Decision Trees

Inductive Learning and Classification

Florin Leon

http://eureka.cs.tuiasi.ro/~fleon/curs_wm.htm

Introduction to Machine Learning

- The ability to learn is one of the most important components of intelligent behavior
- A system good in doing a specific job
  - Performs costly computations to solve the problem
  - Does not remember solutions
  - Every time it solves the problem, it performs the same sequence of computations again

What is learning?

- Learning denotes changes in the system that are adaptive in the sense that they enable the system to do the same task or tasks drawn from the same population *more efficiently and more effectively* the next time (Herbert A. Simon)
- A computer program learns if it *improves its performance* at some task through experience (T. Mitchell)

Why should machines learn?

- Learning is essential for unknown environments
  - Everything in the environment cannot be anticipated
  - Designer lacks omniscience
- Learning is an alternative to explicit design
  - Expose the agent to reality rather than trying to tell it about reality
  - Lazy designer

Issues involved in the learning programs

- Learning involves changes in the learner
- Learning involves generalization from experience
  - Performance should improve not only on the repetition of the same task but also on similar tasks in the domain
  - The learner is given a limited experience to acquire knowledge that will generalize correctly to unseen instances of the domain. This is the problem of ***induction***
- Learning algorithms must generalize heuristically – they must select the important aspects of their experience

Inductive concept learning: definitions

- ***Induction*** is reasoning from properties of individuals to properties of sets of individuals
- Given U the universal set of objects (observations), a ***concept*** C is a subset of objects in U, $C \subseteq U$
- Examples:
  - C is a set of all black birds (if U is a set of all birds)
  - C is a set of mammalians (if U is a set of all animals)

Inductive concept learning: definitions

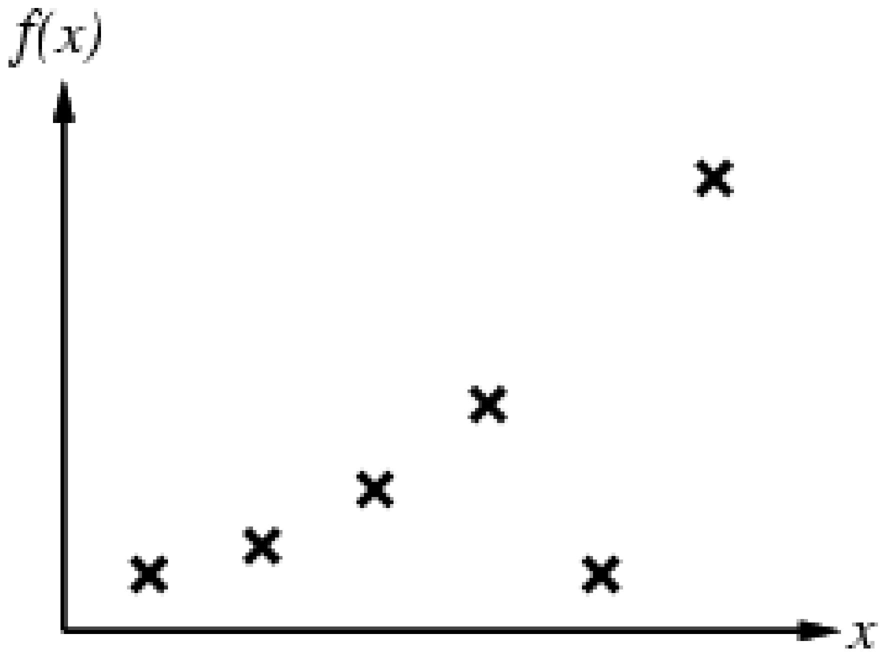
- ***Concept learning***
  - To learn a concept C means to be able to recognize which objects in U belong to C
- ***Inductive concept learning***
  - Given a sample of positive and negative training examples of the concept C
  - Find a procedure (a predictor, a classifier) able to tell, for each $x \in U$, whether $x \in C$

Supervised learning

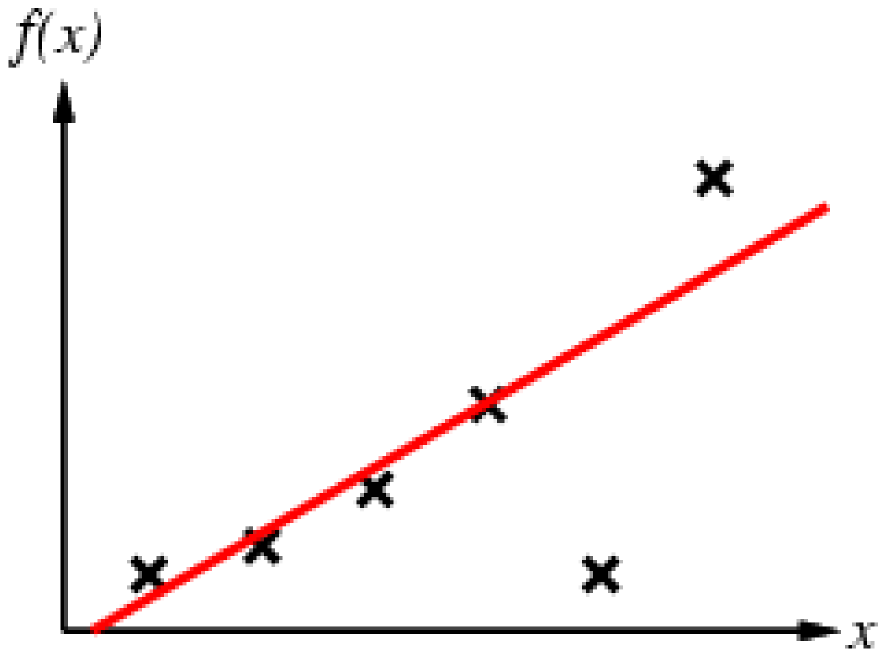
- Example: Inductive Learning
- Simplest form: learn a function from examples
- f is the target function
- An example is a pair $(x, f(x))$
- Problem: find a hypothesis h such that $h \approx f$, given a training set of examples

Inductive Learning

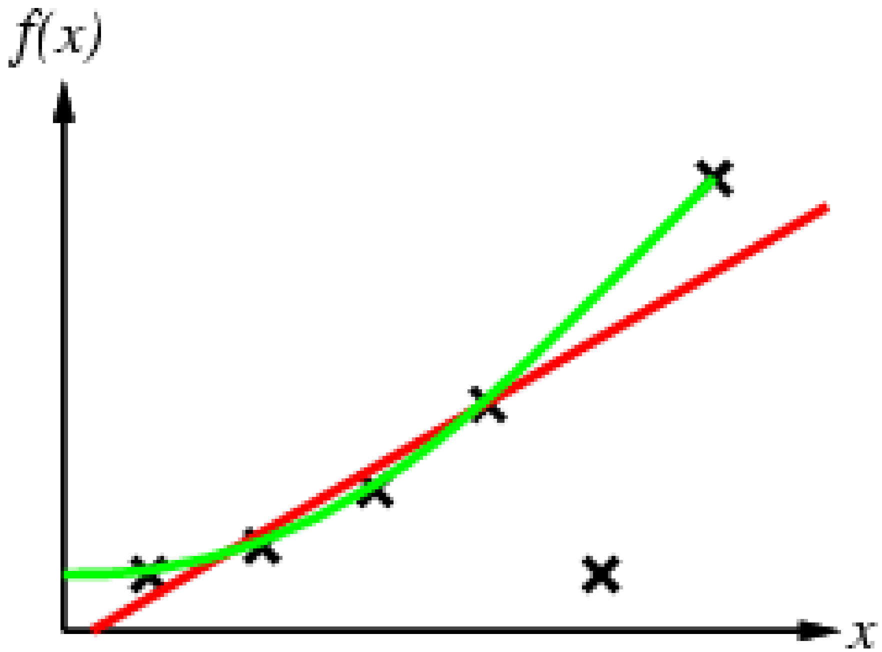
- Construct/adjust h to agree with f on training set (h is consistent if it agrees with f on all examples)
- Example: curve fitting



