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 (already 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.

```
#install.packages('pls')
library(ISLR)
library(pls)
```

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

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

```
library(glmnet)
```

```
## Loading required package: Matrix
```

```
## Loading required package: foreach
```

```
## Loaded glmnet 2.0-12
```

```
data(Khan)
attach(Khan)
dim(Khan$xtrain)
```

```
## [1]   63 2308
```

```
dim(Khan$xtest)
```

```
## [1]   20 2308
```

```
length(Khan$ytrain)
```

```
## [1] 63
```

```
length(Khan$ytest)
```

```
## [1] 20
```

```
combined.x <- rbind(xtrain, xtest)
dim(combined.x)
```

```
## [1]   83 2308
```

```
combined.y <- c(ytrain, ytest)
length(combined.y)
```

```
## [1] 83
```

```
combined.data = cbind.data.frame(combined.x, y=c(combined.y))
```

```
## Warning in data.row.names(row.names, rowsi, i): some row.names duplicated:
## 64,65,66,67,68,69,70,71,72,73,74,75,76,77,78,79,80,81,82,83 --> row.names
## NOT used
```

```
dim(combined.data)
```

```
## [1]   83 2309
```

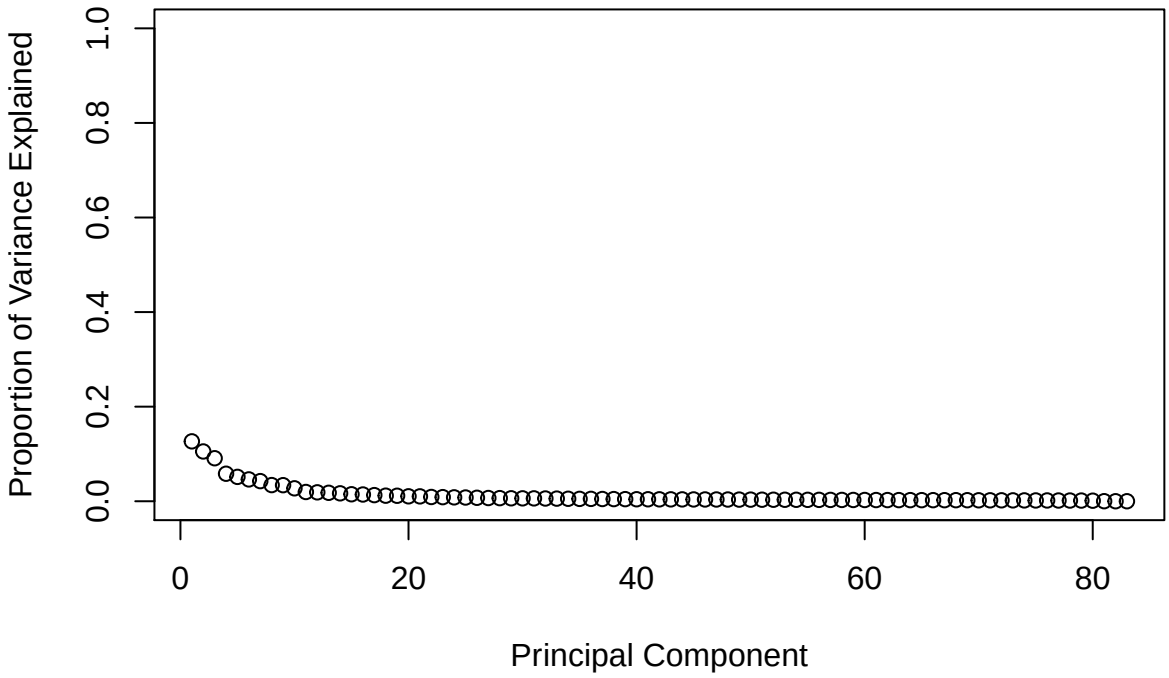
```
# check that we dont have null rows
sum(is.na(combined.data ))
```

```
## [1] 0
```

