Data Science for Business
Lecture #6
Classification and Regression Trees

Prof. Alan L. Montgomery

The University of Hong Kong & Carnegie Mellon University, Tepper School of Business

email: alanmontgomery@cmu.edu

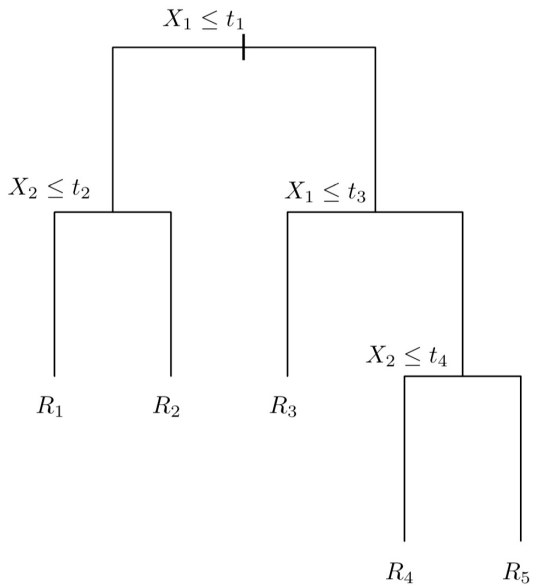
Overview of Decision Trees

Popular non-parametric model for classification and regression

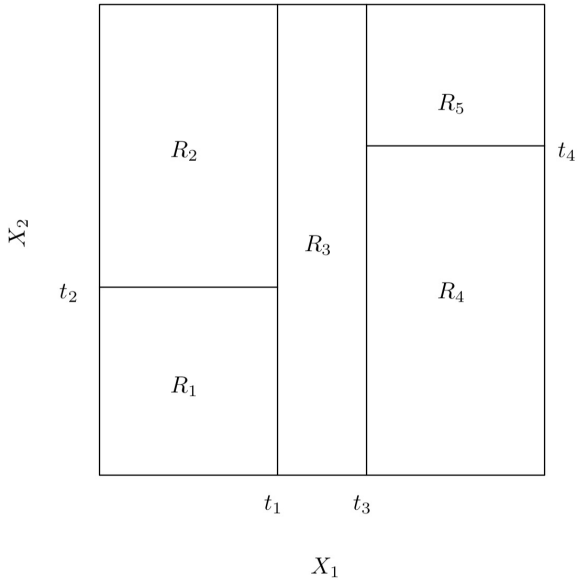
- Prediction model is a set of decision rules that are organized in a tree. The tree is constructed from training data using recursive partitions.
- Intuitive. The tree rules tend to be easy to communicate and interpret.
- Overfitting. Danger of overfitting is high. Avoid by cross-validating or using bagging or random forests.

What can decision trees represent

Rules

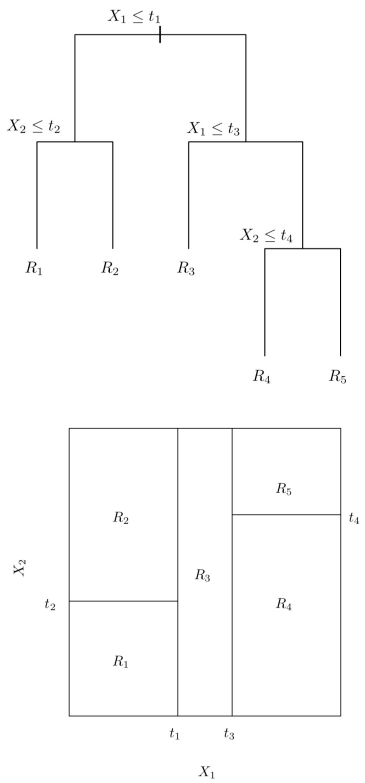


partitions

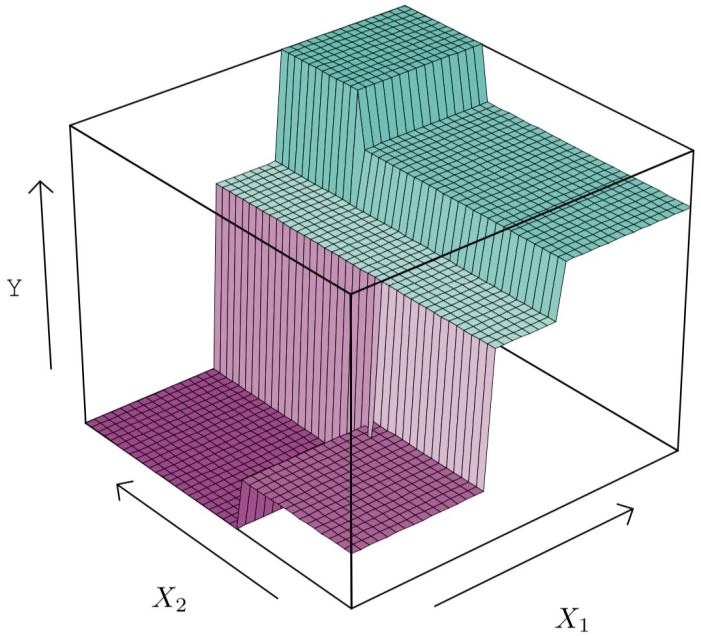


Source: An Introduction to Statistical Learning, http://ww-bcf.usc.edu/~gareth/ISL/