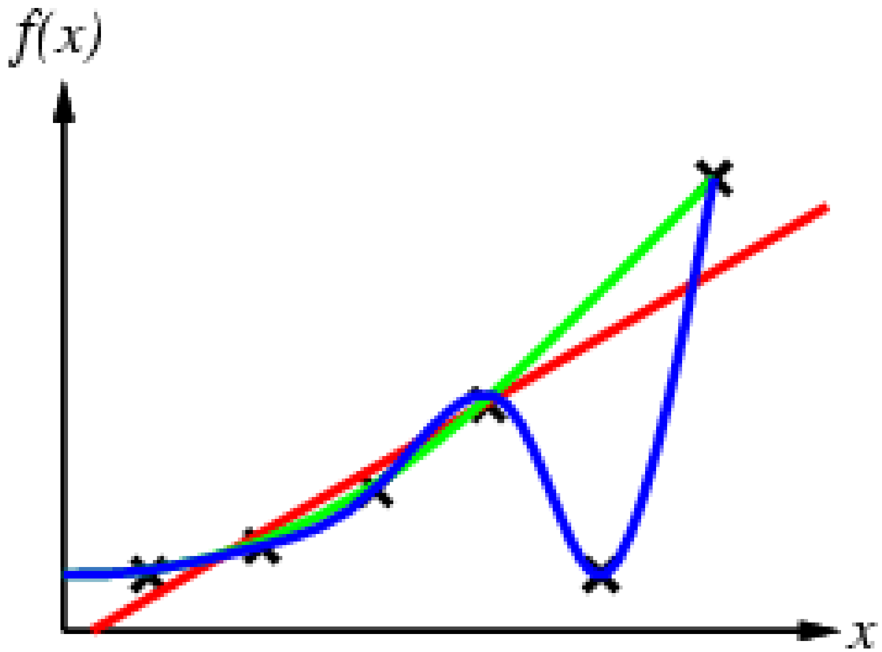
Inductive Learning



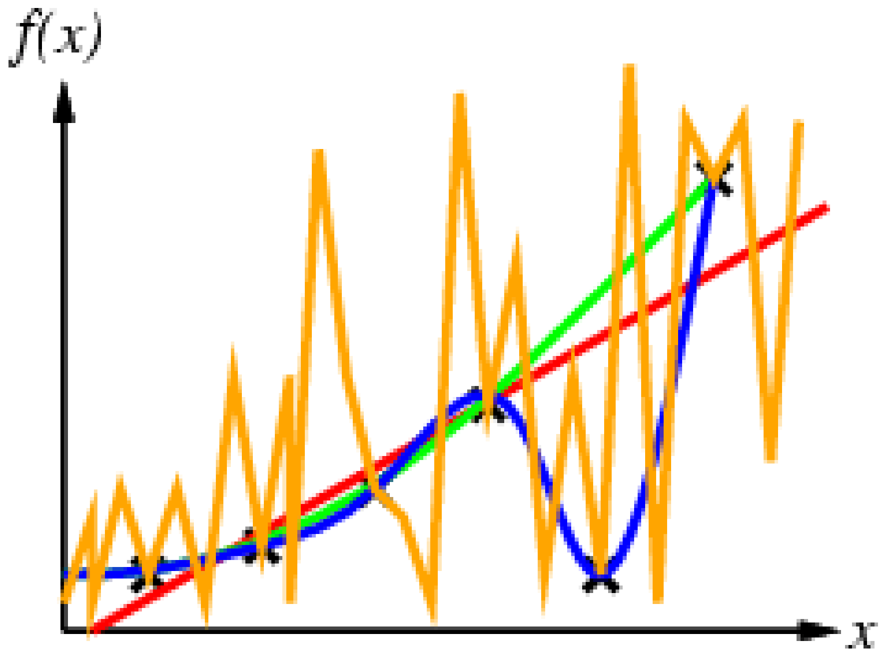
Inductive Learning



Inductive Learning

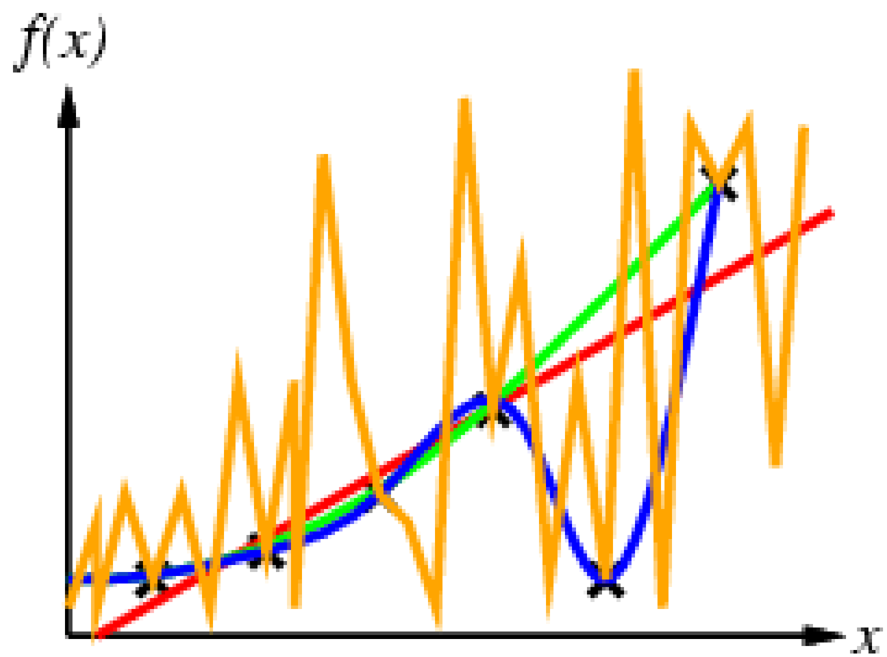


Inductive Learning



Occam's razor

- Prefer the simplest hypothesis consistent with data
- All other things being equal, complex models tend not to generalize as well



Classification vs. regression

- Main idea: learning a relationship between inputs (vector **x**) and an output (y) from data
- The only difference between classification and regression is whether the output variable is discrete or continuous
- Classification estimates the discrete output y, usually known as the "class"
- Regression estimates the function f such that y = f(**x**) with some confidence measure

Classification

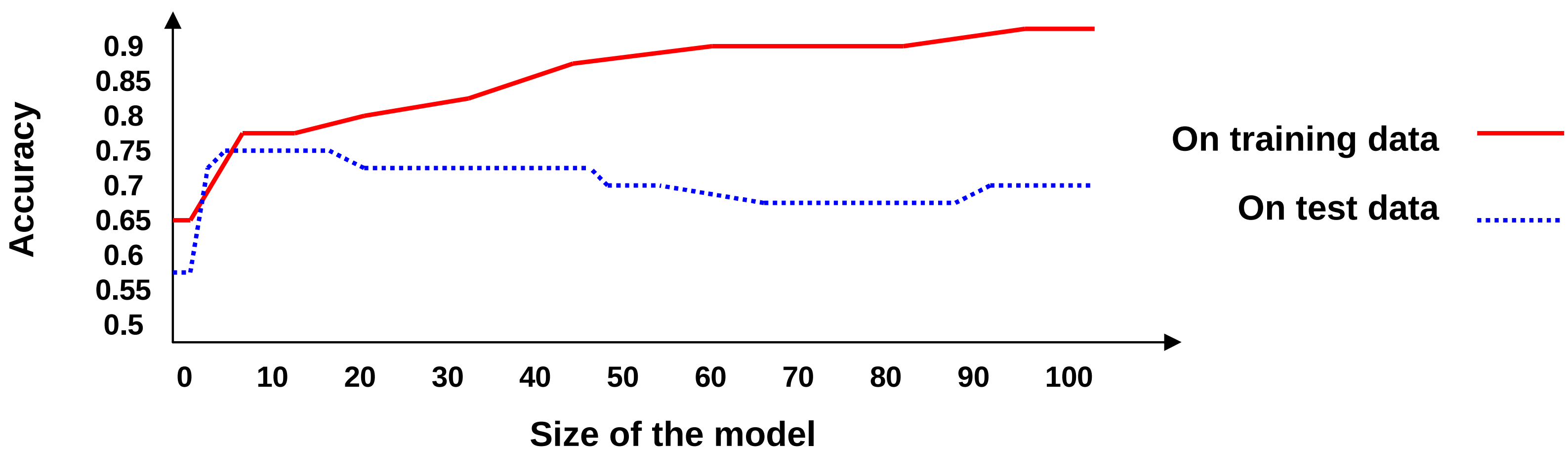
- In order to control a complex environment, the agent must reduce the number and diversity of stimuli
- One strategy is classification (or categorization)
  - Establishing classes that include a group of objects that have some common attributes

Classification: definition

- Given a collection of records *(training set)*
  - Each record contains a set of attributes, one of the attributes is the class
- Classification is finding a model for the class attribute as a function of the values of other attributes
- Goal: *previously unseen* records should be assigned a class as accurately as possible
  - A test set is used to determine the accuracy of the model
  - Usually, the given data set is divided into training and test sets, with training set used to build the model and test set used to validate it

Generalization

- Overfitting is finding overly complex functions to account for noise or irrelevant data
- An overfit model performs well on the training set, but usually has poor generalization capabilities
- Generating a test set:
  - 1/3 – 2/3 split: 2/3 to train, 1/3 to test
  - Cross-validation: s buckets, s-1 to train, s^{th} to test, repeat s times
  - Leave one out: n-1 records to train, n^{th} to test, repeat n times



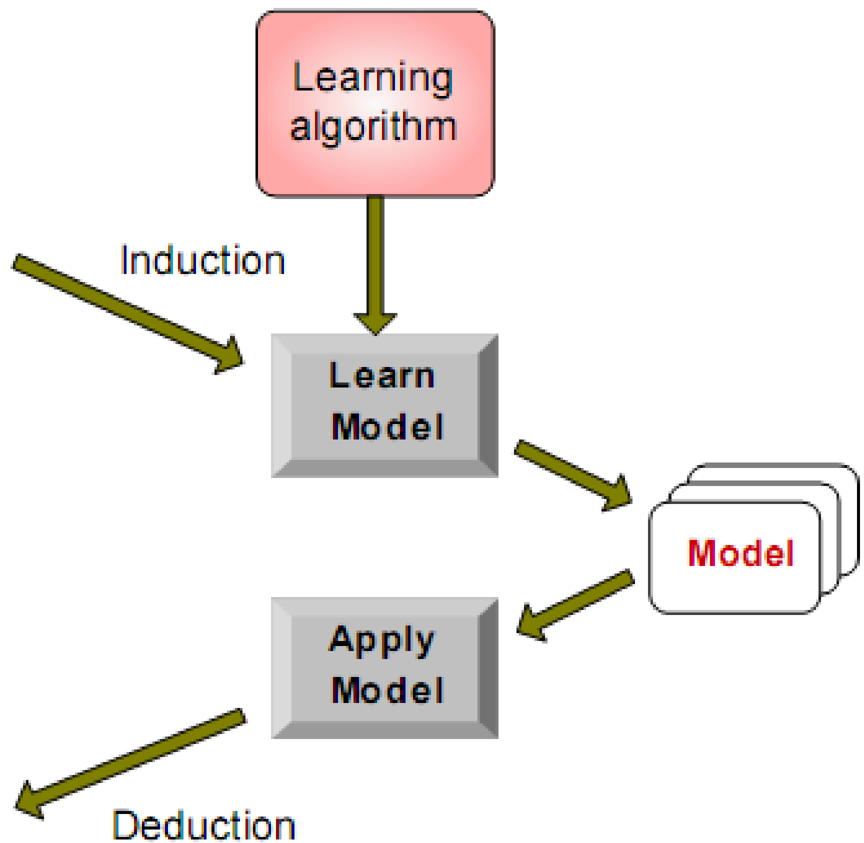
Classification examples

- Predicting tumor cells as benign or malignant
- Classifying credit card transactions as legitimate or fraudulent
- Classifying secondary structures of proteins
- Categorizing news stories as finance, weather, entertainment, sports, etc.

