PSS718 - Data Mining

Lecture 4 - Decision Trees

Asst.Prof.Dr. Burkay Genç

Hacettepe University

November 13, 2016

So far...

- Cluster Analysis : Which observations are alike?
- Association Rules : Does X imply Y?

Next...

- Decision Trees : What determines the target variable?

What is it?

- A traditional building block of data mining
- First developed in 1980s
- Have been the most widely deployed machine-learning based data mining model builder
  - Because it is simple
  - Easy to view
  - Easy to understand
  - Easy to explain
- Do not always deliver the best performance
- Can represent both classification and regression models

What is a tree?

- Used in many different fields, such as medicine, logic, problem solving, and management science
- ... also a traditional computer science structure for organizing data
- Usually presented upside down

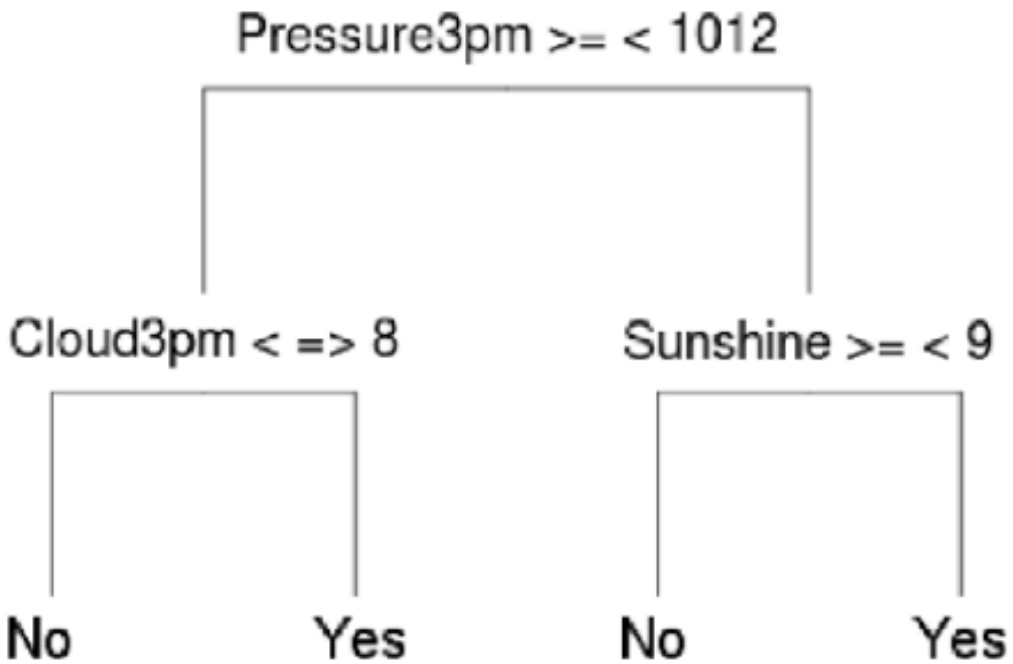
node a split in the tree

root a node with no parents

leaf a node with no children

What is a decision tree?

- Each non-leaf node corresponds to a question / test
- The leaf nodes are the decisions



Alternatives

- At each non-leaf node, we can test for any variable
- At each test, we can use many values of that variable (possibly infinite)
- How do we find the best test for each non-leaf node?
- What will be the depth of the tree?
- What will be the most accurate decision set?

Target variable

- Consider the weather data

```
> table(w$RainTomorrow)/nrow(w)

       No       Yes
0.8196721 0.1803279
```

- We want to find a variable and a value for it, such that the resulting partition has maximum homogeneity in terms of the target variable
- IOW, we want one partition with increased Yes ratio, and another with increased No ratio

Partition on Sunshine

- Consider testing for Sunshine < 9

```
> s <- split(w$RainTomorrow, w$Sunshine < 9)
> table(s[['TRUE']])/length(s[['TRUE']])

       No       Yes
0.7164179 0.2835821
> table(s[['FALSE']])/length(s[['FALSE']])

        No        Yes
0.95061728 0.04938272
```

- An obvious improvement
  - Especially for (Sunshine >= 9) $\rightarrow$ (RainTomorrow = 'No')
- Remember confidence in association mining?
- Can we do better? What is the best question to ask?

