Statistical Methods - HW6

2023-02-27

Problem 1

Implement k-fold cross-validation and sequential model selection for linear regression models.

a. Write a function cv.lm(x, y, k) which estimates the prediction error of the linear regression model with y as response using k-fold cross-validation

```
cv.lm <- function(x,y,k) {
  # Intercept only model handler
  if(length(x) == 0) {
    x <- rep(1, length(y))
  }
  # Create dataframes for x and y
  x <- as.data.frame(x)
  y <- as.data.frame(y)
  n <- nrow(x)
  nu <- sample(n)
  x_cv <- as.data.frame(x[nu,])
  y_cv <- as.data.frame(y[nu,])
  # Create folds for cross-validation
  folds <- cut(seq(1, n), breaks = k, labels = FALSE)
  # Initialize vector to store mean squared errors
  cv_err <- rep(NA, k)
  for(i in 1:k) {
    # Split data into training and validation sets
    x_train <- x_cv[folds != i,]
    y_train <- y_cv[folds != i,]
    x_test <- x_cv[folds == i,]
    y_test <- y_cv[folds == i,]
    # Fit linear regression model on training set
    fit <- lm(y_train ~ ., data = as.data.frame(x_train))
    # Predict on validation set
    pred_val <- predict(fit, as.data.frame(x_test))
    res_val = (y_test - pred_val)
    # Calculate squared error and store in vector
    cv_err[i] <- sqrt(mean(res_val^2))
  }
  # Return the mean error across all folds
  return(mean(cv_err))
}
```

b. Write a function SequentialSelection(x, y, method) which computes the forward selection path for linear regression from 'intercept only' to 'full model' and chooses the model on that path using different criteria specified by method. The function should support these methods: • method = "AdjR2": Sequentially include the columns of x and choose the model that gives the largest adjusted R2.
• method = "AIC": Sequentially include the columns of x and choose the model that gives the smallest

AIC.

- method = "CV5": Sequentially include the columns of x and choose the model that gives the smallest 5-fold cross-validation prediction error.

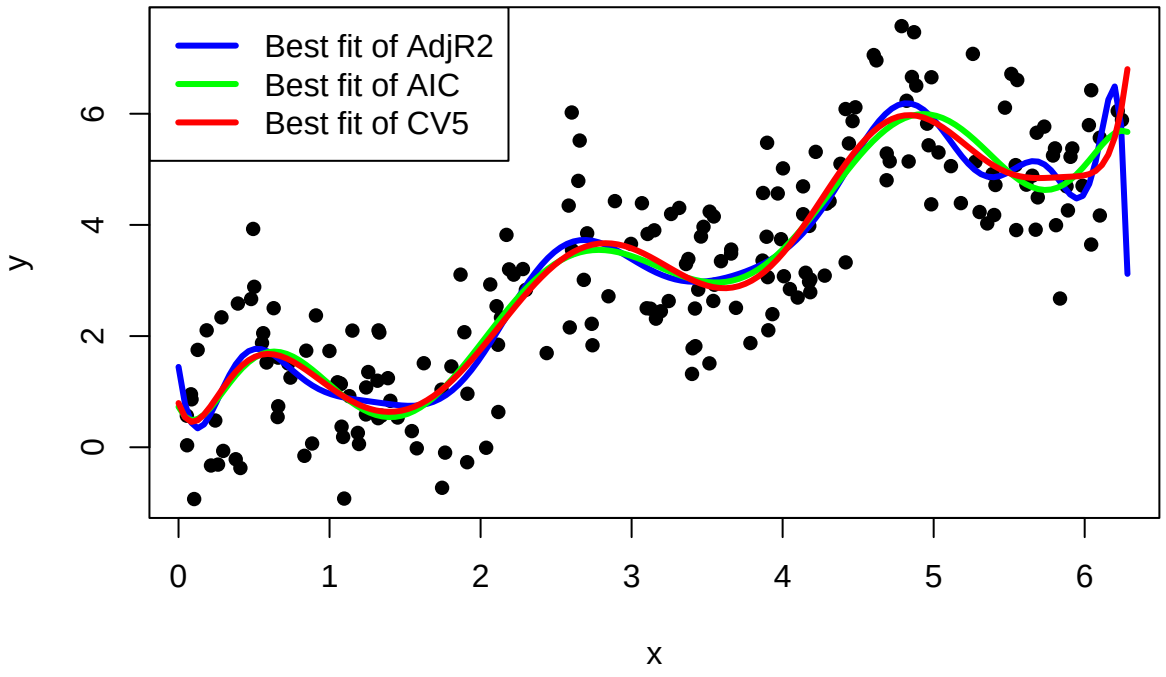
```
SequentialSelection <- function(x, y, method) {
  # Create a vector to store the metrics for each method
  metrics <- vector(mode = "numeric", length = ncol(x))
  # Loop through each feature in x
  for(i in 1:ncol(x)) {
    # Initialize the current model by fitting the linear regression model
    curr_model <- lm(y ~ ., data = data.frame(x[,1:i]))
    # Calculate the metric for each method
    if(method == "AdjR2") {
      # Calculate adjusted R2 for each model
      metrics[i] <- summary(curr_model)$adj.r.squared
    } else if(method == "AIC") {
      # Calculate AIC for each model
      metrics[i] <- AIC(curr_model)
    } else if(method == "CV5") {
      # Calculate 5-fold cross-validation prediction error for each model
      metrics[i] <- cv.lm(x[,1:i], y, 5)
    }
  }
  # Choose the best model based on the metrics for each method
  if(method == "AdjR2") {
    # Choose the model that gives the largest adjusted r2 value
    best_model = which.max(metrics)-1
  } else if(method == "AIC") {
    # Choose the model that gives the smallest AIC
    best_model = which.min(metrics)-1
  } else if(method == "CV5") {
    # Choose the model that gives the smallest 5-fold cv prediction error
    best_model = which.min(metrics)-1
  }
  # Return the best model
  return(best_model)
}
```