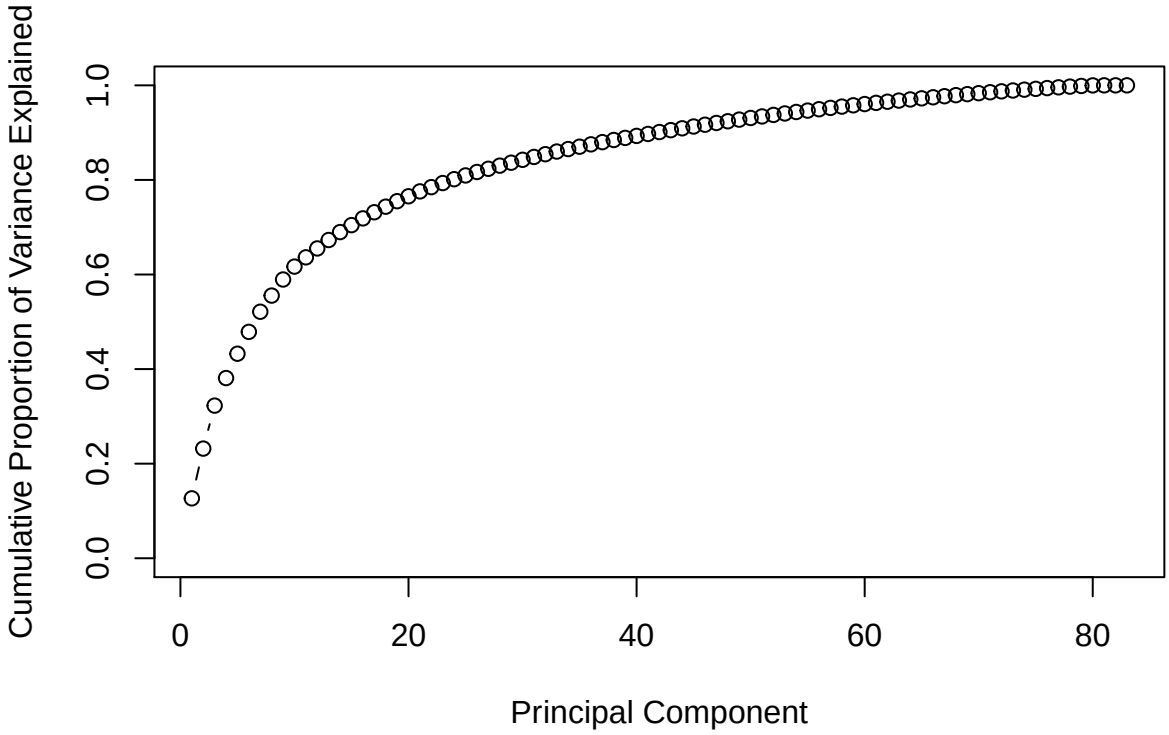
```
pr.out =prcomp (combined.x, scale. = TRUE)
pr.var =pr.out$sdev ^2
pve=pr.var/sum(pr.var)
pve
```

```
##  [1] 1.266252e-01 1.052379e-01 9.092753e-02 5.820361e-02 5.151983e-02
##  [6] 4.618788e-02 4.256964e-02 3.418087e-02 3.398202e-02 2.731900e-02
## [11] 1.955594e-02 1.882159e-02 1.783653e-02 1.673751e-02 1.473695e-02
## [16] 1.426834e-02 1.292415e-02 1.178657e-02 1.158539e-02 1.042219e-02
## [21] 1.031721e-02 8.982597e-03 8.681011e-03 8.196707e-03 7.883448e-03
## [26] 7.287455e-03 6.789752e-03 6.604350e-03 6.242117e-03 6.147680e-03
## [31] 6.073132e-03 5.801113e-03 5.573140e-03 5.249522e-03 5.006847e-03
## [36] 4.976833e-03 4.691174e-03 4.522583e-03 4.420575e-03 4.201096e-03
## [41] 4.154143e-03 4.034940e-03 3.964198e-03 3.925203e-03 3.820333e-03
## [46] 3.774622e-03 3.692536e-03 3.624177e-03 3.484710e-03 3.423489e-03
## [51] 3.275500e-03 3.219890e-03 3.130548e-03 3.057025e-03 2.944326e-03
## [56] 2.917823e-03 2.805291e-03 2.757048e-03 2.710547e-03 2.618016e-03
## [61] 2.542607e-03 2.467689e-03 2.392655e-03 2.374112e-03 2.315697e-03
## [66] 2.274818e-03 2.245763e-03 2.221023e-03 2.159843e-03 2.006610e-03
## [71] 1.978566e-03 1.927010e-03 1.820805e-03 1.770254e-03 1.667809e-03
## [76] 1.654841e-03 1.576376e-03 1.501168e-03 1.432417e-03 1.175906e-03
## [81] 8.222788e-05 4.529594e-07 2.497242e-32
```

```
plot(pve , xlab="Principal Component", ylab="Proportion of Variance Explained", ylim=c(0,1) ,type='b')
```




```
plot(cumsum (pve ), xlab="Principal Component", ylab ="Cumulative Proportion of Variance Explained",
     ylim=c(0,1), type='b')
```




```
# we can also use PCR to get the result
# pcr.fit=pcr(y~., data=combined.data ,scale=TRUE , validation ="CV")
# plot(pcr.fit$Xvar/pcr.fit$Xtotvar, xlab="Principal Component", ylab="Proportion of Variance Explained
```

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?

