Assignment 6 (S-670)

FNU Anirudh

November 20, 2015

**Note:-** Question 1 and 2 , done at last

Solution 3

library(aplpack)

## Loading required package: tcltk

# a)  
x = c(576, 635, 558, 578, 666, 580, 555, 661, 651, 605, 653, 575, 545, 572, 594)  
y = c(339, 330, 281, 303, 344, 307, 300, 343, 336, 313, 312, 274, 276, 288, 296)  
Data = data.frame(x,y)  
colnames(Data) <- list("LSAT","GPA")  
mean = 0.5\*log(1.77637/(1-0.77637))   
n = 15  
var = 1/(n -3)  
se = sqrt(var)/sqrt(n)  
CI= mean + c(-1,1)\*1.96\*se  
CI

## [1] 0.8900774 1.1822569

# b)  
  
calculatePV = function(data) {  
 n = length(data[[1]])  
 rho = cor(data, method="pearson")[1,2]  
 yall = 0.5\*log((1+rho)/(1-rho))  
 PV = numeric(n)  
 for( i in 1:n) {  
 rhominusi = cor(data[-i,], method="pearson")[1,2]  
 yminusi = 0.5\*log((1+rhominusi)/(1- rhominusi))  
 PV[i] = n\*yall - (n-1)\*yminusi  
 }  
 PV  
}  
PVAll = calculatePV(Data)  
JKEstimate = mean(PVAll)  
JKEstimate

## [1] 0.9170373

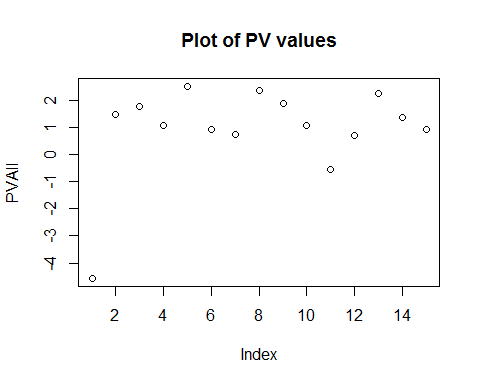
varJK = sum((PVAll - JKEstimate)^2)/15\*14  
CI = JKEstimate + c(-1,1)\*qt(0.975,df=nrow(Data)-1)\*sqrt(varJK)  
CI

## [1] -12.31516 14.14923

# c)  
stem.leaf(PVAll)

## 1 | 2: represents 1.2  
## leaf unit: 0.1  
## n: 15  
## LO: -4.56529576001235  
## 2 -0. | 5  
## -0\* |   
## 0\* |   
## 6 0. | 7799  
## (4) 1\* | 0034  
## 5 1. | 78  
## 3 2\* | 23  
## 1 2. | 5

plot(PVAll,main="Plot of PV values")



PVre = calculatePV(Data[-1,])  
JM = mean(PVre)  
JM

## [1] 1.359238

varJKRe = sum((PVre - JM)^2)/(14\*(14-1))  
varJKRe

## [1] 0.1096831

#seJKRecalc = sqrt(varJK) #0.33  
CI = JM + c(-1,1)\*qt(0.975,df=13)\*sqrt(varJKRe)  
CI

## [1] 0.6437573 2.0747180

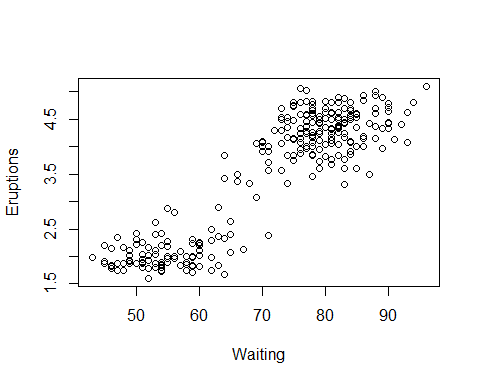
# d)  
bootstrap = function(data, nsim) {  
 theta = numeric(nsim)  
 varTheta = numeric(nsim)  
   
 n = length(data[[1]])  
 index = 1:n  
 for (i in 1:nsim){  
 sampleindex= sample(index,n,replace=TRUE)  
 PViter = calculatePV(data[sampleindex, ])  
 theta[i] = mean(PViter)  
 varTheta[i] = sum((PViter - theta[i])^2)/(n\*(n-1))  
 }  
   
 ciLower = mean(theta) - 1.96\*mean(varTheta)  
 ciUpper = mean(theta) + 1.96\*mean(varTheta)  
   
 output = list(thetaBS = mean(theta), varBS = mean(varTheta),  
 theta = theta, varTheta = varTheta,  
 ciLower = ciLower, ciUpper = ciUpper)  
 output  
}  
Results = bootstrap(Data, 10)  
Results

## $thetaBS  
## [1] 1.050227  
##   
## $varBS  
## [1] 0.1946566  
##   
## $theta  
## [1] 0.7980123 1.4864029 1.1025587 1.1333013 0.7835986 0.5434859 0.9603396  
## [8] 0.8628983 1.9851469 0.8465256  
##   
## $varTheta  
## [1] 0.77476139 0.06224270 0.05472431 0.04101526 0.10988584 0.07823188  
## [7] 0.20508282 0.22140352 0.09468920 0.30452915  
##   
## $ciLower  
## [1] 0.6687001  
##   
## $ciUpper  
## [1] 1.431754

\*\* e) Effect of Outliers on Confidence Interval is being reduced by Bootstrapping and Jacknifing. We got a) 0.89, 1.182 b)-12.31 and 14.14 first and then 0.643,2.074 after removing outlier d) 0.408 and 1.551. Effect of Bootstrapping is reduced greatly by bootstrapping.

