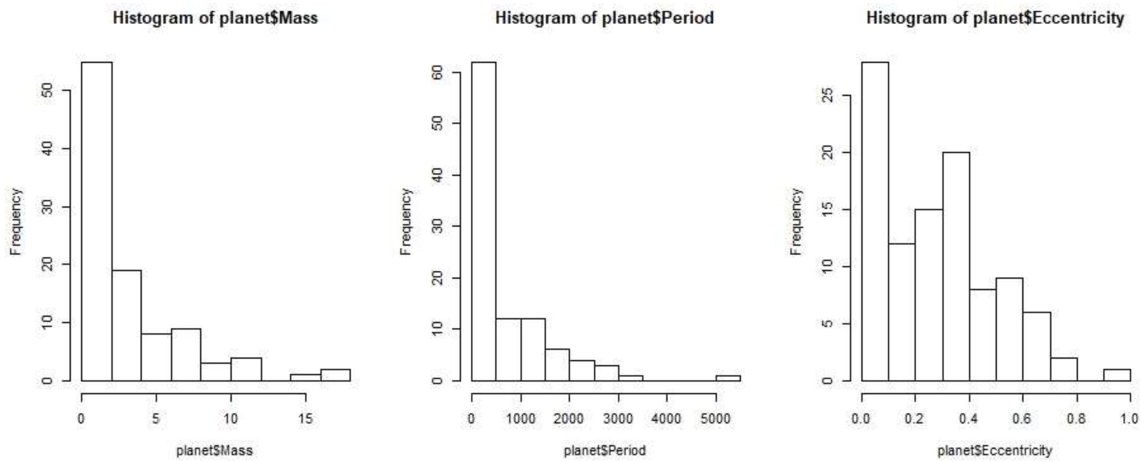
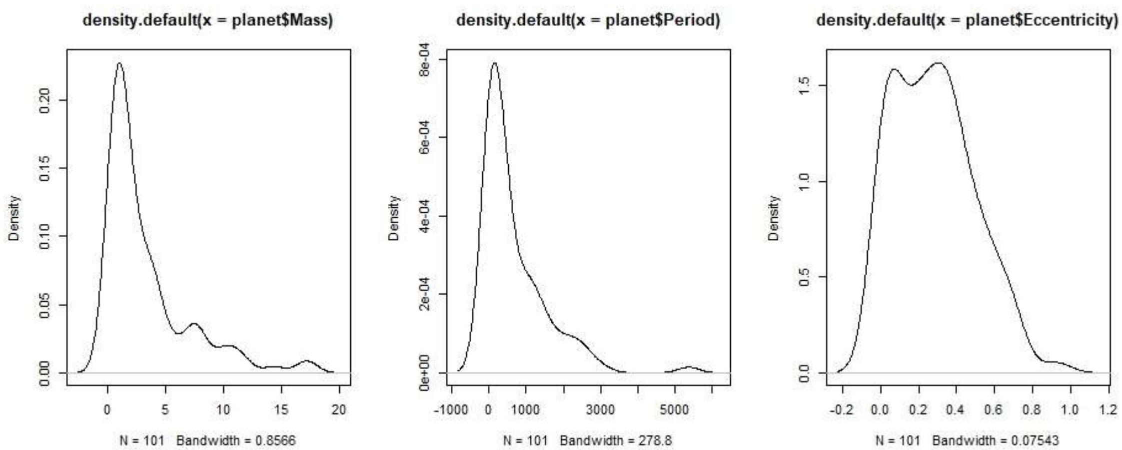
Mini Project #2

Name ： Xi Cui

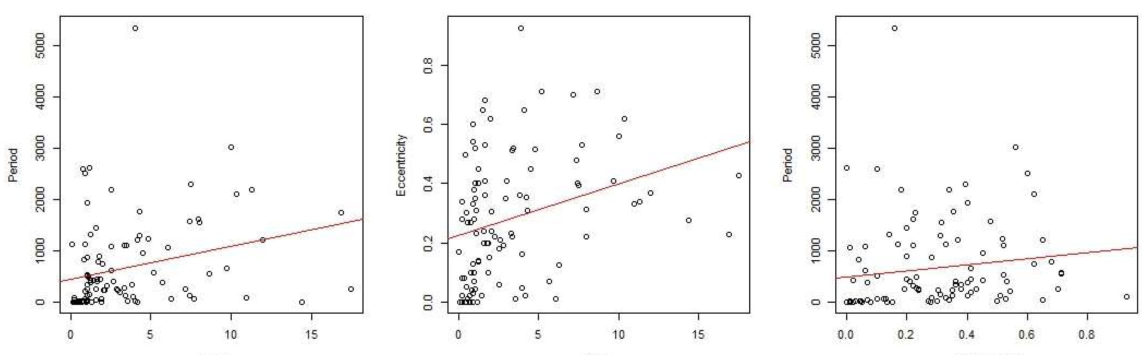
Q1

a) There are 101 observation of three variables: Mass, Period, Eccentricity. All three variables are numeric.




