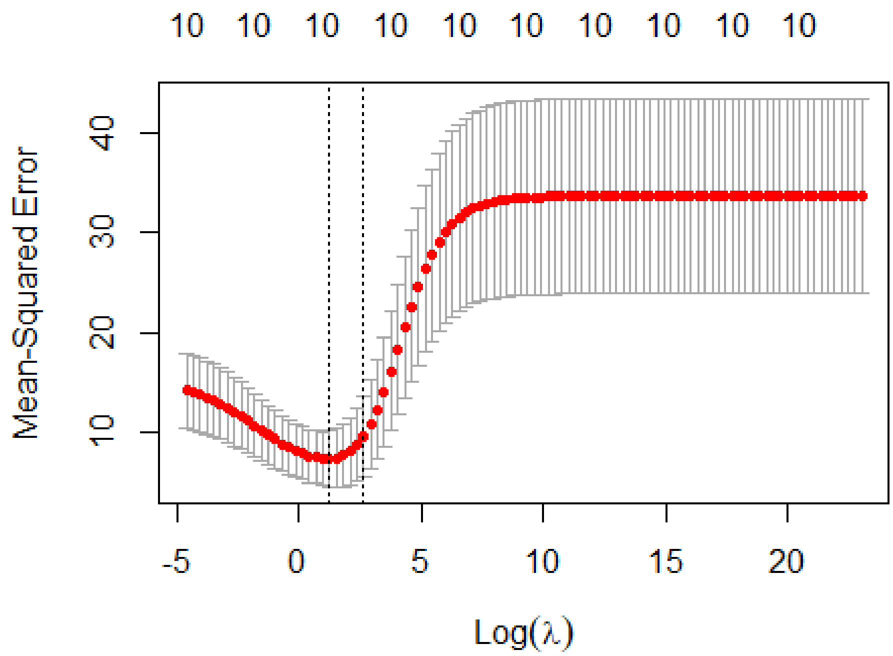
```
## Hornet Sportabout 18.7   8  360 175 3.15 3.440 17.02  0  0    3    2
## Valiant           18.1   6  225 105 2.76 3.460 20.22  1  0    3    1

cat("Dimensions of training data: ", dim(train))

## Dimensions of training data:  26 11

cat("\nDimensions of testing data: ", dim(test))

##
## Dimensions of testing data:  6 11

#fit a linear model using mpg as target response
lm <- lm(mpg ~ ., data=train)
summary (lm)

##
## Call:
## lm(formula = mpg ~ ., data = train)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -3.0238 -1.6448 -0.2359  0.7270  5.6116
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 17.87836   20.11527   0.889    0.388
## cyl         -0.40777    1.14174  -0.357    0.726
## disp         0.01432    0.02184   0.656    0.522
## hp          -0.02233    0.02514  -0.888    0.388
## drat        -0.78630    2.43324  -0.323    0.751
## wt          -3.56516    2.23639  -1.594    0.132
## qsec         0.72054    0.80297   0.897    0.384
## vs           0.51670    2.27364   0.227    0.823
## am           1.87919    2.35810   0.797    0.438
## gear         1.44036    1.85758   0.775    0.450
## carb        -0.16122    0.96871  -0.166    0.870
##
## Residual standard error: 2.698 on 15 degrees of freedom
## Multiple R-squared:  0.8643, Adjusted R-squared:  0.7739
## F-statistic: 9.555 on 10 and 15 DF,  p-value: 7.622e-05
```

#Explain what features are relevant based on t-statistic #A smaller t-value indicates high similarity/relationship between two features.

#Based on the t-value observations from the linear model fit, we can say that features hp and wt (with wt being the most relevant) are selected as the most relevant features.

```
#What are the associated coefficient values for relevant features?
lm$coefficients
```

```
## (Intercept)         cyl        disp          hp        drat          wt
## 17.87835971 -0.40776819  0.01432014 -0.02233375 -0.78629698 -3.56515667
##        qsec          vs          am        gear        carb
##  0.72054326  0.51670020  1.87919113  1.44035680 -0.16121614

#The out-of-sample test set performance (using predict) on the regular linear
model:
y_hat <- predict(lm, test, type = "response")
y <- test[ , "mpg"]
MSE <- mean((y_hat - y)^2)
cat("OOS MSE on testing data: ", MSE)

## OOS MSE on testing data:  8.823345

#Perform a ridge regression using glmnet

#pass x matrix and y vector:
x <- model.matrix(mpg ~ ., data=train )[, -1]
y <- train$mpg

#Perform ridge regression using 100 values of lambda for tuning
grid <- 10^seq(10, -2, length = 100)
ridge.mod <- cv.glmnet(x, y, alpha = 0, lambda = grid, parallel=TRUE)

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

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

min_lambda <- ridge.mod$lambda.min
cat("Minimum value of Lambda: ", min_lambda)

## Minimum value of Lambda:  3.511192

#Plot training MSE as a function of log Lambda
plot(ridge.mod)
```




```
# Fitting Ridge Regression with optimal lambda
ridge.mod2 <- glmnet(x, y, alpha = 0, lambda = min_lambda)

coef(ridge.mod2)

## 11 x 1 sparse Matrix of class "dgCMatrix"
##                      s0
## (Intercept) 21.58560850
## cyl         -0.41252847
## disp        -0.00530685
## hp          -0.01037774
## drat         0.75754158
## wt          -1.08253237
## qsec         0.13080551
## vs           0.86473981
## am           1.24208873
## gear         0.63172887
## carb        -0.45619319
```

