Assignment 2

Nic Salmon

2024-11-05

Question 1

1(a)

Consider a simple linear regression model with Gaussian errors. We simulate data from the model

$$Y = X\boldsymbol{\beta} + \epsilon,$$

where $\epsilon \sim \text{Normal}(0, \sigma^2)$, using the following code:

```
set.seed(123)
n <- 100
fake_data <- data.frame(X1 = runif(n, 0, 1),
                        X2 = runif(n, 0, 1),
                        X3 = runif(n, 0, 1),
                        X4 = runif(n, 0, 1),
                        X5 = runif(n, 0, 1),
                        X6 = runif(n, 0, 1),
                        X7 = runif(n, 0, 1),
                        X8 = runif(n, 0, 1),
                        X9 = runif(n, 0, 1),
                        X10 = runif(n, 0, 1))
fake_data$Y <- fake_data$X1 + 2*fake_data$X2 + fake_data$X3 +
  2*fake_data$X4 + fake_data$X5 + 2*fake_data$X6 +
  fake_data$X7 + 2*fake_data$X8 + fake_data$X9 +
  2*fake_data$X10 + rnorm(n, 0, 1)
```

Fit 11 models using these data and the `lm` function in R. The first model should include only the intercept, the second model should include the intercept and the first predictor, the third model should include the intercept and the first two predictors, and so on.

```
models <- list()
for (i in 1:11) {
  if (i == 1) {
    formula <-  as.formula("Y ~ 1")
  } else {
    formula <- as.formula(paste("Y ~ ", paste(paste0("X", 1:(i - 1)),collapse =  "+" )))
    }
  models[[i]] <- lm(formula, data = fake_data)
}
models
```

```
## [[1]]
##
## Call:
## lm(formula = formula, data = fake_data)
##
## Coefficients:
## (Intercept)
##       7.445
##
##
## [[2]]
##
## Call:
## lm(formula = formula, data = fake_data)
##
## Coefficients:
## (Intercept)           X1
##      7.3612       0.1689
##
##
## [[3]]
##
## Call:
## lm(formula = formula, data = fake_data)
##
## Coefficients:
## (Intercept)           X1           X2
##       6.200        0.338        2.094
##
##
## [[4]]
##
## Call:
## lm(formula = formula, data = fake_data)
##
## Coefficients:
## (Intercept)           X1           X2           X3
##      5.8218       0.3897       2.1733       0.6416
##
##
## [[5]]
##
## Call:
## lm(formula = formula, data = fake_data)
##
## Coefficients:
## (Intercept)           X1           X2           X3           X4
##      4.4565       0.6004       2.2818       0.5112       2.5758
##
##
## [[6]]
##
## Call:
## lm(formula = formula, data = fake_data)
```

```
##
## Coefficients:
## (Intercept)          X1          X2          X3          X4          X5
##      3.7488      0.7852      2.2583      0.6550      2.3335      1.3948
##
##
## [[7]]
##
## Call:
## lm(formula = formula, data = fake_data)
##
## Coefficients:
## (Intercept)          X1          X2          X3          X4          X5
##      2.4137      0.9958      2.3319      0.9988      2.5012      1.1543
##          X6
##      2.0408
##
##
## [[8]]
##
## Call:
## lm(formula = formula, data = fake_data)
##
## Coefficients:
## (Intercept)          X1          X2          X3          X4          X5
##      1.7353      1.0533      2.2046      0.9901      2.5727      1.1020
##          X6          X7
##      2.1526      1.2975
##
##
## [[9]]
##
## Call:
## lm(formula = formula, data = fake_data)
##
## Coefficients:
## (Intercept)          X1          X2          X3          X4          X5
##      1.0916      0.8954      2.1161      1.0434      2.5712      0.6727
##          X6          X7          X8
##      2.3012      1.2862      1.8143
##
##
## [[10]]
##
## Call:
## lm(formula = formula, data = fake_data)
##
## Coefficients:
## (Intercept)          X1          X2          X3          X4          X5
##      0.5042      1.1295      2.0704      1.2618      2.3752      0.7467
##          X6          X7          X8          X9
##      2.4542      1.2199      1.6739      0.9690
##
##
```

```
## [[11]]
##
## Call:
## lm(formula = formula, data = fake_data)
##
## Coefficients:
## (Intercept)           X1           X2           X3           X4           X5
##      0.2646       0.9656       1.8074       1.1590       2.0674       0.7489
##          X6           X7           X8           X9          X10
##      2.2718       0.9710       1.6800       0.8071       1.8871
```

1(b)

