Lab 3

Attila Lazar

28.10.2020

data

We load the dataset

```
load("data/dat.RData")
data <- d[, names(d) %in% c('y', paste0('X', 20:65))]
set.seed(123)
n <- nrow(data)
```

1)

a)

We train a model on the training dataset using all variables except "X61"

```
train <- sample(1:n, round(n*2/3))
test <- (1:n) [-train]
xnames <- paste0('X', setdiff(20:65, 61))
full_formula <- as.formula(paste('y~', paste(xnames, collapse="+")))
model1 <- lm(full_formula, data, subset=train)
```

For calculating the trimmed MSE we use the following function. The funcion trimms only the high values of the given vector, thus is more suitable for calculating trimmed MSEs than the "mean" function.

```
rtmean <- function(x,trim = 0) {
  x <- sort(x)
  v <- x[1:floor(length(x)*(1-trim))]
  mean(v)
}
```

MSE train

```
rtmean((data[train, 'y'] - predict(model1, data[train,]))^2)
```

```
## [1] 0.2787313
```

MSE train right-trimmed by 10%

```
rtmean((data[train, 'y'] - predict(model1, data[train,]))^2, trim=0.1)
```

```
## [1] 0.1163575
```

MSE test

```
rtmean((data[test, 'y'] - predict(model1, data[test,]))^2)
```

```
## [1] 0.9699528
```

MSE test right-trimmed by 10%

```
rtmean((data[test, 'y'] - predict(model1, data[test,]))^2, trim=0.1)
```

```
## [1] 0.1230137
```

Since the trimmed mean leaves out the worst errors the trimmed MSE is much better then the untrimmed. MSE values are also always worse when calculated with the test-set.

b)

We plot the calculated MSEs in one parallel boxplot. We also add one more box to the plot for the "leave-one-out" calculation.

