Homework 3

EDUARDO GALEOTE ESCALERA

A20552496

CSP 571

Data Preparation and Analysis

October 2023

Contents

1 Recitation Exercises

1.1 Chapter 6

1.1.1 Excercise 1

We perform best subset, forward stepwise, and backward stepwise selection on a single data set. For each approach, we obtain p + 1 models, containing 0, 1, 2, . . . , p predictors. Explain your answers:

a) *Which of the three models with k predictors has the smallest training RSS?*
Best subset selection as the lowest training RSS as it fits models for every possible combination of predictors. When p is very large, this increases the chance of finding models that fit the training data very well.

b) *Which of the three models with k predictors has the smallest test RSS?*
Best test RSS could be provided by any of the models. Bet subset considers more models than the other two, and the best model found on the training set could also be the best one for a test set. Forward and backward stepwise consider a lot fewer models, but might find a model that fits the test set very well as it tends to avoid over fitting when compared to best subset.

c) *True or False:*

- *The predictors in the k-variable model identified by forward stepwise are a subset of the predictors in the (k+1)-variable model identified by forward stepwise selection.*
True; the k+1 model is derived by adding a predictor that gives greatest improvement to previous k model.
- *The predictors in the k-variable model identified by backward stepwise are a subset of the predictors in the (k + 1)- variable model identified by backward stepwise selection.*
True; the k model is derived by removing the least useful predictor from the k+1 model.
- *The predictors in the k-variable model identified by forward stepwise are a subset of the predictors in the (k+1)-variable model identified by backward stepwise selection.*
False; forward and backward stepwise can select different predictors.
- *The predictors in the k-variable model identified by best subset are a subset of the predictors in the (k + 1)-variable model identified by best subset selection.*
False; forward and backward stepwise can select different predictors.

1.1.2 Excercise 2

For parts (a) through (c), indicate which of i. through iv. is correct. Justify your answer.

a) *The lasso, relative to least squares, is:*
(iii) Less flexible and will give improved prediction accuracy when its increase in bias is less than its decrease in variance. As lambda increases, flexibility of fit decreases, and so the estimated coefficients decrease with some being zero. This leads to a substantial decrease in the variance of the predictions for a small increase in bias.

b) *Repeat (a) for ridge regression relative to least squares.*
(iii) same as (a), except every variable has a non-zero coefficient.

c) *Repeat (a) for non-linear methods relative to least squares.*
(ii) Non-linear models will generally be more flexible, and so predictions tend to have a higher variance and lower bias. So predictions will improve if the variance rises less than a decrease in the bias (bias-variance trade off).

1.1.3 Excercise 3

Suppose we estimate the regression coefficients in a linear regression model by minimizing for a particular value of s. For parts (a) through (e), indicate which of i. through v. is correct. Justify your answer.

a) *As we increase s from 0, the training RSS will:*
(i) Decrease steadily. As s increases the constraint on beta decreases and the RSS reduces until we reach the least squares answer.

b) *Repeat (a) for test RSS.*
(ii) Decreases initially as RSS in reduced from the maximum value (when B=0) as we move towards the best training value for B1. Eventually starts increasing as the B values start fitting the training set extremely well, and so over fitting the test set.

c) *Repeat (a) for variance.*
(iii) Steadily increase.

d) *Repeat (a) for (squared) bias.*
(iv) Steadily decrease.

e) *Repeat (a) for the irreducible error.*
(v) Remains constant.

1.1.4 Excercise 4

Suppose we estimate the regression coefficients in a linear regression model by minimizing for a particular value of λ. For parts (a) through (e), indicate which of i. through v. is correct. Justify your answer.

a) *As we increase λ from 0, the training RSS will:*
(iii) Steadily increase. As lambda increases the constraint on beta also increases, this means beta becomes progressively smaller. As such the training RSS will steadily increase to the maximum value (when B 0 at very large lambda values.)

b) *Repeat (a) for test RSS.*

(ii) Decreases initially and then starts increasing in a U shape.When lambda=0, we get a least squares fit that has high variance and low bias. As lambda increases, the flexibility of the fit decreases, so reducing variance of predictions for a small increase in bias. This results in more accurate predictions and so the test RSS will decrease initially. As lambda increases beyond the ideal point, we start seeing a much greater increase in bias than the reduction in variance, and so predictions will become more biased. Consequently, we will see a rise in the test RSS.

c) *Repeat (a) for variance.*
(iv) Steadily decrease.

d) *Repeat (a) for (squared) bias.*
(iii) Steadily increase.

e) *Repeat (a) for the irreducible error.*
(v) Remains constant.

