Statistics 202: Data Mining

Classification & Decision Trees

Based in part on slides from textbook, slides of Susan Holmes

©Jonathan Taylor

October 19, 2012

Classification

Problem description

- We are given a data matrix $\boldsymbol{X}$ with either continuous or discrete variables such that each row $X_i \in \mathcal{F}$ and a set of labels $\boldsymbol{Y} \in \mathcal{L}$.
- For a k-class problem, $\#\mathcal{L} = k$ and we can think of $\mathcal{L} = \{1, \dots, k\}$.
- Our goal is to find a classifier

$$f : \mathcal{F} \to \mathcal{L}$$

that allows us to predict the label of a new observation given a new set of features.

Classification

A supervised problem

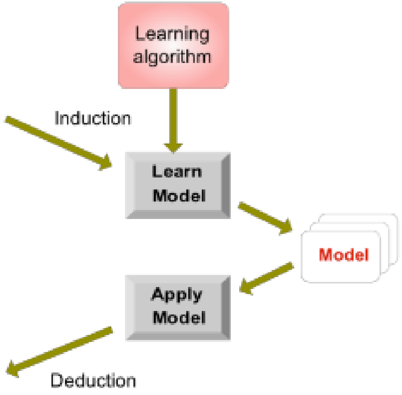
- Classification is a supervised problem.
- Usually, we use a subset of the data, the *training set* to learn or estimate the classifier yielding $\hat{f} = \hat{f}_{training}$.
- The performance of $\hat{f}$ is measured by applying it to each case in the *test set* and computing

$$\sum_{j \in test} L(\hat{f}_{training}(\boldsymbol{X}_j), \boldsymbol{Y}_j)$$

Classification

Tid	Attrib1	Attrib2	Attrib3	Class
1	Yes	Large	125K	No
2	No	Medium	100K	No
3	No	Small	70K	No
4	Yes	Medium	120K	No
5	No	Large	95K	Yes
6	No	Medium	60K	No
7	Yes	Large	220K	No
8	No	Small	85K	Yes
9	No	Medium	75K	No
10	No	Small	90K	Yes

Training Set



Tid	Attrib1	Attrib2	Attrib3	Class
11	No	Small	55K	?
12	Yes	Medium	80K	?
13	Yes	Large	110K	?
14	No	Small	95K	?
15	No	Large	67K	?

Test Set

Classification

Examples of classification tasks

- Predicting whether a tumor is benign or malignant.
- Classifying credit card transactions as fraudulent or legitimate.
- Predicting the type of a given tumor among several types.
- Cateogrizing a document or news story as one of {finance, weather, sports, etc.}

Classification

Common techniques

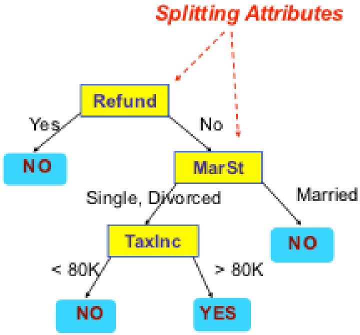
- Decision Tree based Methods
- Rule-based Methods
- Discriminant Analysis
- Memory based reasoning
- Neural Networks
- Naïve Bayes
- Support Vector Machines

Classification trees

categorical categorical continuous class

Tid	Refund	Marital Status	Taxable Income	Cheat
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes

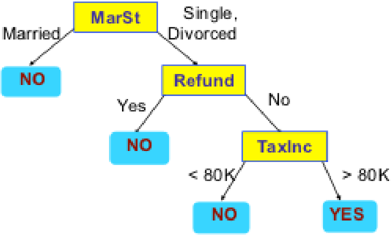
Training Data



Model: Decision Tree

Classification trees

	categorical	categorical	continuous	class
Tid	**Refund**	**Marital Status**	**Taxable Income**	**Cheat**
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes

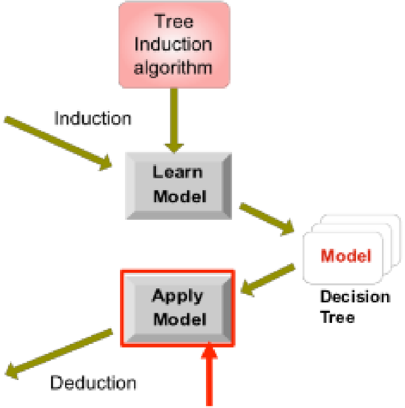


There could be more than one tree that fits the same data!