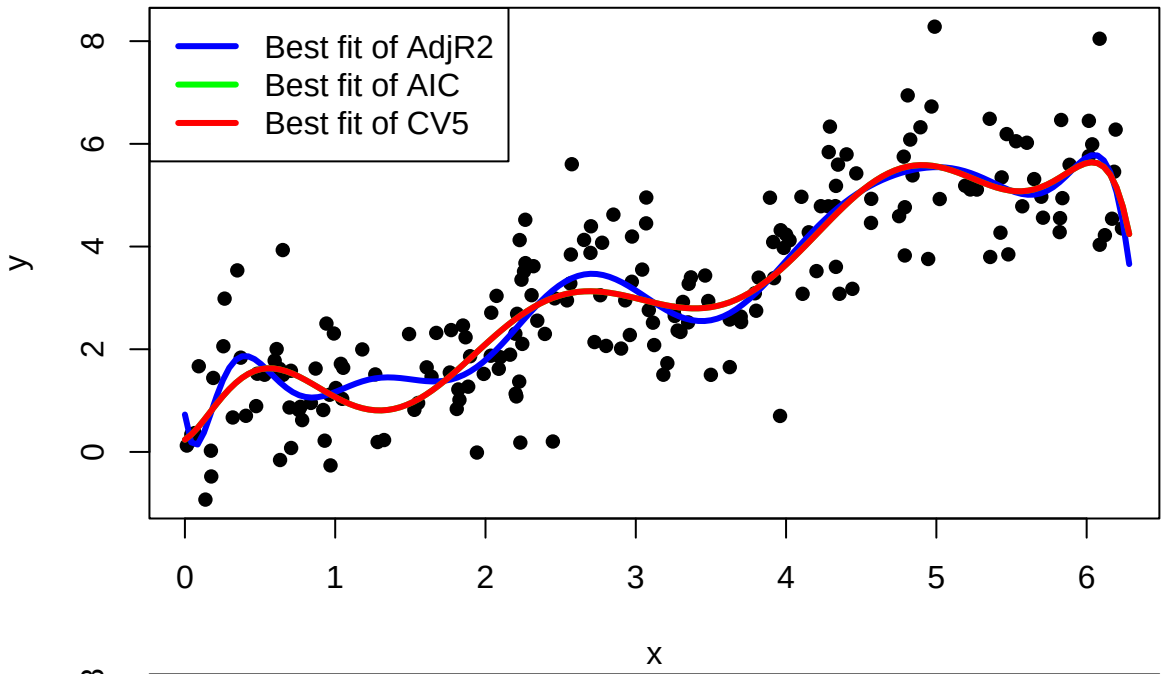
Problem 2

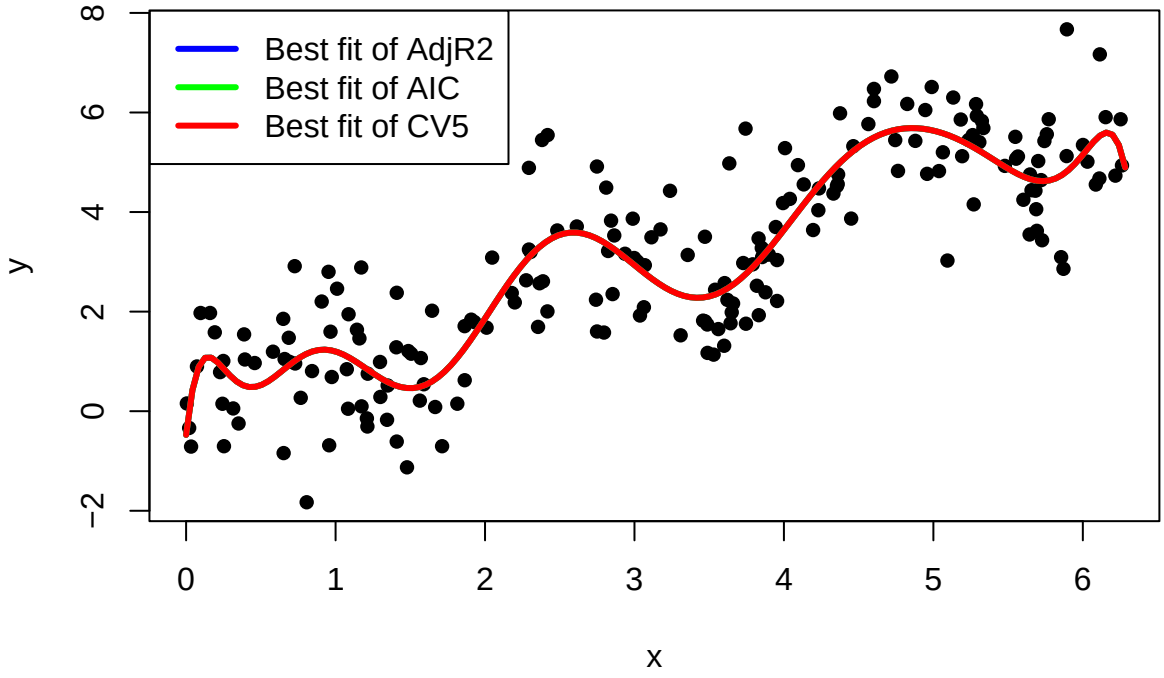
Consider a regression setting where the predictor variable is real valued and the goal is to fit a polynomial model. Specifically, we assume that x1,…,xn are iid uniform in [0,2] and conditional on these, y1,…,yn are independent, with yi normal with mean sin(3xi) + xi and variance 1. Take n = 200 and set the maximum degree at 20. Perform simulations (at least 100 data instances) to compare the choice of degree by the sequential model selection methods in Problem 1. Produce plots of 3 example data instances and their best model fits according to different methods. Produce plots of the distribution of the polynomial degrees chosen by the different methods over all simulated instances. Offer comments on what you observe.

```
#' Choose 3 sample data instances and loop over
#' to find the best fit model based on each method
for (i in 1:3) {
  n <- 200
  # Generate x and y based on given distributions
  x <- runif(n, 0, 2*pi)
```

```r
  y <- rnorm(n, mean = sin(3*x) + x, sd = 1)
  # Max degree
  max.degree <- 20
