lab 5 Jin Kweon (3032235207)

Jin Kweon

10/1/2017

For the next two sections of this lab assignment, assume that the data comes from the following model:

$y_i = x_i^T \beta \ + \ \epsilon_i$ where i = 1, ... , n and noises are iid normal with mean 0 and variacne σ^2 and x is fixed.

```
lm1 <- lm(mpg ~ disp + hp, data = mtcars)
sum <- summary(lm1)
sum
```

```
##
## Call:
## lm(formula = mpg ~ disp + hp, data = mtcars)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -4.7945 -2.3036 -0.8246  1.8582  6.9363
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) 30.735904   1.331566  23.083  < 2e-16 ***
## disp        -0.030346   0.007405  -4.098 0.000306 ***
## hp          -0.024840   0.013385  -1.856 0.073679 .
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.127 on 29 degrees of freedom
## Multiple R-squared:  0.7482, Adjusted R-squared:  0.7309
## F-statistic: 43.09 on 2 and 29 DF,  p-value: 2.062e-09
```

```
lowbound <- sum$coefficients[2,1] - qt(0.975, 29) * sum$coefficients[2,2]
upbound <- sum$coefficients[2,1] + qt(0.975, 29) * sum$coefficients[2,2]

list(lowbound, upbound)
```

```
## [[1]]
## [1] -0.04549091
##
## [[2]]
## [1] -0.01520165
```

```
confint(lm1, "disp", level = 0.95)
```

```
##            2.5 %      97.5 %
## disp -0.04549091 -0.01520165
```

1. Our null is beta being equal to zero. (whether we drop the variable or not)
2. Alternative hypothesis will be when beta is not being equal to zero. It is two-sided as t-test contains negative and positive numbers. "***" stands for a number between 0 and 0.001. "." stands for a number between 0.05 and 0.1.
3. No we do not reject the null. P-value is not in the critical region.

MSE = $bias^2$ + variance.

Q. what do you mean by "you can never prove the null"? ==> cuz we never know the null is really true. We just either reject or not reject the null based on our testing. Q. what is orthogonal polynomials in "raw" in poly() function? ==> if we use orthogonal polynomials, then we do gram schmidt to get orthognoal vectors (for inner product $\int_0^1 fg = 0$) Q. So, training set is the entire data set and test set is also the entired data set, as well??? ===> yes

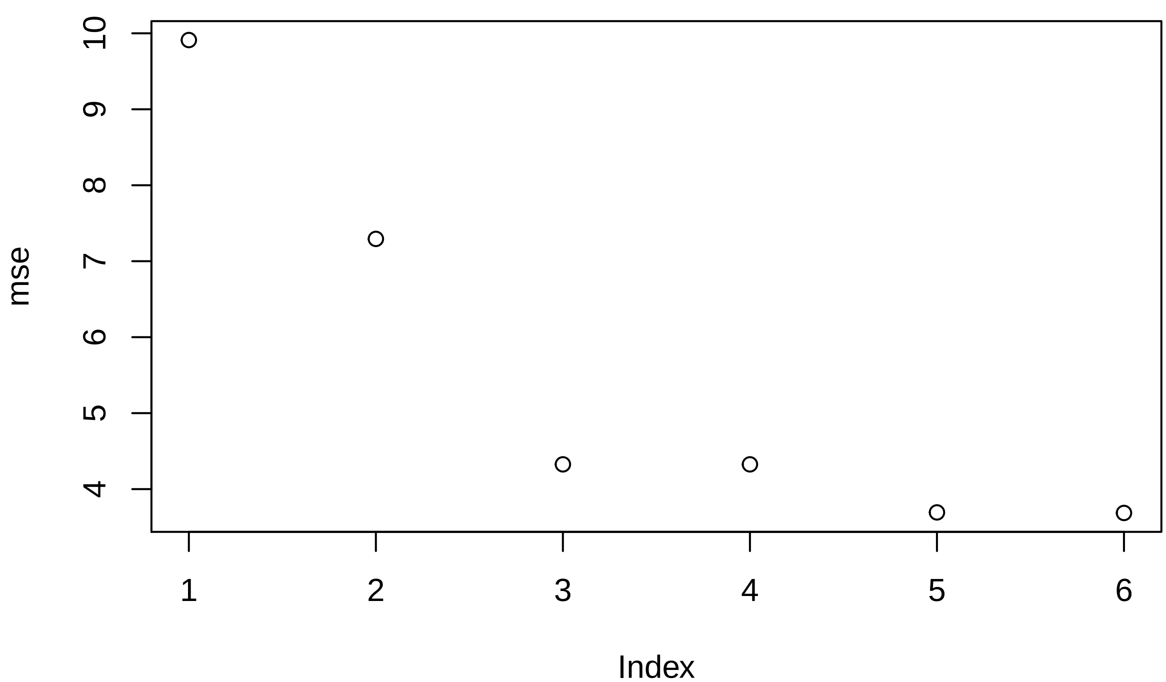
```
#4.
tstat <- (sum$coefficients[2,1] + 0.05) / sum$coefficients[2,2]
pt(tstat, 29, lower.tail = F)
```

```
## [1] 0.00638547
```

```
print("We would reject the one-way test.")
```

```
## [1] "We would reject the one-way test."
```

```
y <- mtcars$mpg
x <- mtcars$disp
mse <- c()

mse_fn <- function(emp_vec){
  for (i in 1:6){
    emp_vec[i] <- (sum((y - lm(y ~ poly(x, i, raw = T))$fitted.values)^2)) / nrow(mtcars)
  }
  return(emp_vec)
}

mse <- mse_fn(mse)

plot(mse)
```



The more power/predictors, the less mse in general.

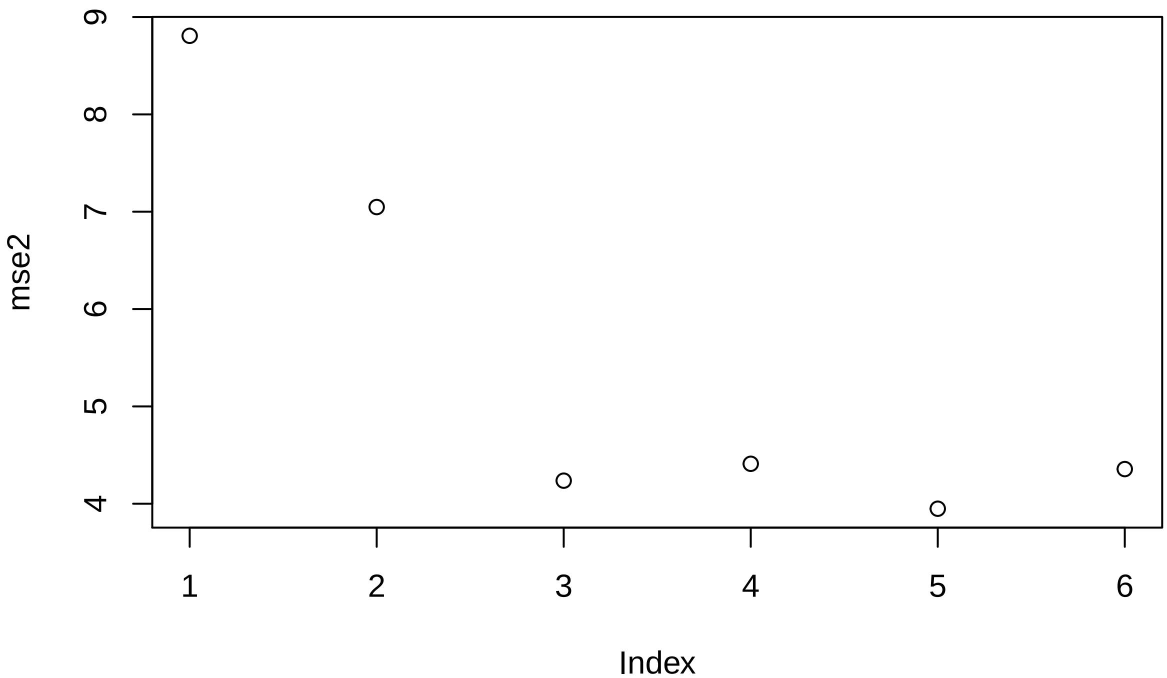
Q. And, holdout random sampling should be without replacement. Right? ==> yes

Q. So for hold out, we get beta hats from training and apply these beta hats into the test set X and get y hat and calculate MSE there. Right? ===> Yes.

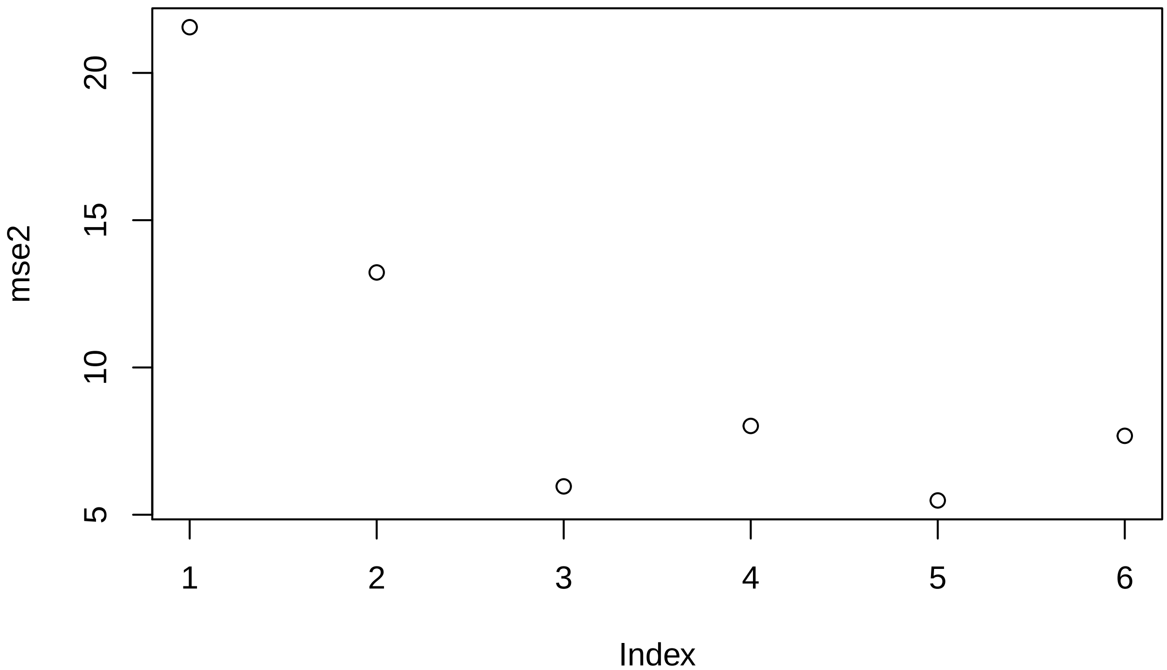
Q. And, when we add up test and training, it should be entire data. Right? Meaning I should get 80% of entire data as training, and the left over will be test?? ==> Yes cuz we dont model to cheat. Test set cannot be partial of trainig set.

```
for (j in 1:5){
  rand  <- c(sample(nrow(mtcars) , size = as.integer(nrow(mtcars) * 0.2), replace = F))

  testsize <- as.integer(nrow(mtcars) * 0.2)
  ytest <- y[rand]
  xtest <- x[rand]

  ytrain <- y[-(rand)]
  xtrain <- x[-(rand)]

  mse2 <- c()

  for (i in 1:6){
    coef <- lm(ytrain ~ poly(xtrain, i, raw = T))$coefficients #get beta hats from training sets.
    yhat_test <- cbind(1, poly(xtest, i, raw = T)) %*% coef #get yhat with beta hats from training and 
    mse2[i] <- sum((ytest - yhat_test)^2) / testsize
  }

  print(mse2)
  plot(mse2)
}
```

