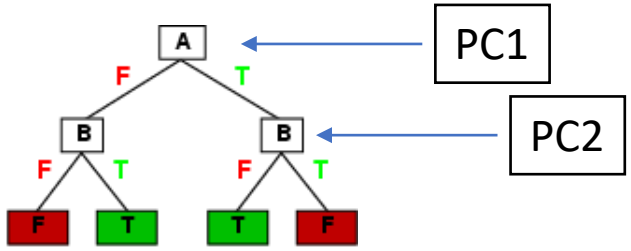
Exercise 4.1.1

Compute the optimal decision point for the first 5 PCAs of a dataset (e.g. a single person) and compute the information gain associated to it (plot 5 graphs, one for each component, and show the highest information gain). See slides for how to compute information gain.

A	B	A xor B
F	F	F
F	T	T
T	F	T
T	T	F



Page 35

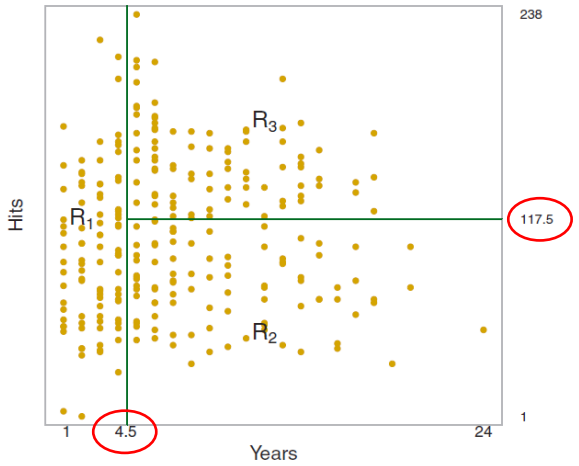


FIGURE 8.2. *The three-region partition for the* Hitters *data set from the regression tree illustrated in Figure 8.1.*

Page 23

- Information Gain (IG) or reduction in entropy from the attribute test:

$$IG(A) = Entropy\ before\ -\ Entropy\ after$$

Page 17

$$Entropy(dice) = -p(s=1)\log[p(s=1)] - p(s=2)\log[p(s=2)] \ldots - p(s=n)\log[p(s=n)]$$

Page 21

1/6 of the time we enter "None", so we weight "None" with 1/6.
Similarly: "Some" has weight: 1/3 and "Full" has weight ½.

p=6, n=6 p_1=0 n_1=2 p_2=4 n_2=0 p_3=2 n_1=4

$$Entropy(A) = \sum_{i=1}^{n} \frac{p_i + n_i}{p + n} Entropy\left(\frac{p_i}{p_i + n_i}, \frac{n_i}{p_i + n_i}\right)$$

weight for each branch

entropy for each branch.

Calculate entropy

```
entropy <- function(S) { # Function to calculate entropy as in the slides
  fullSum <- 0
  for( i in (0:9) ) {
    if( nrow(S) > 0 ) # Make sure that there is something in the list
      { pi <- nrow(S[ S[,1] == i , ]) / nrow(S) }else{pi <- 0}
    if(pi > 0 ){
    fullSum <- fullSum - pi * log2(pi)
    }
  }
  return(fullSum)
}
```

Decision point

Try a series of 'SplitPoint' from min(x) to max(x), and calculate their corresponding information gain.

```
id_pca1 <- id_pca$x[, 1]
Pts <- seq(min(id_pca1), max(id_pca1), length.out=200)
for( splitP in (1:200)){
   S1 <- id[ id_pca1 < Pts[splitP], ]   # Perform splits
   S2 <- id[ id_pca1 >= Pts[splitP], ]
   s1 <- nrow(S1)
   s2 <- nrow(S2)
   ent <- ( s1 * entropy(S1) )/(s1 + s2) + ( s2 * entropy(S2) )/(s1 + s2)  # Calculate
entropy
   entList[splitP] <-  entBefore - ent        # Information gain is calculated
 }
```

Exercise 4

When you work on raw data, you may need to generate a formula to link the ‘cypher’ to 324 pixels.

```
model.randomforest <- randomForest(V1 ~ . , data = id)
```