Bivariate relationships



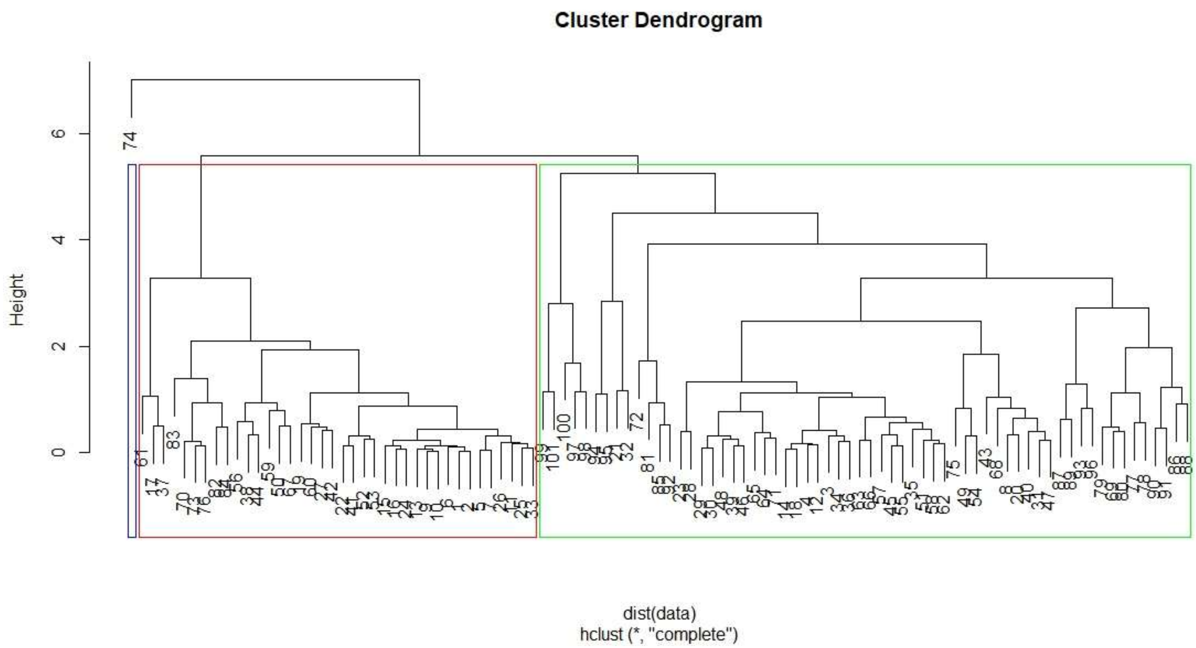

```
> cor(planet)
                  Mass    Period Eccentricity
Mass         1.0000000 0.2684085    0.3049333
Period       0.2684085 1.0000000    0.1445935
Eccentricity 0.3049333 0.1445935    1.0000000
```

All three variables are nor normal distributed. The correlation between every two variables are not very strong. Mass and Eccentricity have heightes correlation.

b) The standardizing is a good idea. As the scales between variables are significantly different, it is inconvenient to compute the distance between observations. By standardizing, some problems cased by large scale can be solved.

c) I will use metric-based distance to cluster the exoplanets.

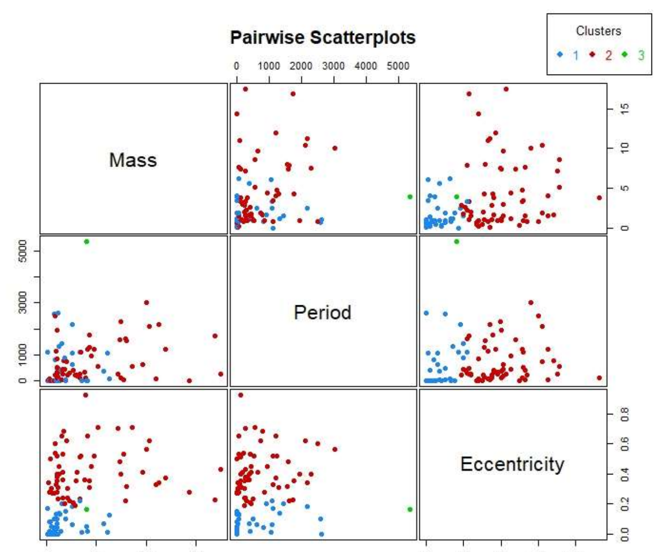
d) Hierarchically Cluster:

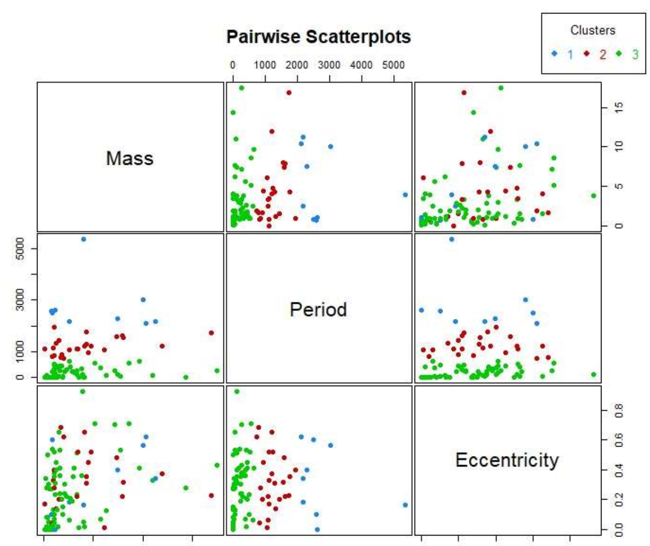



Summary of the cluster-specific means of the three variables:

```
> aggregate(planet, by=list(cluster=groups),mean)
  cluster     Mass    Period Eccentricity
1       1 1.703316  488.7479   0.07077105
2       2 4.311774  699.7941   0.41269355
3       3 4.000000 5360.0000   0.16000000
```

Pair-wise scatterplots:




e) K-Means Clustering (K=3)

Summary of the cluster-specific means of the three variables:

```
Cluster means:
      Mass    Period Eccentricity
1 5.390000 2767.2444    0.3283333
2 4.233333 1235.9729    0.3232917
3 2.734500  187.5163    0.2606221
```

Pair-wise scatterplots: up right

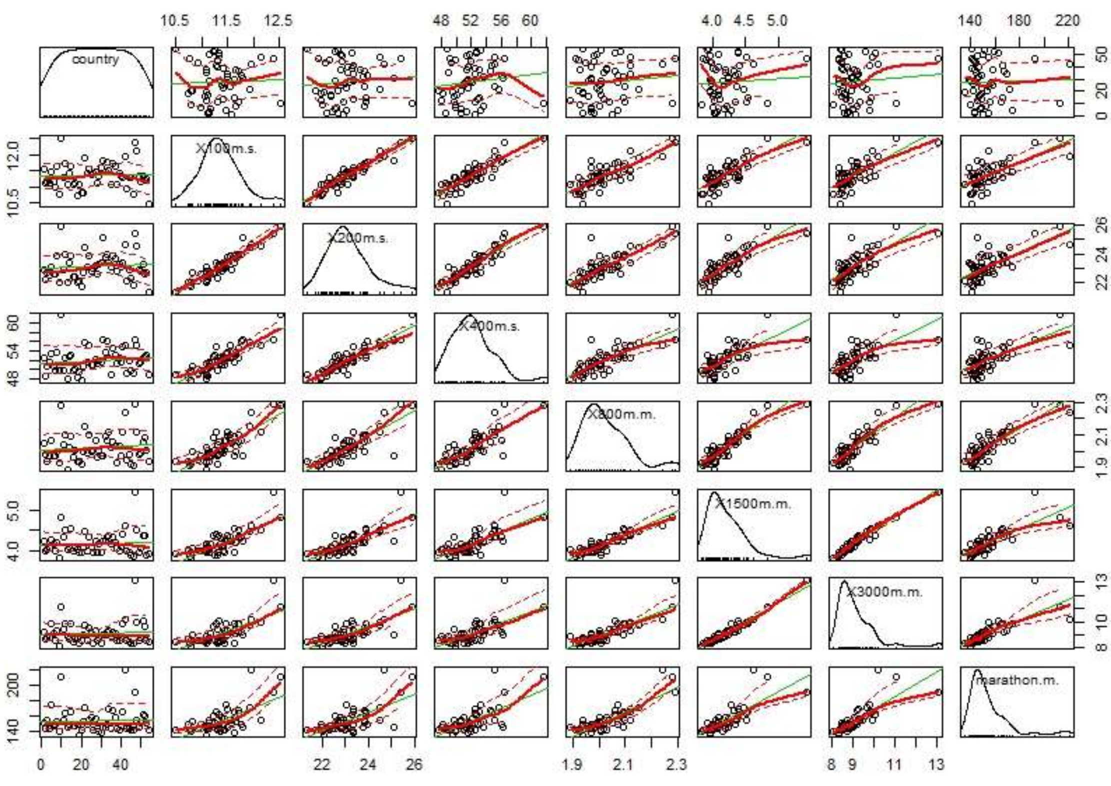
Comparation:

| | Km.out | | |
|---|---|---|---|
| groups | 1 | 2 | 3 |
| 1 | 3 | 8 | 27 |
| 2 | 5 | 16 | 41 |
| 3 | 1 | 0 | 0 |

Conclusion: The result of two clustering method is different, the data distributed more obviously in three clusters in K-Means.

Q2

a) There are 54 observations of 8 variables: country, X100m.s., X200m.s., X400m.s., X800m.m., X1500m.m., X3000m.m., marathon.m.. County is Factor with 54 levels, the rest 7 variables are numeric.




All 8 variables are nor normal distributed, 7 variables have long right tail. There is strong relation between variables except "country".

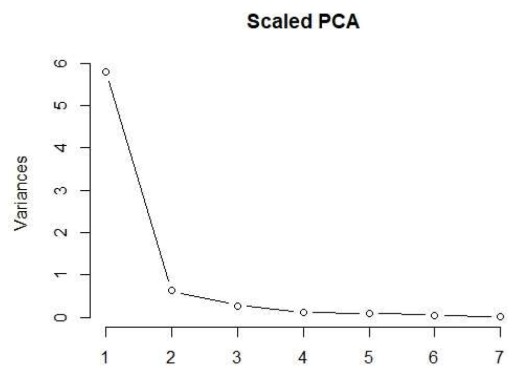
b) Standardizing the variable before performing the analysis will be a good idea. The wild range of variance will influence the result of principle components, we should standardize the variables.

c) R output:

```
> summary(track.pca)
```

```
Importance of components:
                          PC1     PC2    PC3     PC4    PC5     PC6     PC7
Standard deviation     2.4099 0.79290 0.5285 0.35292 0.3016 0.23349 0.11959
Proportion of Variance 0.8297 0.08981 0.0399 0.01779 0.0130 0.00779 0.00204
Cumulative Proportion  0.8297 0.91947 0.9594 0.97717 0.9902 0.99796 1.00000
```

The first 2 principal components can explain 91.94% variance. The scatterplot of scaled PCA shows the most o bvious change in slope occurs at component 2, we can argue that the first 2 components should be retained

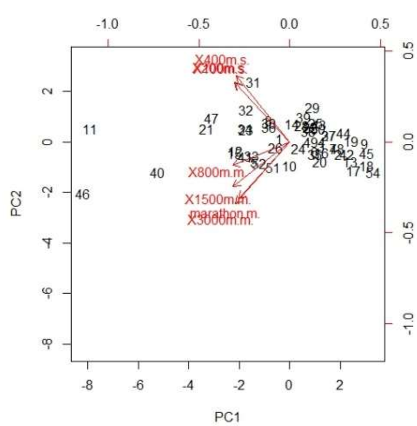



d)

```
                   PC1        PC2
X100m.s.    -0.3777657  0.4071756
X200m.s.    -0.3832103  0.4136291
X400m.s.    -0.3680361  0.4593531
X800m.m.    -0.3947810 -0.1612459
X1500m.m.   -0.3892610 -0.3090877
X3000m.m.   -0.3760945 -0.4231899
marathon.m. -0.3552031 -0.3892153
```

Cumulative proportion explained by the two components is 91.94%.

Biplot:




The first Principle component is interpreted as a measure of athletic excellence of a given nation and the second as the relative strength of a nation at various running distances.

