Decision Trees

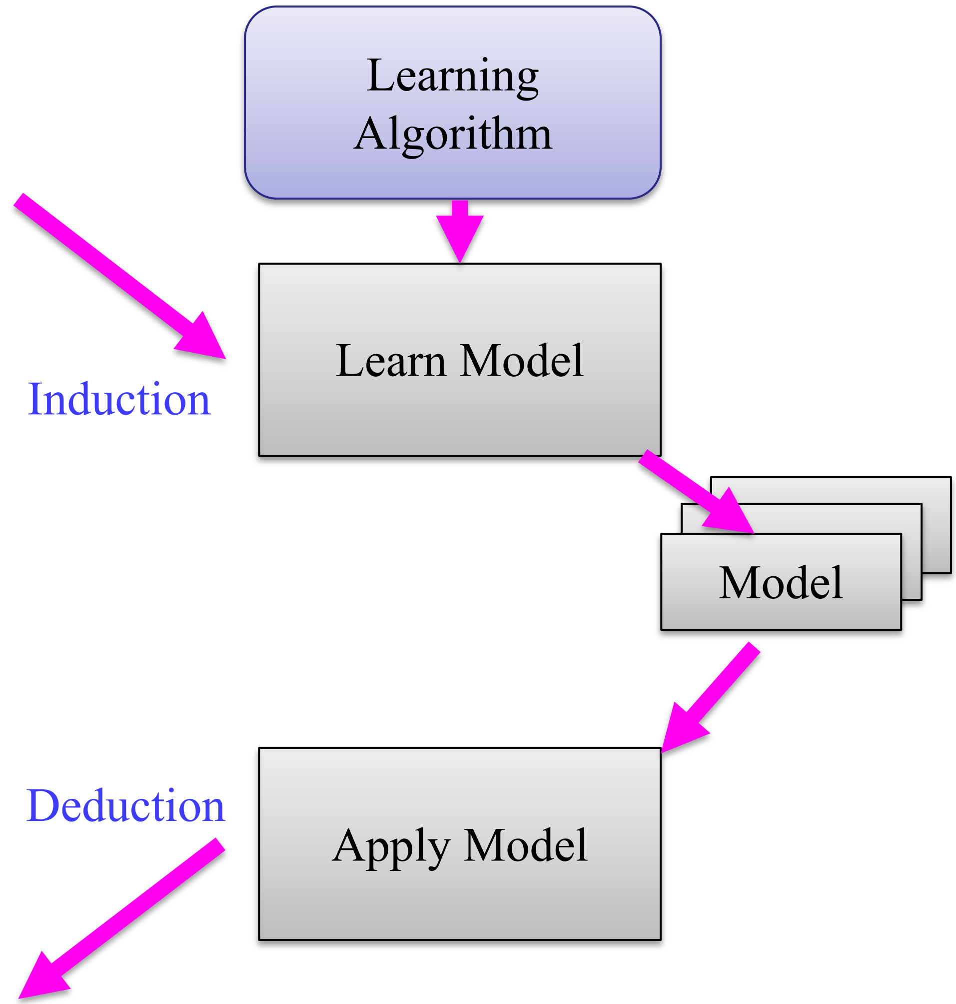
15/04/2024

Koustav Rudra

Illustrating Classification Task

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
1	No	Small	55k	?
2	Yes	Medium	80k	?
3	Yes	Large	110k	?
4	No	Small	95k	?
5	No	Large	67k	?

Test Set

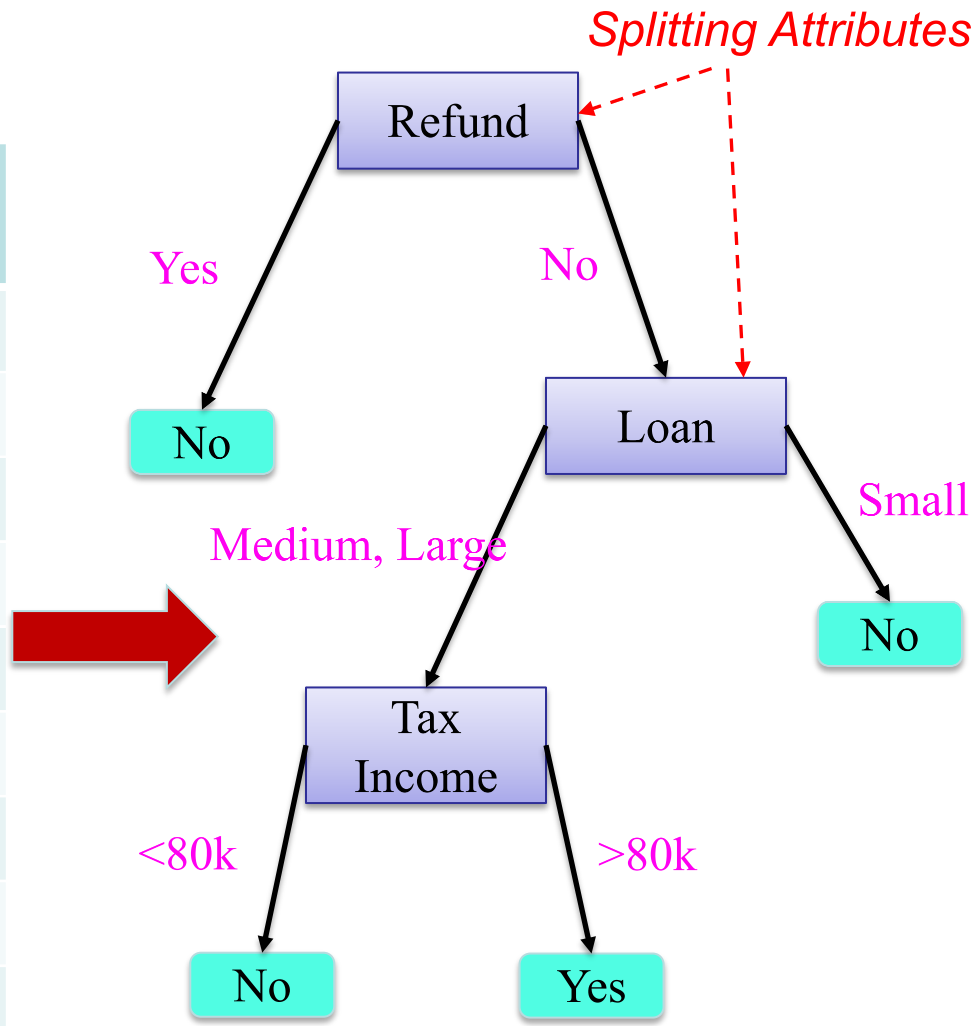
Intuition behind a decision tree

- Ask a series of questions about a given record
  - Each question is about one of the attributes
  - Answer to one question decides what question to ask next (or if a next question is needed)
  - Continue asking questions until we can infer the class of the given record

Example of a Decision Tree

categorical categorical continuous class

Tid	Refund	Loan Status	Taxable Income	Cheat
1	Yes	Medium	125k	No
2	No	Small	100k	No
3	No	Medium	70k	No
4	Yes	Small	120k	No
5	No	Large	95k	Yes
6	No	Small	60k	No
7	Yes	Large	220k	No
8	No	Medium	85k	Yes
9	No	Small	75k	No
10	No	Medium	90k	Yes



Model: Decision Tree

Structure of a decision tree

- Decision tree: hierarchical structure
  - One root node: no incoming edge, zero or more outgoing edges
  - Internal nodes: exactly one incoming edge, two or more outgoing edges
  - Leaf or terminal nodes: exactly one incoming edge, no outgoing edge

