STAT 206 Lab 8

Due Monday, November 27, 5:00 PM

General instructions for labs: You are encouraged to work in pairs to complete the lab. Labs must be completed as an R Markdown file. Be sure to include your lab partner (if you have one) and your own name in the file. Give the commands to answer each question in its own code block, which will also produce plots that will be automatically embedded in the output file. Each answer must be supported by written statements as well as any code used.

Agenda: Fit polynomial regression models to the electricity usage data, use K-fold cross-validation to automatically select degree of the polynomial

Polynomial regression

The polynomial regression model posits that a response variable Y and explanatory variable X are related by the equation.

$$Y = \sum_{j=0}^{d} \beta_j X^j + \epsilon \, .$$

The number d is called the degree of the polynomial. Polynomial regression reduces to linear regression when $d = 1$. Its flexibility and complexity increase as d increases. The cases $d = 2$ and $d = 3$ are usually referred to as quadratic and cubic. The polynomial regression model can be expressed as a $d+1$ parameter linear model by considering $(X_0, X_1, X_2, \ldots, X_d)$ as explanatory variables. This is done by `poly()` and can be combined with `lm()` to fit a polynomial regression model. In the following example, we fit a degree-3 polynomial, or cubic, regression model using variables y and x in the dataframe `df`.

```
degree <- 3
obj <- lm(y ~ poly(x, degree), data = df)
```

‘electemp’ dataset

The ‘electemp’ dataset has 55 observations on monthly electricity usage and average temperature for a house in Westchester County, New York

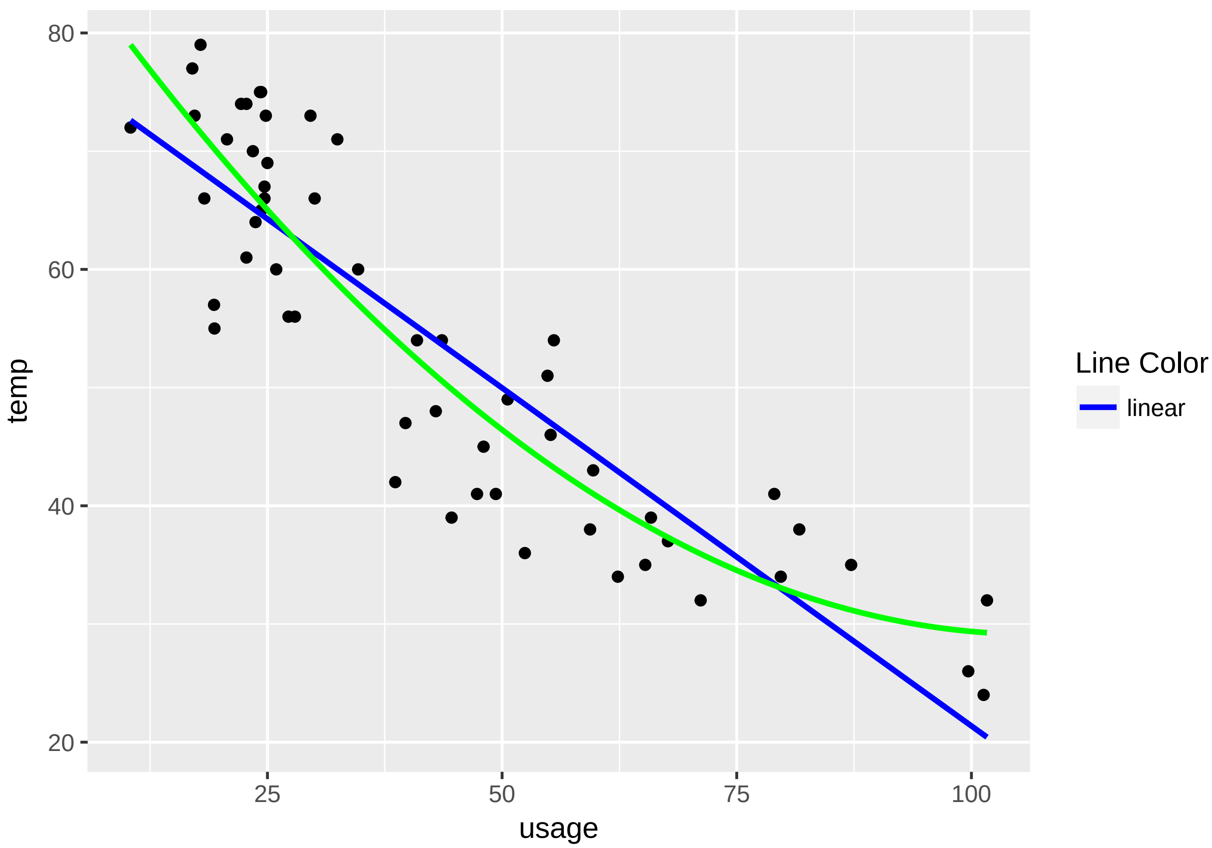
```
url <- 'http://www.faculty.ucr.edu/~jflegal/electemp.txt'
electemp <- read.table(url)
```

