STAT 206 Lab 8_Lihua_Xu

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

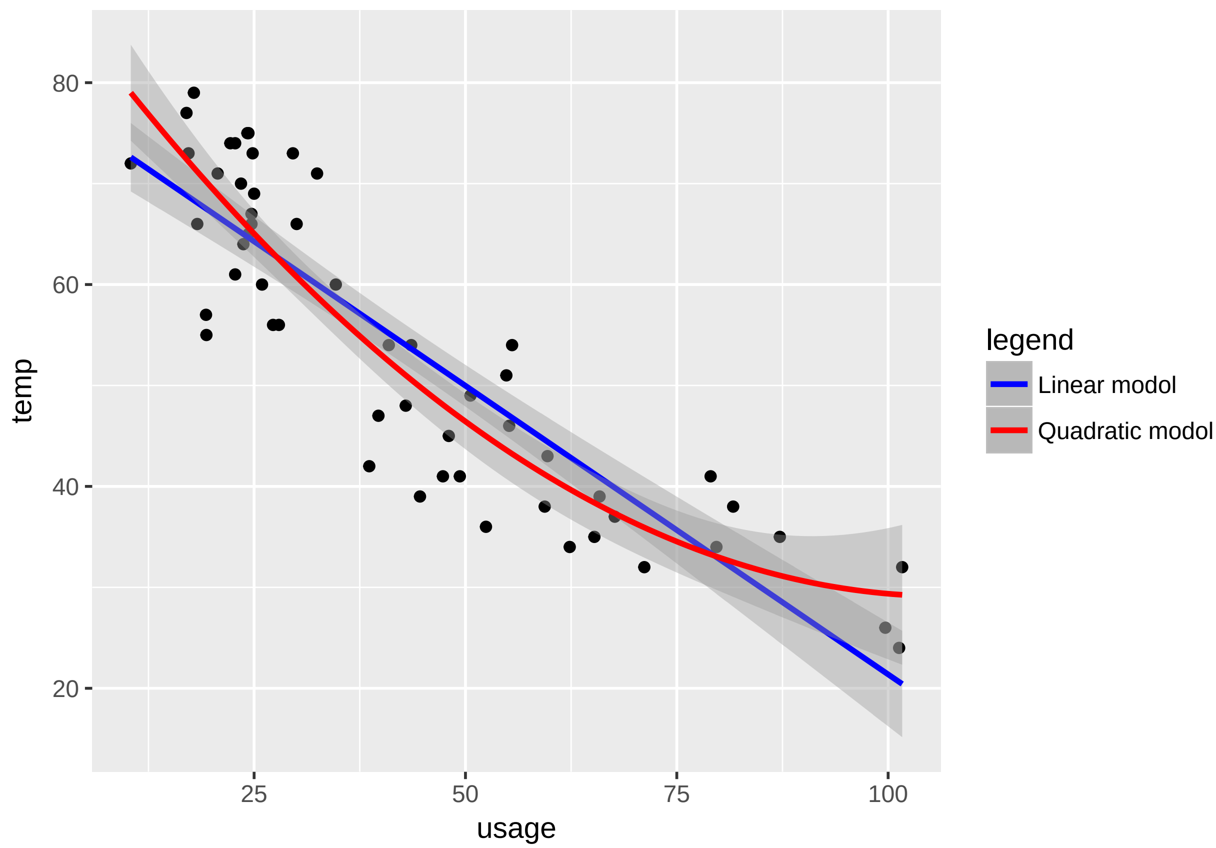
```
url <- 'http://www.faculty.ucr.edu/~jflegal/electemp.txt'
electemp <- read.table(url)
```

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
```

```
ggplot(data=electemp,aes(x=usage,y=temp))+geom_point()+
  geom_smooth(aes(colour="Linear modol"),method = "lm", formula = y ~ x)+
  geom_smooth(aes(colour="Quadratic modol"),method = "lm", formula = y ~ poly(x,2))+
  scale_colour_manual(name="legend", values=c("blue", "red"))
```



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

```
#Yes, they both fit the datat but I think the quadratic regression model
#fit the data better thant the linear model.
```

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.

