1. #[P(Class is RED | X) >= 0.5] = 6, hence using **majority vote** approach the specific value of X should be classified as RED.

   Average[P(Class is RED | X)] = 0.45, hence using **average probability** approach the specific value of X should be classified as GREEN.

2.

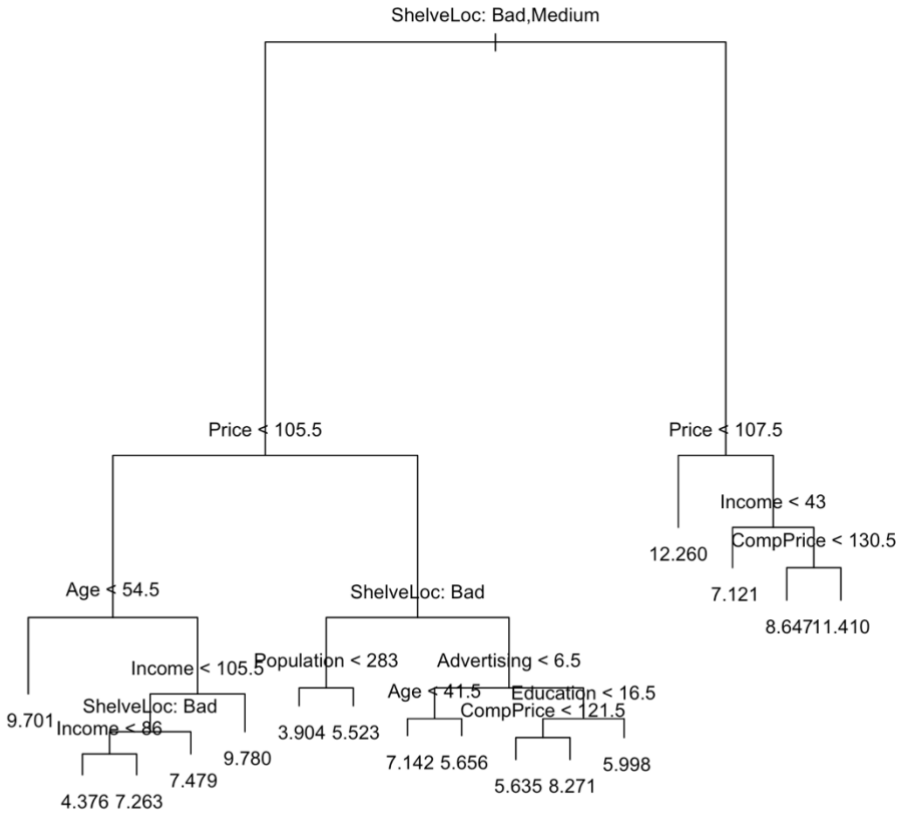
   a.

```
library(ISLR)
cardata <- Carseats
View(cardata)

#2a -> Create train and test data sets
set.seed(123)
train_ind <- sample(seq_len(nrow(cardata)), size = 300)

train <- cardata[train_ind, ]
test <- cardata[-train_ind, ]
```

   b.

```
#2b -> fit a regression tree

install.packages("tree")
library(tree)

reg.tree <- tree(Sales~.,data=train)
summary(reg.tree)

plot(reg.tree)
text(reg.tree,pretty=0)

#Test prediction accuracy of regression tree
tree.pred <- predict(reg.tree,test)
test.mse <- mean((test$Sales-tree.pred)^2)
test.mse
```

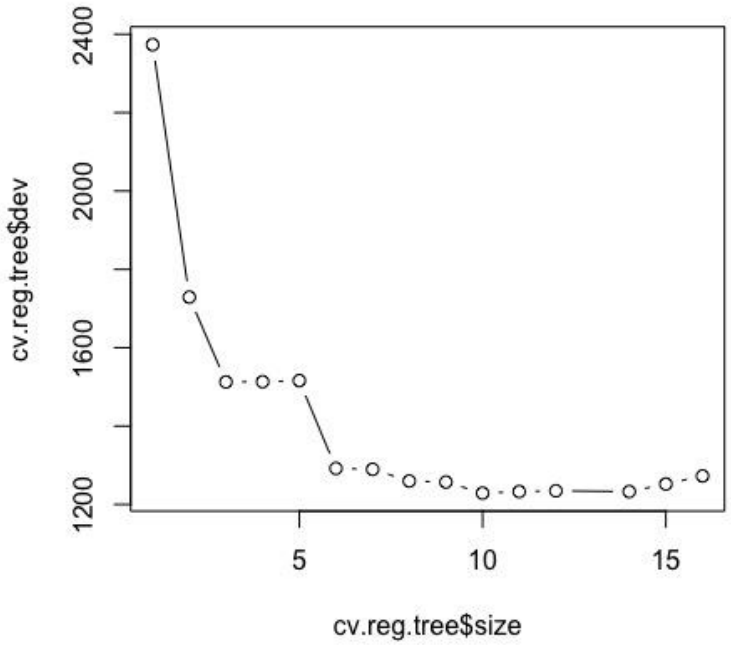
Below: Regression tree for Sales.



