ass1

Mike

2024-08-02

Answer 1

From the result below, we just need to select one principal component $PC1$ which achieve variance 0.9384 and its pca scores is shown below.

```
data<-read.table('prostate 14.37.25.xls')
train_data<-data[data$train==TRUE,]
test_data<-data[data$train!="TRUE",]
y_train<-train_data[,c(9)]
y_test<-test_data[,c(9)]
x_train<-train_data[,-c(9,10)]
x_test<-test_data[,-c(9,10)]
pca<-prcomp(x_train)
summary(pca)
```

```
## Importance of components:
##                            PC1     PC2     PC3     PC4     PC5     PC6     PC7
## Standard deviation     29.4060 7.21172 1.41079 1.34762 0.68256 0.46478 0.37463
## Proportion of Variance  0.9384 0.05644 0.00216 0.00197 0.00051 0.00023 0.00015
## Cumulative Proportion   0.9384 0.99489 0.99705 0.99902 0.99952 0.99976 0.99991
##                            PC8
## Standard deviation     0.28713
## Proportion of Variance 0.00009
## Cumulative Proportion  1.00000
```

```
partition<-c('\n --------------------partition','--------------------------',' \n ','\n')
# cat(partition,"scores:")
print(head(pca$x[,1]))#score
```

```
##          1          2          3          4          5          6
## -27.365833 -26.774435  -5.622874 -26.778802 -26.438735 -27.374914
```

```
# cat(partition,"loading:")
print(pca$rotation)#loading
```

```
##                  PC1           PC2          PC3           PC4          PC5
## lcavol   0.020591221  0.0265387132 -0.142232054 -6.956528e-01  0.684585338
## lweight  0.001299867  0.0207058403  0.103780226 -1.264851e-01  0.026607683
## age      0.074902045  0.9943389702 -0.060160693  3.863294e-02 -0.017168666
## lbph    -0.001235501  0.0651013954  0.945448635 -2.720302e-01 -0.095182089
## svi      0.006885640 -0.0007646719 -0.079766507 -1.459095e-01 -0.036343550
## lcp      0.031602066 -0.0042692119 -0.254650547 -6.342199e-01 -0.707680710
## gleason  0.018323565  0.0147525919  0.002410873  7.443879e-05  0.138190610
## pgg45    0.996283400 -0.0753812694  0.017084139  3.242546e-02  0.007146172
##                  PC6           PC7          PC8
```

```
## lcavol   0.022657297  0.136430921  0.0832174094
## lweight  0.480664775 -0.847251637  0.1523772950
## age      0.007330385  0.014801980 -0.0005358024
## lbph    -0.067988418  0.107010938 -0.0524723579
## svi      0.150176810 -0.078518367 -0.9707011123
## lcp     -0.124892771 -0.007299128  0.1242492955
## gleason -0.851666858 -0.495591217 -0.0969384035
## pgg45    0.016856627  0.007194603  0.0026069949
```

```
cat(partition,"scores:")
```

```
##
##  --------------------partition --------------------------
##
##  scores:
```

```
pca$x[,1]
```

```
##           1           2           3           4           5           6
## -27.3658333 -26.7744355  -5.6228736 -26.7788017 -26.4387353 -27.3749141
##           8          11          12          13          14          16
## -26.7430156 -26.2240350 -26.4101056   3.5851268 -21.0501072 -26.1233480
##          17          18          19          20          21          23
##   4.0650538 -26.0516430 -28.0389396 -25.8544879 -26.6553045 -26.6909302
##          24          27          29          30          31          33
##  33.5315378  43.5581682  53.7056873 -26.0849719 -26.2083372 -25.7576197
##          35          37          38          39          40          41
## -26.3612176 -10.5245747 -11.3337848   9.0442867 -21.8521984  53.1660318
##          43          45          46          47          51          52
## -26.4152897  -6.1537142 -11.5347973  74.6899589  23.8509116 -26.2734507
##          56          58          59          60          61          63
## -11.1691700 -27.4214246  -5.9011175  13.3596343 -25.6240337  69.1470322
##          67          68          69          70          71          72
##  43.8772631 -15.6523270 -25.9378142 -20.6651219  33.3223740  -0.3853749
##          75          76          77          78          79          81
##  -5.8125617  24.0176933  34.1881491 -15.2257520  44.0153495  13.7932721
##          82          83          85          86          87          88
##  33.1335246   4.6978662   3.4009741  33.6832883 -26.7154841   3.5483716
##          89          90          91          92          93          94
##  33.7538671  49.4399647 -25.9370762 -11.5105537  33.9418535  12.2657399
##          96
##  53.8722976
```

```
cat(partition,"loading:")
```

```
##
##  --------------------partition --------------------------
##
##  loading:
```

```
pca$rotation[,1]
```

```
##       lcavol      lweight          age         lbph          svi          lcp
##  0.020591221  0.001299867  0.074902045 -0.001235501  0.006885640  0.031602066
##      gleason        pgg45
##  0.018323565  0.996283400
```