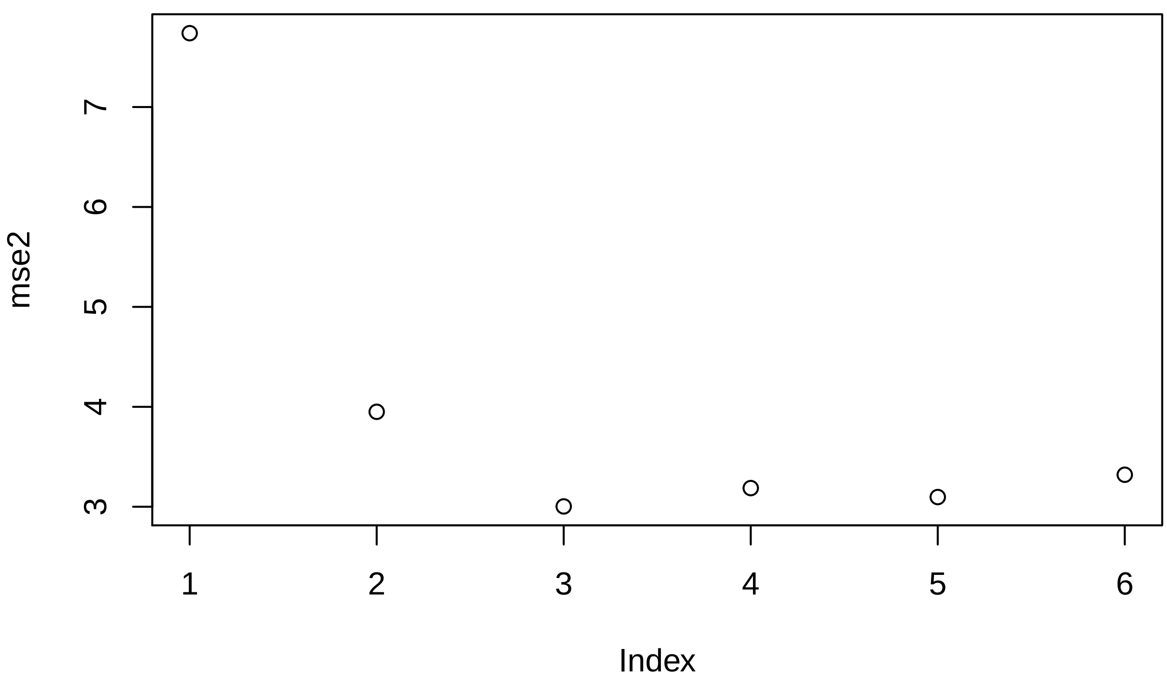
```
## [1] 8.806870 7.047812 4.236967 4.410343 3.948822 4.356148
```



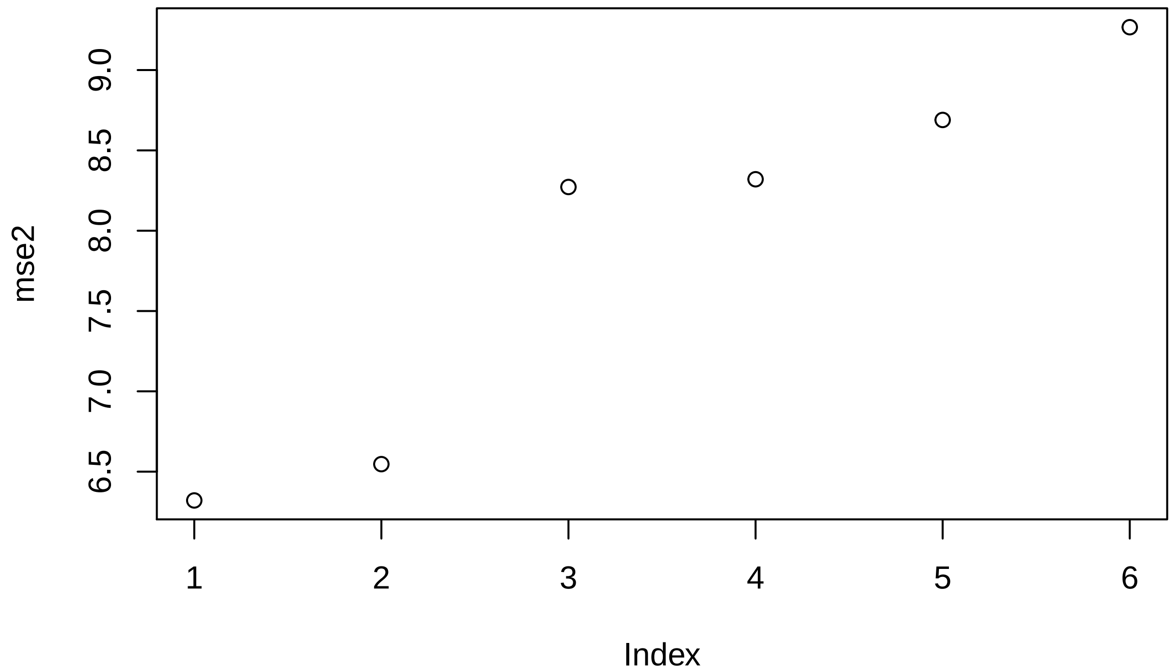

```
## [1] 21.552001 13.225544  5.962679  8.014849  5.485204  7.679725
```



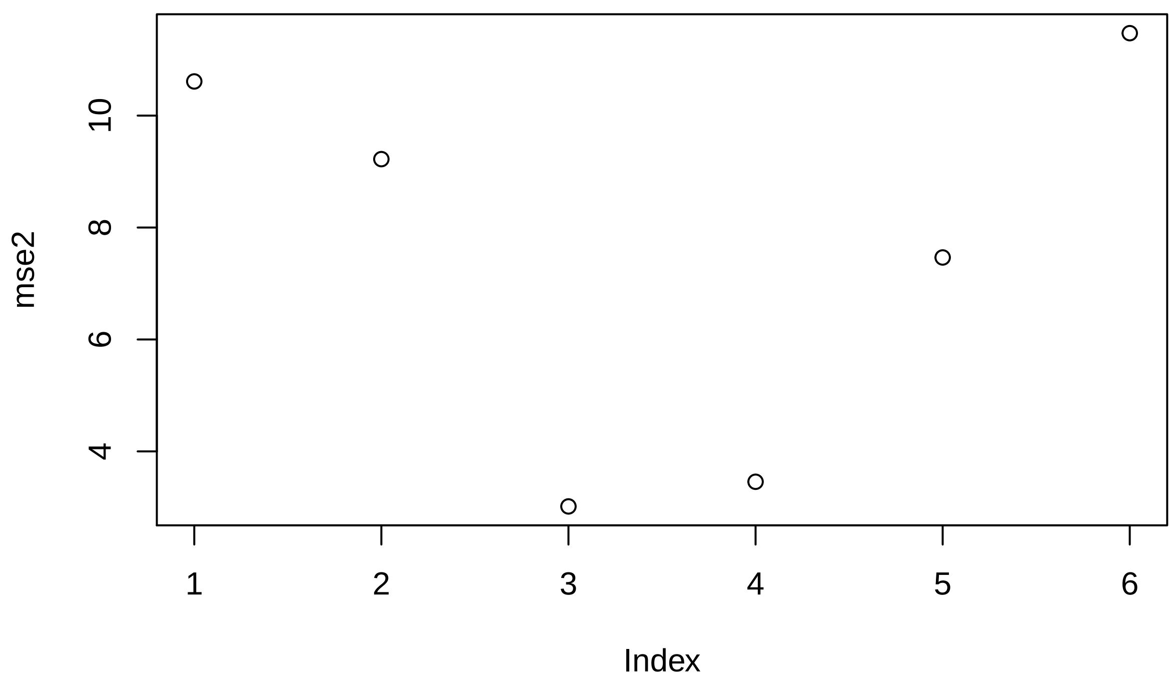

```
## [1] 7.739102 3.950187 3.003422 3.186763 3.097149 3.320194
```




```
## [1] 6.320833 6.546851 8.272179 8.320337 8.689484 9.266805
```




```
## [1] 10.610230  9.222065  3.017085  3.457284  7.465131 11.472516
```



Q. So, fold can have different numbers as they are randomly picked?? ==> Right.

Q. So, can I say cross validation MSE is reasonable, as they are randomly picked and get the average of each fold of MSEs? ==> Yes. (holdout and cross validation are good. Bootstrap is not really common, meaning not really good.)

