Stat 897 Fall 2017 Data Analysis Assignment 7

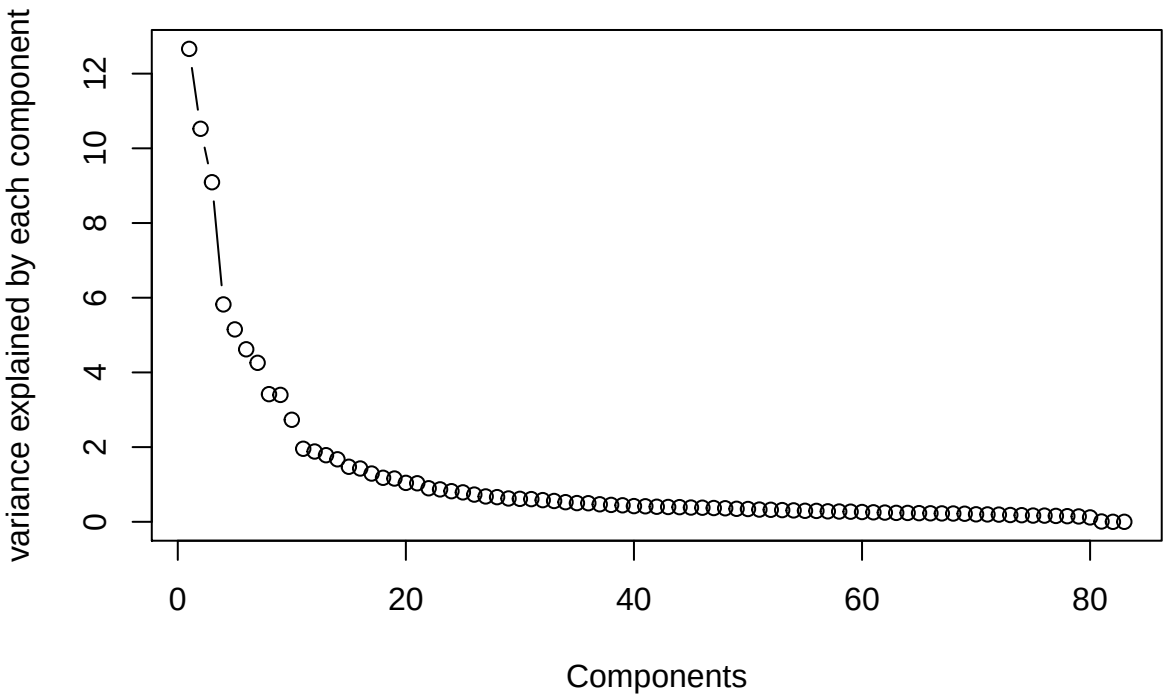
Penn State

Due October 15, 2017

In this assignment we will use the Khan data found in the ISLR library. This dataset contains 2308 gene expressions on 83 individuals (split into test and train) with a response variable of tumor measurement. We will use dimension reduction in prediction and compare results with shrinkage methods.

a) The data is alreay split into test and train in the `Khan` object. First, combine the xtrain and xtest datasets and run PCA on the combined predictor data set. Plot the components versus percentage variance explained.

```
suppressWarnings(library(ISLR))
data(Khan)
Khan$xall = rbind(Khan$xtrain, Khan$xtest)
pca = prcomp(Khan$xall, scale. = T)
plot(100 * pca$sdev ^ 2 / sum(pca$sdev ^ 2), type = 'b',
     ylab = 'variance explained by each component', xlab = 'Components')
```