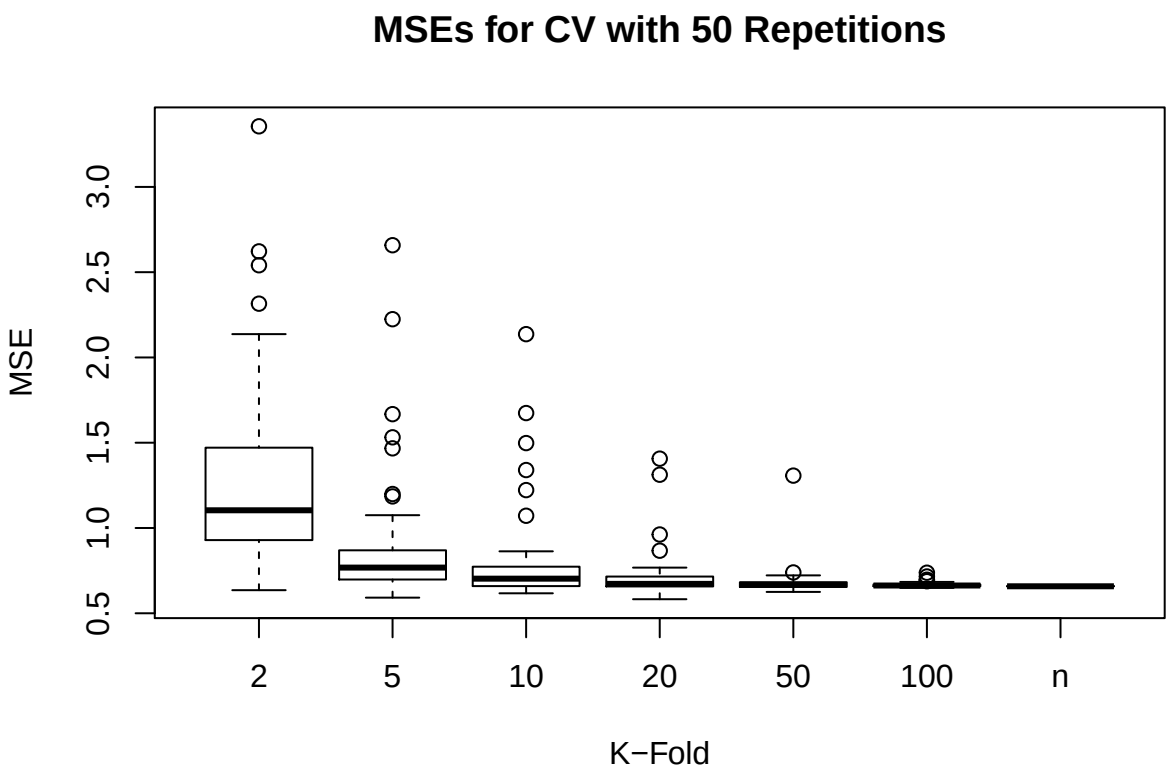
```
library(cvTools)
```

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

```
## Loading required package: robustbase
```

```
mses <- matrix(ncol=50, nrow=6)
i <- 1
for (k in c(2,5,10,20,50,100)) {
  #print(k)
  ret <- cvFit(lm, formula=full_formula, data=data, y=data$y, K=k, R=50, cost=mspe, seed = 123)
  mses[i,] <- ret$reps
  i <- i + 1
}

loo <- cvFit(lm, formula=full_formula, data=data, y=data$y, type="leave-one-out", K=n, cost=mspe, seed =
boxplot(mses[1,], mses[2,], mses[3,], mses[4,], mses[5,], mses[6,], loo$cv, names=c("2","5","10","20","
```



The median MSE is decreasing till k=20, after that it becomes more cosistent between repetitions. This is

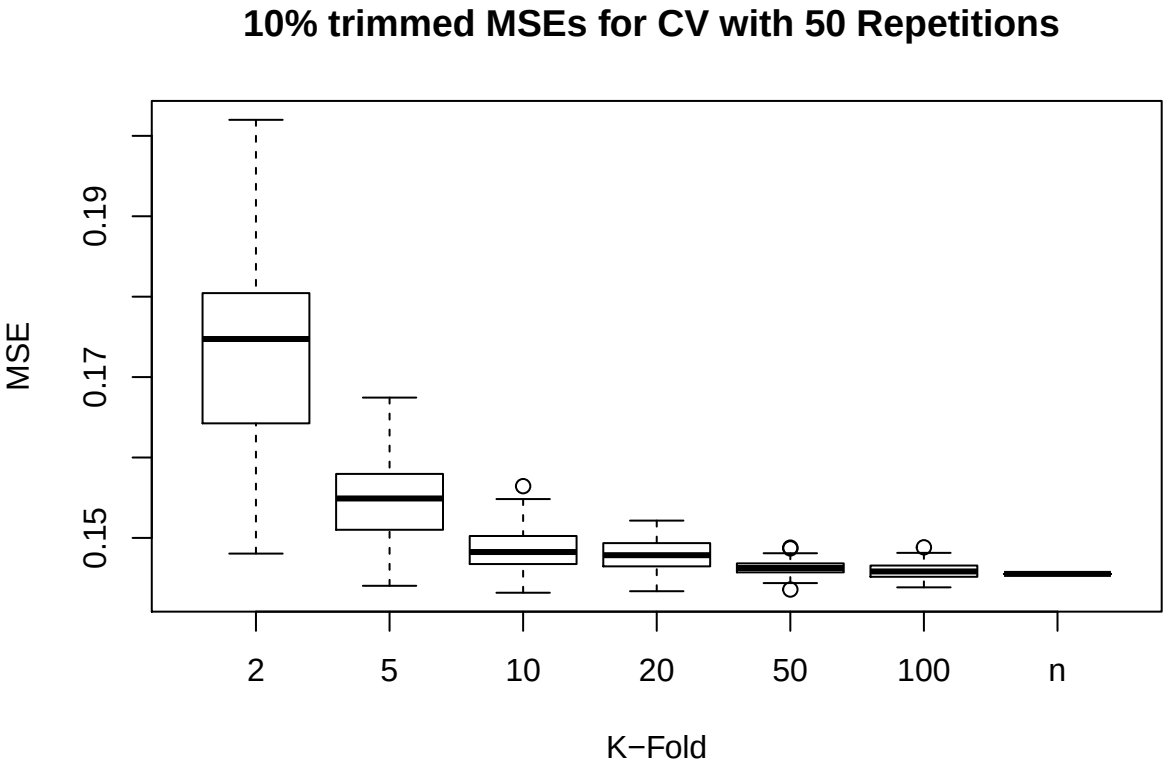
because of smaller test samples from which the MSE is calculated. Leave-one-out (K=n) CV yields simmular MSE then 20 Fold CV.