```
cv_poly <- function(K,V,D){
n <- nrow(V)
cv.error <- matrix(nrow=K, 1)
foldid <-sample(rep(1:K, length = n))
for(i in 1:K) {
  cv.error[i, ] <- sapply(D, function(LINEAR) {
    obj <- lm(temp ~ poly(usage, D), data = subset(V, foldid != i))
    y.hat <- predict(obj, newdata = subset(V, foldid == i))
    temp_data <- (subset(V, foldid == i)$temp - y.hat)^2
    pse <- mean(temp_data)
    return(pse)})
}
return(as.matrix(cv.error))
}
```

4. Use `cv_poly()` to estimate the MSPE of polynomial regression on the electricity usage data by $K = 10$-

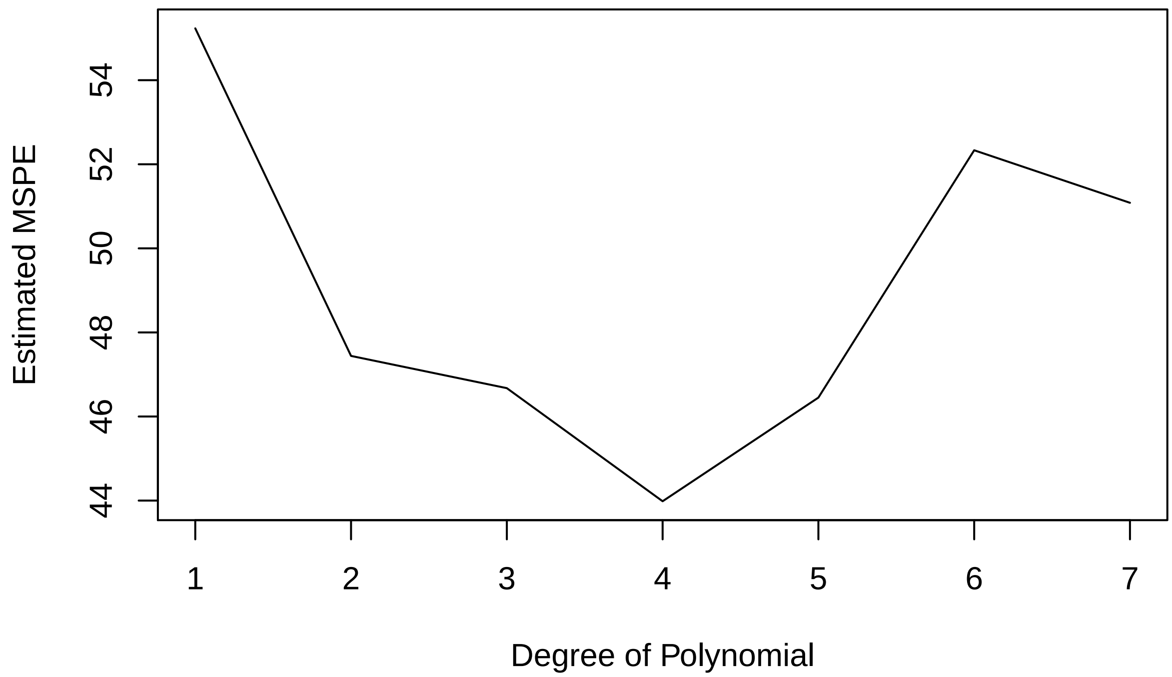
fold cross-validation for $d = 1, 2, 3, 4, 5, 6, 7, 8$. Note that `cv_poly()` should return a matrix, call it `cv_error` with K rows corresponding to the K different validation sets.

