Section 9.3 | Classification-Decision Tree

Mohammad Saqib Ansari

2023-12-20

Decision Trees

Decision Trees are versatile supervised machine learning algorithms that can perform both classification and regression tasks. They mimic human decision-making processes by breaking down a complex decision-making process into a series of simpler decisions. In the context of predictive modeling, they recursively split the dataset into subsets based on the most significant attributes or features. These splits create a tree-like structure, where each internal node represents a decision based on a feature, each branch represents the outcome of that decision, and each leaf node represents the final decision or outcome.

Structure of a Decision Tree:

1. **Root Node**: It represents the entire dataset and is the starting point for the tree.
2. **Decision Nodes (Internal Nodes)**: These nodes split the dataset into smaller subsets based on specific feature conditions. Each decision node tests a particular feature and outcome based on the feature's value.
3. **Branches**: Represent the possible outcomes of a decision node based on the feature's conditions.
4. **Leaf Nodes**: Also known as terminal nodes, these nodes contain the final decision or prediction. They don't split further and represent the final outcome or class label.

How Decision Trees Work:

1. **Attribute Selection**: Decision Trees determine the best attribute to split the dataset at each node. Various algorithms like ID3, C4.5, CART (Classification and Regression Trees), or Gini Index are used to measure the importance of different attributes for splitting.
2. **Splitting Criteria**: The splitting process aims to maximize information gain (or decrease in impurity) at each node. The goal is to create homogeneous subsets in terms of the target variable.
3. **Recursive Splitting**: After the initial split, the process continues recursively for each subset created until a stopping criterion is met. This could be reaching a maximum depth, having a minimum number of samples in a node, or other conditions.
4. **Prediction**: When new data is presented, it traverses the tree from the root node through various decision nodes according to the feature values until it reaches a leaf node, where the final prediction or classification is made.

Advantages of Decision Trees:

1. **Interpretability**: Decision Trees are easy to understand and interpret. They mimic human decision-making, making the logic behind predictions transparent.
2. **Handling Non-linearity**: They can handle non-linear relationships between features and the target variable effectively.
3. **No Assumptions about Data**: They don't make any assumptions about the distribution of data or relationships between features.

Disadvantages of Decision Trees:

1. **Overfitting**: Decision Trees can easily overfit the training data, especially when the tree is deep or not pruned properly.
2. **Instability**: Small variations in the data can lead to a completely different tree, making them less stable compared to other algorithms.
3. **Bias Towards Dominant Classes**: In classification problems with imbalanced class distributions, decision trees tend to be biased toward the dominant class.

Implementation in R:

In R, decision trees can be implemented using packages like `rpart`, `tree`, `randomForest`, or `C50`. The `rpart` package, for instance, provides functionality for recursive partitioning for classification and regression.

Example R code snippet for fitting a decision tree using `rpart` package:

Decision Tree

```
# Reading the Titanic dataset
datt <- read.csv("titanic.csv")

# Displaying the first few rows of the dataset
head(datt)
```

Step 1 | Reading the data

```
##   PassengerId Survived Pclass    Sex Age SibSp Parch    Fare
## 1           1        0      3   male  22     1     0  7.2500
## 2           2        1      1 female  38     1     0 71.2833
## 3           3        1      3 female  26     0     0  7.9250
## 4           4        1      1 female  35     1     0 53.1000
## 5           5        0      3   male  35     0     0  8.0500
## 6           7        0      1   male  54     0     0 51.8625
##   Embarked
## 1        S
## 2        C
## 3        S
## 4        S
## 5        S
## 6        S
```

```r
# Summary statistics of the dataset
summary(datt)
```

```
##   PassengerId       Survived          Pclass
##  Min.   :  1.0   Min.   :0.0000   Min.   :1.00
##  1st Qu.:222.8   1st Qu.:0.0000   1st Qu.:1.00
##  Median :445.0   Median :0.0000   Median :2.00
##  Mean   :448.6   Mean   :0.4045   Mean   :2.24
##  3rd Qu.:677.2   3rd Qu.:1.0000   3rd Qu.:3.00
##  Max.   :891.0   Max.   :1.0000   Max.   :3.00
##      Sex                 Age            SibSp
##  Length:712         Min.   : 0.42   Min.   :0.000
##  Class :character   1st Qu.:20.00   1st Qu.:0.000
##  Mode  :character   Median :28.00   Median :0.000
##                     Mean   :29.64   Mean   :0.514
##                     3rd Qu.:38.00   3rd Qu.:1.000
##                     Max.   :80.00   Max.   :5.000
##      Parch             Fare          Embarked
##  Min.   :0.0000   Min.   :  0.00   Length:712
##  1st Qu.:0.0000   1st Qu.:  8.05   Class :character
##  Median :0.0000   Median : 15.65   Mode  :character
##  Mean   :0.4326   Mean   : 34.57
##  3rd Qu.:1.0000   3rd Qu.: 33.00
##  Max.   :6.0000   Max.   :512.33
```

```r
# Dimensions of the dataset
dim(datt)
```

```
## [1] 712   9
```