  # Data frame of polynomial x based on the max degree = 20
  xp <- poly(x, degree = max.degree, raw = TRUE)
  xp <- data.frame(model.matrix(~xp))
  # Store selected degrees for each method
  selected_degree_adjR2 = SequentialSelection(xp, y, "AdjR2")
  selected_degree_aic = SequentialSelection(xp, y, "AIC")
  selected_degree_cv5 = SequentialSelection(xp, y, "CV5")
  #' Plot of 3 example data instances and
  #' their best model fits according to different methods
  plot(x, y, pch = 16)
  fit = lm(y ~ poly(x, selected_degree_adjR2, raw = TRUE))
  points = seq(0, 2*pi, len = 150)
  value = predict(fit, data.frame(x = points))
  lines(points, value, col = "blue", lwd = 3)
  fit = lm(y ~ poly(x, selected_degree_aic, raw = TRUE))
  points = seq(0, 2*pi, len = 150)
  value = predict(fit, data.frame(x = points))
  lines(points, value, col = "green", lwd = 3)
  fit = lm(y ~ poly(x, selected_degree_cv5, raw = TRUE))
  points = seq(0, 2*pi,len = 150)
  value = predict(fit, data.frame(x = points))
  lines(points, value, col = "red", lwd = 3)
  # Add legend to the plots
  legend('topleft', c('Best fit of AdjR2', 'Best fit of AIC', 'Best fit of CV5'),
         col = c('blue', 'green', 'red'), lwd = 3)
}
```