Q. How to assign different names for each looping cv several times?.... ==> assign them as lists.

Q. For number 5, so, by the definition from https://en.wikipedia.org/wiki/Cross-validation_(statistics), I don't think n-fold cross validation is the same as leave one out cross validation. n-fold cv has n-folds, and thus, there is only 1 element for each fold and we compute n times. However, leave one out cv has 1 observation as a testing set, and leave n-1 as a training set but those n-1 does not have to be divided into n-1 groups. So, n-1 can be divided into 2 groups, 3 groups, etc. ???? ===> No. they are the same. Wikipedia did not say it but when leave one out cross validation, other rest should have 1 for each fold.

```
fold <- createFolds(mtcars$disp)
fold
```

```
## $Fold01
## [1]  3  5 10 24
##
## $Fold02
## [1] 17 22 27
##
## $Fold03
## [1]  7 14 20
##
## $Fold04
## [1]  6 18 26 29
##
## $Fold05
## [1]  9 15
##
## $Fold06
## [1] 11 16
##
## $Fold07
## [1] 13 19 21
##
## $Fold08
## [1]  2 12 25 30
##
## $Fold09
## [1]  8 28 31
##
## $Fold10
## [1]  1  4 23 32
```

```
for (k in 1:3){
  cv <- matrix(0, 6, 10)
  fold <- createFolds(mtcars$disp)
  for (i in 1:10){
    ytest <- y[fold[[i]]] #Assign test on the fold
    xtest <- x[fold[[i]]]

    ytrain <- y[-(fold[[i]])] #Assign train on all observations except the ones in the fold
    xtrain <- x[-(fold[[i]])]

    for (j in 1:6){
      coef2 <- lm(ytrain ~ poly(xtrain, j, raw = T))$coefficients #get beta hats from training sets.
      yhat_test2 <- cbind(1, poly(xtest, j, raw = T)) %*% coef2 #get yhat with beta hats from training
      cv[j,i] <- sum((ytest - yhat_test2)^2) / length(ytest)
    }
  }
  print(cv)
}
```

```
##           [,1]     [,2]      [,3]     [,4]     [,5]     [,6]     [,7]
## [1,] 21.502298 13.55437 10.023155 2.084125 7.342543 6.649979 2.366911
## [2,] 11.328696 15.88113 12.609010 1.116814 4.345578 9.650645 1.089579
## [3,]  3.924389 10.09592  5.443970 5.372020 4.857411 3.522418 4.959331
## [4,]  3.947696 10.40392  5.458696 5.625817 4.885671 3.541559 5.135622
## [5,]  3.150453 10.02968 16.216497 5.361999 2.632756 3.295990 4.471313
## [6,]  3.587025 10.10576 17.462760 5.458907 3.726852 3.873860 4.683216
##          [,8]      [,9]     [,10]
## [1,] 7.670977 50.804224 2.1172852
## [2,] 6.892349 34.049111 0.4335074
## [3,] 3.881686 14.103979 3.5310115
## [4,] 4.077362 15.422091 3.5510476
## [5,] 1.902578  8.497730 4.5936129
## [6,] 2.217097  9.061066 4.7348782
##           [,1]      [,2]     [,3]      [,4]      [,5]     [,6]      [,7]
## [1,] 19.884546 18.106845 6.042862 21.241582 16.711482 5.945914 2.1209551
## [2,]  8.606172  7.926383 4.324356 19.354133 12.702997 9.907980 0.9645657
## [3,]  3.618587  3.469097 2.551448  6.053712  7.629742 5.266013 5.8681455
## [4,]  3.768515  3.469427 2.633993  6.327842 11.349675 5.348232 6.0764542
## [5,]  2.578725  6.580390 1.267396 10.559475  8.081594 4.991494 5.9368162
## [6,]  3.659892  7.279847 1.373708 10.604071  8.074771 4.997383 6.3313302
##            [,8]     [,9]     [,10]
## [1,]  0.5599166 5.895169 19.656879
## [2,] 11.8034551 4.981495 14.728957
## [3,]  4.8372709 4.276961  9.289342
## [4,]  5.0968753 4.693121 12.138803
## [5,] 12.0819645 2.831920  8.932148
## [6,] 27.6431618 2.808900  9.068566
##           [,1]      [,2]      [,3]      [,4]      [,5]      [,6]      [,7]
## [1,] 12.180128 4.4267132  5.449726 0.7246283 12.109235 24.114481 10.310837
## [2,]  4.638807 0.5702108 13.472327 0.7539829  5.691774 12.913576 12.448732
## [3,]  1.241309 0.9880024  6.530877 6.8884509  1.623071  4.338218  7.688937
## [4,]  1.358940 1.0698812  7.098599 6.8885207  1.690765  4.338325  7.826365
## [5,]  2.449048 0.2684121  4.784445 5.5251688  1.983069  5.621179  5.891564
## [6,]  2.446965 0.3748168  6.758127 5.4891847  1.938780  5.742943  6.199265
##           [,8]      [,9]     [,10]
## [1,] 21.237987 18.718264  3.186830
## [2,] 18.537649 14.266714  6.147953
## [3,]  2.744865  8.379066 11.431485
## [4,]  2.835219 12.003791 11.631290
## [5,]  6.468778 12.270887 10.507837
## [6,] 25.074642 13.900627 10.556143
```

```
cvmse <- matrix(0, 6, 10)
for (i in 1:10){
  ytest <- y[fold[[i]]] #Assign test on the fold
  xtest <- x[fold[[i]]]

  ytrain <- y[-(fold[[i]])] #Assign train on all observations except the ones in the fold
  xtrain <- x[-(fold[[i]])]

  for (j in 1:6){
    coef2 <- lm(ytrain ~ poly(xtrain, j, raw = T))$coefficients #get beta hats from training sets.
    yhat_test2 <- cbind(1, poly(xtest, j, raw = T)) %*% coef2 #get yhat with beta hats from training an
```

```r
    cvmse[j,i] <- sum((ytest - yhat_test2)^2) / length(ytest)
  }
}

print(cvmse)
```

```
##            [,1]      [,2]      [,3]      [,4]      [,5]      [,6]      [,7]
## [1,] 12.180128 4.4267132  5.449726 0.7246283 12.109235 24.114481 10.310837
## [2,]  4.638807 0.5702108 13.472327 0.7539829  5.691774 12.913576 12.448732
## [3,]  1.241309 0.9880024  6.530877 6.8884509  1.623071  4.338218  7.688937
## [4,]  1.358940 1.0698812  7.098599 6.8885207  1.690765  4.338325  7.826365
## [5,]  2.449048 0.2684121  4.784445 5.5251688  1.983069  5.621179  5.891564
## [6,]  2.446965 0.3748168  6.758127 5.4891847  1.938780  5.742943  6.199265
##            [,8]      [,9]     [,10]
## [1,] 21.237987 18.718264  3.186830
## [2,] 18.537649 14.266714  6.147953
## [3,]  2.744865  8.379066 11.431485
## [4,]  2.835219 12.003791 11.631290
## [5,]  6.468778 12.270887 10.507837
## [6,] 25.074642 13.900627 10.556143
```

```r
#Or, use predict function to do it for you.....

cvmse2 <- matrix(0, 6, 10)
newmtcar <- as.data.frame(cbind(mpg = mtcars$mpg, disp = mtcars$disp))

for (i in 1:10){
  ytest2 <- y[fold[[i]]] #Assign test on the fold
  xtest2 <- x[fold[[i]]]

  ytrain2 <- y[-(fold[[i]])] #Assign train on all observations except the ones in the fold
  xtrain2 <- x[-(fold[[i]])]

  for (j in 1:6){
    lms2 <- lm(mpg ~ poly(disp, j, raw = T), data = newmtcar[-(fold[[i]]), ]) #get beta hats from train
    yhat_test3 <- predict(lms2, newdata = newmtcar[fold[[i]],  ]) #get yhat with beta hats from training
    cvmse2[j,i] <- sum((ytest2 - yhat_test3)^2) / length(ytest2)
  }
}
print(cvmse2)
```

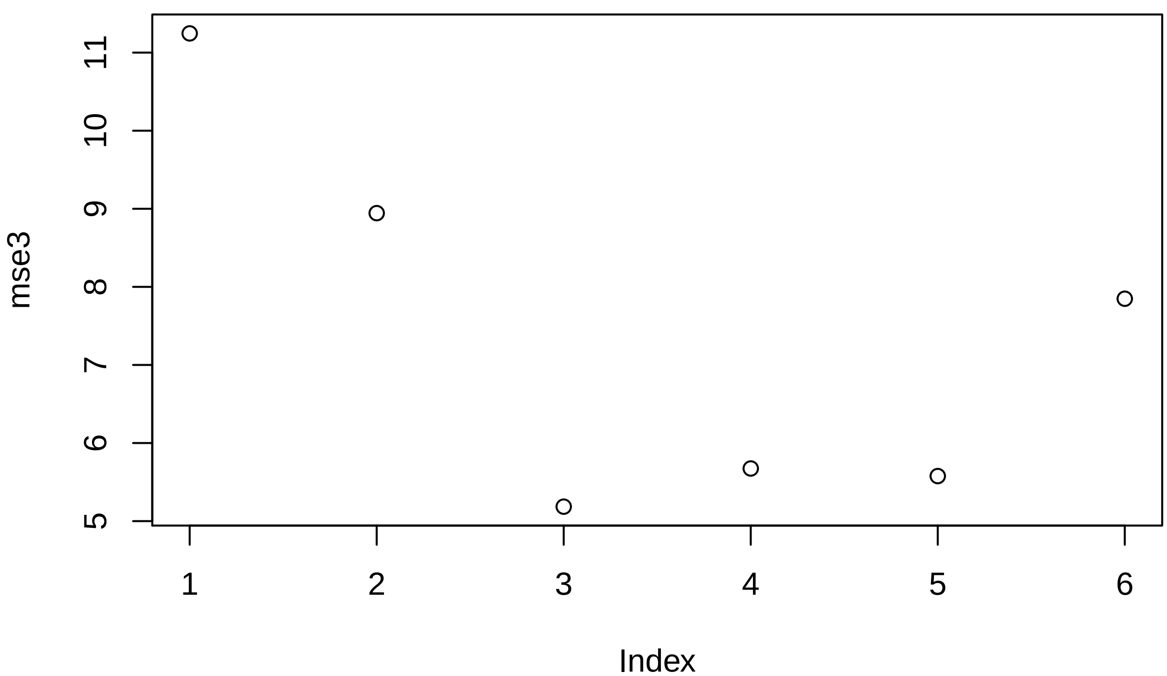
```
##            [,1]      [,2]      [,3]      [,4]      [,5]      [,6]      [,7]
## [1,] 12.180128 4.4267132  5.449726 0.7246283 12.109235 24.114481 10.310837
## [2,]  4.638807 0.5702108 13.472327 0.7539829  5.691774 12.913576 12.448732
## [3,]  1.241309 0.9880024  6.530877 6.8884509  1.623071  4.338218  7.688937
## [4,]  1.358940 1.0698812  7.098599 6.8885207  1.690765  4.338325  7.826365
## [5,]  2.449048 0.2684121  4.784445 5.5251688  1.983069  5.621179  5.891564
## [6,]  2.446965 0.3748168  6.758127 5.4891847  1.938780  5.742943  6.199265
##            [,8]      [,9]     [,10]
## [1,] 21.237987 18.718264  3.186830
## [2,] 18.537649 14.266714  6.147953
## [3,]  2.744865  8.379066 11.431485
## [4,]  2.835219 12.003791 11.631290
## [5,]  6.468778 12.270887 10.507837
```

```
## [6,] 25.074642 13.900627 10.556143
```

```r
identical(cvmse, cvmse2)
```

```
## [1] TRUE
```

```r
#These codes below will not work because I do not use precict function in one data frame. If I do not u

#https://stackoverflow.com/questions/31879271/how-can-i-add-a-hashtag-sign-to-many-lines-in-r-command

#for (i in 1:10){
#  ytest2 <- y[fold[[i]]] #Assign test on the fold
#  xtest2 <- x[fold[[i]]]

#  ytrain2 <- y[-(fold[[i]])] #Assign train on all observations except the ones in the fold
#   xtrain2 <- x[-(fold[[i]])]
#
#   for (j in 1:6){
#     fit <-lm(ytrain2 ~ poly(xtrain2, j, raw = T), data = as.data.frame(cbind(ytrain2, xtrain2)))
#     predict(fit , newdata = cbind(1, poly(xtest2, j, raw = T)))
#     cvmse2[j,i] <- sum((ytest2 - yhat_test3)^2) / length(ytest2)
#   }
# }

mse3 <- rowMeans(cvmse)

plot(mse3)
```



CVMSE does not stay the same, as they are randomly picked… .

So, we said leave one out cv because we leave only 1 observation as the testing/validation set and all others as the training set.

In general, Leave p out cross-validation requires training and validating the model C_p^n times, where n is the number of observations in the original sample. There is really no fold concept in leave p out, but leave n out can be equal to n fold cv as we have n groups and have 1 observation for each fold.