Solution 4

rrline1 <- function(x,y) {  
 n3 <- floor((length(x)+1.99)/3)  
 x.order <- order(x)  
 medxL <- median(x[x.order][1:n3])  
 medxR <- median(rev(x[x.order])[1:n3])  
 medyL <- median(y[x.order][1:n3])  
 medyR <- median(rev(y[x.order])[1:n3])  
 slope1 <- (medyR - medyL)/(medxR - medxL)  
 int1 <- median(y - slope1 \* x)  
 # print(c(paste("Intercept = ", format(round(int1,5))),  
 # paste("Slope = ",format(round(slope1,5)))))  
 newy <- y - slope1\*x - int1  
 sumres <- sum(abs(newy))  
 list(a=int1, b=slope1, sumres = sumres, res=newy)  
}  
#Code courtesy: Prof David King Lecture Notes  
run.rrline <- function(x,y,iter=5) {  
 out.coef <- matrix(0,iter,3)  
 newy <- y  
 for (i in 1:iter) {  
 rr <- rrline1(x,newy)  
 out.coef[i,] <- c(rr$a,rr$b,rr$sumres)  
 newy <- rr$res  
 }  
 dimnames(out.coef) <- list(format(1:iter),c("a","b","|res|"))  
 aa <- sum(out.coef[,1])  
 bb <- sum(out.coef[,2])  
 cc <- sum(abs(y - aa - bb\*x))  
 res <- y - aa - bb\*x  
 out.coef <- rbind(out.coef,c(aa,bb,cc))  
 #print(round(out.coef,5))  
 list(a = aa, b = bb, res = res, coef=out.coef)  
}  
bootprog = function (x,nsim)  
{  
 # This program is a silly program which will be used to estimate the  
 # bootstap error of the sample median statistic  
 # the input data is a vector x of data.  
 # nsim is the number of bootstrap simulations  
 n = length(x)  
 index = 1:n  
 m = median(x)  
 stat = numeric(nsim)  
 ooberr = numeric(nsim)  
 for (i in 1:nsim){  
 sampleindex= sample(index,n,replace=TRUE)  
 stat[i] = median(x[sampleindex])  
 oobindex = setdiff(index,unique(sampleindex))  
 oobdat = x[oobindex]  
 ooberr[i] = sum((oobdat-stat[i])^2)/length(oobindex)  
 }  
 bias = m - mean(stat)  
 variance = var(stat)  
 se = sqrt(variance)  
 avgooberr = mean(ooberr)  
 output = list(bias=bias,var=variance,se=se,avgooberr=avgooberr)  
 output  
}  
  
cvprog = function (x,nfold)  
{  
 # This program is a silly program which will be used to estimate the  
 # crossvalidation error of the sample median statistic  
 # the input data is a vector x of data.  
 # nfold is the number of folds you want to divide your data up into  
 n = length(x)  
 m = floor(n/nfold)  
 # Generally speaking n/nfold would be an integer, however if it is not  
 # and the remainder of n/nfold is k then we will take the extra k datapoints  
 # and give them to the first k folds.  
 folds = rep(1:nfold,m)  
 k = n - length(folds)  
 if(k>0){folds = c(folds,1:k)}  
 # now folds is of length n and we can randomly permute the indicies  
 foldindicies = sample(folds,n,replace=FALSE)  
 m = median(x)  
 stat = numeric(nfold)  
 cverr = numeric(nfold)  
 for (i in 1:nfold){  
 b = foldindicies == i  
 stat[i] = median(x[!b])  
 cverr[i] = sum((x[b]-stat[i])^2)/length(x[b])  
 }  
 bias = m - mean(stat)  
 variance = var(stat)  
 se = sqrt(variance)  
 avgcverr = mean(cverr)  
 output = list(bias=bias,var=variance,se=se,avgcverr=avgcverr)  
 output  
}  
  
getOOBforBootstrap = function(oobdata) {  
 # Based on the class slides and hints from the professor,  
 # this function calculates rrline for  
 # the oob data to get a and b, shuffles the residuals,   
 # adds them to the original data  
 # calculate rr line again to get new a and b, this is repeated till we have   
 # n estimates of a and b where n is the number of oob samples  
 # oobdata  
 originalData = oobdata  
 n = length(originalData[[1]])  
 aOutofBag = numeric(n)  
 bOutofBag = numeric(n)  
 for (q in 1: n) {  
 results = run.rrline(originalData[[1]], originalData[[2]])  
 residuals = results$res  
 aOutofBag[q] = results$a  
 bOutofBag[q] = results$b  
 shuffledResiduals = sample(residuals)  
 originalData[[2]] = oobdata[[2]] + shuffledResiduals  
 }   
 list(a0 = aOutofBag, b0 = bOutofBag)  
}  
  
  
  
rrlineWithBootstrap = function(data, nsim) {  
 # This function runs rrline with bootstrapping  
 n = length(data[[1]])  
 index = 1:n  
   
 # We maintain 2 different stats and oob for each a and b  
 statA = numeric(nsim)  
 statB = numeric(nsim)  
 ooberrA = numeric(nsim)  
 ooberrB = numeric(nsim)  
   