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
11	No	Small	55K	?
12	Yes	Medium	80K	?
13	Yes	Large	110K	?
14	No	Small	95K	?
15	No	Large	67K	?

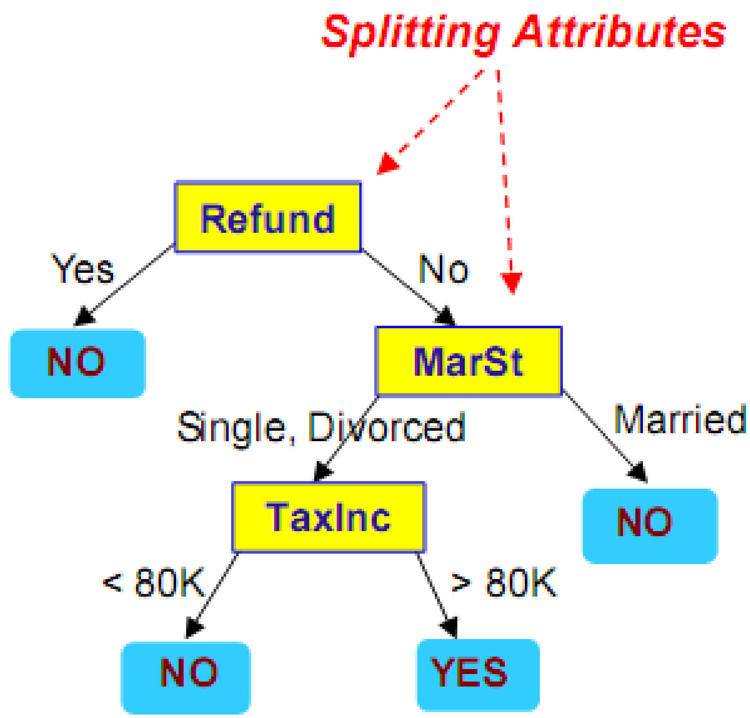
Test Set

Decision 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

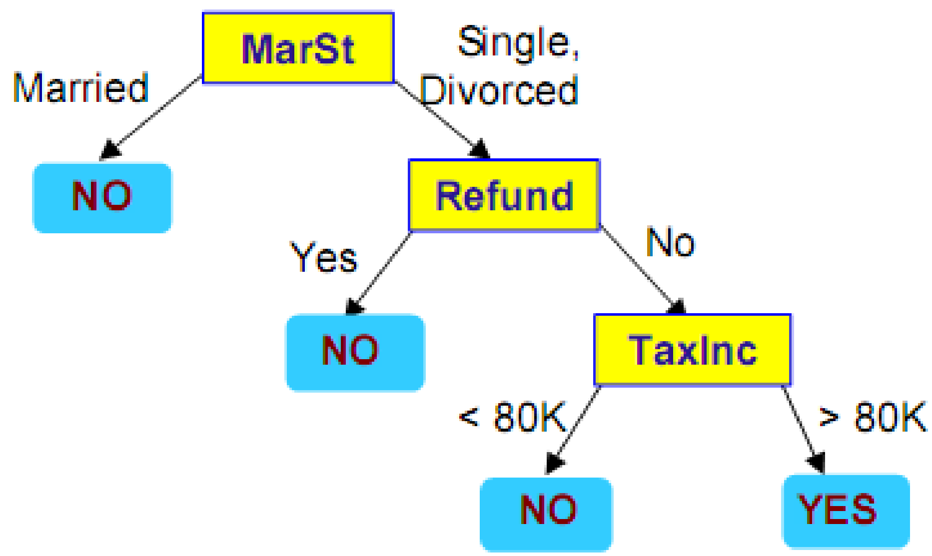


Model: Decision Tree

Another possible tree

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

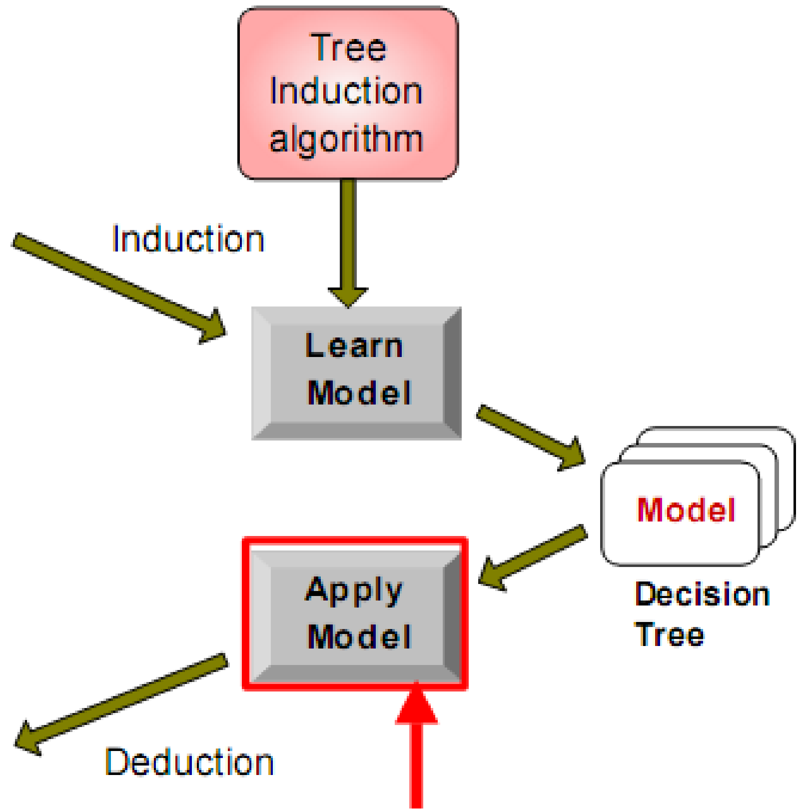


There could be more than one tree that fits the same data!

DT 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

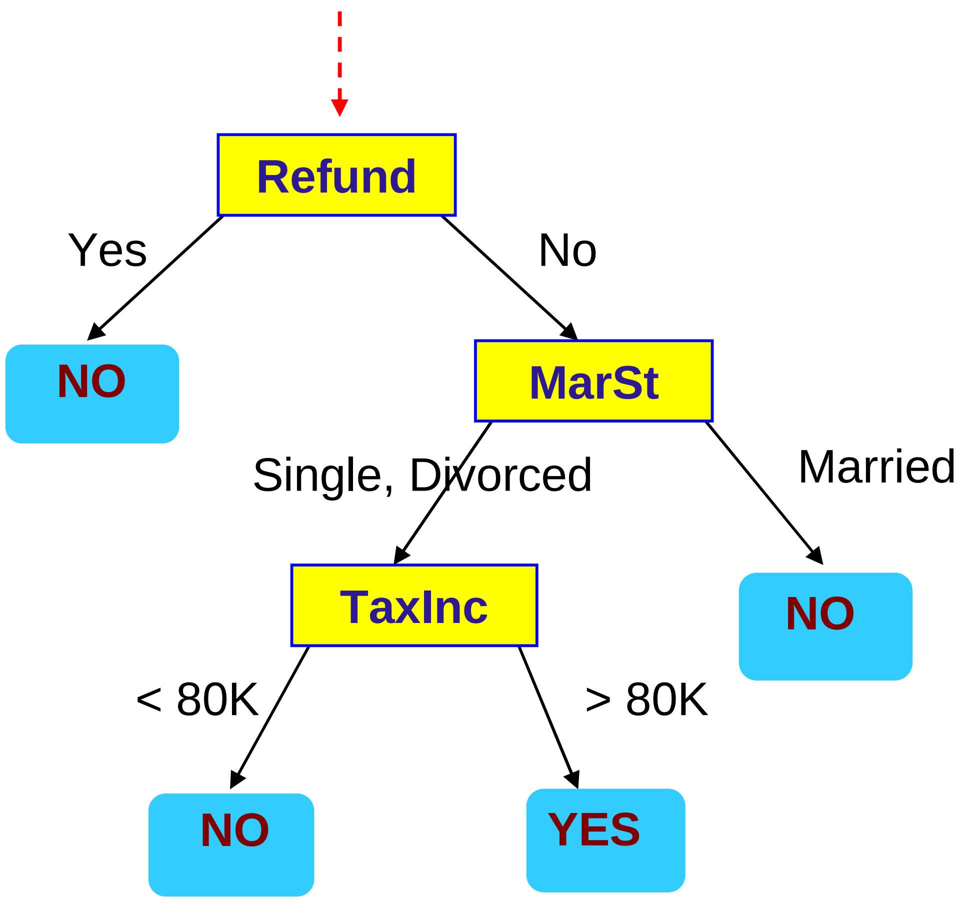


Tid	Attrib1	Attrib2	Attrib3	Class
11	No	Small	55K	?
12	Yes	Medium	80K	?
13	Yes	Large	110K	?
14	No	Small	95K	?
15	No	Large	67K	?