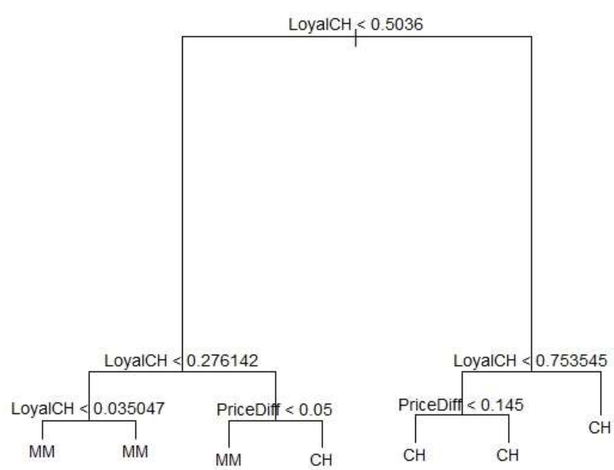
e) Ranking of scores on First PC:

```
 track$country track.pca.x...1.
1           USA      3.299148823
2           GER      3.047516603
3           RUS      3.042948214
4           CHN      2.989466907
5           FRA      2.518345696
6           GBR      2.442706280
7           CZE      2.406030321
8           POL      2.273765780
9           ROM      2.123005711
10          AUS      1.931642887
```

The above show top 10 nations based on their score on the first principal component. It tells the USA is the top on country and GER is the second, which correspond with my intuitive notion of athletic excellence for the various countries.

Q3

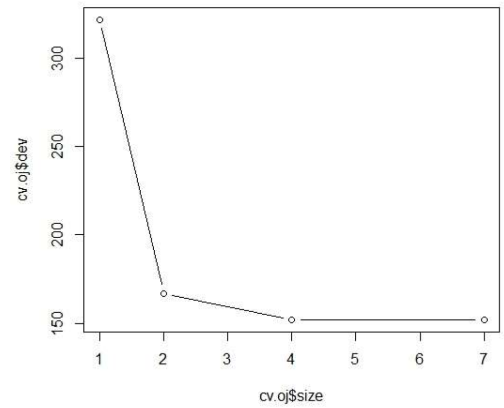
a) Data preparing:
b) The result of fitting tree model shows that dataset is separated into 7 regions as there are 7 leaves of tree structure, the plot shows the regions $R_1$to $R_7$ from left to right.

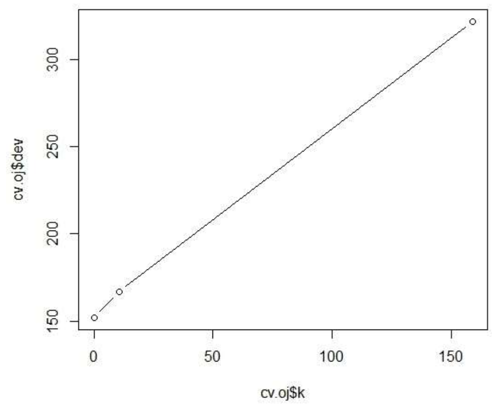



$R_1 = \{X|LoyalCH < 0.035047\}; R_2 = \{X|0.2761 > LoyalCH \geq 0.035047\};$
$R_3 = \{X|0.5036 > LoyalCH \geq 0.2761, PriceDiff < 0.05\};$
$R_4 = \{X|0.5036 > LoyalCH \geq 0.2761, PriceDiff \geq 0.05\}$
$R_5 = \{X|0.5036 < LoyalCH < 0.7535, PriceDiff < 0.145\};$
$R_6 = \{X|0.5036 < LoyalCH < 0.7535, PriceDiff \geq 0.145\}$
$R_7 = \{X|LoyalCH \geq 0.753545\}$

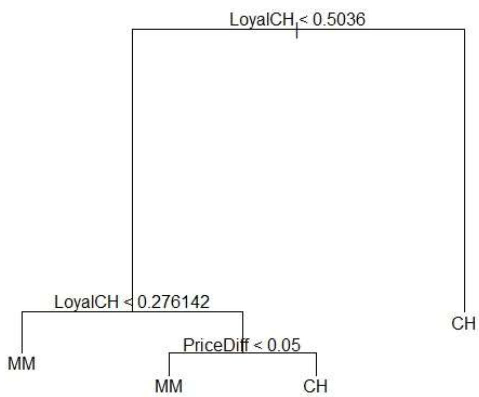
Where $R_1, R_2, R_3$ belongs to MM, and $R_4, R_5, R_6, R_7$ belongs to CH