Please set the seed with "34"" here at the beginning of your code for part (b) and only here.

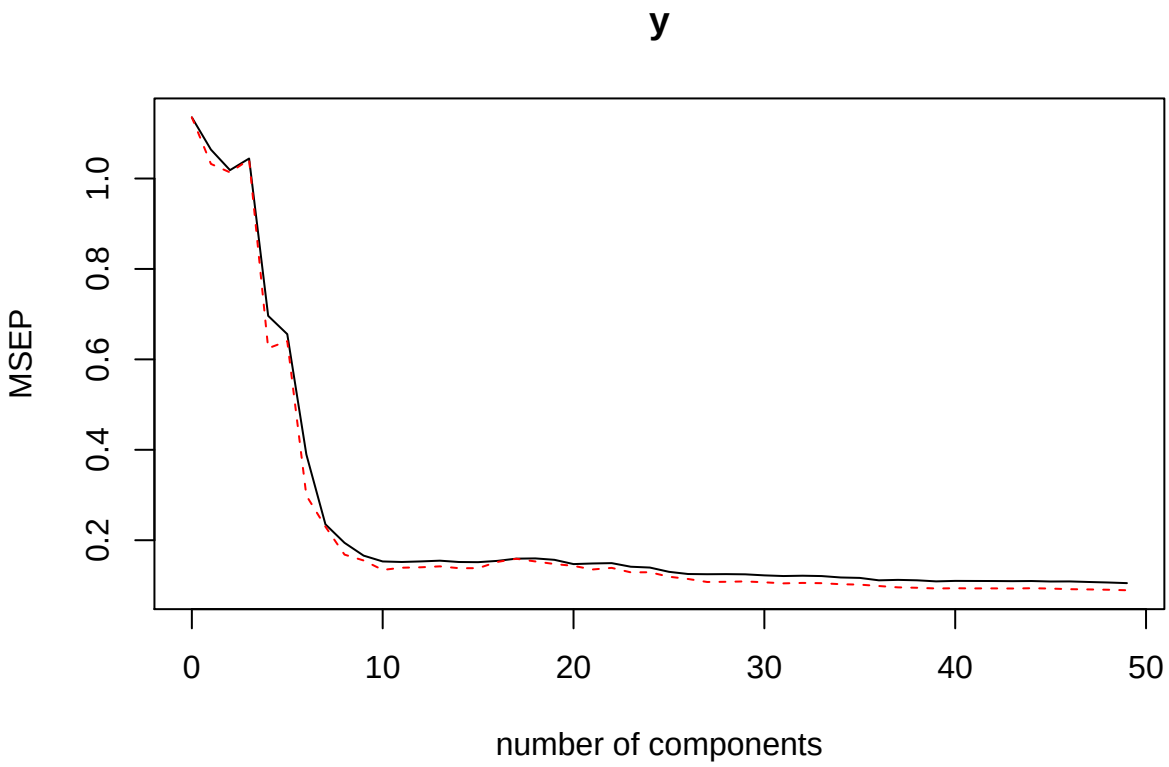
```
set.seed(34)
train = cbind.data.frame(xtrain, y=c(ytrain))
test = cbind.data.frame(xtest, y=c(ytest))

pcr.fit=pcr(y~., data=train, scale=TRUE , validation ="CV", segments=5)
summary(pcr.fit)
```

```
## Data:    X dimension: 63 2308
##  Y dimension: 63 1
## Fit method: svdpc
## Number of components considered: 49
##
## VALIDATION: RMSEP
## Cross-validated using 5 random segments.
```

```
##        (Intercept)  1 comps  2 comps  3 comps  4 comps  5 comps  6 comps
## CV           1.065    1.032    1.009    1.022   0.8344   0.8098   0.6242
## adjCV        1.065    1.016    1.007    1.021   0.7903   0.8003   0.5466
##        7 comps  8 comps  9 comps  10 comps  11 comps  12 comps  13 comps
## CV      0.4850   0.4408   0.4075    0.3912    0.3898    0.3914    0.3934
## adjCV   0.4796   0.4101   0.3940    0.3666    0.3731    0.3744    0.3771
##        14 comps  15 comps  16 comps  17 comps  18 comps  19 comps
## CV       0.3897    0.3892    0.3929    0.3991    0.3998    0.3958
## adjCV    0.3717    0.3718    0.3898    0.3988    0.3912    0.3839
##        20 comps  21 comps  22 comps  23 comps  24 comps  25 comps
## CV       0.3837    0.3855    0.3865    0.3761    0.3736    0.3608
## adjCV    0.3784    0.3678    0.3726    0.3591    0.3594    0.3455
##        26 comps  27 comps  28 comps  29 comps  30 comps  31 comps
## CV       0.3539    0.3533    0.3537    0.3529    0.3497    0.3475
## adjCV    0.3381    0.3281    0.3287    0.3300    0.3270    0.3234
##        32 comps  33 comps  34 comps  35 comps  36 comps  37 comps
## CV       0.3486    0.3475    0.3430    0.3416    0.3337    0.3351
## adjCV    0.3250    0.3240    0.3205    0.3191    0.3143    0.3095
##        38 comps  39 comps  40 comps  41 comps  42 comps  43 comps
## CV       0.3337    0.3302    0.3317    0.3315    0.3314    0.3309