i. 8 of the variables were used in constructing the tree: ShelveLoc, Price, Age, Income, Population, Advertising, Education, and CompPrice.
ii. The sum of squared errors of the regression tree is 2.44.
iii. The plot of the tree indicates that whenever the shelving location of the car seat is good and the price of the car seat is less than 107.5 then the location sells more car seats than if the price was above 107.5. Similarly, if the car seat is in a bad or medium position and the price of the car seat is below 105.5 then the location sells more car seats than if the car seat were priced over 105.5.
iv. Test set MSE = 6.04481

c.

```
#2c -> find optimal tree size using CV
cv.reg.tree <- cv.tree(reg.tree, ,FUN=prune.tree)
plot(cv.reg.tree$size,cv.reg.tree$dev,type='b')

#prune tree
prune.reg.tree <- prune.tree(reg.tree,best = 10)
plot(prune.reg.tree)
text(prune.reg.tree,pretty=0)

pred.prune <- predict(prune.reg.tree,test)
prune.mse <- mean((pred.prune-test$Sales)^2)
prune.mse
```



The plot of the sum of the squared errors of the cross-validation tree against the size of the tree (above) indicates that the optimal tree size is 10 nodes.

Test set MSE = 6.151869 so pruning the tree did not improve the test error rate.

```
              %IncMSE IncNodePurity
CompPrice   26.5696262    212.096205
Income      15.9484623    155.727980
Advertising 22.8502244    183.821821
Population  -0.4408909     80.938298
Price       72.5240092    664.851755
ShelveLoc   72.8271745    592.899492
Age         20.6253948    209.166037
Education    3.5324388     69.705602
Urban       -0.6140856      8.533007
US          -0.5066339      7.892298
```

From the importance table above the two most important predictors of sales are price and shelve location.

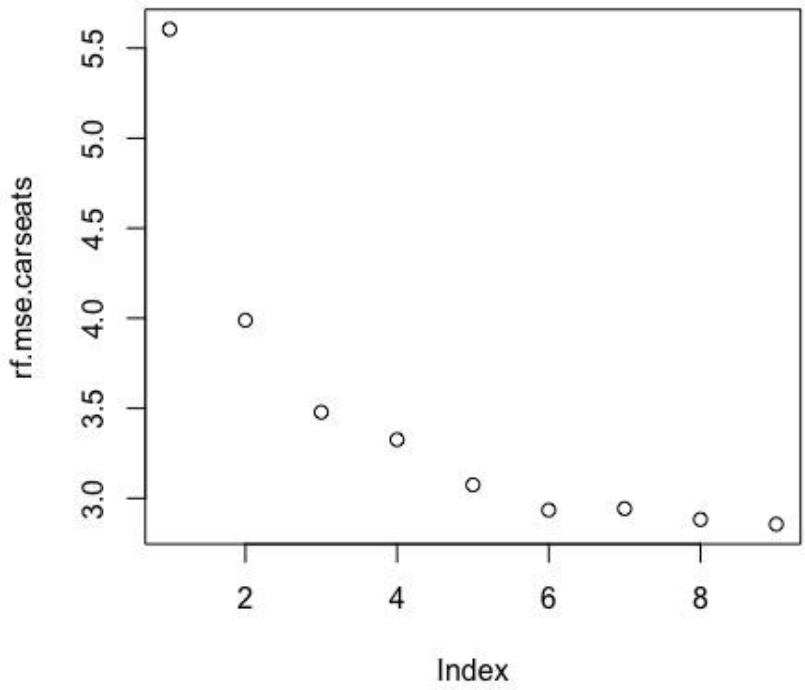
d.

```
#2d -> perform bagging (rf with m=p)
install.packages("randomForest")
library(randomForest)
set.seed(1)
bag.carseats <- randomForest(Sales~.,data=train,mtry=10,importance=T)
bag.carseats

#Test prediction accuracy of bagging
bag.pred <- predict(bag.carseats,test)
mse.bag <- mean((bag.pred-test$Sales)^2)
mse.bag
```