- Each leaf node assigned a class label

- Each non-leaf node contains a test condition on one of the attributes

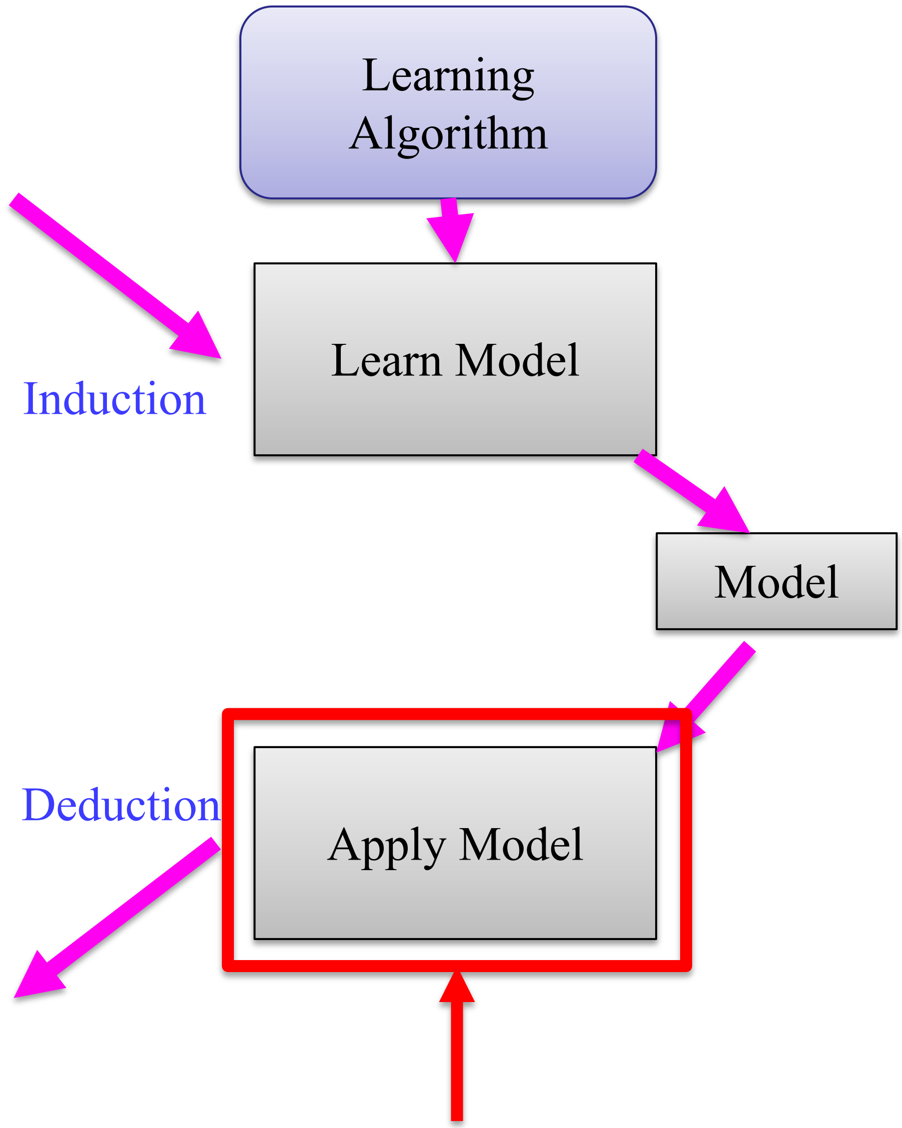
Applying a Decision-Tree Classifier

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

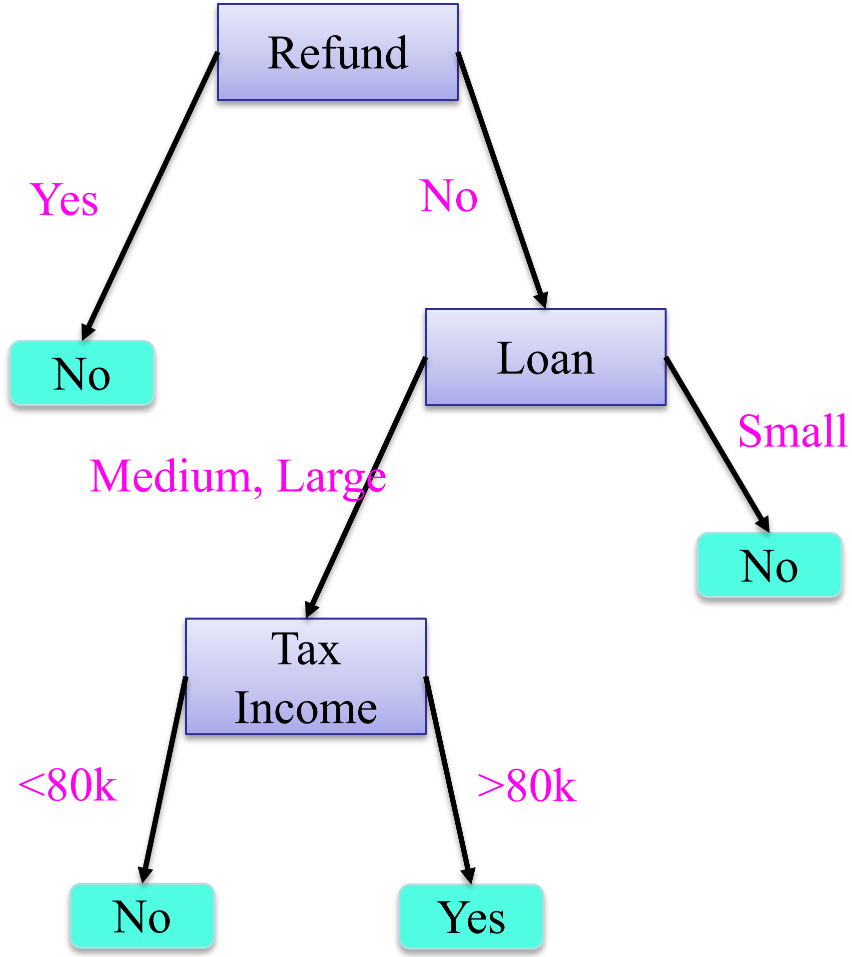
Tid	Attrib1	Attrib2	Attrib3	Class
1	No	Small	55k	?
2	Yes	Medium	80k	?
3	Yes	Large	110k	?
4	No	Small	95k	?
5	No	Large	67k	?

Test Set



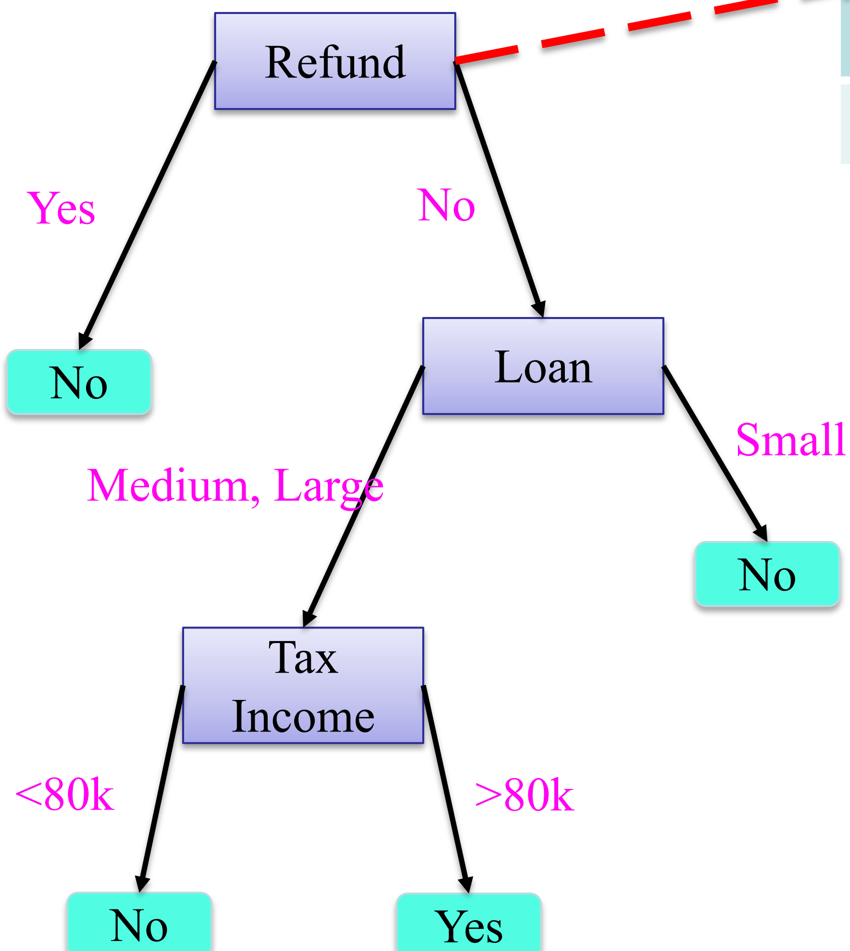
Applying Model to Test Data

Refund	Loan Status	Taxable Income	Cheat
No	Small	80k	?



