Fall 2004, CIS, Temple University

CIS527: Data Warehousing, Filtering, and Mining

Lecture 7

☐ Decision Trees

Lecture slides taken from:

– Vipin Kumar (http://www-users.cs.umn.edu/~kumar/csci5980/index.html)

Classification: Definition

- Given a collection of records (*training set*)
  - Each record contains a set of *attributes*, one of the attributes is the *class*.

- Find a *model* for class attribute as a function of the values of other attributes.

- Goal: <u>previously unseen</u> 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.
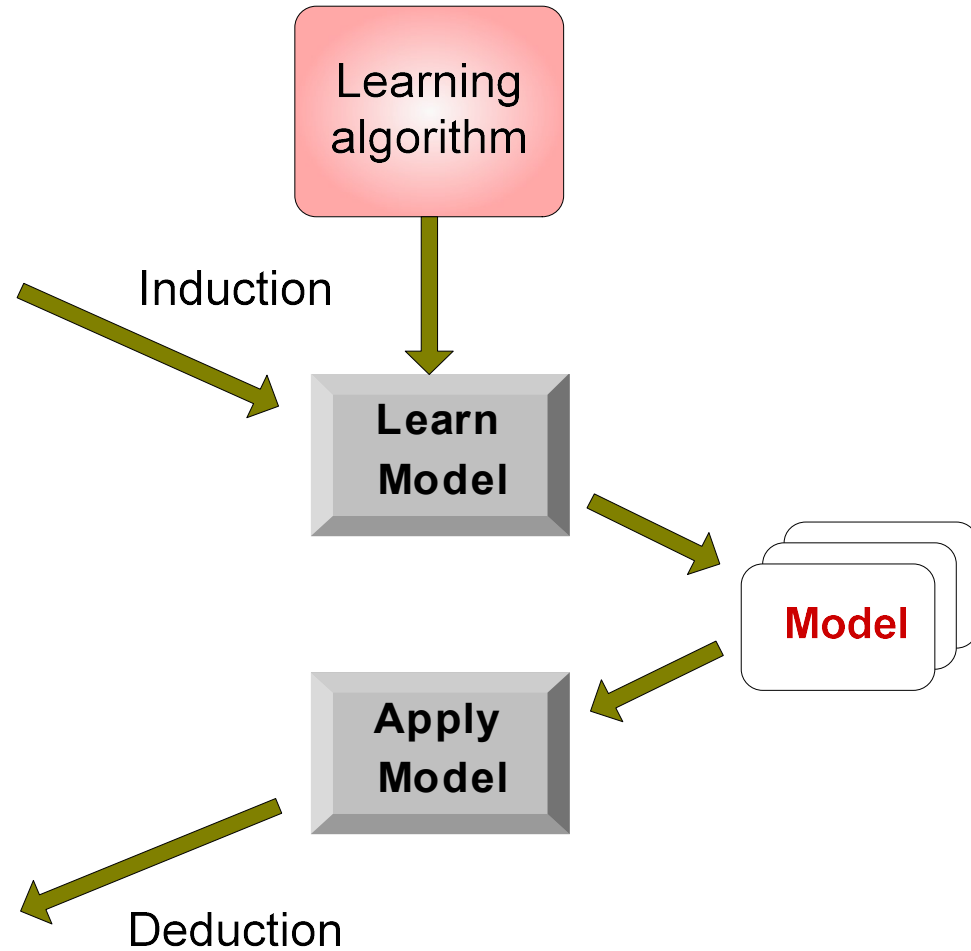
Illustrating Classification Task

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

Learning algorithm

Induction

Learn Model

Model

Apply Model

| 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

Deduction

Examples of Classification Task

- Predicting tumor cells as benign or malignant

- Classifying credit card transactions as legitimate or fraudulent

- Classifying secondary structures of protein as alpha-helix, beta-sheet, or random coil
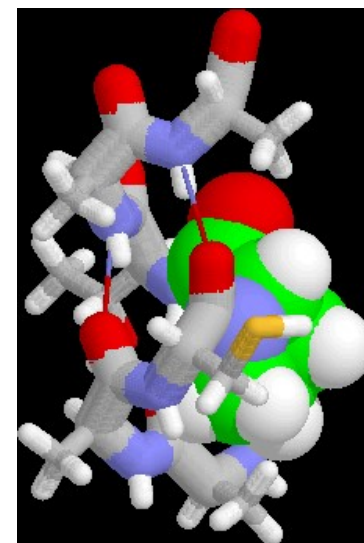
- Categorizing news stories as finance, weather, entertainment, sports, etc
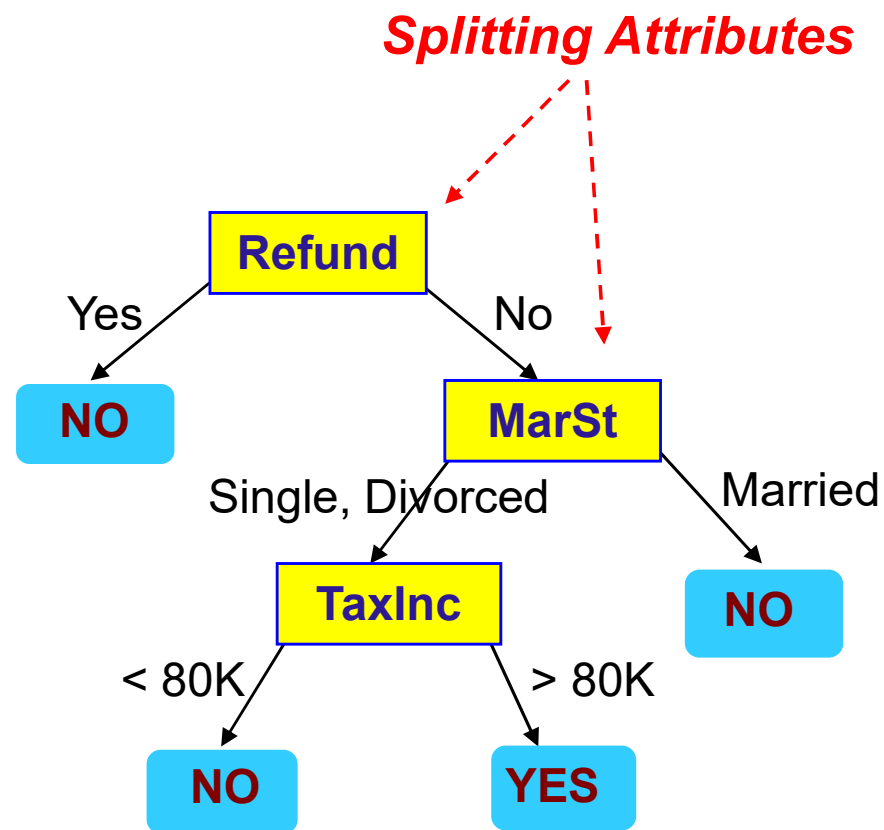
Classification Techniques

- Decision Tree based Methods

- Rule-based Methods

- Memory based reasoning

- Neural Networks

- Naïve Bayes and Bayesian Belief Networks

- Support Vector Machines

Example of a Decision 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

categorical   *categorical*   *continuous*   *class*

Training Data

Splitting Attributes

Refund

Yes → **NO**

No → **MarSt**

Single, Divorced → **TaxInc**

Married → **NO**

< 80K → **NO**

> 80K → **YES**

Model: Decision Tree

Another Example of Decision 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

categorical *categorical* *continuous* *class*

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

Decision Tree 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

Tree Induction algorithm

Induction

Learn Model

Model

Decision Tree

Apply Model

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

Deduction

Apply Model to Test Data

Start from the root of tree.

Test Data

Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?