Answer2

It is a simple linear regression according to Answer 1.After applying the rotation(linear combination) got in Answer 1 to training data as well as the testing data ,we can get the estimated intercept as 2.452345, estimated slope as 0.018513 and testing MSE as 1.056794.

```
x_pca<-pca$x[,1]
model<-lm(y_train~ x_pca)
summary(model)
```

```
##
## Call:
## lm(formula = y_train ~ x_pca)
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -2.51077 -0.69819  0.08337  0.80422  2.05763
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 2.452345   0.132729  18.476  < 2e-16 ***
## x_pca       0.018513   0.004548   4.071 0.000129 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.086 on 65 degrees of freedom
## Multiple R-squared:  0.2032, Adjusted R-squared:  0.1909
## F-statistic: 16.57 on 1 and 65 DF,  p-value: 0.0001294
```

```
x_p<-as.matrix(x_test)%*%pca$rotation[,1] # under pca rotation transform test predictor
y_p<-predict(model,data.frame(x_pca=x_p))
mse<-mean((y_test-y_p)^2)
cat(partition,"MSE")
```

```
##
##  --------------------partition --------------------------
##
##  MSE
```

```
mse
```

```
## [1] 1.056794
```

Answer 3

Simply applying glmnet package to do ridge estimation and the lasso estimation. Nearly the best λ is achieved by 0.09 and 0.0013 respectively.MSE is 0.4940807 and 0.5172441 respectively.

```
library(glmnet)
```