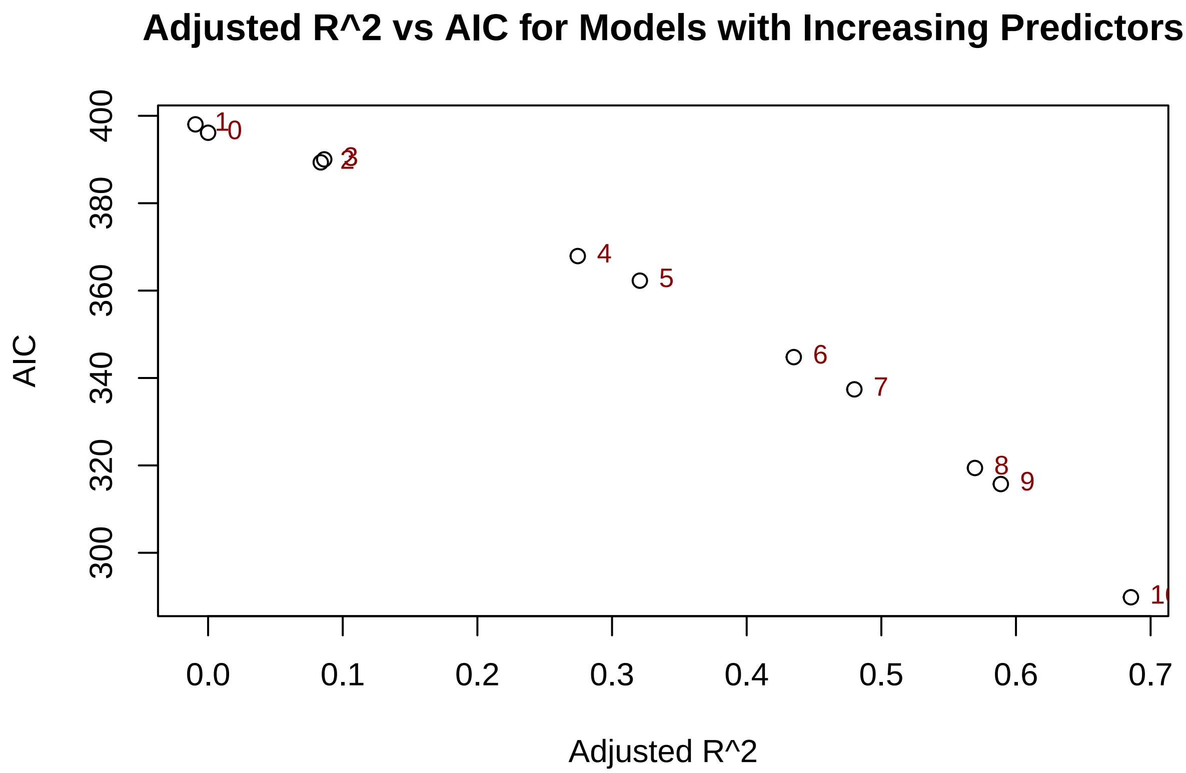
Create a plot with the adjusted R^2 on the x-axis and the AIC on the y-axis. Label each point with the number of variables used. What do you observe?

(Hint: if our model is called `lm1`, we can extract the adjusted R^2 using `summary(lm1)$adj.r.squared` and the AIC using `AIC(lm1)`)

```r
# Use sapply to extract the adjusted R^2 and AIC values
adj_r_squared <- sapply(models, FUN = function(model) summary(model)$adj.r.squared)
AIC_values <- sapply(models, FUN = AIC)

# Create the plot
plot(x = adj_r_squared, y = AIC_values,
  xlab = "Adjusted R^2", ylab = "AIC",
  main = "Adjusted R^2 vs AIC for Models with Increasing Predictors"
)

# Add labels to the points
text(x = adj_r_squared, y = AIC_values,
  labels = 1:11 - 1, pos = 4, cex = 0.8, col = "darkred"
)
```



There is a clear negative relationship between the adjusted R^2 and the AIC. This is expected as the AIC is a measure of model fit that penalises the number of parameters in the model. We also see that the AIC is minimised when the adjusted R^2 is maximised when we use all 10 predictors.

1(c)

We posit the following model for the relationship between the response variable Y and the predictor variable X:

$$Y = \beta_0 + \beta_1 X + \epsilon$$

where $\epsilon \sim \text{Lapace}(0, \sigma)$. The likelihood function for the model is given by:

$$l(\hat{y}, \hat{\sigma}; x) = \frac{1}{2\hat{\sigma}} \exp\left(-\frac{|y - \hat{y}|}{\hat{\sigma}}\right),$$

where $\hat{y} = \hat{\beta}_0 + \hat{\beta}_1 x$. Let's say that we have n observations, then the maximum likelihood estimate for σ is given by:

$$\hat{\sigma} = \frac{1}{n} \sum_{i=1}^{n} |y_i - \hat{y}_i|.$$

Write out an expression for the AIC in terms of the mean absolute error.

