Homework 4

Jacopo Liera

2023-10-28

Exercise 1

A binary dependent variable is generated by:

$$Pr(Y = 1|X) = q + (1 - 2q) \ \cdot \nVdash \left[\sum_{j=1}^{J} X_j > \frac{J}{2} \right]$$

where 1[] is the indicator function$X \sim U(0,1)^p$, $0 \leq q \leq \frac{1}{2}$, $J \leq p$ is some predefined (even) number. Describe the probability surface and give the Bayes error rate.

Solution

The sum of n independent random variables $X_j \sim U(0,1)$ is Irwin-Hall Distributed, with cdf:

$$F_X = \frac{1}{n!} \sum_{k=0}^{|x|} (-1)^k \binom{n}{k} (x-k)^{n-1}$$

with support $x \in [0, n]$. Also, $E[X] = \frac{n}{2}$. Given the median being also $\frac{n}{2}$, we know the distribution function to be symmetric, further implying that we have 50% probability to get 1 out of the indicator function, and 50% to get a 0, no matter the J.

From Chapter 2 we have the Bayes Error defined as:

$$Err_{Bayes} = 1 - E\left[max_{j \in (0,1)} \ P(Y = j|X) \right]$$

We also have that:

$$max_{j \in (0,1)} \ P(Y = j|X) = \begin{cases} P(Y = 1|X) \ \ if \ \ P(Y = 1|X) > 0.5 \\ \\ 1 - P(Y = 1|X) \ \ if \ \ P(Y = 1|X) \leq 0.5 \end{cases}$$

Since the $\nVdash[]$ takes values of either 0 or 1 50% of the times and $0 \leq q \leq 0.5$, we have that:

$$max_{j \in (0,1)} \ P(Y = j|X) = \begin{cases} P(Y = 1|X) = 1 - q \ \ if \ \ \nVdash[] = 1 \\ \\ P(Y = 0|X) = 1 - q \ \ if \ \ \nVdash[] = 0 \end{cases}$$

Clearly, we have that $max_{j \in (0,1)} \ P(Y = j|X) = 1 - q$ and therefore:

$$Err_{Bayes} = 1 - E\left[max_{j \in (0,1)} \ P(Y = j|X) \right] = q$$

Exercise 9

We assume data with the following data generation process:

$$x = y + \epsilon$$

where y is a categorical variable with values 1, 2, 3, which occur with equal probability and $\epsilon \sim N(0, 0.2)$ independent.

- Draw 100 data sets of size 100

Here we simply sample the classes in y, the white noise ϵ to generate the observations x.

```r
set.seed(1910)
data_gen <- function(n_samples){

  i <- 1
  vmat <- list()

  repeat{

    # Break Condition
    if(i == (n_samples + 1)) break

    # Generate 100 random numbers
    y <- sample(c(1, 2, 3), size = 100, replace = T)
    eps <- rnorm(100, mean = 0, sd = sqrt(0.2))
    x <- y + eps

    # Put it in a matrix
    tmp <- data.frame(x, y, eps)
    vmat[[i]] <-  tmp

    i <- i + 1
  }

  return(vmat)
}

vmat <- data_gen(n_samples = 100)
```

- Determine the sum of the misclassification rates, Gini indices and deviance criteria weighted with the number of observations in each subgroup for the subgroups obtained when splitting the observations using x with thresholds 1.5, 2, and 2.5 and y as dependent variable in the classification problem.

```r
# Impurity measures functions
misc_err <- function(p) return( 1 - max(p) )
gini <- function(p) return( sum(p * (1-p)) )
deviance <- function(p) return( (-1) * sum(p * log(p)) )

# Given thresholds
thresholds <- c(1.5, 2, 2.5)
```

```
# Function that calculates the impurity given the impurty measure function,
# the threshold for the split for one single dataset.

impurity_calculate <- function(df, class = "y", col = "x", thresh, FUN){

  # Slice the dataset given the threshold into 2 subgroups
  df$split <- cut(df[, col], c(-Inf, thresh, Inf))

  # 0 is the <= thresh , 1 is the >thresh
  levels(df$split) <- c(0, 1)

  # Number of observations for weighting later on
  nobs <- table(df$split)

  # Derive a matrix to count the observations for each split
  class_mat <- table(df[, c("split", class)])
  calc_mat <- t(apply(class_mat, 1, function(x) x/sum(x)))

  # Impurity function act here
  res <- apply(calc_mat, 1, FUN)

  # Pay attention to NaN Values
  res[is.nan(res)] <- 0

  # Weighted impurity measure for the split.
  res_w <- res/nobs

  # Return the overall impurity as the sum for each group
  return(sum(res_w))
}

generate_error_tables <- function(impurity_criterion, thresholds, df_list){

  j <- 1

  out_mat <- matrix(rep(0, 300), nrow = 100)
  colnames(out_mat) <- thresholds

  for(t in thresholds){

    for(i in 1:length(df_list)){

      tmp <- impurity_calculate(df_list[[i]], class = "y", col = "x", thresh = t, FUN = impurity_criteri
      out_mat[i, j] <- tmp

    }

    j <- j + 1

  }

  return(out_mat)
}

# Generate error tables: these contain the sum of the errors for each split for each dataset
deviance_table <- generate_error_tables(deviance, thresholds, vmat)
gini_table <- generate_error_tables(gini, thresholds, vmat)
misc_table <- generate_error_tables(misc_err, thresholds, vmat)

sum_table <- rbind(colSums(deviance_table), colSums(gini_table), colSums(misc_table))
rownames(sum_table) <- c("Deviance", "Gini", "Misclassification")

kableExtra::kable(sum_table)
```