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

```
## Loaded glmnet 4.1-8
```

```
y_train<-as.matrix(y_train)
x_train<-as.matrix(x_train)
y_test<-as.matrix(y_test)
x_test<-as.matrix(x_test)
model_ridge<-glmnet(x_train,y_train,alpha = 0,nlambda = 120)
```

```r
model_lasso<-glmnet(x_train,y_train,alpha = 1)
print(model_ridge)
```

```
##
## Call:  glmnet(x = x_train, y = y_train, alpha = 0, nlambda = 120)
##
##      Df  %Dev Lambda
## 1    8  0.00 878.90
## 2    8  0.52 813.40
## 3    8  0.56 752.80
## 4    8  0.60 696.80
## 5    8  0.65 644.90
## 6    8  0.70 596.80
## 7    8  0.76 552.40
## 8    8  0.82 511.30
## 9    8  0.88 473.20
## 10   8  0.95 437.90
## 11   8  1.03 405.30
## 12   8  1.11 375.10
## 13   8  1.20 347.20
## 14   8  1.29 321.30
## 15   8  1.39 297.40
## 16   8  1.50 275.30
## 17   8  1.62 254.80
## 18   8  1.75 235.80
## 19   8  1.89 218.20
## 20   8  2.03 202.00
## 21   8  2.19 186.90
## 22   8  2.36 173.00
## 23   8  2.55 160.10
## 24   8  2.74 148.20
## 25   8  2.96 137.20
## 26   8  3.18 126.90
## 27   8  3.43 117.50
## 28   8  3.69 108.70
## 29   8  3.97 100.60
## 30   8  4.27  93.14
## 31   8  4.59  86.20
## 32   8  4.94  79.78
## 33   8  5.31  73.84
## 34   8  5.70  68.34
## 35   8  6.12  63.25
## 36   8  6.57  58.54
## 37   8  7.05  54.18
## 38   8  7.56  50.15
## 39   8  8.10  46.41
## 40   8  8.68  42.95
## 41   8  9.29  39.76
## 42   8  9.93  36.79
## 43   8 10.61  34.05
## 44   8 11.34  31.52
## 45   8 12.10  29.17
## 46   8 12.90  27.00
## 47   8 13.74  24.99
```

```
## 48    8 14.62  23.13
## 49    8 15.55  21.40
## 50    8 16.51  19.81
## 51    8 17.52  18.33
## 52    8 18.57  16.97
## 53    8 19.65  15.70
## 54    8 20.78  14.54
## 55    8 21.94  13.45
## 56    8 23.14  12.45
## 57    8 24.36  11.52
## 58    8 25.62  10.67
## 59    8 26.90   9.87
## 60    8 28.21   9.14
## 61    8 29.53   8.46
## 62    8 30.87   7.82
## 63    8 32.23   7.24
## 64    8 33.58   6.70
## 65    8 34.95   6.20
## 66    8 36.30   5.74
## 67    8 37.66   5.31
## 68    8 39.00   4.92
## 69    8 40.33   4.55
## 70    8 41.63   4.21
## 71    8 42.92   3.90
## 72    8 44.18   3.61
## 73    8 45.41   3.34
## 74    8 46.61   3.09
## 75    8 47.78   2.86
## 76    8 48.91   2.65
## 77    8 50.01   2.45
## 78    8 51.06   2.27
## 79    8 52.08   2.10
## 80    8 53.05   1.94
## 81    8 53.99   1.80
## 82    8 54.89   1.66
## 83    8 55.74   1.54
## 84    8 56.56   1.43
## 85    8 57.34   1.32
## 86    8 58.09   1.22
## 87    8 58.79   1.13
## 88    8 59.47   1.05
## 89    8 60.11   0.97
## 90    8 60.71   0.90
## 91    8 61.29   0.83
## 92    8 61.83   0.77
## 93    8 62.35   0.71
## 94    8 62.84   0.66
## 95    8 63.30   0.61
## 96    8 63.74   0.56
## 97    8 64.16   0.52
## 98    8 64.55   0.48
## 99    8 64.91   0.45
## 100   8 65.26   0.41
## 101   8 65.59   0.38
```

```
## 102  8 65.90   0.35
## 103  8 66.18   0.33
## 104  8 66.46   0.30
## 105  8 66.71   0.28
## 106  8 66.94   0.26
## 107  8 67.17   0.24
## 108  8 67.37   0.22
## 109  8 67.56   0.21
## 110  8 67.74   0.19
## 111  8 67.90   0.18
## 112  8 68.05   0.16
## 113  8 68.19   0.15
## 114  8 68.31   0.14
## 115  8 68.43   0.13
## 116  8 68.54   0.12
## 117  8 68.63   0.11
## 118  8 68.72   0.10
## 119  8 68.80   0.09
## 120  8 68.87   0.09
```