Test Set

Apply model to test data

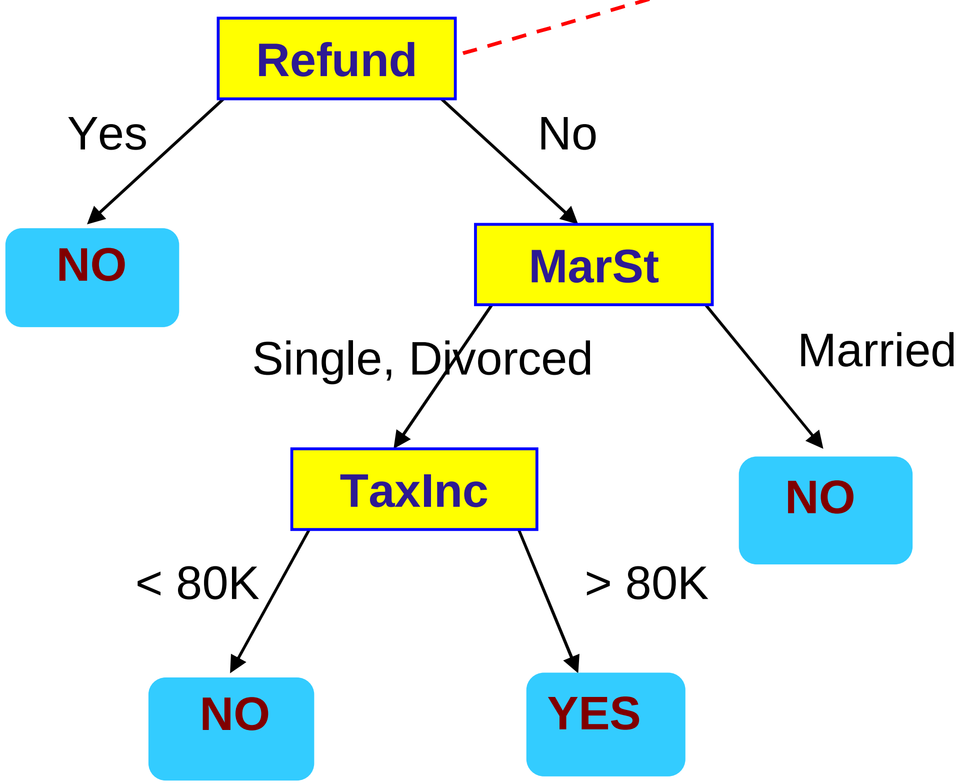
Start from the root of tree



Test Data

Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	**?**

Apply model to test data



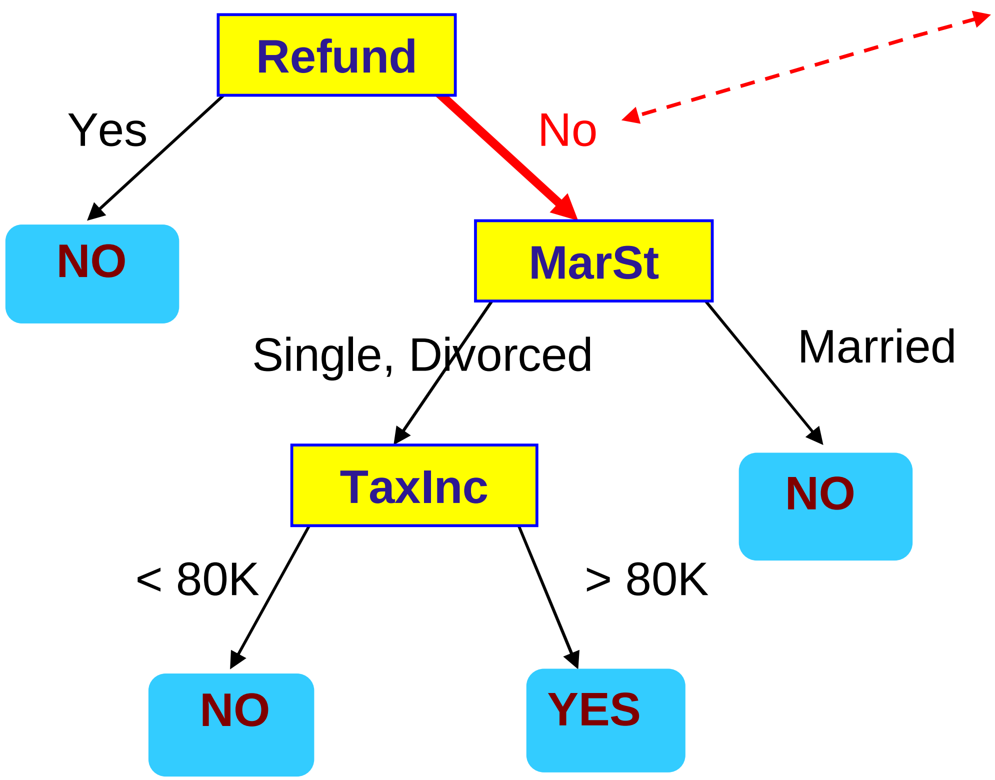
Test Data

Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?

Apply model to test data

Test Data

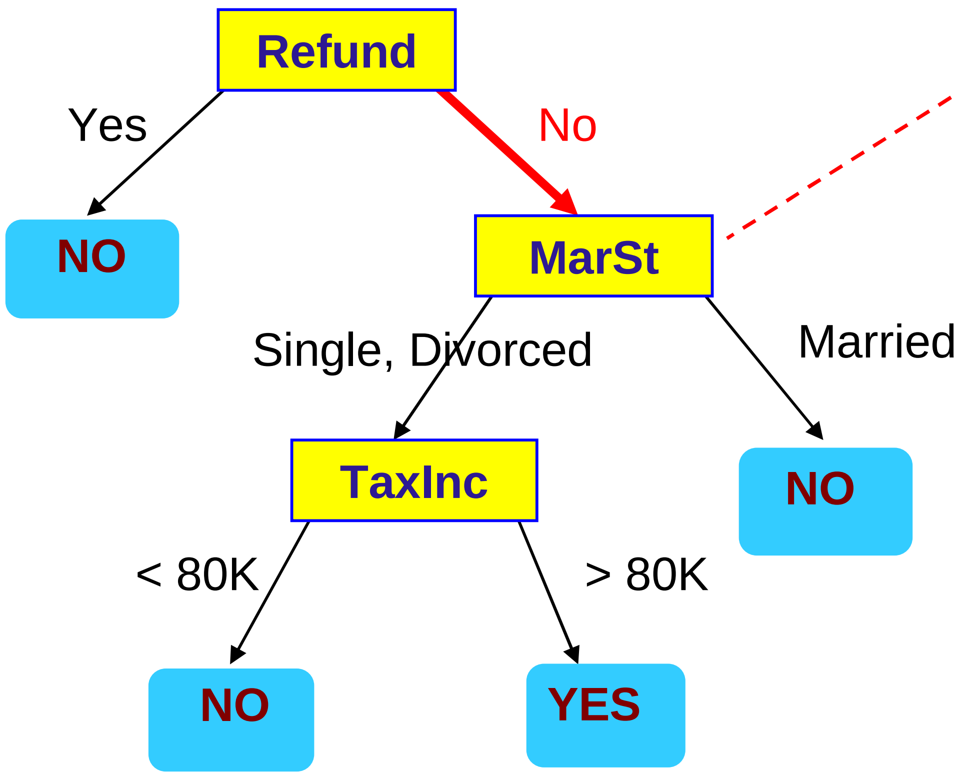
Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?



Apply model to test data

Test Data

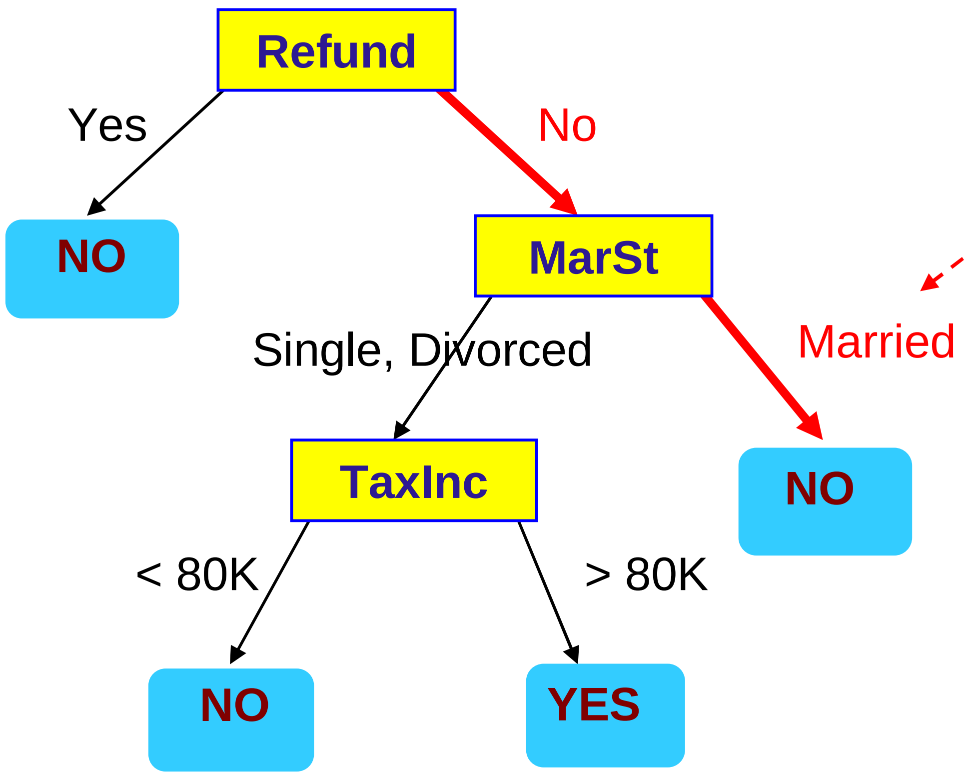
Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?



