Decision Trees and Random Forests

DSC365: Introduction to Data Science

2026-10-27

Outline

- Tree-based machine learning methods
- Example Classification Scenario
- Decision Trees: Classification
- Evaluation
- Decision Trees for Regression
- Comparison with other methods
- Random Forests
- Example: Identifying Important Variables
- Summary

Announcements

Lab 5 (Linear Models)

- Due Tonight at 11:59 pm in Blueline

Quiz 3: in class on Thursday (October)

- Covers: Prediction/KNN/Tree-Based Methods
- If we don’t get to it in class today, it won’t be on the quiz, but you might need it for the lab.

Tree-based machine learning methods

Tree-Based Methods

- Can be used for both regression and classification
  - *Regression* models $\longrightarrow$ *quantitative* response variable
  - *Classification* models $\longrightarrow$ *categorical* response
- These involve stratifying or segmenting the predictor space into a number of simple regions

Example Classification Scenario

Example: Marijuana legalization

The General Social Survey is a wide-ranging survey conducted biannually to measure cultural shifts in American society. We can use the GSS to get an idea of how popular opinion has changed.

Goal: Use Age to predict people's opinion of marijuana legalization.

```
    Don't know          Legal      No answer Not applicable      Not legal 
           438           3185             10           3065           2725 
```

Let's Clean Our Data! Yay!

Data: GSS2016.csv

Data cleaning: `decision-trees-random-forests.R`

- Let's only look at one year, say 2016, and remove "Not applicable from our response"
- Want just two groups for responses: Legal and Not legal
- Change variables to proper type

```
GSS <- GSS %>%
  filter(YEAR==2016) %>%
  filter(GRASS != 'Not applicable')
GSS <- GSS %>%
  mutate(LEGAL = ifelse(GRASS=='Legal', 'Legal', 'Not legal'))
GSS$AGE <- as.numeric(GSS$AGE)
```

Testing data v. Training data

```
1 set.seed(4)
2 test_id <- sample(1:nrow(GSS), size=round(0.2*nrow(GSS)))
3 TEST <- GSS[test_id,]
4 TRAIN <- GSS[-test_id,]
```

How many people in the training data set support marijuana legalization?

```
1 TRAIN %>% group_by(LEGAL) %>% summarize(n=n())
```

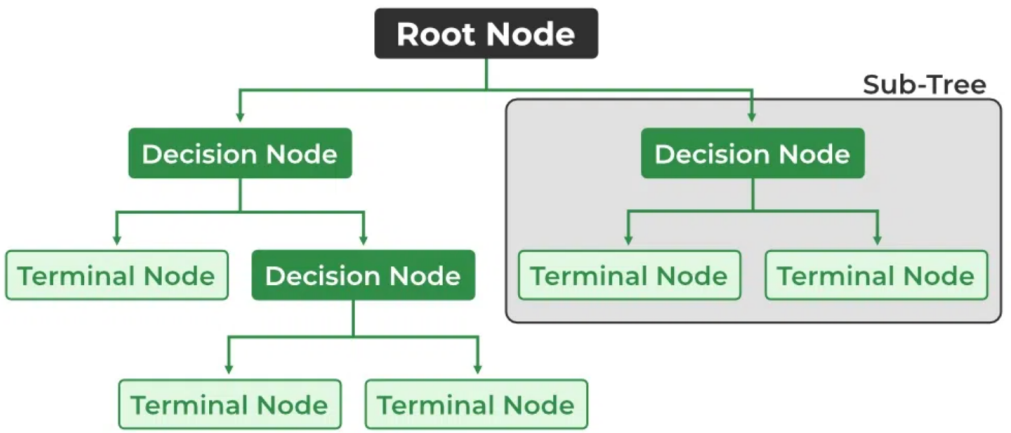
```
# A tibble: 2 × 2
  LEGAL         n
  <chr>     <int>
1 Legal       911
2 Not legal   654
```

Decision Trees: Classification

What is a decision tree

Decision trees: A tree-like model of decisions and their possible consequences

- Has flowchart-like structure in which each…



How Decision Trees classify

How to decide to split?

- As the number of predictors increases, so does the number of possible trees.
  - Optimal decision trees do not exist (Hyafil & Rivest, 1976)
  - So several competing methods for building decision trees

How Decision Trees classify (cont)

Recursive Partitioning: Split infinite # of times until reach stopping criterion.

Split: Gini Index (default):

$$G = \sum_{k=1}^{K} \hat{p}_{mk}(1 - \hat{p}_{mk})$$

where $\hat{p}_{mk}$ is the proportion of training observations in m^{th} region from k^{th} class.