1.1.5 Excercise 5

It is well-known that ridge regression tends to give similar coefficient values to correlated variables, whereas the lasso may give quite different coefficient values to correlated variables. We will now explore this property in a very simple setting. Suppose that $n = 2$, $p = 2$, $x_{11} = x_{12}$, $x_{21} = x_{22}$. Furthermore, suppose that $y_1 + y_2 = 0$ and $x_{11} + x_{21} = 0$ and $x_{12} + x_{22} = 0$, so that the estimate for the intercept in a least squares, ridge regression, or lasso model is zero: $\hat{\beta}_0 = 0$.

a) *Write out the ridge regression optimization problem in this setting.*

b) *Argue that in this setting, the ridge coefficient estimates satisfy $\hat{\beta}_1 = \hat{\beta}_2$.*

c) *Write out the lasso optimization problem in this setting.*

d) *Argue that in this setting, the lasso coefficients $\hat{\beta}_1$ and $\hat{\beta}_2$ are not unique—in other words, there are many possible solutions to the optimization problem in (c). Describe these solutions.*

1.2 Chapter 7

1.2.1 Excercise 2

Suppose that a curve $\hat{g}$ is computed to smoothly fit a set of n points using the following formula. Provide example sketches of $\hat{g}$ in each of the following scenarios.
In general, when $\lambda = \infty$, the penalty term is so large that it forces a function g chosen to minimize the RSS into being perfectly smooth. This is because the penalty term reduces the variability in g.

a) $\lambda = \infty$, $m = 0$.
When $\lambda = \infty$, $g = 0$. So, $\hat{g} = 0$.

b) $\lambda = \infty$, $m = 1$.
When $\lambda = \infty$, $g' = 0$ (slope=0). So, $\hat{g} = constant$(say a horizontal line).

c) $\lambda = \infty$, $m = 2$.
When $\lambda = \infty$, $g'' = 0$ (the change in slope=0).So, $\hat{g}$ must be a straight line with a slope, say g$_h at = cx + d.\lambda = \infty, m = 3.$
$When \lambda = \infty$, $g''' = 0$(change in second derivative=0). So, $\hat{g}$ must be a quadratic curve, say g$_h at = cx^2 + dx + e.$

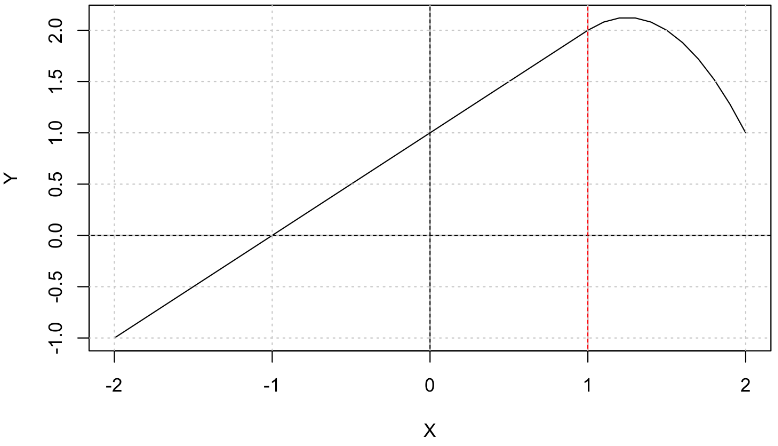
d) $\lambda = 0$, $m = 3$.
When $\lambda = 0$ the penalty term has no effect, so we get a curve that interpolates all the n points perfectly (RSS Train = 0).