Apply model to test data

Test Data

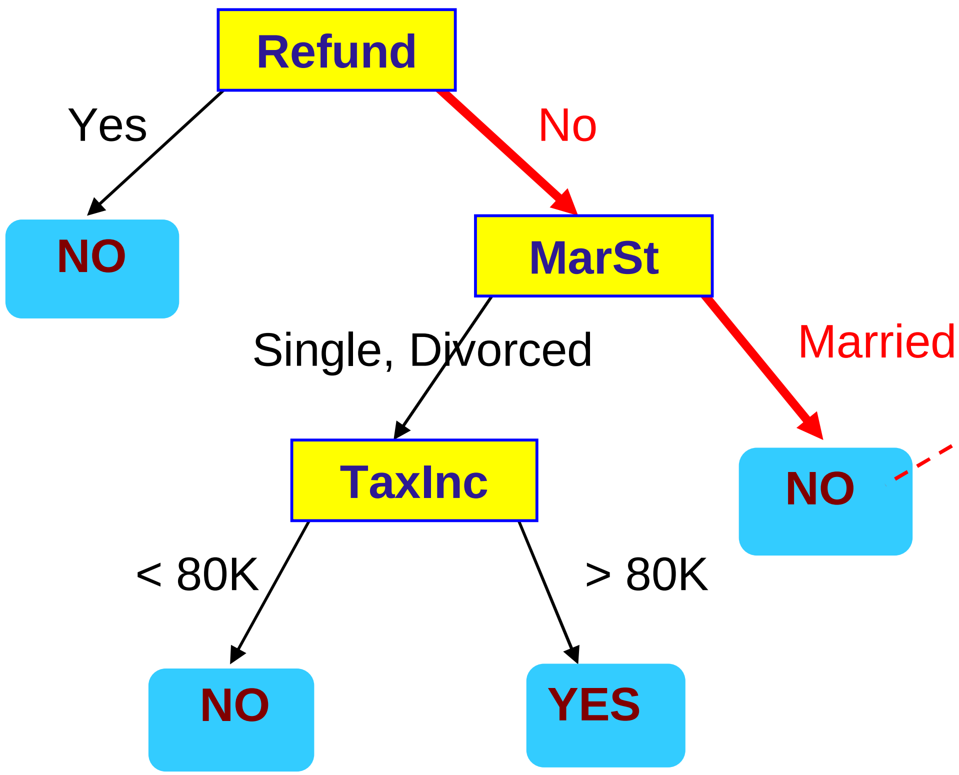
Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?



Apply model to test data

Test Data

Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?

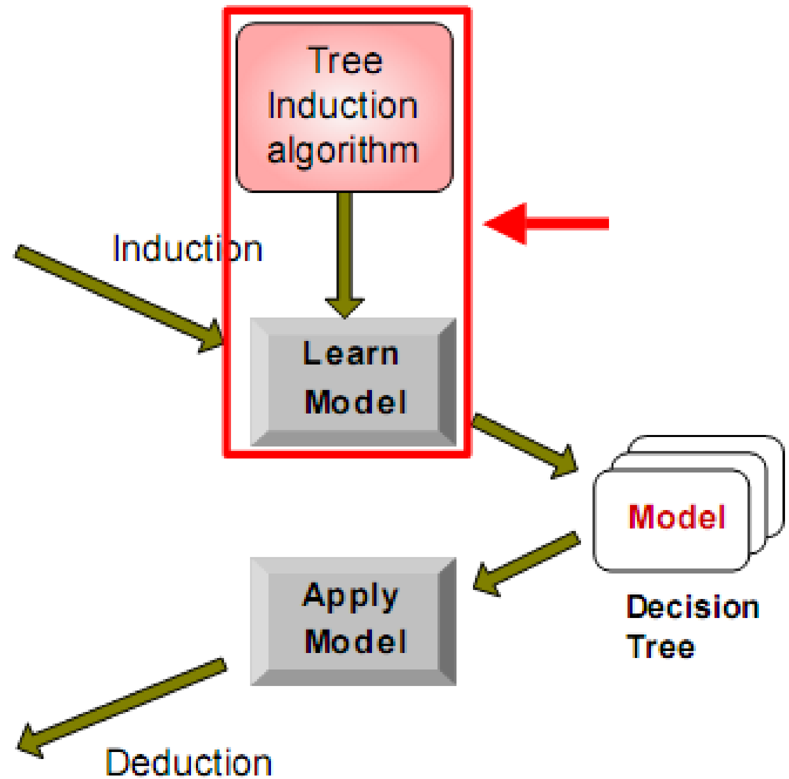


Assign Cheat to "No"

DT 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



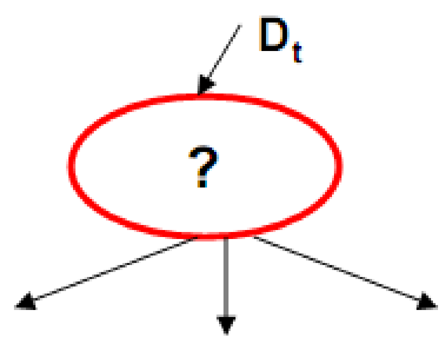
Tid	Attrib1	Attrib2	Attrib3	Class
11	No	Small	55K	?
12	Yes	Medium	80K	?
13	Yes	Large	110K	?
14	No	Small	95K	?
15	No	Large	67K	?

Test Set

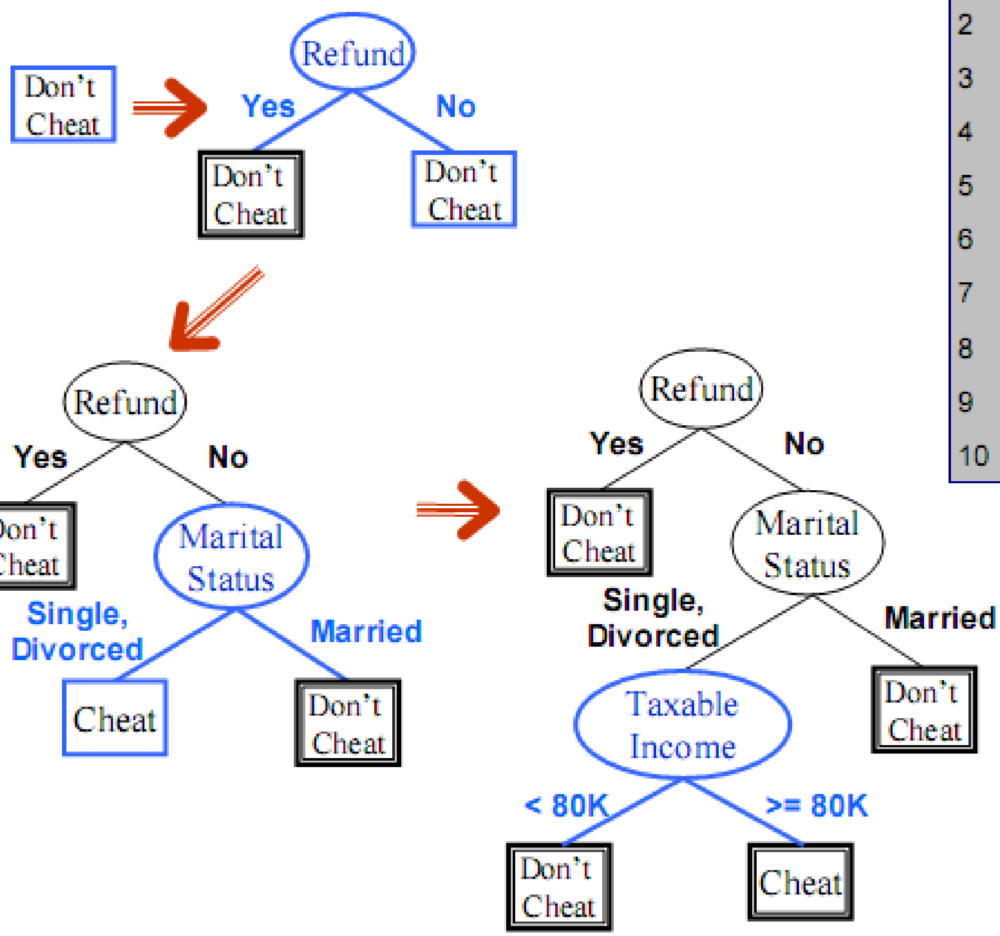
Decision tree induction

- Let D_t be the set of training records that reach a node t
- General procedure (Hunt's algorithm):
  - 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 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	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



Hunt's algorithm



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

Decision tree induction

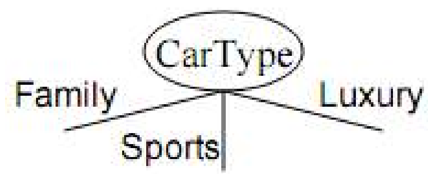
- Greedy strategy
  - Split the records based on an attribute test that optimizes certain criterion
- Determine how to split the records
  - How to specify the attribute test condition?
  - How to determine the best split?