Applying a decision tree rule

Tid	Attrib1	Attrib2	Attrib3	Class
1	Yes	Large	125K	No
2	No	Medium	100K	No
3	No	Small	70K	No
4	Yes	Medium	120K	No
5	No	Large	95K	Yes
6	No	Medium	60K	No
7	Yes	Large	220K	No
8	No	Small	85K	Yes
9	No	Medium	75K	No
10	No	Small	90K	Yes

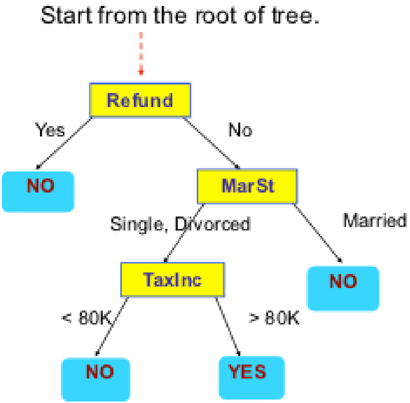
Training Set



Tid	Attrib1	Attrib2	Attrib3	Class
11	No	Small	55K	?
12	Yes	Medium	80K	?
13	Yes	Large	110K	?
14	No	Small	95K	?
15	No	Large	67K	?

Test Set

Applying a decision tree rule



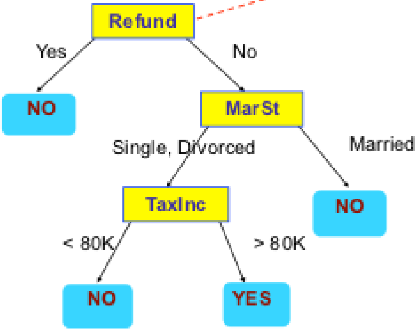
Test Data

Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?

Applying a decision tree rule

Test Data

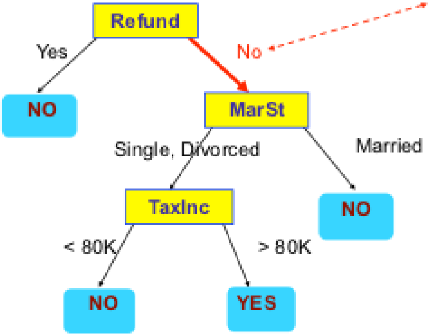
Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?



Applying a decision tree rule

Test Data

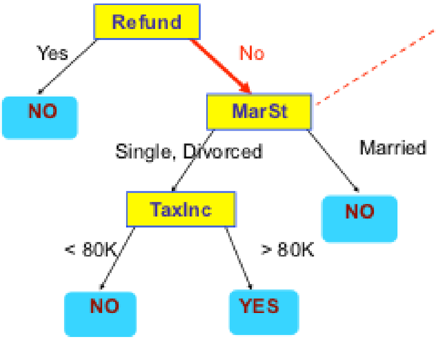
Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?



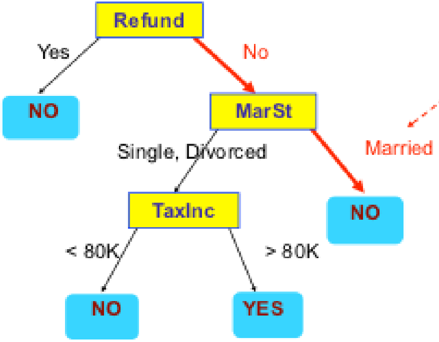
Applying a decision tree rule

Test Data

Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?



Applying a decision tree rule



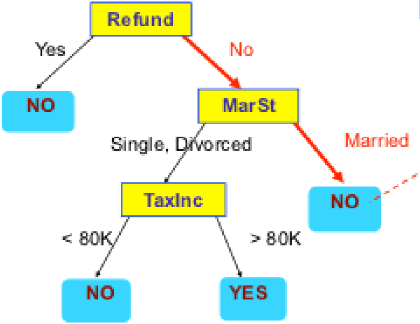
Test Data

Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?