Refund
- Yes → **NO**
- No → **MarSt**
  - Single, Divorced → **TaxInc**
    - < 80K → **NO**
    - > 80K → **YES**
  - Married → **NO**
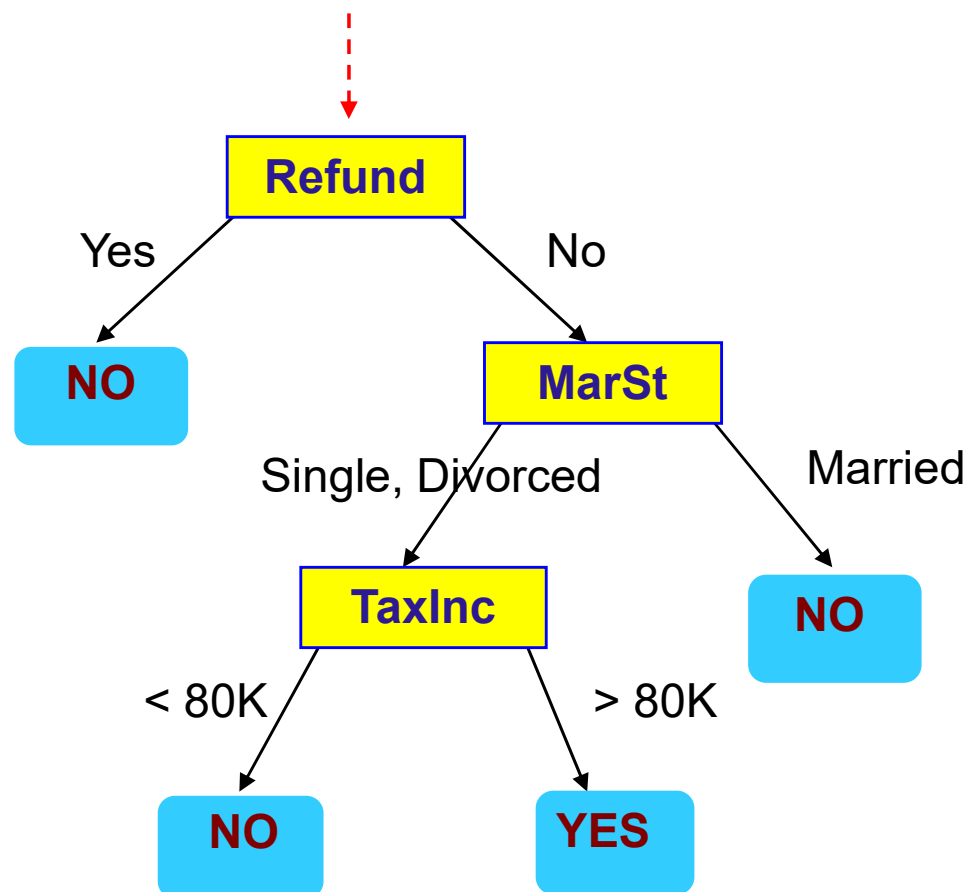
Apply Model to Test Data

Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?

Apply Model to Test Data

Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?

```
          Refund
       Yes /    \ No
          /      \
        NO       MarSt
          Single, Divorced /    \ Married
                          /      \
                      TaxInc     NO
                < 80K /    \ > 80K
                     /      \
                   NO       YES
```

Apply Model to Test Data

Test Data

Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?

```
        Refund
      Yes /    \ No
        /        \
      NO         MarSt
          Single, Divorced /    \ Married
                         /        \
                      TaxInc       NO
                 < 80K /   \ > 80K
                     /       \
                   NO        YES
```

Apply Model to Test Data

CSci 5980 Spring 2004

Test Data

Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?

Refund

Yes → NO

No → MarSt

MarSt:
- Single, Divorced → TaxInc
- Married → NO

TaxInc:
- < 80K → NO
- > 80K → YES
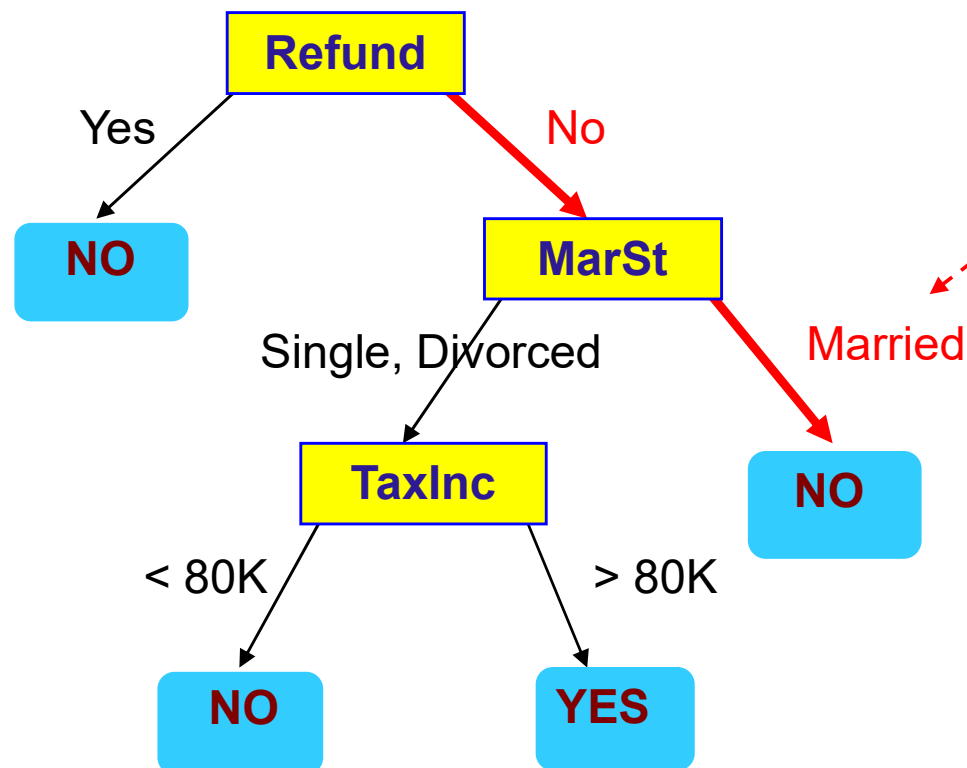
Apply Model to Test Data

Test Data

Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?

Assign Cheat to "No"