1. Create a scatterplot of `temp` and `usage` with `ggplot2` that includes the least squares fits of a linear and quadratic regression models. You should also include a legend on the plot.

```
library(ggplot2)
```

```
## Warning: package 'ggplot2' was built under R version 3.4.2
```

```
pic = ggplot(electemp, aes( x =  usage, y = temp))  +
  geom_point()  +
  geom_smooth(method='lm', se = FALSE, aes( colour = 'linear')) +
  geom_smooth(method = "lm", formula = y ~ x + I(x^2), color = "green", se = FALSE, aes(colour = 'square
  scale_colour_manual(name="Line Color", values=c(squared="red", linear="blue"))

pic
```



2. Does the linear or quadratic model fit the data better?

```
#it looks like the quadratic fit line captures the data a little bit better than the straight line
```

3. Write a function `cv_poly()` that performs K-fold cross-validation to estimate the mean squared prediction error (MSPE) of polynomial regression. It takes vectors x and y containing observations of the explanatory and response variables, a vector degree of the degrees of polynomial models to fit, and a number K indicating the number of folds for cross-validation. It returns a $K \times D$ matrix, where K is the number of folds and D is the number of different degree models that are being fit. The entries of the matrix are the MSPE for each fold and degree polynomial model being fit.

```
#This code is very similar to the code we used in class lecture.
#The main difference is the use of lm for the obj instead of loess
#This is so we can control the degree of poly

cv_poly = function(K, data, s = C()) {
  n = nrow(data)
  cv.error = matrix(nrow = K, ncol = length(s))
  foldid = sample(rep(1:K, length = n))
  answers = c()

  for(i in 1:K) {
    cv.error[i, ] = sapply(s, function(span) {
      obj = lm(temp ~ poly(usage, span), data = electemp)
      y.hat = predict(obj, newdata = subset(electemp, foldid == i))
      pse <- mean((subset(electemp, foldid == i)$temp - y.hat)^2)
```

```
    })

  }
  return(cv.error)
}
```