 # Run rrline to get initial statistic on entire data,  
 # for confirmatory purposes only  
 results = run.rrline(data[[1]], data[[2]])  
 a = results$a  
 b = results$b  
   
 #Run nsim times  
 for (i in 1:nsim){  
 sampleindex= sample(index,n,replace=TRUE)  
 results = run.rrline(data[[1]][sampleindex], data[[2]][sampleindex])  
 statA[i] = results$a  
 statB[i] = results$b  
   
 oobindex = setdiff(index,unique(sampleindex))  
 oobResults = getOOBforBootstrap(data[oobindex,])  
   
 ooberrA[i] = sum((oobResults$a0-statA[i])^2)/length(oobindex)  
 ooberrB[i] = sum((oobResults$b0-statB[i])^2)/length(oobindex)  
 }  
   
 # Calculate bias and variance  
 biasA = a - mean(statA)  
 varianceA = var(statA)  
 biasB = b - mean(statB)  
 varianceB = var(statB)  
   
 # Calculate standard error and average oob error  
 seA = sqrt(varianceA)  
 seB = sqrt(varianceB)  
 avgooberrA = mean(ooberrA)  
 avgooberrB = mean(ooberrB)  
   
 output = list(a= mean(statA), b = mean(statB),  
 biasA=biasA,varA=varianceA, biasB = biasB, varB = varianceB,  
 seA=seA,seB = seB, ooberrA = ooberrA, ooberrB = ooberrB,  
 avgooberrA=avgooberrA, avgooberrB = avgooberrB)  
 output  
}  
  
q4Data = data.frame(faithful$waiting, faithful$eruptions)  
colnames(q4Data) <- c("Waiting", "Eruptions")  
plot(q4Data)



Solution 5

library(DAAG)

## Loading required package: lattice

getOOBforCV = function(oobdata) {  
originalData = oobdata  
 n = length(originalData[[1]])  
 aOutofBag = numeric(n)  
 bOutofBag = numeric(n)  
 for (q in 1: n) {  
 results = run.rrline(originalData[[1]], originalData[[2]])  
 residuals = results$res  
 aOutofBag[q] = results$a  
 bOutofBag[q] = results$b  
 shuffledResiduals = sample(residuals)  
 originalData[[2]] = oobdata[[2]] + shuffledResiduals  
 }  
 print(aOutofBag)  
 print(bOutofBag)  
 list(a0 = aOutofBag, b0 = bOutofBag)  
}  
newcvprog = function (x,nfold)  
{  
 n = length(x)  
 m = floor(n/nfold)  
  
 folds = rep(1:nfold,m)  
 k = n - length(folds)  
 if(k>0){folds = c(folds,1:k)}  
 foldindicies = sample(folds,n,replace=FALSE)  
 statA = numeric(nfold)  
 statB = numeric(nfold)  
  
 ooberrA = numeric(nfold)  
 ooberrB = numeric(nfold)  
 results = run.rrline(x[[1]], x[[2]])  
 a = results$a  
 b = results$b  
  
 statA = numeric(nfold)  
 statB = numeric(nfold)  
  
 cverrA = numeric(nfold)  
 cverrB = numeric(nfold)  
  
 for (i in 1:nfold){  
 print(paste("Fold ", i))  
 p = foldindicies == i  
 resultiter = run.rrline(x[[1]][!p], x[[2]][!p])  
 statA[i] = resultiter$a  
 statB[i] = resultiter$b  
  
 print(statA)  
 print(statB)  
 oobResults = getOOBforCV(x[p,])  
 cverrA[i] = sum((oobResults$a0-statA[i])^2)/length(p)  
 cverrB[i] = sum((oobResults$b0-statB[i])^2)/length(p)  
  
 }  
 biasA = a - mean(statA)  
 varianceA = var(statA)  
  
  
 biasB = b - mean(statB)  
 varianceB = var(statB)  
  
 seA = sqrt(varianceA)/sqrt(nfold)  
 seB = sqrt(varianceB)/sqrt(nfold)  
  
 avgcverrA = mean(cverrA)  
 avgcverrB = mean(cverrB)  
  
 output = list(biasA=biasA, varA=varianceA,  
 biasB = biasB, varB = varianceB,  
 seA=seA, seB=seB,  
 avgcverrA=avgcverrA, avgcverrB = avgcverrB)  
 output  
}  
q5Data = data.frame(faithful$waiting, faithful$eruptions)  
colnames(q5Data) <- c("Waiting", "Eruptions")  
resultsq5 = newcvprog(q5Data, 3)

