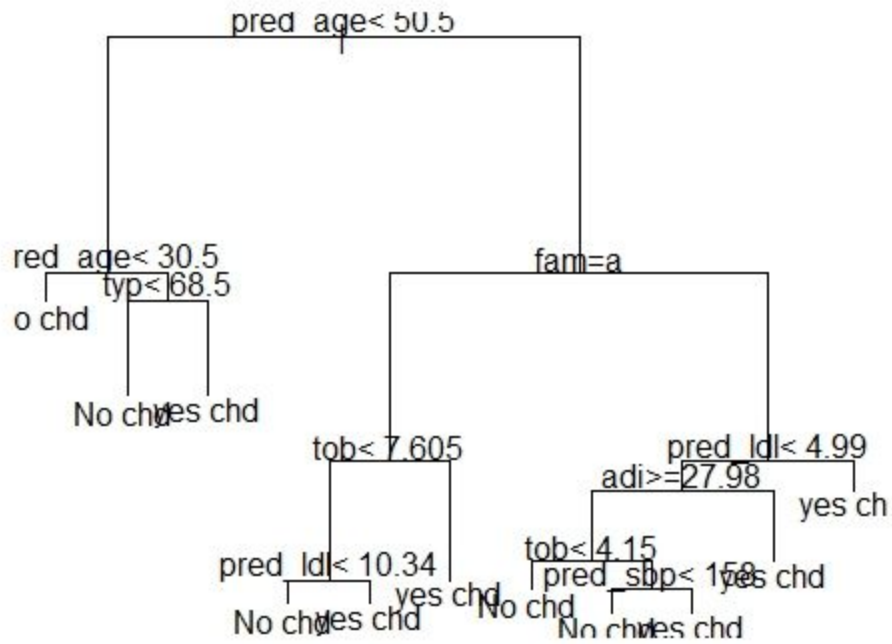
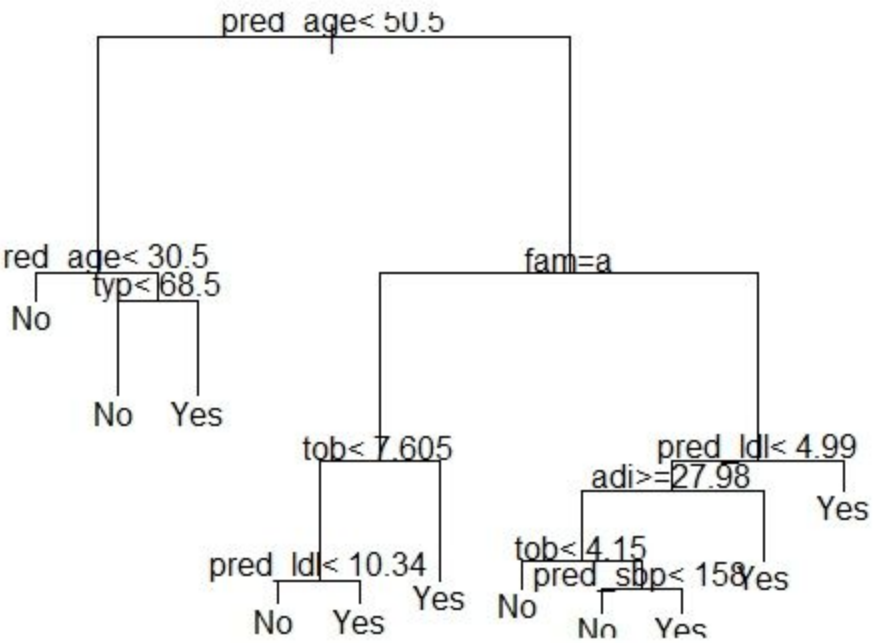
Alexandrea Stylianou
12/5/16
Homework 5
Commands and Answers

```
setwd("D:/Fall 2016/STAT 667")
heart_data <- read.csv("heartdisease.csv")
library(rpart)
pred_chd<-heart_data$chd
pred_sbp<-heart_data$sbp
tob<-heart_data$tobacco
pred_ldl<-heart_data$ldl
adi<-heart_data$adiposity
fam<-heart_data$famhist
typ<-heart_data$typea
obes<-heart_data$obesity
alc<-heart_data$alcohol
pred_age<-heart_data$age
n<-462
#Problem 1: [20 points]


#Construct the following classification trees where the original tree was grown using the Gini information criteria,
#continuing until the terminal nodes are homogeneous or until the node contains too few observations for further
#splitting (use 20 observations).

heartstat <- factor(heart_data$chd, levels = 0:1,labels=c("No chd","yes chd"))
heart_fit <- rpart(heartstat~pred_age+alc+obes+typ+fam+adi+pred_ldl+tob+pred_sbp,data = heart_data,method = 'class',control = rpart.control(minsplit = 5))
print(heart_fit)
plot(heart_fit)
text(heart_fit)
```