How to specify test condition

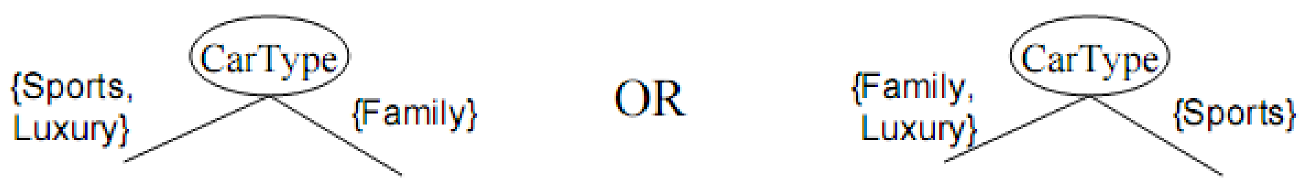
- Depends on attribute types
  - Nominal
  - Ordinal
  - Continuous
- Depends on the number of ways to split
  - 2-way split
  - Multi-way split

Nominal attributes

- Multi-way split
  - Using as many partitions as distinct values

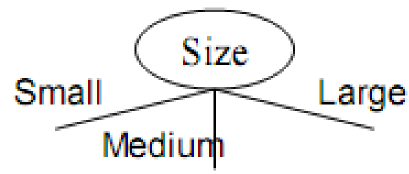


- Binary split
  - Dividing values into two subsets
  - It needs to find the optimal partitioning

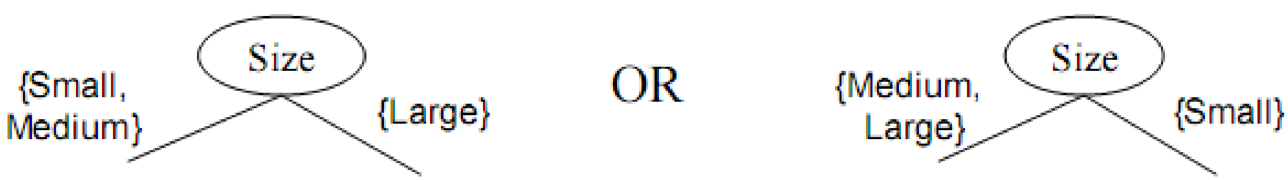


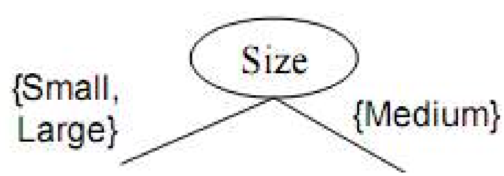
Ordinal attributes

- Multi-way split
  - Using as many partitions as distinct values



- Binary split
  - Dividing values into two subsets
  - It needs to find the optimal partitioning



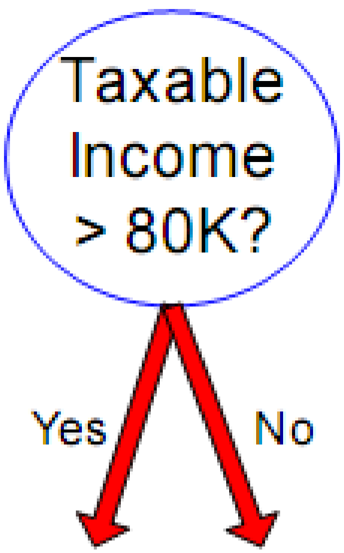


?

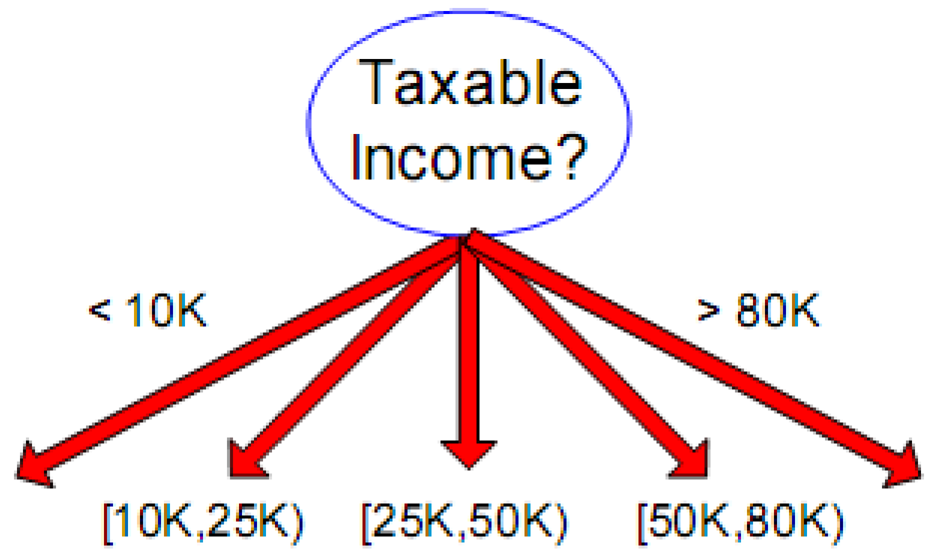
Continuous attributes

- Discretization is used to form ordinal categorical attributes
  - Equal interval
  - Equal frequency
  - Clustering
- Binary decision: $(A \leq v)$ or $(A > v)$
  - Considers all possible splits and finds the best one
  - Usually is more computationally intensive

Splitting based on continuous attributes



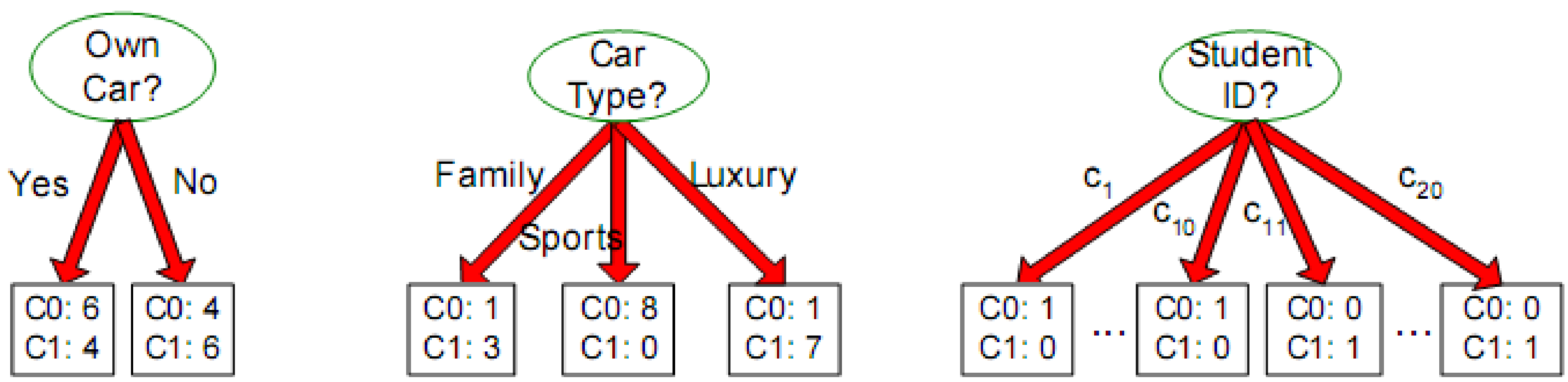
(i) Binary split



(ii) Multi-way split

How to determine the best split

- Before splitting: 10 records of class 0, and 10 records of class 1
- Which test condition is the best?



How to determine the best split

- Greedy approach: nodes with ***homogeneous*** class distributions are preferred
- It needs a measure of node impurity

C0: 5 C1: 5	C0: 9 C1: 1
Non-homogeneous, **High degree of impurity**	**Homogeneous,** **Low degree of impurity**