## [1] "Fold 1"  
## [1] -1.383236 0.000000 0.000000  
## [1] 0.06937156 0.00000000 0.00000000  
## [1] -1.98705398 -1.85157440 -1.02493458 -2.27895288 -1.26071143  
## [6] -1.17486614 -1.52658021 -2.08683503 -1.55093277 -1.26634000  
## [11] -1.69391234 -1.44354647 -0.86776925 -4.19342070 0.35967328  
## [16] -3.48963733 -1.31501007 -2.16262319 -3.08048847 -2.17034854  
## [21] -1.33264022 -2.67857198 -0.83823602 -3.42049430 -1.58324913  
## [26] -3.63507794 -0.58396768 -2.42314875 -3.63182220 -0.88549330  
## [31] -0.63405830 -3.56451877 -0.20388043 -2.55674944 -1.99389760  
## [36] -1.03677598 -4.03389934 -3.19522654 -2.88946506 -2.76369166  
## [41] -2.21186786 -0.60829479 -2.14590194 -4.87380944 -0.99857402  
## [46] -2.75953024 -0.38431723 1.00586106 3.03877663 0.45077412  
## [51] -4.50248327 -2.84051424 -1.81118058 -2.54900028 -1.36671207  
## [56] 1.27303516 -5.08573957 -6.01235675 -0.18317027 -3.40823471  
## [61] -1.22853528 2.40684736 -2.91652824 -4.91242480 0.45222949  
## [66] -3.30884238 -2.60840335 -1.48820333 -5.73990232 -2.53561204  
## [71] 0.03255303 -1.52410631 -0.89172743 -0.13440970 -0.52863047  
## [76] -3.56646707 -1.46956238 -2.93470033 2.00085475 -0.14130132  
## [81] -1.86389050 -0.79868506 -2.78823369 1.16606372 -2.58596474  
## [86] -4.78062957 -1.92351949 -1.31113979 -3.50209008 -4.74256242  
## [91] 0.06150328 -1.99895934 1.36813174 -0.72035368 -0.19388860  
## [96] -0.63155035 0.65590714 -4.53798082 -2.65799314 -3.29100572  
## [101] -2.58516096 -2.04362627 -3.07030494 -4.36995475 1.46278865  
## [106] -0.68663850 2.38835129 -2.71760236 0.66506486 -2.88764928  
## [111] -1.23868645 -3.78419263 0.68422411 0.71175988 -1.53346970  
## [116] 2.44173824 -1.91590589 3.66923558 -6.12056850 -2.76974213  
## [121] -3.77306082 -2.87731398 -0.86157740 0.37329925 -0.07388773  
## [126] -2.74165658 3.48002719 -2.91504609 1.21915073 -4.95309295  
## [131] -3.13320360 -3.23023708 0.14631105 -3.23112256 -0.63034855  
## [136] -0.19132357  
## [1] 0.077441358 0.075708781 0.063360607 0.081069676 0.067257771  
## [6] 0.064461660 0.071831073 0.076162308 0.071437745 0.064936277  
## [11] 0.073156473 0.070403670 0.062591839 0.105912324 0.045726206  
## [16] 0.098561268 0.067234620 0.079526686 0.094481332 0.078898058  
## [21] 0.069850737 0.086244181 0.060842739 0.098538046 0.071030986  
## [26] 0.102384926 0.059223611 0.084829704 0.098444145 0.061148999  
## [31] 0.060706616 0.098284696 0.052140321 0.086491575 0.074507688  
## [36] 0.063608024 0.106978869 0.096517302 0.090425695 0.087302123  
## [41] 0.081337717 0.056429958 0.077842141 0.122427571 0.063115345  
## [46] 0.086468319 0.054367009 0.037554568 0.002563788 0.041774139  
## [51] 0.114960540 0.089746293 0.074644160 0.084938265 0.067246243  
## [56] 0.026168877 0.123408699 0.136930649 0.050782276 0.096682743  
## [61] 0.065062811 0.015077274 0.091180976 0.118347776 0.044375087  
## [66] 0.094896666 0.087896927 0.071672270 0.130405528 0.087125318  
## [71] 0.043039622 0.068940093 0.061471175 0.048515411 0.059257687  
## [76] 0.098801723 0.069745300 0.094331207 0.020461766 0.051755971  
## [81] 0.075082033 0.057163096 0.091277477 0.030707131 0.086451338  
## [86] 0.117709337 0.076557321 0.064936961 0.100365935 0.113929957  
## [91] 0.050401057 0.076176495 0.033334300 0.056294676 0.053521481  
## [96] 0.059200494 0.035662808 0.117274419 0.088330073 0.096431852  
## [101] 0.085075637 0.082834241 0.092981934 0.107752041 0.025054446  
## [106] 0.055766536 0.015876670 0.088555273 0.038202720 0.092173542  
## [111] 0.062096828 0.105519902 0.038602423 0.033638100 0.069519577  
## [116] 0.016006264 0.075821126 -0.005306682 0.134865285 0.088596138  
## [121] 0.105273943 0.090190557 0.057797156 0.043962166 0.048871664  
## [126] 0.092012537 -0.004896847 0.090037018 0.033506156 0.118190533  
## [131] 0.096074394 0.098219861 0.045316818 0.096036829 0.052127872  
## [136] 0.049181097  
## [1] "Fold 2"  
## [1] -1.383236 -1.987054 0.000000  