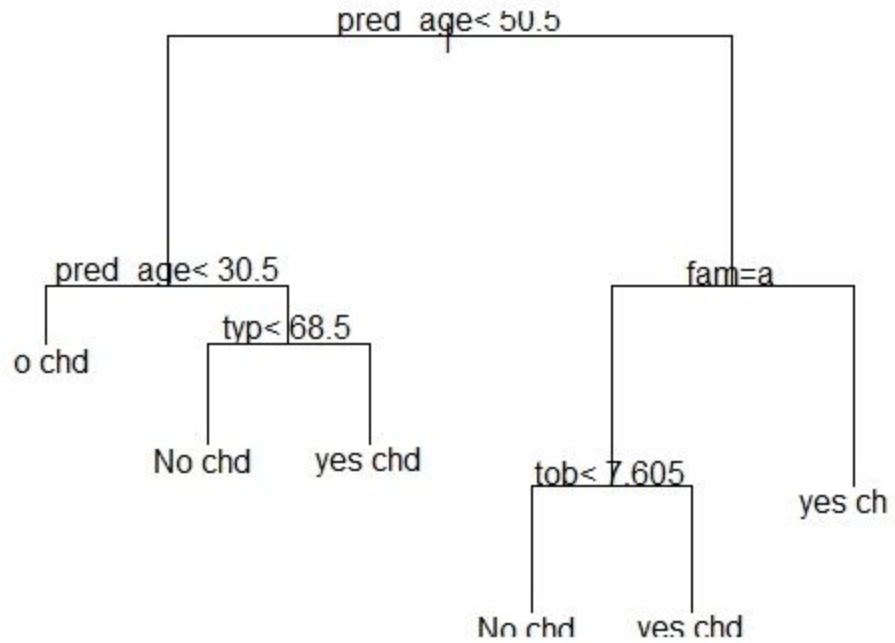

```
# node), split, n, loss, yval, (yprob)
# * denotes terminal node
#
# 1) root 462 160 0 (0.65367965 0.34632035)
# 2) pred_age< 50.5 290  64 0 (0.77931034 0.22068966)
# 4) pred_age< 30.5 108   8 0 (0.92592593 0.07407407) *
#   5) pred_age>=30.5 182  56 0 (0.69230769 0.30769231)
# 10) typ< 68.5 170  46 0 (0.72941176 0.27058824) *
#   11) typ>=68.5 12   2 1 (0.16666667 0.83333333) *
#   3) pred_age>=50.5 172  76 1 (0.44186047 0.55813953)
# 6) fam=Absent 82  33 0 (0.59756098 0.40243902)
# 12) tob< 7.605 58  16 0 (0.72413793 0.27586207)
# 24) pred_ldl< 10.34 56  14 0 (0.75000000 0.25000000) *
#   25) pred_ldl>=10.34 2   0 1 (0.00000000 1.00000000) *
#   13) tob>=7.605 24   7 1 (0.29166667 0.70833333) *
#   7) fam=Present 90  27 1 (0.30000000 0.70000000)
#   14) pred_ldl< 4.99 39  18 1 (0.46153846 0.53846154)
```

```
# 28) adi>=27.985 20   7 0 (0.65000000 0.35000000)
# 56) tob< 4.15 10   1 0 (0.90000000 0.10000000) *
#   57) tob>=4.15 10   4 1 (0.40000000 0.60000000)
# 114) pred_sbp< 158 6   2 0 (0.66666667 0.33333333) *
#   115) pred_sbp>=158 4   0 1 (0.00000000 1.00000000) *
#   29) adi< 27.985 19   5 1 (0.26315789 0.73684211) *
#   15) pred_ldl>=4.99 51   9 1 (0.17647059 0.82352941) *


#a) Report the final classification tree when the resubstitution error is used in determining the tree.
pred<-predict(heart_fit,type='class')
mc<-table(heart_data$chd,pred)
print(mc)
pred
err.resub<-1.0-(mc[1,1]+mc[2,2])/sum(mc)
print(err.resub)
#[1] 0.2034632
High=ifelse(pred_chd<=0.20,"No","Yes")
heart_data = data.frame(heart_data,High)
heart_fit_a<-rpart(High~pred_age+alc+obes+typ+fam+adi+pred_ldl+tob+pred_sbp)
```