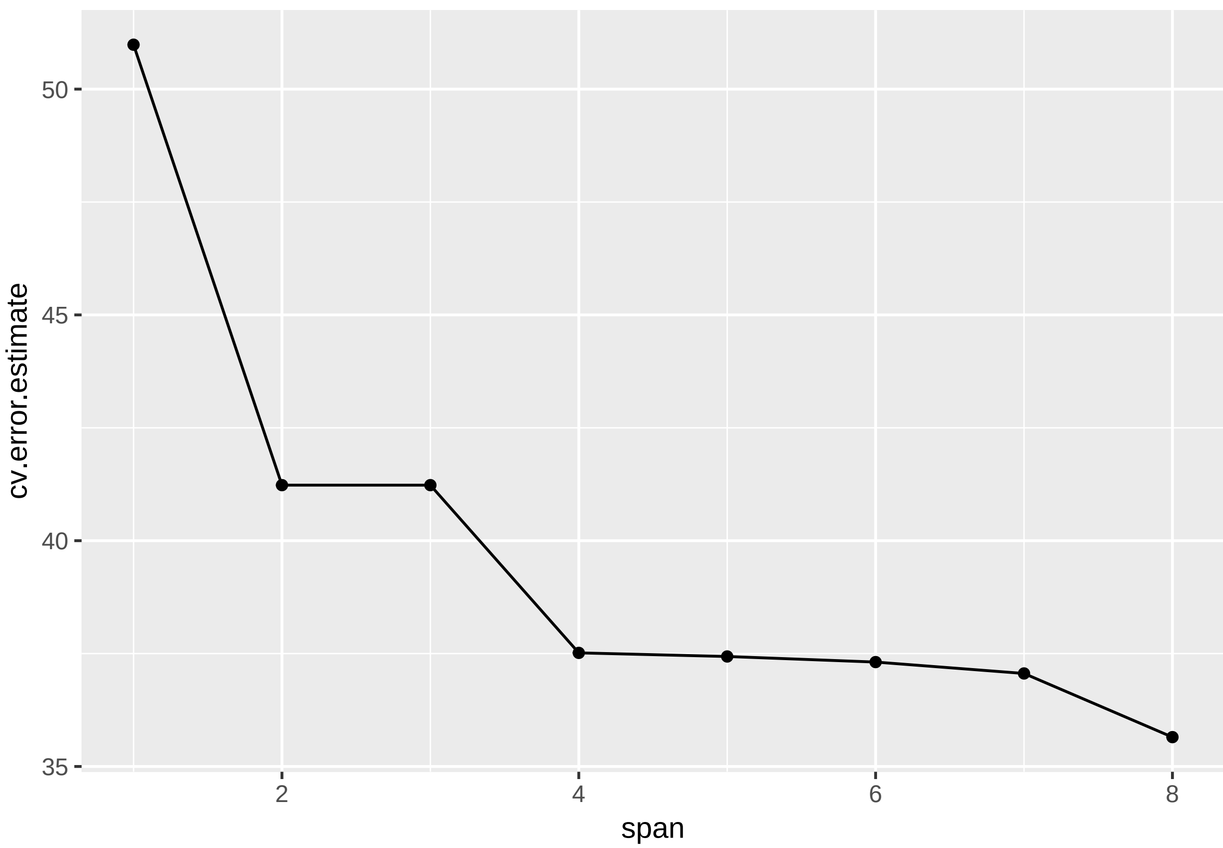
4. Use `cv_poly()` to estimate the MSPE of polynomial regression on the electricity usage data by $K = 10$-fold cross-validation for $d = 1, 2, ..., 8$. Note that `cv_poly()` should return a matrix, call it `cv_error` with K rows corresponding to the K different validation sets.

```
s = seq(from = 1, to = 8, by = 1)

cv_poly(K = 10, data = electemp, s = seq(from = 1, to = 8, by = 1))
```

```
##            [,1]     [,2]     [,3]     [,4]      [,5]      [,6]      [,7]
##  [1,]  48.09505 13.52704 13.50545 16.26367 15.898043 15.343202 17.329292
##  [2,]  62.16981 65.88162 65.71618 49.55351 51.664566 49.704784 47.136983
##  [3,]  49.11482 53.04520 53.00827 48.77345 49.115902 47.299473 44.397779
##  [4,]  49.07737 47.51575 47.61885 53.09619 51.905457 54.073062 58.175183
##  [5,] 100.76474 90.41148 90.46418 89.97779 87.099270 87.608778 82.573684
##  [6,]  27.46050 28.06511 28.14419 20.50515 21.277704 22.014835 22.984403
##  [7,]  56.69058 35.69584 35.67714 30.05148 30.075514 30.982982 32.633490
##  [8,]  37.65557 30.48502 30.47570 27.46262 27.315069 27.257290 26.951313
##  [9,]  62.54940 32.21981 32.23805 28.29474 28.678999 28.373276 30.398228
## [10,]  17.37877 10.77353 10.78492  6.18129  6.230603  5.801774  4.725737
##            [,8]
##  [1,] 13.542079
##  [2,] 48.133721
##  [3,] 40.104627
##  [4,] 55.591493
##  [5,] 68.196973
##  [6,] 25.005244
##  [7,] 32.880471
##  [8,] 26.563167
##  [9,] 42.084562
## [10,]  4.223125
```