c)

To calculate trimmed MSEs we replace the cost function "mspe" to the trimmed cost function "tmspe"

```
mses <- matrix(ncol=50, nrow=6)
i <- 1
for (k in c(2,5,10,20,50,100)) {
  ret <- cvFit(lm, formula=full_formula, data=data, y=data$y, K=k, R=50, cost=tmspe, costArgs = list(tr
  mses[i,] <- ret$reps
  i <- i + 1
}

loo <- cvFit(lm, formula=full_formula, data=data, y=data$y, type="leave-one-out", K=n, cost=tmspe, cost
boxplot(mses[1,], mses[2,], mses[3,], mses[4,], mses[5,], mses[6,], loo$cv, names=c("2","5","10","20","
```



The trimmed MSEs are much smaller. Since we trimmed the MSE values there are also no outliers.

d)

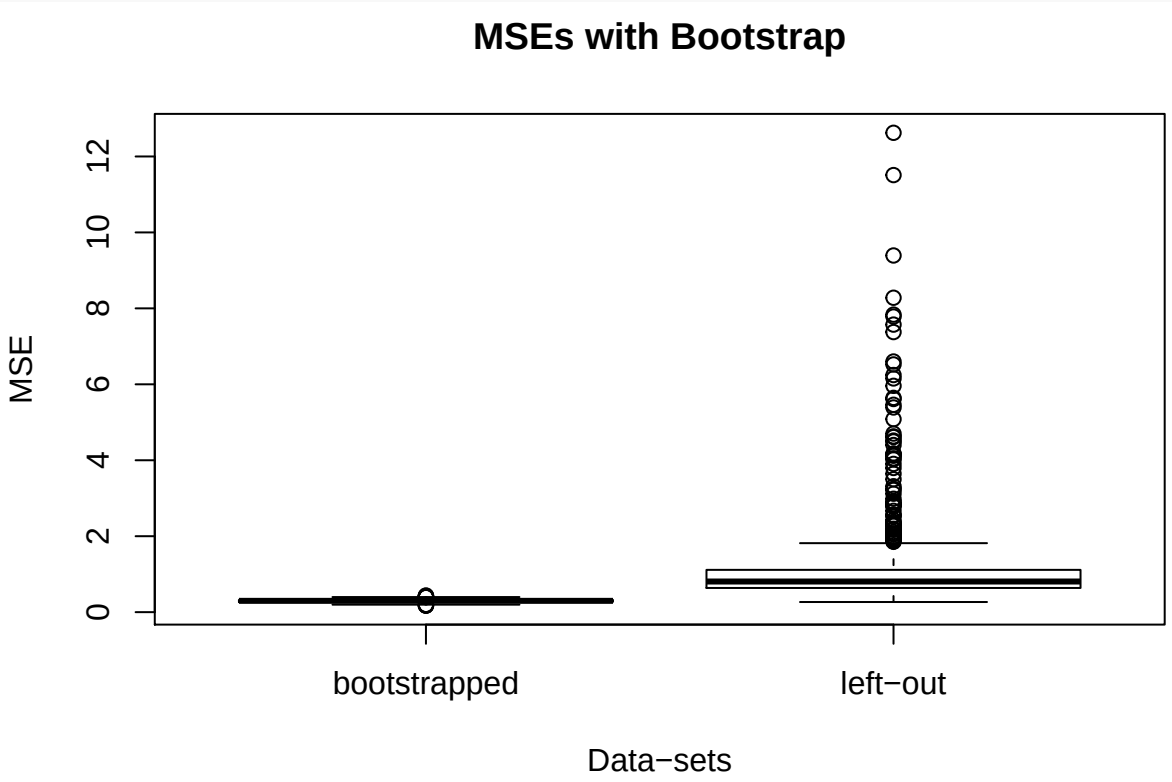
For bootrapping and calculating MSEs, we use the following function

```
bootsrapf <- function(formula, d, n=1000, trim=0) {
  #print(formula)
  mse_train <- vector() #mse-s on bootraped data
  mse_test <- vector() #mse-s on left-out data
  ltest <- vector() #nr of left-out data samples
```

```
  resp <- as.character(formula[[2]])  # extract response variable from formula
  for (m in c(1:n)) {
    bsrp <- sample(1:nrow(d), replace=TRUE)
    #print(bsrp)
    model2 <- lm(formula, data=d[bsrp,])
    mse_train <- append(mse_train, rtmean((d[bsrp, resp] - predict(model2, d[bsrp,]))^2, trim=trim))
    mse_test <- append(mse_test, rtmean((d[-bsrp, resp] - predict(model2, d[-bsrp,]))^2, trim=trim))
    ltest <- append(ltest, nrow(d[-bsrp,]))
  }
  retval <- list("mse_train" = mse_train, "mse_test" = mse_test, "ltest" = ltest)
  return(retval)
}
```