#By observing the coefficients in the regular linear model and the ridge regression model from above, the coefficients have come closer to zero. Since none of the coefficients are zero, ridge reggression here does not perform variable selection but instead performs shrinkage.

```
#pass x matrix on testing
x_test <- model.matrix(mpg~. ,test)[,-1]
```

```r
# Predict out-of-sample test set performance
y_hat_test <- predict(ridge.mod2, s = min_lambda, newx = x_test)
y_test <- test[, "mpg"]
test_MSE <- mean((y_hat_test - y_test)^2)

cat("Out-of-Sample test set performance MSE: ", test_MSE)

## Out-of-Sample test set performance MSE:  10.14883
```

#By observing the out of sample test set performance MSE from the regular linear model (3.97) and the ridge regression model (2.33), the MSE has decreased which shows that the ridge regression model has performed better. After observing the coefficients and the MSE of both models, we can say that the ridge regression model has performed shrinkage.

```r
#Problem 2

#Load swiss sample dataset into dataframe
swiss <- data.frame(swiss)
set.seed(5)

#Perform a 80/20 test-train split
split <- sample(c(rep(0, 0.8 * nrow(swiss)), rep(1, 0.2 * nrow(swiss))))
train <- swiss[split == 0, ]
test <- swiss[split == 1, ]

summary(swiss)

##    Fertility      Agriculture     Examination      Education    
##  Min.   :35.00   Min.   : 1.20   Min.   : 3.00   Min.   : 1.00  
##  1st Qu.:64.70   1st Qu.:35.90   1st Qu.:12.00   1st Qu.: 6.00  
##  Median :70.40   Median :54.10   Median :16.00   Median : 8.00  
##  Mean   :70.14   Mean   :50.66   Mean   :16.49   Mean   :10.98  
##  3rd Qu.:78.45   3rd Qu.:67.65   3rd Qu.:22.00   3rd Qu.:12.00  
##  Max.   :92.50   Max.   :89.70   Max.   :37.00   Max.   :53.00  
##     Catholic       Infant.Mortality
##  Min.   :  2.150   Min.   :10.80   
##  1st Qu.:  5.195   1st Qu.:18.15   
##  Median : 15.140   Median :20.00   
##  Mean   : 41.144   Mean   :19.94   
##  3rd Qu.: 93.125   3rd Qu.:21.70   
##  Max.   :100.000   Max.   :26.60   

head(swiss)

##              Fertility Agriculture Examination Education Catholic
## Courtelary        80.2        17.0          15        12     9.96
## Delemont          83.1        45.1           6         9    84.84
## Franches-Mnt      92.5        39.7           5         5    93.40
## Moutier           85.8        36.5          12         7    33.77
## Neuveville        76.9        43.5          17        15     5.16
```

```
## Porrentruy        76.1        35.3           9         7    90.57
##                Infant.Mortality
## Courtelary                 22.2
## Delemont                   22.2
## Franches-Mnt               20.2
## Moutier                    20.3
## Neuveville                 20.6
## Porrentruy                 26.6
```

```
cat("Dimensions of training data: ", dim(train))
```

```
## Dimensions of training data:  38 6
```

```
cat("\nDimensions of testing data: ", dim(test))
```

```
##
## Dimensions of testing data:  9 6
```

```
#fit a linear model using Fertility as target response
lm <- lm(Fertility ~ ., data=train)
summary (lm)
```

```
##
## Call:
## lm(formula = Fertility ~ ., data = train)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -15.8999  -4.8833   0.3608   4.4918  14.6768
##
## Coefficients:
##                  Estimate Std. Error t value Pr(>|t|)
## (Intercept)      66.13831   11.75847   5.625 3.23e-06 ***
## Agriculture      -0.15836    0.07813  -2.027 0.051092 .
## Examination      -0.26886    0.32164  -0.836 0.409404
## Education        -0.82147    0.21968  -3.739 0.000724 ***
## Catholic          0.10987    0.04499   2.442 0.020312 *
## Infant.Mortality  1.05061    0.41608   2.525 0.016722 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 7.297 on 32 degrees of freedom
## Multiple R-squared:  0.7224, Adjusted R-squared:  0.679
## F-statistic: 16.65 on 5 and 32 DF,  p-value: 4.262e-08
```

#Explain what features are relevant based on t-statistic #A smaller t-value indicates similarity/relationship between two features.