What can decision trees represent



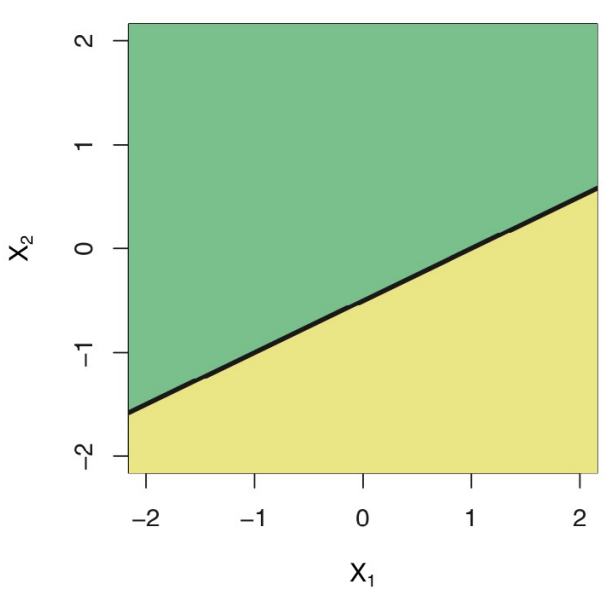
PERSPECTIVE PLOT



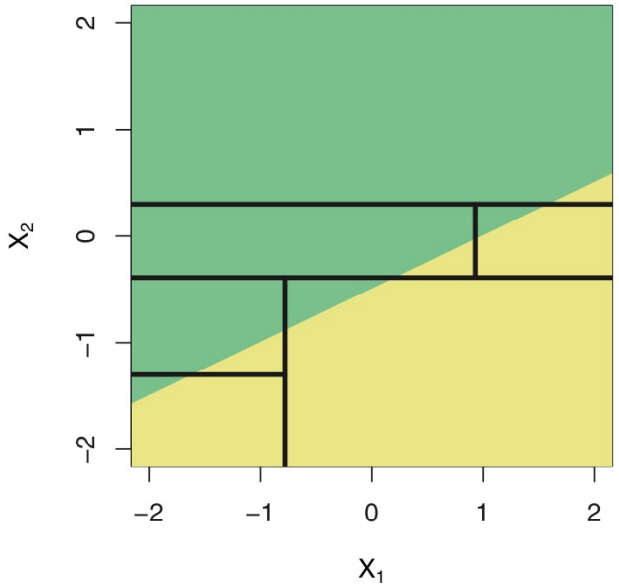
Source: An Introduction to Statistical Learning, http://ww-bcf.usc.edu/~gareth/ISL/