1.2.2 Excercise 3

Suppose we fit a curve with basis functions $b_1(X) = X$, $b_2(X) = (X-1)^2 I(X \geq 1)$. (Note that $I(X \geq 1)$ equals 1 for $X \geq 1$ and 0 otherwise.) We fit the linear regression model

$$Y = \beta_0 + \beta_1 b_1(X) + \beta_2 b_2(X) + \varepsilon,$$

and obtain coefficient estimates $\hat{\beta}_0 = 1$, $\hat{\beta}_1 = 1$, $\hat{\beta}_2 = -2$. Sketch the estimated curve between $X = -2$ and $X = 2$. Note the intercepts, slopes, and other relevant information.



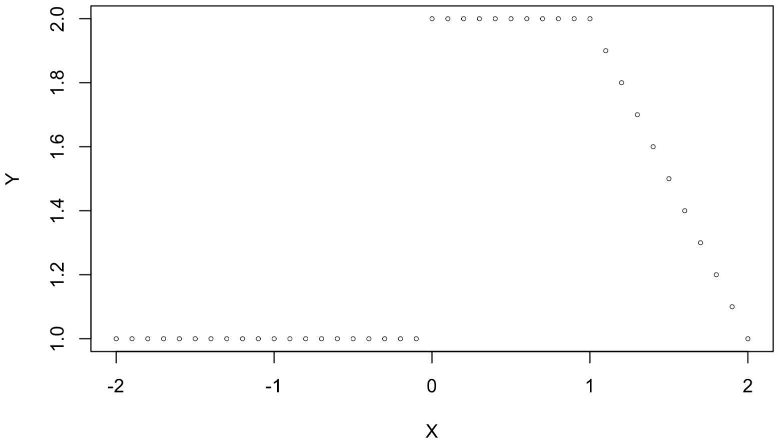
The curve is linear when $-2 < X \leqslant 1$, this portion has a slope and y intercept of 1. The curve then takes a quadratic shape when $1 < X \leqslant 2$.

1.2.3 Excercise 4

Suppose we fit a curve with basis functions $b_1(X) = I(0 \leq X \leq 2) - (X-1)I(1 \leq X \leq 2)$ and $b_2(X) = (X-3)I(3 \leq X \leq 4) + I(4 < X \leq 5)$. We fit the linear regression model

$$Y = \beta_0 + \beta_1 b_1(X) + \beta_2 b_2(X) + \varepsilon,$$

and obtain coefficient estimates $\hat{\beta}_0 = 1$, $\hat{\beta}_1 = 1$, $\hat{\beta}_2 = 3$. Sketch the estimated curve between $X = -2$ and $X = 6$. Note the intercepts, slopes, and other relevant information.



The chart consists of straight lines and a linear section with a slope of -2.

1.2.4 Excercise 5

Consider two curves, $\hat{g}_1$ and $\hat{g}_2$, defined by

$$\hat{g}_1 = \arg\min_g \left\{ \sum_{i=1}^{n} (y_i - g(x_i))^2 + \lambda \int_{L_1}^{L_2} \left(g^{(3)}(x) \right)^2 dx \right\},$$

$$\hat{g}_2 = \arg\min_g \left\{ \sum_{i=1}^{n} (y_i - g(x_i))^2 + \lambda \int_{L_1}^{L_2} \left(g^{(4)}(x) \right)^2 dx \right\},$$

where $g(m)$ represents the mth derivative of g.

a) *As $\lambda \to \infty$, will $\hat{g}_1$ or $\hat{g}_2$ have the smaller training RSS?*
$\hat{g}_2$ is more flexible due to the higher order of the penalty term than $\hat{g}_1$, so it will likely have a lower training RSS.

b) *As $\lambda \to \infty$, will $\hat{g}_1$ or $\hat{g}_2$ have the smaller test RSS?*
This depends on the shape of the underlying function for the dataset used. Generally, $\hat{g}_1$ will perform better on less flexible functions, and $\hat{g}_2$ will perform better on more flexible functions.