#Based on the t-value observations from the linear model fit, we can say that features Education and Agriculture (with Education being the most relevant) are selected as the most relevant features.

```r
#What are the associated coefficient values for relevant features?
lm$coefficients
```

```
##      (Intercept)      Agriculture      Examination        Education
##       66.1383136       -0.1583572       -0.2688619       -0.8214705
##         Catholic Infant.Mortality
##        0.1098728        1.0506114
```

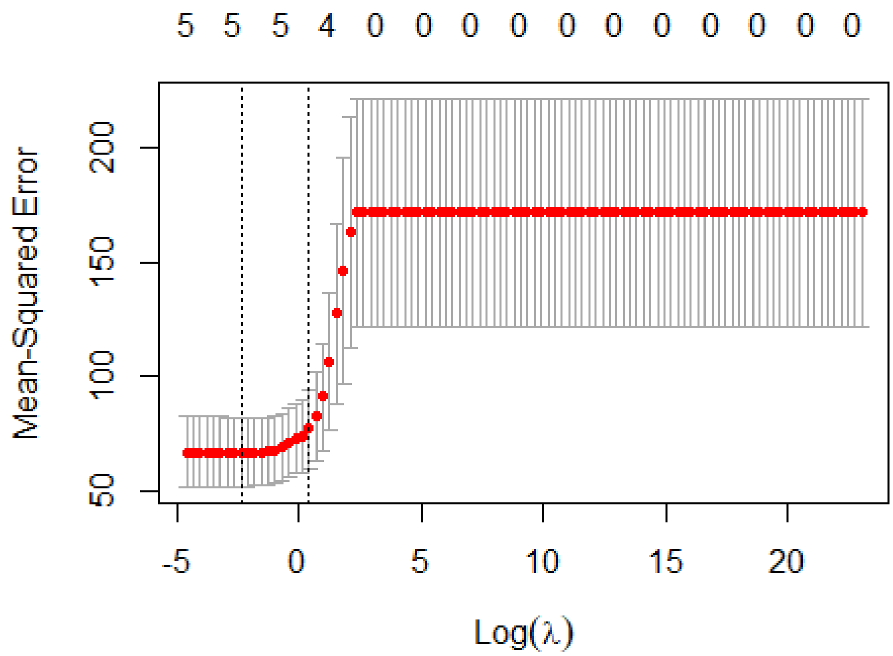
```r
#The out-of-sample test set performance (using predict) on the regular linear
model:
y_hat <- predict(lm, test, type = "response")
y <- test[ , "Fertility"]
MSE <- mean((y_hat - y)^2)
cat("OOS MSE on testing data: ", MSE)
```

```
## OOS MSE on testing data:  45.75983
```

```r
#Perform a lasso regression using glmnet

#pass x matrix and y vector:
x <- model.matrix(Fertility ~ ., data=train )[, -1]
y <- train$Fertility

#Perform lasso regression using 100 values of lambda for tuning
grid <- 10^seq(10, -2, length = 100)
lasso.mod <- cv.glmnet(x, y, alpha = 1, lambda = grid, parallel=TRUE)

min_lambda <- lasso.mod$lambda.min
cat("Minimum value of Lambda: ", min_lambda)
```

```
## Minimum value of Lambda:  0.09326033
```

```r
#Plot training MSE as a function of log Lambda
plot(lasso.mod)
```




```r
# Fitting Lasso Regression with optimal lambda
lasso.mod2 <- glmnet(x, y, alpha = 1, lambda = min_lambda)

coef(lasso.mod2)

## 6 x 1 sparse Matrix of class "dgCMatrix"
##                          s0
## (Intercept)      65.0640375
## Agriculture      -0.1417047
## Examination      -0.2608186
## Education        -0.7940812
## Catholic          0.1057848
## Infant.Mortality  1.0498219
```

#By observing the coefficients in the regular linear model and the lasso regression model from above, there isn't much difference, however, some shrinkage is observed.

#Typically, with lasso, we would expect to see shrinkage and then variable selection occuring. However, since none of the coefficients have shrunk to 0, we can say that our lasso model only performs shrinkage and not both shrinkage and variable selection.

```r
#pass x matrix on testing
x_test <- model.matrix(Fertility~. ,test)[,-1]

# Predict out-of-sample test set performance
y_hat_test <- predict(lasso.mod2, s = min_lambda, newx = x_test)
y_test <- test[, "Fertility"]
```

```r
test_MSE <- mean((y_hat_test - y_test)^2)

cat("Out-of-Sample test set performance MSE: ", test_MSE)
```

```
## Out-of-Sample test set performance MSE:  46.35392
```

