

Stat 897 Fall 2017 Data Analysis Assignment 5

Penn State

Due September 24, 2017

(1) In the first part of this assignment we will use the College data found in the ISLR library. Split the data into a training set of size 100 and test set with the rest. Your goal is to predict the number of applications received using the other variables in the data set.

```
library(leaps)
library(ISLR)
library(quantmod)

## Loading required package: xts
## Loading required package: zoo
##
## Attaching package: 'zoo'
## The following objects are masked from 'package:base':
##
##   as.Date, as.Date.numeric
## Loading required package: TTR
## Version 0.4-0 included new data defaults. See ?getSymbols.
library(glmnet)

## Loading required package: Matrix
## Loading required package: foreach
## Loaded glmnet 2.0-12
#install.packages('plotmo')
library(plotmo)

## Loading required package: plotrix
## Loading required package: TeachingDemos
data("College")

set.seed(1)
trainingRows=sample(nrow(College), 100, replace = FALSE)
train = College[trainingRows,]
test = College[-trainingRows,]
```

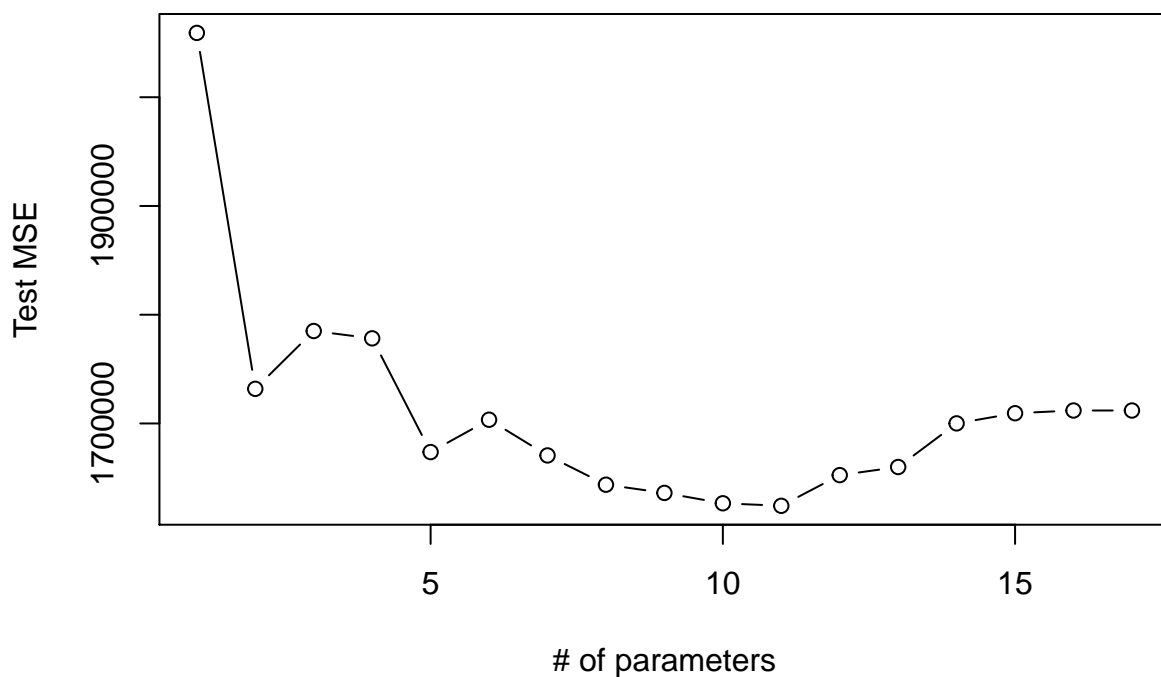
(a) Fit a “best” model obtained from your previous assignment on the training set and report the test error for this model.

We will use the test MSE to find the best model

```

regfit.best=regsubsets (Apps~.,data=train, nvmax=17)
test.mat=model.matrix (Apps~.,data=test)
test.val.errors =rep(NA ,17)
for(i in 1:17){
  coefi=coef(regfit.best ,id=i)
  pred=test.mat [,names(coefi)] %*% coefi
  test.val.errors [i]= mean(( test$Apps-pred)^2)
}
plot(test.val.errors ,type='b', xlab='# of parameters', ylab='Test MSE')

```



We see that we get the lowest test MSE for model with 11 parameters:

```
which.min(test.val.errors)
```

```
## [1] 11
```

```
test.val.errors[11]
```

```
## [1] 1624278
```

The test MSE is reported to be 1624278

(b) Fit a ridge regression model on the training set, with λ chosen by cross-validation. Report the test error obtained.

```

train.mat <- model.matrix(Apps ~ ., data = train)
test.mat <- model.matrix(Apps ~ ., data = test)

```

```
cv.ridge <- cv.glmnet(train.mat, train$Apps, alpha = 0)
bestlam.ridge <- cv.ridge$lambda.min
bestlam.ridge
```

```
## [1] 634.627
```

We see that the value of lambda a that results in the smallest crossvalidation error is 635. The test MSE associated with this value of lambda is:

```
grid = 10^ seq (10,-2, length =100)
fit.ridge =glmnet(train.mat, train$Apps, alpha = 0, lambda = grid, thresh = 1e-12)
pred.ridge = predict (fit.ridge, s=bestlam.ridge, newx=test.mat)
mean(( pred.ridge - test$Apps)^2)
```

```
## [1] 1680229
```

The test MSE with ridge is: 1680229

(c) Fit a lasso model on the training set, with λ chosen by cross-validation. Report the test error obtained.

```
cv.lasso <- cv.glmnet(train.mat, train$Apps, alpha = 1)
bestlam.lasso <- cv.lasso$lambda.min
bestlam.lasso
```

```
## [1] 72.96669
```

We see that the value of lambda a that results in the smallest crossvalidation error is 73 The test MSE associated with this value of lambda is:

```
fit.lasso <- glmnet(train.mat, train$Apps, alpha = 1, lambda = grid, thresh = 1e-12)
pred.lasso=predict (fit.lasso, s=bestlam.lasso, newx=test.mat)
mean(( pred.lasso - test$Apps)^2)
```

```
## [1] 1621135
```

The test MSE with lasso is: 1621135

(d) Compare the result obtained in (a) – (c). How accurately can we predict the number of college applications? Is there much difference among the test errors resulting from different approaches?

We find that: Best model with 11 parameters test MSE: 1624278 Ridge regression test MSE: 1680229 Lasso test MSE: 1621135

The test MSE is lowest for lasso, closely followed by best model with 11 parameters. Ridge regression has the highest MSE.