Q, When you said "test on the model that is not in the sample," are you talking about the number itself or the iteration? So, lets say original data is 1 1 2 3 4 5 and when we get sample 1 (first one) 2 2 2 3 5, then do our test set includes the second one but does not include the first one. Right? ==> Right. Order matters, not the number itself. Also, when we have 2 2 2 for training, we use all of them although it is repetion since we sample with replacement. Right? ==> yes!

```
bootmse <- matrix(0, 6, 400)

sd <- c()
for (i in 1:400){
  trainrow <- sample(nrow(mtcars), nrow(mtcars), replace = T) #sample the row number for training.
  ytrain3 <- y[trainrow]
  xtrain3 <- x[trainrow]

  ytest3 <- y[-(trainrow)]
  xtest3 <- x[-(trainrow)]
  for (j in 1:6){
    coef3 <- lm(ytrain3 ~ poly(xtrain3, j, raw = T))$coefficients #get beta hats from training sets.
    yhat_test3 <- cbind(1, poly(xtest3, j, raw = T)) %*% coef3 #get yhat with beta hats from training a
    bootmse[j,i] <- sum((ytest3 - yhat_test3)^2) / length(xtest3)
  }
}
print(bootmse)
```

```
##           [,1]       [,2]      [,3]      [,4]      [,5]      [,6]
## [1,] 22.061052   9.675936 11.348389 13.971967 12.204276  6.507804
## [2,] 15.323618  30.074190  9.720899 10.209569  9.268578  7.201565
## [3,]  8.885566  18.833766  5.342195  5.849273  5.101093  3.641401
## [4,]  8.982156 155.488191  5.322446  5.867028  5.061890  3.943989
## [5,]  7.277555 341.923319  6.687805  5.383381  5.612236  7.915487
## [6,]  7.398866  91.794788 20.770506  5.045753 13.636480 11.588735
##            [,7]      [,8]      [,9]     [,10]     [,11]     [,12]
## [1,]   2.454001 10.191379 11.358867 15.447946 10.808976 15.115690
## [2,]  29.676763  7.329827  5.739296 10.191321  9.231082  8.833404
## [3,]   7.849467  4.388817  4.337770  5.978359  4.222998  4.305203
## [4,]  18.050804  5.906793  4.359050  6.104798  4.321362 15.270928
## [5,]   4.982361  5.152581  3.296925  8.014713 13.005971 15.537225
## [6,] 109.030707  9.447026  4.001078 11.605186 12.985683 35.089194
##           [,13]     [,14]     [,15]     [,16]     [,17]      [,18]
## [1,]  10.684626 19.529821 11.300665 18.039699 12.875749  14.337423
## [2,]  12.766664 14.018362  8.482566 13.932012  7.617037  12.388737
## [3,]   5.159947  7.379558  4.015818  7.550341  4.647218   6.082730
## [4,]   4.935652 14.240991  4.165758  7.679880  5.345001  13.303783
## [5,]   4.194980 12.043695  3.137561  5.354689  4.377889   4.564543
## [6,] 129.294045 23.563379 13.717248  5.365954  4.501377 221.233080
##          [,19]     [,20]     [,21]     [,22]     [,23]      [,24]
## [1,] 17.401995 11.828188 13.006104 10.853381  8.688040   8.793158
## [2,] 14.935623  9.776974 19.468908 10.214770  9.522490  44.125412
## [3,]  9.409739  5.349423  6.517791  8.466495  7.382548  30.228018
## [4,] 13.476752  6.895574 21.432296  8.729652  7.404956 186.501530
## [5,] 14.630792  6.560532 15.388338  8.432194  8.614807 116.116696
## [6,] 13.513108  6.607089 93.953932  8.489613 10.909780 600.328162
##          [,25]     [,26]     [,27]     [,28]    [,29]     [,30]    [,31]
## [1,] 13.144907 22.821452 14.576224 16.085422 8.297724 10.410327 5.772798
## [2,]  9.406218 14.722293  8.888291 13.622665 8.767289  7.175251 4.921687
```

```
## [3,]  5.900342  6.307970  3.970212  8.181632 5.329385  7.246015 6.681623
## [4,]  5.902593  6.120140  3.986550  9.707979 8.639913  9.064673 6.587509
## [5,]  5.998782  8.510205  7.369389  9.645525 5.047696  7.504969 6.457262
## [6,] 10.575941  8.649776  7.370740  9.319717 5.911432 10.423298 6.526221
##           [,32]     [,33]    [,34]     [,35]     [,36]     [,37]    [,38]
## [1,]  4.801875 22.233946 8.569925 11.575210  9.542118 11.408036 7.224547
## [2,] 26.675755 16.147871 7.796876  7.067143  7.243052  7.681736 7.063104
## [3,] 14.331731  6.145081 3.884755  7.476884  3.032069  3.624076 4.879164
## [4,] 11.063957  6.294011 4.739496  7.370528  3.031840  3.668486 5.280643
## [5,] 11.358412 20.589442 5.188536  7.927531  7.498461  6.535582 4.475031
## [6,] 46.150341 94.578277 5.220278  7.757168 19.743029  6.281184 4.510706
##           [,39]     [,40]     [,41]     [,42]     [,43]     [,44]
## [1,]   8.746101 12.469458 11.339403 18.574862 20.281580  9.734499
## [2,]  49.284801  8.222524  9.989257 12.916123 14.329424  8.404603
## [3,]  28.909336  5.845965  8.069987  7.898409  8.024404  6.901721
## [4,] 217.795333  6.170789  8.069977 10.766092  9.029349  8.349634
## [5,] 148.712669  5.710900  7.919101  8.873377  9.356721  5.367069
## [6,] 350.588800  6.060014  9.250944  7.360936  9.695266 10.489670
##           [,45]     [,46]     [,47]       [,48]     [,49]     [,50]
## [1,] 10.456173 11.597020 16.053565    13.25687 11.519375 12.388637
## [2,]  6.923409 10.292993  9.552603    28.49645  6.706295  6.679001
## [3,]  6.482733  5.985250  4.759717    27.43284  6.084994  3.581312
## [4,]  6.389632  6.637833  7.449663   480.87415  6.725330  4.673312
## [5,]  6.196379  6.043861  7.626601 11005.60207  6.808590  8.680795
## [6,]  6.709467  6.590024  8.753854  4719.73002 10.086412  8.894665
##           [,51]     [,52]     [,53]     [,54]     [,55]    [,56]     [,57]
## [1,] 15.522245 10.153266 10.476494 12.321244 18.951016 12.70441 13.999739
## [2,] 10.237604  9.423685  7.734151  8.932674 14.116848 11.23902 10.372778
## [3,]  7.573503  6.476237  8.546323  5.151797  8.687166  7.19336  5.820226
## [4,]  7.527253  6.601638  9.521474  5.357143  9.955578 10.27017  6.952566
## [5,]  6.360138  6.083253  8.156442  6.542436  9.605722 11.33479  6.325023
## [6,] 64.825361 15.443741  8.140332  7.217412 10.303188 12.93746  6.930155
##           [,58]     [,59]     [,60]    [,61]      [,62]     [,63]    [,64]
## [1,] 16.090179 20.831033  8.765487 3.757954   9.776713 12.059162 9.602034
## [2,] 11.227247 12.270849 10.762057 5.468129  13.445008 10.249355 8.561458
## [3,]  4.639680  7.141581  7.082917 7.747277   3.931144  5.348948 5.807777
## [4,]  4.708440 11.905831  7.239953 8.980840   8.156888  5.545910 7.194056
## [5,]  8.464180  7.912253  8.405262 7.891786   9.772765  5.326323 6.729282
## [6,]  9.389319  6.918496  9.265250 7.926989 272.748621  5.430871 6.549733
##         [,65]     [,66]    [,67]     [,68]    [,69]     [,70]     [,71]
## [1,] 5.575889 16.695418 7.509531 14.278137 7.608232 16.794111 15.416243
## [2,] 4.910631  9.835912 2.998549  8.854197 5.421595 13.135596 13.133895
## [3,] 7.846108  5.835157 5.246555  5.567701 4.264003  7.576998  7.793165
## [4,] 7.938981  5.863561 5.536577  5.586999 4.473986  7.607849  7.792615
## [5,] 7.786971  4.670031 6.764680  5.186026 3.086318  6.115576  6.558284
## [6,] 7.968810 31.877060 6.969341  5.147089 3.571291  6.128100  6.580077
##           [,72]    [,73]     [,74]     [,75]     [,76]     [,77]     [,78]
## [1,] 10.606213 5.196153  24.17102 11.664469 12.241046  16.74401 15.223724
## [2,]  7.295394 4.646407  35.84850 10.824160 10.283620  51.16614  9.565933
## [3,]  3.439578 6.695154  36.23141  6.725282  5.766609  38.64736  5.044783
## [4,]  4.616469 6.703038 133.72813  6.816468  7.722016  61.39293  6.395940
## [5,]  3.004932 5.306264  78.32891  6.226502  6.296734  76.90470  6.924239
## [6,]  3.997630 6.001261 483.13034  6.482475  6.309347 607.88440  7.108203
##            [,79]     [,80]     [,81]    [,82]    [,83]     [,84]     [,85]
```

```
## [1,]   9.343880 15.287520 11.023187 9.295058 8.665729 12.518540 10.885304
## [2,]  10.138510 12.672045  8.608629 9.394459 7.829866  9.125752  6.599515
## [3,]   4.133792  6.498352  7.664375 8.496118 3.978855  5.452576  4.503008
## [4,]  11.852206 21.650192 10.156618 8.534359 4.345976  5.703499  4.896533
## [5,]   3.687327 11.002879  9.050725 7.194381 5.980310  4.877483  4.972551
## [6,] 138.255521 25.693234  9.106133 7.143653 7.879248  8.813974  5.165008
##          [,86]     [,87]     [,88]     [,89]     [,90]     [,91]    [,92]
## [1,] 15.850780 12.065437 11.612021 25.669283  7.570541 15.420331 7.606464
## [2,] 12.298528  8.551716 11.005514 19.155878 11.908040 11.767646 7.244273
## [3,]  7.794408  4.027932  7.266340  8.198813  7.755279  6.091279 5.857346
## [4,]  7.477958  4.010326 10.521069  9.556133  8.719904  6.069978 5.963407
## [5,]  6.418111  5.827345  7.150222 38.599257  8.236233  4.042526 8.749445
## [6,]  9.288885  6.159125  9.189718 70.609593 17.044268  4.339016 9.340026
##          [,93]     [,94]     [,95]     [,96]    [,97]     [,98]     [,99]
## [1,] 11.647543 13.438296 11.914485 16.230234 7.409303 12.314822 13.630262
## [2,]  9.967868  8.550060  8.661334 12.628222 6.698548  6.538138 10.179811
## [3,]  7.705453  5.843196  7.175210  6.932160 9.977246  5.038188  6.401895
## [4,]  7.715682  5.845219  7.101517  7.295184 9.974257  4.926858  6.613292
## [5,]  6.873066  7.440218  6.965593  7.901930 9.317881  4.446965  6.242406
## [6,]  9.080322  8.039058  6.994149 13.727588 9.388738  6.814916  6.313934
##         [,100]    [,101]    [,102]    [,103]    [,104]    [,105]    [,106]
## [1,] 11.101632 13.345362 10.678478 10.893239 12.219592 11.669450 13.229760
## [2,]  5.858790 10.051861 17.777089 12.663061  9.738781  7.734166  8.558184
## [3,]  5.826358  5.956597  9.644170  7.474735  4.945066  5.357448  6.314203
## [4,]  6.460233  6.314216  8.521111  7.864421  7.285184  7.305764  6.271997
## [5,]  5.601656  6.953003  6.758913  8.181264  6.427016  6.583708  6.981270
## [6,]  6.264559  7.537056 21.737940  8.192324  9.684695 10.161789  7.017874
##         [,107]    [,108]   [,109]     [,110]    [,111]    [,112]    [,113]
## [1,] 16.545180 14.627258 7.243276   7.469095 12.627182  8.715114  8.325425
## [2,] 10.988431  9.571003 6.470967  51.392238 23.570674  7.318503  8.500525
## [3,]  5.757466  5.714383 8.987980  16.671269 10.979815  6.094516  4.541491
## [4,] 19.175459  9.269181 9.894530 255.882303 18.281374  7.818172  5.786399
## [5,] 11.257669  7.582708 7.756783 182.771604  6.710639  7.998482  4.659230
## [6,] 11.233500 10.184228 8.642900  17.426208 83.980289 