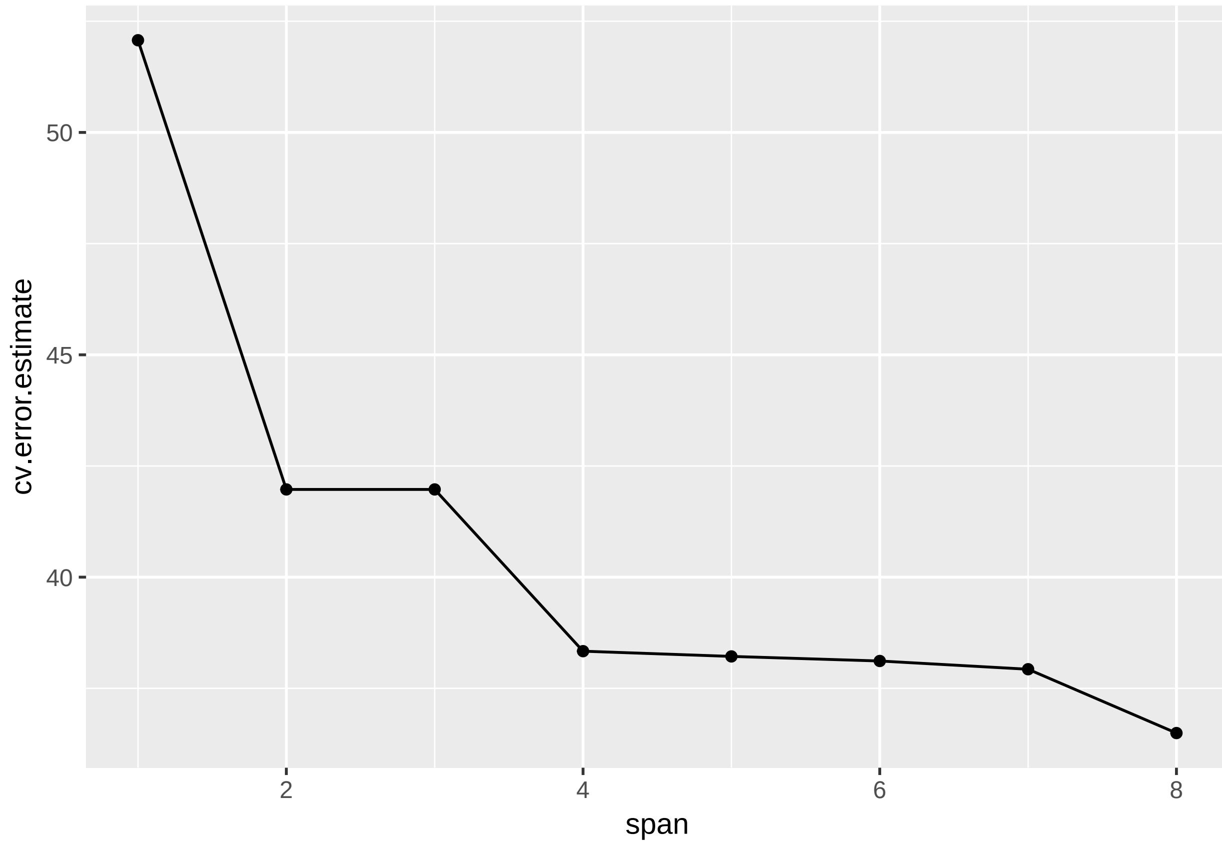
5. Plot the estimated MSPE (by averaging across the K folds) versus degree of the polynomial. What degree polynomial would you select according to cross-validation?

```
# We first must get the average of the columns which is getting the average per poly degree
# From the graph we can see that a poly degree of 8 yields the least squared deviation

k_10 = cv_poly(K = 10, data = electemp, s = seq(from = 1, to = 8, by = 1))
cv.error.estimate <- colMeans(k_10)
qplot(1:8, cv.error.estimate, geom=c('line', 'point'), xlab='span')
```

