STATS 415 Final Project

Group 15

```
library(tidyverse)

## ─ Attaching packages ─────────────────────────────── tidyverse 1.2.1 ─

## ✔ ggplot2 3.2.1     ✔ purrr   0.3.3
## ✔ tibble  2.1.3     ✔ dplyr   0.8.3
## ✔ tidyr   1.0.0     ✔ stringr 1.4.0
## ✔ readr   1.3.1     ✔ forcats 0.4.0

## ─ Conflicts ────────────────────────────────── tidyverse_conflicts() ─
## ✖ dplyr::filter() masks stats::filter()
## ✖ dplyr::lag()    masks stats::lag()

library(FNN)
library(parallel)
library(glmnet)

## Loading required package: Matrix

##
## Attaching package: 'Matrix'

## The following objects are masked from 'package:tidyr':
##
##     expand, pack, unpack

## Loaded glmnet 3.0-2

library(pls)

##
## Attaching package: 'pls'

## The following object is masked from 'package:stats':
##
##     loadings

#load dataset
dat = read.csv("/Users/1604091407/Downloads/final_project.csv")
```

2.1

```
# use blank dataframe to b joined
dat_blank = data.frame(rep(1:nrow(dat),1))

# h index
h = c(3,10,30)
```

```r
#use for loop to create and join 9 columns to dat_blank
for(i in 1:3){
  for(j in 1:3){
  dat_col = dat[,i+1]
  dat_temp = dat %>% mutate(shift = lag(dat_col,h[j])) %>% mutate(temp =
ifelse(is.na(shift),dat[1,i+1],shift) , last = (dat[,i+1] - temp) / temp)
  dat_blank = cbind(dat_blank, dat_temp$last)
  }
}
#drop the first column
dat_blank = dat_blank[,-1]

#rename the column
name_first = rep(NA,9)
for(i in 1:3){
  for(j in 1:3){
    name_first[j +(i-1)*3] = paste0("Asset_",i,"_BRet_",h[j])
  }
}
colnames(dat_blank) = name_first
```

```r
#export csv
dat_blank = dat_blank %>%  round(4)
dat_2.1 = write.csv(dat_blank,"/Users/l604091407/Desktop/stats 415
hw/final_project/bret.csv",row.names=FALSE)
```

#2.2

```r
dat_cor = dat_blank[,c(1,4,7)]
left = nrow(dat_cor) - 21*24*60
first_index = c(rep(1,21*24*60),(1:left))
second_index = 1:nrow(dat_blank)



Rho_1_2 = rep(NA, 524160)
Rho_2_3 = rep(NA, 524160)
Rho_1_3 = rep(NA, 524160)

for(i in 1:nrow(dat_cor)){
  temp_dat_1 = dat_cor$Asset_1_BRet_3[first_index[i]:second_index[i]]
  temp_dat_2 = dat_cor$Asset_2_BRet_3[first_index[i]:second_index[i]]
  temp_dat_3 = dat_cor$Asset_3_BRet_3[first_index[i]:second_index[i]]

  Rho_1_2[i] = cor(temp_dat_1,temp_dat_2)
  Rho_1_3[i] = cor(temp_dat_1,temp_dat_3)
```

```
  Rho_2_3[i] = cor(temp_dat_2,temp_dat_3)
}
```

```
corr_dat = corr_dat %>% round(4)
corr_dat[is.na(corr_dat)] = 0
corr_dat
write_csv(corr_dat,"/Users/1604091407/Desktop/stats 415
hw/final_project/corr.csv")

#fill NA with value of 0
corr_dat[is.na(corr_dat)] = 0
```

2.3

Remark: As we can see from the summary of linear regression below, all the backward returns of Asset 2 and Asset 3 are significant in predicting the forward return of Asset 1 except the 30-min backward return of asset 3.

The in-sample correlation between $\hat{r}_f\,(t,10)$ and $r_f(t,10)$ is 0.0678748, and the out-of-sample correlation between $\hat{r}_f\,(t,10)$ and $r_f(t,10)$ is 0.04067645.