Now encoding embarked variable and data preprocessing:

```r
# Convert 'Embarked' and "Sex" columns to factor variables
datt$Embarked <- factor(datt$Embarked)
datt$Sex <- factor(datt$Sex)

# Displaying unique values in 'Embarked' column
unique(datt$Embarked)
```

```
## [1] S C Q
## Levels: C Q S
```

```r
unique(datt$Sex)
```

```
## [1] male   female
## Levels: female male
```

```r
# Mapping 'Embarked' categories to numeric values (0, 1, 2)
datt$Embarked <- as.numeric(datt$Embarked) - 1
datt$Sex <- as.numeric(datt$Sex) - 1
```

```r
# Selecting relevant columns for analysis
datT <- datt[, c(2, 3, 4, 5, 6)]

# Displaying the first few rows of the processed dataset
head(datT)
```

```
##   Survived Pclass Sex Age SibSp
## 1        0      3   1  22     1
## 2        1      1   0  38     1
## 3        1      3   0  26     0
## 4        1      1   0  35     1
## 5        0      3   1  35     0
## 6        0      1   1  54     0
```

```r
# Summary statistics of the processed dataset
summary(datT)
```

```
##     Survived          Pclass          Sex              Age
##  Min.   :0.0000   Min.   :1.00   Min.   :0.0000   Min.   : 0.42
##  1st Qu.:0.0000   1st Qu.:1.00   1st Qu.:0.0000   1st Qu.:20.00
##  Median :0.0000   Median :2.00   Median :1.0000   Median :28.00
##  Mean   :0.4045   Mean   :2.24   Mean   :0.6362   Mean   :29.64
##  3rd Qu.:1.0000   3rd Qu.:3.00   3rd Qu.:1.0000   3rd Qu.:38.00
##  Max.   :1.0000   Max.   :3.00   Max.   :1.0000   Max.   :80.00
##      SibSp
##  Min.   :0.000
##  1st Qu.:0.000
##  Median :0.000
##  Mean   :0.514
##  3rd Qu.:1.000
##  Max.   :5.000
```

```r
# Loading the required library for decision tree
library(rpart)

# Constructing a decision tree considering all available features
DT <- rpart(Survived ~ ., data = datT)
DT
```

Step 2 | Constructing and Analyzing the Tree

```
## n= 712
##
## node), split, n, deviance, yval
##       * denotes terminal node
##
##  1) root 712 171.5056000 0.40449440
##    2) Sex>=0.5 453  73.9072800 0.20529800
##      4) Age>=6.5 429  63.1794900 0.17948720
```

```
##        8) Pclass>=1.5 330  34.3909100 0.11818180 *
##        9) Pclass< 1.5 99  23.4141400 0.38383840
##         18) Age>=53 22   2.5909090 0.13636360 *
##         19) Age< 53 77  19.0909100 0.45454550 *
##      5) Age< 6.5 24   5.3333330 0.66666670
##       10) SibSp>=2.5 9   0.8888889 0.11111110 *
##       11) SibSp< 2.5 15   0.0000000 1.00000000 *
##    3) Sex< 0.5 259  48.1853300 0.75289580
##      6) Pclass>=2.5 102  25.3431400 0.46078430
##       12) Age>=38.5 12   0.9166667 0.08333333 *
##       13) Age< 38.5 90  22.4888900 0.51111110 *
##      7) Pclass< 2.5 157   8.4840760 0.94267520 *
```

```r
# Constructing a decision tree considering 'Sex' and 'Age' as features
DT2 <- rpart(Survived ~ Sex + Age, data = datT)
DT2
```