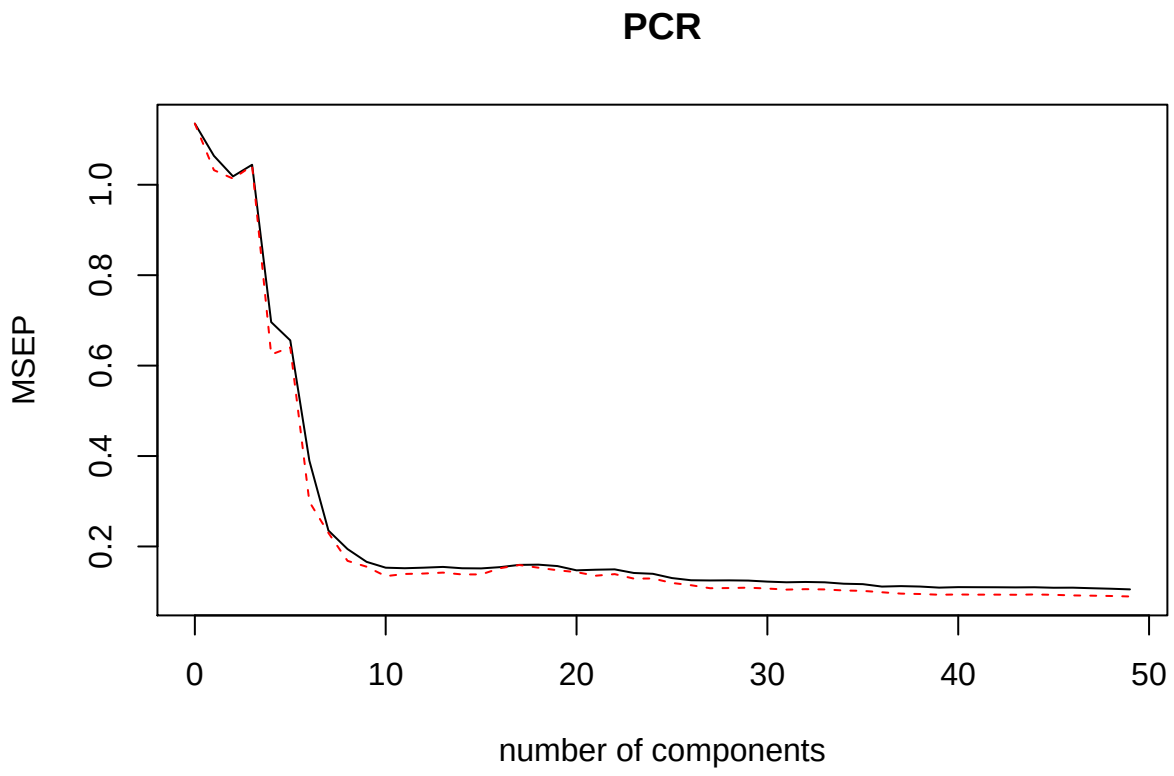
b) Fit a PCR model on the training set (you will need library `pls`), with M chosen by 5-fold cross-validation. Report the test mse obtained, along with the value of M selected by cross-validation. What is the percent variance explained on the combined predictor dataset for the number of components chosen here?

```
suppressWarnings(library(pls))
```

```
##
## Attaching package: 'pls'
```

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

```
set.seed(34)
pcr.fit = pcr(Khan$ytrain ~ Khan$xtrain, scale = TRUE, validation = 'CV', segments = 5)
### MSE vs number of PCs for PCR
validationplot(pcr.fit, val.type = "MSEP", main = "PCR")
```




```
#MSEP(pcr.fit)
### 49 components gives the smallest MSEP
M.pcr = which.min(MSEP(pcr.fit)$val[2, 1, ]) - 1
pcr.pred = predict(pcr.fit, newdata = Khan$xtest, ncomp = M.pcr)
pcr.test.error = mean((Khan$ytest - pcr.pred) ^ 2)
### M chosen by cross validation
M.pcr
```

```
## 49 comps
##       49
```

```
### test error of PCR
pcr.test.error
```

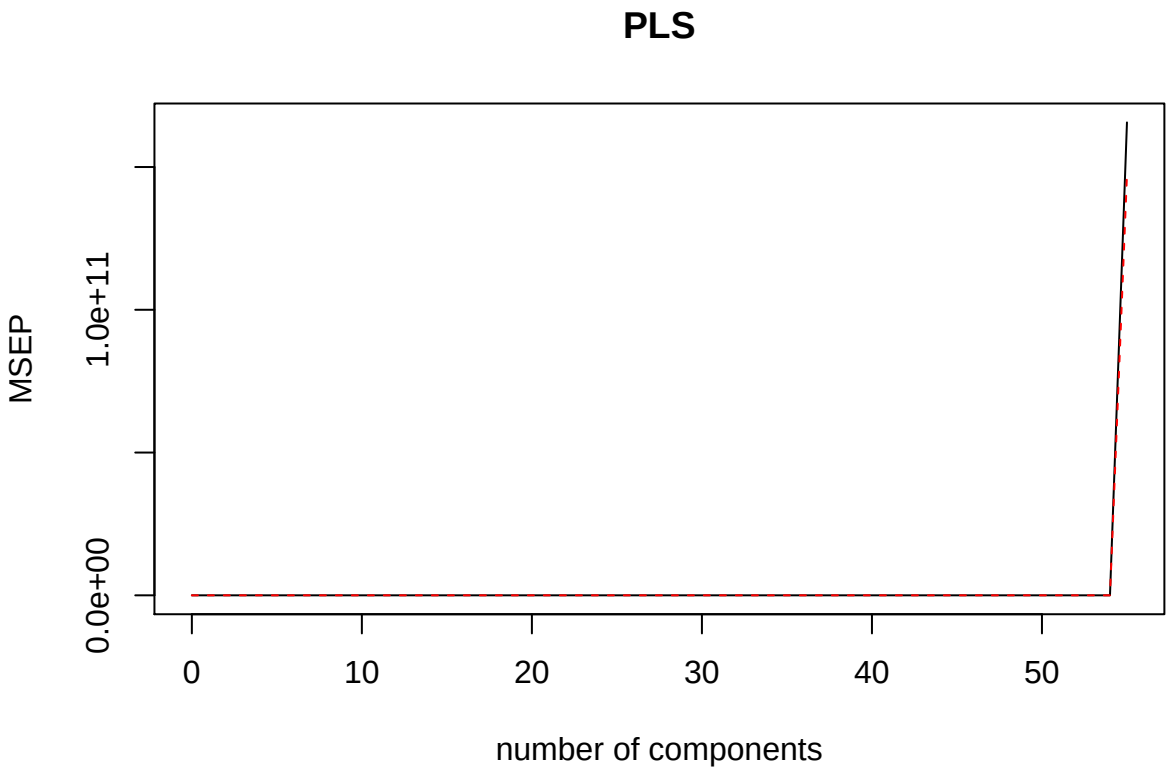
```
## [1] 0.1194346
```

```
## precent explained
100 * sum(pca$sdev[1:M.pcr] ^ 2) / sum(pca$sdev ^ 2)
```

```
## [1] 92.75518
```