```
mspe_10_8 <- cbind()
for (m in 1:8){
value <- cv_poly(10,electemp,m)
mspe_10_8 <- cbind(mspe_10_8,value)
}
mspe_10_8
```

```
##           [,1]      [,2]     [,3]      [,4]      [,5]      [,6]     [,7]
##  [1,] 68.57367  58.69994 58.15999 106.39961 74.231120 61.366365 41.27608
##  [2,] 98.01042  12.35388 47.83360  95.25356 28.854795 70.393529 33.08363
##  [3,] 82.51065  38.00864 64.53054  16.26847 41.216204 93.159119 67.71689
##  [4,] 19.24480  26.47806 55.67950  21.99821  9.654364 80.406401 54.20998
##  [5,] 29.42145  59.90728 21.17318  16.54667 57.930385 16.754214 52.83144
##  [6,] 39.99214  80.54832 16.65599  45.84007 40.430688 66.866439 61.10995
##  [7,] 40.43865  35.32734 15.16904  34.48271 79.184614  6.351718 34.52371
##  [8,] 77.73382  21.56376 81.52350  23.52984 32.595966 46.140435 56.59822
##  [9,] 43.09971  35.62276 20.31659  20.07214 77.385734 34.790819 70.47592
## [10,] 53.30140 105.90197 85.70149  59.45058 23.009171 47.098661 39.00688
##             [,8]
##  [1,]   41.39394
##  [2,]   50.59452
##  [3,]   14.71821
##  [4,]   25.31679
##  [5,]   74.43585
##  [6,]   62.31552
##  [7,]   21.89229
##  [8,]   42.98513
##  [9,]  129.90748
## [10,] 2718.14379
```

6. 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?

```
#It seems degree < 8 make sense.
plot(1:7,colMeans(mspe_10_8)[1:7],type="l",xlab="Degree of Polynomial",ylab="Estimated MSPE")
```




```
#For me, I will chose degree=4 as the estimate MSPE is the lowest.
```

7. 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?

```
#K=5
mspe_5_8 <- cbind()
for (m in 1:8){
value <- cv_poly(5,electemp,m)
mspe_5_8 <- cbind(mspe_5_8,value)
}
mspe_5_8
```

```
##          [,1]     [,2]     [,3]     [,4]     [,5]     [,6]      [,7]
## [1,] 62.33949 34.97595 45.35934 33.20663 65.74567 23.73025  24.65239
## [2,] 26.47687 63.43707 65.22669 29.54450 37.94594 51.70135 140.61404
## [3,] 44.05892 57.74683 50.57194 39.44185 18.21248 36.44290  64.29610
## [4,] 97.91813 41.63141 53.09175 48.72802 68.68635 45.92005  56.54894
## [5,] 48.82725 51.59956 39.24921 62.76634 53.11779 66.21482  68.26003
##            [,8]
## [1,]   38.84775
## [2,] 1181.61369
## [3,]   40.46051
## [4,]   80.96501
## [5,]   36.28147
```

```
#leave-one-out cross-validation
loocv <- function(D){
```

```r
loocv_tmp <- matrix(NA, nrow = nrow(electemp), 1)
index <- 1:nrow(electemp)
for (k in 1:nrow(electemp)) {
  loocv_tmp[k, ] <- sapply(D, function(LINEAR) {
    obj <- lm(temp ~ poly(usage, D), data = subset(electemp, index != k))
    y.hat <- predict(obj, newdata = subset(electemp, index == k))
    temp_data <- (subset(electemp, index == k)$temp - y.hat)^2
    pse <- mean(temp_data)
    return(pse)})
}
return(loocv_tmp)}

loocv_mspe_8 <- cbind()
for (m in 1:8){
value <- loocv(m)
loocv_mspe_8 <- cbind(loocv_mspe_8,value)
}
loocv_mspe_8
```