Applying a decision tree rule

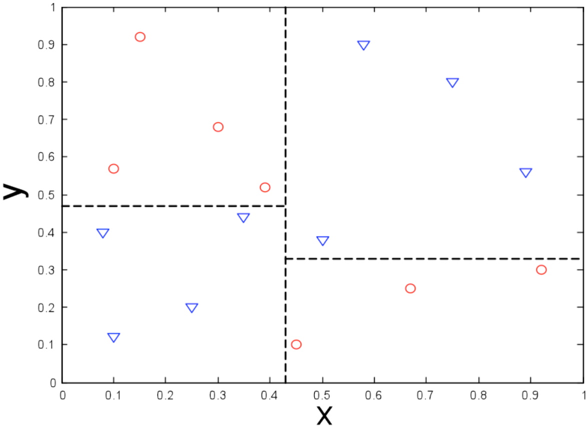
Test Data

Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?

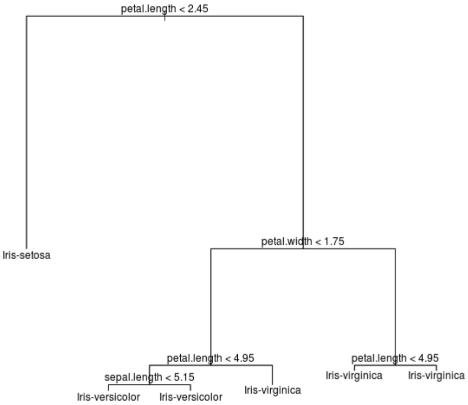


Assign Cheat to “No”