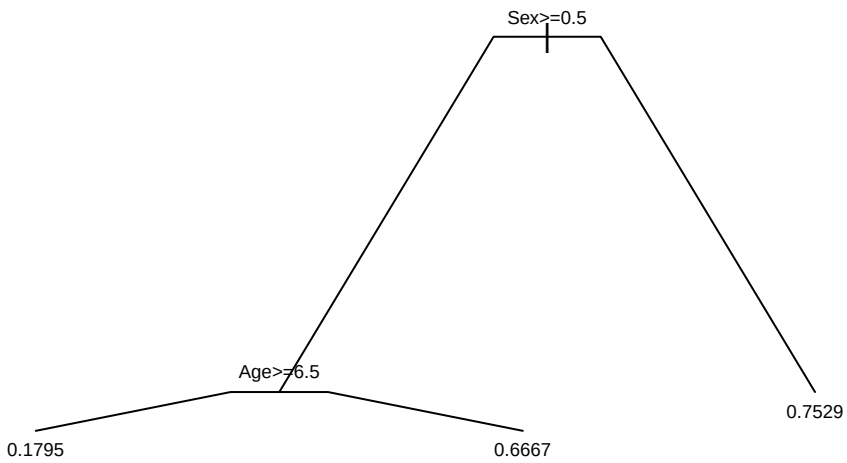
```
## n= 712
##
## node), split, n, deviance, yval
##       * denotes terminal node
##
## 1) root 712 171.505600 0.4044944
##   2) Sex>=0.5 453  73.907280 0.2052980
##     4) Age>=6.5 429  63.179490 0.1794872 *
##     5) Age< 6.5 24   5.333333 0.6666667 *
##   3) Sex< 0.5 259  48.185330 0.7528958 *
```

Assuming you have a decision tree object named DT2:

```r
# Plotting the decision tree
plot(DT2,
     branch = 0.2,         # Length of branches
     compress = TRUE,      # Compress the plot
     margin = 0.1,         # Margin size
     main = "Decision Tree 2"
)

# Adjusting text size in the plot for better readability
text(DT2,
     cex = 0.6,            # Reducing text size for nodes
     col = "black"         # Setting color for text
)
```

Decision Tree 2




```
# Displaying a summary of the decision tree
summary(DT2)
```

```
## Call:
## rpart(formula = Survived ~ Sex + Age, data = datT)
##   n= 712
##
##           CP nsplit rel error    xerror       xstd
## 1 0.28811304      0 1.0000000 1.0020421 0.01463386
## 2 0.03145357      1 0.7118870 0.7138369 0.03607742
## 3 0.01000000      2 0.6804334 0.6951600 0.03680813
##
## Variable importance
## Sex Age
##  89  11
##
## Node number 1: 712 observations,    complexity param=0.288113
##   mean=0.4044944, MSE=0.2408787
##   left son=2 (453 obs) right son=3 (259 obs)
##   Primary splits:
##       Sex < 0.5  to the right, improve=0.28811300, (0 missing)
##       Age < 6.5  to the right, improve=0.02599207, (0 missing)
##   Surrogate splits:
##       Age < 15.5 to the right, agree=0.64, adj=0.012, (0 split)
##
## Node number 2: 453 observations,    complexity param=0.03145357
```

```
##   mean=0.205298, MSE=0.1631507
##   left son=4 (429 obs) right son=5 (24 obs)
##   Primary splits:
##       Age < 6.5  to the right, improve=0.07298961, (0 missing)
##
## Node number 3: 259 observations
##   mean=0.7528958, MSE=0.1860437
##
## Node number 4: 429 observations
##   mean=0.1794872, MSE=0.1472715
##
## Node number 5: 24 observations
##   mean=0.6666667, MSE=0.2222222
```

Cost-Sensitive Consideration: When working with decision trees, especially in classification problems like predicting survival in the Titanic dataset, considering the costs associated with misclassification is crucial.