```
##             [,1]        [,2]        [,3]        [,4]         [,5]
##  [1,]  79.0333740  64.8199353  65.1660979  45.3095240  43.86898592
##  [2,]   6.9070537   3.0158672   3.0191839   0.1220666   0.04871812
##  [3,] 119.3535248 193.4268131 195.6111097 195.3695927 205.37944694
##  [4,]   2.1315554   4.3473479   4.4298912  14.4856780  12.14394371
##  [5,]  25.4116956  19.4237287  21.3441044  14.9159418  12.44760056
##  [6,]  90.9759789  37.1152653  37.4263680  18.6210726  17.27481698
##  [7,]  46.1052705  11.1673100  11.2572717   2.8309182   4.47868752
##  [8,]   1.0764383   7.3254786   7.3373668  23.4322865  22.04499648
##  [9,]  54.0587170 127.5624085 127.7293530 184.6833622 181.48718180
## [10,]  14.7519547  18.8788143  19.1465336  35.0979640  35.93625361
## [11,]  19.5123139   4.6818972   4.7075095   2.6798747   1.97202310
## [12,] 113.7417857  93.5787034  93.9272971  70.8028159  68.80635568
## [13,]  76.1170954  52.1157485  52.1212094  37.9443536  35.90058028
## [14,]   2.5534874   0.4970544   0.4941646   0.4830291   0.68268216
## [15,]  22.1006669  39.2112809  39.2407507  54.7934984  58.61377318
## [16,]   0.4206752   9.3521084   9.3668266  26.2579424  27.47385055
## [17,]  65.4987183  67.7100781  76.3008254  28.1119994  30.76146133
## [18,]  46.8638835  17.6941160  18.9467961   1.0109748   0.14664948
## [19,] 114.4364607  54.5676085  55.3932104  34.2912518  31.82182184
## [20,] 217.8178469 154.5723379 161.6993897 158.8623753 157.12498711
## [21,]  45.8097725  44.6775264  45.7637881  69.8208000  69.11635382
## [22,]  23.7210685  16.6505372  16.7322864   7.2640206   6.68680536
## [23,]  19.9862158   0.7546233   0.8090995   4.2419914   3.67175505
## [24,]  72.2003897  23.9379578  25.2696499  40.8044243  39.32854602
## [25,]  70.4568642  44.8291416  44.8282596  33.1024643  31.04963047
## [26,]   4.8411365  30.2374816  30.8988693  25.4195779  27.87931973
## [27,]  51.5305897  53.1760109  54.2528770  80.5905080  80.49504339
## [28,]  81.4925253  41.9574178  43.8521347  39.5278601  36.14877989
## [29,]  43.1657292  34.6785214  38.8224307   3.6630241   6.65868819
## [30,]   1.1510068   1.0102854   1.1733929   9.6212834   8.04058645
## [31,]   8.8725785   0.1198320   0.1179177   0.1719785   1.01666491
## [32,]   3.8729275   0.8848861   0.9551944   1.9695972   0.78265854
## [33,]  15.0768357  60.0939399  60.1300139 100.0133239  97.73855405
## [34,]   1.6510789   9.1811790   9.5346790   5.8064968   8.11032045
```

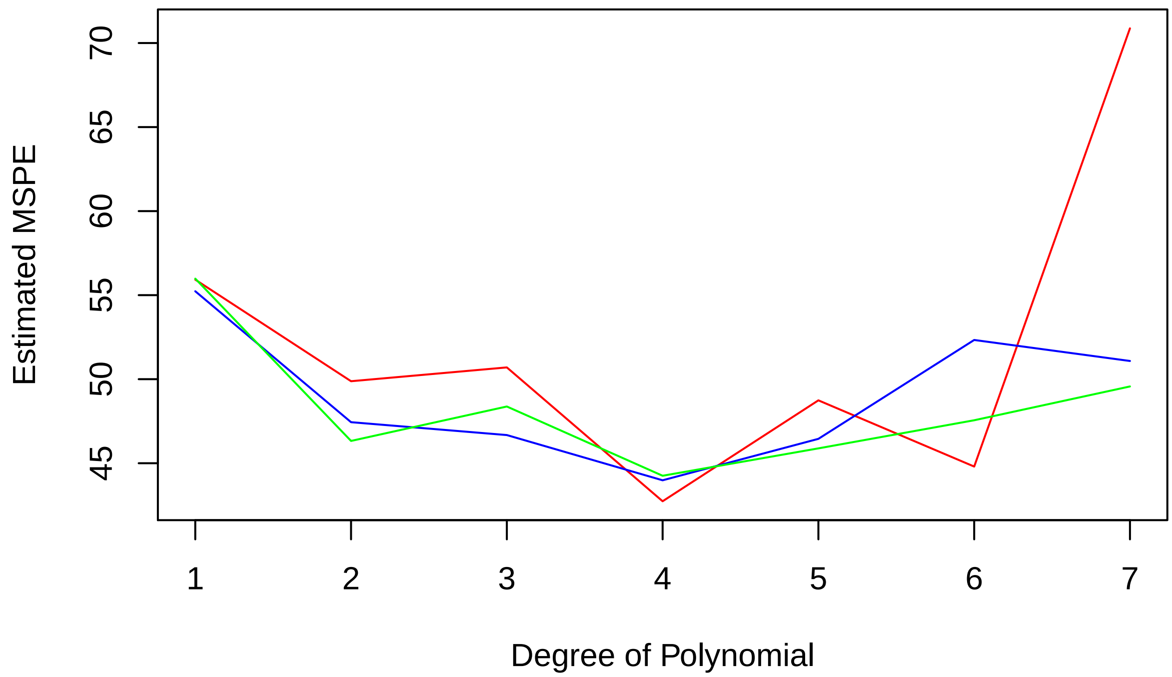
```