## [1] 0.06937156 0.07744136 0.00000000  
## [1] -1.38323613 -1.50441408 -1.35794621 -2.21841013 -2.07457476  
## [6] -1.63005288 -3.00377925 -2.01935974 -0.58032496 -2.21824706  
## [11] -2.21891659 -2.42186388 -1.58113389 -1.55255441 -0.01608162  
## [16] -2.15056020 1.23154443 0.22281423 0.46259140 -2.67539120  
## [21] 0.19703414 -1.57828043 -3.32392576 -0.55331357 0.24572479  
## [26] -0.23890730 -0.75002349 -3.13536047 -3.95263865 -1.96145723  
## [31] 0.96353767 -0.60045992 -0.57903756 -1.99926910 0.23627231  
## [36] -3.45121836 -3.30370125 -0.35890940 -3.29976133 0.23275710  
## [41] -1.97581487 -0.12286225 -2.03834015 -1.39639808 -1.15037185  
## [46] 2.71909225 -3.31339472 -1.73526002 0.46135041 -3.47777648  
## [51] -2.71169975 -2.89369638 2.32048608 -1.69614847 -2.12978391  
## [56] -0.63672833 -3.60801335 -3.84103404 1.13849938 0.67992916  
## [61] -1.51750096 -2.55014370 -0.82362056 -1.35784763 -4.22844749  
## [66] 1.25837146 -2.94841252 3.06315034 -3.35457089 -1.89874957  
## [71] -0.69784678 -2.21702707 -3.42803141 -6.51011977 -6.46494022  
## [76] -0.21498298 -4.75283257 -0.54635044 4.58997527 0.11228264  
## [81] -0.84628756 -1.46901540 -7.43369738 -0.70419297 -0.53962408  
## [86] 3.13964278 -3.09573989 -4.27591199 -0.33223146 -4.20837043  
## [91] 0.93080203 -4.34404654 -3.45753890 -5.23601645 -2.20847433  
## [96] -1.73655225 -5.53488505 -3.85987859 -1.30546163 0.85884061  
## [101] -2.57600772 0.89037901 0.01752309 -0.34003249 2.84147618  
## [106] -1.88346683 5.26945351 3.81962424 0.51379925 1.86433787  
## [111] 0.89307839 0.33684986 2.42964228 1.37270690 1.95257582  
## [116] -5.00922827 -2.93720148 -7.59686349 -3.96800907 -5.46367601  
## [121] -3.81016844 1.60107347 -3.84968114 2.93714987 0.53358601  
## [126] 0.15866848 -3.28199507 -0.67427140 -1.13081869 3.14423288  
## [131] -8.35129873 -3.01652882 1.15727320 -1.08989010 -0.22479737  
## [136] -1.73578920  
## [1] 0.069371558 0.069688116 0.069557838 0.080146879 0.078356527  
## [6] 0.072053426 0.090380378 0.077543886 0.059857217 0.076977433  
## [11] 0.080194037 0.082590712 0.073226524 0.072201671 0.048594953  
## [16] 0.080171892 0.034501707 0.046289664 0.043144619 0.084814409  
## [21] 0.050863879 0.069910661 0.094781580 0.056621718 0.047681585  
## [26] 0.052129872 0.061304448 0.091451989 0.105280208 0.079340377  
## [31] 0.037321374 0.060866067 0.055107242 0.076880830 0.042239975  
## [36] 0.099175749 0.095799847 0.053970497 0.096342174 0.047363703  
## [41] 0.076667091 0.053156542 0.077178028 0.068939546 0.064374158  
## [46] 0.017116403 0.092867125 0.072357113 0.043189648 0.097316203  
## [51] 0.087899165 0.089080997 0.019439401 0.074372058 0.078274922  
## [56] 0.059706476 0.096731739 0.102752550 0.036325157 0.040774443  
## [61] 0.071132028 0.083157999 0.066477393 0.064004933 0.107530757  
## [66] 0.036554122 0.088987002 0.008524082 0.095817143 0.077187673  
## [71] 0.057214617 0.079961665 0.097632751 0.136840731 0.136194446  
## [76] 0.052075387 0.117731703 0.053753679 -0.009967500 0.051992733  
## [81] 0.057954902 0.071847996 0.149679805 0.057459615 0.054866651  
## [86] 0.009197247 0.093667977 0.110113629 0.055594636 0.106520524  
## [91] 0.038115447 0.106943536 0.099037573 0.124784422 0.080386146  
## [96] 0.070074940 0.127888921 0.101686647 0.070174227 0.037415951  
## [101] 0.084551702 0.039054755 0.048849028 0.054017511 0.016436912  
## [106] 0.069233009 -0.020003696 -0.003111683 0.041821704 0.032544783  
## [111] 0.035613378 0.045141275 0.018331564 0.030893348 0.025597871  
## [116] 0.115918107 0.088604931 0.150484775 0.103779143 0.123356714  
## [121] 0.102231271 0.027458988 0.101351973 0.010381686 0.044538314  
## [126] 0.049198206 0.093581632 0.060411514 0.064607009 0.013105688  
## [131] 0.163656646 0.091494498 0.037330740 0.065417397 0.050550936  
## [136] 0.075723387  
## [1] "Fold 3"  
## [1] -1.383236 -1.987054 -2.007400  
## [1] 0.06937156 0.07744136 0.07780000  
## [1] NA  
## [1] NA