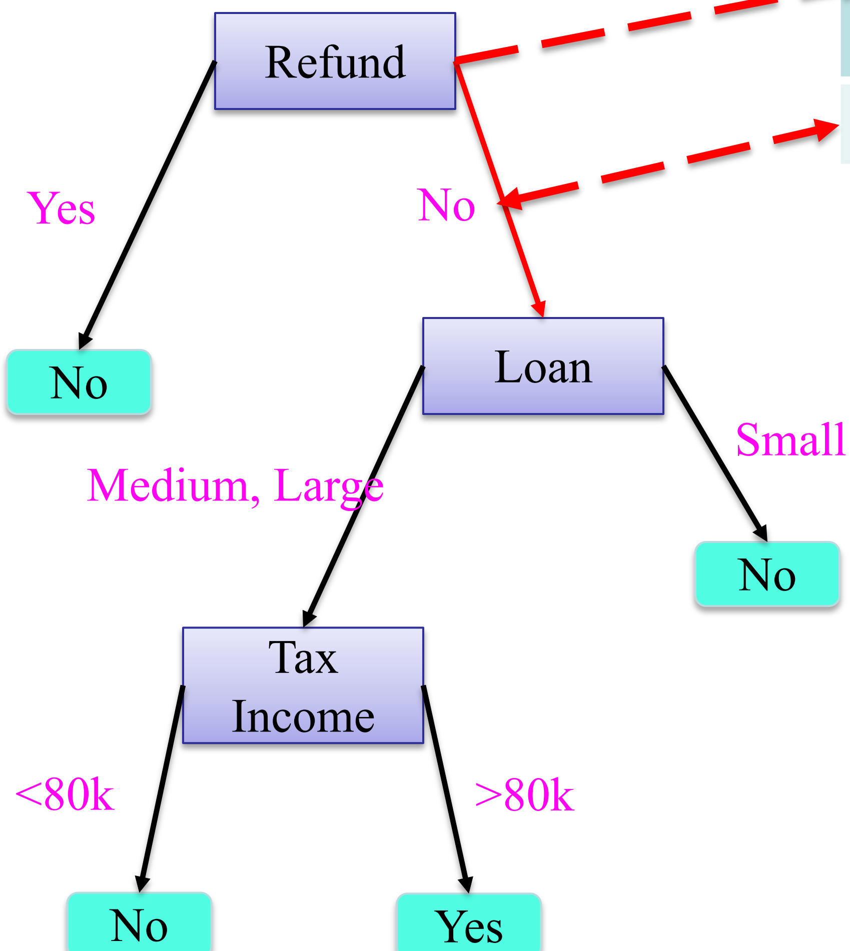
Once a decision tree has been constructed (learned), it is easy to apply it to test data

Applying Model to Test Data



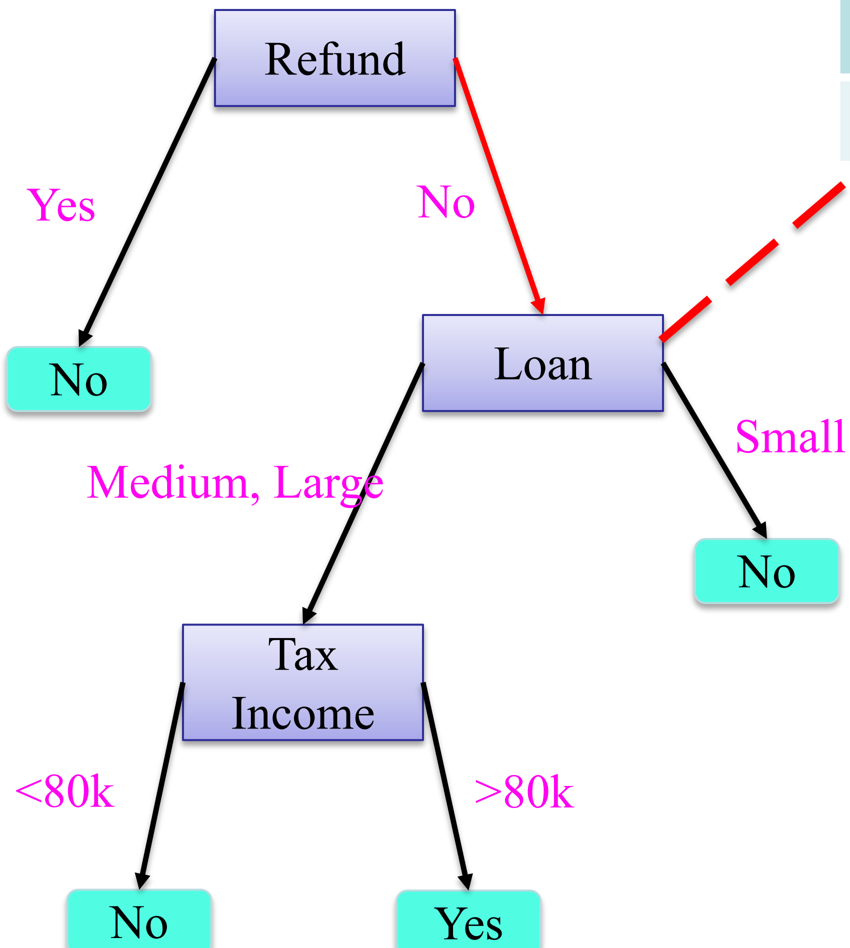
Refund	**Loan Status**	**Taxable Income**	**Cheat**
No	Small	80k	?

Applying Model to Test Data



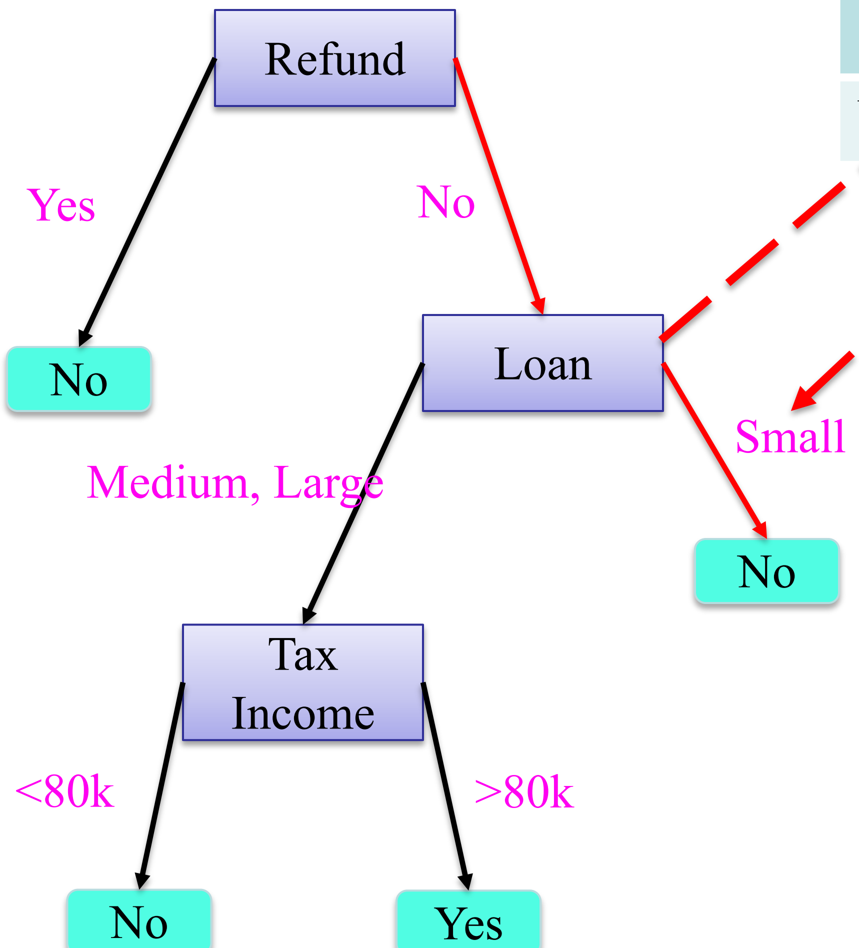
Refund	Loan Status	Taxable Income	Cheat
No	Small	80k	?