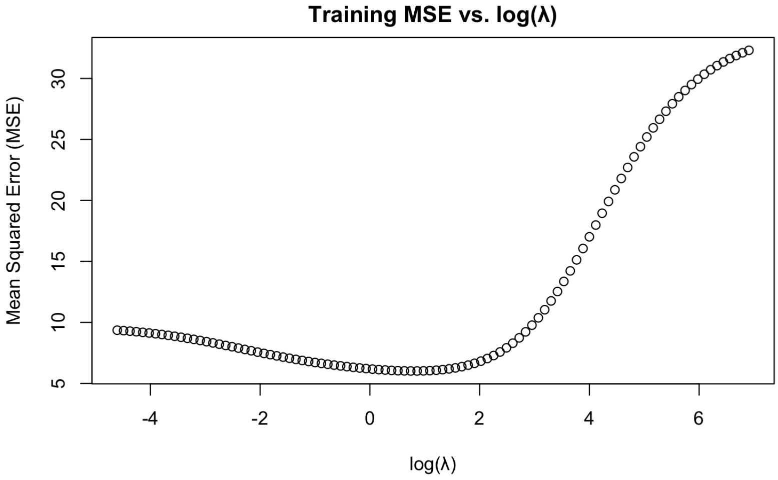
c) *For $\lambda = 0$, will $\hat{g}_1$ or $\hat{g}_2$ have the smaller training and test RSS?*
The penalty terms will be zero for both equations, so training and test terms will be equal.

2 Practicum Exercises

2.1 Problem 1

Load the mtcars sample dataset from the built-in datasets (data(mtcars)) into R using a dataframe. Perform a basic 80/20 test-train split on the data (you may use caret, the sample method, or manually) and fit a linear model with mpg as the target response, and all other variables as *predictors/features (you will need to set up a dummy variable for am). What features are selected as relevant based on resulting t-statistics? What are the associated coefficient values for relevant features? Perform a ridge regression using the glmnet package from CRAN, specifying a vector of 100 values of λ for tuning. Use cross-validation (via cv.glmnet) to determine the minimum value for λ - what do you obtain? (Hint: You can use doMC in order to speed-up your cross-validation by specifying parallel=TRUE in your glmnet calls.). Plot training MSE as a function of λ (you may also use log λ). What is out-of-sample test set performance (using predict), and how do the coefficients differ versus the regular linear model? Has ridge regression performed shrinkage, variable selection, or both?*

Based on the t-stadistics, non of the features are statistically significant. Their p-values are much larger than 0.05. Associated coefficients: (Intercept)=-2.81 cyl=0.76 disp=0.01 hp=-0.01 drat=2.24 wt=-2.73 qsec=0.54 vs=1.22 am=1.74



Minimum lambda based on cross-validation: 2.104904. Out-of-sample Test Set MSE: 13.74783. The ridge coefficients are: (Intercept) 21.114868163 cyl -0.314939506 disp -0.003822596 hp -0.011538535 drat 1.166751426 wt -1.203628568 qsec 0.037092518 vs 0.719553384 am 1.509774082 gear 0.875410900 carb -0.606423793 Ridge regression introduces a penalty term (L2 regularization) that adds a constraint to the optimization problem. This constraint limits the size of the coefficients, shrinking them towards zero. This shrinkage helps to mitigate the problem of multicollinearity and reduces the variance of the estimates.

```r
# Load required libraries
library(caret)

# Load the mtcars dataset
data(mtcars)

# Set seed for reproducibility
set.seed(123)

# Create a dummy variable for am (automatic or manual transmission)
mtcars$am_dummy <- factor(mtcars$am, levels = c(0, 1), labels = c("Automatic", "
    Manual"))

# Perform an 80/20 train-test split
index <- createDataPartition(mtcars$mpg, p = 0.8, list = FALSE)
train_data <- mtcars[index, ]
test_data <- mtcars[-index, ]

# Fit a linear model with mpg as the target response
```

```r
lm_model <- lm(mpg ˜ . - am, data = train_data)

# Print the summary of the linear model
summary(lm_model)

# Load required libraries
library(caret)
library(glmnet)
library(doMC)

# Set seed for reproducibility
set.seed(123)

# Load the mtcars dataset
data(mtcars)

# Create a dummy variable for am (automatic or manual transmission)
mtcars$am_dummy <- factor(mtcars$am, levels = c(0, 1), labels = c("Automatic", "
    Manual"))

# Perform an 80/20 train-test split
index <- createDataPartition(mtcars$mpg, p = 0.8, list = FALSE)
train_data <- mtcars[index, ]
test_data <- mtcars[-index, ]

# Set the number of cores for parallel processing
num_cores <- 4
registerDoMC(cores = num_cores)

# Prepare the data
X_train <- as.matrix(train_data[, -c(1, 12)])  # Exclude ’mpg’ and ’am_dummy’
y_train <- train_data$mpg

# Specify a vector of 100 values of   for tuning
lambda_values <- 10^seq(3, -2, length.out = 100)

# Perform ridge regression with cross-validation
ridge_cv <- cv.glmnet(X_train, y_train, alpha = 0, lambda = lambda_values,
    parallel = TRUE)

# Identify the minimum   based on cross-validation
min_lambda <- ridge_cv$lambda.min
cat("Minimum lambda based on cross-validation:", min_lambda, "\n")

# Stop parallel processing
stopImplicitCluster()

# Plot training MSE as a function of log(lambda)
plot(log(ridge_cv$lambda), ridge_cv$cvm, type = "b", xlab = "log(lambda)", ylab
    = "Mean Squared Error (MSE)", main = "Training MSE vs. log(lambda)")

# Identify the minimum lambda based on cross-validation
min_lambda <- ridge_cv$lambda.min
cat("Minimum lambda based on cross-validation:", min_lambda, "\n")

# Prepare the test data
X_test <- as.matrix(test_data[, -c(1, 12)])  # Exclude ’mpg’ and ’am_dummy’
y_test <- test_data$mpg

# Evaluate out-of-sample test set performance
test_predictions <- predict(ridge_cv, newx = X_test, s = min_lambda)
test_mse <- mean((test_predictions - y_test)^2)
cat("Out-of-sample Test Set MSE:", test_mse, "\n")
```

```

# Extract coefficients from the ridge regression model
ridge_coefficients <- coef(ridge_cv, s = min_lambda)

# Print ridge regression coefficients
print(ridge_coefficients)
```