```r
cat(partition,"ridge_coefficents:")
```

```
##
##  --------------------partition --------------------------
##
##  ridge_coefficents:
```

```r
coef(model_ridge,s=0.09)
```

```
## 9 x 1 sparse Matrix of class "dgCMatrix"
##                       s1
## (Intercept)  0.090110627
## lcavol       0.491074013
## lweight      0.600889206
## age         -0.014738172
## lbph         0.137794826
## svi          0.677948910
## lcp         -0.115072918
## gleason      0.018028062
## pgg45        0.007040325
```

```r
print(model_lasso)
```

```
##
## Call:  glmnet(x = x_train, y = y_train, alpha = 1)
##
##    Df  %Dev  Lambda
## 1   0   0.00 0.87890
## 2   1   9.13 0.80080
## 3   1  16.70 0.72970
## 4   1  22.99 0.66480
## 5   1  28.22 0.60580
## 6   1  32.55 0.55200
## 7   1  36.15 0.50290
## 8   1  39.14 0.45820
## 9   2  42.81 0.41750
```

```
## 10  2 45.98 0.38040
## 11  3 48.77 0.34660
## 12  3 51.31 0.31590
## 13  3 53.42 0.28780
## 14  3 55.18 0.26220
## 15  3 56.63 0.23890
## 16  3 57.84 0.21770
## 17  5 59.17 0.19840
## 18  5 60.45 0.18070
## 19  5 61.51 0.16470
## 20  5 62.39 0.15010
## 21  5 63.12 0.13670
## 22  5 63.72 0.12460
## 23  5 64.23 0.11350
## 24  5 64.65 0.10340
## 25  5 64.99 0.09424
## 26  5 65.28 0.08587
## 27  5 65.52 0.07824
## 28  5 65.72 0.07129
## 29  5 65.89 0.06496
## 30  6 66.05 0.05919
## 31  6 66.32 0.05393
## 32  6 66.53 0.04914
## 33  7 66.76 0.04477
## 34  7 67.21 0.04079
## 35  7 67.59 0.03717
## 36  7 67.90 0.03387
## 37  7 68.16 0.03086
## 38  7 68.37 0.02812
## 39  7 68.55 0.02562
## 40  7 68.70 0.02334
## 41  7 68.82 0.02127
## 42  7 68.93 0.01938
## 43  7 69.01 0.01766
## 44  7 69.08 0.01609
## 45  7 69.14 0.01466
## 46  7 69.19 0.01336
## 47  7 69.23 0.01217
## 48  7 69.26 0.01109
## 49  7 69.29 0.01010
## 50  7 69.31 0.00921
## 51  7 69.33 0.00839
## 52  7 69.35 0.00764
## 53  7 69.36 0.00696
## 54  7 69.37 0.00635
## 55  7 69.38 0.00578
## 56  7 69.39 0.00527
## 57  8 69.39 0.00480
## 58  8 69.40 0.00437
## 59  8 69.41 0.00399
## 60  8 69.41 0.00363
## 61  8 69.42 0.00331
## 62  8 69.42 0.00301
## 63  8 69.42 0.00275
```

```
## 64  8 69.43 0.00250
## 65  8 69.43 0.00228
## 66  8 69.43 0.00208
## 67  8 69.43 0.00189
## 68  8 69.43 0.00172
## 69  8 69.43 0.00157
## 70  8 69.43 0.00143
## 71  8 69.43 0.00130
```

```
cat(partition,"lasso_coefficents:")
```

```
##
##  --------------------partition --------------------------
##
##  lasso_coefficents:
```

```
coef(model_lasso,s=0.0013)
```

```
## 9 x 1 sparse Matrix of class "dgCMatrix"
##                       s1
## (Intercept)  0.370504240
## lcavol       0.572828510
## lweight      0.613240087
## age         -0.018695828
## lbph         0.143783872
## svi          0.731284287
## lcp         -0.199930956
## gleason     -0.020960257
## pgg45        0.009156105
```

```
mse_ridge<-mean((y_test-predict(model_ridge,x_test,s=0.09))^2)
mse_lasso<-mean((y_test-predict(model_lasso,x_test,s=0.0013))^2)
cat(partition,"ridge_mse:")
```

```
##
##  --------------------partition --------------------------
##
##  ridge_mse:
```

```
mse_ridge
```

```
## [1] 0.4940807
```

```
cat(partition,"lasso_mse:")
```

```
##
##  --------------------partition --------------------------
##
##  lasso_mse:
```

```
mse_lasso
```

```
## [1] 0.5172441
```