	1.5	2	2.5
Deviance	1.306588	0.9548826	1.290397
Gini	1.482440	1.8168190	1.527013
Misclassification	1.109248	1.4005282	1.136874

- Calculate the best threshold according to each of the three impurity measures for each of the 100 data sets. Summarize and interpret the results.

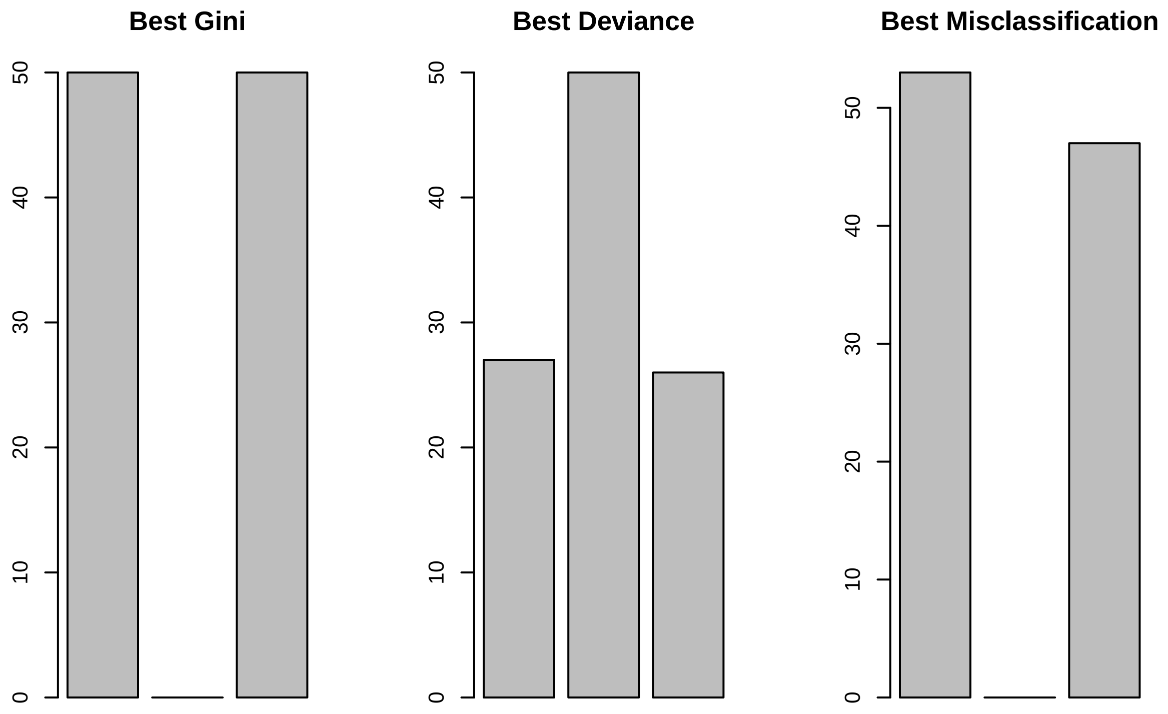
```
best <- function(vector) as.integer(vector == min(vector))

best_deviance <- apply(t(apply(deviance_table, 1, best)), 2, sum)
best_gini <- apply(t(apply(gini_table, 1, best)), 2, sum)
best_misc <- apply(t(apply(misc_table, 1, best)), 2, sum)

best_matrix <- matrix(c(best_misc, best_gini, best_deviance), nrow = 3, byrow = T)
rownames(best_matrix) <- c("Misclasssification", " Gini", "Deviance")
colnames(best_matrix) <- c("1.5", "2", "2.5")

kableExtra::kable(best_matrix)
```

	1.5	2	2.5
Misclasssification	53	0	47
Gini	50	0	50
Deviance	27	50	26


Best Gini
Best Deviance
Best Misclassification
0
10
20
30
40
50