#It seems that the correlation structure is stable over the year. However, there is some fluctuations in the beginning of the year

```
#dataset of forward return with asset_1 and h = 10
index_forward = c(11:524160,rep(524160,10))

dat_lead = dat %>% mutate(rf_10 = (Asset_1[index_forward] - Asset_1)/Asset_1)

#combine the rf_10 and the backward returns in 2.1
dat_forward = cbind(rf_10 = dat_lead$rf_10 ,dat_blank)

train_set = dat_forward[1:366912,]
test_set =  dat_forward[366913:524160,]
```

```
#fit the training dataset with linear regression
lm_mod = lm(rf_10~.,data = train_set)

summary(lm_mod)

##
## Call:
## lm(formula = rf_10 ~ ., data = train_set)
```

```
##
## Residuals:
##       Min        1Q    Median        3Q       Max
## -0.147289 -0.000919  0.000010  0.000928  0.085484
##
## Coefficients:
##                   Estimate Std. Error t value Pr(>|t|)
## (Intercept)     -1.036e-05  4.758e-06  -2.178 0.029391 *
## Asset_1_BRet_3   4.078e-02  4.077e-03  10.002  < 2e-16 ***
## Asset_1_BRet_10  1.708e-02  2.538e-03   6.730 1.70e-11 ***
## Asset_1_BRet_30  6.252e-03  1.261e-03   4.958 7.11e-07 ***
## Asset_2_BRet_3   2.591e-02  2.228e-03  11.626  < 2e-16 ***
## Asset_2_BRet_10 -5.405e-03  1.435e-03  -3.766 0.000166 ***
## Asset_2_BRet_30  8.884e-03  7.609e-04  11.677  < 2e-16 ***
## Asset_3_BRet_3   1.834e-02  2.247e-03   8.164 3.26e-16 ***
## Asset_3_BRet_10  3.418e-03  1.441e-03   2.372 0.017716 *
## Asset_3_BRet_30 -9.622e-04  7.295e-04  -1.319 0.187194
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 0.002882 on 366902 degrees of freedom
## Multiple R-squared:  0.004607,   Adjusted R-squared:  0.004583
## F-statistic: 188.7 on 9 and 366902 DF,  p-value: < 2.2e-16
```

```r
#the prediciton with linear regression wrt to training set
lm_predict_train = predict(lm_mod,newdata = train_set)

#the prediciton with linear regression wrt to testing set
lm_predict_test = predict(lm_mod,newdata = test_set)

#in-sample correlation
cor(lm_predict_train,train_set$rf_10)
```

```
## [1] 0.0678748
```

```r
#out-of-sample correlation
cor(lm_predict_test,test_set$rf_10)
```

```
## [1] 0.04067645
```

```r
lm_cor = rep(NA,524160)

#calculate the three week backward returns
left = length(lm_cor) - 21*24*60
first_index = c(rep(1,21*24*60),(1:left))
second_index = 1:length(lm_cor)

#using the train model to predict the whole year rf(t,10)
```