Stop: Complexity parameter

Fitting A Decision Tree (Classification)

```r
#install.packages('rpart')
library(rpart)
rpart(LEGAL~AGE, data=TRAIN, na.action = na.pass)
```

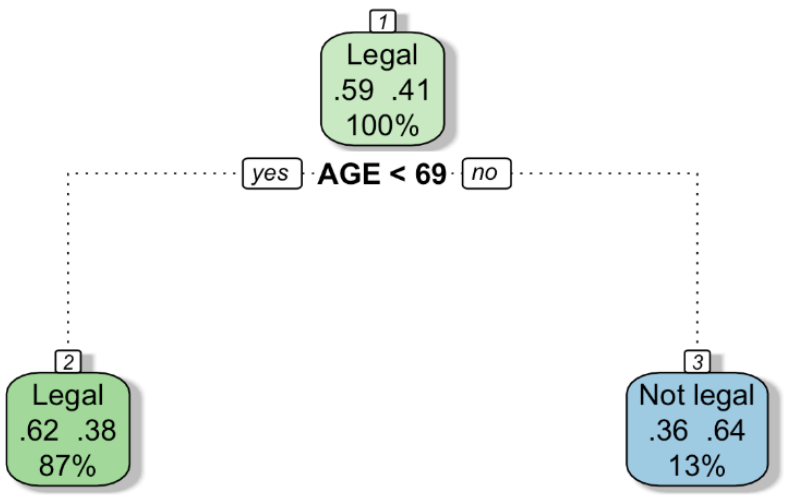
```
n= 1565

node), split, n, loss, yval, (yprob)
      * denotes terminal node

1) root 1565 654 Legal (0.5821086 0.4178914)
  2) AGE< 68.5 1350 521 Legal (0.6140741 0.3859259) *
  3) AGE>=68.5 215  82 Not legal (0.3813953 0.6186047) *
```

Visualizing a Decision Tree (Classification)

```
1 #install.packages("rattle")
2 library(rattle)
3 tree <- rpart(LEGAL~AGE, data=TRAIN, na.action = na.pass)
4 fancyRpartPlot(tree)
```



Evaluation

Evaluating a decision tree

- Confusion Matrix
- Classification Accuracy

Evaluating a decision tree: Confusion Matrix

```
TRAIN <- TRAIN %>%
  mutate(Legal_Tree = predict(tree, type='class'))

confusion_train <- tally(Legal_Tree~LEGAL, data=TRAIN)
confusion_train
```

```
           LEGAL
Legal_Tree  Legal Not legal
  Legal       829       521
  Not legal    82       133
```

```
TEST <- TEST %>%
  mutate(Legal_Tree = predict(tree, type='class', newdata = TEST))

confusion_test <- tally(Legal_Tree~LEGAL, data=TEST)
confusion_test
```

```
           LEGAL
Legal_Tree  Legal Not legal
  Legal       203       145
  Not legal    12        31
```

Evaluating a decision tree: Classification Accuracy

Training Accuracy:

```
1 sum(diag(confusion_train))/nrow(TRAIN)
```

```
[1] 0.6146965
```

Testing Accuracy:

```
1 sum(diag(confusion_test))/nrow(TEST)
```

```
[1] 0.5984655
```

Tuning Parameter: Complexity parameter

Using recursive partitioning can overfit the training data - Prune Tree

```
printcp(tree)
```

```
Classification tree:
rpart(formula = LEGAL ~ AGE, data = TRAIN, na.action = na.pass)

Variables actually used in tree construction:
[1] AGE

Root node error: 654/1565 = 0.41789

n= 1565

        CP nsplit rel error  xerror     xstd
1 0.077982      0   1.00000 1.00000 0.029834
2 0.010000      1   0.92202 0.94037 0.029544
```

- Changing complexity parameter:

```
rpart(LEGAL~AGE, data=TRAIN, na.action = na.pass,
      control = rpart.control(cp = 0.05))
```

Decision Trees for Regression

Decision Trees (Regression)

How to decide to split?

- Find regions (R_j) that minimizes the residual sum of squares

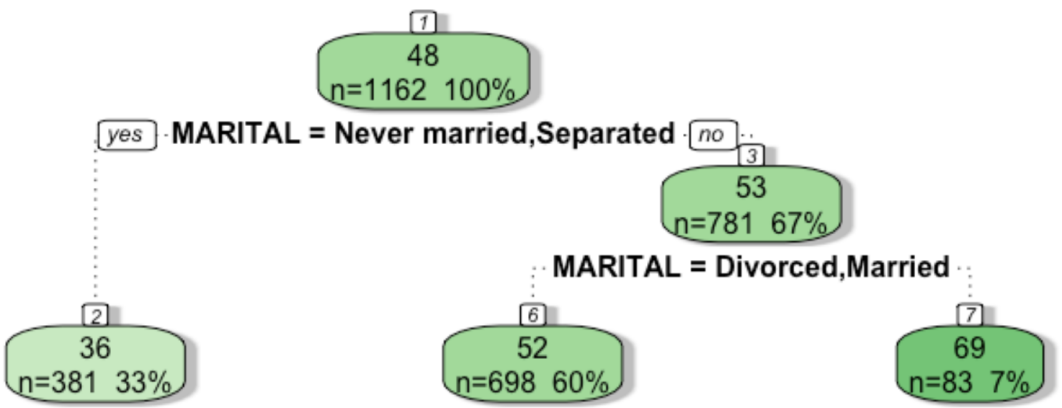
$$RSS = \sum_{j=1}^{J} \sum_{i \in R_j} (y_i - \hat{y}_{R_j})^2$$