MSE for lasso and best model are relatively close than the ridge test MSE.

(e) Now partition the data into a training set of size 600 and a test set with the rest. Compare the test errors from the “best” linear regression model, ridge and lasso models. Note that, the “best” model here may not be the “best” model obtained before.

```

trainingRows=sample (nrow(College), 600, replace = FALSE)
train = College[trainingRows,]
test = College[-trainingRows,]

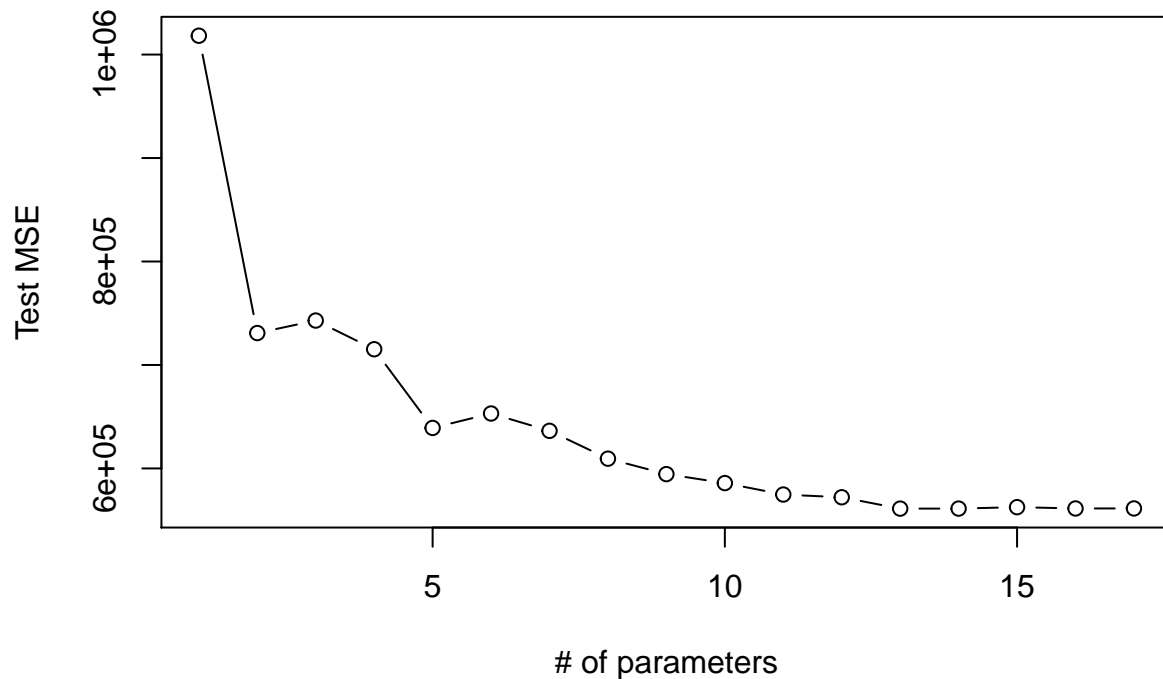
```

We will use the test MSE to find the best model

```

regfit.best=regsubsets (Apps~.,data=train, nvmax=17)
test.mat=model.matrix (Apps~.,data=test)
test.val.errors =rep(NA ,17)
for(i in 1:17){
  coefi=coef(regfit.best ,id=i)
  pred=test.mat [,names(coefi)] %*% coefi
  test.val.errors [i]= mean(( test$Apps-pred)^2)
}
plot(test.val.errors ,type='b', xlab='# of parameters', ylab='Test MSE')

```



```

which.min(test.val.errors)

```

```
## [1] 13
```

```
test.val.errors[which.min(test.val.errors)]
```

```
## [1] 561171.9
```

We see that we get the lowest test MSE for model with 13 parameters. It is a different result than the last time.

The test MSE for the selected model is: 561171.9

(b) Fit a ridge regression model on the training set, with λ chosen by cross-validation. Report the test error obtained.

```
train.mat <- model.matrix(Apps ~ ., data = train)
test.mat <- model.matrix(Apps ~ ., data = test)
cv.ridge <- cv.glmnet(train.mat, train$Apps, alpha = 0)
bestlam.ridge <- cv.ridge$lambda.min
bestlam.ridge
```

```
## [1] 413.419
```

We see that the value of lambda a that results in the smallest crossvalidation error is 413. The test MSE associated with this value of lambda is:

```
fit.ridge = glmnet(train.mat, train$Apps, alpha = 0, lambda = grid, thresh = 1e-12)
pred.ridge = predict (fit.ridge, s=bestlam.ridge, newx=test.mat)
mean(( pred.ridge - test$Apps)^2)
```

```
## [1] 691818.8
```

The test MSE: 691818.8

(c) Fit a lasso model on the training set, with λ chosen by cross-validation. Report the test error obtained.

```
cv.lasso <- cv.glmnet(train.mat, train$Apps, alpha = 1)
bestlam.lasso <- cv.lasso$lambda.min
bestlam.lasso
```

```
## [1] 2.657488
```

We see that the value of lambda a that results in the smallest crossvalidation error is 3. The test MSE associated with this value of lambda is:

```
fit.lasso <- glmnet(train.mat, train$Apps, alpha = 1, lambda = grid, thresh = 1e-12)
pred.lasso=predict (fit.lasso, s=bestlam.lasso, newx=test.mat)
mean(( pred.lasso - test$Apps)^2)
```

```
## [1] 561649.3
```

The test MSE: 561649.3

(f) Do you see any difference between the two sets of results? Comment.

We find that (600 training set): Best model with 13 parameters test MSE: 561171.9 Ridge regression test MSE: 691818.8 Lasso test MSE: 561649.3

The overall MSE has reduced but we observe a similar pattern as before with the diff that the test MSE is lowest for best model, closely followed by lasso. Ridge regression has the highest MSE.

MSE for lasso and best model are relatively close than the ridge test MSE.

(100 training set): Best model with 11 parameters test MSE: 1624278 Ridge regression test MSE: 1680229 Lasso test MSE: 1621135

(2) The file Sp.Rdv contains daily returns for 501 stocks in 2016. It was created as follows.