10.330055 11.511460
##         [,114]    [,115]    [,116]    [,117]    [,118]    [,119]    [,120]
## [1,]  5.336375 12.210058 10.488643  9.715333 12.635152 11.401024 12.845628
## [2,]  6.568419 11.987227  8.593297  9.254608  8.747220  6.980358 11.687341
## [3,]  6.154623  8.616857  5.425139  5.545431  9.923283  5.959132  6.337990
## [4,]  6.152607  8.601330  5.386007  8.561172 10.194112  5.832570  6.207005
## [5,]  7.366230  7.569382  6.085971  8.876215 12.174656  5.858992  5.812311
## [6,] 17.419305 10.844630  6.215968 10.457985 11.034625  5.853362  6.458368
##         [,121]    [,122]    [,123]    [,124]    [,125]   [,126]    [,127]
## [1,] 12.502481 13.013751 16.437475 12.789990 13.946207 6.870610 12.190228
## [2,]  7.760528 10.507864 11.431646  9.269810 13.135480 5.794221  8.356638
## [3,]  4.611804  6.571789  6.232376  4.038049  8.157892 3.605558  6.242773
## [4,]  4.600495 10.005482  6.681033  4.041207  8.592891 4.000867  6.260784
## [5,]  4.284414  9.104390  6.548768  3.930870  6.972439 5.917135  7.382714
## [6,]  4.437497  9.168721  5.794815 13.540438  6.962227 5.912389  8.339580
##         [,128]    [,129]    [,130]    [,131]    [,132]   [,133]    [,134]
## [1,] 13.192987 11.704972 18.031733  9.503194 15.397972 6.609550 20.395755
## [2,]  7.466234  9.230379 13.299524 10.026979 11.743274 3.338345 15.285012
## [3,]  3.352261  6.095789  9.404087  7.856633  6.075860 3.100047 10.518700
## [4,]  3.347939  7.145577 13.212817  8.985045  6.511648 3.274938 10.644772
## [5,]  5.376445  6.409125 10.314108  8.810975  3.908780 3.514960  9.545759
```

```
## [6,]  6.317803  9.044771 10.719809  9.277458 28.334141 3.495061 10.182667
##         [,135]    [,136]    [,137]    [,138]    [,139]    [,140]   [,141]
## [1,] 17.558470 10.739951 11.294147 16.262977 12.941862 20.194330 6.520739
## [2,] 10.771226  8.321017  7.963079 13.450201  9.806626 15.086947 8.456019
## [3,]  6.320261  6.433215  4.948283  7.019740  4.332447  7.380592 6.679826
## [4,]  6.181188  6.547633  5.044171 10.738144  4.463365  7.953982 6.700651
## [5,]  5.408943  5.600611  5.552529  8.460321  4.152265  6.006434 7.147856
## [6,]  5.252928 18.726648 58.091152  9.031323  5.272826  6.100503 8.168974
##         [,142]    [,143]   [,144]    [,145]   [,146]   [,147]   [,148]
## [1,] 13.401779  7.109777 9.761720  7.002856 7.644879 9.164620 6.775830
## [2,]  9.615378 12.523765 6.639387  6.117509 8.244165 7.892801 4.494324
## [3,]  6.399362  7.382606 5.511099  2.503792 6.858839 5.592020 7.724413
## [4,]  6.626910  9.868514 6.611769  2.577669 6.986730 7.744620 8.318729
## [5,]  7.325644 11.021637 5.585262  8.700696 5.958567 6.405897 7.051513
## [6,]  8.155213 11.027765 6.107954 11.726266 6.029649 6.467239 8.460477
##         [,149]    [,150]    [,151]    [,152]    [,153]    [,154]    [,155]
## [1,] 13.469361 16.197120 10.819417 10.918027 22.388683 10.733317 10.657257
## [2,]  7.497114 12.628200 11.821979  8.129653 14.932486  9.161751  6.570095
## [3,]  5.679502  8.448628  9.246307  4.620049  7.886107  4.356839  4.266761
## [4,]  9.567701 10.125759  9.239310  5.282301 11.171916  4.466354  4.268043
## [5,]  7.040753  8.348227  7.715298  3.763089 10.038542  7.326512  3.525306
## [6,]  7.792572  9.026919  7.772356  3.530279 10.035104 10.051989  4.438058
##         [,156]    [,157]   [,158]    [,159]     [,160]    [,161]    [,162]
## [1,]  5.200721 13.223870 5.864313 14.461869  11.262271 14.599783 12.514396
## [2,]  5.772563 11.896715 3.068703 10.702198  16.865524 11.849265  8.792765
## [3,]  8.673372  7.387709 4.910914  5.128408   6.508360  5.444543  5.076421
## [4,] 12.045125 10.492266 4.893470  5.200723  10.878243  5.003962  8.849310
## [5,]  9.803176  8.972690 4.689110  7.501425   6.001572  3.961413  6.837537
## [6,] 13.821978 11.974155 4.661400 10.694448 427.736858  3.937891  6.798878
##         [,163]    [,164]    [,165]    [,166]    [,167]    [,168]    [,169]
## [1,] 12.863256  4.157531  9.597575  8.106446 13.166975 11.210153 14.163951
## [2,] 11.065459  6.309200  7.951928 19.096550  8.881919  7.657912 10.712562
## [3,]  7.131643 11.023517  2.096988  6.412711  5.511876  4.705494  6.701013
## [4,]  7.575029 11.018689  2.156998 13.746855  9.214066  4.725301  6.567532
## [5,]  6.392785  9.480086  5.662640  8.408420 12.256860  3.846586  5.376265
## [6,]  6.439065  9.824113 11.289197 74.841198 11.608305 14.344051 27.982582
##         [,170]   [,171]     [,172]    [,173]    [,174]    [,175]   [,176]
## [1,] 12.397513 9.294244   7.300602 18.753328 15.957300 14.212985 9.907617
## [2,]  8.571346 6.485550  13.247831 14.606913 11.119356 11.021339 5.958032
## [3,]  4.755656 5.115181   5.229330  6.589629  6.340356  6.038332 5.880039
## [4,]  5.337142 5.117642  12.115794  6.923732  6.583767  5.895982 6.331168
## [5,]  4.677005 4.707403   4.891280 24.956620  6.731869  6.454868 7.606640
## [6,]  4.765339 5.426590 151.718568 29.462642 11.785523 10.805646 7.744214
##         [,177]    [,178]    [,179]    [,180]    [,181]    [,182]    [,183]
## [1,] 15.625210 20.214546  14.38198 21.608662 19.302563 15.990147 12.223251
## [2,] 16.691363 13.113295  37.49313 14.625083 14.907151 11.558746  9.237968
## [3,]  7.548641  6.685515  35.53282  8.100545 10.172050  5.115966  4.465832
## [4,] 14.727026  7.090921 211.05574 10.140165 10.174956  9.694935  4.736197
## [5,] 18.443085  5.518015 295.12766  9.100912  8.361670  7.688323  3.545606
## [6,]  9.683276  8.061450 586.62841 11.674466  9.262032  8.267322  3.624830
##        [,184]    [,185]   [,186]    [,187]    [,188]    [,189]    [,190]
## [1,] 6.954511  6.662124 4.371182 11.533092  8.835474 14.176463  9.481505
## [2,] 4.107300  5.141074 2.670855  8.634566 12.548665  9.205813 30.517477
## [3,] 3.191506 10.789059 4.337229  4.220292  9.406058  5.453718 12.402995
```

```
## [4,] 3.139489 10.554684 4.542122  4.618868  9.391062  5.453150 12.085046
## [5,] 3.740218  6.452083 3.580964  4.587950  8.561020  5.225693  9.013551
## [6,] 5.440552  9.787678 3.619429  4.713268  8.450178  5.488318 60.323604
##         [,191]    [,192]    [,193]    [,194]    [,195]    [,196]    [,197]
## [1,] 15.149190 13.495139 14.284399  7.194396  12.10639 12.568788 10.690160
## [2,] 10.347752  8.886574 10.532295  6.619237  32.02779 11.131985  9.816506
## [3,]  5.965787  6.741556  6.964330  5.789178  11.67495  8.934498  6.676888
## [4,]  6.007155  6.771013 12.715669  6.906897 473.76622  9.102043  6.796587
## [5,]  5.394757  6.985945  9.629379  9.336302 423.16964 10.663519  6.180535
## [6,]  8.007672  7.669685  8.940478 10.700695 102.58070 14.709335  5.915190
##         [,198]   [,199]    [,200]    [,201]    [,202]    [,203]    [,204]
## [1,]  8.395226 9.066955  6.129491 11.147173 18.513679 12.439156  7.673446
## [2,] 31.609675 8.309809  5.943850  7.807032 14.332007  8.436511 10.393344
## [3,]  6.386179 4.660509  5.761469  5.771993  6.750047  4.322886  5.459532
## [4,]  8.977301 4.636271  6.365221 11.299411  7.621031  5.291169  6.611831
## [5,]  5.556209 5.390403  6.171456 10.826182  7.893748  4.279575  8.270848
## [6,] 85.197425 9.607700 11.708533 87.199524 33.918153  4.684037  8.751165
##         [,205]    [,206]    [,207]   [,208]    [,209]    [,210]    [,211]
## [1,]  9.276999 13.806883 13.529759 9.470128  5.887874 16.148656 12.534900
## [2,] 13.810980 10.515435  9.277022 8.880179 12.200710 15.281683 10.784283
## [3,]  6.506648  5.973248  4.889743 7.608353 10.355056  9.825420  4.844341
## [4,]  5.644177  5.972296  6.763360 7.965407 10.911911 11.026095  4.724738
## [5,]  6.179726  6.532583  6.219621 6.831119  8.784728  9.252506  5.962593
## [6,] 68.752704 10.968366  6.267844 6.992545  9.035027  9.254348  5.802437
##         [,212]   [,213]    [,214]    [,215]    [,216]    [,217]    [,218]
## [1,] 15.494488 14.14875 10.887783 11.137095 11.565716 16.076019  9.470341
## [2,] 10.095602 15.96531 12.132848  9.567613  7.082498 12.794245 10.346943
## [3,]  4.510487 11.44906  6.705377  7.648775  5.181180  9.218970  6.150647
## [4,]  4.852058 11.45280 11.772055  8.800314  6.221569  9.183461  7.500282
## [5,]  3.461238 11.20581  9.575681  8.698677  4.586937 13.013418  9.040184
## [6,]  3.456341 12.91435 14.136684  9.170067 16.436266 14.541409 11.935209
##         [,219]    [,220]   [,221]    [,222]    [,223]   [,224]    [,225]
## [1,] 13.910266 17.436764 9.766215 13.534157 11.943796 7.505998 18.545316
## [2,]  9.984924 14.797808 6.848697  7.300985  8.672586 6.229928 15.004992
## [3,]  4.995302  8.824150 3.787830  7.396277  3.854787 2.595958  6.737373
## [4,]  8.708587  8.864990 3.762337  6.946498  4.476290 2.714356 10.000994
## [5,]  6.296207  7.956548 4.721326 11.820888  4.023202 3.166420  6.294966
## [6,]  9.376732  7.828646 4.866337 11.306632  4.233645 6.137728  6.268357
##         [,226]    [,227]    [,228]   [,229]    [,230]    [,231]    [,232]
## [1,] 10.532698 15.592923 15.025860 24.89052 12.914986 10.437571 15.532441
## [2,]  9.285913  8.696653 10.276945 26.86150  9.955354  7.272406 14.756261
## [3,]  4.471539  5.134821 10.654723 31.03734  5.334388  3.945281  7.995880
## [4,]  4.915079  7.179219 11.154762 19.29832  5.733583  3.906624  8.167738
## [5,]  3.737875  4.692268  9.271202 14.37078  4.529106  3.090898  7.457640
## [6,] 18.112663 21.210984  9.566865 63.57279  4.616625  5.589654  8.459323
##         [,233]    [,234]   [,235]    [,236]    [,237]   [,238]    [,239]
## [1,] 18.110751 17.835285 5.261689  9.757318 11.222163 9.780388 11.709412
## [2,] 13.729197 11.983612 4.133304  6.733120  7.482999 5.673656  7.045843
## [3,]  7.580860  7.190465 2.782555  5.962169  3.347331 4.432012  4.111286
## [4,]  7.980814  9.205044 2.760833  5.796846  3.620137 4.933097  5.091593
## [5,]  5.607288  8.010983 2.825299  5.904480  3.879790 3.710279  6.296988
## [6,]  6.015849  7.674764 3.660389 13.741526  4.392253 3.891484  6.354163
##         [,240]    [,241]    [,242]    [,243]    [,244]    [,245]   [,246]
## [1,] 14.203407 14.477226 12.361686  7.677317 13.439915  7.747554 7.558789
```