Trees are good at detecting interactions

True relationship

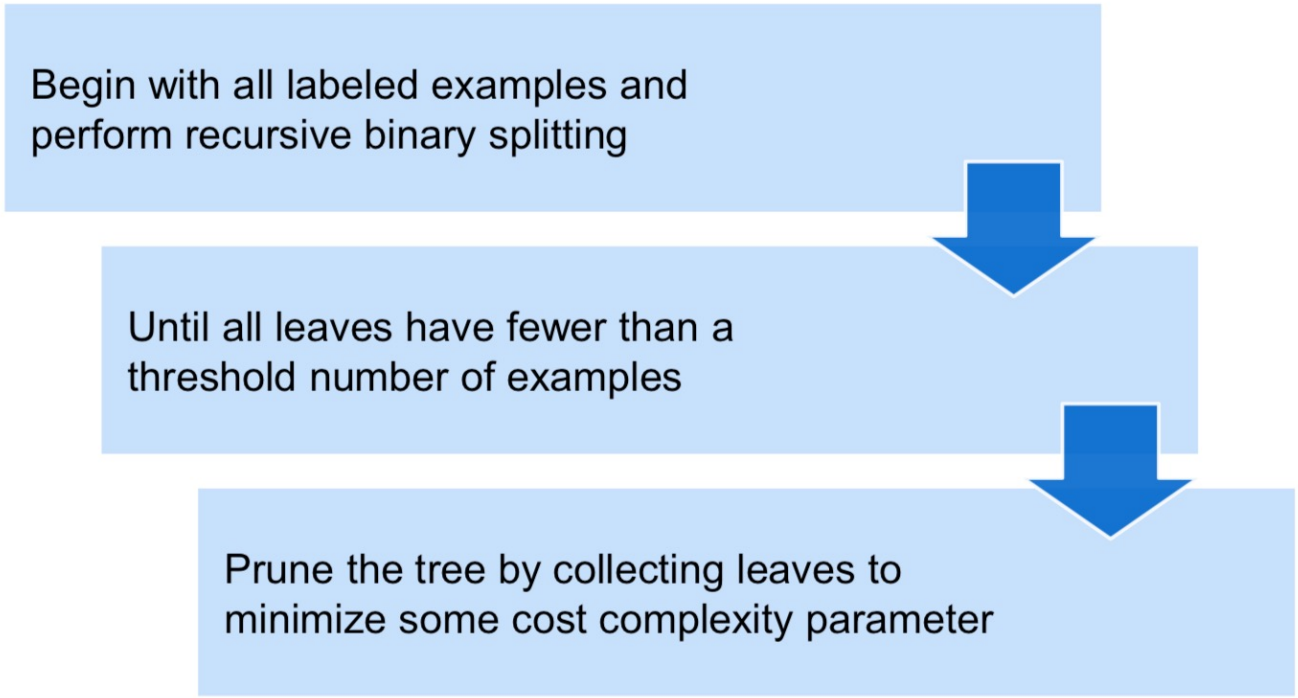


Tree approximation

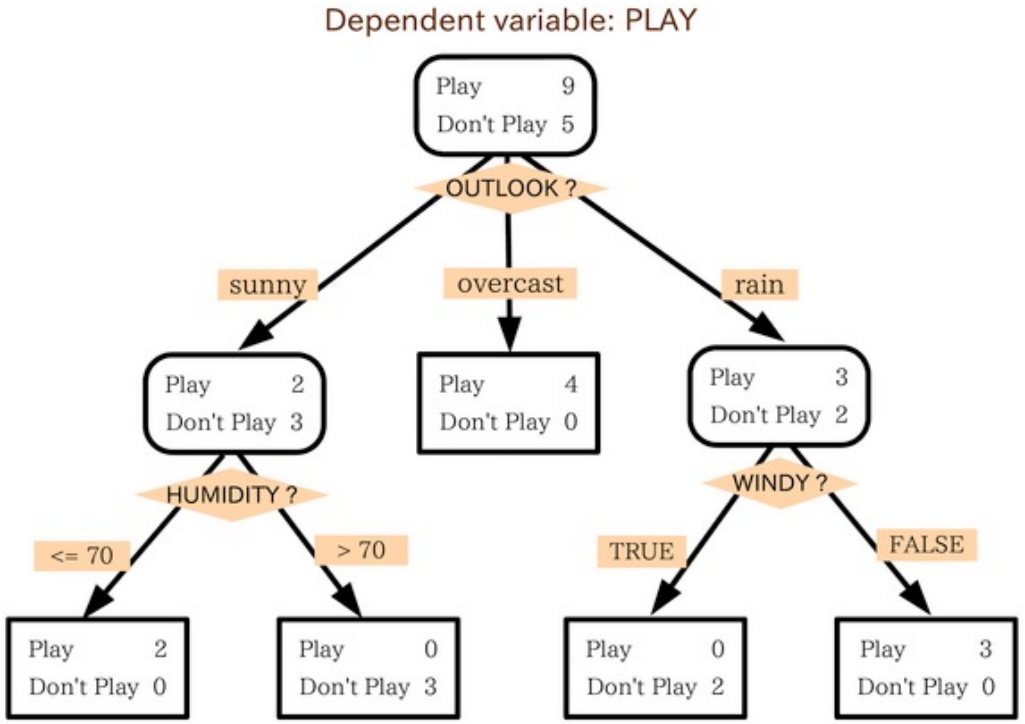


Source: An Introduction to Statistical Learning, http://ww-bcf.usc.edu/~gareth/ISL/

Underlying Method



How would we build this tree?



Start with the data

Day	Outlook	Temperature	Humidity	Wind	PlayTennis
d1	sunny	hot	high	weak	no
d2	sunny	hot	high	strong	no
d3	overcast	hot	high	weak	yes
d4	rain	mild	high	weak	yes
d5	rain	cool	normal	weak	yes
d6	rain	cool	normal	strong	no
d7	overcast	cool	normal	strong	yes
d8	sunny	mild	high	weak	no
d9	sunny	cool	normal	weak	yes
d10	rain	mild	normal	weak	yes
d11	sunny	mild	normal	strong	yes
d12	overcast	mild	high	strong	yes
d13	overcast	hot	normal	weak	yes
d14	rain	mild	high	strong	no

Find “split”, say *Outlook*

Original data: 9Y, 5N. If we use Outlook we have three splits: 2Y,3N 4Y,0N 3Y,2N

Day	Outlook	Temperature	Humidity	Wind	PlayTennis
d1	sunny	hot	high	weak	no
d2	sunny	hot	high	strong	no
d3	overcast	hot	high	weak	yes
d4	rain	mild	high	weak	yes
d5	rain	cool	normal	weak	yes
d6	rain	cool	normal	strong	no
d7	overcast	cool	normal	strong	yes
d8	sunny	mild	high	weak	no
d9	sunny	cool	normal	weak	yes
d10	rain	mild	normal	weak	yes
d11	sunny	mild	normal	strong	yes
d12	overcast	mild	high	strong	yes
d13	overcast	hot	normal	weak	yes
d14	rain	mild	high	strong	no

Consider all potential "splits"

Several alternative variables to split upon:

Outlook:	2Y, 3N	4Y, 0N	3Y, 2N
Temperature:	2Y, 2N	4Y, 2N	3Y, 1N
Humidity:	3Y, 4N	6Y, 1N	
Wind:	6Y, 2N	3Y, 3N	

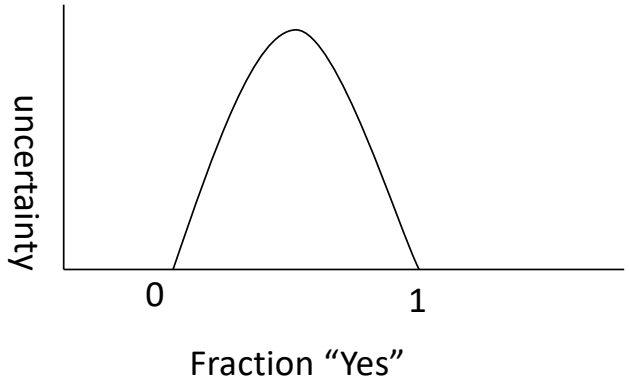
Which to choose?

Measures of Uncertainty

Measure uncertainty: 0 to some max value.