(You don't need to run this, i.e. leave eval=FALSE)

```
library("quantmod")
sp = read.csv("SP500HistoricalComponents.csv")
Symbols = sp[which(sp$X12.31.2016 == "X"), "Ticker"]
StartDate = '2016-01-01'
EndDate = '2016-12-31'

Stocks = lapply(Symbols, function(sym) {
  print(sym)
  dailyReturn(na.omit(getSymbols(as.character(sym), from = StartDate, to = EndDate,
                                auto.assign = FALSE, src = "yahoo")))
})

SPreturns = do.call(merge, Stocks)
colnames(SPreturns) = Symbols

#Clean out cols with NA
spreturns = SPreturns[, colSums(is.na(SPreturns)) == 0]
save(spreturns, file = "spreturns.Rda")
```

Make sure you downloaded the data from the assignment page, and you can load it using:

```
load("hw5_spreturns.Rda")
```

I have constructed a secret long-only portfolio chosen from these stocks. It contains between five and twenty stocks. The daily return of this portfolio for each trading day of 2016 is in the object `portfolioreturns` which you can load from file with:

```
load("hw5_portfolioreturnsstatic.Rda")
dim(spreturns)
```

```
## [1] 252 501
```

```
dim(portfolioreturns)
```

```
## [1] 252 1
```

Your goal is to recover the stocks and weights of the secret portfolio.

Note that you can think of a portfolio as a vector of nonnegative weights that sum to one. For simplicity, we are assuming that this portfolio is rebalanced daily at the closing prices. Then if the daily returns vector on date d is r_d and the weight vector is w_d , the daily return for the portfolio is the dot product $r_d \cdot w_d$. If this were a buy-and-hold portfolio, we would have to back into the returns more carefully.

(a) First try to fit an ordinary regression (lm) with portfolio returns as the response and spretuns as the predictors. What happens? What problem do you run into?

```
options(max.print=1000000)
portWithReturnsAndStocks <- cbind(portfolioReturns, spretuns)
dim(portWithReturnsAndStocks)

## [1] 252 502

colnames(portWithReturnsAndStocks)[1] <- "portfolioReturns"

lm.fit=lm(portfolioReturns~., data=portWithReturnsAndStocks)
summary(lm.fit)

##
## Call:
## lm(formula = portfolioReturns ~ ., data = portWithReturnsAndStocks)
##
## Residuals:
## ALL 252 residuals are 0: no residual degrees of freedom!
##
## Coefficients: (250 not defined because of singularities)
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  0.001525      NA      NA      NA
## A            -0.752916      NA      NA      NA
## AAL          -0.057577      NA      NA      NA
## AAP          -0.094218      NA      NA      NA
## AAPL         0.060840      NA      NA      NA
## ABBV         0.461477      NA      NA      NA
## ABC          -0.240886      NA      NA      NA
## ABT          1.247800      NA      NA      NA
## ACN          -0.449216      NA      NA      NA
## ADBE         0.060757      NA      NA      NA
## ADI          -0.329032      NA      NA      NA
## ADM          0.038342      NA      NA      NA
## ADP          0.908372      NA      NA      NA
## ADS         -0.128100      NA      NA      NA
## ADSK         0.132905      NA      NA      NA
## AEE          -2.217198      NA      NA      NA
## AEP          2.235636      NA      NA      NA
## AES          0.057907      NA      NA      NA
## AET          0.192574      NA      NA      NA
## AFL          1.537212      NA      NA      NA
## AGN          -0.264545      NA      NA      NA
## AIG          0.425684      NA      NA      NA
## AIV          0.349113      NA      NA      NA
```

## AIZ	-0.095779	NA	NA	NA
## AJG	0.006105	NA	NA	NA
## AKAM	-0.213798	NA	NA	NA
## ALB	-0.540041	NA	NA	NA
## ALK	0.362265	NA	NA	NA
## ALL	-0.303270	NA	NA	NA
## ALLE	0.078936	NA	NA	NA
## ALXN	-0.583297	NA	NA	NA
## AMAT	-0.584111	NA	NA	NA
## AME	-0.823864	NA	NA	NA
## AMG	0.413978	NA	NA	NA
## AMGN	-0.177249	NA	NA	NA
## AMP	0.384805	NA	NA	NA
## AMT	0.195539	NA	NA	NA
## AMZN	-0.219948	NA	NA	NA
## AN	0.345714	NA	NA	NA
## ANTM	0.729511	NA	NA	NA
## AON	1.196071	NA	NA	NA
## APA	-0.488080	NA	NA	NA
## APC	0.104617	NA	NA	NA
## APD	0.467921	NA	NA	NA
## APH	0.236326	NA	NA	NA
## ARNC	-0.007023	NA	NA	NA
## ATVI	-0.409678	NA	NA	NA
## AVB	0.296708	NA	NA	NA
## AVGO	0.204381	NA	NA	NA
## AVY	0.393299	NA	NA	NA
## AWK	0.028447	NA	NA	NA
## AXP	0.627870	NA	NA	NA
## AYI	0.147942	NA	NA	NA
## AZO	0.468430	NA	NA	NA
## BA	-0.682487	NA	NA	NA
## BAC	0.186240	NA	NA	NA
## BAX	-0.737112	NA	NA	NA
## BBBY	-0.216040	NA	NA	NA
## BBT	-1.653938	NA	NA	NA
## BBY	0.104649	NA	NA	NA
## BCR	0.481008	NA	NA	NA
## BDX	-0.587362	NA	NA	NA
## BEN	-0.728245	NA	NA	NA
## BF.B	-0.144899	NA	NA	NA
## BHI	0.154007	NA	NA	NA
## BIIB	0.440162	NA	NA	NA
## BK	-0.855546	NA	NA	NA
## BLK	0.109083	NA	NA	NA
## BLL	-0.329718	NA	NA	NA
## BMY	-0.068873	NA	NA	NA
## BRK.B	0.760065	NA	NA	NA
## BSX	-0.135188	NA	NA	NA
## BWA	-0.210303	NA	NA	NA
## BXP	-0.003172	NA	NA	NA
## C	0.204513	NA	NA	NA
## CA	-0.183412	NA	NA	NA
## CAG	0.234695	NA	NA	NA

