

# lab7

YILIN LI

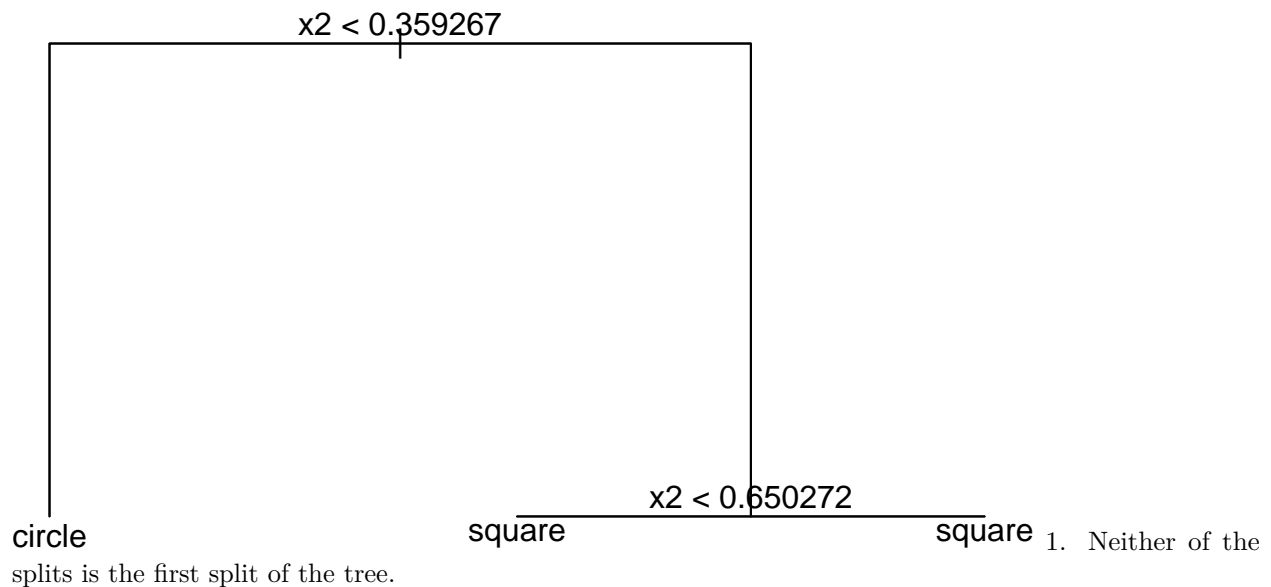
2019/11/8

## Data generation

```
set.seed(75)
n <- 16
x1 <- runif(n)
x2 <- runif(n)
group <- as.factor(sample(1:3, n, replace = TRUE))
levels(group) <- c("circle", "triangle", "square")
df <- data.frame(x1, x2, group)
df[1, 2] <- .765 # tweaks to make a more interesting configuration
df[9, 1] <- .741
df <- df[-7, ]
```