Answer 4

Clearly the ridge regression achieve the least mse and the both mse in Answer 3 is lower than mse in Answer 2. Intuitively, we could consider the Answer 2 as a linear regression without regularization and it is not the

OLS estimation thus have large mse while OLS emstimation of linear regression is one of the special cases of ridge regression model or the lasso regression with $\lambda = 0$ thus must higher than the optimal case.

However,why ridge perform better than lasso is hard to explain which I think is caused by randomness.

Answer 5

According to the lecture note, under true model setup ,we could derive the estimated variance by formula below

$$\sigma^2 (X^T X + \lambda n I)^{-1} (X^T X)(X^T X + \lambda n I)^{-1}$$

where X is the training covarites' matrix and σ need to be further approximated by follows.(concern: since the bais is from training data and mse is from test data may occurring variance to be negative.)

$$\sigma^2 = Var(\hat{\beta}) = MSE - Bias^2(\hat{\beta}) \approx MSE_{test} - ([n\lambda(X^T X + n\lambda I)\hat{\beta}]^{-1})^T [n\lambda(X^T X + n\lambda I)\hat{\beta}]^{-1}$$

```
# any way to get the variance which is the sum of all variance of the estimated coefficients is accordi

n<-nrow(x_train)
w<-solve(t(x_train)%*%(x_train)+0.09*n*diag(8))
sigma_sq<-mse_ridge-t(n*0.09*w)%*%(n*0.09*w)
cat(partition,"Sigma_squared:",'\n')
```

```
##
##  --------------------partition -------------------------
##
##  Sigma_squared:
```

```
sigma_sq
```

```
##            lcavol   lweight       age      lbph       svi       lcp   gleason
## lcavol  0.4770562 0.4983827 0.4940786 0.4962432 0.5170989 0.5035885 0.4951956
## lweight 0.4983827 0.4018246 0.4959390 0.5049963 0.5241374 0.4935150 0.5271048
## age     0.4940786 0.4959390 0.4938381 0.4941677 0.4941213 0.4940064 0.4954803
## lbph    0.4962432 0.5049963 0.4941677 0.4901028 0.4810243 0.4933781 0.4867009
## svi     0.5170989 0.5241374 0.4941213 0.4810243 0.2313725 0.5279705 0.4811686
## lcp     0.5035885 0.4935150 0.4940064 0.4933781 0.5279705 0.4738530 0.4903443
## gleason 0.4951956 0.5271048 0.4954803 0.4867009 0.4811686 0.4903443 0.4602029
## pgg45   0.4939410 0.4932274 0.4940678 0.4942587 0.4945323 0.4943689 0.4947637
##             pgg45
## lcavol  0.4939410
## lweight 0.4932274
## age     0.4940678
## lbph    0.4942587
## svi     0.4945323
## lcp     0.4943689
## gleason 0.4947637
## pgg45   0.4940607
```

```
cov_m<-w*t(x_train)%*%(x_train)*w*sigma_sq
cat(partition,"Covariance matrix:",'\n')
```

```
##
##  --------------------partition -------------------------
##
##  Covariance matrix:
```

```r
cov_m
```

```
##               lcavol       lweight           age          lbph           svi
## lcavol  3.433054e-02  1.154355e-03  9.589139e-05  1.175543e-05  1.061792e-03
## lweight 1.154355e-03  8.421804e-01  1.894708e-02  3.331749e-04  1.105061e-03
## age     9.589139e-05  1.894708e-02  1.239955e-02  5.422180e-06  1.088674e-06
## lbph    1.175543e-05  3.331749e-04  5.422180e-06  4.182890e-03 -2.767067e-05
## svi     1.061792e-03  1.105061e-03  1.088674e-06 -2.767067e-05  2.436955e-02
## lcp     2.360950e-03 -5.526296e-06 -1.925461e-05 -8.158116e-06  1.152381e-03
## gleason 3.771483e-05  1.150541e-01  5.902495e-02  2.684350e-04  3.529049e-04
## pgg45   6.501233e-07  3.700386e-04  9.632069e-06  1.029936e-07  9.512890e-06
##                   lcp       gleason        pgg45
## lcavol   2.360950e-03  3.771483e-05 6.501233e-07
## lweight -5.526296e-06  1.150541e-01 3.700386e-04
## age     -1.925461e-05  5.902495e-02 9.632069e-06
## lbph    -8.158116e-06  2.684350e-04 1.029936e-07
## svi      1.152381e-03  3.529049e-04 9.512890e-06
## lcp      2.360292e-02 -1.923857e-04 1.255705e-04
## gleason -1.923857e-04  1.080488e+00 1.423169e-03
## pgg45    1.255705e-04  1.423169e-03 8.328631e-05
```

```r
cat(partition,"Variance :","\n")
```

```
## 
##  --------------------partition -------------------------- 
## 
##  Variance :
```

```r
diag(cov_m)
```

```
##       lcavol      lweight          age         lbph          svi          lcp 
## 3.433054e-02 8.421804e-01 1.239955e-02 4.182890e-03 2.436955e-02 2.360292e-02 
##      gleason        pgg45 
## 1.080488e+00 8.328631e-05
```

```r
sum(diag(cov_m))
```

```
## [1] 2.021637
```