Decision Tree 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

Induction

Tree Induction algorithm

Learn Model

Model

Decision Tree

Apply Model

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

Deduction

Decision Tree Induction

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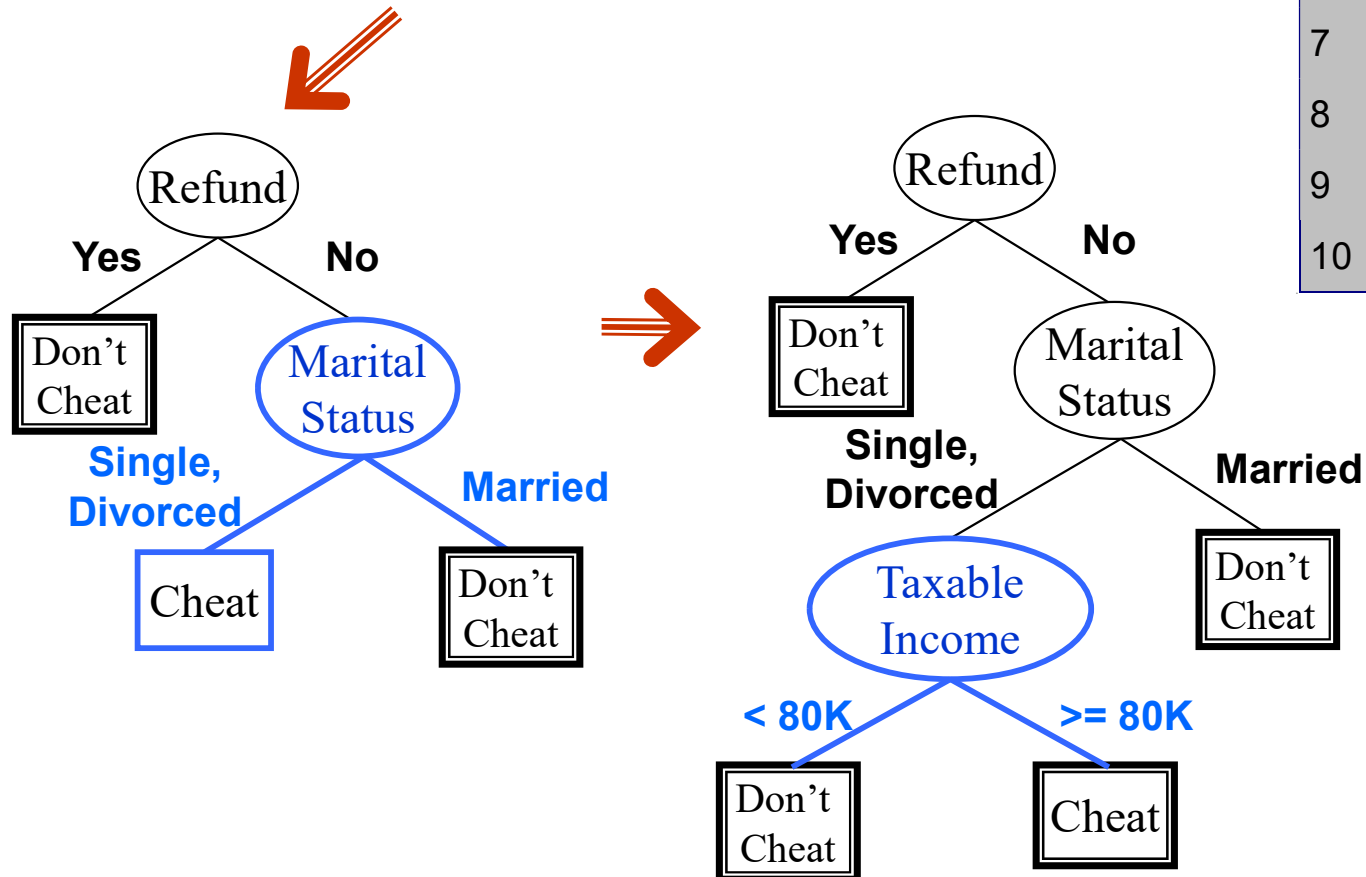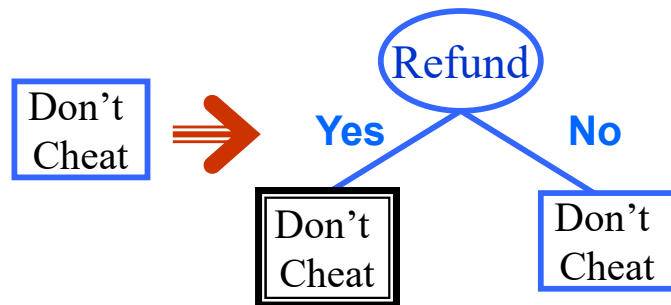 - SLIQ,SPRINT

General Structure of Hunt's Algorithm

- Let D_t là tập các bản ghi huấn luyện mà hướng tới nút t

- General Procedure:

  - If D_t chứa các bản ghi thuộc cùng 1 lớp y_t, then t là nút lá gán nhãn y_t

  - If D_t là tập rỗng thì t là 1 nút lá gán nhãn bởi lớp mặc định, y_d

  - If D_t chứa các bản ghi thuộc nhiều hơn 1 lớp, sử dụng một thuộc tính kiểm tra để phân chia dữ liệu thành các tập con nhỏ hơn. Áp dụng thủ tục trên một cách đệ quy cho mỗi tập con.

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

D_t

?

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

Don't Cheat ⇒

Refund
- **Yes** → Don't Cheat
- **No** → Don't Cheat

Refund
- **Yes** → Don't Cheat
- **No** → **Marital Status**
  - **Single, Divorced** → Cheat
  - **Married** → Don't Cheat

Refund
- **Yes** → Don't Cheat
- **No** → **Marital Status**
  - **Single, Divorced** → **Taxable Income**
    - **< 80K** → Don't Cheat
    - **>= 80K** → Cheat
  - **Married** → Don't Cheat

Tree Induction

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

- Issues
  - Determine how to split the records
    - Xác định cụ thể các điều kiện kiểm tra thuộc tính ?
    - Xác định xem việc phân tách đó đã là tốt nhất chưa ?
  - Determine when to stop splitting

How to Specify Test Condition?

- Depends on attribute types
  - Nominal
  - Ordinal
  - Continuous

- 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.
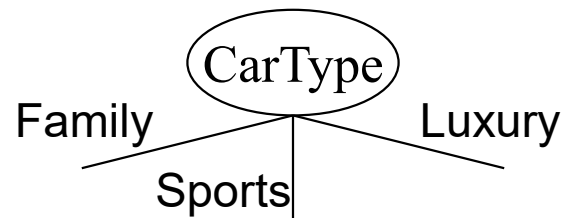
CarType

Family — Sports — Luxury

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

CarType

{Sports, Luxury} — {Family}

OR

CarType

{Family, Luxury} — {Sports}
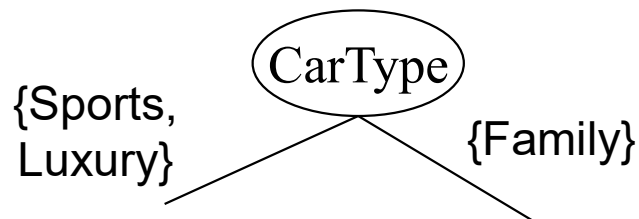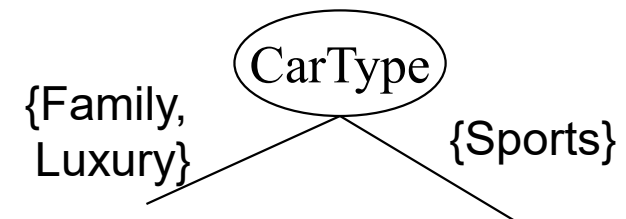
Splitting Based on Ordinal Attributes

- Multi-way split: Use as many partitions as distinct values.
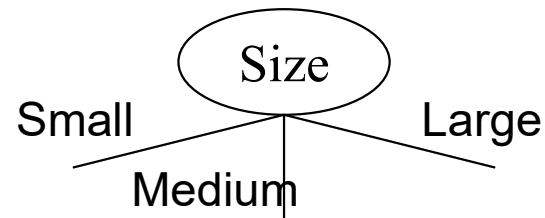
Size
Small
Medium
Large

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

Size
{Small, Medium}
{Large}

hay

Size
{Medium, Large}
{Small}
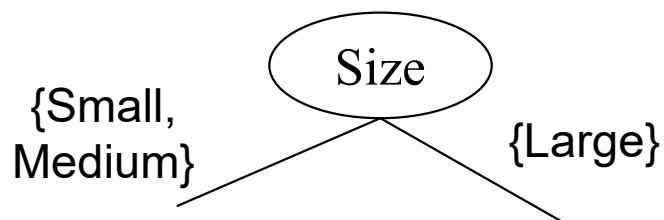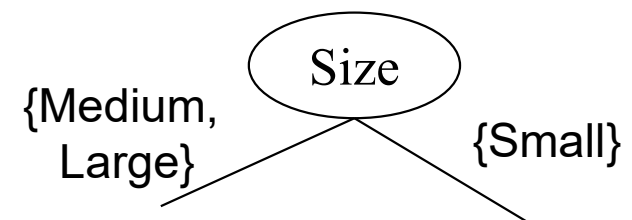
- What about this split?