## [35,]  24.9590380  13.7340106  13.7410879   6.2559035   5.43197616
## [36,] 135.3518916 148.2656164 152.5362984 122.1023227 125.91322979
## [37,] 126.2477378 156.5999951 162.8594783 138.3672592 148.83411738
## [38,]  22.2839980  28.9174413  29.7507790  17.0233042  18.55076830
## [39,]   0.1341597  12.6857541  12.9777331  22.3147788  27.79828885
## [40,]  38.6488292   7.6409247   7.7563381   1.1123749   0.66692514
## [41,] 165.2223500  88.3232895  88.4186095  56.3169199  56.30623550
## [42,]  84.5057996  36.1573187  37.0282058  23.9711711  31.32613527
## [43,]  37.6621448  17.1022564  18.5578166  30.2722736  37.55078904
## [44,] 204.7598780 127.1915328 130.4354053 104.6100976 101.21483206
## [45,] 168.2748135 254.7805907 257.5744779 257.9740474 270.08407484
## [46,]   1.0532409   5.0072208   5.0317253  12.5320010  13.89065803
## [47,]   0.4183998  64.3093491 105.0651928   0.8465707  50.46696769
## [48,] 122.8973826  62.0570168  64.1842490  77.1896921  74.67826472
## [49,] 112.1986974  91.4161259  91.7180710  69.1043141  67.06841460
## [50,]   0.1287347   0.4236938   0.4302910   4.1016706   4.74153940
## [51,]   1.3881878   2.7263502   2.7946442   5.1296497   7.92063146
## [52,]  37.4829799   9.6923608  10.0501993   4.8535866   3.31792991
## [53,]  41.8337560  12.5910385  13.1019271   9.2657459  14.17405223
## [54,]  14.8644219  51.0744839  61.6937867  21.2607806  26.45806117
## [55,] 176.0531175  14.0102963  17.0396322  77.2637973  72.05189509
##              [,6]         [,7]         [,8]
##  [1,]  42.7348092  39.78117814 3.061676e+01
##  [2,]   0.0184145   0.01684796 9.270633e-01
##  [3,] 203.5890735 198.00511617 1.835414e+02
##  [4,]  13.1701644  10.37823316 6.707140e+00
##  [5,]   8.3533353  72.14777839 1.710731e+01
##  [6,]  15.2597812  13.48348756 2.151868e+01
##  [7,]   3.8826246   6.13350435 1.047979e+01
##  [8,]  26.1411524  24.27486488 1.237679e+01
##  [9,] 196.6912964 192.10520245 1.613927e+02
## [10,]  38.5660458  43.77864551 5.419671e+01
## [11,]   2.7963343   3.47324971 3.055278e+00
## [12,]  67.9495606  64.77001470 5.292767e+01
## [13,]  36.9122209  35.92445200 2.823953e+01
## [14,]   0.8327489   1.40416290 4.107559e+00
## [15,]  57.5472949  58.94062306 7.124878e+01
## [16,]  32.4046978  34.49271053 2.264332e+01
## [17,]  27.6770541  41.29509989 4.133843e+01
## [18,]   5.2763807   1.60143610 9.768286e+00
## [19,]  29.7137771  26.45396570 3.049137e+01
## [20,] 166.9794011 161.95062458 1.281371e+02
## [21,]  76.7376096  85.76498248 9.083168e+01
## [22,]   6.1546712   4.89250783 2.003122e+00
## [23,]   5.8456875  10.58531048 3.002047e+01
## [24,]  47.7470600  64.05419718 1.148509e+02
## [25,]  32.6553956  32.44787915 2.620621e+01
## [26,]  25.1106060  20.99085305 1.157245e+01
## [27,]  87.4751668  97.10742944 1.065023e+02
## [28,]  38.9547365  35.41998401 1.912417e+01
## [29,]   6.3377878  11.19876284 3.492138e+00
## [30,]  10.4990319   6.54568839 8.747486e+00
## [31,]   2.2438618   3.12752838 9.161235e-02
## [32,]   0.3362358   0.03396016 3.072841e+00
```