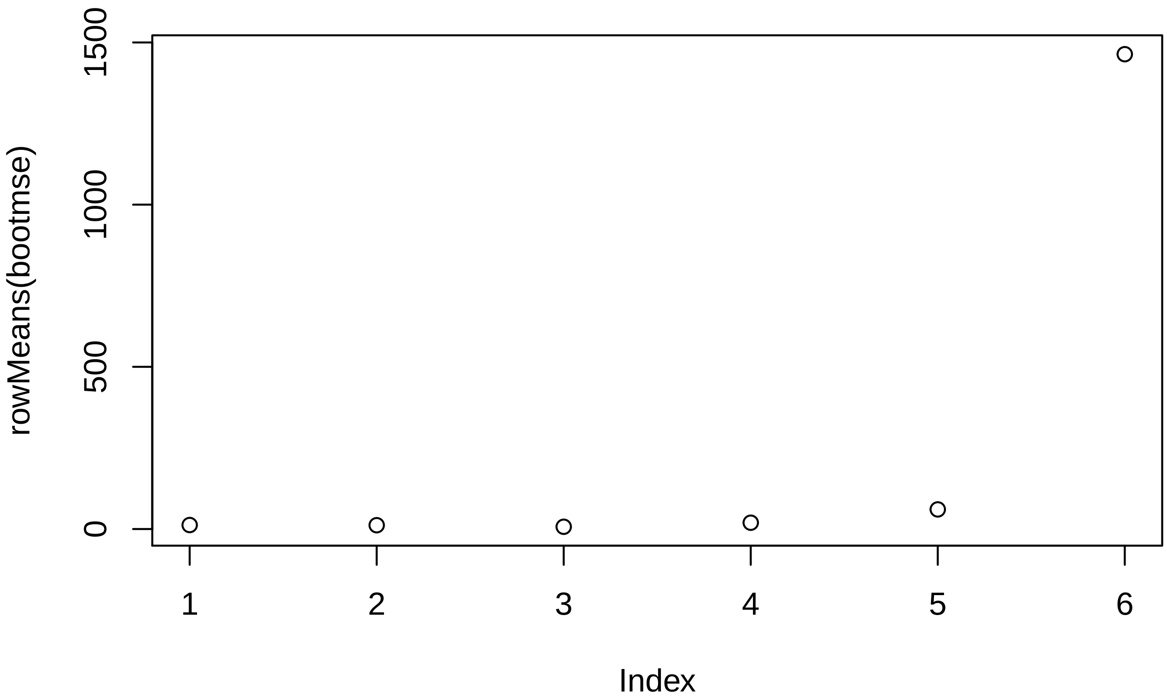
```
## [2,] 12.052691 10.719370  8.345934  8.043084  8.025998  6.124340 4.748203
## [3,]  7.381017  5.420318  5.160123  8.075465  5.339784  5.283665 2.347870
## [4,]  7.974884  6.844730  9.515662 12.669905 10.047284  5.221047 2.333323
## [5,]  6.703228  6.706661  7.459364 10.698056  5.842388  6.883918 6.663073
## [6,]  7.658195  6.386936  7.592840  9.116316 19.531208 10.598455 9.842479
##          [,247]   [,248]    [,249]    [,250]    [,251]    [,252]    [,253]
## [1,] 10.991913 7.387589 17.311178  10.85415 12.816748 14.775230 10.034097
## [2,]  8.098101 3.753363 20.139577  21.73955 12.952156 11.051902  8.058506
## [3,]  7.225568 6.768444  8.443964  11.97438  8.378885  7.384873  5.634659
## [4,]  7.468197 6.353975 11.660650 189.42449  9.371159  8.281980  5.897203
## [5,]  4.869639 5.558471 54.461955 236.42741  6.397053  6.684599  5.945422
## [6,]  9.378025 5.582139 63.252255 336.07652  9.670347  6.680005  9.136890
##          [,254]    [,255]    [,256]    [,257]    [,258]    [,259]    [,260]
## [1,]  9.550531 15.700214 13.675054  6.784281 13.202587  5.717368 11.248808
## [2,]  8.538005 10.679419 10.420994  9.836177 17.872577  7.470367 11.220036
## [3,]  5.375315  7.133857  6.283022  6.422829  5.976511  6.493448  8.268887
## [4,]  5.825879  7.162541  7.368162  6.295778 28.142397  6.532440  8.215251
## [5,]  4.693904  8.112349  6.583517  6.656861  9.732927  6.936342  7.164600
## [6,] 57.772323 28.363423  7.635638 14.208351  9.834617 14.094327  7.843295
##         [,261]   [,262]    [,263]    [,264]    [,265]   [,266]    [,267]
## [1,] 8.618345 8.708279 12.494380 15.120162 11.762819 6.721778 24.258087
## [2,] 4.661435 8.292578 20.331743 10.499696  7.816188 3.641034 17.952483
## [3,] 3.499116 4.063089  7.348715  4.856321  4.529725 3.055878  6.691019
## [4,] 3.531243 4.489605  6.308158  5.126473  4.438795 5.884852 11.280167
## [5,] 4.731627 6.951029  4.714251  5.850093  5.030206 6.065092  6.933810
## [6,] 5.435820 8.426058 51.679942  8.503906  5.101385 6.069117  8.925998
##         [,268]     [,269]    [,270]   [,271]        [,272]    [,273]
## [1,] 8.329825   12.29964  21.70193 6.664988 1.304611e+01 11.706474
## [2,] 7.271534   44.64130  18.15580 7.587181 4.852247e+01  9.199103
## [3,] 7.813481   14.49891  13.45056 6.232582 3.983131e+00  4.350988
## [4,] 7.693100  157.42019  10.11643 7.990088 7.856203e+02  4.436825
## [5,] 6.501228   62.61464  15.43167 6.784930 5.927783e+03  6.820101
## [6,] 6.652034 4412.68421 765.99531 6.376702 3.307395e+05 10.934784
##          [,274]    [,275]    [,276]    [,277]    [,278]    [,279]    [,280]
## [1,] 15.950398 10.660487 10.943134 12.197619 20.129739 14.032798 13.737020
## [2,] 12.023522  7.517881 12.372681  8.459273 14.408401 12.134072 11.346733
## [3,]  8.653414  6.109746  8.373326  6.724531  8.749184  8.601721  8.427007
## [4,]  9.883952  6.081543 10.150489  7.368039 11.272195  8.562462  8.721303
## [5,]  7.494846  4.408356  8.937218  7.143616  9.097138  9.134924  8.149434
## [6,]  8.067923  4.481921  9.091342  8.007685 10.176934  9.533463 11.184276
##          [,281]     [,282]    [,283]   [,284]    [,285]    [,286]    [,287]
## [1,] 12.875246  14.118784  7.701721 6.909087 20.302109 13.590789 12.853320
## [2,]  9.447111  18.385773 10.356595 8.677709 17.379668  9.887297  9.891466
## [3,]  5.212696   8.268924  5.528690 6.457428 10.081632  5.851260  4.403852
## [4,]  6.187011  12.778495  5.719527 6.695590 10.296840  9.438946  4.502928
## [5,]  7.118340   5.847900  7.064413 6.158669  7.651490  8.272529  5.753368
## [6,]  7.436443 122.151449 13.163852 7.459971  7.776064  8.957063  6.338131
##          [,288]    [,289]    [,290]    [,291]    [,292]    [,293]    [,294]
## [1,] 16.024871 15.307394 10.107252 10.349553 11.081630 10.470163  9.935529
## [2,] 10.929883  9.949402  9.011008  7.887450  9.818130  8.456000  8.256523
## [3,]  5.866074  5.972936  9.234263  4.794259  6.675226  7.377225  5.261869
## [4,]  6.206539  9.882331  9.146539  4.902405  7.010059  7.411178  5.442795
## [5,]  4.600601  6.928499  8.396430  4.454133  6.226294  6.048097  4.394903
## [6,]  5.447123  6.884568 11.309062  7.045291  6.216782  6.165439 20.557009
```

```
##        [,295]    [,296]    [,297]      [,298]    [,299]    [,300]    [,301]
## [1,] 5.870481 10.833617 11.214380  26.838140 11.362318 10.590375 10.896324
## [2,] 2.267006 10.085223 10.678098  23.616368 10.573109  8.024063 24.575355
## [3,] 2.618094  7.640165  6.785832   6.609471  6.934673  5.472579  8.268958
## [4,] 3.102777  9.026305  6.855697   6.569259  6.893617  8.057728 11.368956
## [5,] 3.576913  7.593470  6.906833  27.210301  6.085483  4.980388  6.364792
## [6,] 3.536167  9.319001  6.752331 120.145992  6.293469  7.131954 52.927854
##         [,302]    [,303]     [,304]    [,305]    [,306]   [,307]    [,308]
## [1,] 22.910797 18.456168   9.577370  6.009893  7.217176 27.38810 12.743185
## [2,] 18.336153 14.134199  35.850235 18.879956  8.423409 15.54423  7.514032
## [3,]  5.383874  6.762612   5.148437  8.253716  4.685714  5.90784  5.293841
## [4,] 24.797268  6.944723 117.796142 22.741784  5.050727 71.98669  8.631611
## [5,]  8.896270  5.936715 753.421641  5.083239  6.779959 22.86464  8.074329
## [6,] 87.533122  8.353264 245.712526 82.184246 12.343815 19.88627  9.705777
##         [,309]    [,310]   [,311]    [,312]    [,313]    [,314]    [,315]
## [1,] 16.846533 13.112033 12.31484 11.651403 12.273583 15.365576 15.405068
## [2,] 11.469297  8.603912 23.39308  5.567253  9.799580 10.866880 12.522402
## [3,]  6.493822  3.586231 10.23640  3.562301  6.108872  6.506803  9.236670
## [4,]  6.945395  4.856093 23.98594  5.422422  7.337576  6.588133 10.664151
## [5,]  6.085384  4.260150 12.00750  8.610542  6.189288  4.936640  8.529447
## [6,]  6.981074  5.208297 15.35022  7.821223  6.138451  7.029763 10.504981
##         [,316]    [,317]    [,318]    [,319]    [,320]    [,321]   [,322]
## [1,] 11.553905 15.224978 13.261643 11.214206 12.716349 10.938404 8.068320
## [2,] 11.894930 11.630520  9.813226  9.887571  8.880782  8.977323 4.548533
## [3,]  5.592204  5.533237  5.418861  4.664543  5.090697  6.835267 3.859564
## [4,] 11.966905  6.123639  5.465343  7.421203  5.424896  6.998996 3.851217
## [5,]  3.909930  4.912063  5.711565  6.376829  