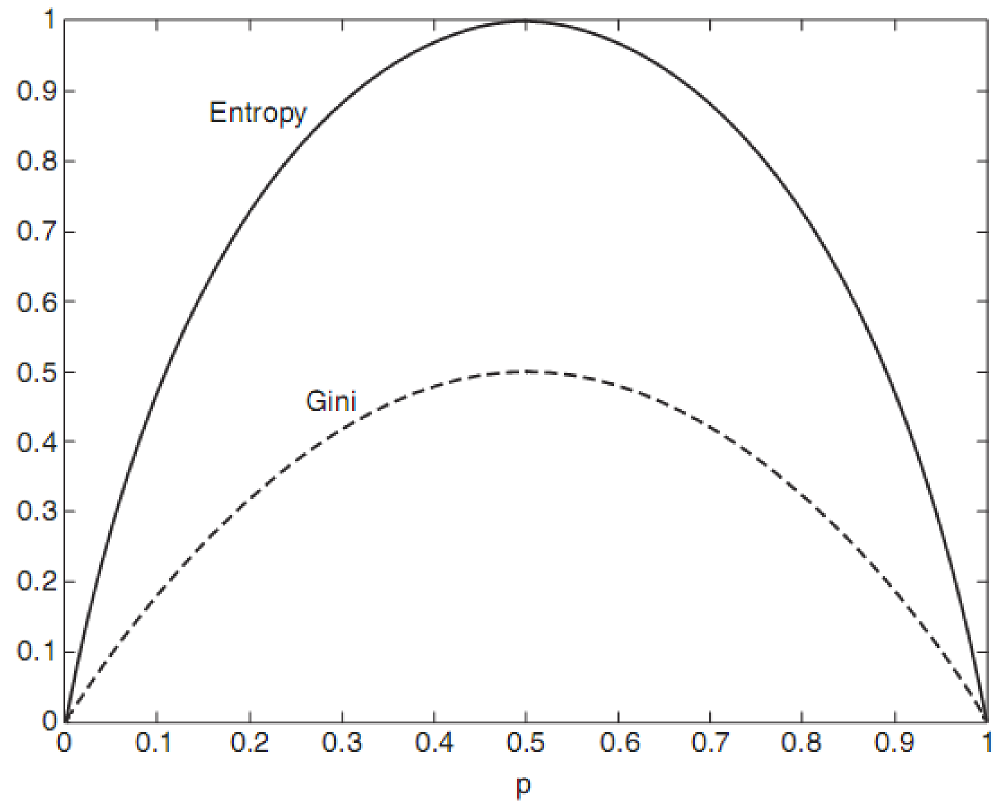
Impurity measures

$$\text{Entropy}(t) = -\sum_{i=0}^{c-1} p(i|t) \log_2 p(i|t)$$

$$\text{Gini}(t) = 1 - \sum_{i=0}^{c-1} [p(i|t)]^2$$

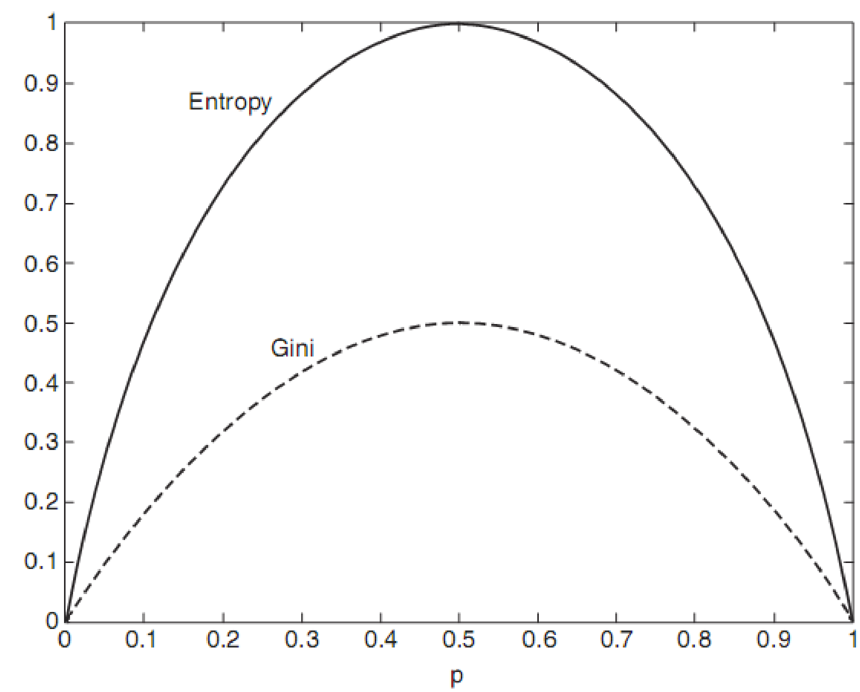
Convention: $0 \cdot \log_2 0 = 0$

Graph for a binary problem:



Impurity measures

- ***Maximum*** when records are equally distributed among all classes, implying the ***least*** interesting information
- ***Minimum*** (0) when all records belong to one class, implying the ***most*** interesting information



Examples

$$\text{Entropy}(t) = -\sum_{i=0}^{c-1} p(i|t)\log_2 p(i|t)$$

$$\text{Gini}(t) = 1 - \sum_{i=0}^{c-1} [p(i|t)]^2$$

Node N_1	Count
Class=0	0
Class=1	6

$$\text{Gini} = 1 - (0/6)^2 - (6/6)^2 = 0$$
$$\text{Entropy} = -(0/6)\log_2(0/6) - (6/6)\log_2(6/6) = 0$$

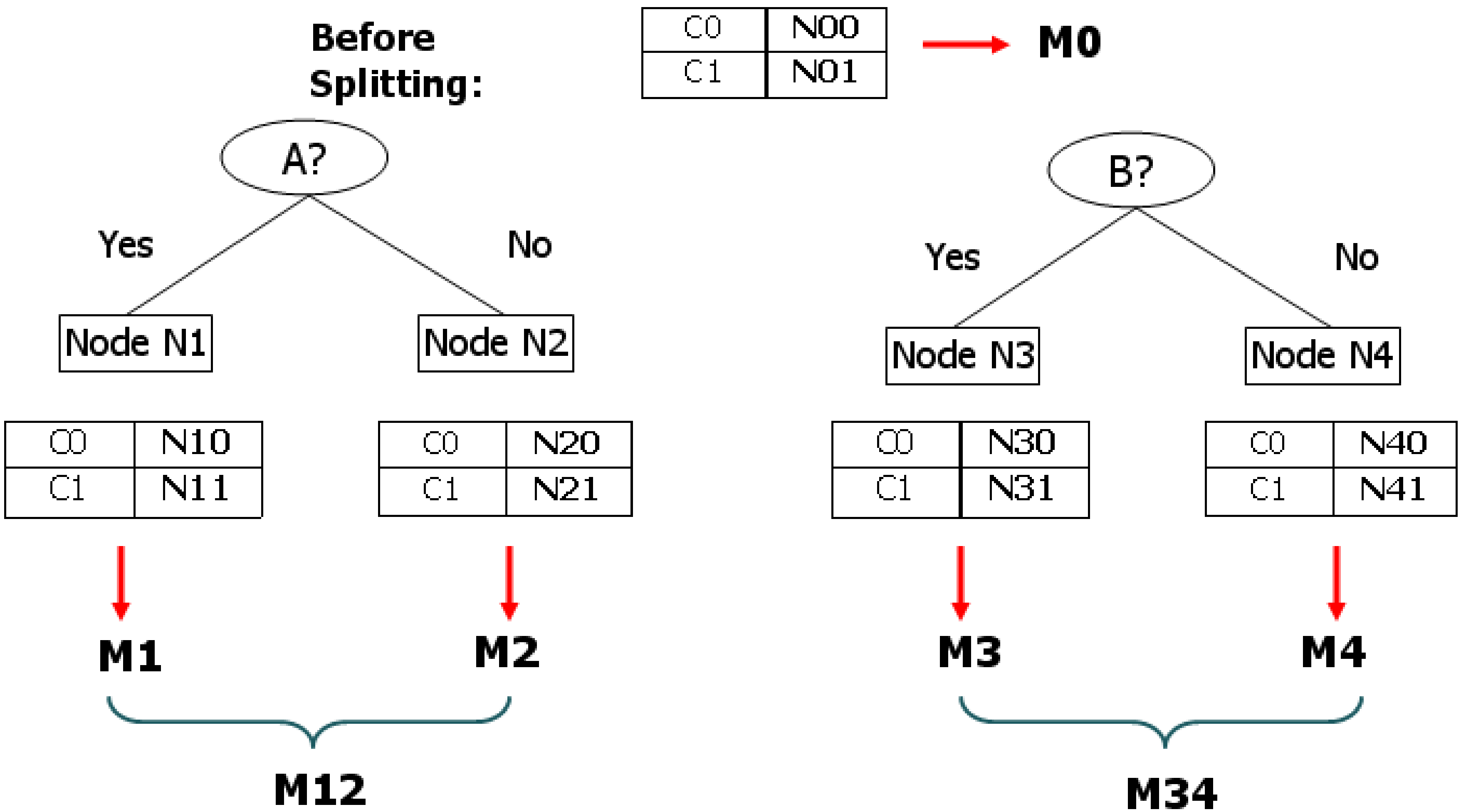
Node N_2	Count
Class=0	1
Class=1	5

$$\text{Gini} = 1 - (1/6)^2 - (5/6)^2 = 0.278$$
$$\text{Entropy} = -(1/6)\log_2(1/6) - (5/6)\log_2(5/6) = 0.650$$

Node N_3	Count
Class=0	3
Class=1	3

$$\text{Gini} = 1 - (3/6)^2 - (3/6)^2 = 0.5$$
$$\text{Entropy} = -(3/6)\log_2(3/6) - (3/6)\log_2(3/6) = 1$$

How to find the best split



Gain = M0 – M12 vs M0 – M34

Splitting

- When a node p is split into k partitions (children), the quality of the split is computed as:

$$GINI_{split} = \sum_{i=1}^{k} \frac{n_i}{n} GINI(i)$$

where: n_i = number of records at child i

n = number of records at node p

- Similar formula for entropy ($p \cdot \log_2 p$)

Information gain

- The "goodness" of a split is determined by the increase in the homogeneity of the resulting subsets
- $\Delta = I(\text{parent}) - \text{sum}_j(N(v_j) / N * I(v_j))$
  - v_j are the resulting partitions (children)
  - N = the number of records in the parent node
  - $N(v_j)$ = the number of records in the child node v_j
- Since I(parent) is the same for all children, a child is selected with the minimum value for $\text{sum}_j(N(v_j) / N * I(v_j))$
- $I(\cdot)$ function can be entropy, Gini index, or other impurity measure

Example: DT induction

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

- Compute information gain for each attribute (Refund, Status, and Income)
- Refund
  - Refund = Yes → 3 records
    - Cheat = Yes → 0
    - Cheat = No → 3
      - Gini = 0
  - Refund = No → 7 records
    - Cheat = Yes → 3
    - Cheat = No → 4
      - Gini = $1 - (3/7)^2 - (4/7)^2 = 0.49$
  - $Gini_{Refund} = (3/10) * 0 + (7/10) * 0.49 = 0.343$

Example: DT induction

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

- Status
  - Status = Divorced → 2 records
    - Cheat = Yes → 1
    - Cheat = No → 1
      - Gini = $1 - (1/2)^2 - (1/2)^2 = 0.5$
  - Status = Married → 4 records
    - Cheat = Yes → 0
    - Cheat = No → 4
      - Gini = $1 - (0/4)^2 - (4/4)^2 = 0$
  - Status = Single → 4 records
    - Cheat = Yes → 2
    - Cheat = No → 2
      - Gini = $1 - (2/4)^2 - (2/4)^2 = 0.5$
  - $Gini_{Status} = (2/10) * 0.5 + (4/10) * 0 + (4/10) * 0.5 = 0.3$
  - Same value if we consider 2 classes {Married} and {Divorced, Single} → optimization problem

DT induction: continuous attributes

- For efficient computation, for each attribute
  - Sort the attribute on values
  - Linearly scan these values, each time updating the count matrix and computing Gini index
  - Choose the split position that has the least Gini index

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

Split Positions →	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		***0.300***		0.343		0.375		0.400		0.420	

DT induction: continuous attributes

- Optimization: compute only for splits when the class value changes
  - 2 candidate splits instead of 11