## CAH	-0.031444	NA	NA	NA
## CAT	0.703032	NA	NA	NA
## CB	0.354181	NA	NA	NA
## CBG	0.101964	NA	NA	NA
## CBS	-0.034096	NA	NA	NA
## CCI	-0.227332	NA	NA	NA
## CCL	-0.058523	NA	NA	NA
## CELG	-0.396638	NA	NA	NA
## CERN	-0.076368	NA	NA	NA
## CF	0.207523	NA	NA	NA
## CFG	-0.189083	NA	NA	NA
## CHD	-0.072280	NA	NA	NA
## CHK	0.090957	NA	NA	NA
## CHRW	-0.336519	NA	NA	NA
## CHTR	-0.649265	NA	NA	NA
## CI	-0.511516	NA	NA	NA
## CINF	-0.015048	NA	NA	NA
## CL	-0.220169	NA	NA	NA
## CLX	1.075131	NA	NA	NA
## CMA	0.785082	NA	NA	NA
## CMCSA	-0.442488	NA	NA	NA
## CME	0.404537	NA	NA	NA
## CMG	-0.035822	NA	NA	NA
## CMI	-0.426753	NA	NA	NA
## CMS	-1.160802	NA	NA	NA
## CNC	0.144513	NA	NA	NA
## CNP	-0.287291	NA	NA	NA
## COF	0.093460	NA	NA	NA
## COG	-0.070436	NA	NA	NA
## COH	0.276752	NA	NA	NA
## COL	-0.715455	NA	NA	NA
## COO	-0.102564	NA	NA	NA
## COP	0.112794	NA	NA	NA
## COST	-0.430260	NA	NA	NA
## COTY	-0.076408	NA	NA	NA
## CPB	-0.986939	NA	NA	NA
## CRM	0.311644	NA	NA	NA
## CSCO	-0.420070	NA	NA	NA
## CSRA	-0.318300	NA	NA	NA
## CSX	-0.050329	NA	NA	NA
## CTAS	0.127489	NA	NA	NA
## CTL	-0.347094	NA	NA	NA
## CTSH	0.177388	NA	NA	NA
## CTXS	0.664393	NA	NA	NA
## CVS	0.577500	NA	NA	NA
## CVX	-0.077712	NA	NA	NA
## CXO	0.898262	NA	NA	NA
## D	-0.389992	NA	NA	NA
## DAL	0.379405	NA	NA	NA
## DD	0.619175	NA	NA	NA
## DE	0.172169	NA	NA	NA
## DFS	-1.076650	NA	NA	NA
## DG	-0.122964	NA	NA	NA
## DGX	-0.098586	NA	NA	NA

## DHI	0.155587	NA	NA	NA
## DHR	-0.053931	NA	NA	NA
## DIS	0.767953	NA	NA	NA
## DISCA	0.466659	NA	NA	NA
## DISCK	-0.821751	NA	NA	NA
## DLPH	-0.016282	NA	NA	NA
## DLR	0.232095	NA	NA	NA
## DLTR	-0.145634	NA	NA	NA
## DNB	0.110330	NA	NA	NA
## DOV	0.478662	NA	NA	NA
## DOW	-0.891012	NA	NA	NA
## DPS	0.495028	NA	NA	NA
## DRI	0.047791	NA	NA	NA
## DTE	1.987056	NA	NA	NA
## DUK	-1.285035	NA	NA	NA
## DVA	0.174313	NA	NA	NA
## DVN	0.116323	NA	NA	NA
## EA	0.306731	NA	NA	NA
## EBAY	-0.184828	NA	NA	NA
## ECL	0.084057	NA	NA	NA
## ED	0.114259	NA	NA	NA
## EFX	-0.154614	NA	NA	NA
## EIX	0.115249	NA	NA	NA
## EL	0.409133	NA	NA	NA
## EMN	-0.350901	NA	NA	NA
## EMR	-0.346147	NA	NA	NA
## ENDP	0.115596	NA	NA	NA
## EOG	-0.782028	NA	NA	NA
## EQIX	0.575762	NA	NA	NA
## EQR	-0.073419	NA	NA	NA
## EQT	0.319864	NA	NA	NA
## ES	1.214052	NA	NA	NA
## ESRX	-0.698489	NA	NA	NA
## ESS	0.377826	NA	NA	NA
## ETFC	0.137954	NA	NA	NA
## ETN	-0.479066	NA	NA	NA
## ETR	-0.598589	NA	NA	NA
## EVHC	0.031668	NA	NA	NA
## EW	0.137961	NA	NA	NA
## EXC	0.237056	NA	NA	NA
## EXPD	-0.492622	NA	NA	NA
## EXPE	0.309651	NA	NA	NA
## EXR	-0.395261	NA	NA	NA
## F	-0.291823	NA	NA	NA
## FAST	0.353287	NA	NA	NA
## FB	-0.196186	NA	NA	NA
## FBHS	0.010088	NA	NA	NA
## FCX	0.046338	NA	NA	NA
## FDX	0.454865	NA	NA	NA
## FE	-0.088282	NA	NA	NA
## FFIV	0.050890	NA	NA	NA
## FIS	-0.423613	NA	NA	NA
## FISV	0.059242	NA	NA	NA
## FITB	0.443553	NA	NA	NA

## FL	0.035503	NA	NA	NA
## FLIR	-0.166893	NA	NA	NA
## FLR	-0.472372	NA	NA	NA
## FLS	0.704143	NA	NA	NA
## FMC	0.107565	NA	NA	NA
## FOX	-0.882660	NA	NA	NA
## FOXA	1.087309	NA	NA	NA
## FRT	-0.198502	NA	NA	NA
## FSLR	0.013091	NA	NA	NA
## FTI	-0.563737	NA	NA	NA
## FTR	0.162302	NA	NA	NA
## GD	-0.173312	NA	NA	NA
## GE	0.691920	NA	NA	NA
## GGP	-0.856091	NA	NA	NA
## GILD	0.259934	NA	NA	NA
## GIS	0.791991	NA	NA	NA
## GLW	-0.186643	NA	NA	NA
## GM	0.207095	NA	NA	NA
## GOOG	-0.371412	NA	NA	NA
## GOOGL	0.595635	NA	NA	NA
## GPC	-0.463923	NA	NA	NA
## GPN	0.178230	NA	NA	NA
## GPS	0.245789	NA	NA	NA
## GRMN	-0.137435	NA	NA	NA
## GS	0.110510	NA	NA	NA
## GT	-0.348271	NA	NA	NA
## GWW	0.207698	NA	NA	NA
## HAL	-0.263731	NA	NA	NA
## HAR	0.263477	NA	NA	NA
## HAS	0.046177	NA	NA	NA
## HBAN	0.350658	NA	NA	NA
## HBI	-0.055808	NA	NA	NA
## HCA	-0.662768	NA	NA	NA
## HCN	0.336948	NA	NA	NA
## HCP	-0.275236	NA	NA	NA
## HD	-0.790137	NA	NA	NA
## HES	-0.408293	NA	NA	NA
## HIG	-0.212661	NA	NA	NA
## HOG	-0.018744	NA	NA	NA
## HOLX	-0.668797	NA	NA	NA
## HON	0.018582	NA	NA	NA
## HP	0.239023	NA	NA	NA
## HPE	0.289372	NA	NA	NA
## HPQ	0.313270	NA	NA	NA
## HRB	0.406856	NA	NA	NA
## HRL	-0.042603	NA	NA	NA
## HRS	-0.225242	NA	NA	NA
## HSIC	0.330772	NA	NA	NA
## HST	0.016855	NA	NA	NA
## HSY	-0.051291	NA	NA	NA
## HUM	-0.373588	NA	NA	NA
## IBM	0.002144	NA	NA	NA
## ICE	-0.081279	NA	NA	NA
## IFF	-0.130718	NA	NA	NA