#By observing the out of sample test set performance MSE from the regular linear model (49.85) and the lasso regression model (47.95), the MSE has slightly decreased which shows that the lasso regression model has performed slightly better.

```r
#Problem 3

#Load Concrete Compressive Strength sample dataset from UCI Repository into dataframe using readxl package
set.seed(5)
CCS_data <- read_excel("D:\\Users\\giris\\Downloads\\Concrete_Data.xls")
colnames(CCS_data) <- c("C1","C2","C3","C4","C5","C6","C7","C8","CCS")

summary(CCS_data)
```

```
##        C1              C2              C3               C4       
##  Min.   :102.0   Min.   :  0.0   Min.   :  0.00   Min.   :121.8  
##  1st Qu.:192.4   1st Qu.:  0.0   1st Qu.:  0.00   1st Qu.:164.9  
##  Median :272.9   Median : 22.0   Median :  0.00   Median :185.0  
##  Mean   :281.2   Mean   : 73.9   Mean   : 54.19   Mean   :181.6  
##  3rd Qu.:350.0   3rd Qu.:142.9   3rd Qu.:118.27   3rd Qu.:192.0  
##  Max.   :540.0   Max.   :359.4   Max.   :200.10   Max.   :247.0  
##        C5               C6               C7              C8        
##  Min.   : 0.000   Min.   : 801.0   Min.   :594.0   Min.   :  1.00  
##  1st Qu.: 0.000   1st Qu.: 932.0   1st Qu.:731.0   1st Qu.:  7.00  
##  Median : 6.350   Median : 968.0   Median :779.5   Median : 28.00  
##  Mean   : 6.203   Mean   : 972.9   Mean   :773.6   Mean   : 45.66  
##  3rd Qu.:10.160   3rd Qu.:1029.4   3rd Qu.:824.0   3rd Qu.: 56.00  
##  Max.   :32.200   Max.   :1145.0   Max.   :992.6   Max.   :365.00  
##       CCS        
##  Min.   : 2.332  
##  1st Qu.:23.707  
##  Median :34.443  
##  Mean   :35.818  
##  3rd Qu.:46.136  
##  Max.   :82.599
```

```r
head(CCS_data)
```

```
## # A tibble: 6 × 9
##      C1    C2    C3    C4    C5    C6    C7    C8   CCS
##   <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl> <dbl>
## 1  540      0     0   162   2.5  1040   676    28  80.0
## 2  540      0     0   162   2.5  1055   676    28  61.9
## 3  332.   142.    0   228   0     932   594   270  40.3
## 4  332.   142.    0   228   0     932   594   365  41.1
```

```
## 5  199.  132.     0   192   0    978.  826.   360  44.3
## 6  266   114      0   228   0    932   670     90  47.0
```

```r
#Using gam function to predict the CCS as a non-linear function of the input features C1-C6

GAM_non_linear <- gam(CCS ~ C1+C2+C3+C4+C5+C6, data=CCS_data)
summary(GAM_non_linear)
```

```
##
## Family: gaussian
## Link function: identity
##
## Formula:
## CCS ~ C1 + C2 + C3 + C4 + C5 + C6
##
## Parametric coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  5.326997  10.510518   0.507 0.612387
## C1           0.108256   0.005214  20.761  < 2e-16 ***
## C2           0.079357   0.006193  12.814  < 2e-16 ***
## C3           0.055928   0.009287   6.022  2.4e-09 ***
## C4          -0.103871   0.027796  -3.737 0.000197 ***
## C5           0.356016   0.110251   3.229 0.001281 **
## C6           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
```

#With an adjusted R^2 of 0.445 from this model, only 45% of the variance is explained in this model which means that there is not a strong correlation between the predictors and response variables.

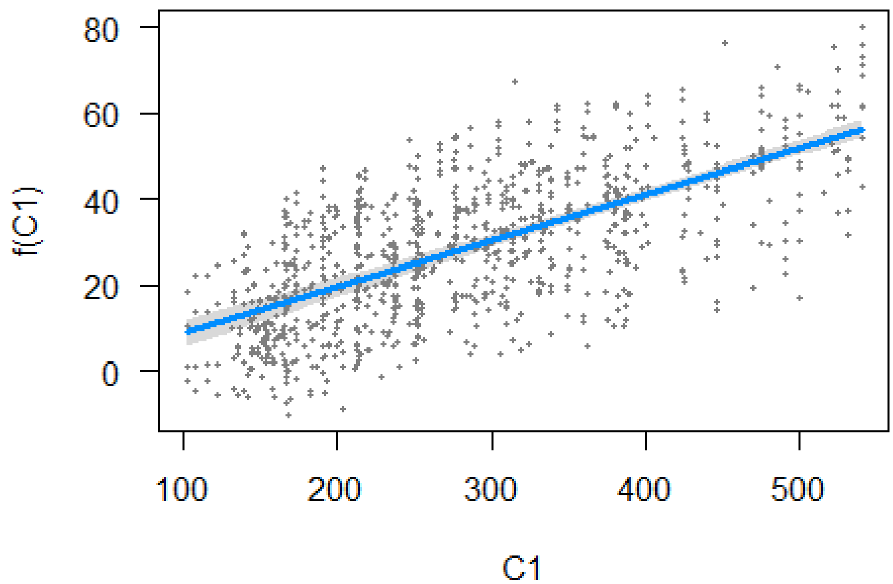
```r
#Creating a GAM with linear terms as well as smoothed terms
GAM_linear <- gam(CCS ~ s(C1)+s(C2)+s(C3)+s(C4)+s(C5)+s(C6), data=CCS_data)
summary(GAM_linear)
```

```
##
## Family: gaussian
## Link function: identity
##
## Formula:
## CCS ~ s(C1) + s(C2) + s(C3) + s(C4) + s(C5) + s(C6)
##
## Parametric coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  35.8178     0.3566   100.4   <2e-16 ***
## ---
```