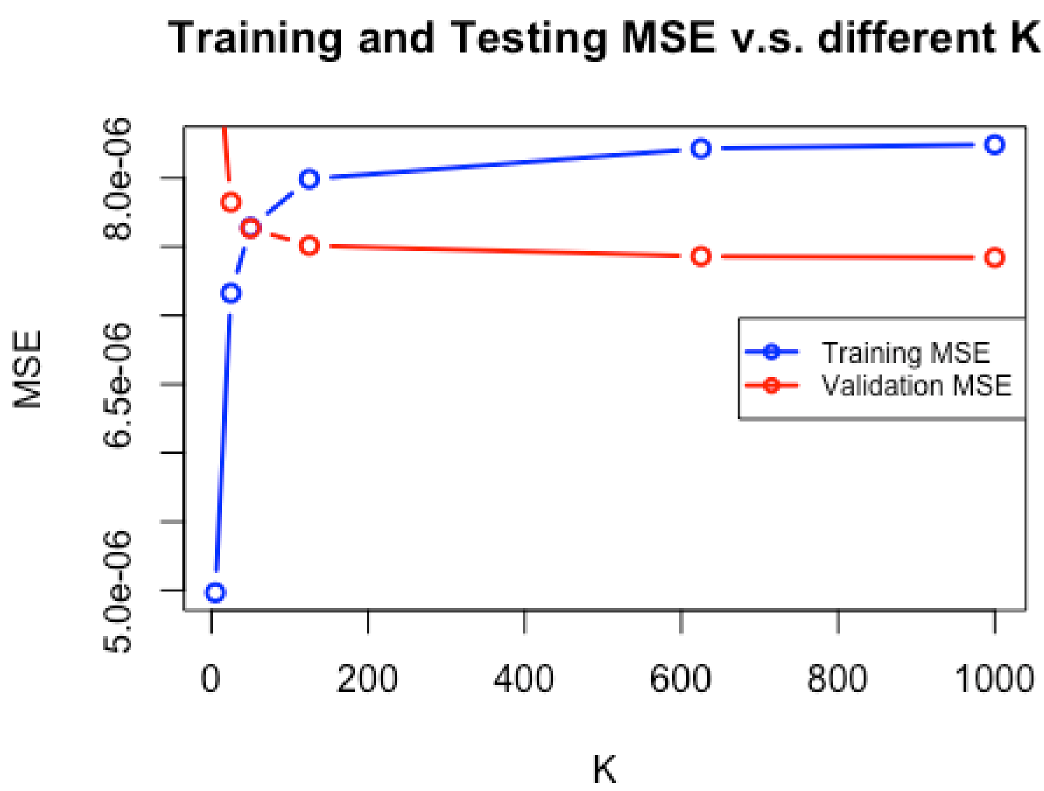
```r
lm_year_pred = predict(lm_mod,newdata = dat_forward)

for(i in 1:524160){
  temp_year_pred = lm_year_pred[first_index[i]:second_index[i]]
  temp_rf_10 = dat_forward$rf_10[first_index[i]:second_index[i]]
  lm_cor[i] = cor(temp_year_pred,temp_rf_10)
}
lm_cor[1] = 0
# plot the three week backward correlation between rf(t,10)_head and rf(t,10)
plot(lm_cor,cex = 0.5,ylab = "correlation coefficient")
```