## adjCV    0.3080    0.3054    0.3063    0.3057    0.3058    0.3053
##        44 comps  45 comps  46 comps  47 comps  48 comps  49 comps
## CV       0.3313    0.3297    0.3300    0.3282    0.3264    0.3242
## adjCV    0.3064    0.3051    0.3029    0.3020    0.3008    0.2990
##
## TRAINING: % variance explained
##    1 comps  2 comps  3 comps  4 comps  5 comps  6 comps  7 comps  8 comps
## X    12.90    25.22    32.86    38.73    44.11    49.13    53.35    56.86
## y    11.75    14.76    19.38    52.97    53.30    81.94    84.89    90.36
##    9 comps  10 comps  11 comps  12 comps  13 comps  14 comps  15 comps
## X    59.91     62.59     64.78     66.86     68.77     70.63     72.37
## y    90.81     92.79     92.97     93.06     93.06     93.26     93.27
##    16 comps  17 comps  18 comps  19 comps  20 comps  21 comps  22 comps
## X     74.01     75.58     76.95     78.15     79.33     80.45     81.45
## y     93.44     93.47     94.19     94.64     94.65     95.81     95.86
##    23 comps  24 comps  25 comps  26 comps  27 comps  28 comps  29 comps
## X     82.37     83.24     84.10     84.93     85.68     86.37     87.05
## y     96.39     96.44     96.89     97.27     97.93     97.94     97.94
##    30 comps  31 comps  32 comps  33 comps  34 comps  35 comps  36 comps
## X     87.72     88.34     88.95     89.53     90.10     90.64     91.17
## y     98.04     98.22     98.24     98.26     98.41     98.47     98.54
##    37 comps  38 comps  39 comps  40 comps  41 comps  42 comps  43 comps
## X     91.68     92.17     92.64     93.11     93.56     93.99     94.41
## y     98.99     99.07     99.13     99.21     99.25     99.26     99.27
##    44 comps  45 comps  46 comps  47 comps  48 comps  49 comps
## X     94.82     95.22     95.61     95.99     96.34     96.69
## y     99.30     99.31     99.51     99.51     99.51     99.52
```

```
validationplot(pcr.fit ,val.type="MSEP")
```




```
(which.min(MSEP(pcr.fit)$val[1,1,]))-1
```