Using bagging the test set MSE = 2.882455.

e.

```
#2e -> random forest with different m
set.seed(1)
rf.mse.carseats <- c(1:9)
for (i in 1:9){
  rf.carseats <- randomForest(Sales~.,data=train,mtry=i,importance=T)
  rf.pred <- predict(rf.carseats,test)
  mse.rf <- mean((rf.pred-test$Sales)^2)
  rf.mse.carseats[i]=mse.rf
}
rf.mse.carseats
plot(rf.mse.carseats)

#rf with m=6
set.seed(1)
rf.carseats.6 <- randomForest(Sales~.,data=train,mtry=6,importance=T)
rf.pred.6 <- predict(rf.carseats.6,test)
mse.rf.6 <- mean((rf.pred.6-test$Sales)^2)
mse.rf.6

#importance
importance(bag.carseats)
importance(rf.carseats.6)
```



The plot above indicates the number of variables tried at each split against its associated test MSE. As m increases the test MSE of the random forest decreases. After 6 variables the decrease in test MSE is very small.

	%IncMSE	IncNodePurity
CompPrice	23.1366974	211.70712
Income	14.5918402	179.80364
Advertising	21.5965451	188.92700
Population	-1.2683059	95.62171
Price	61.3049749	608.07100
ShelveLoc	68.2213381	574.33775
Age	21.9235436	225.24398
Education	3.1288198	81.60381
Urban	0.3628133	10.66051
US	4.0792759	14.10837

From the importance table above the two most important predictors of sales (for a random forest with m=6) are price and shelve location.

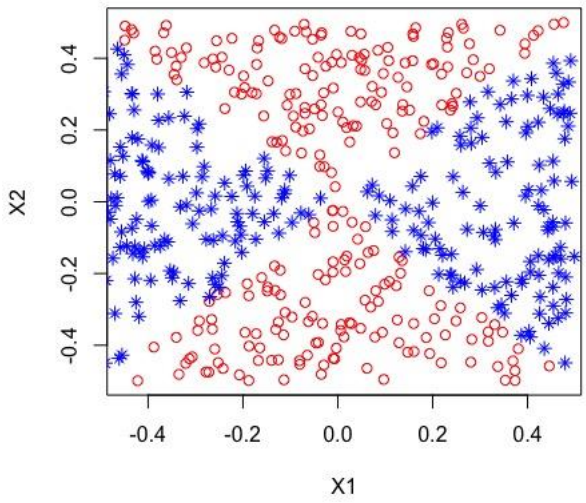
3.

a.

```r
#3a -> create data set with quadratic decision boundary
x1=runif (500) -0.5
x2=runif (500) -0.5
y=1*( x1^2-x2^2 > 0)

my.data <- data.frame(x1,x2,y)
View(my.data)
```

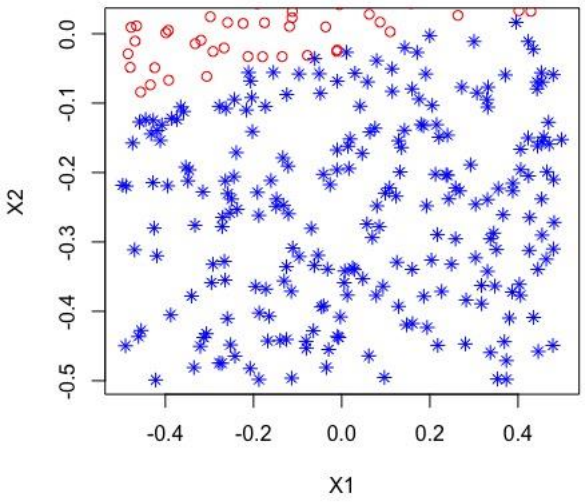
b.

```r
#3b -> plot data
plot(x1[y == 0], x2[y == 0], col = "red", xlab = "X1", ylab = "X2", pch = 1)
points(x1[y == 1], x2[y == 1], col = "blue", pch = 8)
```