2.2 Problem 2

Load the swiss sample dataset from the built-in datasets (data(swiss)) into R using a dataframe. Perform a basic 80/20 test-train split on the data (you may use caret, the sample method, or manually) and fit a linear model with Fertility as the target response, and all other variables as predictors/features. What features are selected as relevant based on resulting t-statistics? What are the associated coefficient values for relevant features? Perform a lasso regression using the glmnet package from CRAN, specifying a vector of 100 values of λ for tuning. Use cross-validation (via cv.glmnet) to determine the minimum value for λ - what do you obtain? (Hint: You can use doMC in order to speed-up your cross-validation by specifying parallel=TRUE in your glmnet calls.). Plot training MSE as a function of λ (you may also use log λ). What is out-of-sample test set performance (using predict), and how do the coefficients differ versus the regular linear model? Has lasso regression performed shrinkage, variable selection, or both?

Based on the t-stadistics, the features that are statistically significant: Agriculture, Education and Catholic. Their p-values are lower than 0.05.

Coefficient Comparison:

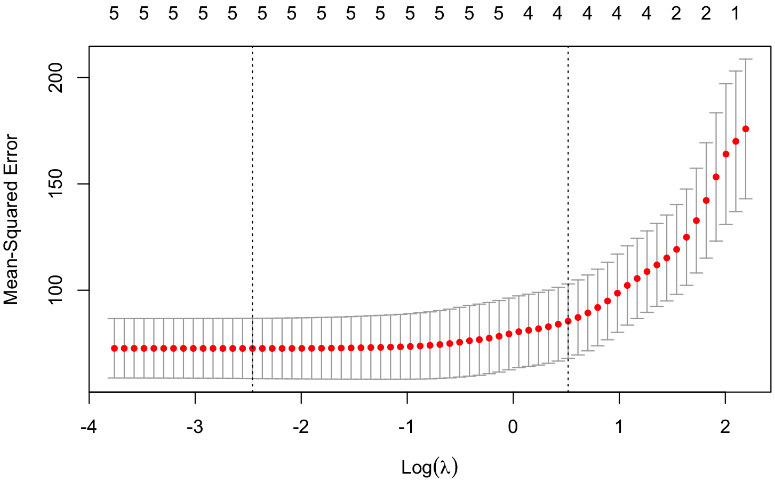
Linear Model Coefficients:

74.1934 -0.186346 -0.3199657 -0.8735478 0.1064816 0.7807456

Lasso Regression Coefficients:

-0.1711915 -0.3119847 -0.8500842 0.1027049 0.7795087

Best lambda: 0.08545762



Out-of-sample test set performance (MSE): Linear Model: 41.09982

Lasso Regression: 40.97843

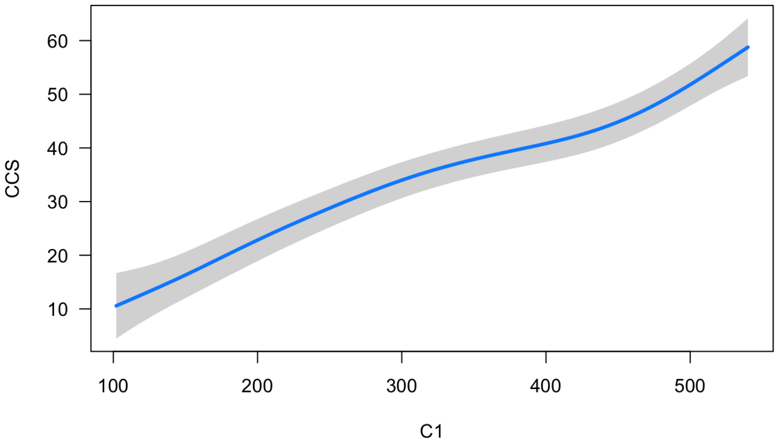
Lasso regression provides a balance between shrinkage and variable selection. The tuning parameter λ controls the trade-off between fitting the data well and keeping the model simple. The cross-validation process, as implemented in the cv.glmnet function, helps in selecting the optimal λ that achieves the best compromise between model complexity and predictive performance.

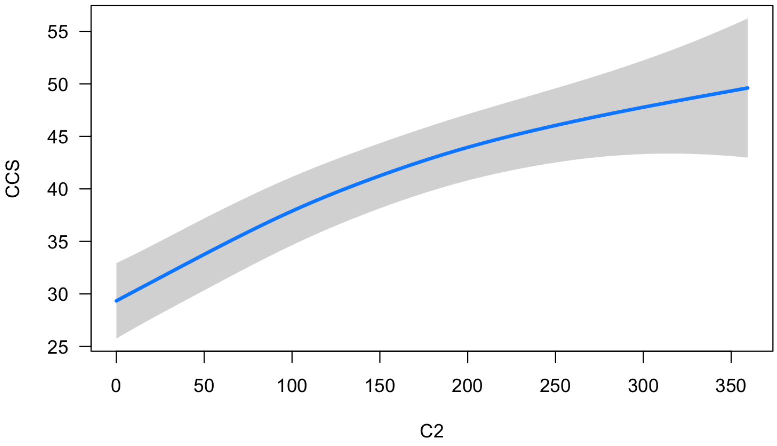
```r
# Load required libraries
library(caret)
library(glmnet)
library(doMC)

# Set the number of cores for parallel processing
num_cores <- 4
registerDoMC(cores = num_cores)

# Load the Swiss dataset
data(swiss)

# Split the data into training (80%) and testing (20%) sets
set.seed(123)  # for reproducibility
train_index <- createDataPartition(swiss$Fertility, p = 0.8, list = FALSE)
train_data <- swiss[train_index, ]
test_data <- swiss[-train_index, ]

# Fit a linear model
linear_model <- lm(Fertility ~ ., data = train_data)
summary(linear_model)

# Fit a lasso regression model
x <- as.matrix(train_data[, -1])  # Exclude the response variable
y <- train_data$Fertility
lasso_model <- cv.glmnet(x, y, alpha = 1, parallel = TRUE, nfolds = 10, nlambda = 100)

# Plot training MSE as a function of log(lambda)
plot(lasso_model)

# Identify the lambda that minimizes cross-validated error
best_lambda <- lasso_model$lambda.min
cat("Best lambda:", best_lambda, "\n")

# Predict using the lasso model on the test set
x_test <- as.matrix(test_data[, -1])
y_pred_lasso <- predict(lasso_model, newx = x_test, s = best_lambda)

# Predict using the linear model on the test set
y_pred_linear <- predict(linear_model, newdata = test_data)

# Compare performance
mse_lasso <- mean((y_pred_lasso - test_data$Fertility)^2)
mse_linear <- mean((y_pred_linear - test_data$Fertility)^2)

cat("Out-of-sample test set performance (MSE):\n")
cat("Linear Model:", mse_linear, "\n")
cat("Lasso Regression:", mse_lasso, "\n")

# Compare coefficients
coef_linear <- coef(linear_model)
coef_lasso <- coef(lasso_model, s = best_lambda)[-1]  # Exclude the intercept

cat("\nCoefficient Comparison:\n")
cat("Linear Model Coefficients:\n", coef_linear, "\n")
cat("Lasso Regression Coefficients:\n", coef_lasso, "\n")

# Check if lasso performed shrinkage, variable selection, or both
cat("\nLasso Regression Summary:\n")
print(lasso_model)
```