$$
\begin{aligned}
\text{AIC} &= 2k - 2\log(l(\hat{y}, \hat{\sigma}; x)), \\
&= 2k - 2\log(\prod_{i=1}^{n}(\frac{1}{2\hat{\sigma}}\exp\left(-\frac{|y_i - \hat{y}_i|}{\hat{\sigma}}\right))), \\
&= 2k - 2\log((2\hat{\sigma})^{-n}\exp\left(-\sum_{i=1}^{n}\frac{|y_i - \hat{y}_i|}{\hat{\sigma}}\right)), \\
&= 2k - 2[-n\log(2\hat{\sigma}) - \frac{\sum_{i=1}^{n}|y_i - \hat{y}_i|}{\hat{\sigma}}], \\
&= 2k + 2[n\log(2\hat{\sigma}) + \frac{\sum_{i=1}^{n}|y_i - \hat{y}_i|}{\hat{\sigma}}], \\
&= 2k + 2[n\log(2) + n\log(\hat{\sigma}) + \frac{\sum_{i=1}^{n}|y_i - \hat{y}_i|}{\frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i|}], \\
&= 2k + 2[n\log(2) + n\log(\text{MAE}) + n], \\
&= 2k + 2n(\log(2) + \log(\text{MAE}) + 1), \\
&= 2k + 2n\log(2) + 2n\log(\text{MAE}) + 2n,
\end{aligned}
$$

Question 2

In this question, we will utilise the `mtcars` dataset in R. The dataset contains information about 32 cars from the 1974 model year.

```
mtcars
```

```
##                      mpg cyl  disp  hp drat    wt  qsec vs am gear carb
## Mazda RX4           21.0   6 160.0 110 3.90 2.620 16.46  0  1    4    4
## Mazda RX4 Wag       21.0   6 160.0 110 3.90 2.875 17.02  0  1    4    4
## Datsun 710          22.8   4 108.0  93 3.85 2.320 18.61  1  1    4    1
## Hornet 4 Drive      21.4   6 258.0 110 3.08 3.215 19.44  1  0    3    1
## Hornet Sportabout   18.7   8 360.0 175 3.15 3.440 17.02  0  0    3    2
## Valiant             18.1   6 225.0 105 2.76 3.460 20.22  1  0    3    1
## Duster 360          14.3   8 360.0 245 3.21 3.570 15.84  0  0    3    4
## Merc 240D           24.4   4 146.7  62 3.69 3.190 20.00  1  0    4    2
## Merc 230            22.8   4 140.8  95 3.92 3.150 22.90  1  0    4    2
## Merc 280            19.2   6 167.6 123 3.92 3.440 18.30  1  0    4    4
## Merc 280C           17.8   6 167.6 123 3.92 3.440 18.90  1  0    4    4
## Merc 450SE          16.4   8 275.8 180 3.07 4.070 17.40  0  0    3    3
## Merc 450SL          17.3   8 275.8 180 3.07 3.730 17.60  0  0    3    3
## Merc 450SLC         15.2   8 275.8 180 3.07 3.780 18.00  0  0    3    3
## Cadillac Fleetwood  10.4   8 472.0 205 2.93 5.250 17.98  0  0    3    4
## Lincoln Continental 10.4   8 460.0 215 3.00 5.424 17.82  0  0    3    4
## Chrysler Imperial   14.7   8 440.0 230 3.23 5.345 17.42  0  0    3    4
## Fiat 128            32.4   4  78.7  66 4.08 2.200 19.47  1  1    4    1
## Honda Civic         30.4   4  75.7  52 4.93 1.615 18.52  1  1    4    2
## Toyota Corolla      33.9   4  71.1  65 4.22 1.835 19.90  1  1    4    1
## Toyota Corona       21.5   4 120.1  97 3.70 2.465 20.01  1  0    3    1
## Dodge Challenger    15.5   8 318.0 150 2.76 3.520 16.87  0  0    3    2
## AMC Javelin         15.2   8 304.0 150 3.15 3.435 17.30  0  0    3    2
## Camaro Z28          13.3   8 350.0 245 3.73 3.840 15.41  0  0    3    4
## Pontiac Firebird    19.2   8 400.0 175 3.08 3.845 17.05  0  0    3    2
## Fiat X1-9           27.3   4  79.0  66 4.08 1.935 18.90  1  1    4    1
```

```
## Porsche 914-2       26.0   4 120.3  91 4.43 2.140 16.70  0  1    5    2
## Lotus Europa        30.4   4  95.1 113 3.77 1.513 16.90  1  1    5    2
## Ford Pantera L      15.8   8 351.0 264 4.22 3.170 14.50  0  1    5    4
## Ferrari Dino        19.7   6 145.0 175 3.62 2.770 15.50  0  1    5    6
## Maserati Bora       15.0   8 301.0 335 3.54 3.570 14.60  0  1    5    8
## Volvo 142E          21.4   4 121.0 109 4.11 2.780 18.60  1  1    4    2
```

