TSF.R

HIMANI

2020-09-26

library(tidyverse)

## -- Attaching packages ----------------------------------------- tidyverse 1.3.0 --

## v ggplot2 3.3.2 v purrr 0.3.4  
## v tibble 3.0.1 v dplyr 1.0.0  
## v tidyr 1.1.0 v stringr 1.4.0  
## v readr 1.3.1 v forcats 0.5.0

## -- Conflicts -------------------------------------------- tidyverse\_conflicts() --  
## x dplyr::filter() masks stats::filter()  
## x dplyr::lag() masks stats::lag()

library(caret)

## Loading required package: lattice

##   
## Attaching package: 'caret'

## The following object is masked from 'package:purrr':  
##   
## lift

library(party)

## Loading required package: grid

## Loading required package: mvtnorm

## Loading required package: modeltools

## Loading required package: stats4

## Loading required package: strucchange

## Loading required package: zoo

##   
## Attaching package: 'zoo'

## The following objects are masked from 'package:base':  
##   
## as.Date, as.Date.numeric

## Loading required package: sandwich

##   
## Attaching package: 'strucchange'

## The following object is masked from 'package:stringr':  
##   
## boundary

#import the data  
data.iris <- read\_csv("C:\\Users\\HIMANI\\Downloads\\Iris.csv")

## Parsed with column specification:  
## cols(  
## Id = col\_double(),  
## SepalLengthCm = col\_double(),  
## SepalWidthCm = col\_double(),  
## PetalLengthCm = col\_double(),  
## PetalWidthCm = col\_double(),  
## Species = col\_character()  
## )

#look at the data  
glimpse(data.iris)

## Rows: 150  
## Columns: 6  
## $ Id <dbl> 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16...  
## $ SepalLengthCm <dbl> 5.1, 4.9, 4.7, 4.6, 5.0, 5.4, 4.6, 5.0, 4.4, 4.9, 5.4...  
## $ SepalWidthCm <dbl> 3.5, 3.0, 3.2, 3.1, 3.6, 3.9, 3.4, 3.4, 2.9, 3.1, 3.7...  
## $ PetalLengthCm <dbl> 1.4, 1.4, 1.3, 1.5, 1.4, 1.7, 1.4, 1.5, 1.4, 1.5, 1.5...  
## $ PetalWidthCm <dbl> 0.2, 0.2, 0.2, 0.2, 0.2, 0.4, 0.3, 0.2, 0.2, 0.1, 0.2...  
## $ Species <chr> "Iris-setosa", "Iris-setosa", "Iris-setosa", "Iris-se...

summary(data.iris)

## Id SepalLengthCm SepalWidthCm PetalLengthCm   
## Min. : 1.00 Min. :4.300 Min. :2.000 Min. :1.000   
## 1st Qu.: 38.25 1st Qu.:5.100 1st Qu.:2.800 1st Qu.:1.600   
## Median : 75.50 Median :5.800 Median :3.000 Median :4.350   
## Mean : 75.50 Mean :5.843 Mean :3.054 Mean :3.759   
## 3rd Qu.:112.75 3rd Qu.:6.400 3rd Qu.:3.300 3rd Qu.:5.100   
## Max. :150.00 Max. :7.900 Max. :4.400 Max. :6.900   
## PetalWidthCm Species   
## Min. :0.100 Length:150   
## 1st Qu.:0.300 Class :character   
## Median :1.300 Mode :character   
## Mean :1.199   
## 3rd Qu.:1.800   
## Max. :2.500

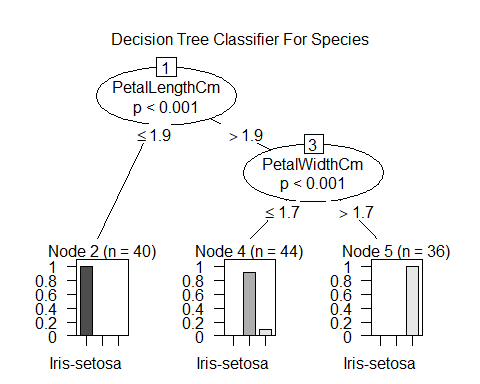
# Split the data into training and test set  
training.samples <- data.iris$Species %>%  
 createDataPartition(p = 0.8, list = FALSE)  
train.data <- data.iris[training.samples,-1]

## Warning: The `i` argument of ``[`()` can't be a matrix as of tibble 3.0.0.  
## Convert to a vector.  
## This warning is displayed once every 8 hours.  
## Call `lifecycle::last\_warnings()` to see where this warning was generated.

test.data <- data.iris[-training.samples,-1]  
  
#Fit a model on the data  
model <- train( Species ~., data = train.data, method = "ctree2",  
 trControl = trainControl("cv", number = 10))  
model

## Conditional Inference Tree   
##   
## 120 samples  
## 4 predictor  
## 3 classes: 'Iris-setosa', 'Iris-versicolor', 'Iris-virginica'   
##   
## No pre-processing  
## Resampling: Cross-Validated (10 fold)   
## Summary of sample sizes: 108, 108, 108, 108, 108, 108, ...   
## Resampling results across tuning parameters:  
##   
## maxdepth mincriterion Accuracy Kappa   
## 1 0.01 0.6583333 0.4875  
## 1 0.50 0.6583333 0.4875  
## 1 0.99 0.6583333 0.4875  
## 2 0.01 0.9500000 0.9250  
## 2 0.50 0.9500000 0.9250  
## 2 0.99 0.9500000 0.9250  
## 3 0.01 0.9500000 0.9250  
## 3 0.50 0.9500000 0.9250  
## 3 0.99 0.9500000 0.9250  
##   
## Accuracy was used to select the optimal model using the largest value.  
## The final values used for the model were maxdepth = 2 and mincriterion = 0.01.

#visualize the Decision Tree  
plot(model$finalModel , main ="Decision Tree Classifier For Species")



# predictions on the test data  
predicted.classes <- model %>% predict(test.data)  
  
# model accuracy rate on test data  
mean(predicted.classes == test.data$Species)

## [1] 0.9333333

# We have an accuracy of 93% , that means our model is good fit to the data .