Size
{Small, Large}
{Medium}
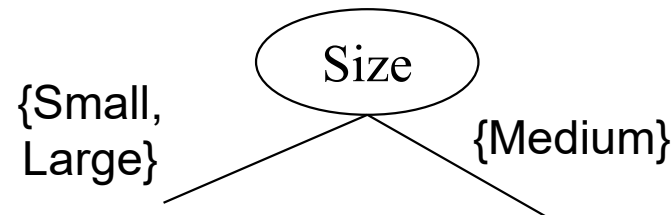
Splitting Based on Continuous Attributes

- Different ways of handling

  - Discretization to form an ordinal categorical attribute

    - Static – discretize once at the beginning

    - Dynamic – ranges can be found by equal interval bucketing, equal frequency bucketing (percentiles), or clustering.

  - Binary Decision: (A < v) or (A ≥ v)

    - consider all possible splits and finds the best cut

    - can be more compute intensive

Splitting Based on Continuous Attributes
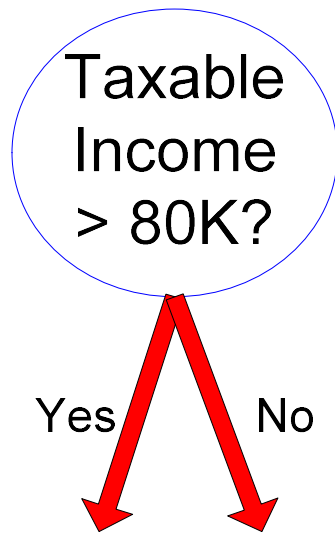

(i) Binary split
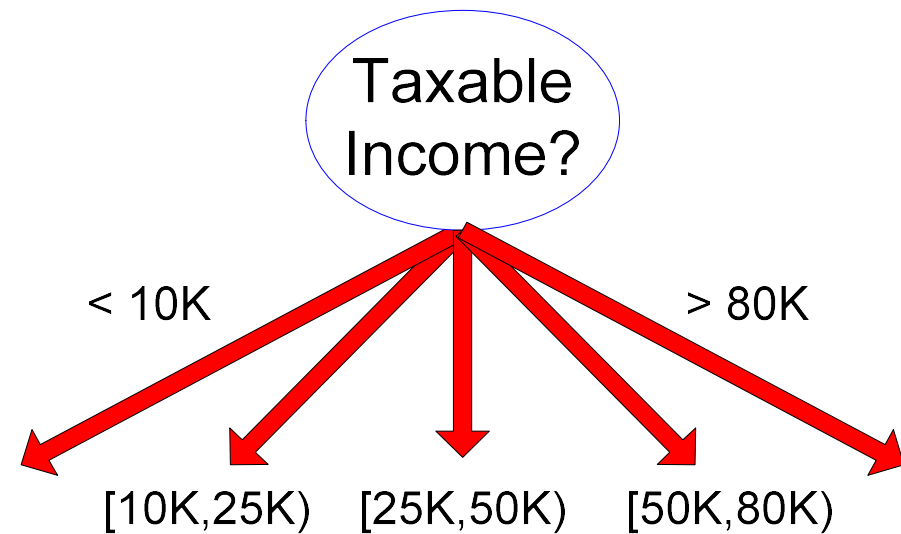
(ii) Multi-way split

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

How to determine the Best Split

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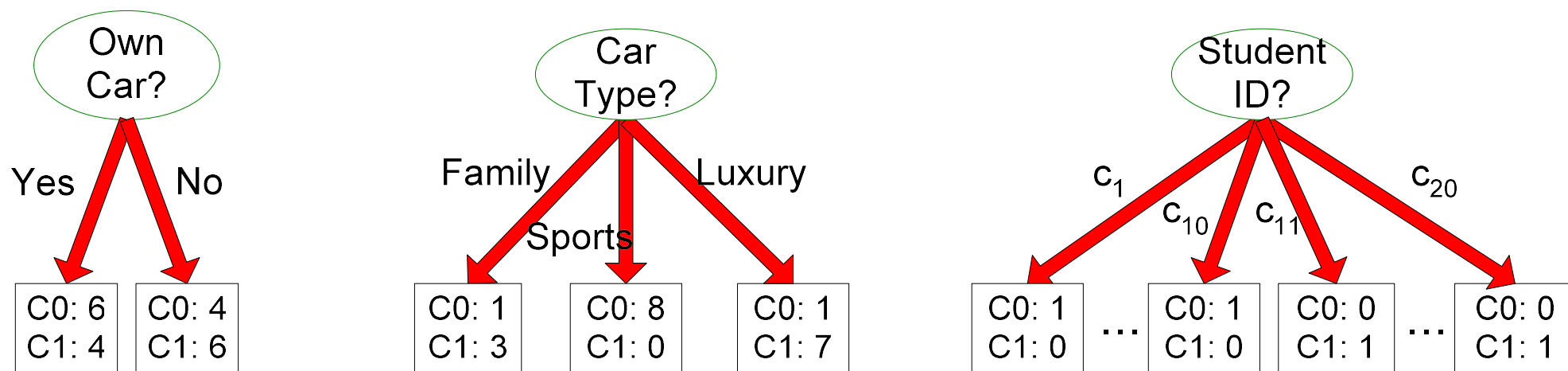
Which test condition is the best?

How to determine the Best Split

- Greedy approach:
  - Các nút có sự phân nhánh với việc **phân phối lớp đồng nhất** được ưu tiên hơn

- Need a measure of node impurity:

| C0: 5 |
| C1: 5 |

| C0: 9 |
| C1: 1 |

Non-homogeneous (không đồng nhất),

High degree of impurity (độ không đồng nhất cao)

Homogeneous (đồng nhất),

Low degree of impurity (độ không đồng nhất thấp)

Measures of Node Impurity
(độ không đồng nhất của các nút)

- Gini Index

- Entropy

- Misclassification error

Measure of Impurity: GINI

☐ Chỉ số Gini của nút t :

$$GINI(t) = 1 - \sum_j [p(j \mid t)]^2$$

(NOTE: $p(j \mid t)$ là tần suất của lớp j tại nút t.

– Maximum ($1 - 1/n_c$) khi các bản ghi được phân bố bằng nhau giữa tất cả các lớp => thông tin có ích thấp nhất

– Minimum (0.0) khi tất cả các bản ghi thuộc 1 lớp => thông tin có ích nhất

C1	0
C2	6
Gini=0.000	

C1	1
C2	5
Gini=0.278	

C1	2
C2	4
Gini=0.444	

C1	3
C2	3
Gini=0.500	

Examples for computing GINI

$$GINI(t) = 1 - \sum_j [p(j \mid t)]^2$$

C1	0
C2	6

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

Gini = 1 – P(C1)^2 – P(C2)^2 = 1 – 0 – 1 = 0

C1	1
C2	5

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

Gini = 1 – (1/6)^2 – (5/6)^2 = 0.278

C1	2
C2	4

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

Gini = 1 – (2/6)^2 – (4/6)^2 = 0.444

Chỉ số GINI tại 1 phân tách

- Used in CART, SLIQ, SPRINT.

- Khi nút p phân chia thành k phần, chất lượng của sự phân chia (split) được tính bằng:
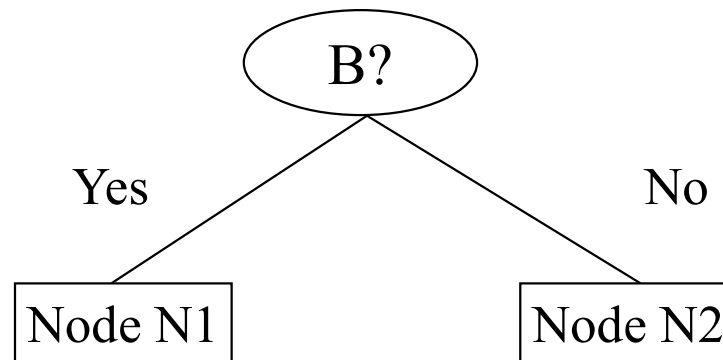
$$GINI_{split} = \sum_{i=1}^{k} \frac{n_i}{n} GINI(i)$$

where, n_i = số các bản ghi tại phần con thứ i,

          n = số các bản ghi tại nút p.