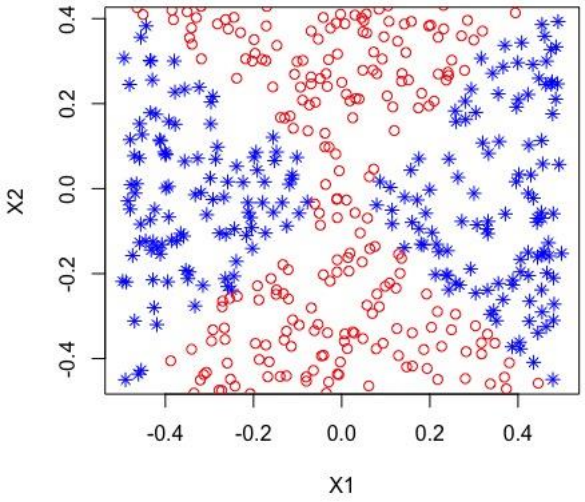
c.

```r
install.packages("e1071")
library(e1071)

#3c -> support vector classifier (svm using linear kernel)
svc.fit <- svm(as.factor(y)~.,data=my.data,kernel="linear",cost=10,scale=F)

#create predictions using svc
svc.pred = predict(svc.fit, my.data)

#split predictions
data.pos = my.data[svc.pred == 1, ]
data.neg = my.data[svc.pred == 0, ]

#Create plots of predicted points
plot(data.pos$x1, data.pos$x2, col = "blue", xlab = "X1", ylab = "X2", pch = 8)
points(data.neg$x1, data.neg$x2, col = "red", pch = 1)
```



d.

```r
#3d -> support vector machine (svm using radial kernel)
svc.fit <- svm(as.factor(y)~.,data=my.data,kernel="radial",gamma=1,cost=1)

#create predictions using svm
svm.pred = predict(svm.fit, my.data)

#split predictions
data.pos.svm = my.data[svm.pred == 1, ]
data.neg.svm = my.data[svm.pred == 0, ]

#Create plots of predicted points
plot(data.pos.svm$x1, data.pos.svm$x2, col = "blue", xlab = "X1", ylab = "X2", pch = 8)
points(data.neg.svm$x1, data.neg.svm$x2, col = "red", pch = 1)
```



e. The linear kernel used in the support vector machine (support vector classifier) did not create an accurate boundary at all. Using a support vector machine with a radial (Gaussian) kernel the predicted boundary (using the entire data set) is very close to the original.

4.

a. This means the proportion of variation explained by the first component is 10%, this is equivalent to saying 90% of the variance is not explained by the first principal component. This is a method of assessing how much of the information of the data set is lost by projecting the data onto the first principal component.

b. A better approach would be to have included another column indicating whether the tissue sample was processed by machine A or B before processing the data. This will improve the proportion of variance explained by the first principal component before applying the two-sample t-test.

c.

```
set.seed(1)
Control <- matrix(rnorm(50 * 1000), ncol = 50)
Treatment <- matrix(rnorm(50 * 1000), ncol = 50)
X <- cbind(Control, Treatment)

# create a linear linear trend in one dimension

X[1, ] <- seq(-18, 18 - .36, .36)
pr.out <- prcomp(scale(X))
summary(pr.out)$importance[, 1]

# Now, adding in A vs B via 10 vs 0 encoding
# and assess the quality of the improved solution.

X <- rbind(X, c(rep(10, 50), rep(0, 50)))
pr.out <- prcomp(scale(X))
summary(pr.out)$importance[, 1]
```

Summary of the first principal component before improvement.

```
Standard deviation Proportion of Variance  Cumulative Proportion
          3.397839                0.115450              0.115450
```

Summary of the first principal component after adding whether the tissue sample was processed by machine A or B.

```
Standard deviation Proportion of Variance  Cumulative Proportion
          3.148148                0.099110              0.099110
```

As it can be seen, by adding whether the tissue sample was processed by machine A or B we have that (11.545-9.911)=1.634% more variance is explained by the first principal component.