```
## [33,] 108.4364656 105.35528522 8.073566e+01
## [34,]   6.2851666   6.21931784 1.838641e+01
## [35,]   5.5185143   4.81986966 1.981854e+00
## [36,] 123.4397121 118.40802899 1.239434e+02
## [37,] 147.4970933 143.82122923 1.779169e+02
## [38,]  16.1343160  13.84468438 1.695464e+01
## [39,]  28.3073459  34.79491678 4.577406e+01
## [40,]   0.3054020   0.04776111 6.345891e-01
## [41,]  53.2960302  53.01734260 8.023525e+01
## [42,]  31.8384442  39.71510082 3.671560e+01
## [43,]  51.7778592  50.09373125 2.358454e+01
## [44,]  99.4463064  94.06767999 8.523378e+01
## [45,] 269.3535478 263.89104490 2.481020e+02
## [46,]  13.9350881  15.46012670 2.293890e+01
## [47,]  57.7467546   9.21037752 1.962290e+04
## [48,]  86.8242661 105.45037191 1.463629e+02
## [49,]  66.3678371  63.32421414 5.161129e+01
## [50,]   4.9947576   6.17974853 1.120589e+01
## [51,]   7.3051036  10.01666419 2.303803e+01
## [52,]   3.2934363   1.90722660 1.433279e-01
## [53,]  16.6259331  21.01533688 1.109619e+01
## [54,]  31.6639769  26.76980539 3.961416e+01
## [55,]  68.5551424 136.19399395 6.183649e+01
```