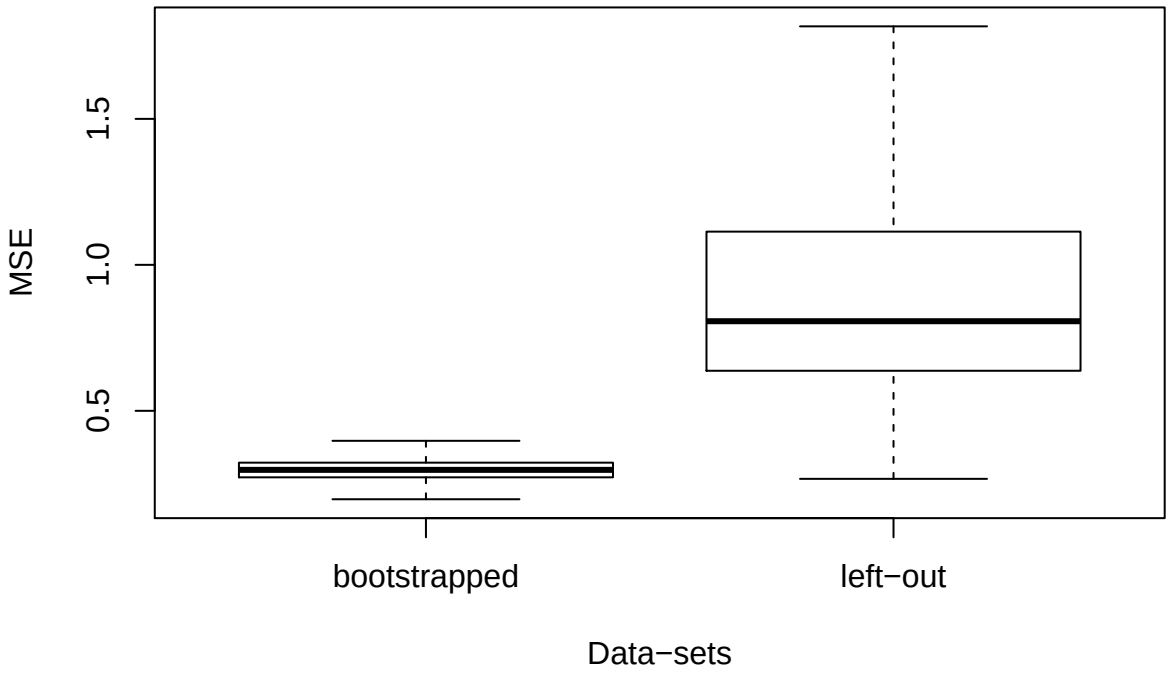
We bootrap 1000 data-sets and plot the resulting MSEs on the bootrapped and left-out observations

```
bs <- bootsrapf(full_formula, data)
boxplot(bs$mse_train, bs$mse_test, names=c("bootstrapped", "left-out"), xlab="Data-sets", ylab="MSE", ma
```