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

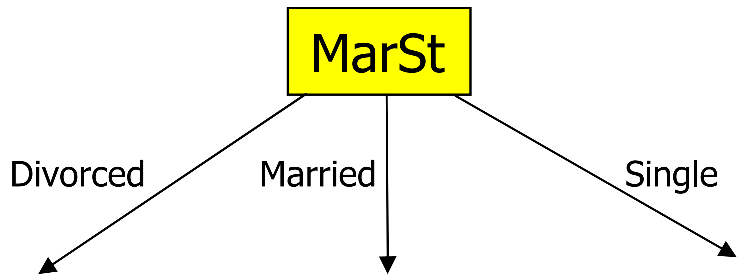
Split Positions	55		65		72		80		87		92		97		110		122		172		230	
	<=	>	<=	>	<=	>	<=	>	<=	>	<=	>	<=	>	<=	>	<=	>	<=	>	<=	>
Yes	0	3	0	3	0	3	0	3	1	2	2	1	3	0	3	0	3	0	3	0	3	0
No	0	7	1	6	2	5	3	4	3	4	3	4	3	4	4	3	5	2	6	1	7	0
Gini	0.420		0.400		0.375		0.343		0.417		0.400		*0.300*		0.343		0.375		0.400		0.420	

First split

- $Gini_{Refund} = 0.343$
- $Gini_{Status} = 0.3$
- $Gini_{Income} = 0.3$
- Splits on both “Status” and “Income” are equally possible
  - But the results can be very different

Recursive procedure

- Let’s consider the “Status” as the first split



Tid	Refund		Taxable Income	Cheat
5	No		95K	Yes
7	Yes		220K	No

Tid	Refund		Taxable Income	Cheat
2	No		100K	No
4	Yes		120K	No
6	No		60K	No
9	No		75K	No

Tid	Refund		Taxable Income	Cheat
1	Yes		125K	No
3	No		70K	No
8	No		85K	Yes
10	No		90K	Yes

Second phase

Tid	Refund		Taxable Income	Cheat
5	No		95K	Yes
7	Yes		220K	No

- Remaining attributes
  - Refund, Income
- Status = Divorced
  - Refund = No → Cheat = Yes
  - Refund = Yes → Cheat = No
  - Gini = 0, split on Refund

Second phase

Tid	Refund		Taxable Income	Cheat
2	No		100K	No
4	Yes		120K	No
6	No		60K	No
9	No		75K	No

- Status = Married → Cheat = No
- No further split necessary

Second phase

Tid	Refund		Taxable Income	Cheat
1	Yes		125K	No
3	No		70K	No
8	No		85K	Yes
10	No		90K	Yes

- Status = Single
  - Refund = Yes → 1 record
    - Cheat = Yes → 0
    - Cheat = No → 1
      - Gini = 0
  - Refund = No → 3 records
    - Cheat = Yes → 2
    - Cheat = No → 1
      - Gini = $1 - (2/3)^2 - (1/3)^2 = 0.444$
  - $Gini_{Refund} = 0 + (3/4) * 0.444 = 0.333$

Second phase

Tid	Refund		Taxable Income	Cheat
1	Yes		125K	No
3	No		70K	No
8	No		85K	Yes
10	No		90K	Yes

- Status = Single

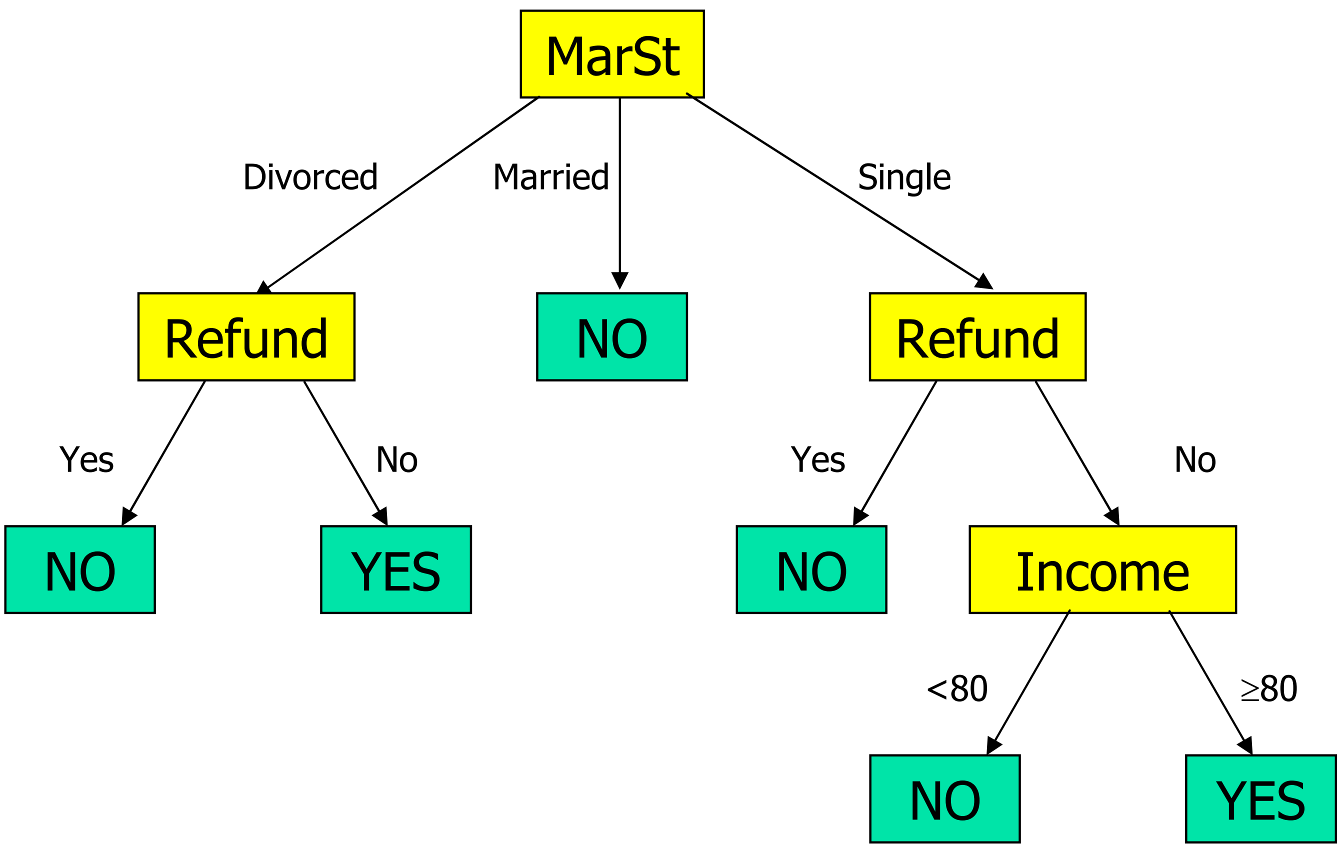
Cheat	No		Yes		Yes		No			
	Taxable Income									
	70		85		90		125			
	65		80		87		110		130	
	<=	>	<=	>	<=	>	<=	>	<=	>
Yes	0	2	0	2			2	0	2	0
No	0	2	1	1			1	1	2	0
Gini	0.5		0.333				0.333		0.5	

Class value unchanged

Second split

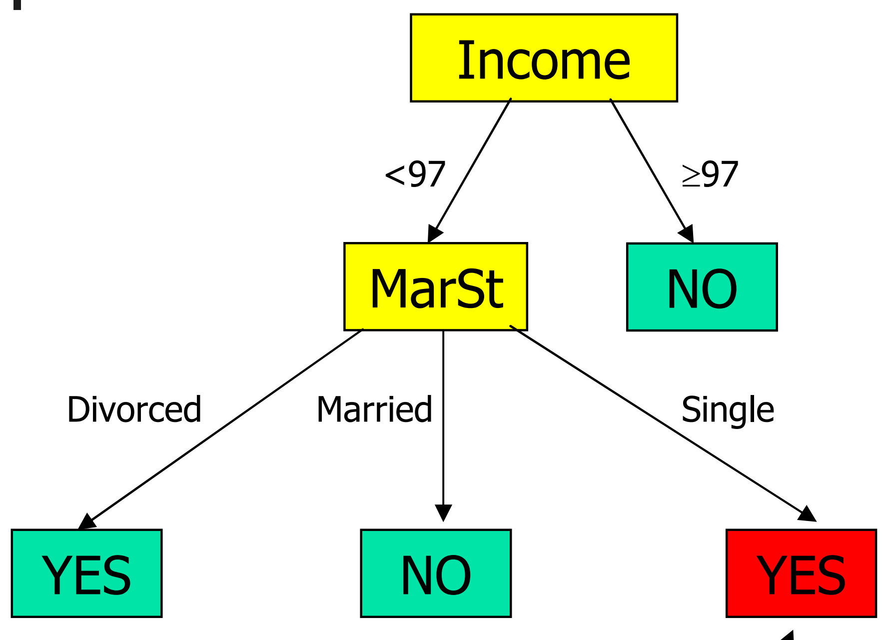
- $Gini_{Refund} = 0.333$
- $Gini_{Income} = 0.333$
- Splits on both “Refund” and “Income” are equally possible
  - Let’s split on “Refund”

Final tree



Tid			Taxable Income	Cheat
3			70K	No
8			85K	Yes
10			90K	Yes

Alternative decision



Tid	Refund		Cheat
3	No		No
8	No		Yes
10	No		Yes

The decision tree has 1 error on the training set

Conclusions

- A DT is usually inexpensive to build
  - Although it requires some computations
- Fast at classifying unknown records
- Easy to interpret
  - Especially for small-sized trees
- A DT can be interpreted as a set of rules
  - E.g. "If *Marital Status* is *Divorced* and *Refund* is *Yes* then *Cheat* is *No*"