Function of ability to predict:

- Perfectly predictable: uncertainty=0
- No information: uncertainty=1 (or highest value).



p≡"Yes", q=1-p ≡"No"

Entropy:

$-p \log_2 p - q \log_2 q$

Gini:

$1-p^2-q^2$

Min:

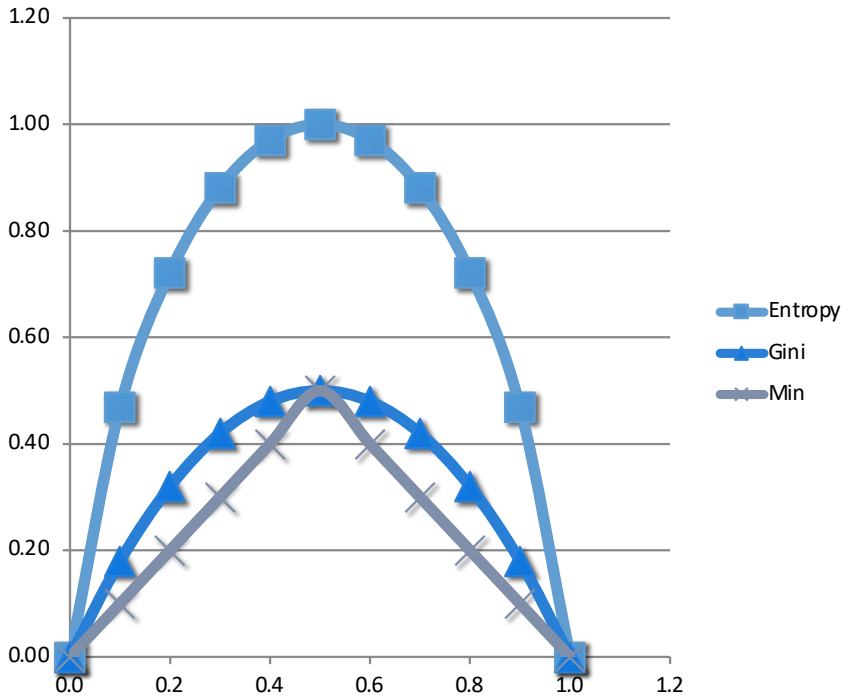
Min{p,q}

Discussion Exercise: Complete the spreadsheet

p	q	Entropy	Gini	Min
0.0	1.0			
0.1	0.9			
0.2	0.8			
0.3	0.7			
0.4	0.6			
0.5	0.5			
0.6	0.4			
0.7	0.3			
0.8	0.2			
0.9	0.1			
1.0	0.0			

Discussion Exercise Solution
Measures increase with uncertainty

p	q	Entropy	Gini	Min
0.0	1.0	#NUM!	0.00	0.00
0.1	0.9	0.47	0.18	0.10
0.2	0.8	0.72	0.32	0.20
0.3	0.7	0.88	0.42	0.30
0.4	0.6	0.97	0.48	0.40
0.5	0.5	1.00	0.50	0.50
0.6	0.4	0.97	0.48	0.40
0.7	0.3	0.88	0.42	0.30
0.8	0.2	0.72	0.32	0.20
0.9	0.1	0.47	0.18	0.10
1.0	0.0	#NUM!	0.00	0.00



Uncertainty of our dataset

Let's use Gini:

$1-p^2-q^2$

Original set has uncertainty value:

$1 - (9/14)^2 - (5/14)^2 = 0.459$

Uncertainty after a split?

Split on *Outlook*: three problems: 2Y,3N 4Y,0N 3Y,2N

$1 - (2/5)^2 - (3/5)^2 = 0.48$

$1 - 1^2 - 0^2 = 0$

$1 - (3/5)^2 - (2/5)^2 = 0.48$

Total uncertainty of three combined:

$(5/14)*0.48 + (4/14)*0 + (5/14)*0.48 = 0.342$

Total gain = $0.459 - 0.342 = 0.117$

Choosing Splitting Variable

Find split with biggest gain in information (decrease in uncertainty)

Temperature: 2Y, 2N 4Y, 2N 3Y,1N

- 2Y, 2N: 0.5
- 4Y, 2N: 0.444
- 3Y, 1N: 0.375 Total: 0.440 Gain: 0.019

Humidity: 3Y,4N 6Y, 1N

- 3Y, 4N: 0.490
- 6Y, 1N: 0.245 Total: 0.367 Gain: 0.092

Wind: 6Y,2N 3Y,3N

- Do yourself!

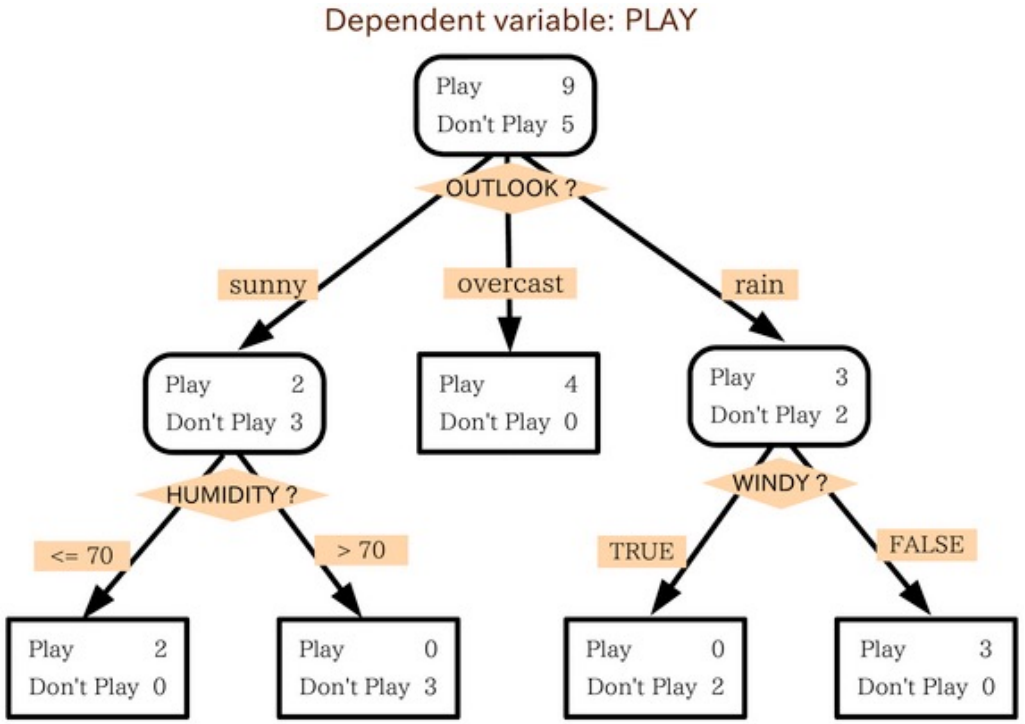
Question: What variable to split upon?