Binary Attributes: Computing GINI Index

- Splits into two partitions
- Effect of Weighing partitions:
  – Larger and Purer Partitions are sought for.

Các bản ghi tại nút B (12):

B?

Yes   No

Node N1   Node N2

	Parent
C1	**6**
C2	**6**
Gini = 0.500	

Gini(N1)
= 1 – (5/6)² – (2/6)²
= 0.194

	N1	**N2**
C1	**5**	**1**
C2	**2**	**4**
Gini=0.333		

Gini(Children)
= 7/12 * Gini(N1) +
**   5/12 * Gini(N2)**
= 0.333 < Gini(Parent)

Gini(N2)
= 1 – (1/6)² – (4/6)²
= 0.528

=> Nên tách vì gini giảm, lượng thông tin nhiều hơn

Categorical Attributes: Computing Gini Index

◻ For each distinct value, gather counts for each class in the dataset

◻ Use the count matrix to make decisions

Multi-way split

	CarType		
	Family	**Sports**	**Luxury**
C1	1	2	1
C2	4	1	1
Gini	**0.393**		

Two-way split
(find best partition of values)

	CarType	
	{Sports, Luxury}	**{Family}**
C1	3	1
C2	2	4
Gini	**0.400**	

	CarType	
	{Sports}	**{Family, Luxury}**
C1	2	2
C2	1	5
Gini	**0.419**	

Continuous Attributes: Computing Gini Index

- Use Binary Decisions based on one value

- Several Choices for the splitting value
  - Number of possible splitting values = Number of distinct values

- Each splitting value has a count matrix associated with it
  - Class counts in each of the partitions, A < v and A $\geq$ v

- Simple method to choose best v
  - For each v, scan the database to gather count matrix and compute its Gini index
  - Computationally Inefficient! Repetition of work.

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

Taxable Income > 80K?

Yes   No

Continuous Attributes: Computing Gini Index...

- 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

Các giá trị được sắp xếp →

Các vị trí phân chia →

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

Alternative Splitting Criteria based on INFO

- Entropy tại nút t:

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

(NOTE: $p(j \mid t)$ là tần suất của lớp j tại nút t.

- Dùng để đo lường sự đồng nhất của một nút.
  - Maximum (log n_c) khi các bản ghi được phân bổ bằng nhau giữa tất cả các lớp => lượng thông tin ít nhất
  - Minimum (0.0) khi tất cả các bản ghi thuộc một lớp => nhiều thông tin nhất

- Entropy based computations are similar to the GINI index computations

Examples for computing Entropy

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

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

Lượng thông tin thu được của 1 phân tách

☐ Information Gain:

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

nút cha p được phân chia thành k phần và ni là số bản ghi trong phần i

– Dùng để đo việc giảm Entropy hay việc giảm lượng thông tin được chứa đựng trong nút đó do nút đó được phân nhánh. Chúng ta sẽ chọn sự phân nhánh giảm lượng thông tin thấp nhất (hay độ đo GAIN lớn nhất, >0 tốt hơn <0)

– Used in ID3 and C4.5

– Nhược điểm: Có xu hướng lựa chọn sự phân chia có nhiều nhánh, mỗi nhánh nhỏ nhưng đồng nhất, không thực tế lắm

Splitting Based on INFO...

☐ Gain Ratio (tỉ lệ lượng thông tin):

$$GainRATIO_{split} = \frac{GAIN_{Split}}{SplitINFO} \qquad SplitINFO = -\sum_{i=1}^{k} \frac{n_i}{n} \log \frac{n_i}{n}$$

nút cha p được phân chia thành k phần và ni là số bản ghi trong phần 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

Splitting Criteria based on Classification Error (Độ đo lỗi phân loại)

- Lỗi phân loại tại nút t :

$$Error(t) = 1 - \max_i P(i \mid t)$$

- Measures misclassification error made by a node.
  - Maximum (1 - $1/n_c$) khi các bản ghi được phân bố bằng nhau giữa tất cả các lớp => thông tin có ích thấp nhất
  - Minimum (0.0) khi tất cả các bản ghi thuộc một lớp => thông tin có ích nhiều nhất

Examples for Computing Error

$$Error(t) = 1 - \max_{i} P(i \mid t)$$

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

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

Error = 1 – max (0, 1) = 1 – 1 = 0

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

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

Error = 1 – max (1/6, 5/6) = 1 – 5/6 = 1/6

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

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

Error = 1 – max (2/6, 4/6) = 1 – 4/6 = 1/3

Comparison among Splitting Criteria

For a 2-class problem:
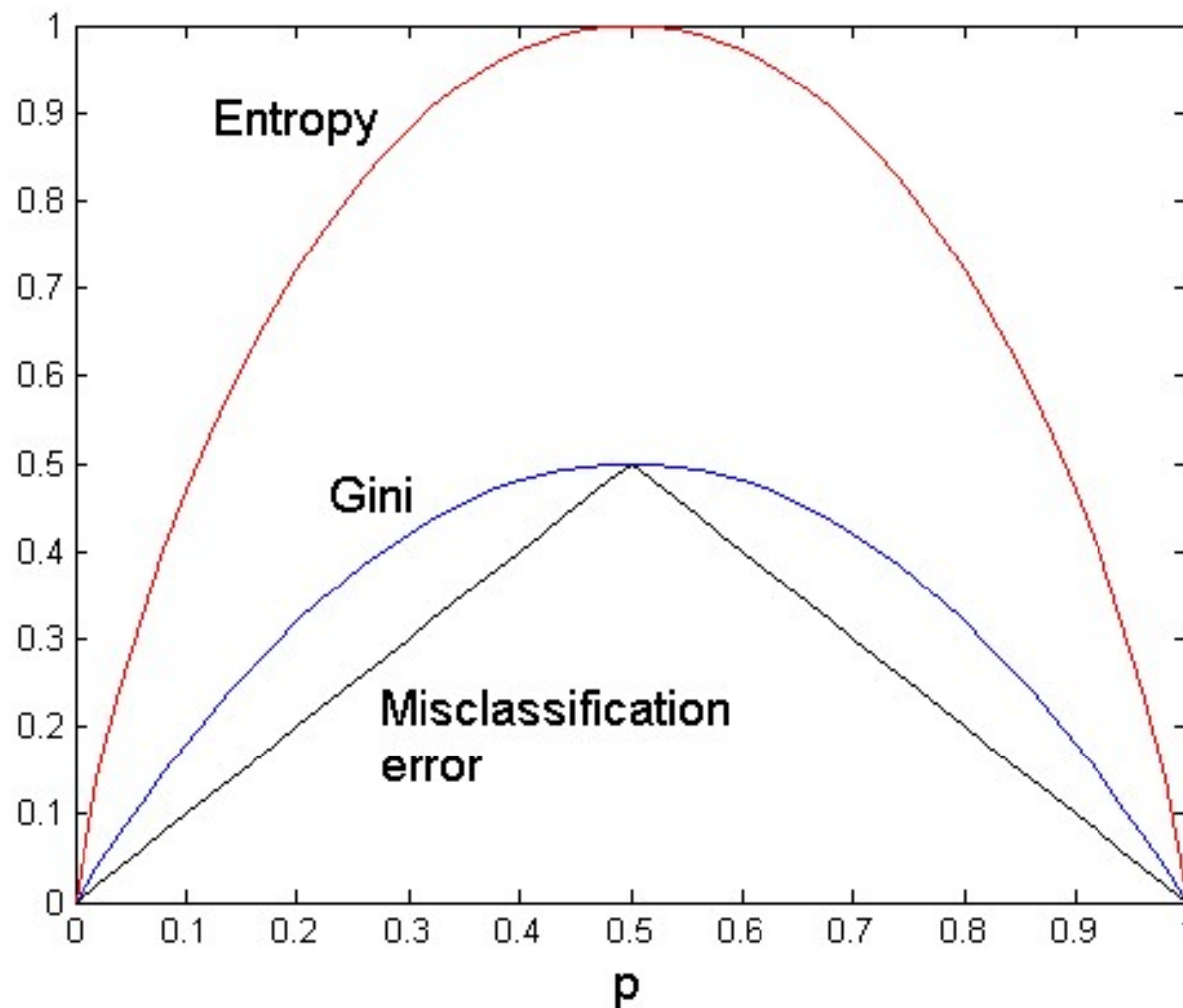
Tree Induction

- 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

Stopping Criteria for Tree Induction

- Dừng phân nhánh 1 nút khi tất cả các bản ghi là cùng 1 lớp

- Dừng phân nhánh 1 nút khi tất cả các bản ghi có các giá trị thuộc tính giống nhau