## ILMN	0.328409	NA	NA	NA
## INTC	-0.253233	NA	NA	NA
## INTU	-0.057137	NA	NA	NA
## IP	-0.724142	NA	NA	NA
## IPG	0.351920	NA	NA	NA
## IR	-0.954283	NA	NA	NA
## IRM	-0.481459	NA	NA	NA
## ISRG	0.595607	NA	NA	NA
## ITW	0.321038	NA	NA	NA
## IVZ	0.105643	NA	NA	NA
## JBHT	0.354974	NA	NA	NA
## JCI	0.217565	NA	NA	NA
## JEC	0.937129	NA	NA	NA
## JNJ	NA	NA	NA	NA
## JNPR	NA	NA	NA	NA
## JPM	NA	NA	NA	NA
## JWN	NA	NA	NA	NA
## K	NA	NA	NA	NA
## KEY	NA	NA	NA	NA
## KHC	NA	NA	NA	NA
## KIM	NA	NA	NA	NA
## KLAC	NA	NA	NA	NA
## KMB	NA	NA	NA	NA
## KMI	NA	NA	NA	NA
## KMX	NA	NA	NA	NA
## KO	NA	NA	NA	NA
## KORS	NA	NA	NA	NA
## KR	NA	NA	NA	NA
## KSS	NA	NA	NA	NA
## KSU	NA	NA	NA	NA
## L	NA	NA	NA	NA
## LB	NA	NA	NA	NA
## LEG	NA	NA	NA	NA
## LEN	NA	NA	NA	NA
## LH	NA	NA	NA	NA
## LKQ	NA	NA	NA	NA
## LLL	NA	NA	NA	NA
## LLTC	NA	NA	NA	NA
## LLY	NA	NA	NA	NA
## LMT	NA	NA	NA	NA
## LNC	NA	NA	NA	NA
## LNT	NA	NA	NA	NA
## LOW	NA	NA	NA	NA
## LRCX	NA	NA	NA	NA
## LUK	NA	NA	NA	NA
## LUV	NA	NA	NA	NA
## LVLT	NA	NA	NA	NA
## LYB	NA	NA	NA	NA
## M	NA	NA	NA	NA
## MA	NA	NA	NA	NA
## MAA	NA	NA	NA	NA
## MAC	NA	NA	NA	NA
## MAR	NA	NA	NA	NA
## MAS	NA	NA	NA	NA

## MAT	NA	NA	NA	NA
## MCD	NA	NA	NA	NA
## MCHP	NA	NA	NA	NA
## MCK	NA	NA	NA	NA
## MCO	NA	NA	NA	NA
## MDLZ	NA	NA	NA	NA
## MDT	NA	NA	NA	NA
## MET	NA	NA	NA	NA
## MHK	NA	NA	NA	NA
## MJN	NA	NA	NA	NA
## MKC	NA	NA	NA	NA
## MLM	NA	NA	NA	NA
## MMC	NA	NA	NA	NA
## MMM	NA	NA	NA	NA
## MNK	NA	NA	NA	NA
## MNST	NA	NA	NA	NA
## MO	NA	NA	NA	NA
## MON	NA	NA	NA	NA
## MOS	NA	NA	NA	NA
## MPC	NA	NA	NA	NA
## MRK	NA	NA	NA	NA
## MRO	NA	NA	NA	NA
## MS	NA	NA	NA	NA
## MSFT	NA	NA	NA	NA
## MSI	NA	NA	NA	NA
## MTB	NA	NA	NA	NA
## MTD	NA	NA	NA	NA
## MU	NA	NA	NA	NA
## MUR	NA	NA	NA	NA
## MYL	NA	NA	NA	NA
## NAVI	NA	NA	NA	NA
## NBL	NA	NA	NA	NA
## NDAQ	NA	NA	NA	NA
## NEE	NA	NA	NA	NA
## NEM	NA	NA	NA	NA
## NFLX	NA	NA	NA	NA
## NFX	NA	NA	NA	NA
## NI	NA	NA	NA	NA
## NKE	NA	NA	NA	NA
## NLSN	NA	NA	NA	NA
## NOC	NA	NA	NA	NA
## NOV	NA	NA	NA	NA
## NRG	NA	NA	NA	NA
## NSC	NA	NA	NA	NA
## NTAP	NA	NA	NA	NA
## NTRS	NA	NA	NA	NA
## NUE	NA	NA	NA	NA
## NVDA	NA	NA	NA	NA
## NWL	NA	NA	NA	NA
## NWS	NA	NA	NA	NA
## NWSA	NA	NA	NA	NA
## O	NA	NA	NA	NA
## OKE	NA	NA	NA	NA
## OMC	NA	NA	NA	NA