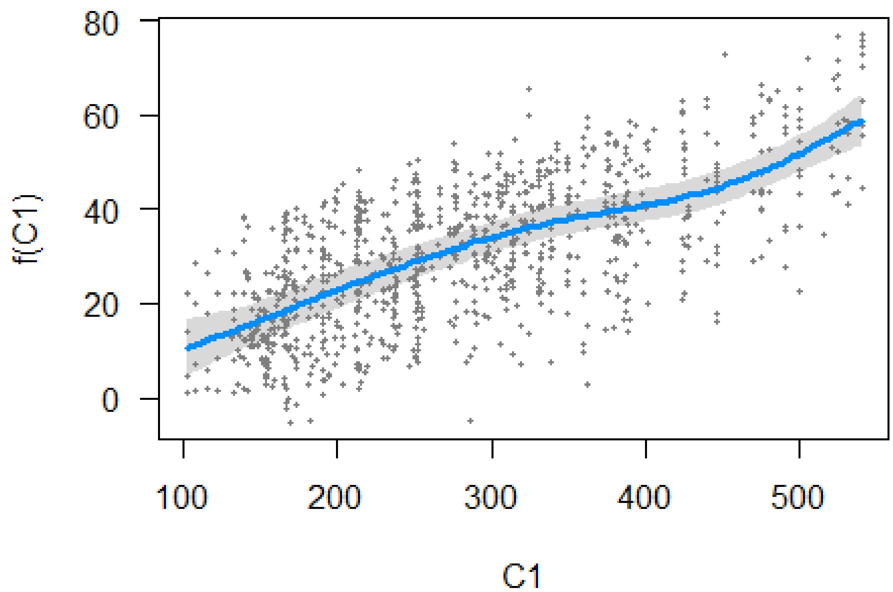
```
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Approximate significance of smooth terms:
##         edf Ref.df      F  p-value    
## s(C1) 4.464  5.513 69.530  < 2e-16 ***
## s(C2) 2.088  2.578 48.091  < 2e-16 ***
## s(C3) 5.332  6.404  1.784    0.101    
## s(C4) 8.567  8.936 13.504  < 2e-16 ***
## s(C5) 7.133  8.143  5.498 1.22e-06 ***
## s(C6) 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
```

#With an adjusted R^2 of 0.531, this model suggests that a little bit more than half of the variance in the outcome data is explained by the model

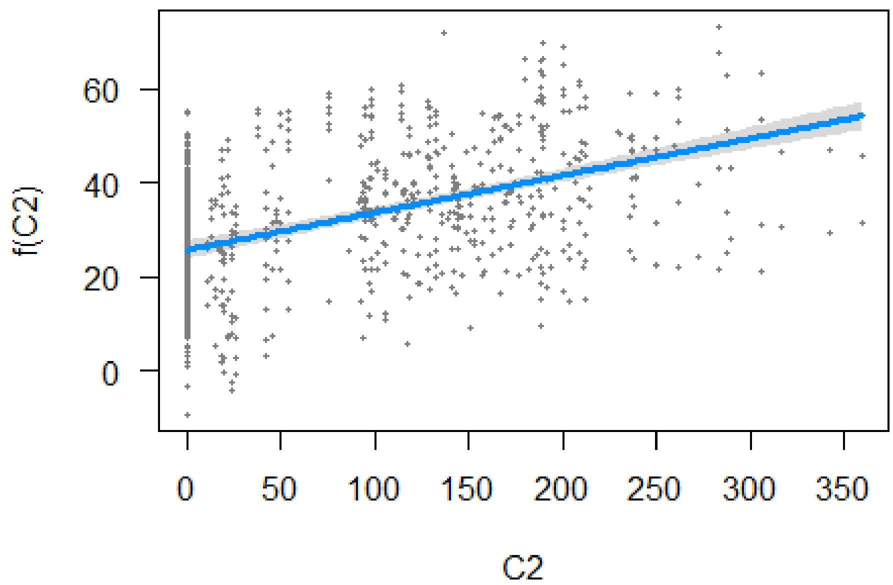
```
#Compute R^2 value for initial non linear model
GAM_non_linear.sse <- sum(fitted(GAM_non_linear) - CCS_data$CCS)^2
GAM_non_linear.ssr <- sum(fitted(GAM_non_linear) - mean(CCS_data$CCS))^2
GAM_non_linear.sst <- GAM_non_linear.sse + GAM_non_linear.ssr

GAM_non_linear.r2 <- 1 - (GAM_non_linear.sse/GAM_non_linear.sst)

cat("R-square value of initial GAM non linear model:", GAM_non_linear.r2)

## R-square value of initial GAM non linear model: 0.4967177

#Compute R^2 value for the linear model with smoothing
GAM_linear.sse <- sum(fitted(GAM_linear) - CCS_data$CCS)^2
GAM_linear.ssr <- sum(fitted(GAM_linear) - mean(CCS_data$CCS))^2
GAM_linear.sst <- GAM_linear.sse + GAM_linear.ssr

GAM_linear.r2 <- 1 - (GAM_linear.sse/GAM_linear.sst)

cat("R-square value of GAM with smoothing:", GAM_linear.r2)

## R-square value of GAM with smoothing: 0.5000744

#Visualize the regression
visreg(GAM_non_linear,'C1')
```