node), split, n, loss, yval, (yprob)
* denotes terminal node
#
1) root 462 160 No (0.65367965 0.34632035)
2) pred_age< 50.5 290 64 No (0.77931034 0.22068966)
4) pred_age< 30.5 108 8 No (0.92592593 0.07407407) *
5) pred_age>=30.5 182 56 No (0.69230769 0.30769231)
10) typ< 68.5 170 46 No (0.72941176 0.27058824) *
11) typ>=68.5 12 2 Yes (0.16666667 0.83333333) *
3) pred_age>=50.5 172 76 Yes (0.44186047 0.55813953)
6) fam=Absent 82 33 No (0.59756098 0.40243902)
12) tob< 7.605 58 16 No (0.72413793 0.27586207) *
13) tob>=7.605 24 7 Yes (0.29166667 0.70833333) *
7) fam=Present 90 27 Yes (0.30000000 0.70000000)
14) pred_ldl< 4.99 39 18 Yes (0.46153846 0.53846154)
28) adi>=27.985 20 7 No (0.65000000 0.35000000)
56) tob< 4.15 10 1 No (0.90000000 0.10000000) *
57) tob>=4.15 10 4 Yes (0.40000000 0.60000000) *

```
#   29) adi< 27.985 19   5 Yes (0.26315789 0.73684211) *
#   15) pred_ldl>=4.99 51   9 Yes (0.17647059 0.82352941) *

#b) Report the final classification tree when the 1-SE rule is used in pruning the tree.
heart_fit_b <- rpart(heartstat~pred_age+alc+obes+typ+fam+adi+pred_ldl+tob+pred_sbp,data =
heart_data,method = 'class',control = rpart.control(minsplit = 20,cp=1e-05))
printcp(heart_fit_b)
#1 se rule: take the xerror and add it to the xstd. (1.07+0.064)=1.39. then compare that number
to the least amount of splits.
fit9<-prune(heart_fit_b,cp = 0.02)
print(fit9)
plot(fit9)
text(fit9)
```




```
# node), split, n, loss, yval, (yprob)
# * denotes terminal node
#
# 1) root 462 160 0 (0.65367965 0.34632035)
```

```
# 2) pred_age< 50.5 290  64 0 (0.77931034 0.22068966)
# 4) pred_age< 30.5 108   8 0 (0.92592593 0.07407407) *
#   5) pred_age>=30.5 182  56 0 (0.69230769 0.30769231)
# 10) typ< 68.5 170  46 0 (0.72941176 0.27058824) *
#   11) typ>=68.5 12   2 1 (0.16666667 0.83333333) *
#   3) pred_age>=50.5 172  76 1 (0.44186047 0.55813953)
# 6) fam=Absent 82  33 0 (0.59756098 0.40243902)
# 12) tob< 7.605 58  16 0 (0.72413793 0.27586207) *
#   13) tob>=7.605 24   7 1 (0.29166667 0.70833333) *
#   7) fam=Present 90  27 1 (0.30000000 0.70000000) *


#c) What is the class-specific misclassification rate for CHD/non-CHD patients for the
classification trees built
#in parts (a) and (b)?
#part A
heart_fit_a<-rpart(High~pred_age+alc+obes+typ+fam+adi+pred_ldl+tob+pred_sbp)
pred_a<-predict(heart_fit_a,type='class')
pred_a

mc_A<-table(heart_data$chd,pred_a)
#No Yes
#0 275  27
#1  71  89

#miss classification for zero  27/462= 0.058<----------class specific for chd
#miss classification for 1 71/462 = 0.1536<----------class specific misclassification for
prediction


heart_fit_b <- rpart(heartstat~pred_age+alc+obes+typ+fam+adi+pred_ldl+tob+pred_sbp,data =
heart_data,method = 'class',control = rpart.control(minsplit = 20,cp=1e-05))
pred_b<-predict(fit9,type = 'class')
pred_b
mc_B<-table(heart_data$chd,pred_b)
mc_B
pred_b
pred_b
# 0   1
# 0 266  36
# 1  70  90
#misclassification for zero 36/462 = 0.077----------class specific for chd
#misclassification for 1 70/462 = 0.151<----------class specific for prediction
```

```
#d) What are the class priors as estimated from the dataset?
#class priors
chd_prior=(sum(heart_data$chd)/length(heart_data$chd))
no_chd_prior = 1-chd_prior
#0.6536797<----------------------------------------------------------class prior