Applying Model to Test Data



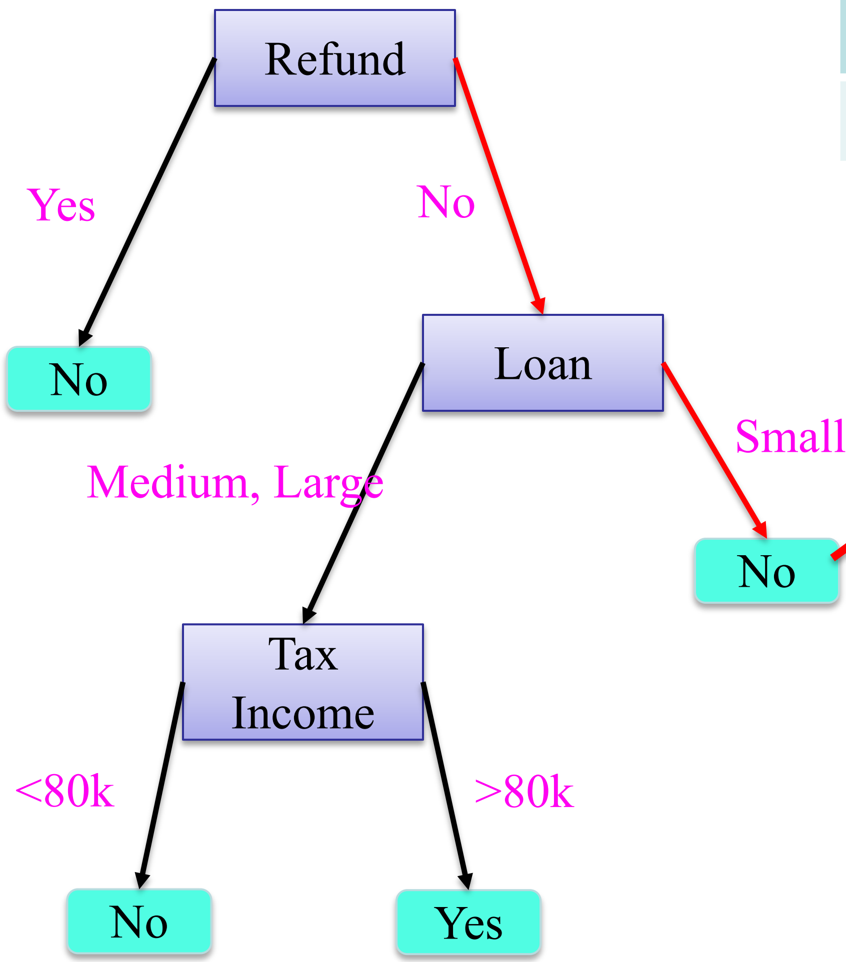
Refund	Loan Status	Taxable Income	Cheat
No	Small	80k	?

Applying Model to Test Data



Refund	Loan Status	Taxable Income	Cheat
No	Small	80k	?

Applying Model to Test Data



Refund	Loan Status	Taxable Income	Cheat
No	Small	80k	?

Assign Cheat to “No”

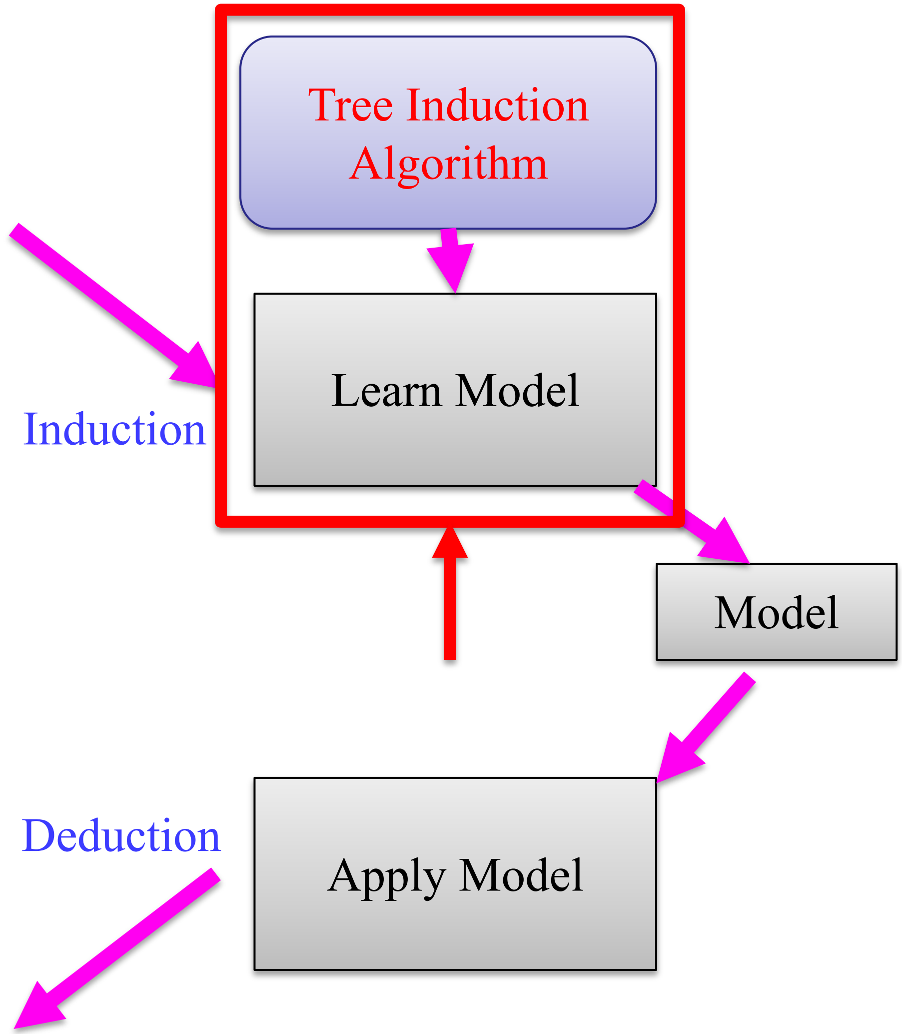
Learning a Decision-Tree Classifier

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
1	No	Small	55k	?
2	Yes	Medium	80k	?
3	Yes	Large	110k	?
4	No	Small	95k	?
5	No	Large	67k	?