2.4

```r
# run KNN with the rest of the K candidates
k_reg_c = c(5,25,50,125,625)
train_mse_knn_c= rep(NA,length(k_reg_c))
test_mse_knn_c = rep(NA,length(k_reg_c))

#run KNN regression
for(i in 1:length(k_reg_c)){
  train_temp = knn.reg(train = train_set[,-1],test = train_set[,-1],y =
train_set$rf_10,k = k_reg_c[i])
  train_mse_knn_c[i] = mean((train_temp$pred - train_set$rf_10)^2)

  test_temp = knn.reg(train = train_set[,-1],test = test_set[,-1],y =
train_set$rf_10,k = k_reg_c[i])
  test_mse_knn_c[i] = mean((test_temp$pred - test_set$rf_10)^2)
}
```

```r
# run KNN with the rest of the K candidates
k_reg_w = c(1000)
train_mse_knn_w= rep(NA,length(k_reg_w))
test_mse_knn_w = rep(NA,length(k_reg_w))


#run KNN regression
for(i in 1:length(k_reg_w)){
  train_temp = knn.reg(train = train_set[,-1],test = train_set[,-1],y =
train_set$rf_10,k = k_reg_w[i])
  train_mse_knn_w[i] = mean((train_temp$pred - train_set$rf_10)^2)

  test_temp = knn.reg(train = train_set[,-1],test = test_set[,-1],y =
train_set$rf_10,k = k_reg_w[i])
  test_mse_knn_w[i] = mean((test_temp$pred - test_set$rf_10)^2)
}
```

```r
k_reg_all = c(k_reg_c,k_reg_w) %>% print()

## [1]    5   25   50  125  625 1000

train_mse_all = c(train_mse_knn_c,train_mse_knn_w) %>% print()

## [1] 4.983888e-06 7.162388e-06 7.640673e-06 7.992362e-06 8.214067e-06
## [6] 8.242839e-06

test_mse_all = c(test_mse_knn_c,test_mse_knn_w) %>% print()

## [1] 9.056682e-06 7.823743e-06 7.633455e-06 7.507690e-06 7.430419e-06
## [6] 7.421158e-06

plot(x = k_reg_all,y = train_mse_all,type = "b",col = "blue",main = "Training
and Testing MSE v.s. different K",xlab = "K",ylab = "MSE",lwd = 2)
lines(x = k_reg_all,y = test_mse_all, type = "b", lwd = 2, col = "red")


legend("right", legend = c("Training MSE", "Validation MSE"),
col = c("blue", "red"), cex = .75, lwd = c(2, 2), pch = c(1, 1), lty = c(1,
1))
```




```
#optimal K based on the validation MSE
best_k = k_reg_all[which.min(test_mse_all)]  %>% print()

## [1] 1000

#in-sample correlation between prediction and response
train_knn_best = knn.reg(train = train_set[,-1],test = train_set[,-1],y =
train_set$rf_10,k = best_k)
cor(train_knn_best$pred,train_set$rf_10) %>% print()

## [1] 0.1181981

#out-of-sample correlation between prediction and response
test_knn_best = knn.reg(train = train_set[,-1],test = test_set[,-1],y =
train_set$rf_10,k = best_k)
cor(test_knn_best$pred,test_set$rf_10) %>% print()

## [1] 0.0432636

#prediction of whole year using the optimal K
knn_best_all = knn.reg(train = train_set[,-1],test = dat_forward[,-1],y =
train_set$rf_10,k = best_k)
knn_best_all$pred %>% head(10)
```

```
##  [1] -1.195285e-04  1.911873e-04  1.527146e-04  3.135965e-05 -5.334502e-05
##  [6] -3.334279e-04 -1.269600e-04 -2.894834e-05 -4.473460e-04 -5.403406e-04
```

2.5 Ridge and Lasso

Hello

```
h_ridge = c(3,10,30,60,120,180,240,360,480,600,720,960,1200,1440)

dat_rl_blank = data.frame(rep(1:nrow(dat),1))

#use for loop to create and join 9 columns to dat_blank
for(i in 1:3){
  for(j in 1:length(h_ridge)){
  dat_rl_col = dat[,i+1]
  dat_rl_temp = dat %>% mutate(shift_rl = lag(dat_rl_col,h_ridge[j])) %>%
mutate(temp = ifelse(is.na(shift_rl),dat[1,i+1],shift_rl) , last = (dat[,i+1]
- temp) / temp)
  dat_rl_blank = cbind(dat_rl_blank, dat_rl_temp$last)
  }
}
#drop the first column
data_rl = dat_rl_blank[,-1]

#rename the column
name_rl = rep(NA,42)
for(i in 1:3){
  for(j in 1:14){
    name_rl[j +(i-1)*14] =   paste0("Asset_",i,"_BRet_",h_ridge[j])
  }
}
colnames(data_rl) = name_rl

data_rl_full = cbind(rf_10 = dat_forward$rf_10,data_rl)
```

Hello

```
# use prediction matix
X_rl = model.matrix(rf_10 ~ .,data = data_rl_full)[, -1]

# y value
Y_rl = data_rl_full$rf_10

train_index = 1:(nrow(data_rl_full)*0.7)

#set up the training/testing X and Y
train_set_x = X_rl[train_index,]
train_set_y = Y_rl[train_index]
```

```
test_set_x = X_rl[-train_index,]
test_set_y = Y_rl[-train_index]


#check if the rows match the original dataset
nrow(train_set_x) + nrow(test_set_x)

## [1] 524160

length(train_set_y) + length(test_set_y)

## [1] 524160
```