## Q1

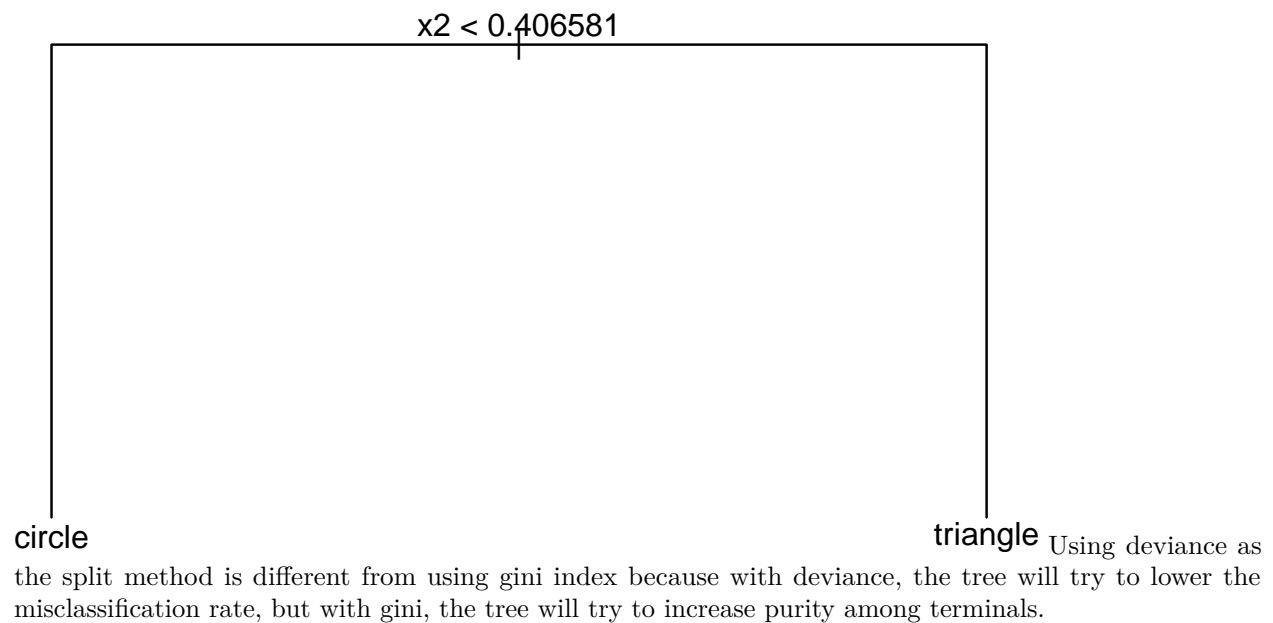
```
library(tree)
t1 = tree(group~., data = df, split = "gini")
plot(t1)
text(t1, pretty = 0)
```



2. Even though the second split predicts all data as square, it's useful because this split could further increase the purity of all the terminals.
3. It would predict "square"

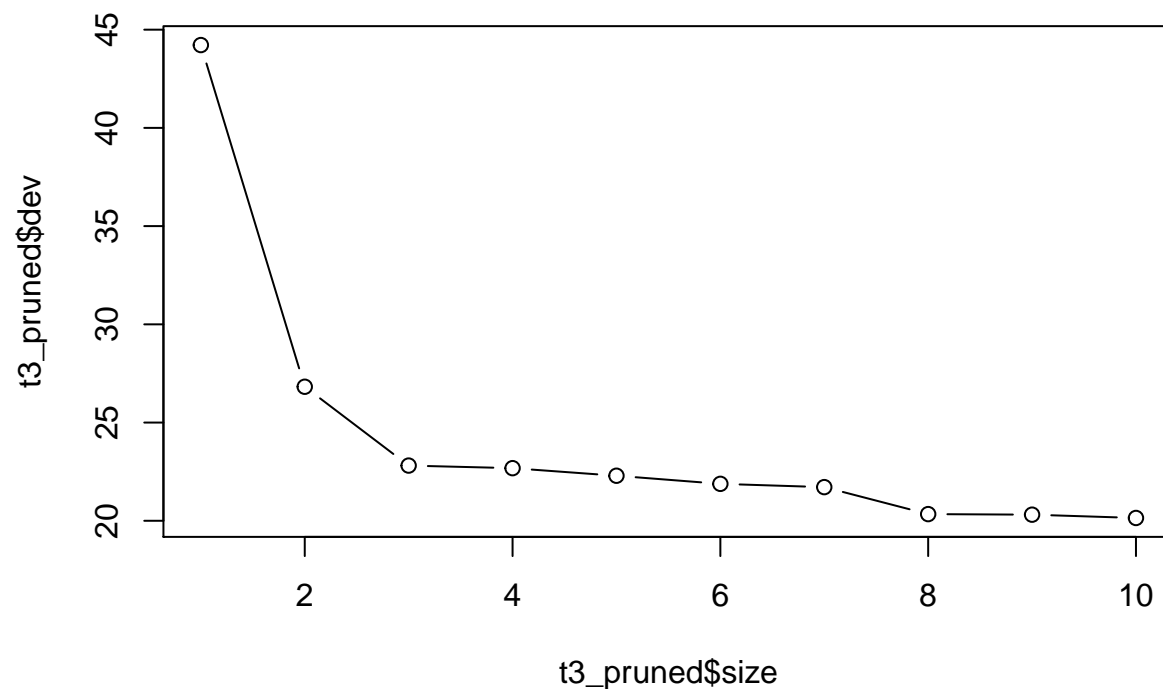
## Q2

```
t2 = tree(group~., data = df, split = "deviance")
plot(t2)
text(t2,pretty = 0)
```