Answer: Which has the highest Gain?

Now what?
Split on variable to create three new problems

Day	Outlook	Temperature	Humidity	Wind	PlayTennis
d3	overcast	hot	high	weak	yes
d7	overcast	cool	normal	strong	yes
d12	overcast	mild	high	strong	yes
d13	overcast	hot	normal	weak	yes
d4	rain	mild	high	weak	yes
d5	rain	cool	normal	weak	yes
d6	rain	cool	normal	strong	no
d10	rain	mild	normal	weak	yes
d14	rain	mild	high	strong	no
d1	sunny	hot	high	weak	no
d2	sunny	hot	high	strong	no
d8	sunny	mild	high	weak	no
d9	sunny	cool	normal	weak	yes
d11	sunny	mild	normal	strong	yes

If we continue to break apart each subtree our final solution becomes



When to stop growing our tree?

Stop when

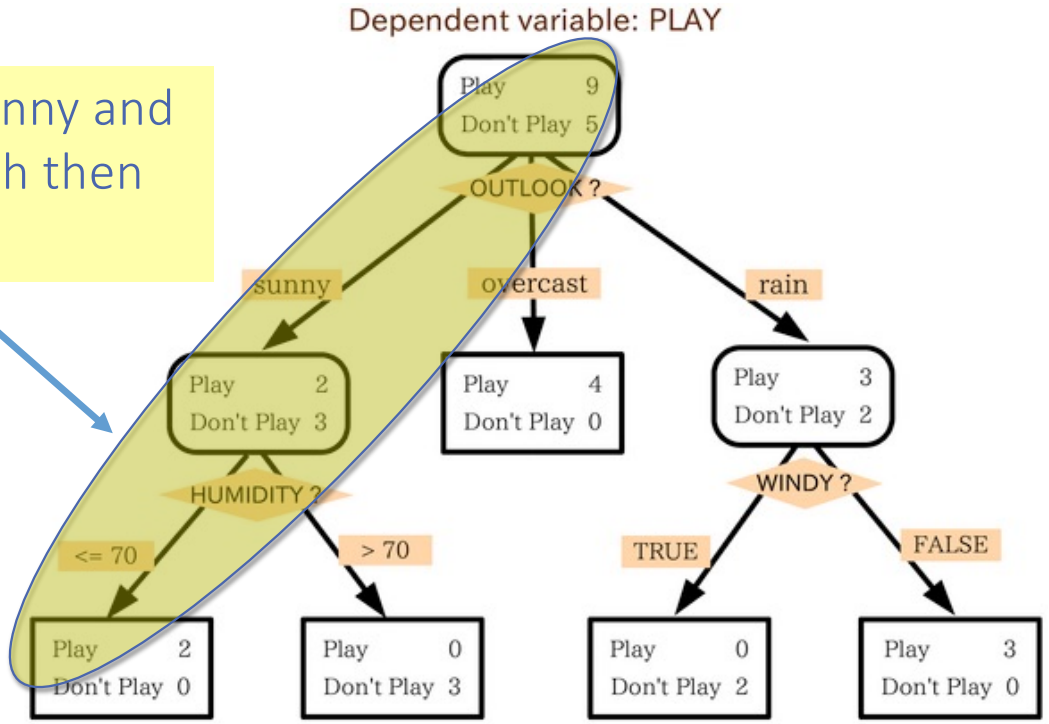
- Data is pure (all one category)
- No split available
- Limits on number of data points or depth from root

Place stop: “Leaf”

- Decision at leaf is most common value at that point

Alternative representation of Decision Tree is to define “Rules” that correspond with each leaf

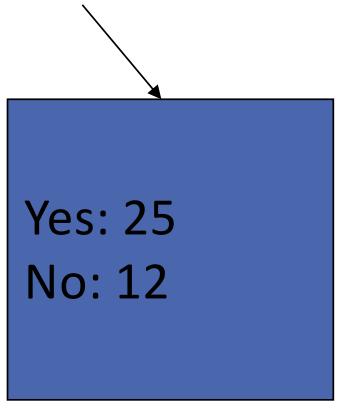
If outlook=sunny and humidity=high then tennis=NO



One caution is that these rules do not imply causation, they only indicate correlations.

Errors or Randomness in Leaves

Generally, data will not result in "pure" leaves: Data in leaves will still have mix of outcomes.



Decision: Yes

Accuracy: 25/37

Relevance: 37/overall data points

Two measures of the quality of a rule:

- *Relevance*: what fraction of the time would such a rule be invoked?
- *Accuracy*: what is the estimated likelihood of a correct value?