4.117358  5.906269 5.956427
## [6,]  4.792733  6.349009  7.077550  8.220724  4.239085  5.810517 5.950045
##         [,323]    [,324]    [,325]    [,326]   [,327]    [,328]    [,329]
## [1,]  8.805191  9.756635 10.265868 15.274848 6.277155 13.942521 16.163324
## [2,]  7.693190  7.697984  6.322640 13.089154 3.438916 10.659524 13.167190
## [3,]  7.936772  5.451131  6.034554  8.086126 6.725776  5.130321  6.000593
## [4,]  8.218747  5.125485  6.037600 12.819045 6.758496  5.935018  8.810055
## [5,]  8.644834 10.454898  4.716686 11.100950 7.892234  5.789532  5.262800
## [6,] 16.175023 13.913075  5.039167 11.858942 7.914459  7.171331  4.644955
##         [,330]    [,331]       [,332]    [,333]    [,334]    [,335]
## [1,] 12.024519 10.485701     18.63805 20.116405  10.65966  9.744398
## [2,] 12.169450  8.346681     22.63152 24.128375  36.92948  7.933238
## [3,]  8.134328  5.433535     23.36649 15.176875  10.26973  7.081669
## [4,] 11.720561  6.263097    570.62824 18.736380 272.72519  7.233219
## [5,]  9.964003  8.214581     43.48214  9.718741 522.34068  6.447522
## [6,] 12.636641 10.002827 233288.29360 49.548627  55.69484 12.191354
##           [,336]    [,337]    [,338]    [,339]    [,340]    [,341]
## [1,]   11.379833 11.440108 11.240765 13.886713 14.753011 10.136626
## [2,]   25.357820  6.901492 11.954009  9.446994 11.658710  8.324708
## [3,]    7.075439  5.046857  6.437542  4.197125  6.253560  5.672774
## [4,]  168.115974  6.554589  8.573186  4.803645  8.373727  6.407874
## [5,]  660.574227  5.440074  8.056105  6.199658  6.758282  5.771901
## [6,] 1027.753935  7.872000  8.779095 12.060299  7.251532  5.932084
##         [,342]    [,343]   [,344]    [,345]   [,346]    [,347]    [,348]
## [1,] 10.970347 11.814599 7.623767 11.093897 8.601760 10.089547 15.613013
## [2,] 11.361809  9.984047 8.343379  7.240152 8.583638 19.807765 13.382973
## [3,]  5.004052  6.688559 6.339115  3.886577 4.967890  9.841728  7.104401
## [4,]  5.319049  6.629360 6.320744  3.725008 5.579866 12.900865  9.336579
```

```
## [5,]  6.639587  6.673730 5.595077  9.672382 9.175156  6.469284  9.711606
## [6,] 11.874830  7.161540 5.766550 11.022695 9.810918 40.615458 11.154984
##          [,349]    [,350]    [,351]    [,352]    [,353]    [,354]    [,355]
## [1,] 10.826821 12.223814 17.278020 13.609141  7.834282 18.970465 14.982673
## [2,]  9.106068  9.324562 13.585461  9.831456  7.210286 12.236338 12.176412
## [3,]  4.757201  4.620481  7.927933  3.363326  5.706771  7.973599  5.848932
## [4,]  5.693403  5.151561 10.419715  3.358043  6.864938 11.686433  8.151632
## [5,]  5.937932  6.098298  7.009454  5.473939  6.041762  8.105256  6.170600
## [6,]  6.134669  4.926955  7.425000 12.815518 11.740001 25.952409  7.689667
##          [,356]    [,357]    [,358]    [,359]    [,360]    [,361]    [,362]
## [1,] 11.497322 10.501092 10.569327  9.434930 15.654087 16.106387  8.803101
## [2,] 23.130586  7.007299 10.142193  9.350733 11.111734 12.868147 20.078782
## [3,]  6.620669  5.277879  5.554640  2.348927  5.001391  6.955297  8.639476
## [4,] 12.443527  5.734161  5.494168  2.840513  8.005259  7.459067  7.940646
## [5,]  7.031139  4.754566  7.772649  5.738182  7.746285  6.093197  5.969406
## [6,] 60.775898  5.077683 11.567041 13.734149 10.086201  6.234699 34.123640
##           [,363]    [,364]    [,365]    [,366]   [,367]   [,368]    [,369]
## [1,]   8.315837 14.499944 12.898798 14.391634 5.976088 9.843923  9.910608
## [2,]  45.264016 13.477730  9.923561  8.025571 4.564360 5.793588 10.285510
## [3,]   4.428088  5.719988  4.961538  9.844617 4.103561 3.355424  6.356835
## [4,] 250.129593 14.707354  5.556770  9.188482 4.077083 3.978001  6.850317
## [5,] 340.940477 22.039559  5.836078  8.606525 4.067208 3.166241  6.170631
## [6,] 269.422807 83.535188  6.119679  9.118902 4.522955 3.148069  6.446900
##          [,370]    [,371]   [,372]    [,373]    [,374]    [,375]     [,376]
## [1,] 11.545978 10.153708 6.623473 14.079037 18.167697 11.375473   8.811657
## [2,] 11.282954  7.927041 4.884685  9.449935 11.598363 10.418267  49.366498
## [3,]  7.581365  6.038316 4.015798  5.684504  7.738879  6.600821  33.019592
## [4,]  7.583588  6.009099 4.100665  6.960690  8.131033  7.546629  96.138395
## [5,]  6.396928  4.860988 3.370794  4.611608  6.916604  7.400711 126.183171
## [6,]  6.432343  5.691302 3.354337  5.136565  7.169217 12.751489 233.864810
##          [,377]    [,378]   [,379]    [,380]    [,381]    [,382]    [,383]
## [1,] 13.712188 10.653137 21.59679 13.301624  7.716521 15.978325 12.001883
## [2,]  9.617986 10.116029 18.13920 11.015218  9.692353 12.001408 11.840355
## [3,]  6.739118  5.824591 12.42098  8.236249  6.446687  5.802834  7.322771
## [4,]  6.767123  5.959100 12.94564  9.391878  6.646294  6.769178  7.338425
## [5,]  9.832990  5.311870 10.77204  7.356002  7.199210  4.606543  6.378437
## [6,] 11.540895 17.746889 10.74259  8.223165 14.122109  6.714497  6.332432
##          [,384]    [,385]    [,386]    [,387]    [,388]    [,389]    [,390]
## [1,] 10.018202 10.912672 25.089735 12.932698  7.412306 10.977876 12.917063
## [2,] 10.411932  9.069478 19.401370  9.611202  6.561326 10.726173 11.328249
## [3,]  8.867902  6.155662  4.359674  4.409806  3.628771  5.527416  4.640135
## [4,]  9.337199  6.266118  4.299121  6.164126  4.369509  6.630426  5.335193
## [5,]  7.554817  6.546479 50.053861  7.621379  5.543002  5.045933  5.557230
## [6,]  8.178050  6.370325 54.566964  9.849158 13.755760  5.134218  7.729562
##          [,391]    [,392]    [,393]   [,394]   [,395]    [,396]    [,397]
## [1,] 13.905184 14.918251  9.032093 8.704580 7.423394 11.143023  8.615147
## [2,] 10.587813 13.085620  9.019910 7.162476 5.335974  6.821579  9.344797
## [3,]  7.152971  7.905699  4.940435 4.789088 6.828571  8.274596  4.202222
## [4,]  8.916794  7.932046  7.094410 6.351646 7.791007  9.078131  4.235783
## [5,]  7.020938  7.646224  6.197446 4.441777 6.016400  8.265502  3.113043
## [6,]  7.347223  8.994479 14.231067 4.723176 5.966268 11.294002 19.385735
##          [,398]    [,399]    [,400]
## [1,] 12.478926 15.012322 13.850896
## [2,] 10.302635 13.409601 13.657370
```

```
## [3,]  6.348131  8.744346  9.002654
## [4,]  6.381438  8.802053  9.462265
## [5,]  8.805261  6.777795  8.285385
## [6,]  8.362911  6.777458  8.878981
```