Decision boundary for tree

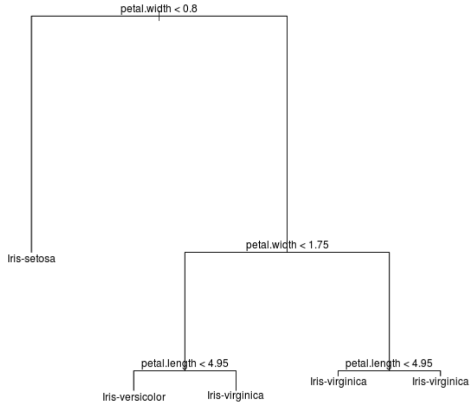


Decision tree for iris data using all features



Decision tree for iris data using `petal.length`, `petal.width`

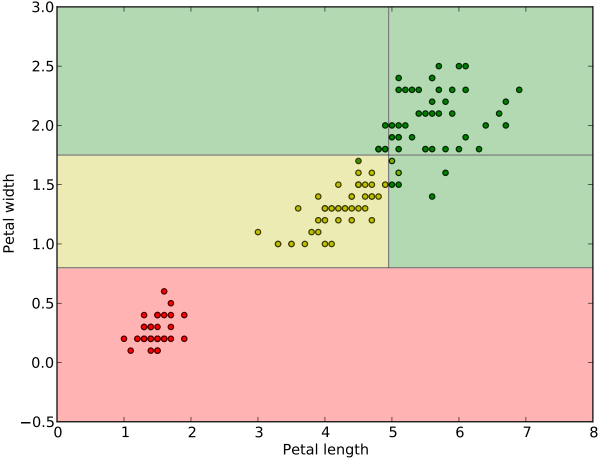
Statistics 202: Data Mining



Regions in `petal.length`, `petal.width` plane

Statistics 202: Data Mining

©Jonathan Taylor



Decision boundary for tree

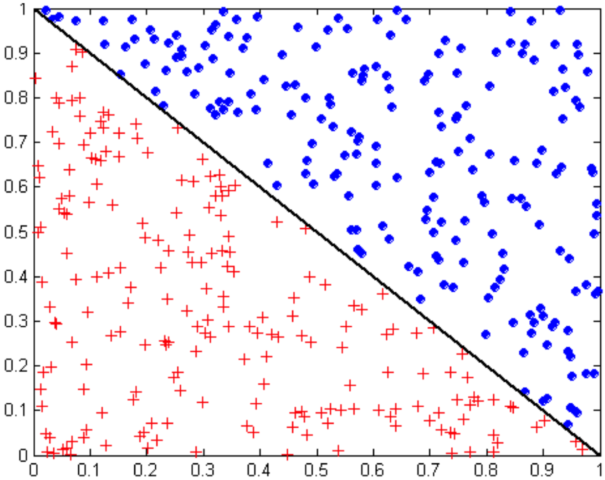
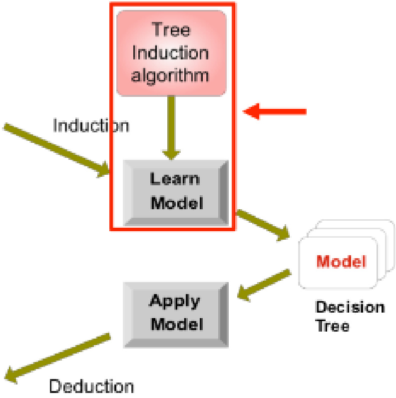


Figure : Trees have trouble capturing structure not parallel to axes

Learning the tree

Tid	Attrib1	Attrib2	Attrib3	Class
1	Yes	Large	125K	No
2	No	Medium	100K	No
3	No	Small	70K	No
4	Yes	Medium	120K	No
5	No	Large	95K	Yes
6	No	Medium	60K	No
7	Yes	Large	220K	No
8	No	Small	85K	Yes
9	No	Medium	75K	No
10	No	Small	90K	Yes

Training Set



Tid	Attrib1	Attrib2	Attrib3	Class
11	No	Small	55K	?
12	Yes	Medium	80K	?
13	Yes	Large	110K	?
14	No	Small	95K	?
15	No	Large	67K	?

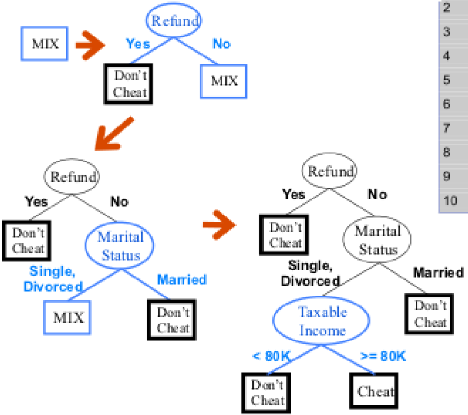
Test Set

Learning the tree

Hunt's algorithm (generic structure)

- Let D_t be the set of training records that reach a node t
- If D_t contains records that belong the same class y_t, then t is a leaf node labeled as y_t.
- If $D_t = \emptyset$, then t is a leaf node labeled by the default class, y_d.
- If D_t contains records that belong to more than one class, use an attribute test to split the data into smaller subsets. Recursively apply the procedure to each subset.
- This splitting procedure is what can vary for different tree learning algorithms . . .

Learning the tree



Tid	Refund	Marital Status	Taxable Income	Cheat
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes

Learning the tree

Issues

Greedy strategy: Split the records based on an attribute test that optimizes certain criterion.

What is the best split: What criterion do we use? Previous example chose first to split on `Refund` ...

How to split the records: Binary or multi-way? Previous example split `Taxable Income` at $\geq 80K$ ...

When do we stop? Should we continue until each node if possible? Previous example stopped with all nodes being completely homogeneous ...

Different splits: ordinal / nominal

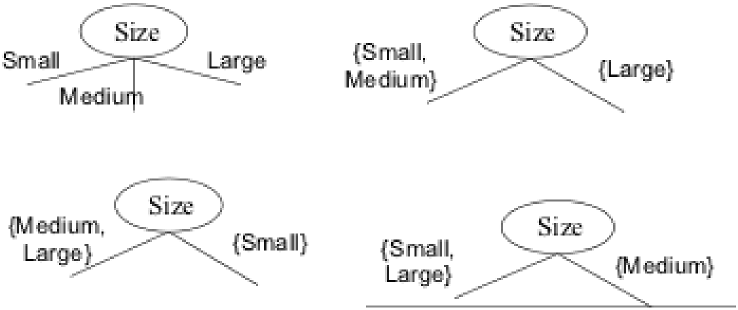


Figure : Binary or multi-way?