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

The lowest CV is for 49 components. This amounts to no dimension reduction. However, from the plot we also see that the cross-validation error is roughly the same when only 30 components are included in the model. Visually from the plot we see that the elbow occurs at 10 components. These points suggests that a model that uses just a small number of components might suffice.

```
pcr.pred=predict (pcr.fit, xtest, ncomp =49)
mean((pcr.pred - ytest)^2)
```

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

The MSE with all the components (49) = 0.1194346

Let's also get the MSE with 30 and 10 components

```
pcr.pred=predict (pcr.fit, xtest, ncomp =30)
mean((pcr.pred - ytest)^2)
```

```
## [1] 0.120898
```

```
pcr.pred=predict (pcr.fit, xtest, ncomp =10)
mean((pcr.pred - ytest)^2)
```

```
## [1] 0.2284649
```

The test MSE with 30 components = 0.120898 The test MSE with 10 components = 0.2284649

We see test MSE with 10 components is quite higher than with 30/49 components.

The last part requires percent variance explained on the combined predictor dataset so we perform the pcr on the combined data set

```
pcr.fit.combined=pcr(y~., data=combined.data, scale=TRUE)
summary (pcr.fit.combined )
```

```
## Data:    X dimension: 83 2308
##  Y dimension: 83 1
## Fit method: svdpc
## Number of components considered: 82
## TRAINING: % variance explained
##    1 comps  2 comps  3 comps  4 comps  5 comps  6 comps  7 comps  8 comps
## X    12.66    23.19    32.28    38.10    43.25    47.87    52.13    55.55
## y    12.77    13.23    13.45    40.92    44.84    63.17    69.42    76.59
##    9 comps  10 comps  11 comps  12 comps  13 comps  14 comps  15 comps
## X    58.94     61.68     63.63     65.51     67.30     68.97     70.44
## y    89.74     89.94     90.06     91.49     92.38     92.60     92.64
##    16 comps  17 comps  18 comps  19 comps  20 comps  21 comps  22 comps
## X     71.87     73.16     74.34     75.50     76.54     77.57     78.47
## y     92.84     92.87     93.38     93.48     93.61     93.72     95.21
##    23 comps  24 comps  25 comps  26 comps  27 comps  28 comps  29 comps
## X     79.34     80.16     80.95     81.68     82.36     83.02     83.64
## y     95.49     96.07     96.15     96.16     96.26     96.41     96.69
##    30 comps  31 comps  32 comps  33 comps  34 comps  35 comps  36 comps
## X     84.26     84.86     85.44     86.00     86.53     87.03     87.52
## y     96.69     97.29     97.30     97.72     97.72     97.84     97.88
##    37 comps  38 comps  39 comps  40 comps  41 comps  42 comps  43 comps
## X     87.99     88.45     88.89     89.31     89.72     90.13     90.52
## y     97.93     97.94     98.01     98.22     98.24     98.27     98.36
##    44 comps  45 comps  46 comps  47 comps  48 comps  49 comps  50 comps
## X     90.92     91.30     91.68     92.04     92.41     92.76     93.10
## y     98.41     98.41     98.41     98.47     98.61     98.62     98.67
##    51 comps  52 comps  53 comps  54 comps  55 comps  56 comps  57 comps
## X     93.43     93.75     94.06     94.37     94.66     94.95     95.23
## y     98.71     98.76     98.84     98.85     98.95     99.13     99.14
##    58 comps  59 comps  60 comps  61 comps  62 comps  63 comps  64 comps
## X     95.51     95.78     96.04     96.30     96.54     96.78     97.02
## y     99.31     99.31     99.54     99.66     99.68     99.72     99.73
##    65 comps  66 comps  67 comps  68 comps  69 comps  70 comps  71 comps
## X     97.25     97.48     97.70     97.92     98.14     98.34     98.54
## y     99.81     99.81     99.86     99.87     99.91     99.91     99.92
##    72 comps  73 comps  74 comps  75 comps  76 comps  77 comps  78 comps
## X     98.73     98.91     99.09     99.26     99.42     99.58     99.73
## y     99.92     99.93     99.94     99.95     99.97     99.97     99.98
##    79 comps  80 comps  81 comps  82 comps
## X     99.87     99.99       100       100
## y    100.00    100.00       100       100
```

The % variance explained of parameters with all the components (49) = 92.76 The % variance explained of response with all the components (49) = 98.62

The % variance explained of parameters with 30 components = 84.26 The % variance explained of response with 30 components = 96.69

