finalproject2.5

Haoran Tian

11/25/2020

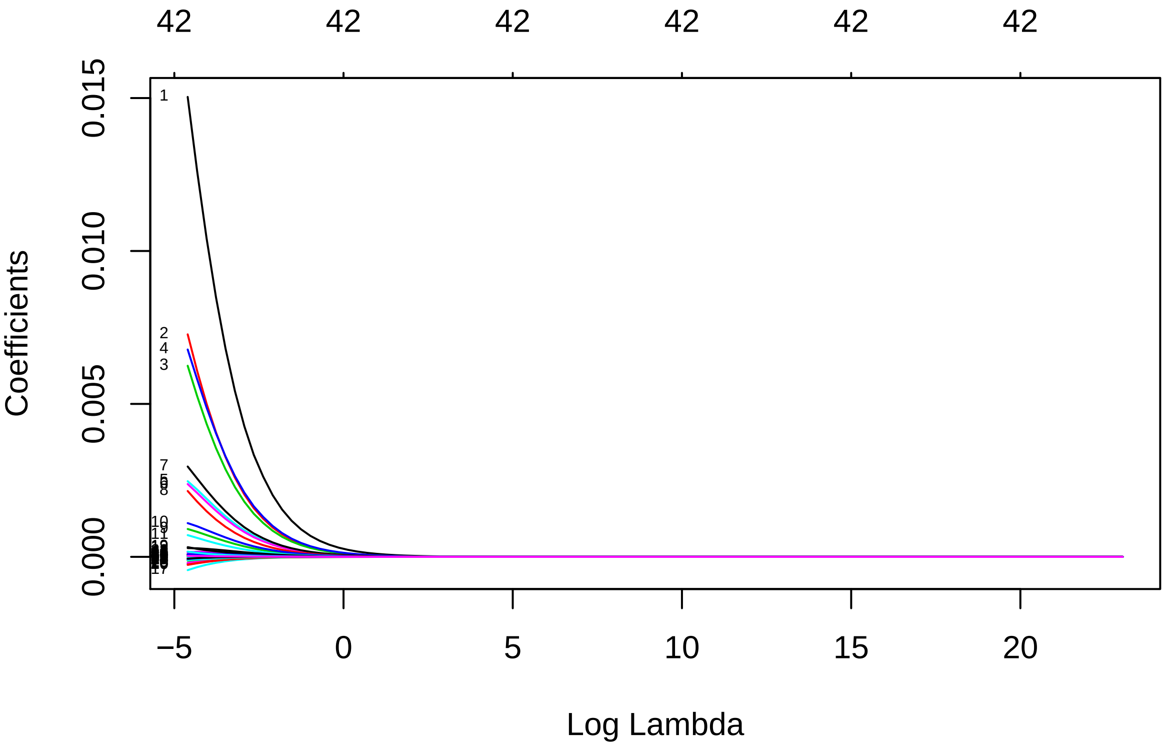
##2.5 Ridge and LASSO

```
#final solution for calculating backward returns

bw_full <- read.csv("final_project-1.csv")
time.horizons = c(3,10,30,60,120,180,240,360,480,600,720,960,1200,1440)

for (time in time.horizons) {
  bw_full = mutate(bw_full, "Asset_1_BRet_{time}" := (bw_full$Asset_1 - ifelse(X > (time-1), lag(bw_ful
  bw_full = mutate(bw_full, "Asset_2_BRet_{time}" := (bw_full$Asset_2 - ifelse(X > (time-1), lag(bw_ful
  bw_full = mutate(bw_full, "Asset_3_BRet_{time}" := (bw_full$Asset_3 - ifelse(X > (time-1), lag(bw_ful
}

bw_full <- bw_full[-c(1:4)]
```

The backward returns in required time horizons are included in bw_full file.