Different splits: continuous

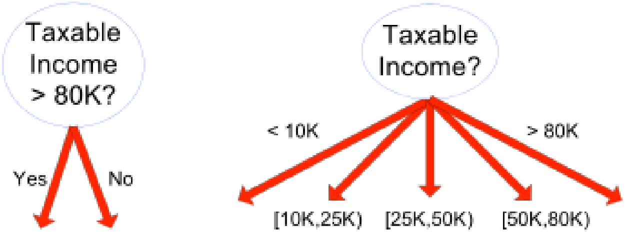


Figure : Binary or multi-way?

Choosing a variable to split on

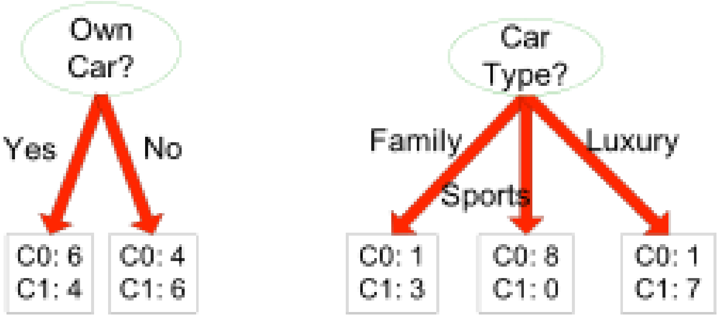


Figure : Which should we start the splitting on?

Learning the tree

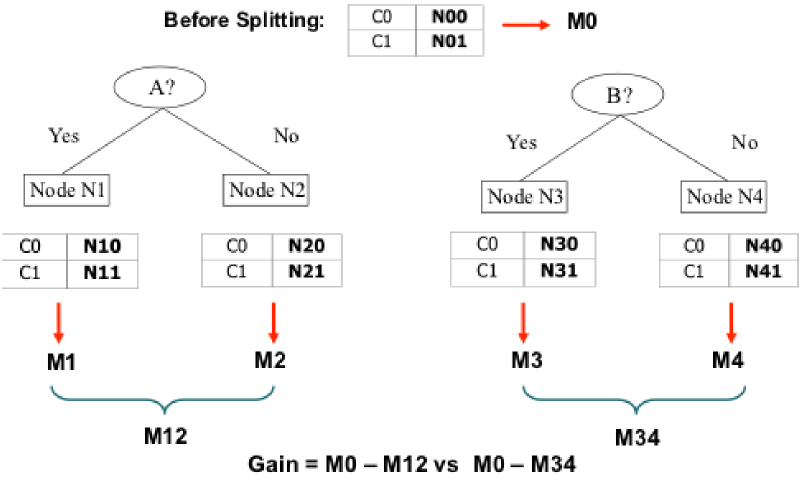
Choosing the best split

- Need some numerical criterion to choose among possible splits.
- Criterion should favor *homogeneous or pure* nodes.
- Common cost functions:
  - Gini Index
  - Entropy / Deviance / Information
  - Misclassification Error

Choosing a variable to split on

Statistics 202: Data Mining

©Jonathan Taylor



Learning the tree

GINI Index

- Suppose we have k classes and node t has frequencies $p_t = (p_{1,t}, \dots, p_{k,t})$.
- Criterion

$$GINI(t) = \sum_{(j,j') \in \{1,\dots,k\}: j \neq j'} p_{j,t} p_{j',t} = 1 - \sum_{j=1}^{l} p_{j,t}^2.$$

- Maximized when $p_{j,t} = 1/k$ with value $1 - 1/k$
- Minimized when all records belong to a single class.
- Minimizing *GINI* will favour *pure* nodes ...