Test Set



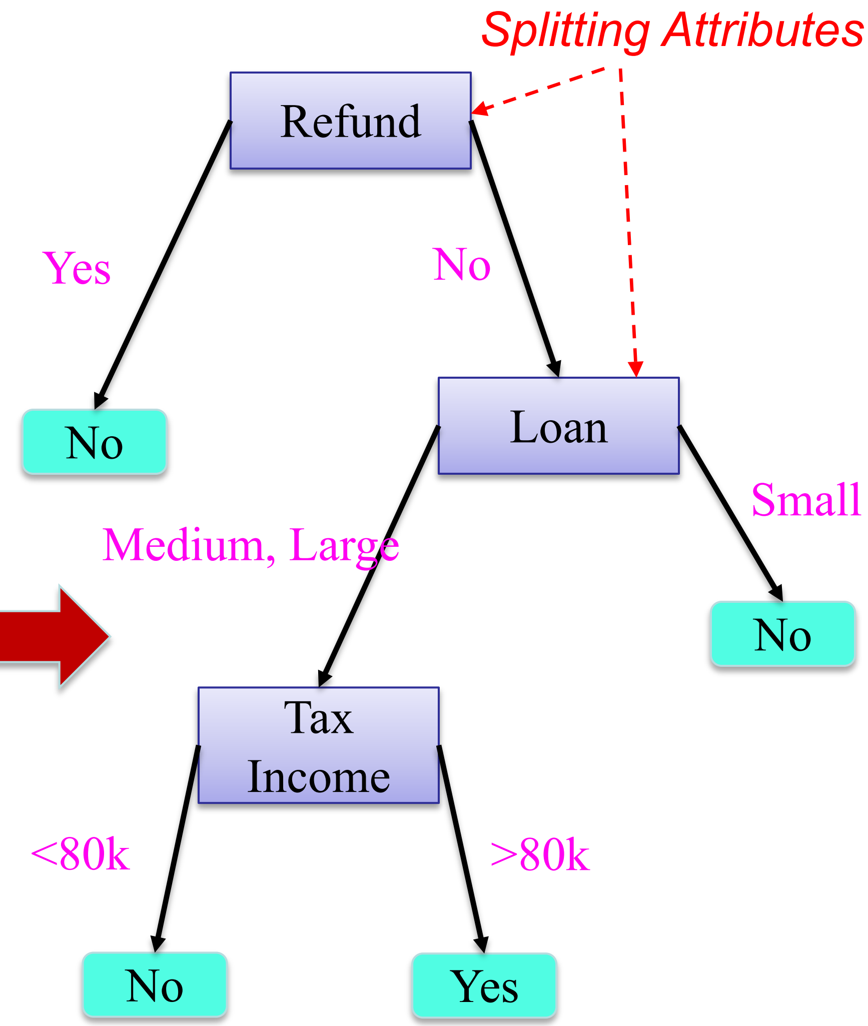
How to learn a decision tree?

A Decision Tree

categorical categorical continuous class

Tid	Refund	Loan Status	Taxable Income	Cheat
1	Yes	Medium	125k	No
2	No	Small	100k	No
3	No	Medium	70k	No
4	Yes	Small	120k	No
5	No	Large	95k	Yes
6	No	Small	60k	No
7	Yes	Large	220k	No
8	No	Medium	85k	Yes
9	No	Small	75k	No
10	No	Medium	90k	Yes

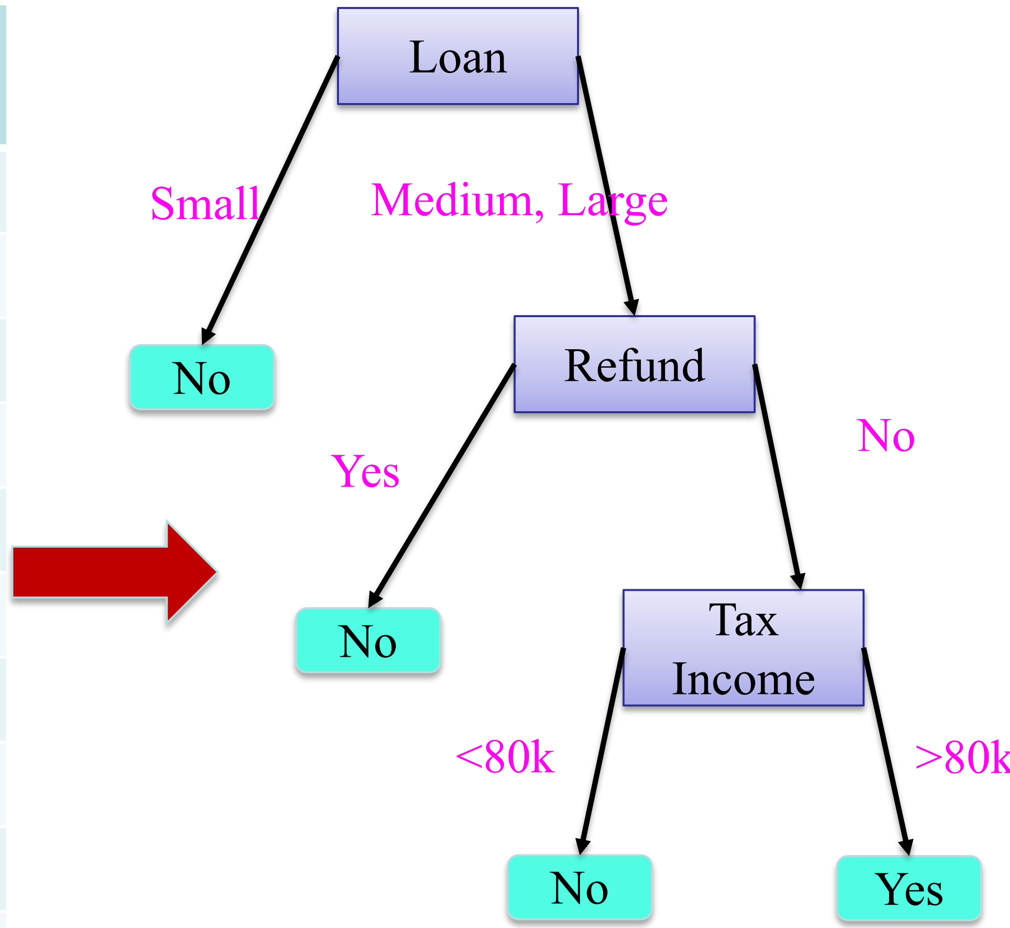
Training data



Model: Decision Tree

Another Decision Tree on same dataset

	categorical	categorical	continuous	class
Tid	**Refund**	**Loan Status**	**Taxable Income**	**Cheat**
1	Yes	Medium	125k	No
2	No	Small	100k	No
3	No	Medium	70k	No
4	Yes	Small	120k	No
5	No	Large	95k	Yes
6	No	Small	60k	No
7	Yes	Large	220k	No
8	No	Medium	85k	Yes
9	No	Small	75k	No
10	No	Medium	90k	Yes



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

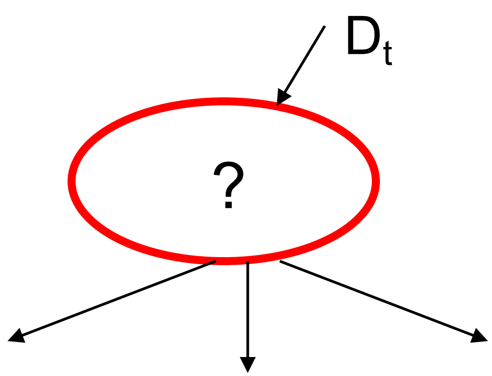
Decision Tree Induction

- Many Algorithms:
  - Hunt's Algorithm (one of the earliest)
  - CART
  - ID3, C4.5
  - SLIQ,SPRINT

General Structure of Hunt's Algorithm

- Let D_t be the set of training records that reach a node t
- General Procedure:
  - If D_t contains records that all belong the same class y_t, then t is a leaf node labeled as y_t
  - If D_t is an empty set, 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

Tid	Refund	Loan Status	Taxabl e Income	Cheat
1	Yes	Medium	125k	No
2	No	Small	100k	No
3	No	Medium	70k	No
4	Yes	Small	120k	No
5	No	Large	95k	Yes
6	No	Small	60k	No
7	Yes	Large	220k	No
8	No	Medium	85k	Yes
9	No	Small	75k	No
10	No	Medium	90k	Yes