For instance, in this survival prediction scenario, misclassifying a passenger who did not survive as having survived (false positive) or misclassifying a surviving passenger as not survived (false negative) can have different consequences or costs. These costs could vary in real-world applications, impacting decisions or actions based on the model predictions.

Adjusting the decision tree model to minimize these misclassification costs often involves techniques like changing class weights, using different evaluation metrics that consider costs (such as weighted precision or recall), or incorporating cost matrices to guide the model's learning process towards reducing these specific misclassification errors.

Step 3 | Choosing the size of the tree

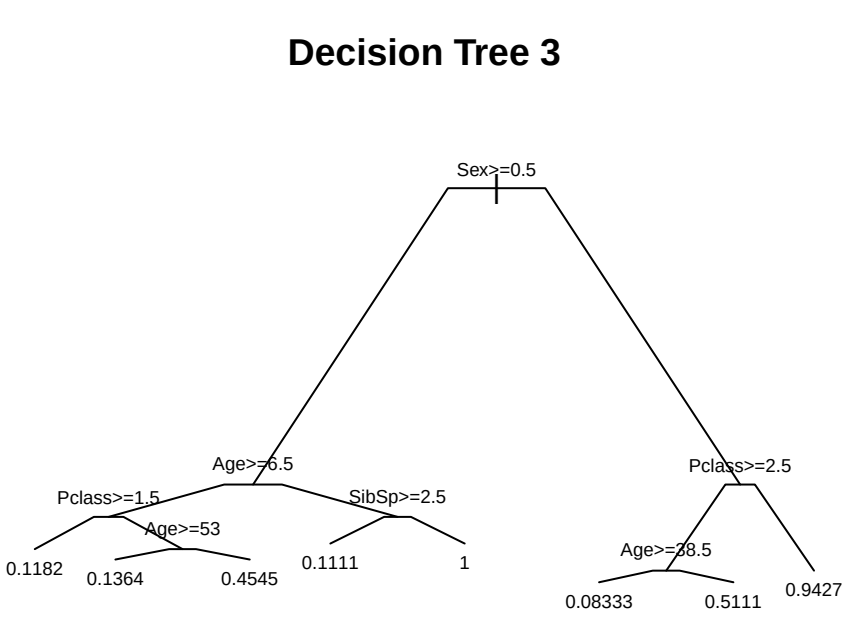
```r
# Building a decision tree with a minimum of 5 observations required in a node for splitting
DT3 <- rpart(Survived ~ ., data = datT, minsplit = 5)
DT3
```

```
## n= 712
##
## node), split, n, deviance, yval
##       * denotes terminal node
##
##  1) root 712 171.5056000 0.40449440
##    2) Sex>=0.5 453  73.9072800 0.20529800
##      4) Age>=6.5 429  63.1794900 0.17948720
##        8) Pclass>=1.5 330  34.3909100 0.11818180 *
##        9) Pclass< 1.5 99  23.4141400 0.38383840
##         18) Age>=53 22   2.5909090 0.13636360 *
##         19) Age< 53 77  19.0909100 0.45454550 *
##      5) Age< 6.5 24   5.3333330 0.66666670
##       10) SibSp>=2.5 9   0.8888889 0.11111110 *
##       11) SibSp< 2.5 15   0.0000000 1.00000000 *
##    3) Sex< 0.5 259  48.1853300 0.75289580
##      6) Pclass>=2.5 102  25.3431400 0.46078430
##       12) Age>=38.5 12   0.9166667 0.08333333 *
```