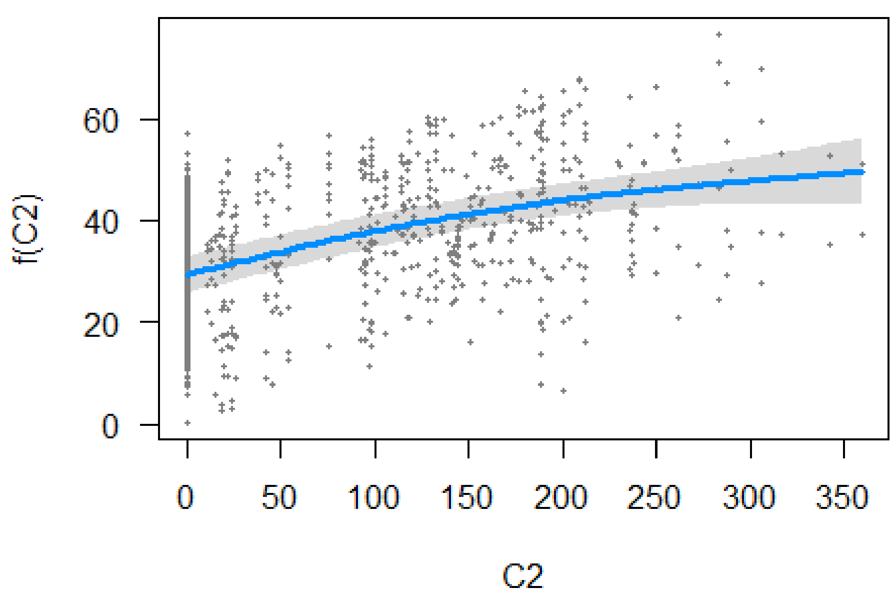

```
visreg(GAM_linear,'C1')
```



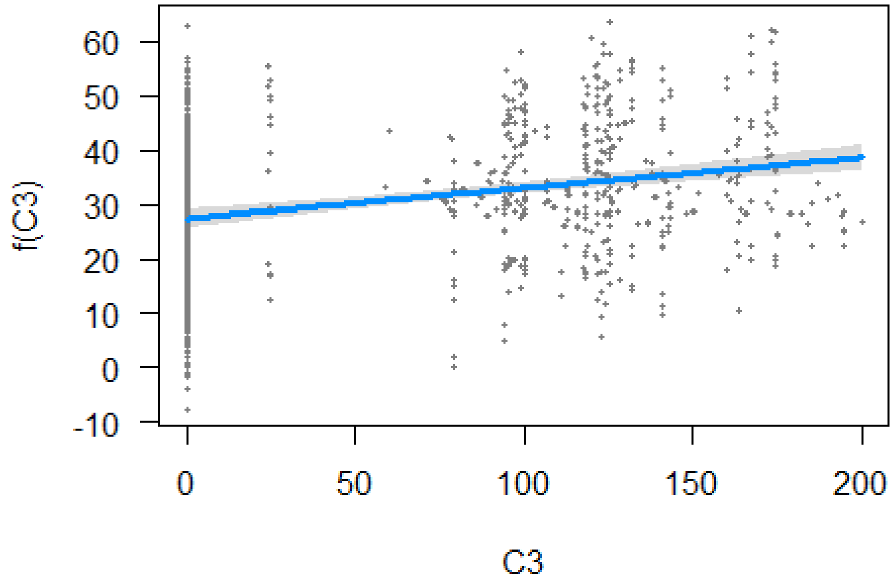

```
visreg(GAM_non_linear,'C2')
```



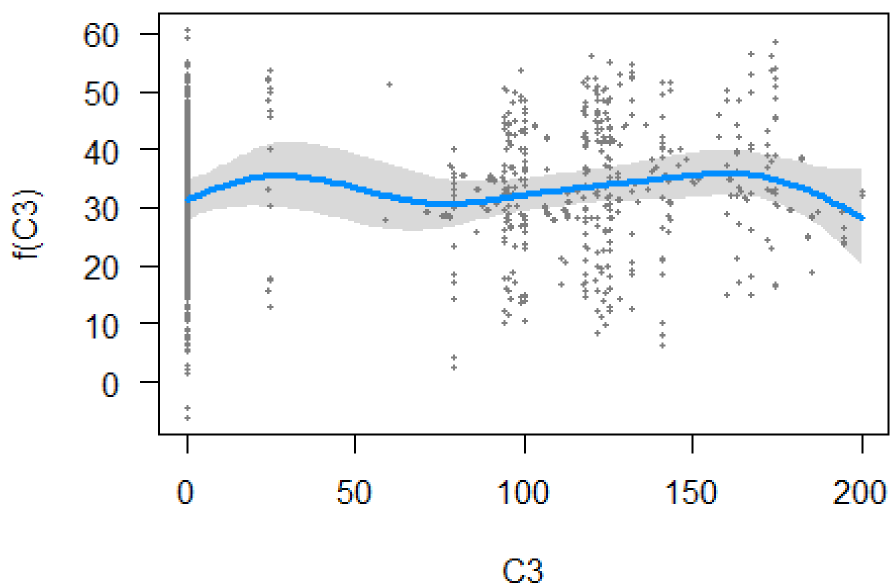

```
visreg(GAM_linear,'C2')
```



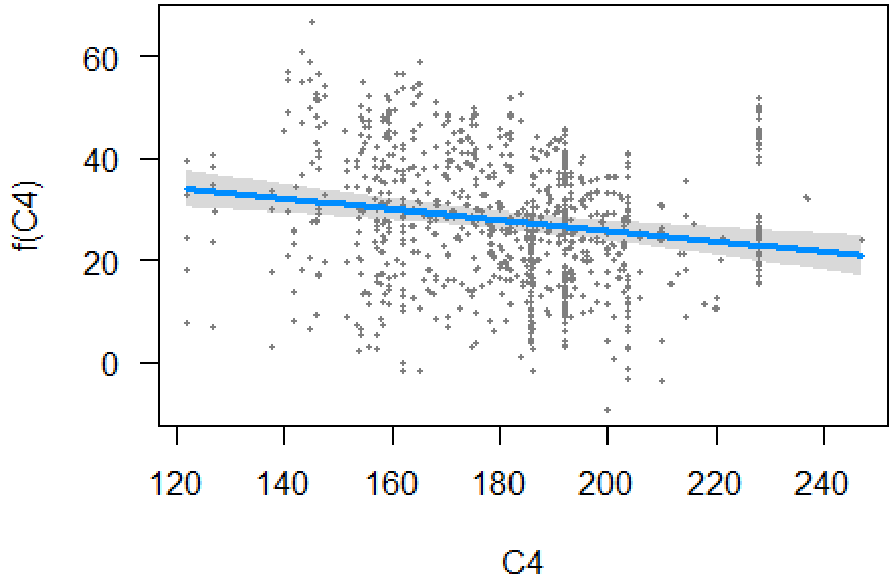

```
visreg(GAM_non_linear,'C3')
```



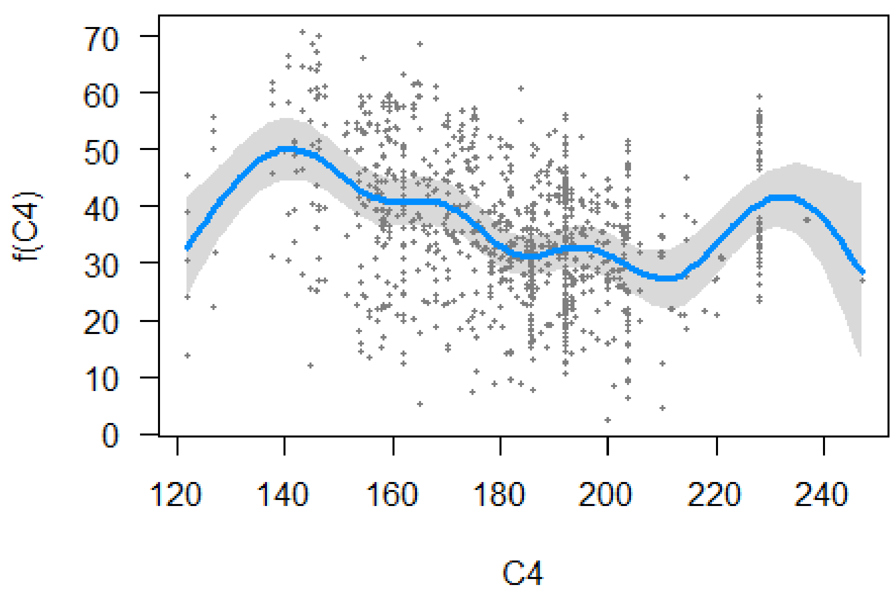

```