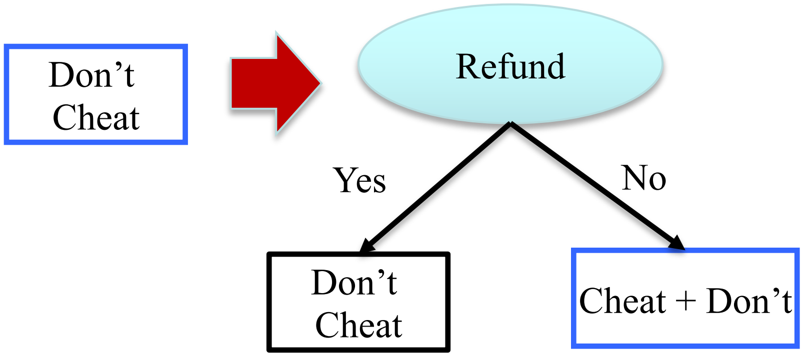
Hunt's Algorithm

Don't
Cheat + Cheat

Tid	Refund	Loan Status	Taxabl e Income	Cheat
1	Yes	Medium	125k	No
2	No	Small	100k	No
3	No	Medium	70k	No
4	Yes	Small	120k	No
5	No	Large	95k	Yes
6	No	Small	60k	No
7	Yes	Large	220k	No
8	No	Medium	85k	Yes
9	No	Small	75k	No
10	No	Medium	90k	Yes

Default class is "Don't cheat" since it is the majority class in the dataset

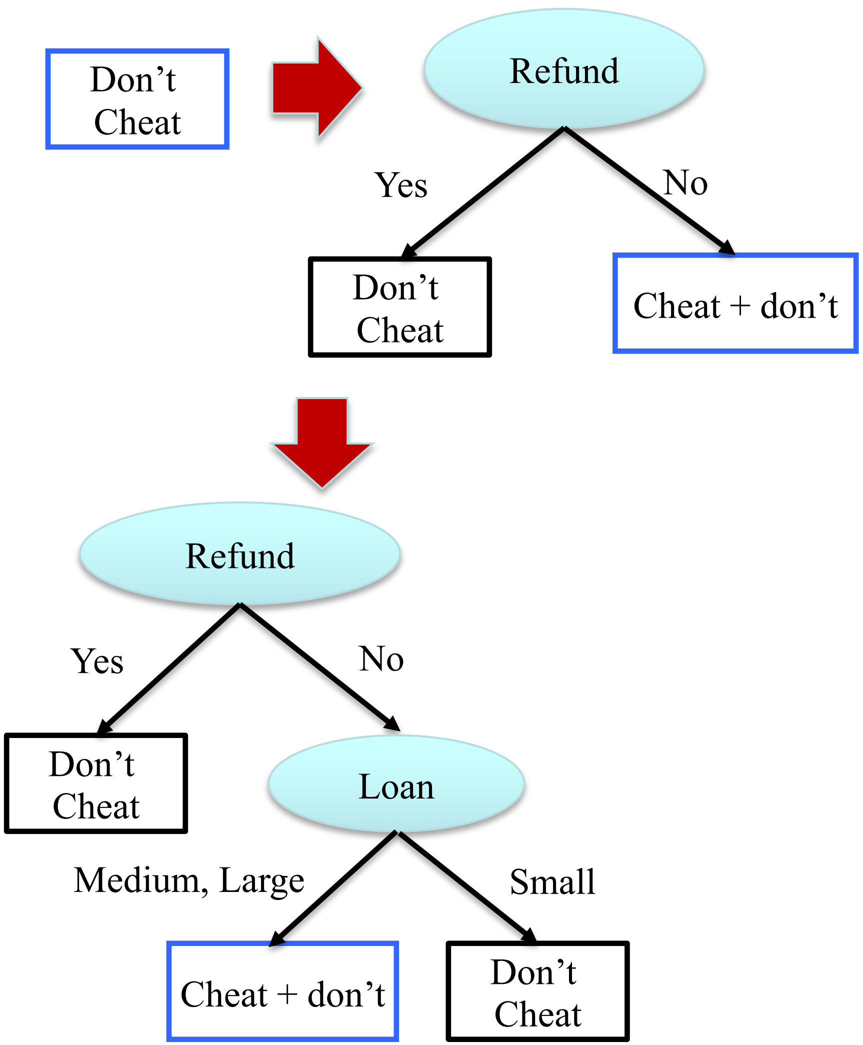
Hunt's Algorithm



Tid	Refund	Loan Status	Taxable Income	Cheat
1	Yes	Medium	125k	No
2	No	Small	100k	No
3	No	Medium	70k	No
4	Yes	Small	120k	No
5	No	Large	95k	Yes
6	No	Small	60k	No
7	Yes	Large	220k	No
8	No	Medium	85k	Yes
9	No	Small	75k	No
10	No	Medium	90k	Yes

For now, assume that "Refund" has been decided to be the best attribute for splitting in some way (to be discussed soon)

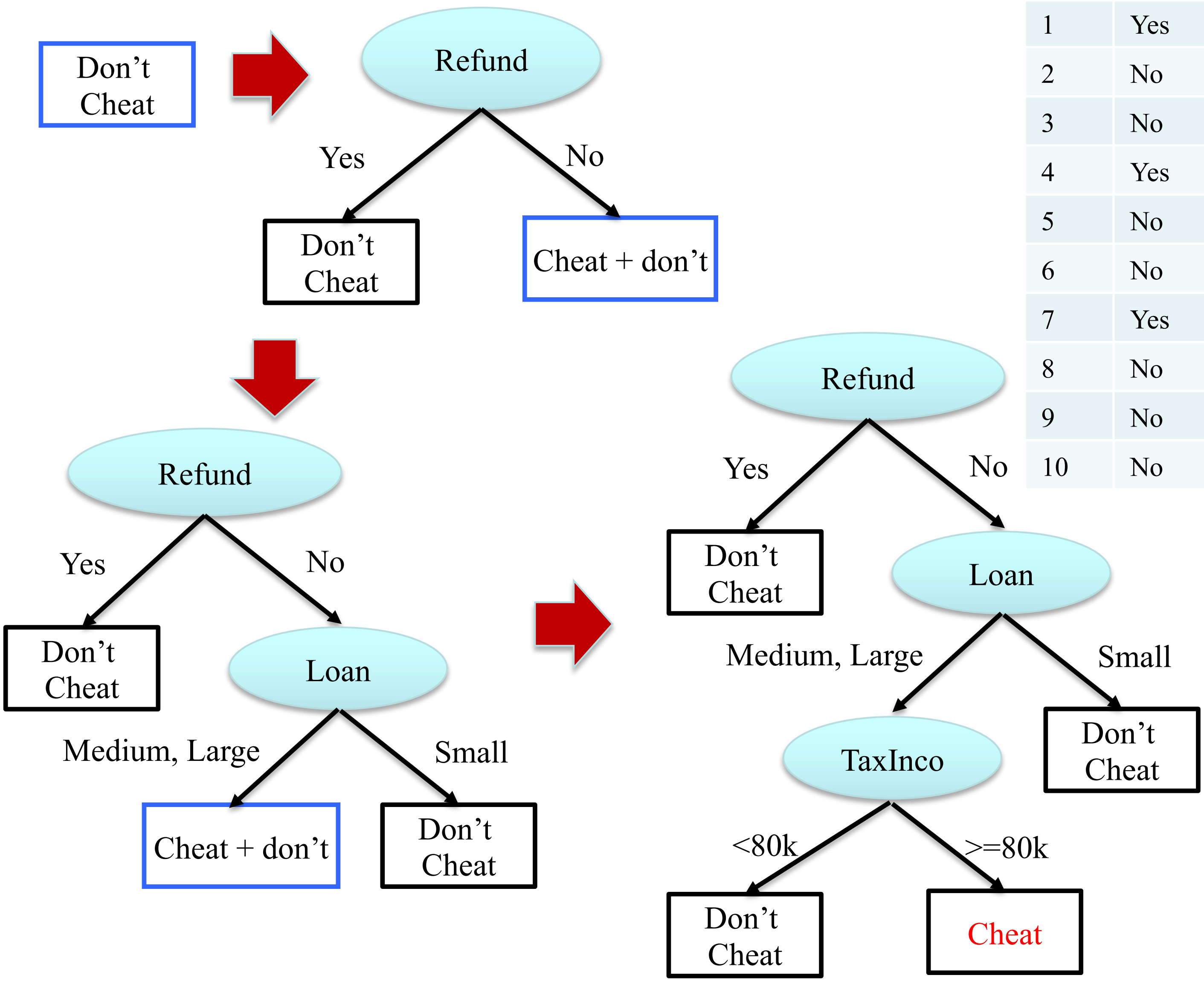
Hunt's Algorithm



Tid	**Refund**	**Loan Status**	**Taxabl e Income**	**Cheat**
1	Yes	Medium	125k	No
2	No	Small	100k	No
3	No	Medium	70k	No
4	Yes	Small	120k	No
5	No	Large	95k	Yes
6	No	Small	60k	No
7	Yes	Large	220k	No
8	No	Medium	85k	Yes
9	No	Small	75k	No
10	No	Medium	90k	Yes