#e) Repeat the classification tree growing/pruning procedure as in part (b) but reversing the
priors as found
#in part (d) for the CHD/non-CHD classes rather than using the priors estimated from the data.
heart_fit_d<-rpart(pred_chd~pred_age+alc+alc+obes+typ+fam+adi+pred_ldl+tob+pred_sbp,met
hod = 'class',control = rpart.control(minsplit = 5),parms=list(prior=c(no_chd_prior, chd_prior)))
#
# node), split, n, deviance, yval
# * denotes terminal node
#
# 1) root 462 104.5887000 0.34632030
# 2) pred_age< 50.5 290  49.8758600 0.22068970
# 4) pred_age< 30.5 108   7.4074070 0.07407407
# 8) tob< 0.51 80   0.9875000 0.01250000 *
#   9) tob>=0.51 28   5.2500000 0.25000000
# 18) alc< 11.105 13   0.0000000 0.00000000 *
#   19) alc>=11.105 15   3.7333330 0.46666670 *
#   5) pred_age>=30.5 182  38.7692300 0.30769230
# 10) typ< 68.5 170  33.5529400 0.27058820
# 20) typ< 53.5 78  11.4871800 0.17948720
# 40) pred_ldl< 5.37 58   5.3793100 0.10344830 *
#   41) pred_ldl>=5.37 20   4.8000000 0.40000000
# 82) alc>=8.365 11   1.6363640 0.18181820 *
#   83) alc< 8.365 9   2.0000000 0.66666670 *
#   21) typ>=53.5 92  20.8695700 0.34782610
# 42) obes>=23.24 71  14.3662000 0.28169010
# 84) typ>=60.5 26   2.6538460 0.11538460 *
#   85) typ< 60.5 45  10.5777800 0.37777780
# 170) tob>=4.1 17   2.4705880 0.17647060 *
#   171) tob< 4.1 28   7.0000000 0.50000000 *
#   43) obes< 23.24 21   5.1428570 0.57142860 *
#   11) typ>=68.5 12   1.6666670 0.83333330 *
#   3) pred_age>=50.5 172  42.4186000 0.55813950
# 6) fam=Absent 82  19.7195100 0.40243900
# 12) tob< 7.605 58  11.5862100 0.27586210
# 24) typ< 42.5 11   0.0000000 0.00000000 *
#   25) typ>=42.5 47  10.5531900 0.34042550
```

```
# 50) adi< 24.435 13   0.9230769 0.07692308 *
#   51) adi>=24.435 34   8.3823530 0.44117650 *
#   13) tob>=7.605 24   4.9583330 0.70833330
# 26) adi>=28.955 15   3.7333330 0.53333330 *
#   27) adi< 28.955 9   0.0000000 1.00000000 *
#   7) fam=Present 90  18.9000000 0.70000000
#   14) pred_ldl< 4.99 39   9.6923080 0.53846150
# 28) adi>=27.985 20   4.5500000 0.35000000
# 56) tob< 4.15 10   0.9000000 0.10000000 *
#   57) tob>=4.15 10   2.4000000 0.60000000 *
#   29) adi< 27.985 19   3.6842110 0.73684210 *
#   15) pred_ldl>=4.99 51   7.4117650 0.82352940 *
#   >
```

```
#f) What is the class-specific misclassification rate for CHD/non-CHD patients for the
classification tree built
#in part (e)?
pred_d<-predict(heart_fit_d,type='class')
pred_d

mc_d<-table(heart_data$chd,pred_d)
pred_d
#0   1
#0 279  23      23/462 = 0.0497<----------misclassification error rate for chd
#1  71  89      71/462 = 0.153<----------misclassification erorr rate for prediction
```

```
########################################
Problem 2

# Using the hmwk5.csv dataset, derive a random forest consisting of 2500 classification trees in
the forest, where
# again the classifier predicts type using all other variables in the hmwk5.csv dataset. Use
set.seed(123) prior to
# running your RF.

setwd("D:/Fall 2016/STAT 667")
library(randomForest)
hmwk5_csv<-read.csv("hmwk5.csv")
type1<-hmwk5_csv$type
```

```
length(type1)
set.seed(123)
tr<-sample(1:79,100,replace=TRUE)
ir.tf<-randomForest(type ~ .,data=hmwk5_csv[tr,],ntree=2500,mtry=2,nodesize=1,importance=T)
plot(ir.tf)
```