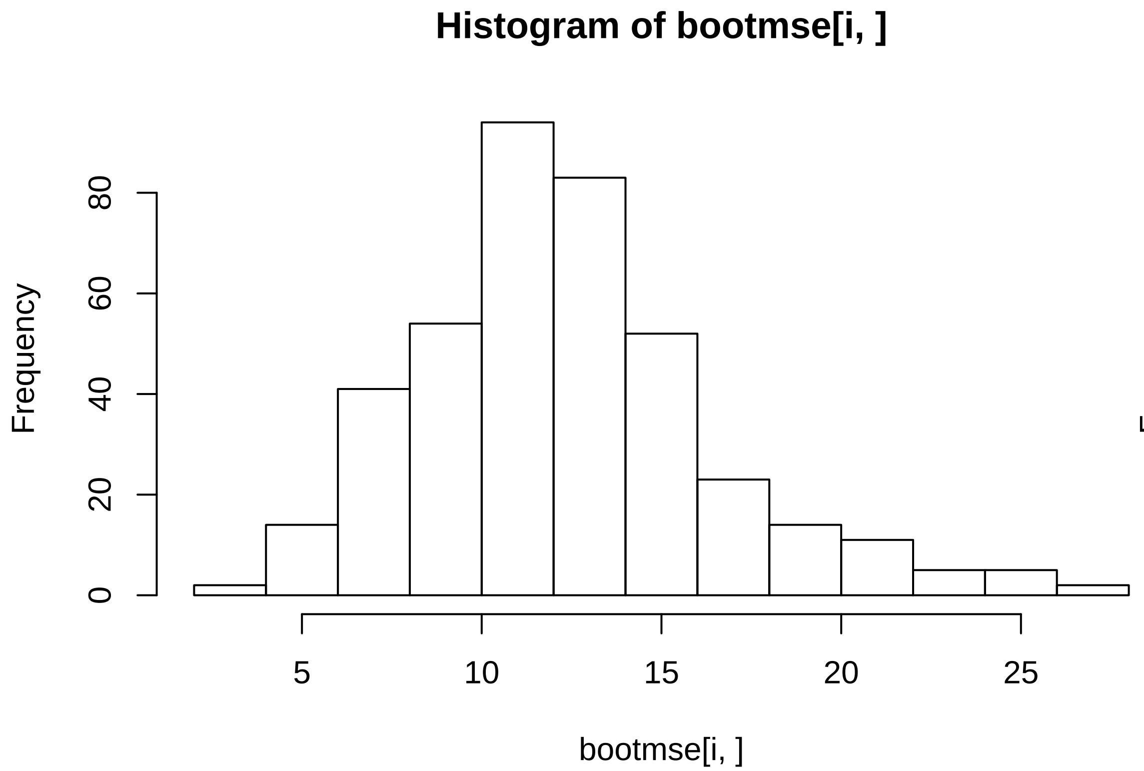
```r
plot(rowMeans(bootmse))
```

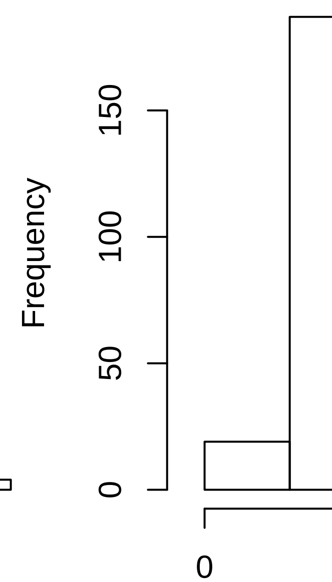


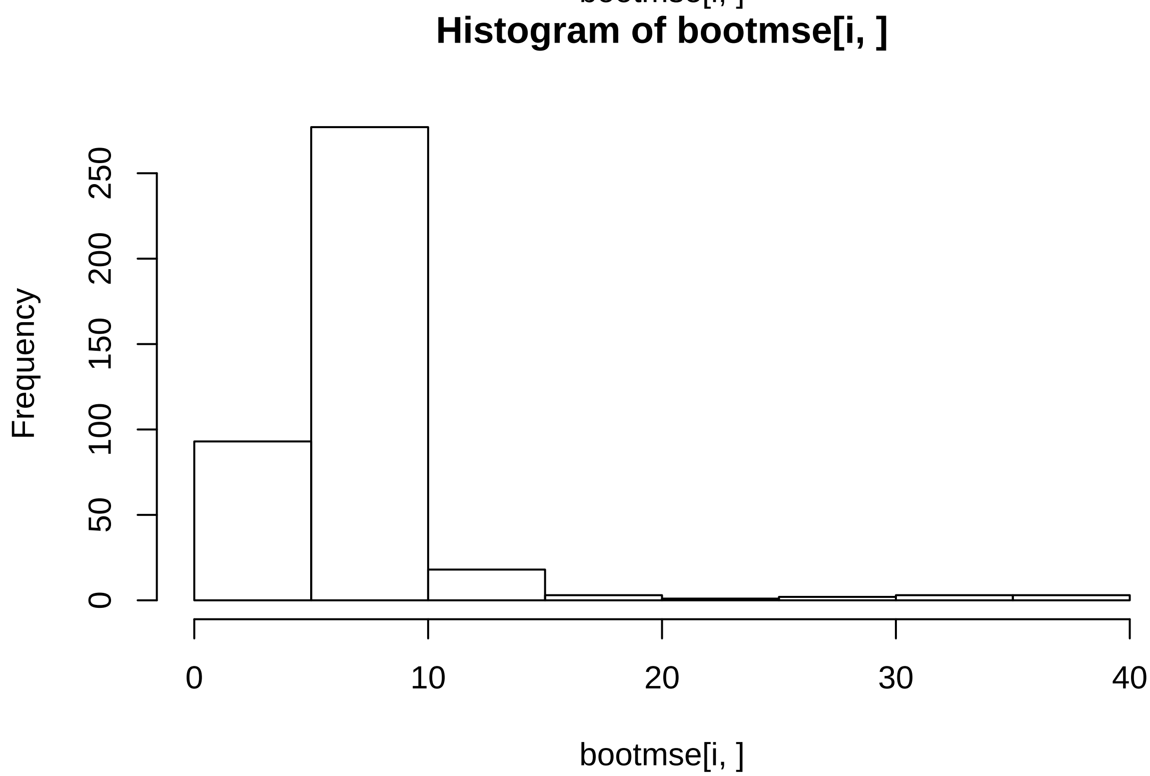

```r
apply(bootmse, 1, sd)
```

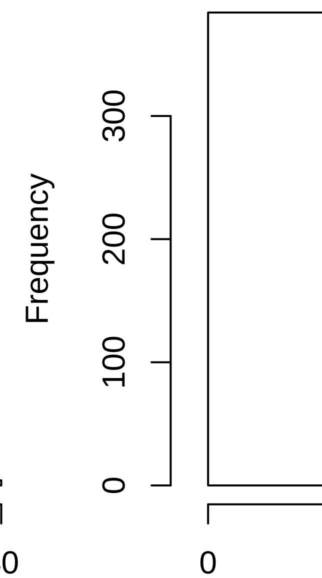
```
## [1]     4.082137     7.424487     4.358966    66.527133   626.811130
## [6] 20211.953385
```

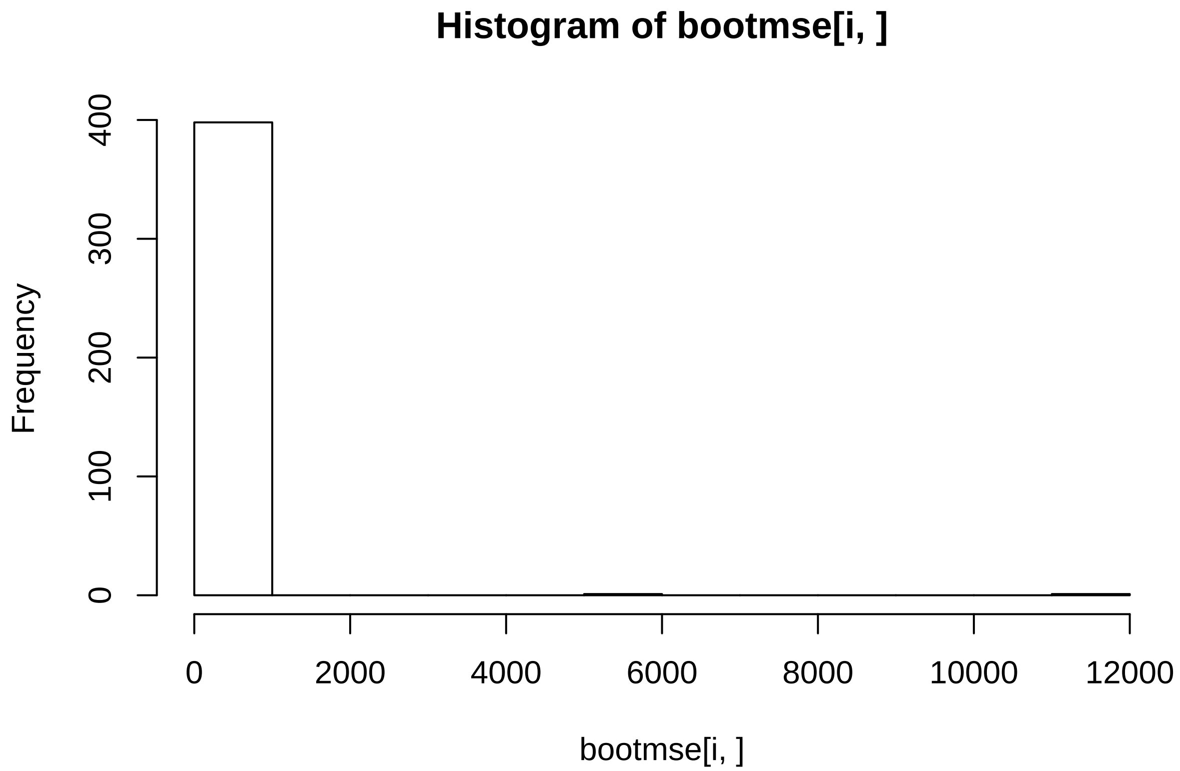
```r
for (i in 1:6){
  hist(bootmse[i, ])
}
```

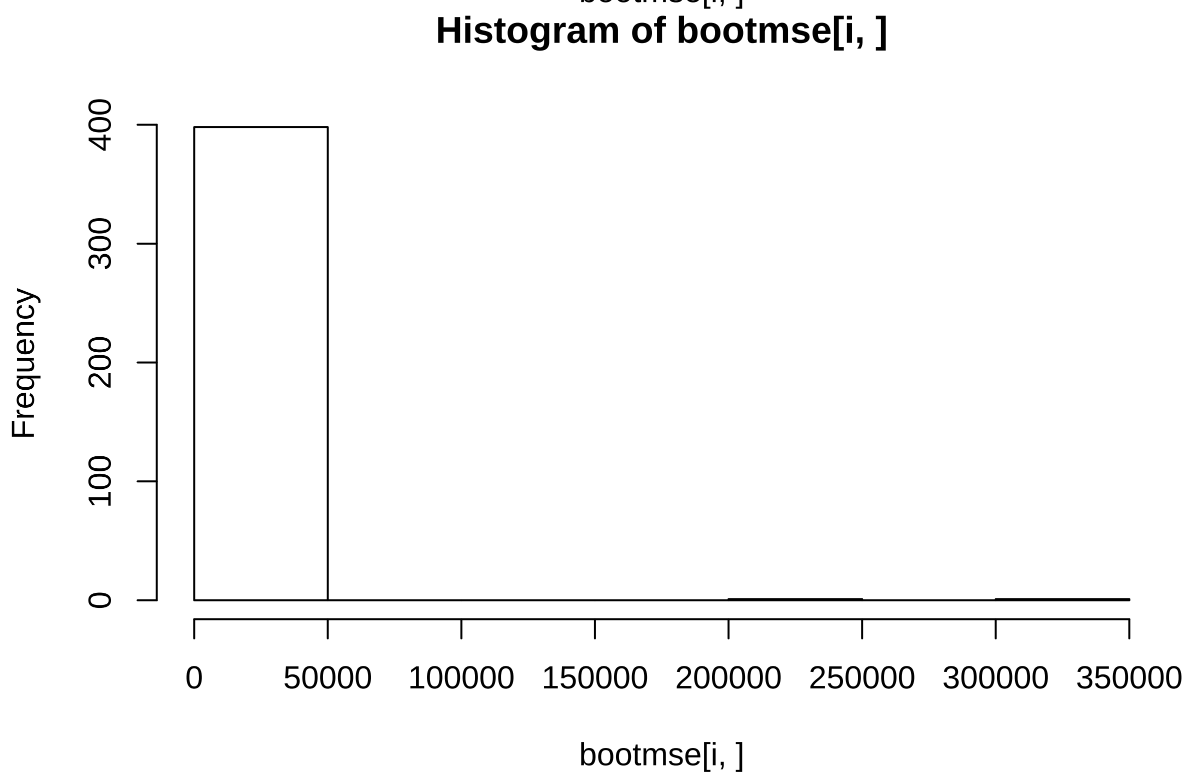

Histogram of bootmse[i,]
Frequency
0
20
40
60
80
5
10
15
20
25
bootmse[i,]


Frequency
0
50
100
150
0


Histogram of bootmse[i,]
Frequency
0
50
100
150
200
250
0
10
20
30
40
bootmse[i,]


Frequency
0
100
200
300
0





Sample with replacement!!!

first, sample with replacement from the data (this serves as the training set); second, train the model on the sampled data; third, test the model on the data that is not in the sample and compute the performance metric.

SD is pretty big as the polynomial has higher power in general, so it means not stable. There will be big gaps between your data. (I think it is because bootstrap works well for big data, but we have less than 40 datas for ours)