test = c(1,2,3,4,5,6,7)  
sd(test)/sqrt(length(test))

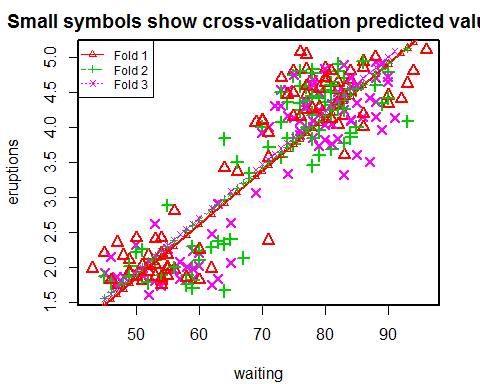
## [1] 0.8164966

sqrt(var(test))/sqrt(length(test))

## [1] 0.8164966

mydata<-as.data.frame(faithful)  
fit<-lm(eruptions~waiting,mydata)  
cv.lm(mydata, fit, m=3)

## Analysis of Variance Table  
##   
## Response: eruptions  
## Df Sum Sq Mean Sq F value Pr(>F)   
## waiting 1 286.5 286.5 1162 <2e-16 \*\*\*  
## Residuals 270 66.6 0.2   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1



##   
## fold 1   
## Observations in test set: 90   
## 2 11 16 18 21 22 26 27 32  
## waiting 54.00 54.000 52.00 84.000 51.00 47.000 83.000 55.000 77.000  
## cvpred 2.15 2.150 2.00 4.443 1.92 1.615 4.367 2.226 3.908  
## eruptions 1.80 1.833 2.17 4.800 1.80 1.750 3.600 1.967 4.467  
## CV residual -0.35 -0.317 0.17 0.357 -0.12 0.135 -0.767 -0.259 0.559  
## 33 36 38 49 51 53 62 65 66  
## waiting 66.00 52.0000 80.000 82.000 75.00 54.000 84.0000 60.000 92.000  
## cvpred 3.07 1.9968 4.137 4.290 3.76 2.150 4.4432 2.608 5.055  
## eruptions 3.37 2.0170 4.833 4.633 4.80 1.833 4.5000 1.817 4.400  
## CV residual 0.30 0.0202 0.696 0.343 1.04 -0.317 0.0568 -0.791 -0.655  
## 68 70 71 72 75 76 78 80 83  
## waiting 78.000 73.0 82.000 56.000 62.000 76.00 78.000 83.000 70.000  
## cvpred 3.984 3.6 4.290 2.303 2.761 3.83 3.984 4.367 3.373  
## eruptions 4.700 4.7 4.033 1.967 1.983 5.07 4.567 3.600 4.100  
## CV residual 0.716 1.1 -0.257 -0.336 -0.778 1.24 0.583 -0.767 0.727  
## 89 90 92 93 97 103 109 122  
## waiting 48.000 86.000 90.000 50.0000 84.000 49.000 86.000 69.000  
## cvpred 1.691 4.596 4.902 1.8439 4.443 1.767 4.596 3.296  
## eruptions 2.167 4.000 4.333 1.8670 4.667 2.100 4.850 4.067  
## CV residual 0.476 -0.596 -0.569 0.0231 0.224 0.333 0.254 0.771  
## 123 132 133 138 139 143 144 147 149  
## waiting 77.000 83.00 56.000 86.000 53.0000 82.000 77.000 80.000 96.000  
## cvpred 3.908 4.37 2.303 4.596 2.0732 4.290 3.908 4.137 5.361  
## eruptions 4.250 4.17 2.800 4.933 2.0330 4.533 4.817 4.633 5.100  
## CV residual 0.342 -0.20 0.497 0.337 -0.0402 0.243 0.909 0.496 -0.261  
## 151 155 161 168 169 170 171 178 181  
## waiting 77.00 71.000 45.000 88.000 52.0000 93.000 49.00 50.000 55.000  
## cvpred 3.91 3.449 1.462 4.749 1.9968 5.131 1.77 1.844 2.226  
## eruptions 5.03 3.567 2.200 5.000 1.9330 4.617 1.92 2.417 1.883  
## CV residual 1.12 0.118 0.738 0.251 -0.0638 -0.514 0.15 0.573 -0.343  
## 182 185 190 191 204 209 211 212 215  
## waiting 77.000 51.000 55.0000 81.000 53.000 49.000 71.00 80.000 64.000  
## cvpred 3.908 1.920 2.2261 4.214 2.073 1.767 3.45 4.137 2.914  
## eruptions 4.583 2.033 2.1830 4.800 1.867 1.933 2.38 4.700 3.417  
## CV residual 0.675 0.113 -0.0431 0.586 -0.206 0.166 -1.07 0.563 0.503  
## 217 218 219 220 221 223 224 226  
## waiting 53.000 94.000 55.000 76.000 50.0000 54.00 75.000 79.0000  
## cvpred 2.073 5.208 2.226 3.832 1.8439 2.15 3.755 4.0609  
## eruptions 2.400 4.800 2.000 4.150 1.8670 1.75 4.483 4.1170  
## CV residual 0.327 -0.408 -0.226 0.318 0.0231 -0.40 0.728 0.0561  
## 228 230 231 232 233 235 237 241 242  
## waiting 78.000 79.000 70.00 54.000 86.000 90.000 54.00 75.000 47.000  
## cvpred 3.984 4.061 3.37 2.150 4.596 4.902 2.15 3.755 1.615  
## eruptions 4.267 4.550 4.08 2.417 4.183 4.450 1.85 4.150 2.350  
## CV residual 0.283 0.489 0.71 0.267 -0.413 -0.452 -0.30 0.395 0.735  
## 248 251 257 258 260 263 265 266  
## waiting 82.0000 54.0000 71.000 83.0000 79.000 58.000 43.000 60.000  
## cvpred 4.2903 2.1497 3.449 4.3667 4.061 2.455 1.309 2.608  
## eruptions 4.3670 2.2000 3.917 4.4500 4.283 1.850 1.983 2.250  
## CV residual 0.0767 0.0503 0.468 0.0833 