Keep partitioning

- Let's continue with Sunshine < 9
- What is next?
  - We need to partition each subset further to refine them
- Keep partitioning until
  - There are no more variables
  - There are no more observations
  - There is no gain left
- This process is called *recursive partitioning* (or divide and conquer)
- At each partition, we choose what looks like the best
  - This is called the *greedy* approach

Information Gain

- Rattle uses information gain measure for deciding between alternative splits
- Based on *entropy* from physics (i.e., the concept of the amount of disorder in a system)
- Disorder : How mixed our dataset is with respect to the values of the target variable
  - All observations have same target value -> entropy is 0
  - 50% Yes and 50% No -> entropy is 1
  - Everything else -> entropy is between 0 and 1

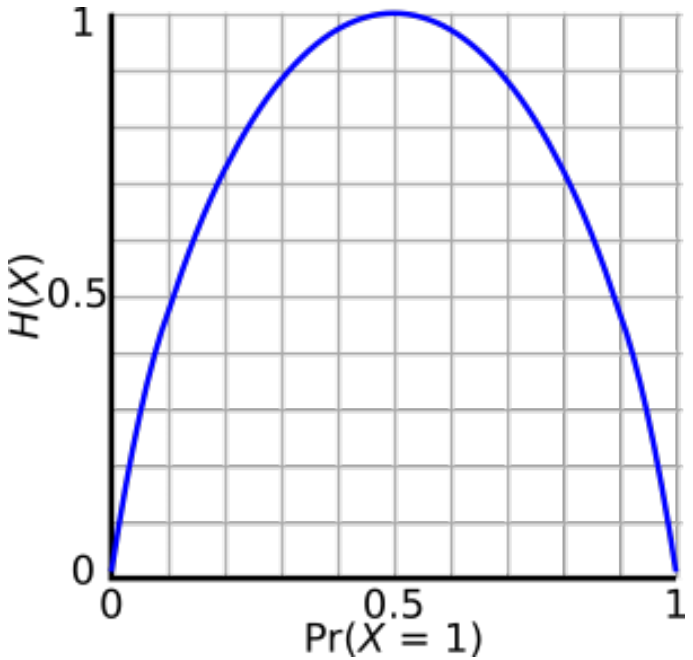
Information need

- Entropy 0 -> There is no need for further information for classification
- Entropy 1 -> There is maximum need for further information
- Entropy 0.5 -> There is some further information needed for full classification

Entropy (Information Need)

$$e(D) = -P(a)\log_2 P(a) - P(b)\log_2 P(b)$$

where a and b are two possible outcomes of the distribution, and $P(x)$ is the probability of outcome x.



Entropy

- Assume 100% a, and 0% b.

$$e(D) = -1.0 \log_2 1.0 - 0 \log_2 0, \qquad \lim_{x \to 0} (x \log_2 x) \to 0$$

$$e(D) = 0$$

- Assume 50% a, and 50% b.

$$e(D) = -0.5 \log_2 0.5 - 0.5 \log_2 0.5$$

$$e(D) = -\log_2 1/2 = -(-1) = 1$$

Entropy

- Each split results in two partitions D_1 and D_2
- $D = D_1 \cup D_2$
- Let E_1 be the entropy of D_1 and E_2 the entropy of D_2
- Compute the combined entropy of the split:

$$e(D, S) = \frac{|D_1|}{|D|} E_1 + \frac{|D_2|}{|D|} E_2$$

- Now compute the information gain due to the split:

$$gain(D, S) = e(D) - e(D, S)$$

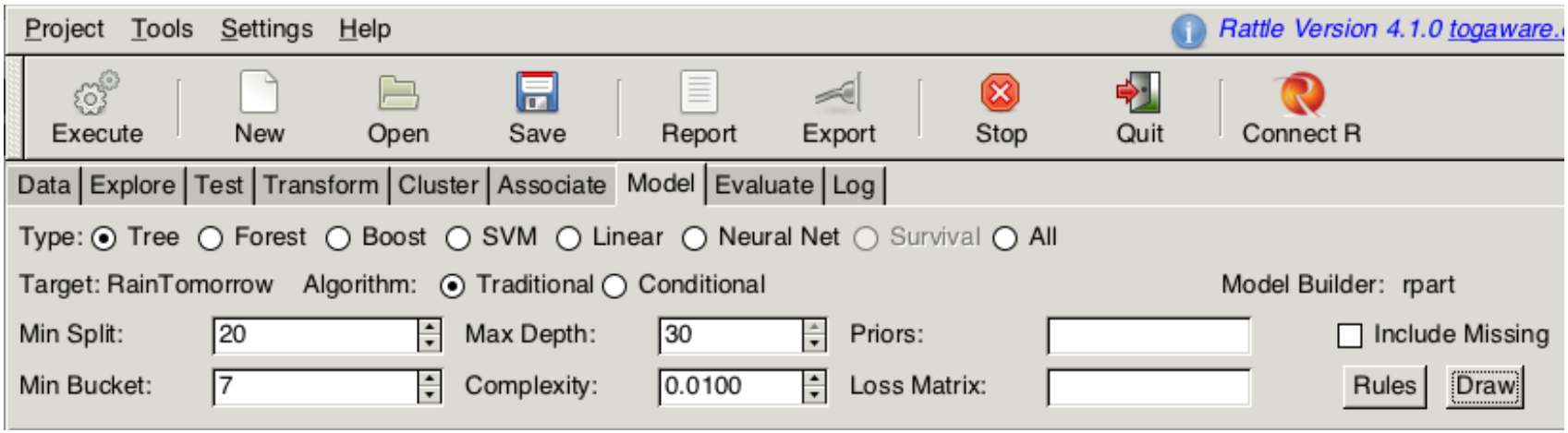
Algorithm Step

- At each step compute the gains for all possible splits
- Choose the split with the most gain
- Repeat

Warning

Continuous variables need to be discretized.

UI


Project Tools Settings Help
Rattle Version 4.1.0 togaware.
Execute
New
Open
Save
Report
Export
Stop
Quit
Connect R
Data Explore Test Transform Cluster Associate Model Evaluate Log
Type: Tree Forest Boost SVM Linear Neural Net Survival All
Target: RainTomorrow Algorithm: Traditional Conditional
Model Builder: rpart
Min Split: 20
Max Depth: 30
Priors:
Include Missing
Min Bucket: 7
Complexity: 0.0100
Loss Matrix:
Rules
Draw

The output

```
Summary of the Decision Tree model for Classification (built using 'rpart'):

n= 256

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

1) root 256 41 No (0.83984375 0.16015625)
  2) Pressure3pm>=1011.9 204 16 No (0.92156863 0.07843137)
    4) Cloud3pm< 7.5 195 10 No (0.94871795 0.05128205) *
    5) Cloud3pm>=7.5 9  3 Yes (0.33333333 0.66666667) *
  3) Pressure3pm< 1011.9 52 25 No (0.51923077 0.48076923)
    6) Sunshine>=8.85 25  5 No (0.80000000 0.20000000) *
    7) Sunshine< 8.85 27  7 Yes (0.25925926 0.74074074) *
```

- n=256 : number of observations
- node) : node ID
- split : How the split was made
- n : number of observations at that node
- loss : number of incorrectly classified observations
- yval : default classification of the node
- yprob : distribution of classes in the node

The output - examined

2) Pressure3pm>=1011.9 204 16 No (0.92156863 0.07843137)

- Node ID is 2
- Node will be split by Pressure3pm variable on value >= 1011.9
- There are 204 observations in this node
- 16 of these are falsely classified
- Default class of this node is "No"
  - So, there are 16 "Yes" observations
- The rate of "No" observations to all is 0.92
- The rate of "Yes" observations to all is 0.08

The model command

command

Classification tree:
rpart(formula = RainTomorrow ~ ., data = crs$dataset[crs$train, c(crs$input, crs$target)], method = "class", parms = list(split = "information"), control = rpart.control(usesurrogate = 0, maxsurrogate = 0))

Variables Used

```
Variables actually used in tree construction:
[1] Cloud3pm     Pressure3pm Sunshine
```