The MSE values have a big variance with maximum values reaching to 20. In order to compare MSE values, we plot them again without outliers.

```
boxplot(bs$mse_train, bs$mse_test, names=c("bootstrapped", "left-out"), outline=FALSE, xlab="Data-sets"
```



The median MSE is much smaller for the bootsrapped data-set, becouse this is our training dataset. MSE of the left-out samples is more representative of the loss.

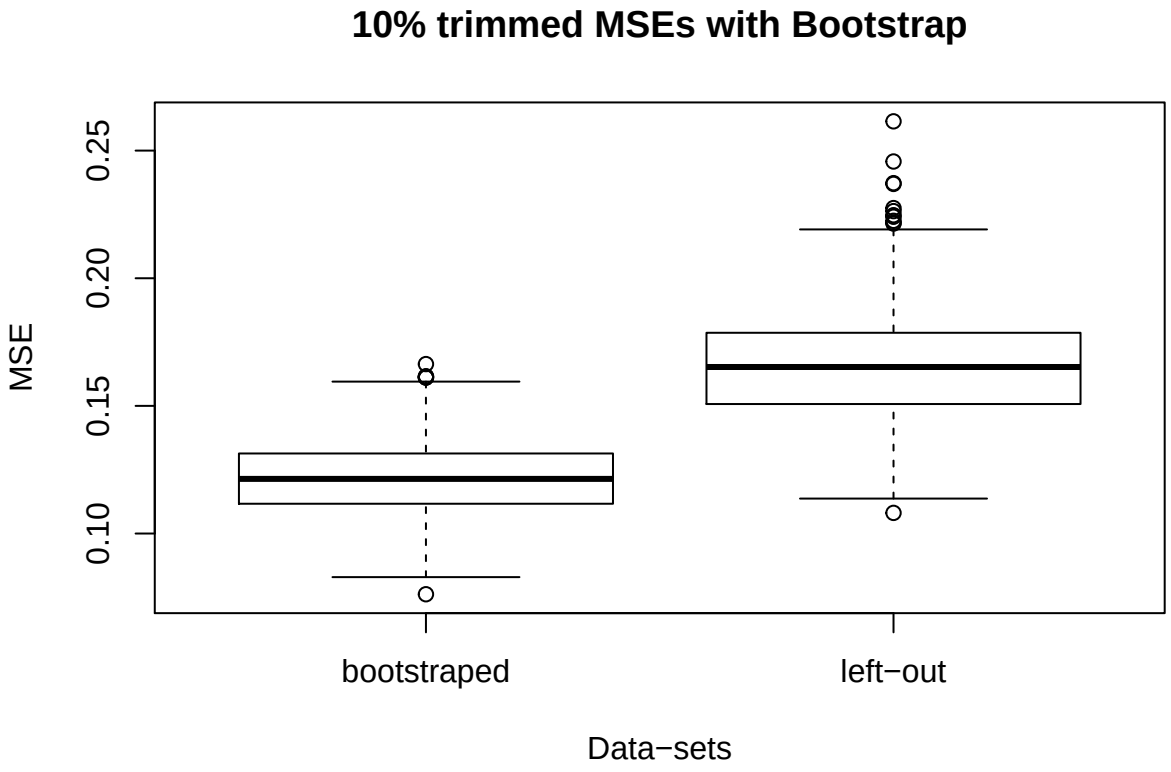
The median MSE for the left-out samples is under 1.0 which is much more than the median MSE calculated By CV with K=10 around 0.5 . This is because the bootrapped trainin-set contains some values multiple times which gives this samples more weight.