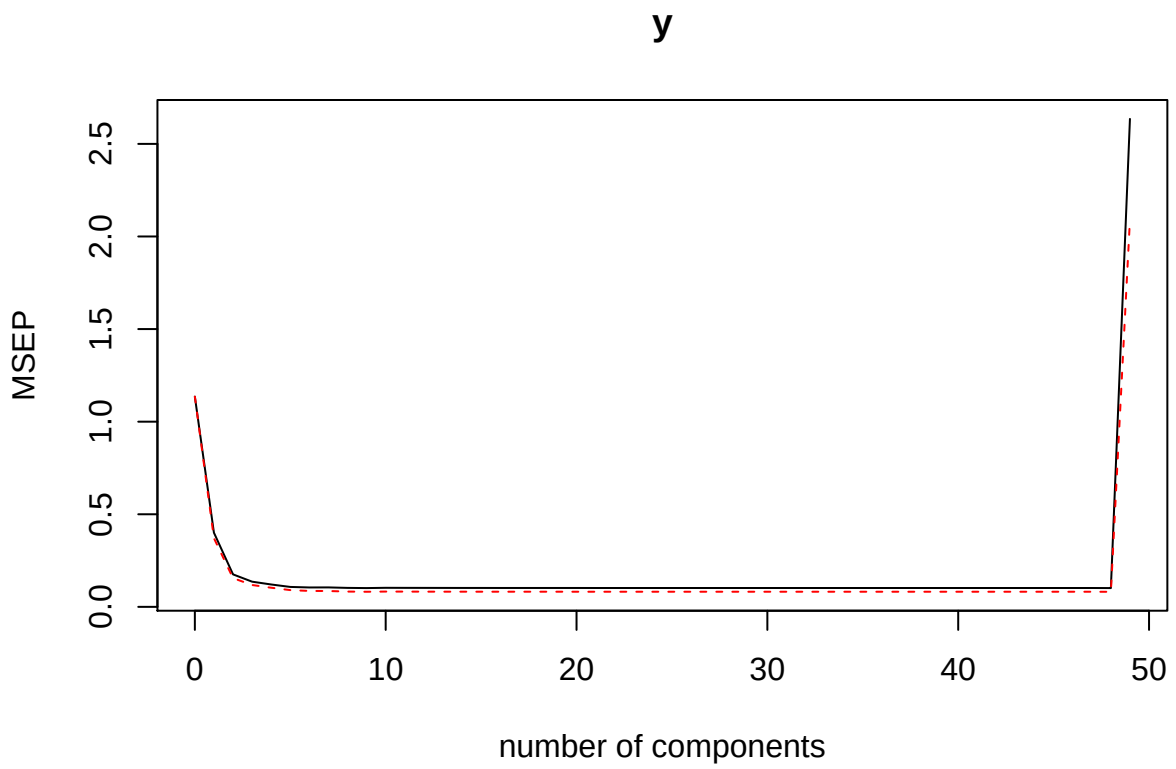
c) Fit a PLS model on the training set, with M chosen by 5-fold cross-validation. Report the test MSE obtained, along with the value of M selected by cross-validation. What is the percent variance explained on the combined predictor dataset for the number of components chosen here?

```
pls.fit=plsr(y~., data=train, scale=TRUE, validation="CV", segments=5)
summary (pls.fit )
```

```
## Data:    X dimension: 63 2308
##  Y dimension: 63 1
## Fit method: kernelpls
## Number of components considered: 49
##
## VALIDATION: RMSEP
## Cross-validated using 5 random segments.
##        (Intercept)  1 comps  2 comps  3 comps  4 comps  5 comps  6 comps
## CV           1.065   0.6329   0.4188   0.3688   0.3476   0.3280   0.3235
## adjCV        1.065   0.6090   0.3938   0.3434   0.3213   0.3007   0.2932
##        7 comps  8 comps  9 comps  10 comps  11 comps  12 comps  13 comps
## CV      0.3241   0.3199   0.3184    0.3207    0.3203    0.3200    0.3197
## adjCV   0.2926   0.2873   0.2855    0.2873    0.2869    0.2866    0.2863
##        14 comps  15 comps  16 comps  17 comps  18 comps  19 comps
## CV       0.3196    0.3195    0.3194    0.3193    0.3193    0.3193
## adjCV    0.2862    0.2861    0.2860    0.2859    0.2859    0.2859
##        20 comps  21 comps  22 comps  23 comps  24 comps  25 comps
## CV       0.3192    0.3191    0.3190    0.3190    0.3190    0.3190
## adjCV    0.2858    0.2858    0.2857    0.2857    0.2856    0.2856
##        26 comps  27 comps  28 comps  29 comps  30 comps  31 comps
## CV       0.3190    0.3190    0.3190    0.3190    0.3190    0.3190
## adjCV    0.2856    0.2856    0.2856    0.2856    0.2856    0.2856
##        32 comps  33 comps  34 comps  35 comps  36 comps  37 comps
## CV       0.3190    0.3190    0.3190    0.3190    0.3190    0.3190
## adjCV    0.2856    0.2856    0.2856    0.2856    0.2856    0.2856
##        38 comps  39 comps  40 comps  41 comps  42 comps  43 comps
## CV       0.3190    0.3190    0.3190    0.3190    0.3190    