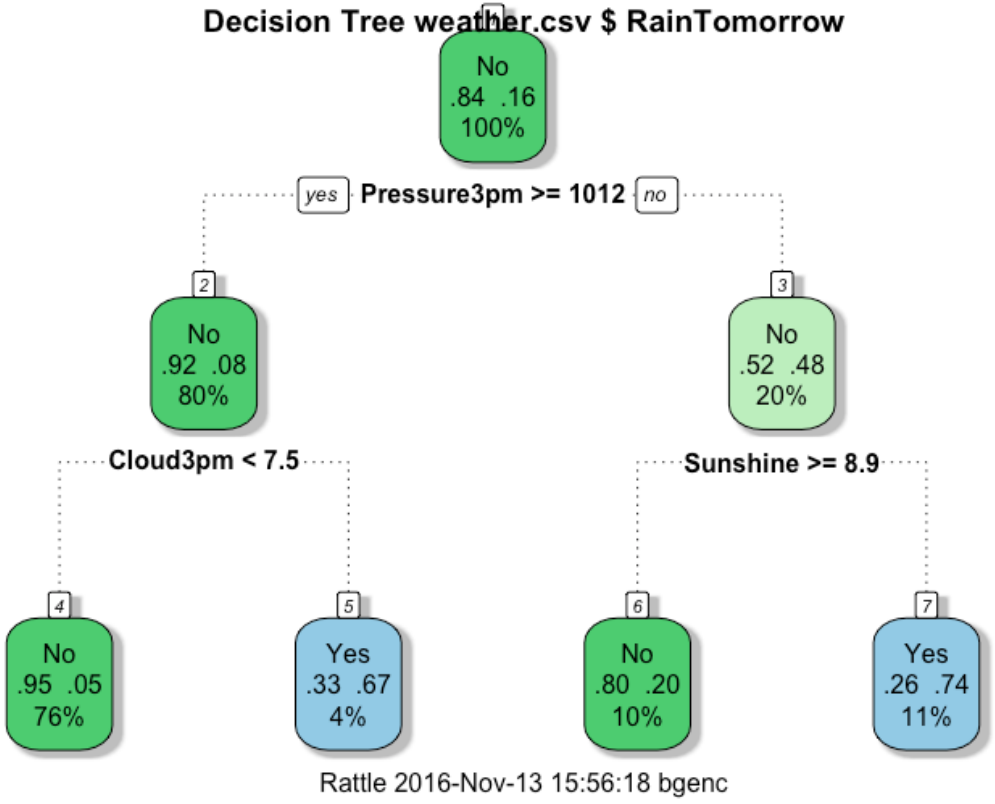
Performance Evaluation

```
Root node error: 41/256 = 0.16016

n= 256

        CP nsplit rel error  xerror    xstd
1 0.158537      0   1.00000 1.00000 0.14312
2 0.073171      2   0.68293 0.80488 0.13077
3 0.010000      3   0.60976 0.97561 0.14169

Time taken: 0.01 secs
```

- Baseline error is 0.16
- error is going down in each step
- CP is complexity
- xerror (cross-validation error) starts to increase at some point, which signals for stopping

Visualization

- Use the Rules button to get a textual representation of the rules
- Use the Draw button for a visual representation



Scoring

- Use the Evaluate tab for scoring and other ways of evaluating the model

Defaults

- Default parameter values are generally quite successful
- We still may need to tamper with them in exceptional cases
- Even more detail is provided by `?rpart`

Modeling method

Definition (method)

The `method=` argument indicates the type of model to be built and is dependent on the target variable.

- For categoric targets, generally classification models are used, hence use `method='class'`
- For numeric targets, use `method='anova'` for an analysis of variance based regression model

Splitting function

Definition (split)

The `split=` argument is used to choose between different splitting functions (measures). The argument appears within the `parms` argument of `rpart()`, which is built up as a named list.

- `split='information'` is the default in Rattle and uses the information gain approach as we described
- `split='gini'` is the default for R (in `rpart()`) and uses the Gini index of diversity.
- Results will usually be very similar

Minimum split

Definition (minsplit)

The `minsplit` argument specifies the minimum number of observations that must exist at a node in the tree before it is considered for splitting. A node is not considered for splitting if it has fewer than `minsplit` observations.

- The `minsplit` argument appears within the `control=` argument of `rpart()`.
- The default value of `minsplit` is 20.
- Leaf nodes can still be constructed that have fewer observations than the `minsplit`, as that is controlled by the `minbucket` argument.

Minimum bucket size

Definition (minbucket)

The `minbucket` argument is the minimum number of observations in any leaf node.

- Default value is 7, or about one-third of the default value of `minsplit`
- If either of these two arguments is specified but not the other, then the default of the unspecified one is taken to be a value such that this relationship holds (i.e., `minbucket` is one-third of `minsplit`)
- Choosing smaller bucket sizes generally leads to larger trees