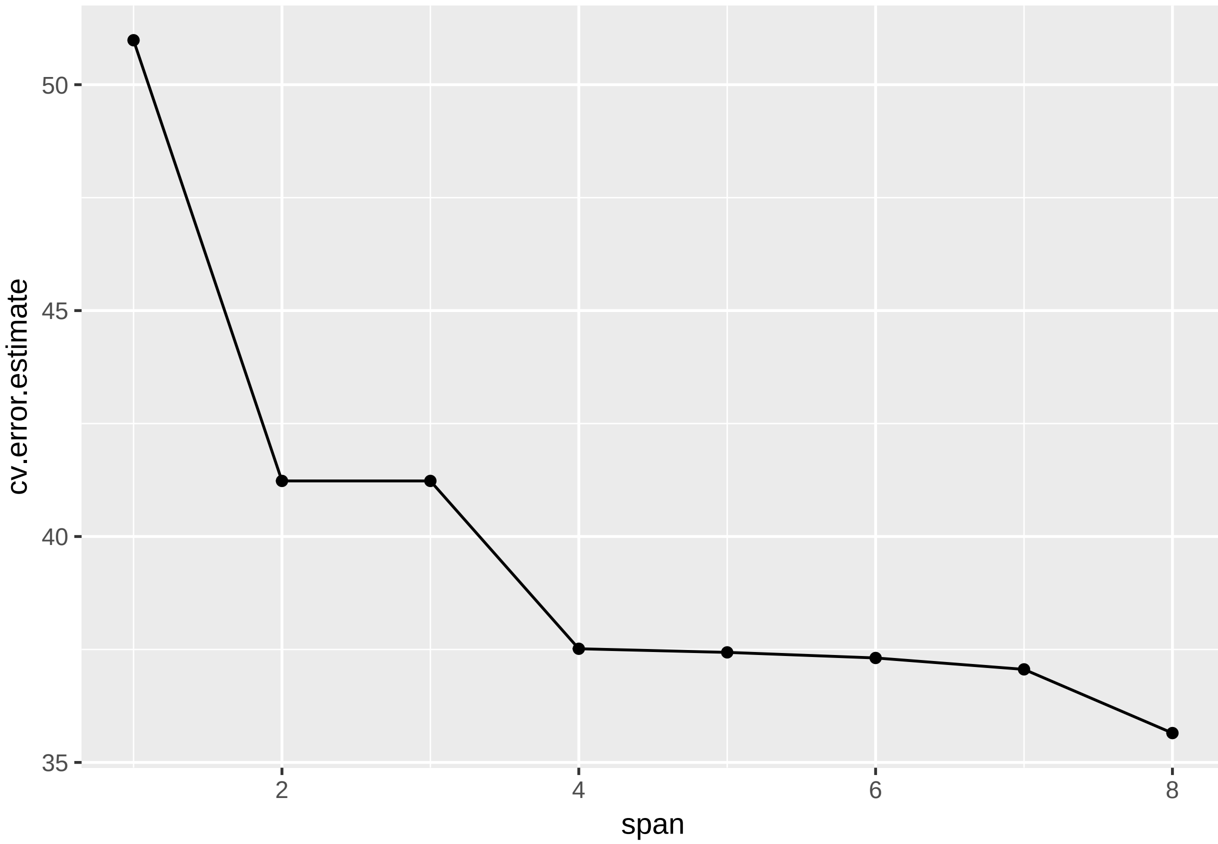


6. Repeat the preceding problem for $K = 5$ and leave-one-out cross-validation ($K = n$). What do you notice about the time it takes to compute the cross-validation? How do the results change with K?

```
#First we will use K=5

k_5 = cv_poly(K = 5, data = electemp, s = seq(from = 1, to = 8, by = 1))
cv.error.estimate <- colMeans(k_5)
qplot(1:8, cv.error.estimate, geom=c('line', 'point'), xlab='span')
```




```r
#Now we will use K = n (55)

k_n = cv_poly(K = 55, data = electemp, s = seq(from = 1, to = 8, by = 1))
cv.error.estimate <- colMeans(k_10)
qplot(1:8, cv.error.estimate, geom=c('line', 'point'), xlab='span')
```