  - Stops when you reach some criterion (cp)

Fitting A Decision Tree (Regression)

Let's suppose we want to use people's political view (POLVIEWS) and marital status (MARITAL) to estimate people's age.

```
1 tree2 <- rpart(AGE ~ POLVIEWS + MARITAL, data=TRAIN)
2 fancyRpartPlot(tree2)
```



Prediction using Decision Tree

We can still use the predict function to predict our regression decision tree outputs.

```
1 TEST <- TEST %>% filter(MARITAL != "No answer")
2
3 predict(tree2, TEST , method = "anova") %>% head()
```

```
       1        2        3        4        5        6
51.93004 51.93004 51.93004 51.93004 35.25934 51.93004
```

Caution: Can only predict using categorical variables located in the Training Set

Comparison with other methods

Trees Versus Linear Models

- If the relationship between the features and the response is well approximated by a linear model:
  - Linear regression

- If instead there is a highly nonlinear and complex relationship between the features and the response as indicated by model:
  - Decision trees

Advantages and Disadvantages of Decision Trees

Advantages

Disadvantages

Random Forests

Random Forests

A random forest is collection of decision trees that are aggregated by majority rule

When to use random forest:

Building a Random Forest

In building a random forest, each time a split in a tree is considered, a random sample of m predictors is chosen as split candidates from the full set of p predictors.

Hence, at each split in the tree, the algorithm is not even allowed to consider a majority of the available predictors. Why is this a good thing?

Example: Identifying Important Variables

Fit random forest

Goal: Which variables are most important for predicting views on marijuana legalization?

```
library(randomForest)

forest_grass <- randomForest(as.factor(LEGAL)~NEWSFROM+HAPPY+RELIG+COURTS+
                             ENERGY+EDUC+ENVIR+POLVIEWS+PARTYID+REGION+
                             INCOME+SEX+DEGREE+AGE+MARITAL+BALLOT,data=TRAI
                             na.action = na.omit, ntree=201, mtry=4)

forest_grass
```

```
Call:
 randomForest(formula = as.factor(LEGAL) ~ NEWSFROM + HAPPY +      RELIG +
COURTS + ENERGY + EDUC + ENVIR + POLVIEWS + PARTYID +      REGION + INCOME +
SEX + DEGREE + AGE + MARITAL + BALLOT,      data = TRAIN, ntree = 201, mtry =
4, na.action = na.omit)
               Type of random forest: classification
                     Number of trees: 201
No. of variables tried at each split: 4

        OOB estimate of  error rate: 33.96%
Confusion matrix:
          Legal Not legal class.error
Legal       704       202   0.2229581
```

Test predictions

```
TEST <- TEST %>%
  mutate(Legal_RF = predict(forest_grass, type='class',
                            newdata = TEST))

TEST$Legal_RF[1:5]
```

```
       1         2         3         4         5
Not legal     Legal     Legal     Legal     Legal
Levels: Legal Not legal
```

```
confusion_test <- tally(Legal_RF~LEGAL, data=TEST)
sum(diag(confusion_test))/nrow(TEST)
```

```
[1] 0.5717949
```

Variable Importance

Since each tree in a random forest uses a different set of variables, we want to keep track of which variables seem to be the most consistently influential.

```
randomForest::importance(forest_grass) %>% as.data.frame() %>%
  rownames_to_column() %>% arrange(desc(MeanDecreaseGini))
```

```
    rowname MeanDecreaseGini
1       AGE        128.51054
2    REGION         68.41636
3  POLVIEWS         65.99878
4    INCOME         52.25881
5    DEGREE         48.04504
6    COURTS         45.31095
7   MARITAL         44.85723
8     RELIG         44.83231
9   PARTYID         41.61337
10   ENERGY         36.68400
11    HAPPY         35.53879
12    ENVIR         34.43033
13     EDUC         29.67761
```

Summary

Modeling Data

- **Linear model**
  - The basic model for *regression* only
  - easy interpretation, but strict assumptions
  - Use cross-validation (CV) to ensure quality of model
- **KNN:**
  - Can be applied to *regression* and *classification*
  - fast, no assumptions, can add data anytime.
  - large number of variables can lower quality
  - use CV to decide k
- **Decision tree:**
  - Can be applied to *regression* and *classification.*
  - good data visualization
  - high variance
- **Random Forest:**
  - a collection of tree models; can be applied to *regression*[1] and *classification.*
  - Hard to interpret, but can output variable importance
    - Can be useful for *variable selection*
  - outperforms linear model when variables are no linearly independent

1. This is uncommon, though