a) What is the out-of-bag estimate of error for the random forest?

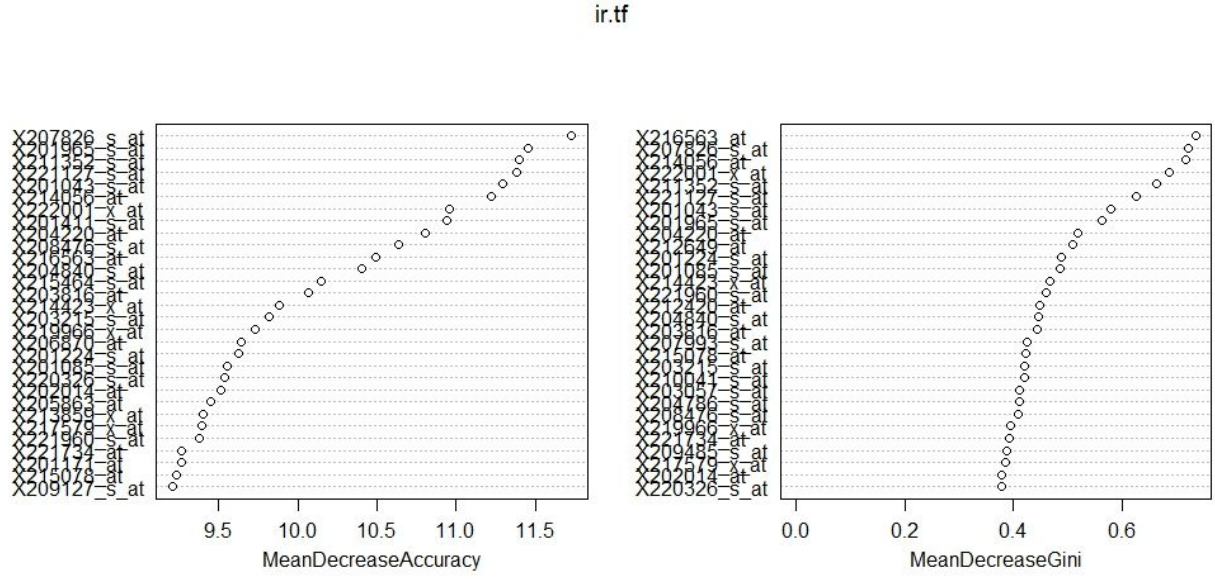
#OOB estimate of error rate: 10%

b) Looking at the confusion matrix, which class is most frequently misclassified?

#early

c) Plot the mean decrease in accuracy and the mean decrease in Gini index variable importance measures.