- Kết thúc sớm trong một số trường hợp đặc biệt (sẽ được thảo luận sau)

Decision Tree Based Classification

◻ Advantages:

– Inexpensive to construct

– Extremely fast at classifying unknown records

– Easy to interpret for small-sized trees

– Accuracy is comparable to other classification techniques for many simple data sets

Example: C4.5

- Simple depth-first construction.

- Uses Information Gain

- Sorts Continuous Attributes at each node.

- Needs entire data to fit in memory.

- Unsuitable for Large Datasets.
    - Needs out-of-core sorting.

- You can download the software from: http://www.cse.unsw.edu.au/~quinlan/c4.5r8.tar.gz

Practical Issues of Classification

- Underfitting and Overfitting

- Missing Values

- Costs of Classification

Underfitting and Overfitting (Example)
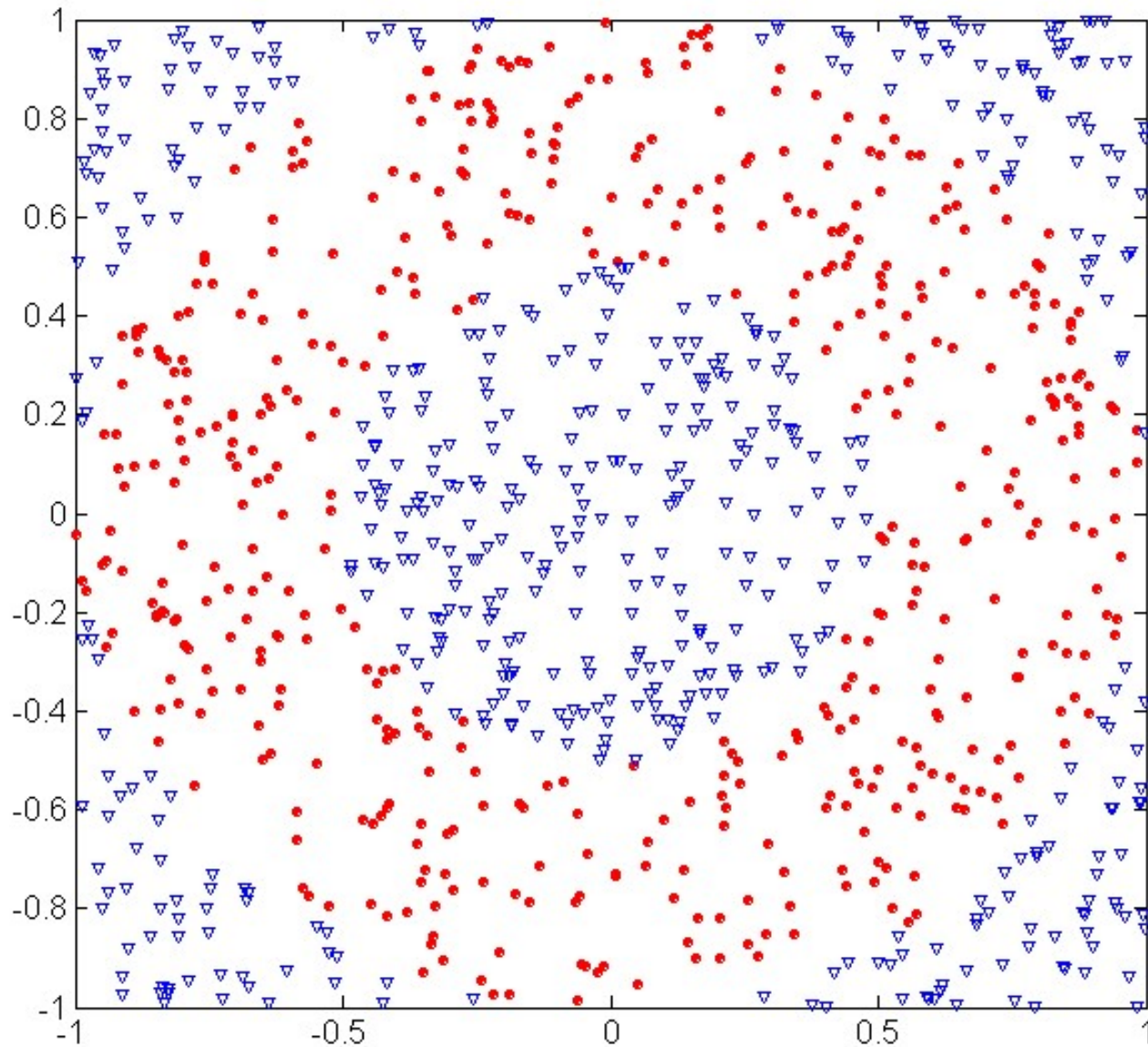
500 circular and 500 triangular data points.

Circular points:

$0.5 \leq \text{sqrt}(x_1{}^2+x_2{}^2) \leq 1$

Triangular points:

$\text{sqrt}(x_1{}^2+x_2{}^2) > 0.5$ or

$\text{sqrt}(x_1{}^2+x_2{}^2) < 1$

Underfitting and Overfitting
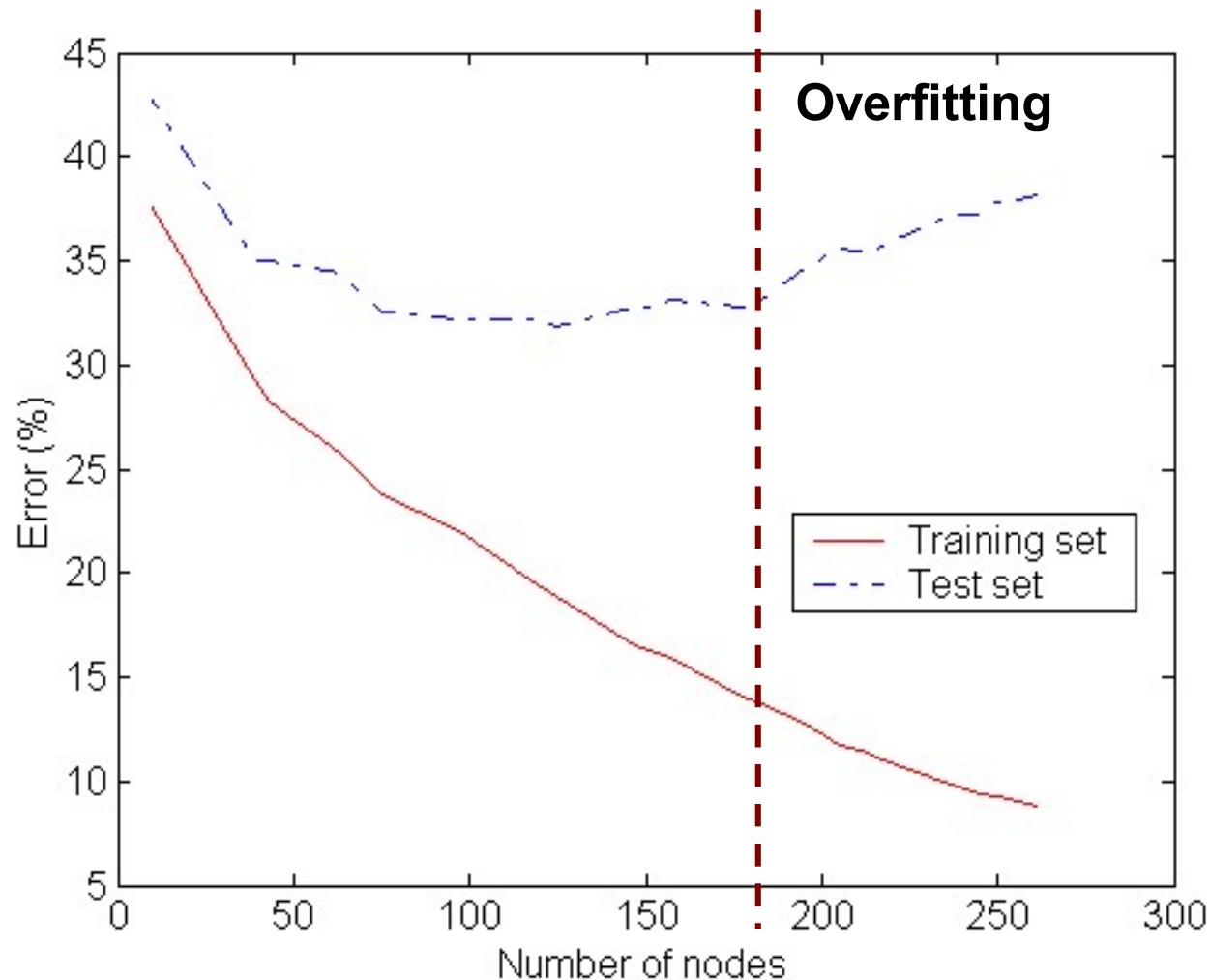
Underfitting: when model is too simple, both training and test errors are large

Overfitting due to Noise
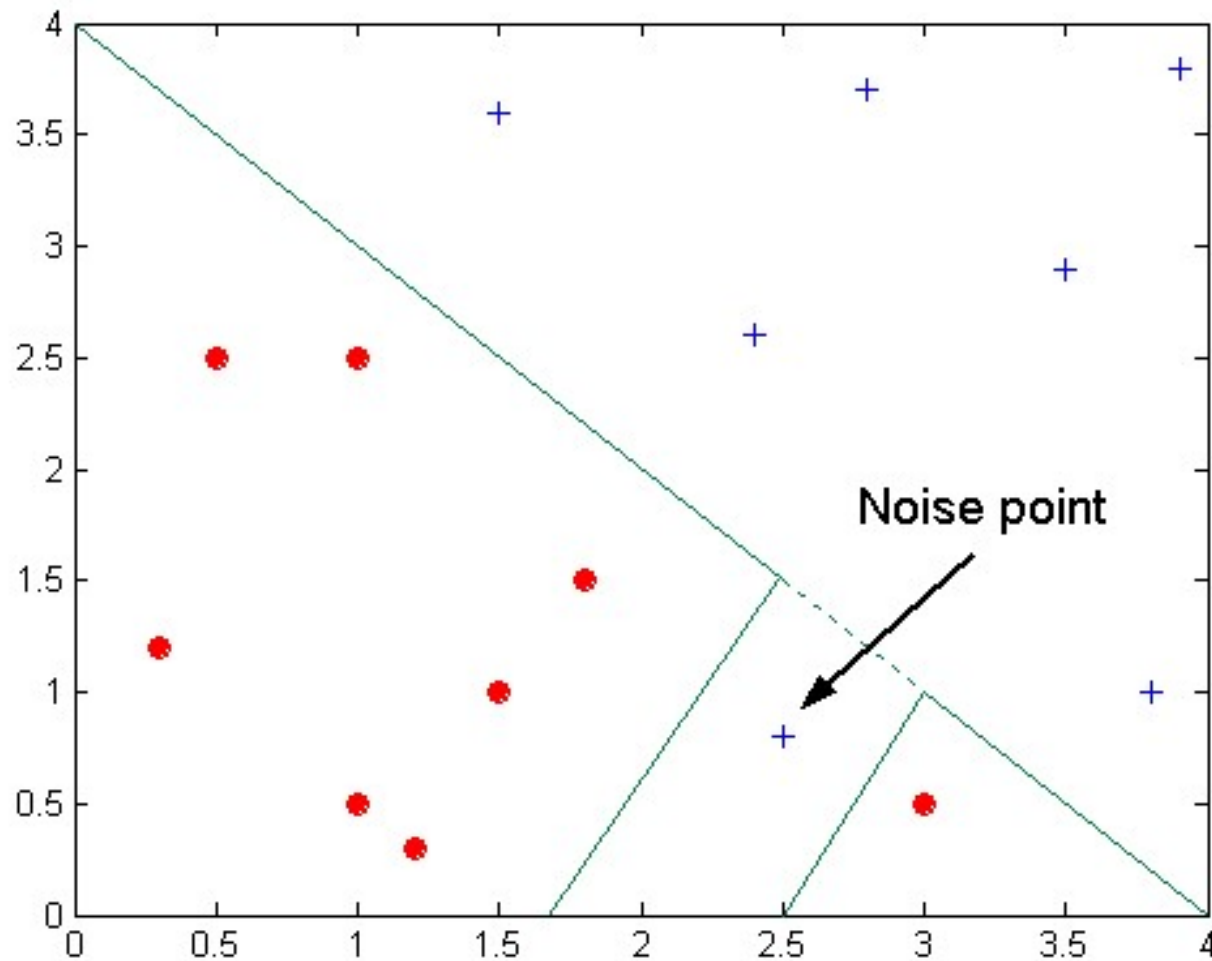
Decision boundary is distorted by noise point

Overfitting due to Insufficient Examples
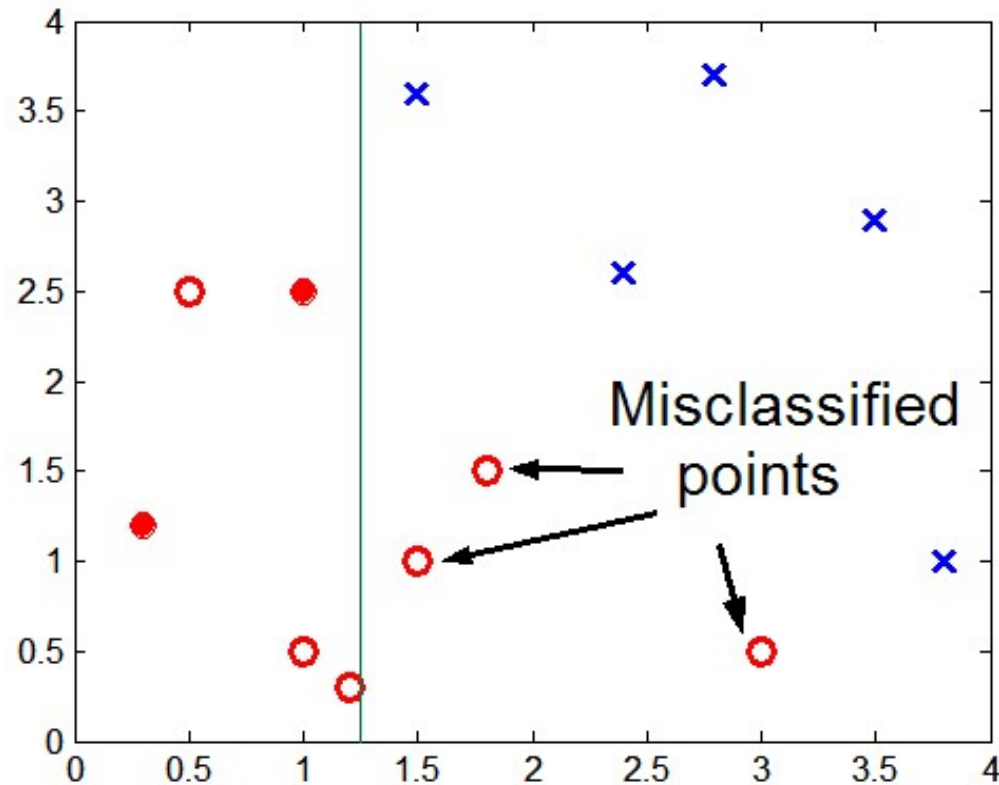
Lack of data points in the lower half of the diagram makes it difficult to predict correctly the class labels of that region