## ORCL	NA	NA	NA	NA
## ORLY	NA	NA	NA	NA
## OXY	NA	NA	NA	NA
## PAYX	NA	NA	NA	NA
## PBCT	NA	NA	NA	NA
## PBI	NA	NA	NA	NA
## PCAR	NA	NA	NA	NA
## PCG	NA	NA	NA	NA
## PCLN	NA	NA	NA	NA
## PDCO	NA	NA	NA	NA
## PEG	NA	NA	NA	NA
## PEP	NA	NA	NA	NA
## PFE	NA	NA	NA	NA
## PFG	NA	NA	NA	NA
## PG	NA	NA	NA	NA
## PGR	NA	NA	NA	NA
## PH	NA	NA	NA	NA
## PHM	NA	NA	NA	NA
## PKI	NA	NA	NA	NA
## PLD	NA	NA	NA	NA
## PM	NA	NA	NA	NA
## PNC	NA	NA	NA	NA
## PNR	NA	NA	NA	NA
## PNW	NA	NA	NA	NA
## PPG	NA	NA	NA	NA
## PPL	NA	NA	NA	NA
## PRGO	NA	NA	NA	NA
## PRU	NA	NA	NA	NA
## PSA	NA	NA	NA	NA
## PSX	NA	NA	NA	NA
## PVH	NA	NA	NA	NA
## PWR	NA	NA	NA	NA
## PX	NA	NA	NA	NA
## PXD	NA	NA	NA	NA
## PYPL	NA	NA	NA	NA
## QCOM	NA	NA	NA	NA
## QRVO	NA	NA	NA	NA
## R	NA	NA	NA	NA
## RAI	NA	NA	NA	NA
## RCL	NA	NA	NA	NA
## REGN	NA	NA	NA	NA
## RF	NA	NA	NA	NA
## RHI	NA	NA	NA	NA
## RHT	NA	NA	NA	NA
## RIG	NA	NA	NA	NA
## RL	NA	NA	NA	NA
## ROK	NA	NA	NA	NA
## ROP	NA	NA	NA	NA
## ROST	NA	NA	NA	NA
## RRC	NA	NA	NA	NA
## RSG	NA	NA	NA	NA
## RTN	NA	NA	NA	NA
## SBUX	NA	NA	NA	NA
## SCG	NA	NA	NA	NA

## SCHW	NA	NA	NA	NA
## SE	NA	NA	NA	NA
## SEE	NA	NA	NA	NA
## SHW	NA	NA	NA	NA
## SIG	NA	NA	NA	NA
## SJM	NA	NA	NA	NA
## SLB	NA	NA	NA	NA
## SLG	NA	NA	NA	NA
## SNA	NA	NA	NA	NA
## SNI	NA	NA	NA	NA
## SO	NA	NA	NA	NA
## SPG	NA	NA	NA	NA
## SPGI	NA	NA	NA	NA
## SPLS	NA	NA	NA	NA
## SRCL	NA	NA	NA	NA
## SRE	NA	NA	NA	NA
## STI	NA	NA	NA	NA
## STJ	NA	NA	NA	NA
## STT	NA	NA	NA	NA
## STX	NA	NA	NA	NA
## STZ	NA	NA	NA	NA
## SWK	NA	NA	NA	NA
## SWKS	NA	NA	NA	NA
## SWN	NA	NA	NA	NA
## SYF	NA	NA	NA	NA
## SYK	NA	NA	NA	NA
## SYMC	NA	NA	NA	NA
## SYY	NA	NA	NA	NA
## T	NA	NA	NA	NA
## TAP	NA	NA	NA	NA
## TDC	NA	NA	NA	NA
## TDG	NA	NA	NA	NA
## TEL	NA	NA	NA	NA
## TGNA	NA	NA	NA	NA
## TGT	NA	NA	NA	NA
## TIF	NA	NA	NA	NA
## TJX	NA	NA	NA	NA
## TMK	NA	NA	NA	NA
## TMO	NA	NA	NA	NA
## TRIP	NA	NA	NA	NA
## TROW	NA	NA	NA	NA
## TRV	NA	NA	NA	NA
## TSCO	NA	NA	NA	NA
## TSN	NA	NA	NA	NA
## TSO	NA	NA	NA	NA
## TSS	NA	NA	NA	NA
## TWX	NA	NA	NA	NA
## TXN	NA	NA	NA	NA
## TXT	NA	NA	NA	NA
## UAL	NA	NA	NA	NA
## UDR	NA	NA	NA	NA
## UHS	NA	NA	NA	NA
## ULTA	NA	NA	NA	NA
## UNH	NA	NA	NA	NA

```

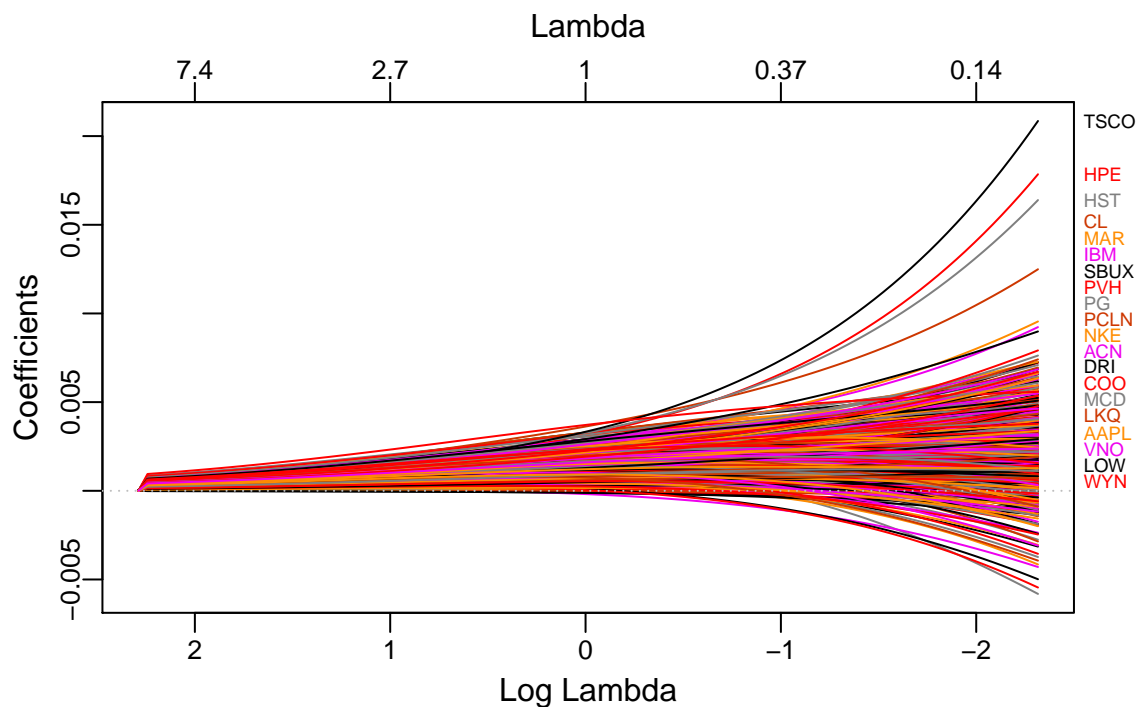
## UNM          NA          NA          NA          NA
## UNP          NA          NA          NA          NA
## UPS          NA          NA          NA          NA
## URBN         NA          NA          NA          NA
## URI          NA          NA          NA          NA
## USB          NA          NA          NA          NA
## UTX          NA          NA          NA          NA
## V            NA          NA          NA          NA
## VAR          NA          NA          NA          NA
## VFC          NA          NA          NA          NA
## VIAB         NA          NA          NA          NA
## VLO          NA          NA          NA          NA
## VMC          NA          NA          NA          NA
## VNO          NA          NA          NA          NA
## VRSK         NA          NA          NA          NA
## VRSN         NA          NA          NA          NA
## VRTX         NA          NA          NA          NA
## VTR          NA          NA          NA          NA
## VZ           NA          NA          NA          NA
## WAT          NA          NA          NA          NA
## WBA          NA          NA          NA          NA
## WDC          NA          NA          NA          NA
## WEC          NA          NA          NA          NA
## WFC          NA          NA          NA          NA
## WFM          NA          NA          NA          NA
## WHR          NA          NA          NA          NA
## WM           NA          NA          NA          NA
## WMB          NA          NA          NA          NA
## WMT          NA          NA          NA          NA
## WRK          NA          NA          NA          NA
## WU           NA          NA          NA          NA
## WY           NA          NA          NA          NA
## WYN          NA          NA          NA          NA
## WYNN         NA          NA          NA          NA
## XEC          NA          NA          NA          NA
## XEL          NA          NA          NA          NA
## XL           NA          NA          NA          NA
## XLNX         NA          NA          NA          NA
## XOM          NA          NA          NA          NA
## XRAY         NA          NA          NA          NA
## XRX          NA          NA          NA          NA
## XYL          NA          NA          NA          NA
## YHOO         NA          NA          NA          NA
## YUM          NA          NA          NA          NA
## ZBH          NA          NA          NA          NA
## ZION         NA          NA          NA          NA
## ZTS          NA          NA          NA          NA
##
## Residual standard error: NaN on 0 degrees of freedom
## Multiple R-squared:      1, Adjusted R-squared:      NaN
## F-statistic:      NaN on 251 and 0 DF, p-value: NA
options(max.print=252)