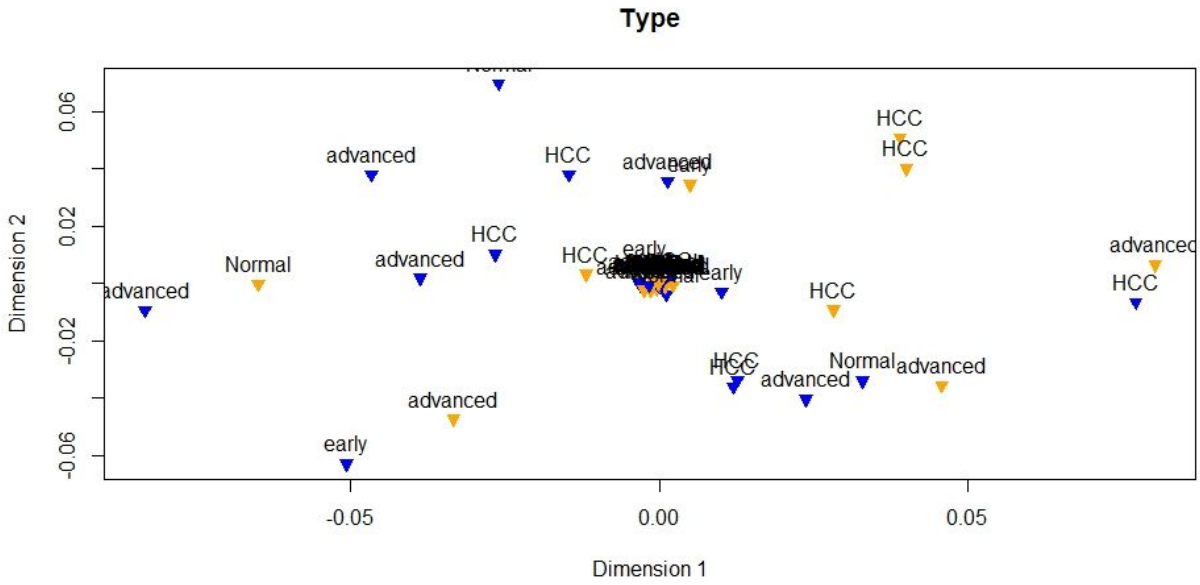
```
varImpPlot(ir.tf)
```



d) Suppose you set a rule such that the jth variable is considered important, if it's mean Decrease in Accuracy

(Dj) is greater than the overall average Mean Decrease in Accuracy (^mD) + 1.96 times the SD (^sD). How

```
# many variables are important, and what are they?

impor <-ir.tf$importance
col <- summary(impor)
#Mean   :0.0028262
sd(new_data_frame)
#0.001783876

a.m.d.a<-0.0028262 + (1.96 * 0.001783876)
impor_data_frame<-data.frame(impor)
new_data_frame<-impor_data_frame$MeanDecreaseAccuracy
impor_data_frame[impor_data_frame$MeanDecreaseAccuracy>0.006322597,]

#8 variables are important

# X203816_at
# X207826_s_at
# X213859_x_at
# X214056_at
# X219966_x_at
# X220326_s_at
# X221127_s_at
# X222001_x_at

# e) Perform multi-dimensional scaling in 2 dimensions using the proximities and produce a scatterplot with a
# label indicating the sample class (type). Which classes are well separated and which are not? Do you
# suspect any outliers? [MDS can be implemented using the *cmdscale* function].

#advanced appears to be well separated from HCC and normal
#outliers do appear. Upon reviewing the graph outliers appear in HCC, and advanced

ir.tf2<-randomForest(type ~ .,data=hmwk5_csv[tr,],ntree=2500,mtry=2,nodesize=1,importance=T,proximity=TRUE)$proximity
cmd_matrix<-cmdscale((ir.tf2),k=2)
print(cmd_matrix)

plot(cmd_matrix,col=c("blue","orange"),bg=c("blue","orange"),pch=25,xlab="Dimension 1",ylab='Dimension 2',main = "Type")
text(cmd_matrix,labels=type1,cex=1.0,pos=3)
```



##
Problem 3

Eighty-eight students took 5 separate tests in the following quantitative subjects: mechanics, vectors, algebra,
analysis, and statistics. The first two tests were closed book, while the last three tests were open book. Download
the test results (see the R workspace scor.R in blackboard) to an appropriate directory. Use the command >
source("scor.R") to get the data matrix into R.

a.) report the eigenvalues and eigenvectors.

```
test_scores<-source('scor.R')
ts_matrix<-as.matrix(test_scores$value)
cov_matrix_test<-cov(ts_matrix, y=NULL)

meh<-eigen(cov_matrix_test,symmetric=TRUE)
meh$values
#$values
#[1] 686.98981 202.11107 103.74731  84.63044  32.15329

#$vectors
#[,1]       [,2]       [,3]        [,4]       [,5]
```

```
#[1,] -0.5054457  0.74874751 -0.2997888  0.296184264 -0.07939388
#[2,] -0.3683486  0.20740314  0.4155900 -0.782888173 -0.18887639
#[3,] -0.3456612 -0.07590813  0.1453182 -0.003236339  0.92392015
#[4,] -0.4511226 -0.30088849  0.5966265  0.518139724 -0.28552169
#[5,] -0.5346501 -0.54778205 -0.6002758 -0.175732020 -0.15123239


# b.) Calculate theta for score data

686.98981/(686.98981+202.11107+103.74731+84.63044+32.15329)

#0.619115<----------------------------------theta for score data

# c.) Use the bootstrap method to estimate the standard error of Theta.
library(bootstrap)

theta <- function(jay) {
  new_values <- eigen(var(scor[jay,]), symmetric=TRUE, only.values=TRUE)$values
  new_values[1] / sum(new_values) }
bootstrap_vales <- bootstrap(1:88, 200, theta)
sd(bootstrap_vales$thetastar)
#[1]  0.0453244<-------------------------------------------------Standard Error of Theta
```