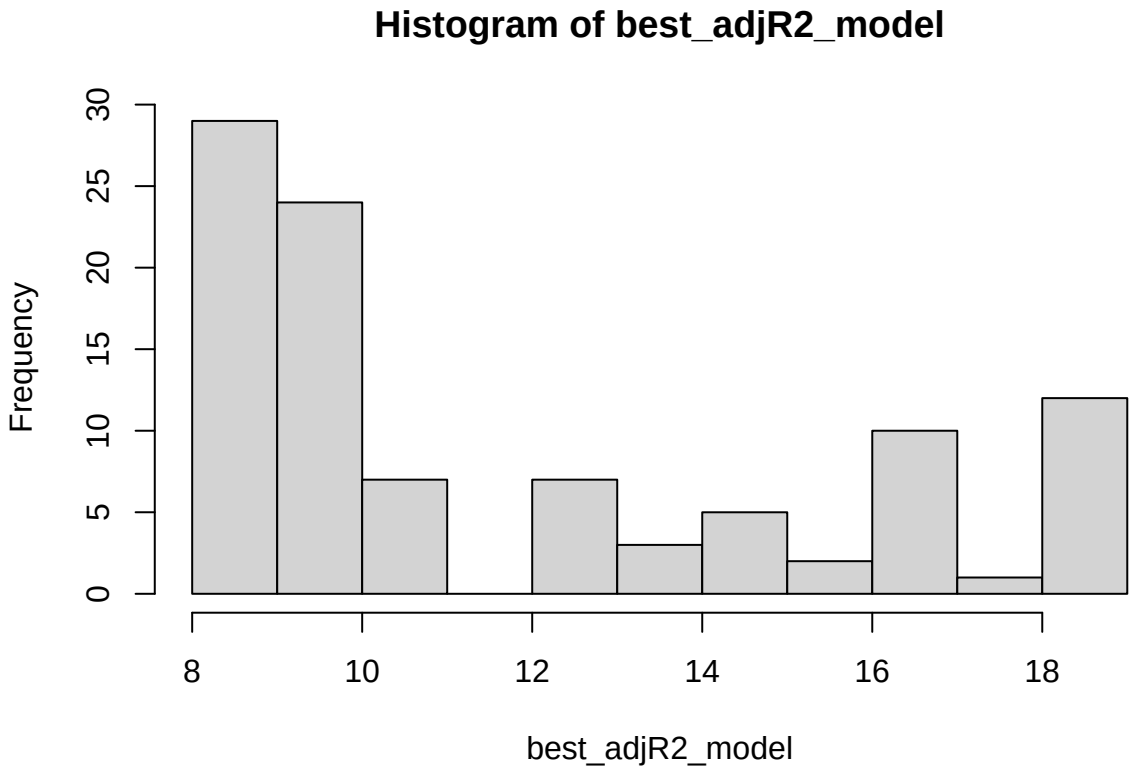




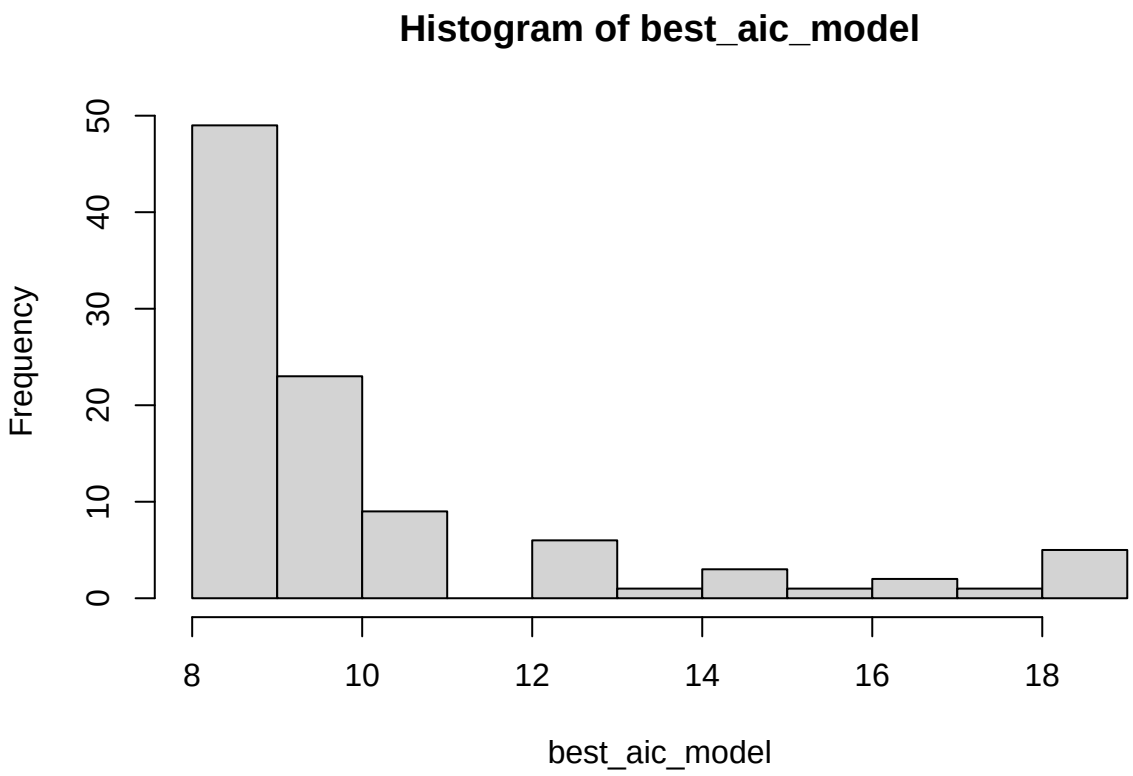



```r
# Create an empty vector to store the best models for each method
best_adjR2_model = c()
best_aic_model = c()
best_cv5_model = c()
# Loop to simulate 100 data instances
for (i in 1:100) {
  n <- 200
  # Max degree
  max.degree <- 20
  # Generate x and y based on given distributions
  x <- runif(n, 0, 2*pi)
  y <- rnorm(n, mean = sin(3*x) + x, sd = 1)
```