visreg(GAM_linear,'C3')
```



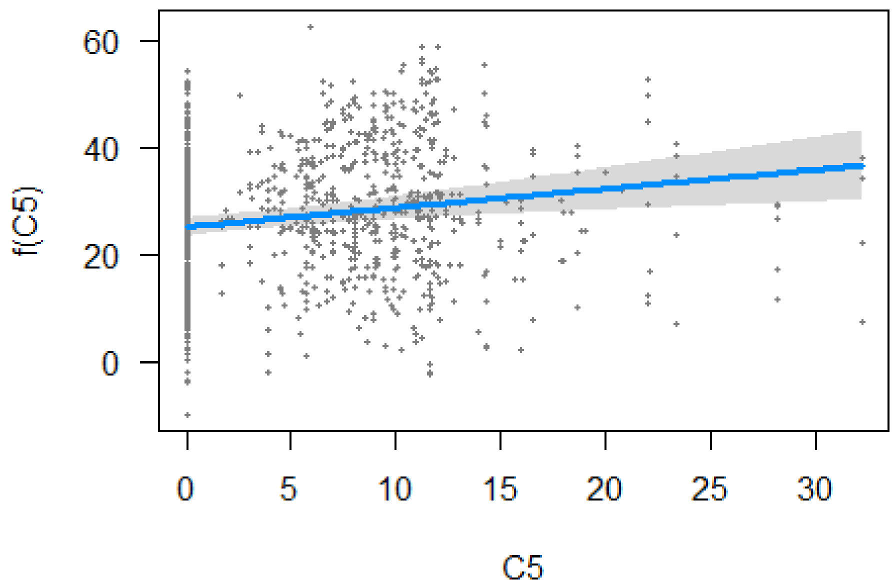

```
visreg(GAM_non_linear,'C4')
```



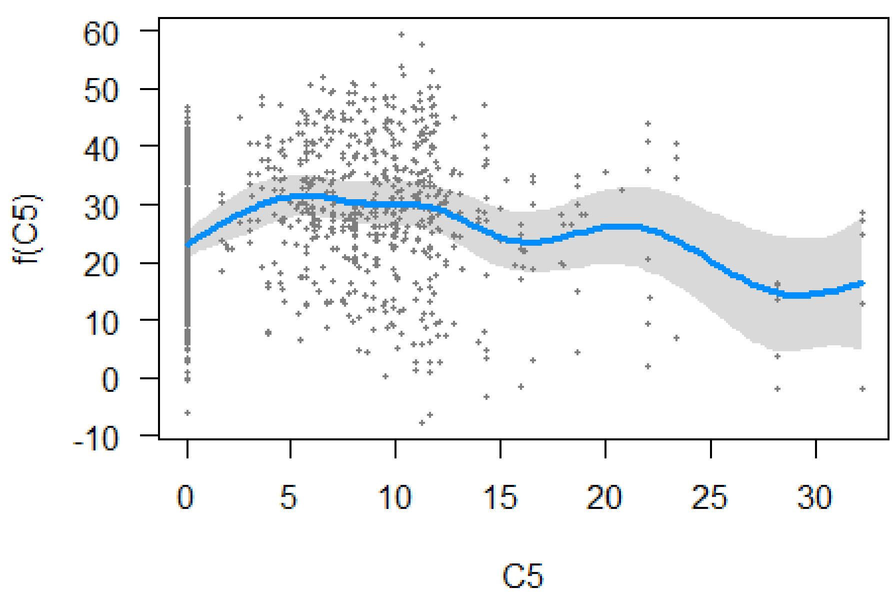

```
visreg(GAM_linear,'C4')
```



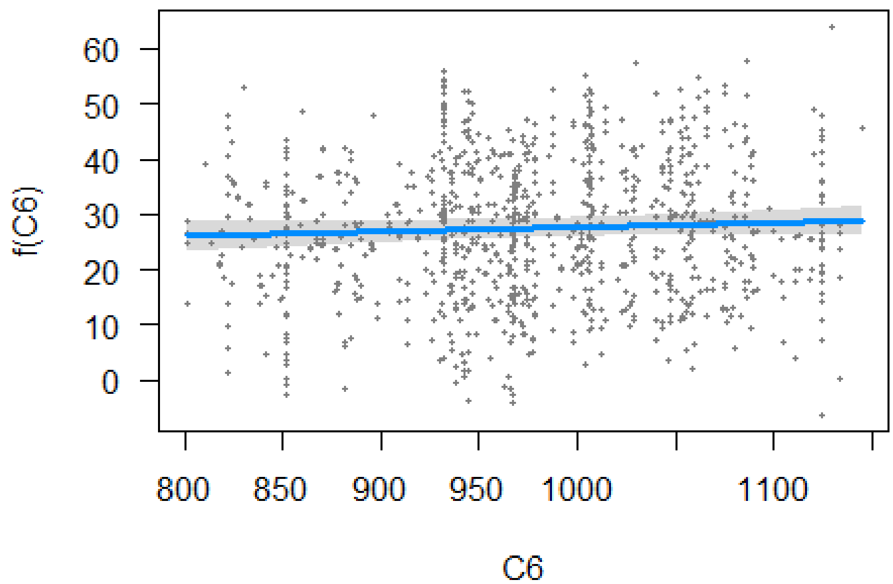

```
visreg(GAM_non_linear,'C5')
```



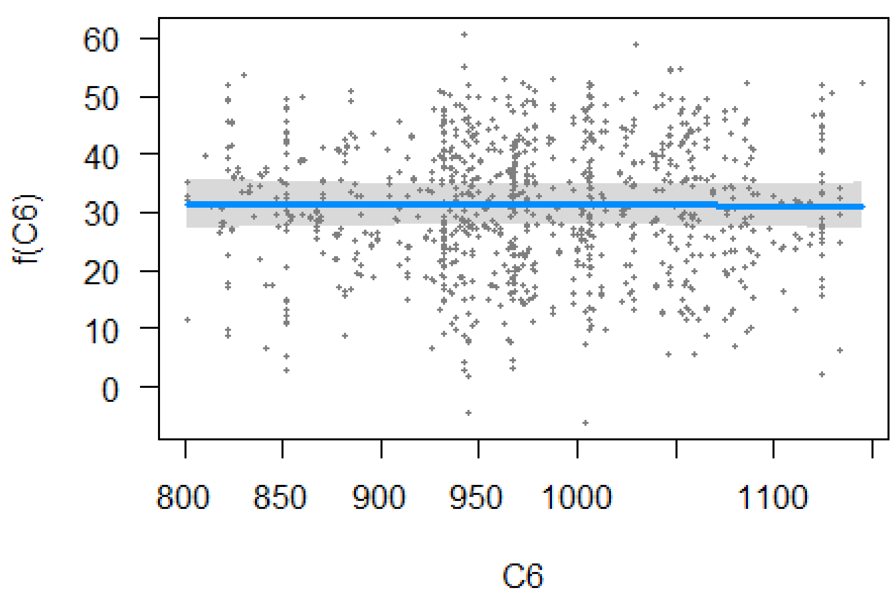

```
visreg(GAM_linear,'C5')
```




```
visreg(GAM_non_linear,'C6')
```




```
visreg(GAM_linear,'C6')
```



#The confidence interval in the above visualizations is represented by grey band around the expected value (blue line).

#After having observed all vizualizations, it can be seen that after applying smoothing, the confidence interval is better and has higher values when compared to the initial model.