```

There is a clear problem of not having any significant parameter in the model.

(b) Now use elasticnet (Lasso, Ridge, or a combination, i.e. glmnet) instead of linear regression to model the secret portfolio. Once get a smaller set of variables from the shrinkage, refit a linear model with only those predictors. [Note, you may want to check out the plotmo and pander packages for nice plots and outputs for your models.]

```
x=model.matrix (portfolioreturns~., portWithReturnsAndStocks)[,-1];
y=portWithReturnsAndStocks$portfolioreturns
grid =10^ seq (10,-2, length =100)
port.ridge =glmnet (x, y,alpha =0, thresh =1e-12)
plot_glmnet(port.ridge, label=20)
```



```
### Use lasso with CV
cv.lasso <- cv.glmnet(x, y, alpha = 1)
bestlam.lasso <- cv.lasso$lambda.min
bestlam.lasso
```

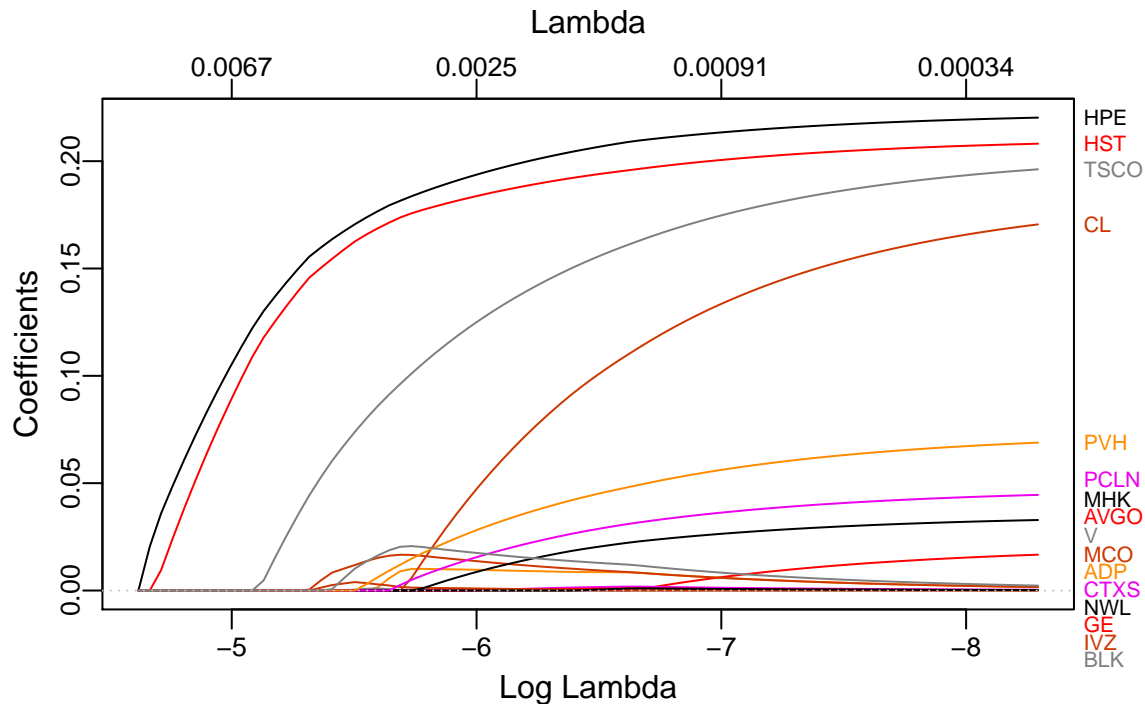
```
## [1] 0.0002745019
```

```
port.lasso =glmnet (x, y, alpha =1, thresh =1e-12)
lasso.coef=predict (port.lasso, type ="coefficients", s=bestlam.lasso )[0:502,]

lasso.coef[lasso.coef !=0]
```

```
## (Intercept)      ADP      AVGO      CL      CTXS
## 1.913808e-05 1.777291e-03 1.630887e-02 1.691821e-01 3.929118e-04
##          GE      HPE      HST      MCO      MHK
## 2.076306e-04 2.200270e-01 2.078506e-01 1.778065e-03 3.260303e-02
```

```
##          NWL          PCLN          PVH          TSCO          V
## 2.195027e-04 4.424482e-02 6.841851e-02 1.954139e-01 2.505628e-03
plot_glmnet(port.lasso, label=20)
```