```r
#The time need to compute the cross-validation decreases as K decrease.
#When K decrease, there will be less rows(subsets).
```

8. 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?

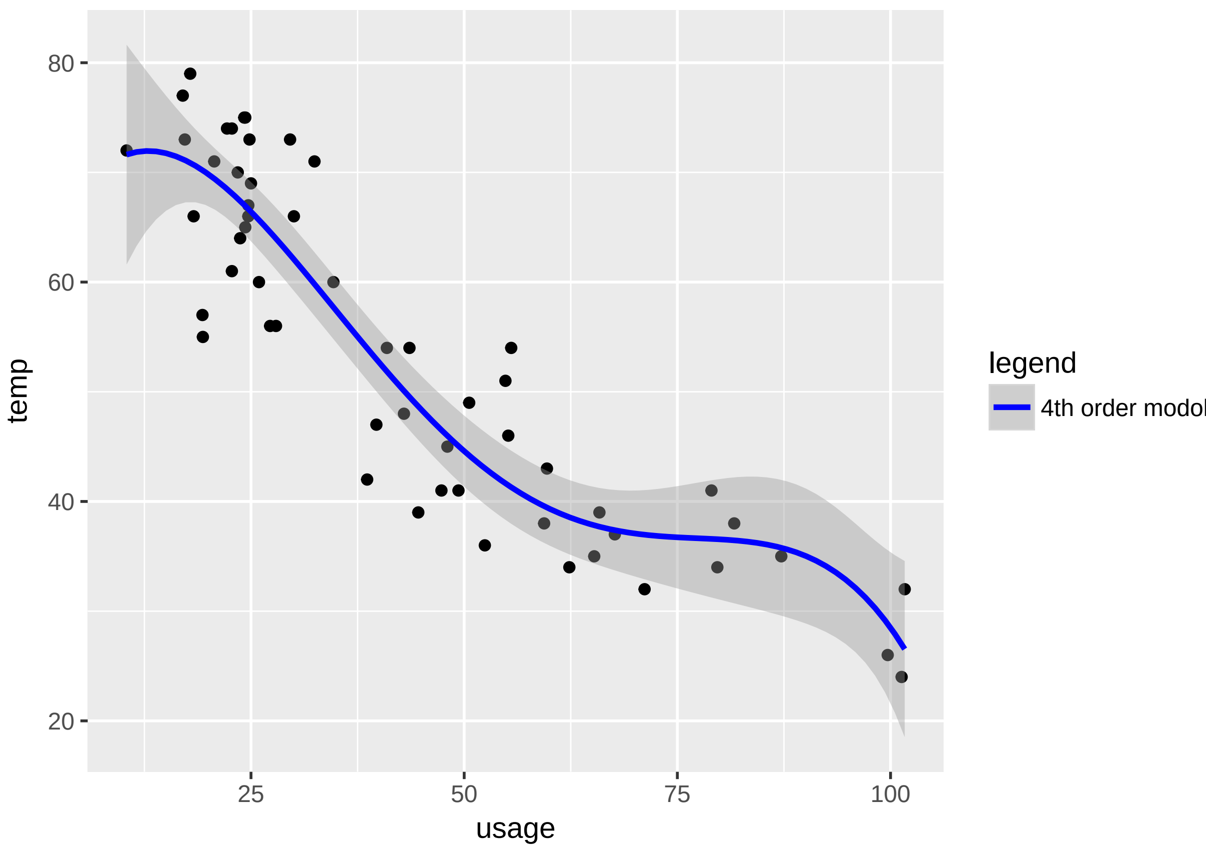
```r
plot(1:7,colMeans(mspe_5_8)[1:7],col="red",type="l",xlab="Degree of Polynomial",ylab="Estimated MSPE")
lines(1:7,colMeans(mspe_10_8)[1:7],col="blue",type="l")
lines(1:7,colMeans(loocv_mspe_8)[1:7],col="green",type="l")
```




```
#the lowest point is also 4(degree). So there are no difference between K=5 ,10 and eave-one-out estima
```

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

```
library(ggplot2)
ggplot(data=electemp,aes(x=usage,y=temp))+geom_point()+
  geom_smooth(aes(colour="4th order modol"),method = "lm", formula = y ~ poly(x,4))+
  scale_colour_manual(name="legend", values=c("blue"))
```



80
60
40
20
temp
25
50
75
100
usage
legend
4th order modol