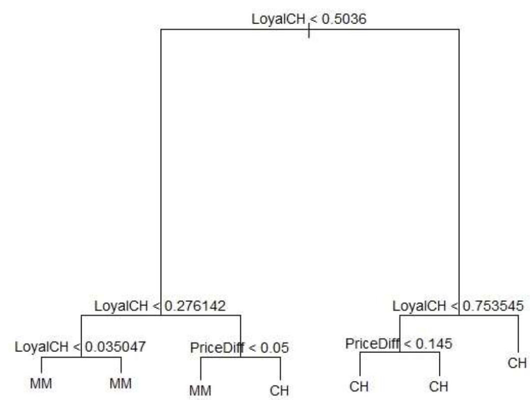
c) Test MSE before pruning: MSE=0.245
By using cross validation, the plot show we can pruning tree to size 7,4,2,1, when size = 7, we get the minimum error rate, but we also can prune tree to size 4, as size 4 tree also include enough information and with low MSE.




After pruning tree to size = 4:




As the plots show, data set will be classified into 4 regions, which less than tree before pruning.

$R_1 = \{X|LoyalCH < 0.2761\}; R_2 = \{X|0.5036 > LoyalCH \geq 0.2761, PriceDiff < 0.05\};$

$R_3 = \{X|0.5036 > LoyalCH \geq 0.2761, PriceDiff \geq 0.05\};$

$R_4 = \{X|LoyalCH \geq 0.5036\}$

Where $R_1, R_2$,belongs to MM, and $R_3$ , $R_4$ belongs to CH.

Test MSE after pruning: MSE=0.245, which is same as tree before pruning. After pruning, the structure of tree is more efficient, less regions but do not increase error rate.

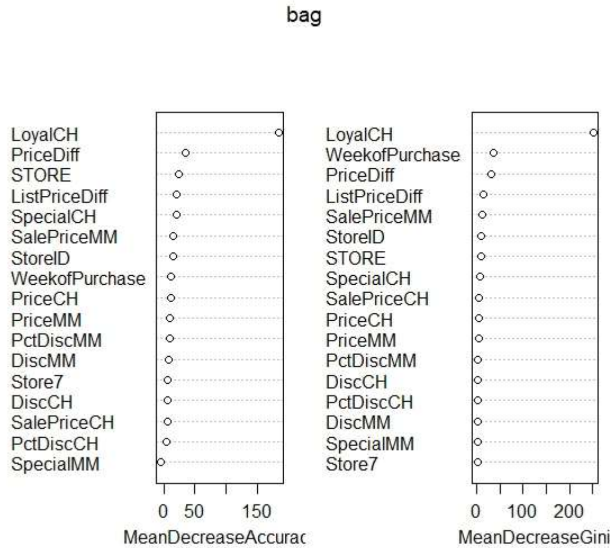
Most important predictor: LoyalCH. Both LoyalCH and PriceDiff are important, but LoyalCH is more efficient and important to divide the data to class.

d) Bagging Approach:

Output of R:
```
> bag
Call:
 randomForest(formula = Purchase ~ ., data = train, mtry = 17,      ntree = 1000, importance = TRUE)
               Type of random forest: classification
```

```
                    Number of trees: 1000
No. of variables tried at each split: 17

       OOB estimate of  error rate: 20.23%
Confusion matrix:
   CH  MM class.error
CH 464  84   0.1532847
MM  92 230   0.2857143
```




Test confusion matrix:

| bag.pred | CH | MM |
|---|---|---|
| CH | 86 | 17 |
| MM | 19 | 78 |

Test MSE: (19+17)/200 = 0.18

Most important predictor: LoyalCH. Righ plot shows.

e) Random Forest:

Test confusion matrix:

| rf.pred | CH | MM |
|---|---|---|
| CH | 89 | 25 |
| MM | 16 | 70 |

Test MSE: (26+15)/200 = 0.205

Most important predictor: LoyalCH.

f) Boosting:

Test confusion matrix:

| boost.pre | 0 | 1 |
|---|---|---|
| 0 | 66 | 17 |
| 1 | 29 | 88 |

Test MSE: (29+17)/200 = 0.23

Most important predictor: LoyalCH.