Which is better?

(10% relevant, 90% accurate)

Or (25% relevant, 80% accurate)

Continuous attributes

What if we had something like degrees?

Day	Outlook	Temperature	Humidity	Wind	degrees	PlayTennis
d9	sunny	cool	normal	weak	44	yes
d7	overcast	cool	normal	strong	52	yes
d5	rain	cool	normal	weak	61	yes
d6	rain	cool	normal	strong	68	no
d12	overcast	mild	high	strong	70	yes
d4	rain	mild	high	weak	73	yes
d8	sunny	mild	high	weak	74	no
d10	rain	mild	normal	weak	75	yes
d11	sunny	mild	normal	strong	77	yes
d14	rain	mild	high	strong	78	no
d1	sunny	hot	high	weak	83	no
d3	overcast	hot	high	weak	85	yes
d2	sunny	hot	high	strong	87	no
d13	overcast	hot	normal	weak	92	yes

- "Automatically" find best split
  - Split on (<=44, >44)
  - Split on (<=52, >52)
  - Continue trying every split
  - Choose best split using highest gain
  - Computationally intensive, but still fast for modern computers
- "Bucket" variables into categories
  - You could choose cold, warm, hot, very hot as categories
  - But beware of "bushiness" which occurs when some attributes have lots of values like postal codes, customer numbers or names. Might want to omit.

Missing values

Several approaches

- Treat “missing” as a category
- Replace “missing” with an average value
- Impute a value for “missing” using the other information

Another issue with Trees is whether it is a Prediction/Regression Problem or Classification Problem

Regression predicts values

Residual Sum of Squares

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

Classification predicts categories

Gini index

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

Cross entropy

$$D = -\sum_{k=1}^{K} \hat{p}_{mk} \log \hat{p}_{mk}$$

We can change our objective function depending upon whether our output is continuous or discrete. The duality of the algorithm is the reason that it is often referred to as "**C**lassification **a**nd **R**egression **T**rees" or simply the acronym "CART".

Avoiding Overfitting of Trees

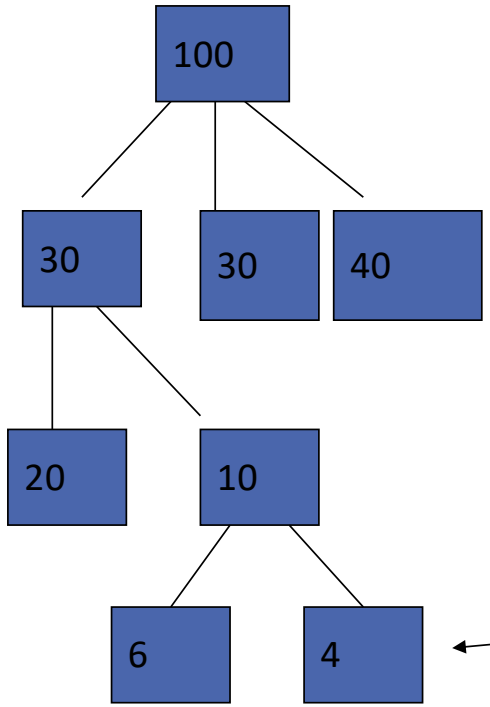
Quality of trees and making sure that they generalize

Overfitting data

Common problem in data mining:

> A model overfits the data if the model if it fits the data significantly better than it fits the overall distribution

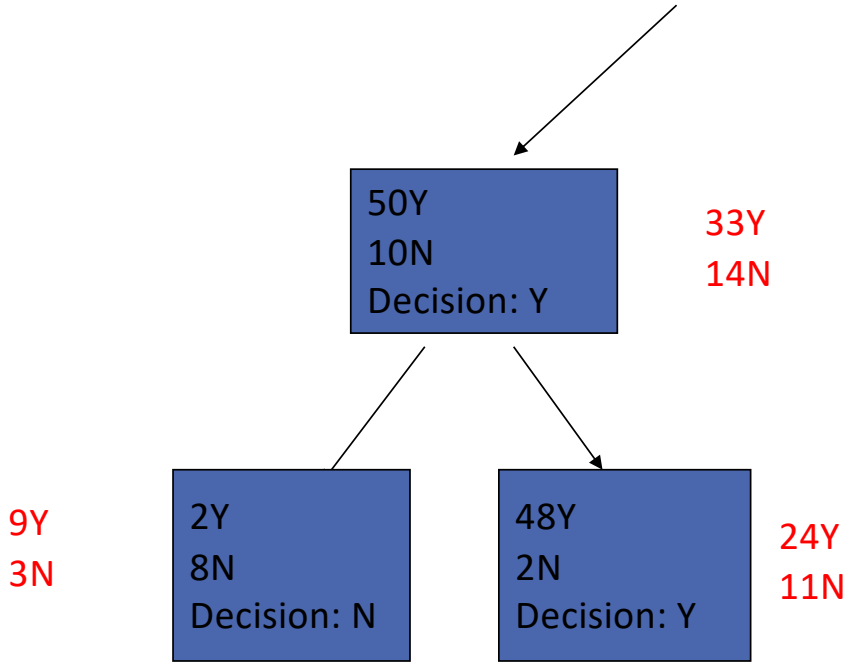
Overfitting and Decision Trees



As tree grows, the number of instances in each node decreases, making decisions less reliable

Even if 4Y 0N, not much confidence

Avoiding Overfitting: Example



Training Data
Validation Data
(not used to create tree)