c) Fit a PLS model on the training set, with M chosen by 5-fold cross-validation. Report the test mse obtained, along with the value of M selected by cross-validation. What is the percent variance explained on the combined predictor dataset for the number of components chosen here?

```
pls.fit = plsr(Khan$ytrain ~ Khan$xtrain, scale = TRUE, validation = 'CV')
### MSE vs number of PCs for PLS
validationplot(pls.fit, val.type = "MSEP", main = "PLS")
```




```
#MSEP(pls.fit)
### 10 components gives the smallest MSEP
M.pls = which.min(MSEP(pls.fit)$val[2, 1, ]) - 1 ## account for intercept only model
pls.pred = predict(pls.fit, newdata = Khan$xtest, ncomp = M.pls)
pls.test.error = mean((Khan$ytest - pls.pred) ^ 2)
### M chosen by cross validation
M.pls
```

```
## 32 comps
##       32
```

```
### test error of PLS
pls.test.error
```

```
## [1] 0.1229626
```

```
## precent explained
100 * sum(pca$sdev[1:M.pls] ^ 2) / sum(pca$sdev ^ 2)
```

```
## [1] 85.44352
```

d) Perform Lasso and Ridge with lambda chosen by 5-fold CV. Compute the test mse. How do your results compare to the PCR and PLS results? Which model would you prefer? (Think about both prediction quality and inference.)

```
suppressWarnings(library(glmnet, quietly = T))
```

```
## Loaded glmnet 2.0-5
```

```
suppressWarnings(library(pander))
cv.lasso = cv.glmnet(Khan$xtrain, Khan$ytrain, alpha = 1, nfolds = 5)
lam.lasso = cv.lasso$lambda.1se
### lambda chosen by 10-fold CV for lasso regresion
lam.lasso
```

```
## [1] 0.04531168
```

```
lasso.fit = glmnet(Khan$xtrain, Khan$ytrain, alpha = 1, lambda = lam.lasso)
lasso.coef = coef(lasso.fit)
lasso.ypred = predict(lasso.fit, newx = Khan$xtest)
lasso.mse = mean((Khan$ytest - lasso.ypred) ^ 2)
### coefficients of the lasso regression
bigcoef = lasso.coef[which(lasso.coef > 0.01), ]
pander(bigcoef)
```

Table 1: Table continues below

(Intercept)	V2	V129	V187	V558	V850	V867
3.042	0.02961	0.02307	0.02895	0.01628	0.03373	0.1202

Table 2: Table continues below

V910	V1003	V1055	V1066	V1105	V1194	V1207	V1301
0.03365	0.2076	0.01184	0.03178	0.0821	0.05353	0.185	0.01995

V1896	V1911	V1955	V2081
0.05852	0.01684	0.03953	0.02009

```
### test error of the lasso regression
lasso.mse
```

```
## [1] 0.1512201
```

```
cv.ridge = cv.glmnet(Khan$xtrain, Khan$ytrain, alpha = 0, nfolds = 5)
lam.ridge = cv.ridge$lambda.1se
### lambda chosen by 10-fold CV for ridge regresion
lam.ridge
```

```
## [1] 17.05956
```

```
ridge.fit = glmnet(Khan$xtrain, Khan$ytrain, alpha = 0, lambda = lam.ridge)
ridge.coef = coef(ridge.fit)
ridge.ypred = predict(ridge.fit, newx = Khan$xtest)
ridge.mse = mean((Khan$ytest - ridge.ypred) ^ 2)
### coefficients of the ridge regression
bigcoef = ridge.coef[which(ridge.coef > 0.01), ]
pander(bigcoef)
```

(Intercept)	V1610	V1647
3.137	0.01145	0.01017

```
### test error of the ridge regression
ridge.mse
```

```
## [1] 0.1731362
```