The list of parameters that we find significant from lasso are: ADP + AVGO + CL + CTXS + GE + HPE + HST + MCO + MHK + NWL + PCLN + PVH + TSCO + V

Lets use these to perform linear regressions.

```
lm.fit=lm(portfolioreturns ~ ADP + AVGO + CL + CTXS + GE + HPE + HST + MCO + MHK + NWL + PCLN + PVH + TSCO + V)
summary(lm.fit)
```

```
## Warning in summary.lm(lm.fit): essentially perfect fit: summary may be
## unreliable
##
## Call:
## lm(formula = portfolioreturns ~ ADP + AVGO + CL + CTXS + GE +
##     HPE + HST + MCO + MHK + NWL + PCLN + PVH + TSCO + V, data = portWithReturnsAndStocks)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2.496e-17 -2.359e-18 -9.540e-19  5.220e-19  7.741e-17
##
## Coefficients:
##              Estimate Std. Error  t value Pr(>|t|)
## (Intercept) -2.512e-19  4.641e-19 -5.410e-01   0.589
## ADP          5.671e-16  6.831e-17  8.302e+00 7.85e-15 ***
```

```
## AVGO      2.066e-02  3.743e-17  5.519e+14  < 2e-16 ***
## CL        1.845e-01  6.587e-17  2.801e+15  < 2e-16 ***
## CTXS      3.996e-17  4.221e-17  9.470e-01    0.345
## GE        -6.976e-17  6.445e-17  -1.082e+00    0.280
## HPE       2.229e-01  2.767e-17  8.056e+15  < 2e-16 ***
## HST       2.110e-01  3.203e-17  6.588e+15  < 2e-16 ***
## MCO       -5.847e-17  4.912e-17  -1.190e+00    0.235
## MHK       3.527e-02  4.782e-17  7.375e+14  < 2e-16 ***
## NWL       -5.484e-17  4.257e-17  -1.288e+00    0.199
## PCLN      4.767e-02  3.429e-17  1.390e+15  < 2e-16 ***
## PVH       7.369e-02  2.768e-17  2.662e+15  < 2e-16 ***
## TSCO      2.043e-01  2.966e-17  6.888e+15  < 2e-16 ***
## V         1.359e-17  5.363e-17  2.530e-01    0.800
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 7.259e-18 on 237 degrees of freedom
## Multiple R-squared: 1, Adjusted R-squared: 1
## F-statistic: 5.138e+31 on 14 and 237 DF, p-value: < 2.2e-16
```

We get much better results in this case.

(c) Here is how the portfolio was created. We're essentially randomly sampling columns from `spreturns` to have non-zero coefficients (`portfoliowts`) and then generating returns from that.

```
t = runif(ncol(spreturns))
thresh = .98
mask = t > thresh
w = runif(ncol(spreturns))
sum(mask) # number of chosen coefficients
portfoliowts = w * mask / sum(w * mask)
myportfolioreturns = spreturns %*% portfoliowts
```

Write a function that takes a threshold as input and produces the total error for estimated weights from lasso to the true weights. To do this you will need (1) a function that generates weights and returns for a given threshold function, (2) a function that takes the the returns and outputs the estimated coefficients from a lasso, and (3) a function that takes the estimated coefficients and returns the error relative to the true weights. Plot the errors for a variety of different thresholds between 0.5 and 1.

```
generateWeightsAndReturns <- function(threshold) {
  t = runif(ncol(spreturns))
  mask = t > threshold
  w = runif(ncol(spreturns))
  portfoliowts = w * mask / sum(w * mask)
  list>Returns=spreturns %*% portfoliowts, Weights=portfoliowts)
}

estimateLassoCoef <- function(returns) {
  portWithReturnsAndStocks <- cbind(returns, spreturns)
  dim(portWithReturnsAndStocks)
  colnames(portWithReturnsAndStocks)[1] <- "portfolioreturns"
```

```

x=model.matrix (portfolioreturns ~ ., data = portWithReturnsAndStocks)[-1]
y=portWithReturnsAndStocks$portfolioreturns
cv.lasso <- cv.glmnet(x,y, alpha = 1)
bestlam.lasso <- cv.lasso$lambda.min

lasso.mod =glmnet (x,y,alpha =1)
lasso.coef <- predict(lasso.mod, type = 'coefficients', s = bestlam.lasso)
lasso.coef
}

# find squared differences
squaredDifferenceBetweenEstimatedAndTrueWeights <- function(estCoef, trueCoef) {
  # The est coef include the intercept that we have to remove
  sum((estCoef[2:502] - trueCoef)^2)
}

wrapper <- function(threshold) {
  weightsAndReturns = generateWeightsAndReturns(threshold)
  squaredDifferenceBetweenEstimatedAndTrueWeights(estimateLassoCoef(weightsAndReturns$Returns), weightsAndReturns$Weights)
}

emitID <- local({
  idCounter <- 0
  function(){
    idCounter <- idCounter + 1L # increment
    formatC(idCounter, width=9, flag=0, format="d") # format & return
  }
})

df <- data.frame(thresholds=numeric(0), differences=numeric(0))
lapply(runif(20, min = 0.5, max = 1), function(x) { df[emitID(),] <- c(x, wrapper(x)) } )

## [[1]]
## [1] 0.68537505 0.00872213
##
## [[2]]
## [1] 0.935989883 0.001428347
##
## [[3]]
## [1] 0.90038080 0.02484564
##
## [[4]]
## [1] 0.588642272 0.007901709
##
## [[5]]
## [1] 0.581638846 0.009328437
##
## [[6]]
## [1] 0.9722446 0.1313716
##
## [[7]]
## [1] 0.722318801 0.009222658
##
## [[8]]

```

```
## [1] 0.539196295 0.005704178
##
## [[9]]
## [1] 0.9927858121 0.0008026959
##
## [[10]]
## [1] 0.507919109 0.009190536
##
## [[11]]
## [1] 0.747441254 0.009499557
##
## [[12]]
## [1] 0.713313122 0.009444134
##
## [[13]]
## [1] 0.690032717 0.008709888
##
## [[14]]
## [1] 0.868472192 0.004661087
##
## [[15]]
## [1] 0.63976918 0.01002324
##
## [[16]]
## [1] 0.828141936 0.004796263
##
## [[17]]
## [1] 0.889863051 0.001697387
##
## [[18]]
## [1] 0.861026253 0.003509371
##
## [[19]]
## [1] 0.991203 0.300440
##
## [[20]]
## [1] 0.77166136 0.00844408
```

```
plot(df)
```