More accurate on validation data if split not done

Summarizing Methods to Avoid Overfitting

1. Limits on number of data points in nodes before forcing it to be a leaf
2. Limits on depth
3. Statistical tests on significance of differences
4. Use of validation data: allow overfitting, then prune away based on new data (data not used to create tree)

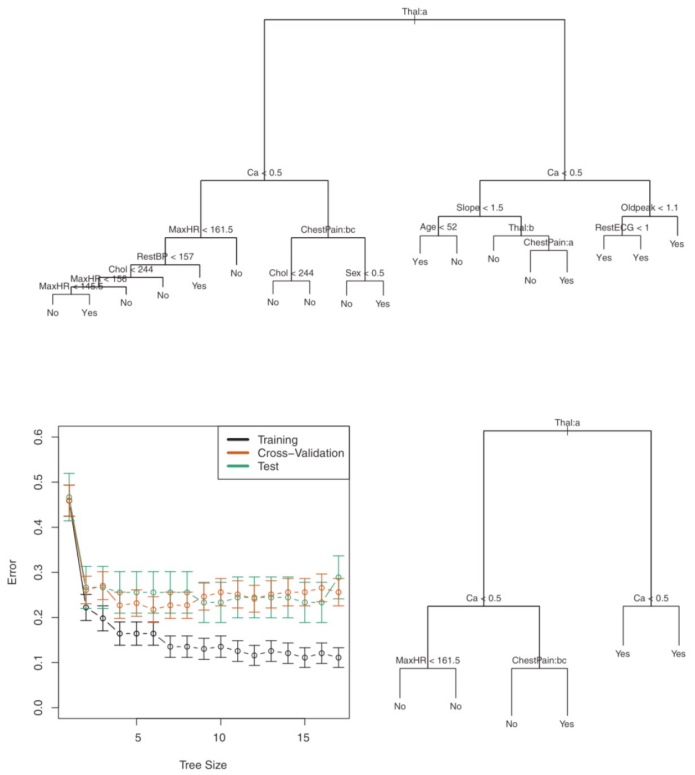


Illustration of overfit tree and pruned tree

Validating and Selecting Trees

Sources:

Brian Mac Namee (Dublin Institute of Technology)

Foster Provost (New York University)

Evaluating

Question:

- How do we know if we have a good model?

Answer:

- If it makes sense and if it predicts well for “new” examples then we usually accept it is a good model.

Problems:

- We may not know which model to use.
- More complex models with many parameters tend to fit better

Simple example of overfitting

Class	r1
leave	A
leave	B
leave	A
leave	B
stay	A
stay	B
stay	B
stay	A

This table has one attribute with its value assigned randomly. Unless we're very unlucky, we shouldn't see a relationship between the random value and the class.

Class	r1	r2	r3
leave	A	B	B
leave	B	A	A
leave	A	A	A
leave	B	A	B
stay	A	B	A
stay	B	B	A
stay	B	B	B
stay	A	A	A

With more random attributes, we can start to mine "useful" patterns:

"if r2 = A then leave else stay" has accuracy = 75% !

(on the these data!)

(hold-out accuracy would be around 50% -- random guessing)

With 9 random variables…

```
J48 unpruned tree
------------------

r9 = B
|   r2 = B
|   |   r1 = A: leave (1.0)
|   |   r1 = B: stay (1.0)
|   r2 = A: leave (3.0)
r9 = A: stay (3.0)

Number of Leaves  :   4

Size of the tree :   7


Time taken to build model: 0 seconds

=== Evaluation on training set ===

Correctly Classified Instances           8               100       %
```

Class	r1	r2	r3	r4	r5	r6	r7	r8	r9
leave	A	B	B	A	B	A	B	A	B
leave	B	A	A	B	B	B	B	B	B
leave	A	A	A	A	A	A	B	B	B
leave	B	A	B	A	A	A	A	B	B
stay	A	B	A	A	A	A	A	B	A
stay	B	B	A	A	B	B	B	B	B
stay	B	B	B	A	B	A	B	A	A
stay	A	A	A	B	B	B	A	B	A

r9 just misses 1

Would more random variables give better trees?

Class	r1	r2	r3	r4	r5	r6	r7	r8	r9	r10	r11	r12	r13	r14	t1	t2
leave	A	B	B	A	B	A	B	A	B	B	B	A	B	A	A	A
leave	B	A	A	B	B	B	B	B	B	A	A	A	A	B	A	A
leave	A	A	A	A	A	A	B	B	B	B	B	A	A	A	B	B
leave	B	A	B	A	A	A	A	B	B	A	B	A	B	B	A	A
stay	A	B	A	A	A	A	A	B	A	A	A	B	B	B	A	B
stay	B	B	A	A	B	B	B	B	B	A	B	B	B	B	B	A
stay	B	B	B	A	B	A	B	A	A	A	B	B	B	B	A	B
stay	A	A	A	B	B	B	A	B	A	A	A	A	B	A	B	A

J48 unpruned tree
---

r9 = B
| r12 = A: leave (4.0)
| r12 = B: stay (1.0)
r9 = A: stay (3.0)

Correctly Classified Instances 8 100%
(Hold-out accuracy of this tree would be around 50%)

Even good "true" patterns will be obscured eventually

Class	r1	r2	r3	r4	r5	r6	r7	r8	r9	r10	r11	r12	r13	r14	t1	t2
leave	A	B	B	A	B	A	B	A	B	B	B	A	B	A	A	A
leave	B	A	A	B	B	B	B	B	B	A	A	A	A	B	A	A
leave	A	A	A	A	A	A	B	B	B	B	B	A	A	A	B	B
leave	B	A	B	A	A	A	A	B	B	A	B	A	B	B	A	A
stay	A	B	A	A	A	A	A	B	A	A	A	B	B	B	A	B
stay	B	B	A	A	B	B	B	B	B	A	B	B	B	B	B	A
stay	B	B	B	A	B	A	B	A	A	A	B	B	B	B	A	B
stay	A	A	A	B	B	B	A	B	A	A	A	A	B	A	B	A