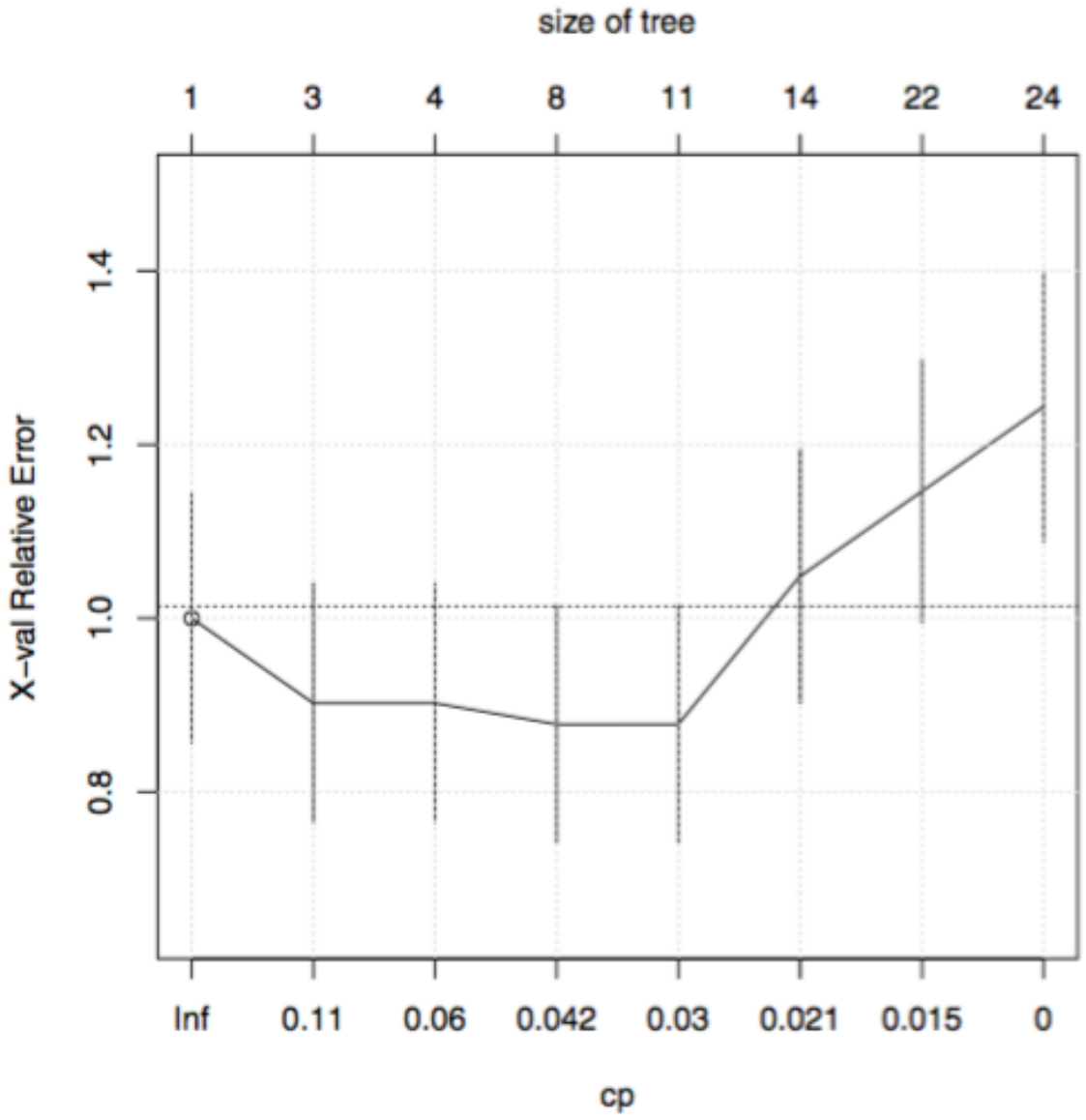
Complexity parameter

Definition (cp)

The complexity parameter is used to control the size of the decision tree and to select an optimal tree size.

- Controls the process of pruning a decision tree
- If a tree is not properly pruned, then it may overfit the training data
- Using cp governs the minimum "benefit" that must be gained at each split of the decision tree in order to make a split worthwhile
- Default is 0.01

Complexity parameter



Priors

Definition (prior)

Priors are used to override class proportions in the population.

- Priors are represented as a list of numbers that sum up to 1.
- The list must be of the same length as the number of classes.

Loss matrix

Definition (loss)

The loss matrix is used to weight different kinds of errors (or loss) differently.

- Often, one type of error is more significant than another type of error
- Consider fraud detection: false positives are fine, but false negatives are dangerous
- We assign weights on the following matrix of observed vs. predicted values:

Pre vs. Obs	False	True
False	TN	FN
True	FP	TP

- We provide values into rattle as a list: 0,FN,FP,0
- An example is the string of numbers 0, 10, 1, 0, which might be interpreted as saying that an actual 1 predicted as 0 (i.e., a false negative) is ten times more unwelcome than a false positive.