#Ridge

```
#Calculate forward returns and add the new variable
Asset_1_lead <- lead(Asset_1, n=10, default=tail(Asset_1, 1))
Asset_1_BRet_10_forward <- (Asset_1_lead - Asset_1) / Asset_1
colnames(Asset_1_BRet_10_forward) <- c("Asset_1_BRet_10_forward")
bw_full <- cbind(bw_full, Asset_1_BRet_10_forward)


train_size <- floor(nrow(bw_full) * 0.7)

data_ridge_train <- bw_full[1: train_size,]
data_ridge_test <- bw_full[-(1:train_size),]

set.seed(42)
train_id <- sample(1:nrow(bw_full), size = trunc(nrow(bw_full)))

X <- model.matrix(Asset_1_BRet_10_forward ~ ., data = bw_full)[, -1]
X_train <- X[1: train_size,]
X_test <- X[-(1: train_size),]

grid = 10^seq(10, -2, length = 100)
ridge.mod = glmnet(x = X_train, y = data_ridge_train$Asset_1_BRet_10_forward, alpha = 0, lambda = grid)
plot(ridge.mod, xvar = "lambda",  label = TRUE )
```



```
set.seed(1)
cv.out = cv.glmnet(x = X_train, y = data_ridge_train$Asset_1_BRet_10_forward, alpha = 0, lambda = grid)
bestlam = cv.out$lambda.min
bestlam
```

```
## [1] 0.01
```

```
ridge.pred_train = predict(ridge.mod, s = bestlam, newx = X_train)
mean((ridge.pred_train - data_ridge_train$Asset_1_BRet_10_forward)^2)
```

```
## [1] 8.312882e-06
```

```
ridge.pred_test = predict(ridge.mod, s = bestlam, newx = X_test)
mean((ridge.pred_test - data_ridge_test$Asset_1_BRet_10_forward)^2)
```

```
## [1] 7.416932e-06
```

```
pred_ridge <- predict(ridge.mod, s = bestlam, newx =  X)



sprintf("Forward return for rf (t, 10) of Asset 1 are included in pred_ridge. The best tuning parameter
```

```
## [1] "Forward return for rf (t, 10) of Asset 1 are included in pred_ridge. The best tuning parameter :
```

```
#In-sample correlation
cor(as.matrix(cbind(ridge.pred_train, data_ridge_train$Asset_1_BRet_10_forward)))
```

```
##              1
## 1 1.00000000 0.07102249
##   0.07102249 1.00000000
```

```
sprintf("The in-sample correlation is %s",
cor(as.matrix(cbind(ridge.pred_train, data_ridge_train$Asset_1_BRet_10_forward)))[[2,1]])
```

```
## [1] "The in-sample correlation is 0.0710224930532306"
```

```
#Out-of-sample correlation
cor(as.matrix(cbind(ridge.pred_test, data_ridge_test$Asset_1_BRet_10_forward)))
```

```
##             1
## 1 1.00000000 0.03644832
##   0.03644832 1.00000000
```

```
sprintf("The out-of-sample correlation is %s",
cor(as.matrix(cbind(ridge.pred_test, data_ridge_test$Asset_1_BRet_10_forward)))[[2,1]])
```

```
## [1] "The out-of-sample correlation is 0.0364483215895639"
```

Lasso

```
lasso.mod = glmnet(x = X_train, y = data_ridge_train$Asset_1_BRet_10_forward, alpha = 1, lambda = grid)
par(mfrow = c(1,2))
plot(lasso.mod)
```

```
## Warning in plotCoef(x$beta, lambda = x$lambda, df = x$df, dev =
## x$dev.ratio, : No plot produced since all coefficients zero
```

```
## NULL
```

```
plot(lasso.mod, xvar = "lambda", label = TRUE)
```

```
## Warning in plotCoef(x$beta, lambda = x$lambda, df = x$df, dev =
## x$dev.ratio, : No plot produced since all coefficients zero
```

```
## NULL
```

```
set.seed(1)
cv.out_lasso = cv.glmnet(x = X_train, y = data_ridge_train$Asset_1_BRet_10_forward, alpha = 1, lambda =
bestlam_lasso = cv.out_lasso$lambda.min

lasso.pred_train = predict(lasso.mod, s = bestlam_lasso, newx = X_train)
mean((lasso.pred_train - data_ridge_train$Asset_1_BRet_10_forward)^2)
```

```
## [1] 8.342602e-06
```

```
lasso.pred_test = predict(lasso.mod, s = bestlam_lasso, newx = X_test)
mean((lasso.pred_test - data_ridge_test$Asset_1_BRet_10_forward)^2)
```

```
## [1] 7.426541e-06
```

```
pred_lasso <- predict(lasso.mod, s = bestlam, newx =  X)

sprintf("Forward return for rf (t, 10) of Asset 1 are included in pred_lasso. The best tuning parameter
```

```
## [1] "Forward return for rf (t, 10) of Asset 1 are included in pred_lasso. The best tuning parameter
```

```
#In-sample correlation
cor(as.matrix(cbind(lasso.pred_train , data_ridge_train$Asset_1_BRet_10_forward)))
```

```
## Warning in cor(as.matrix(cbind(lasso.pred_train,
## data_ridge_train$Asset_1_BRet_10_forward))): the standard deviation is zero
```

```
##    1
## 1  1 NA
##   NA  1
```

```r
sprintf("The in-sample correlation is %s",
cor(as.matrix(cbind(lasso.pred_train , data_ridge_train$Asset_1_BRet_10_forward)))[[2,1]])
```

```
## Warning in cor(as.matrix(cbind(lasso.pred_train,
## data_ridge_train$Asset_1_BRet_10_forward))): the standard deviation is zero

## [1] "The in-sample correlation is NA"
```

```r
#Out-of-sample correlation
cor(as.matrix(cbind(lasso.pred_test, data_ridge_test$Asset_1_BRet_10_forward)))
```

```
## Warning in cor(as.matrix(cbind(lasso.pred_test,
## data_ridge_test$Asset_1_BRet_10_forward))): the standard deviation is zero

##    1
## 1  1 NA
##   NA  1
```

```r
sprintf("The out-of-sample correlation is %s",
cor(as.matrix(cbind(lasso.pred_test, data_ridge_test$Asset_1_BRet_10_forward)))[[2,1]])
```

```
## Warning in cor(as.matrix(cbind(lasso.pred_test,
## data_ridge_test$Asset_1_BRet_10_forward))): the standard deviation is zero

## [1] "The out-of-sample correlation is NA"
```

Both in-sample and out- of-sample correlation are not available since the standard deviation of lasso prediction is zero, because lasso shrink all coefficients to almost zero as evidenced in high lambda_lasso.