0.222 -0.605 0.674 -0.358  
## 268 271 272  
## waiting 81.0000 46.000 74.000  
## cvpred 4.2138 1.538 3.679  
## eruptions 4.1170 1.817 4.467  
## CV residual -0.0968 0.279 0.788  
##   
## Sum of squares = 23.6 Mean square = 0.26 n = 90   
##   
## fold 2   
## Observations in test set: 91   
## 1 4 5 6 12 13 15 23  
## waiting 79.000 62.000 8.50e+01 55.000 84.000 78.000 83.000 78.000  
## cvpred 4.087 2.825 4.53e+00 2.305 4.458 4.012 4.384 4.012  
## eruptions 3.600 2.283 4.53e+00 2.883 3.917 4.200 4.700 3.450  
## CV residual -0.487 -0.542 9.73e-04 0.578 -0.541 0.188 0.316 -0.562  
## 30 35 40 41 42 44 47 48 50  
## waiting 79.000 74.000 90.00 80.000 58.000 58.000 64.00 53.0000 59.000  
## cvpred 4.087 3.716 4.90 4.161 2.528 2.528 2.97 2.1567 2.602  
## eruptions 4.433 3.833 4.78 4.350 1.883 1.750 3.83 2.1000 2.000  
## CV residual 0.346 0.117 -0.12 0.189 -0.645 -0.778 0.86 -0.0567 -0.602  
## 57 58 59 60 63 64 67 82  
## waiting 71.000 64.00 77.000 81.0000 48.0000 82.000 78.000 82.0000  
## cvpred 3.493 2.97 3.938 4.2351 1.7856 4.309 4.012 4.3093  
## eruptions 3.717 1.67 4.567 4.3170 1.7500 4.800 4.167 4.3330  
## CV residual 0.224 -1.31 0.629 0.0819 -0.0356 0.491 0.155 0.0237  
## 85 88 94 98 99 100 108 110  
## waiting 73.000 80.000 78.000 75.0000 51.000 82.000 52.000 81.000  
## cvpred 3.641 4.161 4.012 3.7897 2.008 4.309 2.082 4.235  
## eruptions 4.067 4.517 4.817 3.7500 1.867 4.900 1.783 3.683  
## CV residual 0.426 0.356 0.805 -0.0397 -0.141 0.591 -0.299 -0.552  
## 111 112 114 115 116 119 125 131 134  
## waiting 75.000 59.000 79.00 59.000 81.000 59.000 88.000 45.000 89.000  
## cvpred 3.790 2.602 4.09 2.602 4.235 2.602 4.755 1.563 4.829  
## eruptions 4.733 2.300 4.42 1.700 4.633 1.817 4.600 1.867 4.333  
## CV residual 0.943 -0.302 0.33 -0.902 0.398 -0.785 -0.155 0.304 -0.496  
## 136 141 142 145 148 152 153 154  
## waiting 82.0000 81.00000 60.000 76.000 49.000 77.0000 65.000 81.000  
## cvpred 4.3093 4.23511 2.676 3.864 1.860 3.9382 3.047 4.235  
## eruptions 4.3830 4.23300 2.233 4.333 2.017 4.0000 2.400 4.600  
## CV residual 0.0737 -0.00211 -0.443 0.469 0.157 0.0618 -0.647 0.365  
## 156 157 158 164 165 166 167 173 174  
## waiting 70.000 81.000 93.00 78.000 66.000 76.000 63.000 77.000 68.0000  
## cvpred 3.419 4.235 5.13 4.012 3.122 3.864 2.899 3.938 3.2701  
## eruptions 4.000 4.500 4.08 3.833 3.500 4.583 2.367 4.583 3.3330  
## CV residual 0.581 0.265 -1.04 -0.179 0.378 0.719 -0.532 0.645 0.0629  
## 175 177 179 184 186 193 195 196  
## waiting 81.0000 73.000 85.000 83.000 78.000 76.000 77.0000 81.00000  
## cvpred 4.2351 3.641 4.532 4.384 4.012 3.864 3.9382 4.23511  
## eruptions 4.1670 4.500 4.000 3.767 4.433 4.800 3.9660 4.23300  
## CV residual -0.0681 0.859 -0.532 -0.617 0.421 0.936 0.0278 -0.00211  
## 199 201 206 207 208 213 214 216  
## waiting 51.000 60.000 46.000 77.000 84.000 49.00000 75.0000 76.000  
## cvpred 2.008 2.676 1.637 3.938 4.458 1.85979 3.7897 3.864  
## eruptions 2.250 2.100 1.783 4.367 3.850 1.86700 3.8330 4.233  
## CV residual 0.242 -0.576 0.146 0.429 -0.608 0.00721 0.0433 0.369  
## 222 234 236 239 240 243 249 250 252  
## waiting 82.0000 50.000 54.000 79.000 64.00 86.000 67.00 74.000 83.0000  
## cvpred 4.3093 1.934 2.231 4.087 2.97 4.606 3.20 3.716 4.3836  
## eruptions 4.2670 2.217 1.883 3.950 2.33 4.933 2.13 4.350 4.4500  
## CV residual -0.0423 0.283 -0.348 -0.137 -0.64 0.327 -1.06 0.634 0.0664  
## 253 254 259 261 262 264 270  
## waiting 73.0000 73.000 56.000 78.000 84.0000 83.000 90.000  
## cvpred 3.6413 3.641 2.379 4.012 4.4578 4.384 4.903  
## eruptions 3.5670 4.500 2.000 4.767 4.5330 4.250 4.417  
## CV residual -0.0743 