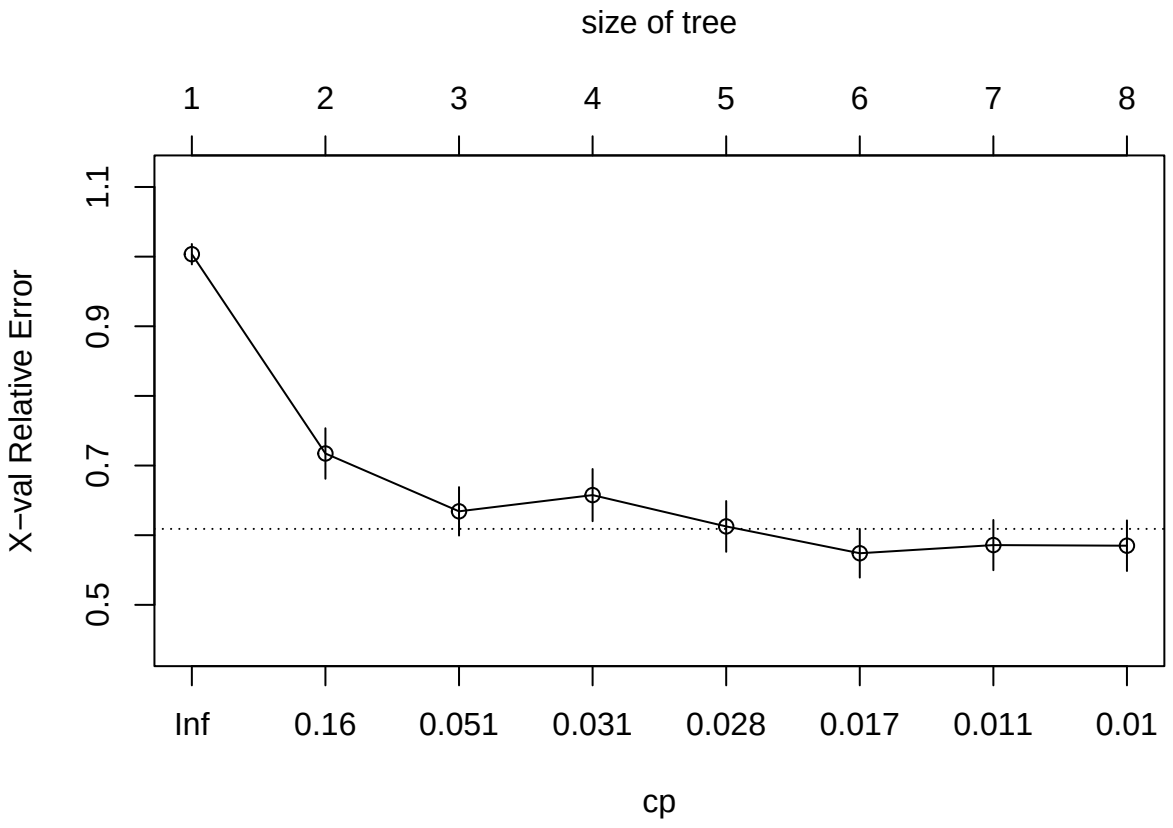
```
##       13) Age< 38.5 90  22.4888900 0.51111110 *
##      7) Pclass< 2.5 157   8.4840760 0.94267520 *
```

```r
# Plotting Decision Tree 3
plot(DT3,
     branch = 0.2,         # Length of branches
     compress = TRUE,      # Compress the plot
     margin = 0.1,         # Margin size
     main = "Decision Tree 3"
)

# Adjusting text size in the plot
text(DT3,
     cex = 0.6,            # Reduce text size for nodes
     col = "black"         # Set color for text
)
```



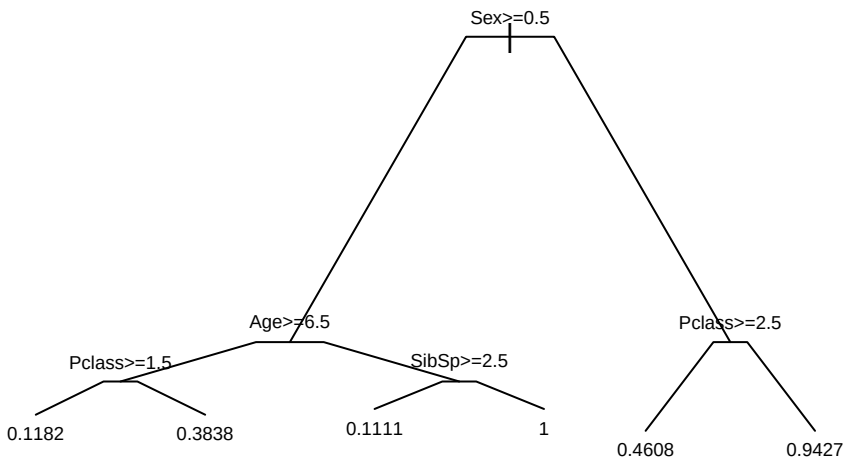

```r
# Pruning the tree using cross-validation
DT3 <- rpart(Survived ~ ., data = datT, minsplit = 5, xval = 53)
plotcp(DT3)
```




```r
# Constructing the final pruned tree based on chosen complexity parameter (cp)
DTF <- rpart(Survived ~ ., data = datT, minsplit = 5, cp = 0.017)

# Plotting the final pruned Decision Tree
plot(DTF,
     branch = 0.2,         # Length of branches
     compress = TRUE,      # Compress the plot
     margin = 0.1,         # Margin size
     main = "Decision Tree Final"
)

# Adjusting text size in the final plot for better readability
text(DTF,
     cex = 0.6,            # Reduce text size for nodes
     col = "black"         # Set color for text
)
```




```
# Prediction and Evaluation

# Predicting Survival based on the final tree for existing data
pred <- predict(DTF, newdata = datT)

# Creating a confusion matrix to assess model accuracy
tab <- table(datT$Survived, round(pred))
tab
```

```
##
##       0   1
##   0 415   9
##   1 125 163
```

```
# Calculating prediction error
error <- sum(round(pred) != datT$Survived) / nrow(datT)
error
```

```
## [1] 0.1882022
```