Hunt's Algorithm

Tid	Refund	Loan Status	Taxable Income	Cheat
1	Yes	Medium	125k	No
2	No	Small	100k	No
3	No	Medium	70k	No
4	Yes	Small	120k	No
5	No	Large	95k	Yes
6	No	Small	60k	No
7	Yes	Large	220k	No
8	No	Medium	85k	Yes
9	No	Small	75k	No
10	No	Medium	90k	Yes



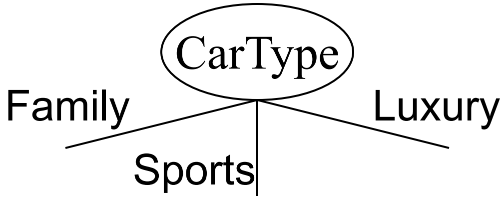
How to Specify Test Condition?

- Depends on attribute types
  - Nominal: two or more distinct values (special case: binary) E.g., Loan status: {small, medium, large}
  - Ordinal: two or more distinct values that have an ordering. E.g. shirt size: {S, M, L, XL}
  - Continuous: continuous range of values

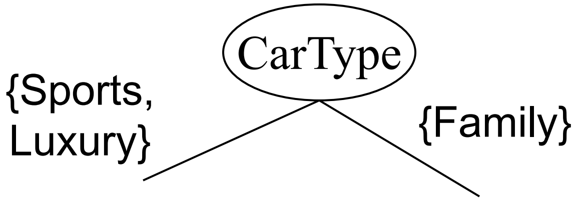
- Depends on number of ways to split
  - 2-way split
  - Multi-way split

Splitting Based on Nominal Attributes

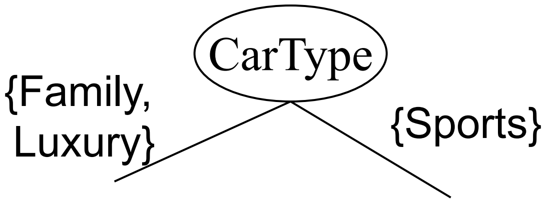
- Multi-way split: Use as many partitions as distinct values.



- Binary split: Divides values into two subsets.
  Need to find optimal partitioning

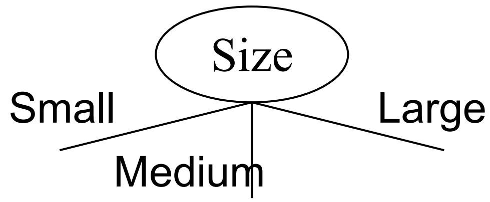


OR

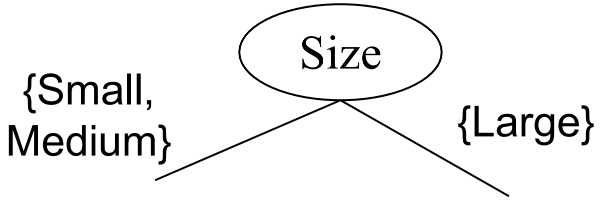


Splitting Based on Ordinal Attributes

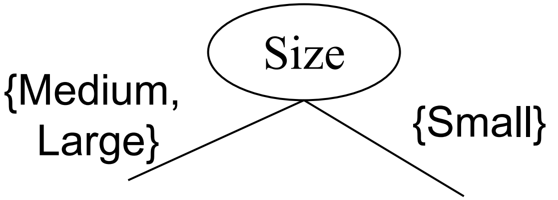
- Multi-way split: Use as many partitions as distinct values.



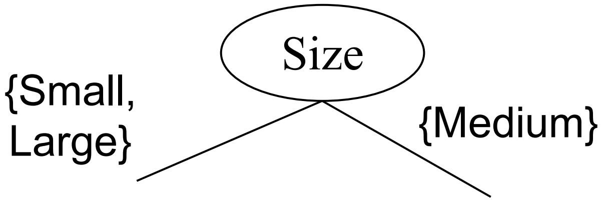
- Binary split: Divides values into two subsets.
  Need to find optimal partitioning.



OR



- What about this split?



Tree Induction

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

- Issues
  - Determine how to split the records
    - How to specify the attribute test condition?
    - How to determine the best split?
  - Determine when to stop splitting

Decision Trees

Finding Best Attribute

15/04/2024

Measures of Node Impurity

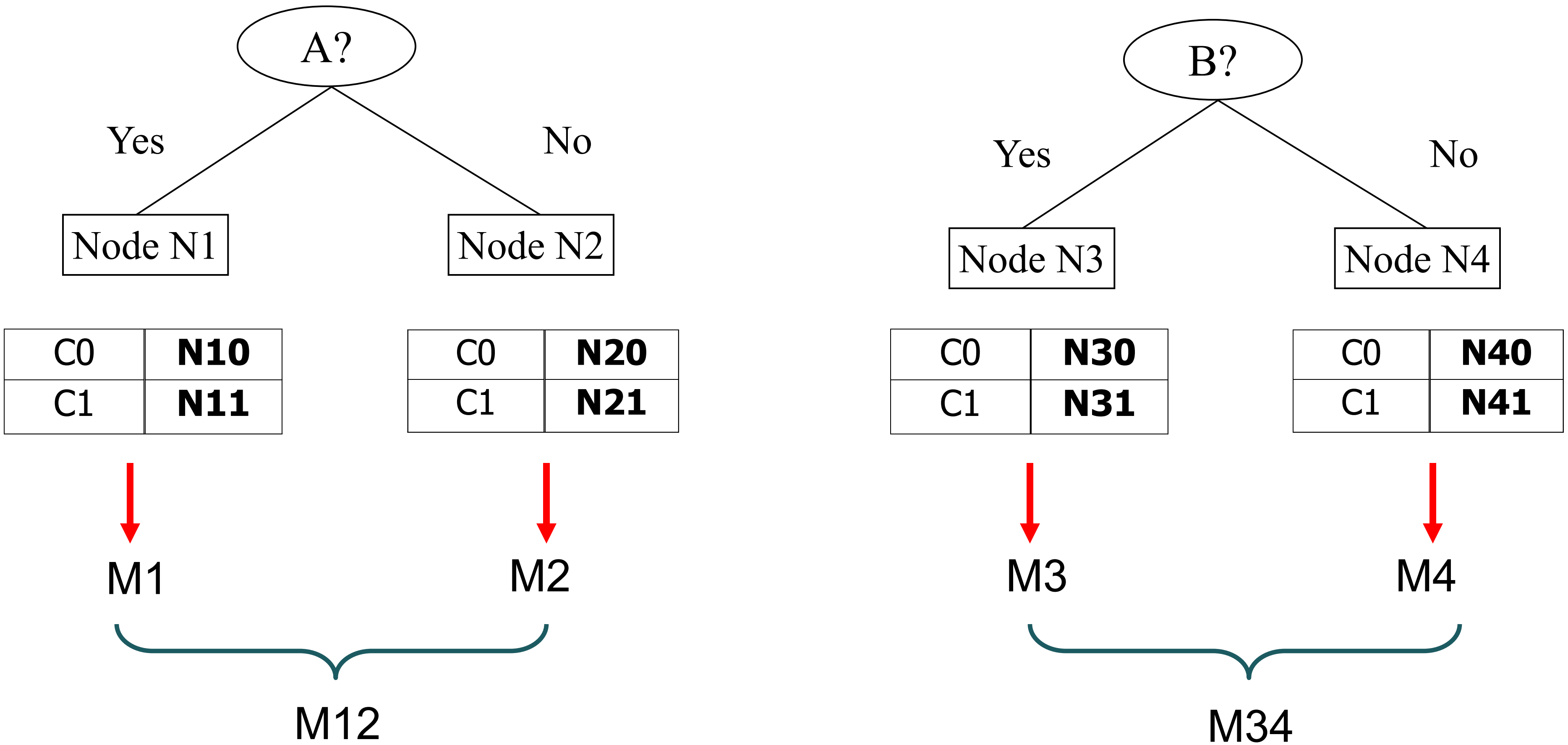
- Gini Index
- Entropy
- Misclassification error