0.859 -0.379 0.755 0.0752 -0.134 -0.486  
##   
## Sum of squares = 22.6 Mean square = 0.25 n = 91   
##   
## fold 3   
## Observations in test set: 91   
## 3 7 8 9 10 14 17 19 20  
## waiting 74.000 88.00 85.00 51.0000 85.000 47.0000 62.00 52.000 79.0000  
## cvpred 3.774 4.85 4.62 2.0067 4.619 1.6993 2.85 2.084 4.1582  
## eruptions 3.333 4.70 3.60 1.9500 4.350 1.7500 1.75 1.600 4.2500  
## CV residual -0.441 -0.15 -1.02 -0.0567 -0.269 0.0507 -1.10 -0.484 0.0918  
## 24 25 28 29 31 34 37 39  
## waiting 69.000 74.000 76.000 78.000 73.000 80.000 48.0000 59.000  
## cvpred 3.390 3.774 3.928 4.081 3.697 4.235 1.7762 2.621  
## eruptions 3.067 4.533 4.083 3.850 4.300 4.033 1.8670 1.833  
## CV residual -0.323 0.759 0.155 -0.231 0.603 -0.202 0.0908 -0.788  
## 43 45 46 52 54 55 56 61 69  
## waiting 84.0000 73.000 83.00 90.000 80.000 54.000 83.000 59.000 65.00  
## cvpred 4.5424 3.697 4.47 5.003 4.235 2.237 4.466 2.621 3.08  
## eruptions 4.5670 4.533 3.32 4.716 4.833 1.733 4.883 2.233 2.07  
## CV residual 0.0246 0.836 -1.15 -0.287 0.598 -0.504 0.417 -0.388 -1.02  
## 73 74 77 79 81 84 86 87  
## waiting 79.000 71.000 60.000 76.0000 75.000 65.000 88.0000 76.0000  
## cvpred 4.158 3.543 2.698 3.9277 3.851 3.082 4.8498 3.9277  
## eruptions 4.500 4.000 2.017 3.8830 4.133 2.633 4.9330 3.9500  
## CV residual 0.342 0.457 -0.681 -0.0447 0.282 -0.449 0.0832 0.0223  
## 91 95 96 101 102 104 105 106 107  
## waiting 60.000 63.00 72.00 62.000 88.000 83.0000 81.000 47.000 84.000  
## cvpred 2.698 2.93 3.62 2.852 4.850 4.4656 4.312 1.699 4.542  
## eruptions 2.200 1.83 4.30 2.483 4.367 4.5000 4.050 1.867 4.700  
## CV residual -0.498 -1.10 0.68 -0.369 -0.483 0.0344 -0.262 0.168 0.158  
## 113 117 118 120 121 124 126 127  
## waiting 89.0000 50.000 85.0000 87.000 53.000 56.000 81.000 45.000  
## cvpred 4.9266 1.930 4.6193 4.773 2.160 2.391 4.312 1.546  
## eruptions 4.9000 2.317 4.6000 4.417 2.617 1.967 3.767 1.917  
## CV residual -0.0266 0.387 -0.0193 -0.356 0.457 -0.424 -0.545 0.371  
## 128 129 130 135 137 140 146 150 159  
## waiting 82.000 55.0000 90.000 46.000 51.000 79.000 59.000 53.00 53.00  
## cvpred 4.389 2.3141 5.003 1.622 2.007 4.158 2.621 2.16 2.16  
## eruptions 4.500 2.2670 4.650 1.833 1.883 3.733 1.983 1.80 1.80  
## CV residual 0.111 -0.0471 -0.353 0.211 -0.124 -0.425 -0.638 -0.36 -0.36  
## 160 162 163 172 176 180 183 187 188  
## waiting 89.00 86.000 58.000 57.000 81.0000 74.000 83.000 84.000 46.000  
## cvpred 4.93 4.696 2.545 2.468 4.3119 3.774 4.466 4.542 1.622  
## eruptions 3.97 4.150 2.000 2.083 4.3330 4.167 4.250 4.083 1.833  
## CV residual -0.96 -0.546 -0.545 -0.385 0.0211 0.393 -0.216 -0.459 0.211  
## 189 192 194 197 198 200 202 203  
## waiting 83.0000 57.000 84.000 87.00 77.000 78.000 82.0000 91.000  
## cvpred 4.4656 2.468 4.542 4.77 4.005 4.081 4.3887 5.080  
## eruptions 4.4170 1.833 4.100 3.50 4.366 4.667 4.3500 4.133  
## CV residual -0.0486 -0.635 -0.442 -1.27 0.361 0.586 -0.0387 -0.947  
## 205 210 225 227 229 238 244 245  
## waiting 78.000 83.0000 78.0000 78.00000 70.00 77.000 63.0000 85.0000  
## cvpred 4.081 4.4656 4.0814 4.08137 3.47 4.005 2.9288 4.6193  
## eruptions 4.600 4.5000 4.0000 4.08300 3.92 4.283 2.9000 4.5830  
## CV residual 0.519 0.0344 -0.0814 0.00163 0.45 0.278 -0.0288 -0.0363  
## 246 247 255 256 267 269  
## waiting 82.000 57.000 88.00 80.000 75.000 46.000  
## cvpred 4.389 2.468 4.85 4.235 3.851 1.622  
## eruptions 3.833 2.083 4.15 3.817 4.750 2.150  
## CV residual -0.556 -0.385 -0.70 -0.418 0.899 0.528  
##   
## Sum of squares = 22.8 Mean square = 0.25 n = 91   
##   
## Overall (Sum over all 91 folds)   
## ms   
## 0.253