Learning the tree

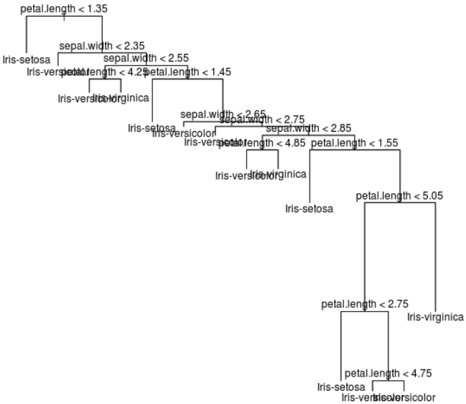
Gain in GINI Index for a potential split

- Suppose t is to be split into j new child nodes $(t_l)_{1 \leq l \leq j}$.
- Each child node has a count n_l and a vector of frequencies $(p_{1,t_l}, \dots, p_{k,t_l})$. Hence they have their own GINI index, $GINI(t_l)$.
- The gain in GINI Index for this split is

$$\text{Gain}(GINI, t \to (t_l)_{1 \leq l \leq j}) = GINI(t) - \frac{\sum_{l=1}^{j} n_l GINI(t_l)}{\sum_{l=1}^{j} n_l}.$$

- Greedy algorithm chooses the biggest gain in GINI index among a list of possible splits.

Decision tree for iris data using all features with GINI



Learning the tree

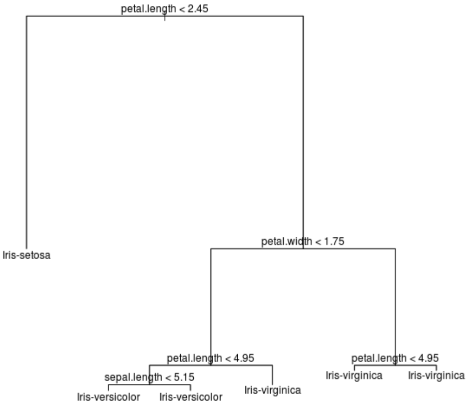
Entropy / Deviance / Information

- Suppose we have k classes and node t has frequencies $p_t = (p_{1,t}, \dots, p_{k,t})$.
- Criterion

$$H(t) = -\sum_{j=1}^{k} p_{j,t} \log p_{j,t}$$

- Maximized when $p_{i,t} = 1/k$ with value $\log k$
- Minimized when one class has no records in it.
- Minimizing entropy will favour *pure* nodes ...

Decision tree for iris data using all features with Entropy



Learning the tree

Gain in entropy for a potential split

- Suppose t is to be split into j new child nodes $(t_l)_{1 \leq l \leq j}$.
- Each child node has a count n_l and a vector of frequencies $(p_{1,t_l}, \dots, p_{k,t_l})$. Hence they have their own entropy $H(t_l)$.
- The gain in entropy for this split is

$$\mathrm{Gain}(H, t \to (t_l)_{1 \leq l \leq j}) = H(t) - \frac{\sum_{l=1}^{j} n_l H(t_l)}{\sum_{l=1}^{j} n_l}.$$

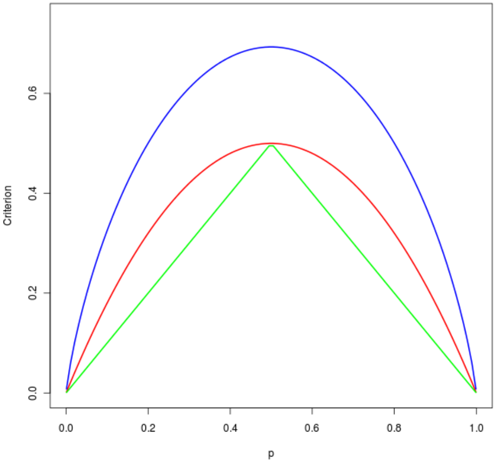
- Greedy algorithm chooses the biggest gain in H among a list of possible splits.

Learning the tree

Misclassification Error

- Suppose we have k classes and node t has frequencies $p_t = (p_{1,t}, \dots, p_{k,t})$.
- The mode is
$$\hat{k}(t) = \operatorname*{argmax}_k p_{k,t}.$$
- Criterion
$$\text{Misclassification Error}(t) = 1 - p_{\hat{k}(t),t}$$
- Not smooth in p_t as $GINI$, H, can be more difficult to optimize numerically.

Different criteria: *GINI*, *H*, *MC*



Learning the tree

Misclassification Error

- Example: suppose parent has 10 cases: $\{7D, 3R\}$
- A candidate split produces two nodes: $\{3D, 0R\}$, $\{4D, 3R\}$.
- The gain in MC is 0, but gain in GINI is $0.42 - 0.342 > 0$.
- Similarly, entropy will also show an improvement ...