Interpretation of the Decision Tree (DTF):

Tree Overview:

- **n=712**: Total number of observations in the dataset used to build the tree.

- The tree structure is represented with nodes, splits, number of observations (n), deviance, and yval (predicted value).

Nodes and Splits:

1. **Root Node (Node 1)**: The initial starting point of the tree with 712 observations. The predicted value (yval) for this node is 0.4044944.
2. **Split based on Sex (Node 2)**:
   - When Sex is greater than or equal to 0.5 (interpreted as 'male'), it creates two branches:
     - **453 observations** satisfy this condition. The predicted value (yval) for this node is 0.2052980.
   - When Sex is less than 0.5 (interpreted as 'female'), it further divides the dataset:
     - **259 observations** satisfy this condition. The predicted value (yval) for this node is 0.7528958.
3. **Subsequent splits**:
   - For males (Node 4 and Node 5):
     - Node 4 and Node 5 split based on the Age feature.
     - Age >= 6.5 and Age < 6.5 create subsets of the data.
     - The predicted values (yval) and number of observations for terminal nodes (denoted by *) are also provided.
   - For females (Node 6 and Node 7):
     - Similar to males, females are split based on the Pclass (passenger class) feature.
     - Terminal nodes with predicted values (yval) and number of observations are also displayed.

Interpretation Summary:

- This decision tree (DTF) demonstrates the hierarchical splitting of the dataset based on selected features (Sex, Age, and Pclass), aiming to predict survival outcomes.
- It provides a clear representation of the conditions leading to different survival predictions for various subgroups within the dataset.
- Terminal nodes (*) signify the final prediction of survival or non-survival based on the specified conditions.

Step 4 | Predicting survival for new values

```
# Generating new data for prediction
new.d <- data.frame(matrix(c(3, 1, 28, 2, 1, 0, 37, 0, 2, 1, 27, 3), nrow = 3,
                           ncol = 4, byrow = TRUE))

# Naming columns to match the training dataset
names(new.d) <- names(datT)[-1]

# Predicting survival for new data using the final pruned tree (DTF)
pred <- predict(DTF, newdata = new.d)
pred
```

```
##         1         2         3
## 0.1181818 0.9426752 0.1181818
```

1. **Step 3**: Choosing the size of the tree involves initially constructing a tree with a minimum split requirement, pruning the tree based on cross-validation, and finally creating the pruned tree using a chosen complexity parameter. It also includes evaluating the model by predicting survival for existing data and calculating the prediction error.

2. **Step 4**: Predicting survival for new data by applying the final pruned decision tree (DTF) to a set of new values.

This creates a decision tree model using the `rpart` function, where `target_variable` is the variable to predict, and `your_data` is your dataset.

Decision Trees are powerful and widely used in various fields due to their simplicity, interpretability, and effectiveness in solving complex problems. However, their performance can be improved by ensemble techniques like Random Forests or by using techniques to tackle overfitting like pruning or using a minimum number of samples per leaf node.