Also there is a big difference between the MSE calculated from bootstrapped and left-out samples.

e)

For the 10% trimmed MSE we use the "rtmean" function from ex 1a, which only trimms the upper side of the vector.

```
bs2 <- bootsrapf(full_formula, data, trim=0.1)
boxplot(bs2$mse_train, bs2$mse_test, names=c("bootstraped", "left-out"), xlab="Data-sets", ylab="MSE", r
```



As expected the MSEs are much better.

2)

In Lab2 we selected the following model using

Stepwise Variable Selection

```
stepwise <- as.formula("y ~ X36 + X64 + X54 + X21 + X37 + X63 + X32 + X45 +
    X26 + X35 + X22 + X29 + X65 + X34 + X23 + X48 + X28 + X53 +
    X57 + X62 + X55 + X58")
stepwise
```

```
## y ~ X36 + X64 + X54 + X21 + X37 + X63 + X32 + X45 + X26 + X35 +
##     X22 + X29 + X65 + X34 + X23 + X48 + X28 + X53 + X57 + X62 +
##     X55 + X58
```

Best Subset Regression

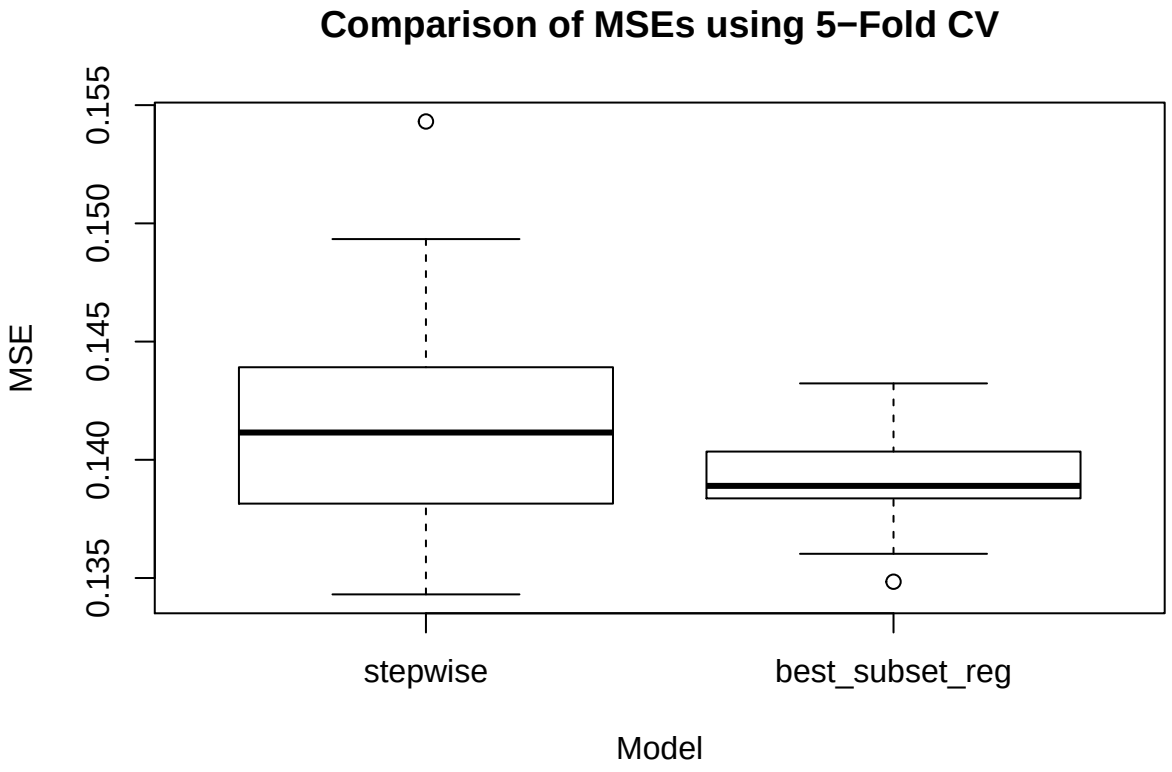
```
best_subset_reg <- as.formula("y~X35+X36+X37+X46+X48+X52")
best_subset_reg
```

```
## y ~ X35 + X36 + X37 + X46 + X48 + X52
```

a)