###ridge regression hello

```
#ridge regression
ridge.mod = glmnet(x = train_set_x,train_set_y,alpha = 0)
ridge.pred.test = predict(ridge.mod, newx = test_set_x)
#test MSE
ridge.test.mse = colMeans((ridge.pred.test - matrix(test_set_y ,
length(test_set_y),ncol(ridge.pred.test)))^2)

ind.min = which.min(ridge.test.mse)
ridge.lambda = ridge.mod$lambda[ind.min]

#prediction for whole year
ridge.pred = predict(ridge.mod, s=ridge.lambda, newx=X_rl)
ridge.pred %>% head(10)

##                  1
## 1  -1.105216e-05
## 2   2.809314e-04
## 3   2.606674e-04
## 4   1.401545e-04
## 5  -1.482366e-04
## 6  -2.238085e-04
## 7  -1.253763e-04
## 8   1.151948e-05
## 9  -6.792211e-05
## 10 -3.365496e-04

#in-sample correlation
cor(ridge.pred[train_index],train_set_y)

## [1] 0.07374825

#out-of-sample correlation
cor(ridge.pred[-train_index],test_set_y)
```

```
## [1] 0.03789348
```

#lasso regression

```
#lasso regression
lasso.mod = glmnet(x = train_set_x,train_set_y,alpha =1)
lasso.pred.test = predict(lasso.mod, newx = test_set_x)
#test MSE
lasso.test.mse = colMeans((lasso.pred.test - matrix(test_set_y ,
length(test_set_y),ncol(lasso.pred.test)))^2)

ind.min.lasso = which.min(lasso.test.mse)
lasso.lambda = lasso.mod$lambda[ind.min.lasso]

#prediction for whole year
lasso.pred = predict(lasso.mod, s=lasso.lambda, newx=X_rl)
lasso.pred %>% head(10)

##                1
## 1  -1.079673e-05
## 2   3.223592e-04
## 3   3.038444e-04
## 4   1.683735e-04
## 5  -1.694856e-04
## 6  -2.436007e-04
## 7  -1.292494e-04
## 8   2.994541e-05
## 9  -4.449895e-05
## 10 -3.208587e-04

#in-sample correlation
cor(lasso.pred[train_index],train_set_y)

## [1] 0.06743857

#out-of-sample correlation
cor(lasso.pred[-train_index],test_set_y)

## [1] 0.03913352
```

#2.6

```
pcr_mse = rep(NA,42)

   asset_pcr = pcr(rf_10~.,data =data_rl_full[train_index,],scale =
TRUE,center = TRUE)

for (i in 1:42){
    pcr_pred = predict(asset_pcr,newdata =data_rl_full[-train_index,],ncomp =
```

```r
i)
    pcr_mse[i] = mean((pcr_pred -data_rl_full[-train_index,]$rf_10)^2)
}
best_ncp = which.min(pcr_mse)
```

```r
#prediction of the whole year
pcr_pred = predict(asset_pcr,data_rl_full
[,names(data_rl_full)!='rf_10'],ncomp=best_ncp)
pcr_pred %>% head(10)
```

```
##  [1] -9.850549e-06  3.199368e-04  2.878643e-04  1.372336e-04 -1.713111e-04
##  [6] -2.573936e-04 -1.516407e-04  6.943068e-06 -8.316171e-05 -3.824973e-04
```

```r
train_size = nrow(data_rl_full) *0.7
#in-sample correlation
cor(data_rl_full[train_index,]$rf_10, pcr_pred[1:train_size]) %>% print()
```

```
## [1] 0.06698264
```

```r
#out-of-sample correlation
cor(data_rl_full[-train_index,]$rf_10, pcr_pred[-(1:train_size)]) %>% print()
```

```
## [1] 0.04155347
```