0.3190
## adjCV    0.2856    0.2856    0.2856    0.2856    0.2856    0.2856
##        44 comps  45 comps  46 comps  47 comps  48 comps  49 comps
## CV       0.3190    0.3190    0.3190    0.3190    0.3183     1.623
## adjCV    0.2856    0.2856    0.2856    0.2856    0.2851     1.446
##
## TRAINING: % variance explained
##    1 comps  2 comps  3 comps  4 comps  5 comps  6 comps  7 comps  8 comps
## X    9.307    18.11    23.36    30.25    37.86    42.89    46.86    49.38
## y   79.394    94.77    97.29    98.51    99.21    99.63    99.80    99.92
##    9 comps  10 comps  11 comps  12 comps  13 comps  14 comps  15 comps
## X    51.87     54.47     56.92     60.12     63.27     64.93     66.94
## y    99.97     99.99     99.99    100.00    100.00    100.00    100.00
##    16 comps  17 comps  18 comps  19 comps  20 comps  21 comps  22 comps
## X     68.61     69.87     71.04     72.17      73.4     74.58     75.81
## y    100.00    100.00    100.00    100.00     100.0    100.00    100.00
##    23 comps  24 comps  25 comps  26 comps  27 comps  28 comps  29 comps
## X     76.58     77.39     78.26     79.26     80.05     81.28      82.2
## y    100.00    100.00    100.00    100.00    100.00    100.00     100.0
##    30 comps  31 comps  32 comps  33 comps  34 comps  35 comps  36 comps
```

```
## X      83.21     84.34        85     85.75     86.31     87.12     87.85
## y     100.00    100.00       100    100.00    100.00    100.00    100.00
##     37 comps  38 comps  39 comps  40 comps  41 comps  42 comps  43 comps
## X      88.52     89.11      89.6     90.09     90.75     91.24     91.74
## y     100.00    100.00     100.0    100.00    100.00    100.00    100.00
##     44 comps  45 comps  46 comps  47 comps  48 comps  49 comps
## X      92.24     92.68     93.14     93.58     94.03     94.42
## y     100.00    100.00    100.00    100.00    100.00    100.00
```

```
validationplot(pls.fit ,val.type="MSEP")
```




```
(which.min(MSEP(pls.fit)$val[1,1,]))-1
```

```
## 48 comps
##       48
```