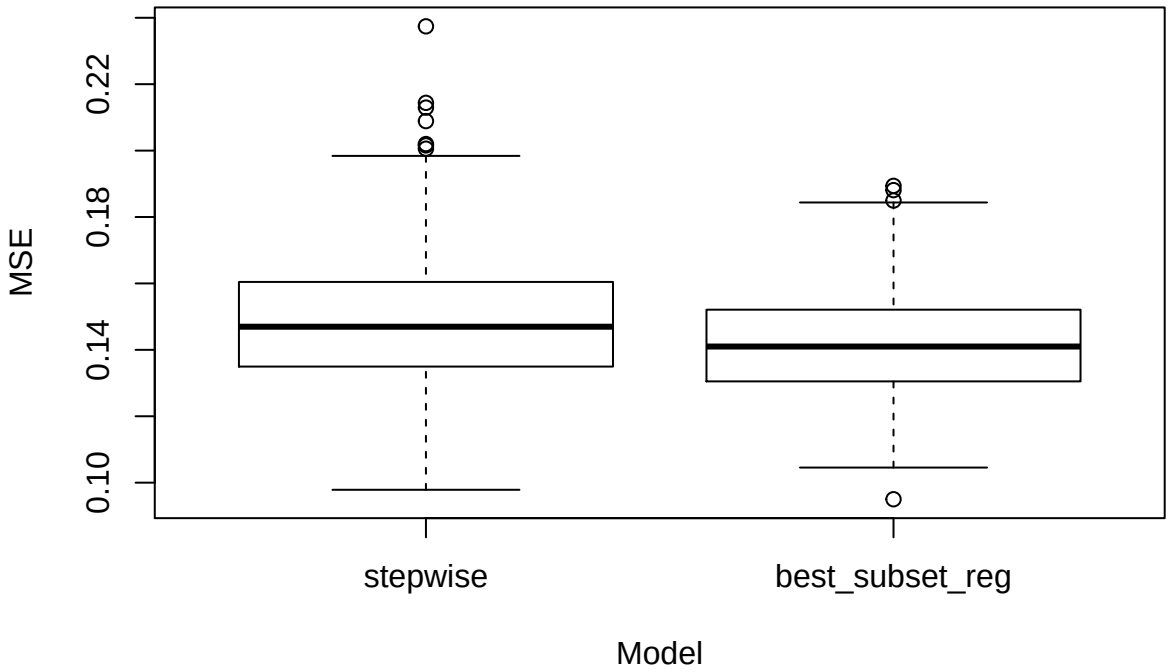
```
cv1 <- cvFit(lm, formula=stepwise, data=data, y=Y, K=5, R=50, cost=tmspe, costArgs = list(trim = 0.1))
cv2 <- cvFit(lm, formula=best_subset_reg, data=data, y=Y, K=5, R=50, cost=tmspe, costArgs = list(trim =

boxplot(cv1$reps[,1], cv2$reps[,1], names=c("stepwise", "best_subset_reg"), xlab="Model", ylab="MSE", ma
```



Compared to the trimmed MSEs calculated using 10-Fold CV, this models are slightly better but much smaller. Our smallest model "best_subset_reg" performs best. Because it contains fewer variables it is less prone to overfitting.

```
bs1 <- bootsrapf(stepwise, data, trim=0.1)
bs2 <- bootsrapf(best_subset_reg, data, trim=0.1)
boxplot(bs1$mse_test, bs2$mse_test, names=c("stepwise", "best_subset_reg"), xlab="Model", ylab="MSE", ma
```



Using bootrap method for MSE calculation yields simular MSEs for both models with the smaller model beeing slightly better.