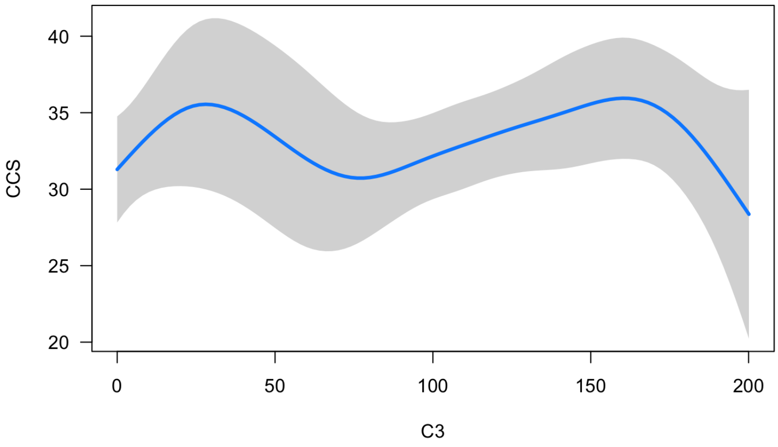
2.3 Problem 3

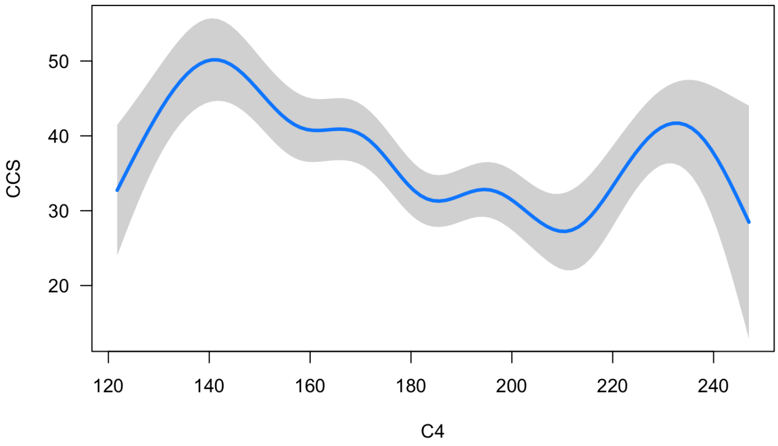
Load the Concrete Compressive Strength sample dataset from the UCI Machine Learning Repository (Concrete Data.xls) into R using a dataframe (Note: You will need to either use the xlsx or readxl packages to load Excel data, or manually save as CSV). Use the mgcv package to create a generalized additive model (via the gam function) to predict the Concrete Compressive Strength (CCS) as a non-linear function of the input components (C1-C6) - compare the R^2 value for a GAM with linear terms as well as smoothed terms (Hint: Use the s() function to apply smoothing using the default bs of tp). Visualize the regression using the visreg package, showing the fit as a function of each predictor with confidence intervals - comment on the quality of the fit at extreme values of the predictors.