The lowest CV is for 48 components but it is almost equal to cv for 9 components. After 9 it only increases slightly and stays constant after 22 components till 48 before increasing drastically at 49 components. We will choose 9.

```
pls.pred=predict (pls.fit, xtest, ncomp =9)
mean((pls.pred - ytest)^2)
```

```
## [1] 0.1244835
```

```
pls.pred=predict (pls.fit, xtest, ncomp =22)
mean((pls.pred - ytest)^2)
```

```
## [1] 0.1230567
```

```
pls.pred=predict (pls.fit, xtest, ncomp =48)
mean((pls.pred - ytest)^2)
```

```
## [1] 0.1229627
```

The test MSE with 9 components = 0.1244835 The test MSE with 22 components = 0.1230567 The test MSE with 48 components = 0.1229627

The last part requires percent variance explained on the combined predictor dataset so we perform the pls on the combined data set

```
pls.fit.combined=plsr(y~., data=combined.data, scale=TRUE)
summary (pls.fit.combined )
```

```
## Data:    X dimension: 83 2308
##  Y dimension: 83 1
## Fit method: kernelpls
## Number of components considered: 82
## TRAINING: % variance explained
##    1 comps  2 comps  3 comps  4 comps  5 comps  6 comps  7 comps  8 comps
## X    9.139    17.38    22.09    28.99    33.88    38.93    45.36    48.37
## y   74.567    93.14    96.47    97.75    98.93    99.37    99.58    99.80
##    9 comps  10 comps  11 comps  12 comps  13 comps  14 comps  15 comps
## X    50.98     54.07     56.66     59.37     61.51     63.01     64.42
## y    99.91     99.96     99.98     99.99    100.00    100.00    100.00
##    16 comps  17 comps  18 comps  19 comps  20 comps  21 comps  22 comps
## X     66.73     68.36     69.34     70.37     71.53      72.6     73.52
## y    100.00    100.00    100.00    100.00    100.00     100.0    100.00
##    23 comps  24 comps  25 comps  26 comps  27 comps  28 comps  29 comps
## X     74.68     75.53     76.28     77.07     77.89     78.49     79.36
## y    100.00    100.00    100.00    100.00    100.00    100.00    100.00
##    30 comps  31 comps  32 comps  33 comps  34 comps  35 comps  36 comps
## X     79.96      80.5     81.27     81.92     82.52     83.09     83.74
## y    100.00     100.0    100.00    100.00    100.00    100.00    100.00
##    37 comps  38 comps  39 comps  40 comps  41 comps  42 comps  43 comps
## X      84.4     84.91     85.42     85.75     86.12     86.54     86.97
## y     100.0    100.00    100.00    100.00    100.00    100.00    100.00
##    44 comps  45 comps  46 comps  47 comps  48 comps  49 comps  50 comps
## X     87.44     87.82     88.25     88.69     89.05     89.41     89.83
## y    100.00    100.00    100.00    100.00    100.00    100.00    100.00
##    51 comps  52 comps  53 comps  54 comps  55 comps  56 comps  57 comps
## X     90.22     90.57     90.96     91.36     91.74     92.17     92.48
## y    100.00    100.00    100.00    100.00    100.00    100.00    100.00
##    58 comps  59 comps  60 comps  61 comps  62 comps  63 comps  64 comps
## X     92.84     93.18     93.49     93.81     94.14     94.46     94.74
## y    100.00    100.00    100.00    100.00    100.00    100.00    100.00
##    65 comps  66 comps  67 comps  68 comps  69 comps  70 comps  71 comps
## X     95.05     95.36     95.67     95.96     96.25     96.57     96.88
## y    100.00    100.00    100.00    100.00    100.00    100.00    100.00
##    72 comps  73 comps  74 comps  75 comps  76 comps  77 comps  78 comps
## X     97.16     97.47     97.75     98.06     98.33     98.59     98.88
## y    100.00    100.00    100.00    100.00    100.00    100.00    100.00
##    79 comps  80 comps  81 comps  82 comps
## X     99.13     99.38     99.62     99.87
## y    100.00    100.00    100.00    100.00
```

The % variance explained of parameters with 9 components = 50.98 The % variance explained of response with 9 components = 99.91

The % variance explained of parameters with 22 components = 73.52 The % variance explained of response with 22 components = 100

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.)

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

cv.ridge <- cv.glmnet(train.mat, train$y, alpha = 0, nfolds = 5)
bestlam.ridge <- cv.ridge$lambda.min
bestlam.ridge
```

```
## [1] 9.762105
```

We see that the value of lambda a that results in the smallest crossvalidation error is 9.762105

The test MSE associated with this value of lambda is:

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

```
## [1] 0.1524335
```

```
ridge.coef  <- predict(fit.ridge, type = 'coefficients', s = bestlam.ridge)
length(ridge.coef[ridge.coef !=0])
```

```
## <sparse>[ <logic> ] : .M.sub.i.logical() maybe inefficient
```

```
## [1] 2309
```

The test MSE with ridge is: 0.1524335

Now we will try Lasso

```
# having an issue where values change in the PDF
set.seed(34)
cv.lasso <- cv.glmnet(train.mat, train$y, alpha = 1, nfolds = 5)
bestlam.lasso <- cv.lasso$lambda.min
bestlam.lasso
```

```
## [1] 0.01705956
```

We see that the value of lambda a that results in the smallest crossvalidation error is 0.01705956

The test MSE associated with this value of lambda is:

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

```
## [1] 0.196568
```

The test MSE with lasso is: 0.196568

To get the coefficients in order to get an idea of how many parameters are used in a lasso model, we will first fit a model with the entire dataset.

```
out=glmnet (combined.x, combined.y, alpha =1, lambda =grid)
lasso.coef  <- predict(out, type = 'coefficients', s = bestlam.lasso)
length(lasso.coef[lasso.coef !=0])
```

```
## <sparse>[ <logic> ] : .M.sub.i.logical() maybe inefficient
```

```
## [1] 70
```

Let's now tabulate the results

The test MSE with the different methods: PCR (49): 0.119 PCR (30): 0.121 PLS (9): 0.124 Ridge: 0.152 Lasso: 0.197

The PCR at 49 components even though with the lowest test MSE is effectively same as least squares. Ridge also uses all the parameters as expected and still has higher MSE than others. Therefore the choice comes between Lasso and PLS with 9 components. The lasso has a clear advantage that it reduces the number of components to 66. While this number appears much higher than the PLS with 9 components - this isn't an apples to apples comparison. The PLS (and same for PCR) components are linear combinations of all the components and therefore effectively still use all components. On the other hand PLS with 9 components has much lesser test MSE as compared to Lasso.

Therefore in conclusion we should use lasso for its interpretability and PLS for its lowest test MSE. The choice between the two should be made in the context of research and its final usage.