J48 unpruned tree

---

r9 = B
| r12 = A: leave (4.0)
| r12 = B: stay (1.0)
r9 = A: stay (3.0)

Together, t1 and t2 predict the class perfectly
(if they're both the same then "leave")
But they're not found by the tree inducer
(here, because neither t variable
in isolation reduces entropy
as much as r9 does accidentally)

Correctly Classified Instances 8 100%
(Hold-out accuracy of this tree would be around 50%.
Hold-out accuracy of the "true" tree would be 100%)

The answer:
Run an experiment

Ideally we would estimate each model and run an experiment to test which model works best in the experimental treatment.

Experiments give us a systematic way of evaluating hypothesis. In our case the "treatment" is the model, and we select the one that works best based upon some predetermined metric – like fit or likelihood or profits.

Why run an experiment and not just choose the model that fits the best?

Estimation with Training Data

The accuracy/error estimates on the training data are not good indicators of performance on future data.

- Q: Why?
- A: Because new data will probably not be exactly the same as the training data!

The accuracy/error estimates on the training data measure the degree of classifier's overfitting.

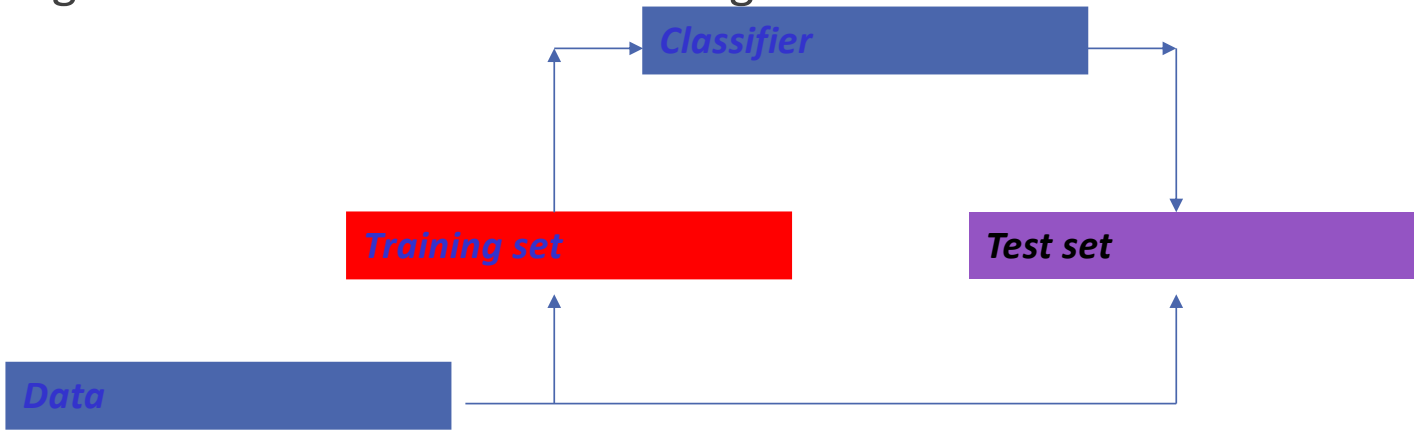
Estimation with Independent Test Data

Estimation with independent test data is used when we have plenty of data and there is a natural way to forming training and test data.

For example: Quinlan in 1987 reported experiments in a medical domain for which the classifiers were trained on data from 1985 and tested on data from 1986.

Hold-out Method
(when direct experiment not available)

The hold-out method splits the data into training data and test data (usually 2/3 for train, 1/3 for test). Then we build a classifier using the train data and test it using the test data.



The hold-out method is usually used when we have thousands of instances, including several hundred instances from each class.

Holdout validation

We are interested in *generalization* – the performance on data not used for training

Given only one data set, we hold out some data for evaluation
- holdout set for final evaluation is called the "test set"

Accuracy on training data is sometimes called "in-sample" accuracy

vs. "out-of-sample" accuracy on test data
- a.k.a. "holdout accuracy"
- an estimate of "generalization accuracy"

Summary

1. Use part of your data for training (“training sample”)
2. Another part for validation (“validation”, “holdout”, “test set”)
3. Third sample for predicting the accuracy of your classifier (“prediction sample”)

Generally, the larger the training data the better the classifier (but returns diminish).

Conclusions

Takeaways for Decision Trees

Interpretable. Decision trees are the most popular classification model because they are easily explainable; they are effective in picking out nonlinear (and combination) relationships between the predictors and the target.

Overfitting. Decision trees are prone to overfitting

Avoid overfitting.

- Prune the tree to the right size using cross-validation
- Build an ensemble model by bagging (sampling the data points to build different trees and aggregating their decisions via a majority)
- Use a random forest that picks different random subsets for different trees to create an ensemble model that avoids overfitting

Validation. All predictive models, and in particular classification models are tested for fidelity and overfitting by cross-validating against a held out sample.

Metrics. A confusion matrix summarizes the performance of the model on a test set. The ROC curve summarizes confusion matrices over a range of thresholds. The AUC, sensitivity and specificity are popular accuracy measures. The problem being studied might suggest more relevant metrics based on the confusion matrix.