2(a)

Five of the variables are more factors than numeric variables. Identify these variables, and convert them to factors.

```r
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  
```

```r
str(mtcars)
```

```
## 'data.frame':    32 obs. of  11 variables:
##  $ mpg : num  21 21 22.8 21.4 18.7 18.1 14.3 24.4 22.8 19.2 ...
##  $ cyl : num  6 6 4 6 8 6 8 4 4 6 ...
##  $ disp: num  160 160 108 258 360 ...
##  $ hp  : num  110 110 93 110 175 105 245 62 95 123 ...
##  $ drat: num  3.9 3.9 3.85 3.08 3.15 2.76 3.21 3.69 3.92 3.92 ...
##  $ wt  : num  2.62 2.88 2.32 3.21 3.44 ...
##  $ qsec: num  16.5 17 18.6 19.4 17 ...
##  $ vs  : num  0 0 1 1 0 1 0 1 1 1 ...
##  $ am  : num  1 1 1 0 0 0 0 0 0 0 ...
##  $ gear: num  4 4 4 3 3 3 3 4 4 4 ...
##  $ carb: num  4 4 1 1 2 1 4 2 2 4 ...
```

It seems like the cyl, vs, am, gear, and carb variables can be converted to factors.

```r
variables_to_factors <- c("cyl", "vs", "am", "gear", "carb")
mtcars[, variables_to_factors] <- lapply(mtcars[, variables_to_factors], factor)
str(mtcars)
```

```
## 'data.frame':    32 obs. of  11 variables:
##  $ mpg : num  21 21 22.8 21.4 18.7 18.1 14.3 24.4 22.8 19.2 ...
##  $ cyl : Factor w/ 3 levels "4","6","8": 2 2 1 2 3 2 3 1 1 2 ...
##  $ disp: num  160 160 108 258 360 ...
##  $ hp  : num  110 110 93 110 175 105 245 62 95 123 ...
##  $ drat: num  3.9 3.9 3.85 3.08 3.15 2.76 3.21 3.69 3.92 3.92 ...
##  $ wt  : num  2.62 2.88 2.32 3.21 3.44 ...
##  $ qsec: num  16.5 17 18.6 19.4 17 ...
##  $ vs  : Factor w/ 2 levels "0","1": 1 1 2 2 1 2 1 2 2 2 ...
##  $ am  : Factor w/ 2 levels "0","1": 2 2 2 1 1 1 1 1 1 1 ...
##  $ gear: Factor w/ 3 levels "3","4","5": 2 2 2 1 1 1 1 2 2 2 ...
##  $ carb: Factor w/ 6 levels "1","2","3","4",..: 4 4 1 1 2 1 4 2 2 4 ...
```

2(b)

We are going to try to predict the number of cylinders (`cyl`) based on the remaining variables. Split the dataset into a training set and a test set. The training set should contain 70% of the data and the test set should contain the remaining 30%.

```r
train_indices <- sample(1:nrow(mtcars), 0.7*nrow(mtcars))
train_set <- mtcars[train_indices, ]
test_set <- mtcars[-train_indices, ]
```

2(c)

Fit a k-nearest neighbours model to the training set using 3-fold cross validation to determine k. What is the optimal value of k and the associate accuracy of the model (for both the training and test set)?

```r
library(class)
library(caret)
```

```
## Loading required package: ggplot2
```

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

```r
# Fit the knn model using 3 fold cross validation whilst optimising k
knn_fit <- train(cyl ~ .,
                 data = train_set,
                 method = "knn",
                 trControl = trainControl(method = "cv",
                                          number = 3))

# Extract the optimal value of k
optimal_k <- knn_fit$bestTune$k
optimal_k
```

```
## [1] 9
```

```r
# Accuracy of the training set
train_pred <- predict(knn_fit, train_set)
train_pred
```

```
##  [1] 8 8 8 4 8 4 4 4 4 8 4 8 4 8 4 8 4 4 8 8 4 4
## Levels: 4 6 8
```

```r
# Calculate the accuracy
accuracy_train <- mean(train_set$cyl == train_pred)
accuracy_train
```

```
## [1] 0.8636364
```

```r
# Accuracy of the test set
test_pred <- predict(knn_fit, test_set)
test_pred
```

```
##  [1] 4 8 4 4 8 8 8 8 8 4
## Levels: 4 6 8
```

```r
# Calculate the accuracy
accuracy_test <- mean(test_set$cyl == test_pred)
accuracy_test
```

```
## [1] 0.6
```