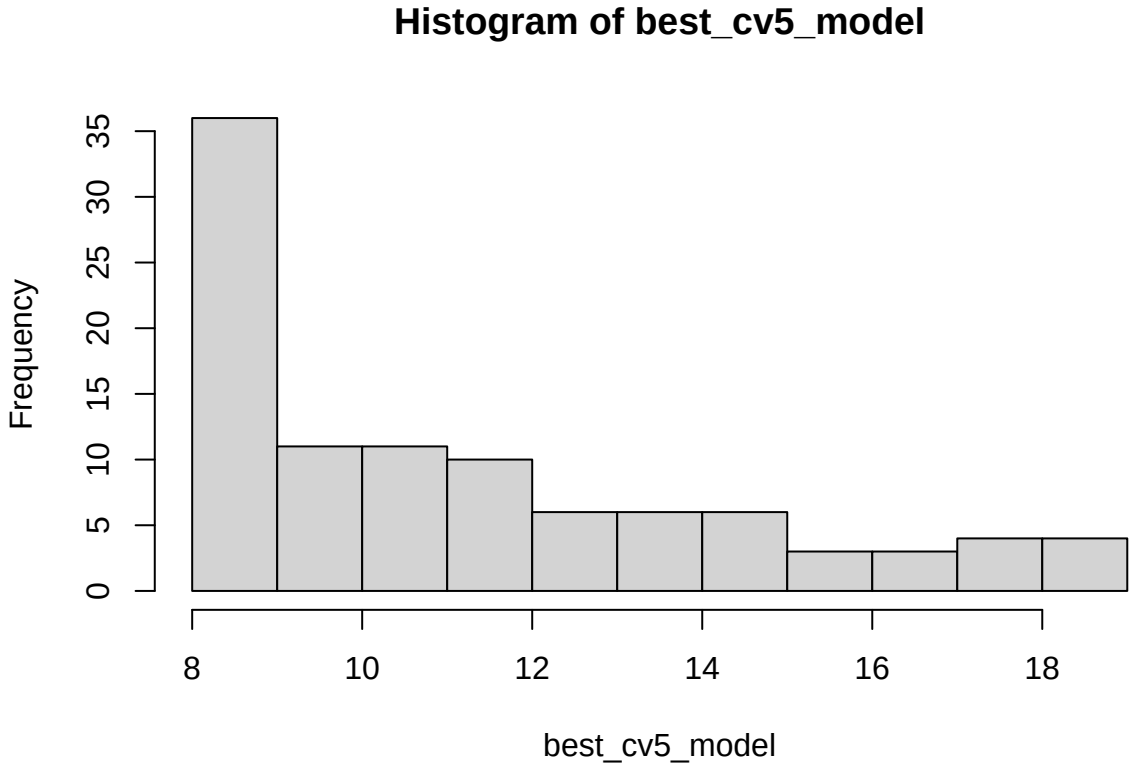
```r
  # Data frame of polynomial x based on the max degree = 20
  xp <- data.frame(poly(x, degree = max.degree, raw = TRUE))
  # Choose the best model for each method
  best_adjR2_model = c(best_adjR2_model, SequentialSelection(xp, y, "AdjR2"))
  best_aic_model = c(best_aic_model, SequentialSelection(xp, y, "AIC"))
  best_cv5_model = c(best_cv5_model, SequentialSelection(xp, y, "CV5"))
}

#' Plot the distribution of the polynomial degrees chosen by
#' the different methods over all simulated instance
hist(best_adjR2_model)
```




```r
hist(best_aic_model)
```




```
hist(best_cv5_model)
```



Observations:

We can observe that the adjR2 method tends to overfit the model as compared to AIC and CV5. One primary reason can be due to the fact that adding more and more features tends to increase the value of adjR2 as compared to adding features in the other models.

In some instances, the best models obtained using the different methods tend to overlap each other as the selected number of features might be same for all the methods.

We can also observe that the distribution of the polynomial degrees chosen by the different methods is more likely in the range where the degree is from 8 to 10. And the distribution of the polynomial degrees tend to be lesser for higher degrees in the range from 10 to 20.

The AdjR2 method tends to select higher degree polynomials more frequently than the other two methods. The AIC method and the CV5 method tend to select lower degree polynomials more frequently than the AdjR2 method.

One possible reason for this difference is that the AdjR2 method penalizes the number of variables less strongly than the AIC method and the CV5 method. As a result, the AdjR2 method tends to favor models with more variables, which corresponds to higher degree polynomials in this case.

On the other hand, the AIC method and the CV5 method penalize the number of variables more strongly. This makes them more likely to favor simpler models with fewer variables, which corresponds to lower degree polynomials in this case.

Overall, the choice of the sequential model selection method can have a significant impact on the selected polynomial degree. In practice, it is important to carefully choose the selection method based on the specific problem at hand and the desired trade-off between model complexity and accuracy.

Contribution Statement:

1. Swetha Arunraj (PID: A59019948):

- Coded Problem 1
- Added comments to the code of Problem 1
- Added description/comments to Problem 1
- Contributed to embed the codes in R Markdown and create the final PDF

2. Harin Raja Radha Krishnan (PID: A59019874):

- Coded Problem 2
- Added comments to the code of Problem 2
- Added description/comments to Problem 2
- Contributed to embed the codes in R Markdown and create the final PDF