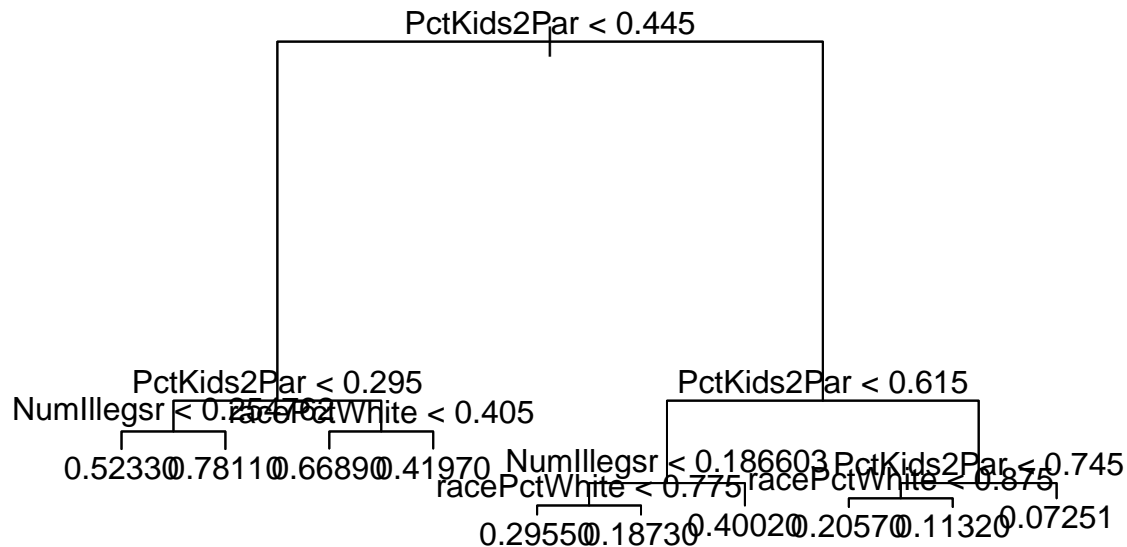


### Q3

```
d = read.csv("http://andrewpbray.github.io/data/crime-train.csv")
d["NumIllegsr"] = sqrt(d$NumIlleg)
t3 = tree(ViolentCrimesPerPop ~ racePctWhite + PctKids2Par + NumIllegsr, data = d)
t3_pruned = cv.tree(t3)
plot(t3_pruned$size, t3_pruned$dev, type = 'b')
```



```
t3_best = prune.tree(t3,method = "deviance",best = 10)
plot(t3_best)
text(t3_best,pretty = 0)
```



Q4

```
test_data <- read.csv("https://bit.ly/2PYS8Ap")
test_data["NumIllegsr"] = sqrt(test_data$NumIlleg)
pd_tree = predict(t3_best, test_data)
MSE_tree = mean((pd_tree - test_data$ViolentCrimesPerPop)^2)
m1 = lm(ViolentCrimesPerPop ~ racePctWhite + PctKids2Par + NumIllegsr, data = d)
pd_lm = predict(m1, test_data)
MSE_lm = mean((pd_lm - test_data$ViolentCrimesPerPop)^2)
```

MSE\_tree

```
## [1] 0.01935877
```

MSE\_lm

```
## [1] 0.0205459
```

So, clearly, the pruned tree has a lower test MSE than our linear fitting.