Choosing the split for a continuous variable

Cheat	No	No	No	Yes	Yes	Yes	No	No	No	No
Taxable Income	60	70	75	85	90	95	100	120	125	220

Split	55		65		72		80		87		92		97		110		122		172		230	
	<=	>	<=	>	<=	>	<=	>	<=	>	<=	>	<=	>	<=	>	<=	>	<=	>	<=	>
Yes	0	3	0	3	0	3	0	3	1	2	2	1	3	0	3	0	3	0	3	0	3	0
No	0	7	1	6	2	5	3	4	3	4	3	4	3	4	4	3	5	2	6	1	7	0
Gini	0.420		0.400		0.375		0.343		0.417		0.400		<u>0.300</u>		0.343		0.375		0.400		0.420	

Learning the tree

Stopping training

- As trees get deeper, or if splits are multi-way the number of data points per leaf node drops very quickly.
- Trees that are too deep tend to overfit the data.
- A common strategy is to "prune" the tree by removing some internal nodes.

Learning the tree

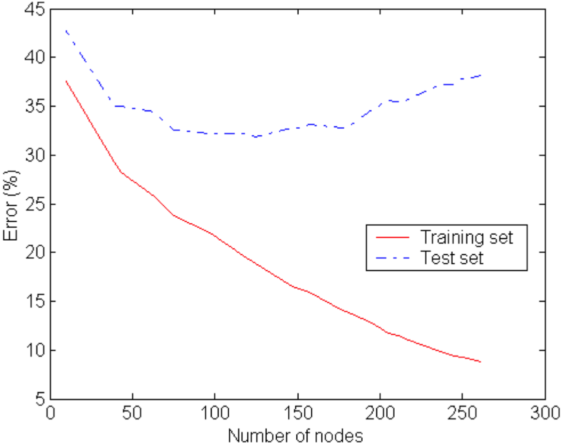


Figure : Underfitting corresponds to the left-hand side, overfit to the right

Learning the tree

Cost-complexity pruning (`tree` library)

- Given a criterion Q like H or $GINI$, we define the cost-complexity of a tree with terminal nodes $(t_j)_{1 \leq j \leq m}$

$$C_\alpha(T) = \sum_{j=1}^{m} n_j Q(t_j) + \alpha m$$

- Given a large tree T_L we might compute $C_\alpha(T)$ for any subtree T of T_L.
- The optimal tree is defined as

$$\hat{T}_\alpha = \operatorname*{argmin}_{T \leq T_L} C_\alpha(T).$$

- Can be found by "weakest-link" pruning. See *Elements of Statistical Learning* for more ...

Learning the tree

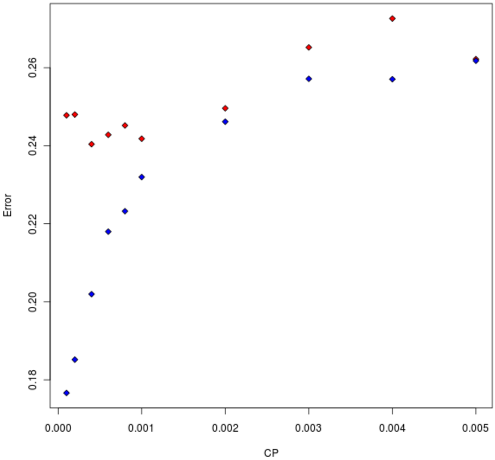
Pre-pruning (`rpart` library)

- These methods stop the algorithm before it becomes a fully-grown tree.
- Examples
  - Stop if all instances belong to the same class (kind of obvious).
  - Stop if number of instances is less than some user-specified threshold. Both `tree`, `rpart` have rules like this.
  - Stop if class distribution of instances are independent of the available features (e.g., using χ^2 test)
  - Stop if expanding the current node does not improve impurity measures (e.g., Gini or information gain). This relates to `cp` in `rpart`.