R OUTPUT: `> summary(boost)`

```
                          var    rel.inf
LoyalCH               LoyalCH 79.7342317
PriceDiff           PriceDiff  9.5024147
StoreID               StoreID  2.2820095
SalePriceMM       SalePriceMM  1.7308702
ListPriceDiff   ListPriceDiff  1.1268622
WeekofPurchase WeekofPurchase  1.0610927
STORE                   STORE  0.8311845
DiscMM                 DiscMM  0.6581720
DiscCH                 DiscCH  0.5849339
SpecialMM           SpecialMM  0.5686670
PctDiscMM           PctDiscMM  0.3789681
PctDiscCH           PctDiscCH  0.3787959
SalePriceCH       SalePriceCH  0.3751084
PriceMM               PriceMM  0.3204249
```

```
SpecialCH         SpecialCH  0.1989276
Store7              Store7  0.1560598
PriceCH            PriceCH  0.1112771
```

g) KNN:

Optimal K = 7;

Test confusion matrix:

```
         test.Y
knn.fit CH MM
     CH 93 42
     MM 12 53
```

Test MSE: (42+12)/200 = 0.27

Most important predictor: LoyalCH.

h) Comparation:

| Model | Test Error rate |
|---|---|
| Decision Tree (size=7) | 0.245 |
| Bagging (B=1000, mtry=17) | 0.18 |
| Rando Forest | 0.205 |
| Boosting | 0.23 |
| KNN(k=7) | 0.27 |

By comparing all the five method, the bagging approach has the lowest Test error rate, random forest also has good performance, so I will recommend bagging. As bagging try more complex structure for each tree and get the average class for 1000 trees, the model is more accurate than other models.