How to Find the Best Split

Before Splitting:

C0	**N00**
C1	**N01**

→ M0



Gain = M0 – M12 vs M0 – M34

Measures of Node Impurity

- Gini Index
- Entropy
- Misclassification error

Alternative Splitting Criteria based on INFO

- Entropy at a given node t:

$$Entropy(t) = -\sum_j p(j \mid t) \log_2 p(j \mid t)$$

  - $p(j \mid t)$ is the relative frequency of class j at node t

- Measures homogeneity of a node

- Entropy of sample S: Average optimal number of bits to encode information about certainty/uncertainty about S

Examples for computing Entropy

$$Entropy(t) = -\sum_j p(j\,|\,t)\log_2 p(j\,|\,t)$$

C1	**0**
C2	**6**

P(C1) = 0/6 = 0   P(C2) = 6/6 = 1

Entropy = – 0 log 0 – 1 log 1 = – 0 – 0 = 0

C1	**1**
C2	**5**

P(C1) = 1/6   P(C2) = 5/6

Entropy = – (1/6) $\log_2$ (1/6) – (5/6) $\log_2$ (1/6) = 0.65

C1	**2**
C2	**4**

P(C1) = 2/6   P(C2) = 4/6

Entropy = – (2/6) $\log_2$ (2/6) – (4/6) $\log_2$ (4/6) = 0.92

Alternative Splitting Criteria based on INFO

- Entropy at a given node t:

$$\boxed{Entropy(t) = -\sum_{j} p(j\,|\,t)\log_2 p(j\,|\,t)}$$

  - *p(j | t)* is the relative frequency of class j at node t

- Measures homogeneity of a node
  - Maximum ($\log n_c$) when records are equally distributed among all classes
    - implying least information
  - Minimum (0.0) when all records belong to one class,
    - implying most information

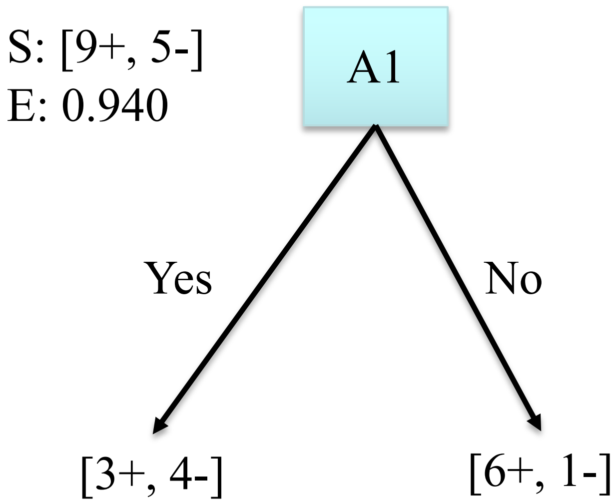
Information Gain

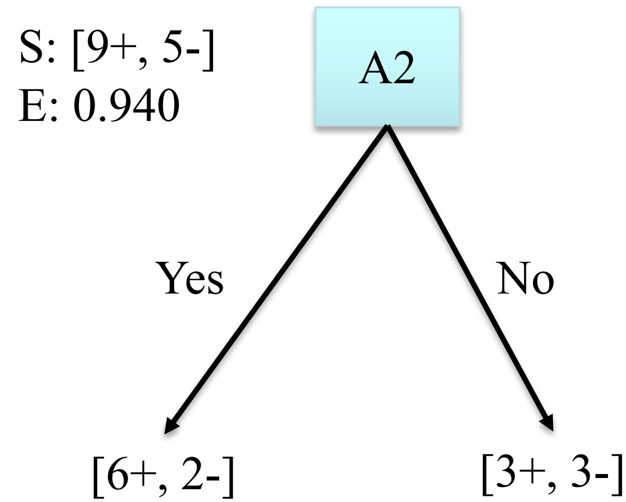
- Measures how well a given attribute separates the training examples according to their target classification

- This measure is used to select among the candidate attributes at each step while growing the tree

- Gain is measure of how much we can reduce uncertainty (Value lies between [0,1])

Information Gain





Entropy(3+,4-) = - (3/7)log(3/7) – (4/7)log(4/7) = 0.985

Entropy(6+,1-) = - (6/7)log(6/7) – (1/7)log(1/7) = 0.592

Gain(S,A1) = 0.940 – (7/14)*0.985 – (7/14)*0.592 = 0.151

Entropy(6+,2-) = - (6/8)log(6/8) – (2/8)log(2/8) = 0.811

Entropy(3+,3-) = - (3/6)log(3/6) – (3/6)log(3/6) = 1.0

Gain(S,A2) = 0.940 – (8/14)*0.811 – (6/14)*1.0 = 0.048

Splitting Based on INFO...

- Information Gain:

$$GAIN_{split} = Entropy(p) - \left(\sum_{i=1}^{k} \frac{n_i}{n} Entropy(i) \right)$$

  - Parent Node p is split into k partitions;
  - n_i is number of records in partition i
- Measures Reduction in Entropy achieved because of the split Choose the split that achieves most reduction (maximizes GAIN)
- Used in ID3 and C4.5

- **Disadvantage:** Tends to prefer splits that result in large number of partitions, each being small but pure

Splitting Based on INFO...

- Gain Ratio:

$$GainRATIO_{split} = \frac{GAIN_{Split}}{SplitINFO}$$

$$SplitINFO = -\sum_{i=1}^{k} \frac{n_i}{n} \log \frac{n_i}{n}$$

  - Parent Node, p is split into k partitions
  - n_i is the number of records in partition i

- Adjusts Information Gain by the entropy of the partitioning (SplitINFO)
  - Higher entropy partitioning (large number of small partitions) is penalized!
- Used in C4.5

- Designed to overcome the disadvantage of Information Gain

Stopping Criteria for Tree Induction

- Stop expanding a node when all the records belong to the same class
- Stop expanding a node when all the records have similar attribute values (if different class values, then usually assign the majority class)
- Early termination, usually to prevent overfitting (to be discussed later)

ID3

ID3(Examples, Target_attribute, Attributes)

- Create a Root node for the tree
- If all examples are positive, Returns single-node tree Root with label +
- If all examples are negative, Returns single-node tree Root with label –
- If Attributes is empty, Returns single-node tree Root, with label = most common value of Target_attribute in Examples

ID3

ID3(Examples, Target_attribute, Attributes)

- Begin
  - A ← Best attribute from Examples
  - The decision attribute for Root ← A
  - For each possible value, v_i, of A,
    - Add a new branch below Root, corresponding to A= v_i
    - Examples_v_i subset of examples with v_i = A
    - If Examples_v_i is empty
      - Add a leaf node with label = most common value of Target_attribute in Examples
    - Else below this new branch add the subtree
      - ID3(Examples_v_i, Target_attribute, Attributes-{A})
- End
- Return Root

Thank You