Training and test error as a function of cp

Statistics 202: Data Mining

©Jonathan Taylor



Evaluating a classifier

	PREDICTED CLASS	
ACTUAL CLASS	Class=Yes	Class=No
Class=Yes	a (TP)	b (FN)
Class=No	c (FP)	d (TN)

Evaluating a classifier

Measures of performance

- Simplest is accuracy

$$\begin{aligned}\text{Accuracy} &= \frac{TP + TN}{TP + TN + FP + FN} \\ &= \text{SMC(Actual, Predicted)} \\ &= 1 - \text{Misclassification Rate}\end{aligned}$$

Evaluating a classifier

Accuracy isn't everything

- Consider an unbalanced 2-class problem with # 1's=10, # 0's=9990.
- Simply labelling everything 0 yields 99.9% accuracy.
- But, this classifier misses all class 1.

Evaluating a classifier

	PREDICTED CLASS		
ACTUAL CLASS	C(i|j)	**Class=Yes**	**Class=No**
	Class=Yes	C(Yes|Yes)	C(No|Yes)
	Class=No	C(Yes|No)	C(No|No)

Learning the tree

Measures of performance

- Classification rule changes to

$$\text{Label}(p, C) = \text{argmin}_i \sum_j C(i|j) p_j$$

- Accuracy is the same as cost if $C(Y|Y) = C(N|N) = c_1$, $C(Y|N) = C(N|Y) = c_2$.

Evaluating a classifier

Cost Matrix	PREDICTED CLASS		
ACTUAL CLASS	C(i|j)	+	-
	+	-1	100
	-	1	0

Model M_1	PREDICTED CLASS		
ACTUAL CLASS		+	-
	+	150	40
	-	60	250

Accuracy = 80%
Cost = 3910

Model M_2	PREDICTED CLASS		
ACTUAL CLASS		+	-
	+	250	45
	-	5	200

Accuracy = 90%
Cost = 4255

Evaluating a classifier

Measures of performance

- Other common ones

$$
\begin{aligned}
\text{Precision} &= \frac{TP}{TP + FP} \\
\text{Specificity} &= \frac{TN}{TN + FP} = TNR \\
\text{Sensitivity} = \text{Recall} &= \frac{TP}{TP + FN} = TPR \\
F &= \frac{2 \cdot \text{Recall} \cdot \text{Precision}}{\text{Recall} + \text{Precision}} \\
&= \frac{2 \cdot TP}{2 \cdot TP + FN + FP}
\end{aligned}
$$

Evaluating a classifier

Measures of performance

- Precision emphasizes $P(p = Y, a = Y)$ & $P(p = Y, a = N)$.
- Recall emphasizes $P(p = Y, a = Y)$ & $P(p = N, a = Y)$.
- $FPR = 1 - TNR$
- $FNR = 1 - TPR$.

Evaluating a classifier

Measure of performance

- We have done some simple training / test splits to see how well our classifier is doing.
- More accurately, this procedure measures how well our algorithm for *learning the classifier* is doing.
- How well this works may depend on

  Model: Are we using the right type of classifier model?

  Cost: Is our algorithm sensitive to the cost of misclassification?

  Data size: Do we have enough data to learn a model?

Evaluating a classifier

Statistics 202: Data Mining

©Jonathan Taylor

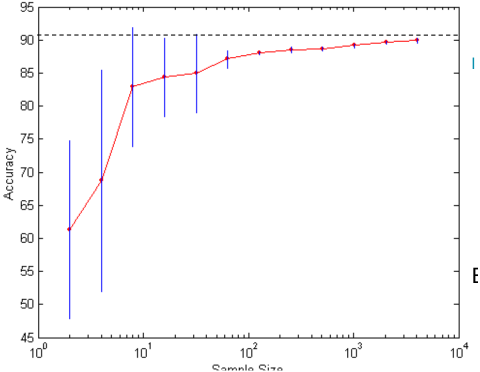


Figure : As data increases, our estimate of accuracy improves, as does the variability of our estimate . . .

Evaluating a classifier

Estimating performance

Holdout: Split into test and training (e.g. 1/3 test, 2/3 training).

Random subsampling: Repeated replicates of holdout, averaging results.

Cross validation: Partition data into K disjoint subsets. For each subset S_i, train on all but S_i, then test on S_i.

Stratified sampling: May be helpful to sample so Y/N class is roughly equal in training data.

0.632 Bootstrap: Combine training error and bootstrap error ...