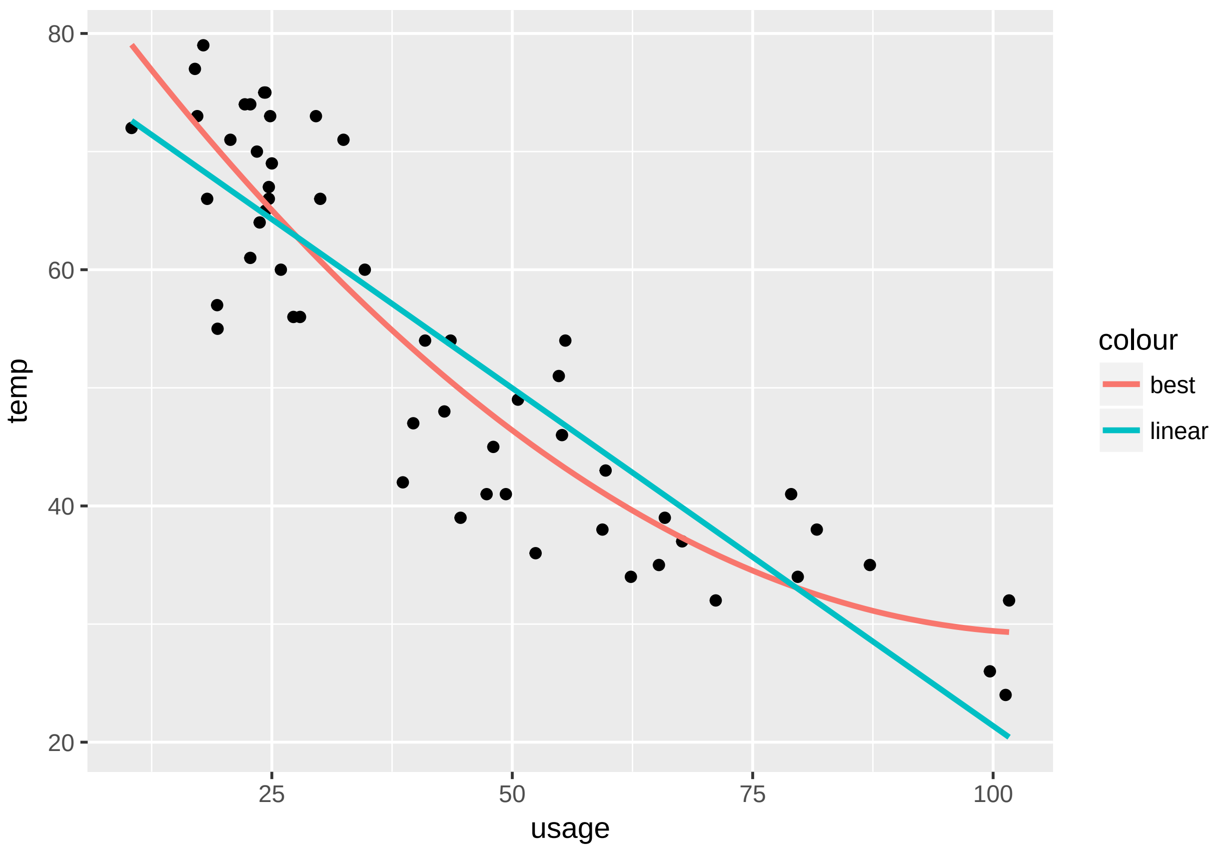

```
#The time it took to run K=n was much longer, we notice that the graph seems identical. This
#Means that running n number of folds (leave-one-out) method is overkill
```

7. Plot the estimated MSPE versus degree of the polynomial. What degree polynomial would you select according to cross-validation? Are there differences between $K = 5$, $K = 10$, and leave-one-out estimates of MSPE?

```
#The plots for the 3 cases are above and they all look identical. In all 3 cases I would choose
#a poly degree of 8
```

8. Reproduce your first plot and add a layer showing the polynomial regression model selected by cross-validation by modifying the following code.

```
s.best <- s[which.min(cv.error.estimate)]
qplot(usage, temp, data=electemp) +
  geom_smooth(method = 'loess', span = s.best, se = FALSE, aes( colour = 'best')) +
  geom_smooth(method='lm', se = FALSE, aes( colour = 'linear'))
```


80
60
40
20
temp
25
50
75
100
usage
colour
best
linear