R-code:

```
#=============Q1=================#
planet<-read.table("C:/Users/xicui/Desktop/stat6340/planet.csv", header = TRUE, sep=",")
#---------a exploratory---------
attach(planet)
str(planet)
summary(planet)
plot(planet,col = c("green","blue"))
par(mfrow = c(1,2))
#histofram
par(mfrow = c(1,3))
hist(planet$Mass)
hist(planet$Period)
hist(planet$Eccentricity)
par(mfrow = c(1,3))
plot(density(planet$Mass))
plot(density(planet$Period))
plot(density(planet$Eccentricity))
#bivariate comparing
par(mfrow =c(1,3))
plot(Mass,Period)
abline(reg = lm(Period ~ Mass), col = "red")
plot(Mass,Eccentricity)
abline(reg = lm(Eccentricity ~ Mass), col = "red")
plot(Eccentricity, Period)
abline(reg = lm(Period ~ Eccentricity), col = "red")
#correlation between two variables
cor(planet)
#-------------d------------
install.packages("mclust")
#standardizing the dataset
data<-scale(planet)
#hierarchically cluster
hc.complete=hclust(dist(data),method="complete")
par(mfrow =c(1,1))
plot(hc.complete)
rect.hclust(hc.complete,k=3,border = c("blue","red","green"))
groups <-cutree(hc.complete, 3)
#pairwise scatterplots
library(mclust)
hc<-clPairs(planet,groups,symbol=16, main="Pairwise Scatterplots")
clPairsLegend('topright', class = hc$class, col = hc$col, pch = hc$pch,cex=0.8,horiz=TRUE, title = "Clusters")
#summarize cluster-specific means of three variables
aggregate(planet, by=list(cluster=groups),mean)
```

```
#-----------e-K means with K=3 cluster----------
km.out <- kmeans(planet, 3, nstart = 1)
km.out
km.out$cluster
#summarize cluster-specific means of three variables
aggregate(planet, by=list(km.out$cluster),mean)
#pairwise scatterplots
km<-clPairs(planet,km.out$cluster,symbol=16, main="Pairwise Scatterplots")
clPairsLegend('topright', class = km$class, col = km$col, pch = km$pch,
              cex=0.8,horiz=TRUE, title = "Clusters")

table(groups,km.out$cluster)
#===============Q2=======================#
track<-read.table("C:/Users/xicui/Desktop/stat6340/track-records-women.csv", header = TRUE, sep=",")
#---------a exploratory------------------------------------
head(track)
str(track)
library(car)
scatterplotMatrix(track)
cor(track[,2:8])
#----------c standardzing and PCA---------------------------
track.standar <- as.data.frame(scale(track[2:8]))
track.pca<-prcomp(track.standar)
summary(track.pca)
screeplot(track.pca, type="lines", main="Scaled PCA")
#------------d--------------------
plot(track.pca$x[,1],track.pca$x[,2])
text(track.pca$x[,1],track.pca$x[,2], track$country, cex=0.7, pos=4, col="red")
#table of components
track.pca$rotation[,1]
track.pca$rotation[,2]
track.pca$rotation[,1:2]
biplot(track.pca, scale=0)
#------------------e Rank the nations based on their score-----------
library(plyr)
score=data.frame(track.pca$x[,1])
rank=cbind(track$country,score)
arrange(rank,desc(score))
#===============Q3=======================#
library(ISLR)
#--------a-------------
#data preparing
attach(OJ)
data<-OJ
```

```
train<-data[1:870,]
test<-data[871:1070,]
#--------b----------
#fit the tree model
library(tree)
tree<- tree(Purchase ~ ., train)
tree
summary(tree)
#plot the tree
plot(tree)
text(tree, pretty = 0, cex = 0.9)
#---------------c    pruning ---------------
#predict class for tesing data
tree.pred <- predict(tree, test, type = "class")
#Compute the confusion matrix
table(tree.pred, test$Purchase)
#Compute the test misclassification rate
(11+38)/200
#Perform cost complexity pruning by CV, guided by misclassification rate
set.seed(3)
cv.oj <- cv.tree(tree, FUN = prune.misclass)
cv.oj

#find best tree size
par(mfrow = c(1, 1))
plot(cv.oj$size, cv.oj$dev, type = "b")
plot(cv.oj$k, cv.oj$dev, type = "b")
cv.oj$size[which.min(cv.oj$dev)]#best size is 7 no need to prune

#try prune of size = 4
prune.oj <- prune.misclass(tree, best = 4)
plot(prune.oj)
text(prune.oj, pretty = 0)
#Compute the test misclassification rate
tree.predict <- predict(prune.oj, test, type = "class")
table(tree.predict, test$Purchase)
#----------d Bagging approach--------
library(randomForest)
set.seed(1)
bag <- randomForest(Purchase ~ ., train, mtry=17, ntree = 1000, importance = TRUE)
bag

#estimate the test error rate
bag.pred<-predict(bag,test,type="class")
```

```
table(bag.pred, test$Purchase)
(19+17)/200
#Get variable importance measure for each predictor
?importance
importance(bag)
varImpPlot(bag)
#-----------e Randomforest approach-------------
set.seed(1)
rf <- randomForest(Purchase ~ ., train, mtry=sqrt(17),ntree = 1000, importance = TRUE)
rf
#estimate the test error rate
rf.pred<-predict(rf,test,type="class")
table(rf.pred, test$Purchase)
(16+25)/200
#Get variable importance measure for each predictor
importance(rf)
varImpPlot(rf)

#--------------f Boosting approach-------------
library(gbm)
train$Purchase<-ifelse(train$Purchase=="CH","1","0")
test$Purchase<-ifelse(test$Purchase=="CH","1","0")

set.seed(1)
#boosting in classifying
boost<- gbm(Purchase ~ ., train, distribution = "bernoulli", n.trees = 1000, interaction.depth = 1,shrinkage = 0.01)
summary(boost)
#estimate the test error rate
predict(boost,test,n.trees = 1000,type = "response")
boost.prob = predict(boost,test, n.trees = 1000, type = "response")
#depending on probability define the class
boost.pre<-ifelse(boost.prob>0.5,"1","0")
#confution matrix
t<-table(boost.pre,test$Purchase)
t
#test MSE
(t[1,2]+t[2,1])/200

#----------g knn approach--------------
library(class)
data$Store7<-ifelse(data$Store7=="Yes",1,0)
train<-data[1:870,]
test<-data[871:1070,]
#prepare training and testing data to predictors and class label
```

```
train.X<-train[,c(2:18)]
train.Y<-train$Purchase
test.X<-test[,c(2:18)]
test.Y<-test$Purchase
#fit knn and find the optimal K
ks <- c(seq(1, 30, by = 1), seq(35, 100, by = 5))
nks <- length(ks)
err.rate.train <- numeric(length = nks)
err.rate.test <- numeric(length = nks)
names(err.rate.train) <- names(err.rate.test) <- ks

for (i in seq(along = ks)) {
  set.seed(1)
  mod.train <- knn(train.X, train.X, train.Y, k = ks[i])
  set.seed(1)
  mod.test <- knn(train.X, test.X, train.Y, k = ks[i])
  err.rate.train[i] <- 1 - sum(mod.train == train.Y)/length(train.Y)
  err.rate.test[i] <- 1 - sum(mod.test == test.Y)/length(test.Y)
}
result <- data.frame(ks, err.rate.train, err.rate.test)
result[err.rate.test == min(result$err.rate.test), ]
#optimal K=7
set.seed(1)
knn.fit <- knn(train.X, test.X, train.Y, k =7 , prob = F)
#test MSE
t<-table(knn.fit,test.Y)
t
(t[1,2]+t[2,1])/200
```