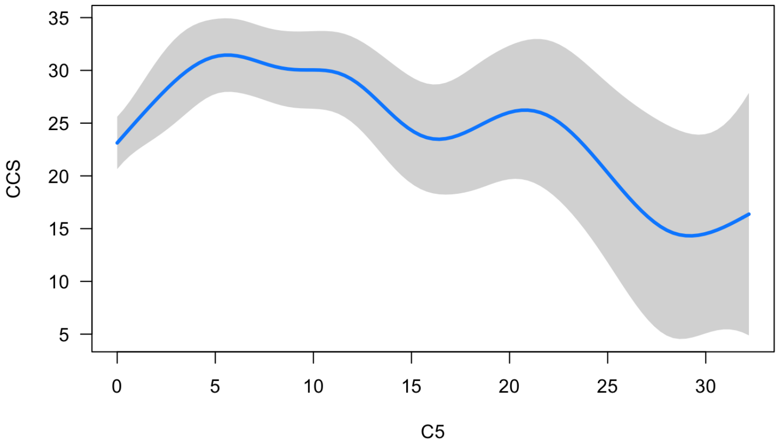
gam linear R^2: 0.4454123
gam smooth R^2: 0.5307281

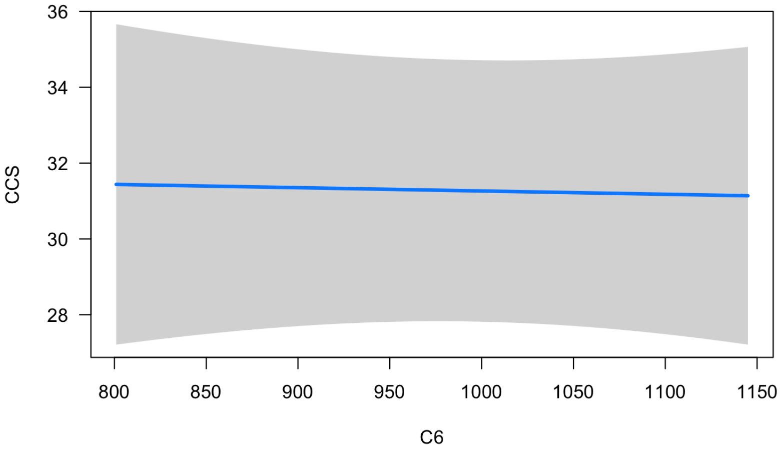





CCS
20
25
30
35
40
0
50
100
150
200
C3


CCS
20
30
40
50
120
140
160
180
200
220
240
C4


CCS
5
10
15
20
25
30
35
0
5
10
15
20
25
30
C5



The fit is good on some features because the width of the confidence intervals are narrow such as C1 and C4. If the confidence intervals are wide, it indicates uncertainty in predictions like C3 and C6. There is uncertainty in the C4 and C5 plots because of the extreme values.

```r
# Load required libraries
library(mgcv)
library(visreg)

# Load the data
concrete_data <- readxl::read_excel("/Users/edugaleote/Downloads/concrete+compressive+strength/Concrete_Data.xls")

colnames(concrete_data)[colnames(concrete_data) == "Cement (component 1)(kg in a m^3 mixture)"] <- "C1"
colnames(concrete_data)[colnames(concrete_data) == "Blast Furnace Slag (component 2)(kg in a m^3 mixture)"] <- "C2"
colnames(concrete_data)[colnames(concrete_data) == "Fly Ash (component 3)(kg in a m^3 mixture)"] <- "C3"
colnames(concrete_data)[colnames(concrete_data) == "Water  (component 4)(kg in a m^3 mixture)"] <- "C4"
colnames(concrete_data)[colnames(concrete_data) == "Superplasticizer (component 5)(kg in a m^3 mixture)"] <- "C5"
colnames(concrete_data)[colnames(concrete_data) == "Coarse Aggregate  (component 6)(kg in a m^3 mixture)"] <- "C6"
colnames(concrete_data)[colnames(concrete_data) == "Concrete compressive strength(MPa, megapascals)"] <- "CCS"

# Create a GAM with linear terms
gam_linear <- gam(CCS ~ C1 + C2 + C3 + C4 + C5 + C6, data = concrete_data)

# Create a GAM with smoothed terms using default bs of tp
gam_smooth <- gam(CCS ~ s(C1) + s(C2) + s(C3) + s(C4) + s(C5) + s(C6), data = concrete_data)

# Compare R2 values
summary(gam_linear)$r.sq
summary(gam_smooth)$r.sq

# Visualize the regression using visreg
visreg(gam_smooth, scale = "response", rug = FALSE)
```