Q5

```
library(randomForest)
```

```
## randomForest 4.6-14
```

```
## Type rfNews() to see new features/changes/bug fixes.
```

```
rf = randomForest(ViolentCrimesPerPop ~ racePctWhite + PctKids2Par + NumIllegsr, data = d, mtry = 3, imp=0)
pd_rf = predict(rf, test_data)
MSE_rf1 = mean((pd_rf - test_data$ViolentCrimesPerPop)^2)
```

```
rf2 = randomForest(ViolentCrimesPerPop ~ racePctWhite + PctKids2Par + NumIllegsr, data = d, mtry = 1, imp=0)
pd_rf2 = predict(rf2, test_data)
```

```
MSE_rf2 = mean((pd_rf2-test_data$ViolentCrimesPerPop)^2)
MSE_rf1
```

```
## [1] 0.00600712
```

```
MSE_rf2
```

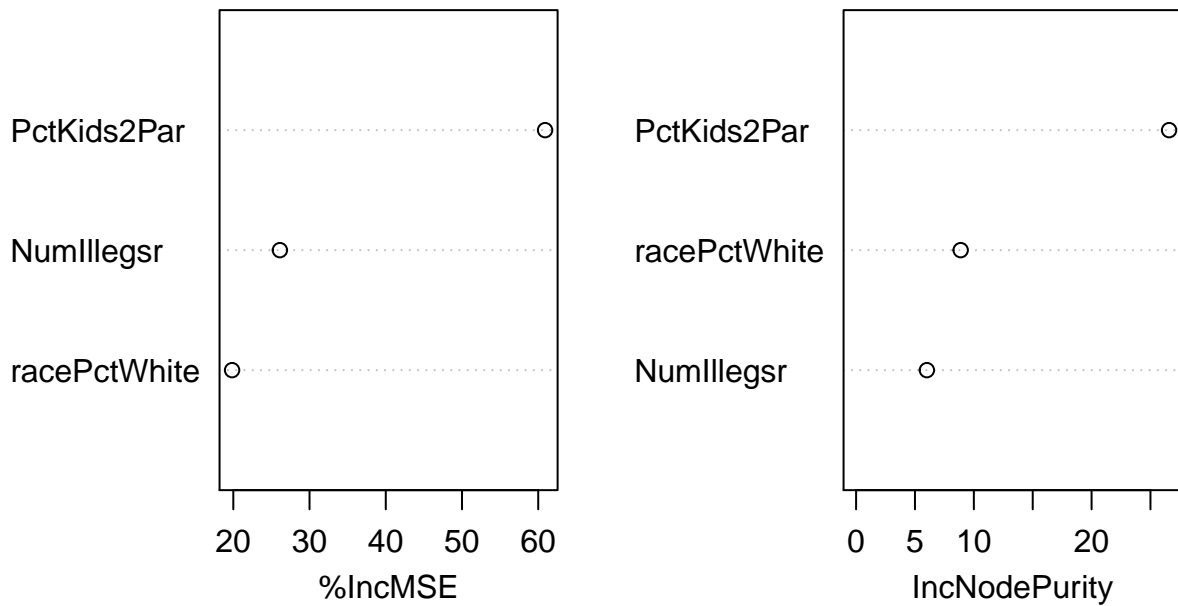
```
## [1] 0.009030661
```

It turns out both the bagging trees and the random forest have lower test MSE than the pruned tree and linear regression model.

## Q6

```
importance = varImpPlot(rf)
```

rf



The result turns out to be quite similar to our analysis in our linear model. “PctKids2Par” has the highest correlation with the “ViolentCrimesPerPop” and here also influences the randomForest model the most.