- Insufficient number of training records in the region causes the decision tree to predict the test examples using other training records that are irrelevant to the classification task

Notes on Overfitting

- Overfitting results in decision trees that are more complex than necessary

- Training error no longer provides a good estimate of how well the tree will perform on previously unseen records

- Need new ways for estimating errors

Estimating Generalization Errors

- Re-substitution errors: error on training (Σ e(t))
- Generalization errors: error on testing (Σ e'(t))

- Methods for estimating generalization errors:
  - Optimistic approach: e'(t) = e(t)
  - Pessimistic approach:
    - For each leaf node: e'(t) = (e(t)+0.5)
    - Total errors: e'(T) = e(T) + N $\times$ 0.5 (N: number of leaf nodes)
    - For a tree with 30 leaf nodes and 10 errors on training (out of 1000 instances):
      Training error = 10/1000 = 1%

      Generalization error = (10 + 30$\times$0.5)/1000 = 2.5%
  - Reduced error pruning (REP):
    - uses validation data set to estimate generalization error

Occam's Razor

☐ Given two models of similar generalization errors, one should prefer the simpler model over the more complex model

☐ For complex models, there is a greater chance that it was fitted accidentally by errors in data

☐ Therefore, one should include model complexity when evaluating a model

How to Address Overfitting

☐ Pre-Pruning (Early Stopping Rule)

- Stop the algorithm before it becomes a fully-grown tree

- Typical stopping conditions for a node:

  ◆ Stop if all instances belong to the same class

  ◆ Stop if all the attribute values are the same

- More restrictive conditions:

  ◆ Stop if number of instances is less than some user-specified threshold

  ◆ 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).

How to Address Overfitting…

☐ Post-pruning

- Grow decision tree to its entirety
- Trim the nodes of the decision tree in a bottom-up fashion
- If generalization error improves after trimming, replace sub-tree by a leaf node.
- Class label of leaf node is determined from majority class of instances in the sub-tree
- Can use MDL for post-pruning

Example of Post-Pruning

Class = Yes	20
Class = No	10
Error = 10/30	

Training Error (Before splitting) = 10/30

Pessimistic error = (10 + 0.5)/30 = 10.5/30

Training Error (After splitting) = 9/30

Pessimistic error (After splitting)

$$= (9 + 4 \times 0.5)/30 = 11/30$$

PRUNE!

Class = Yes	8	Class = Yes	3	Class = Yes	4	Class = Yes	5
Class = No	4	Class = No	4	Class = No	1	Class = No	1
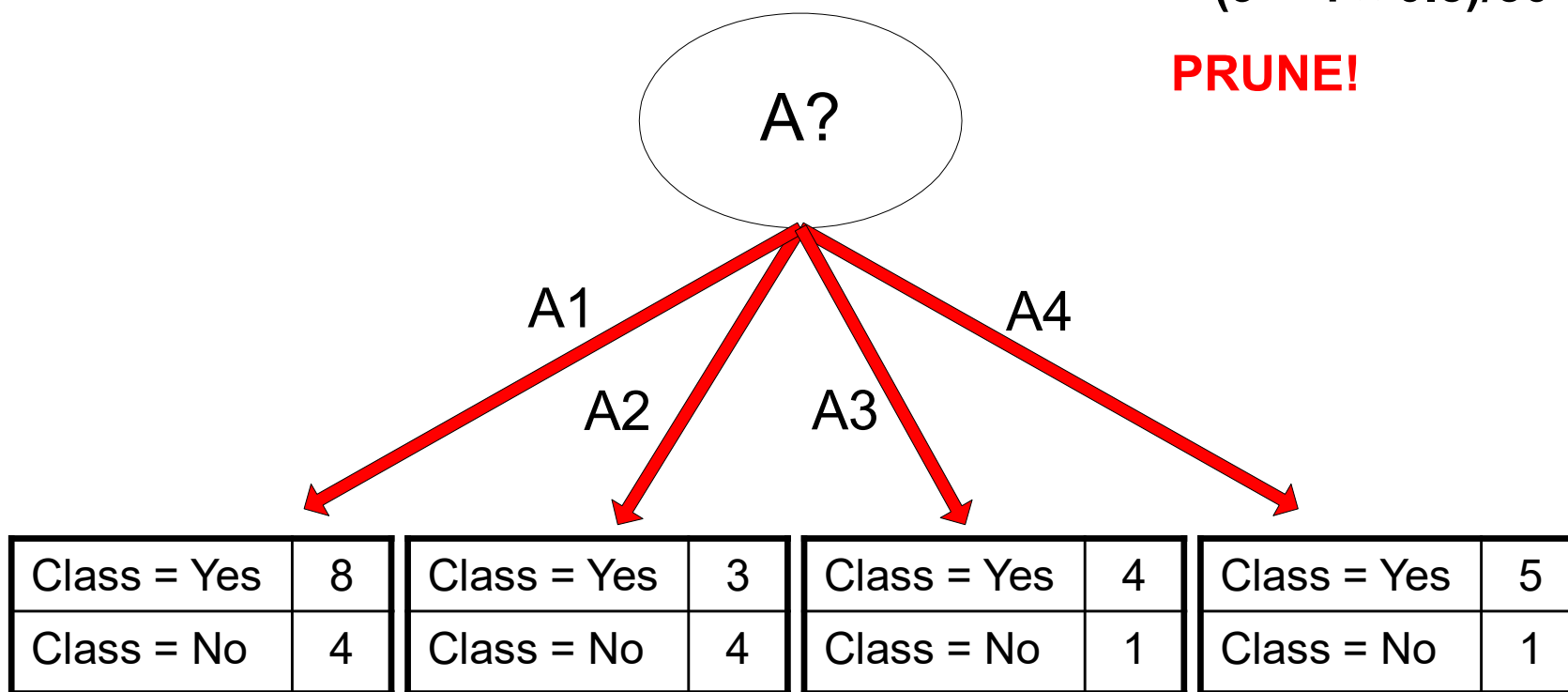
Examples of Post-pruning

– ## Optimistic error?

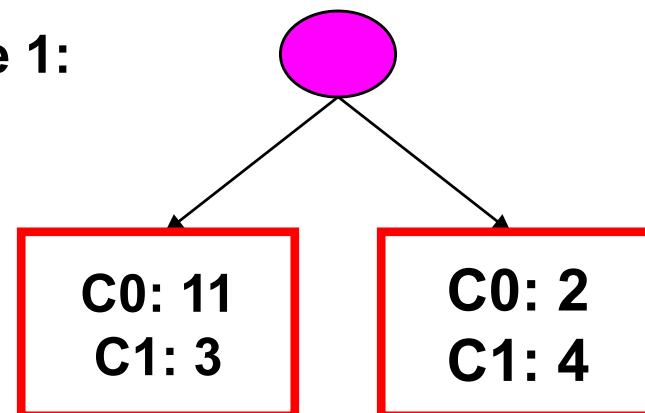**Don't prune for both cases**

– ## Pessimistic error?

Don't prune case 1, prune case 2
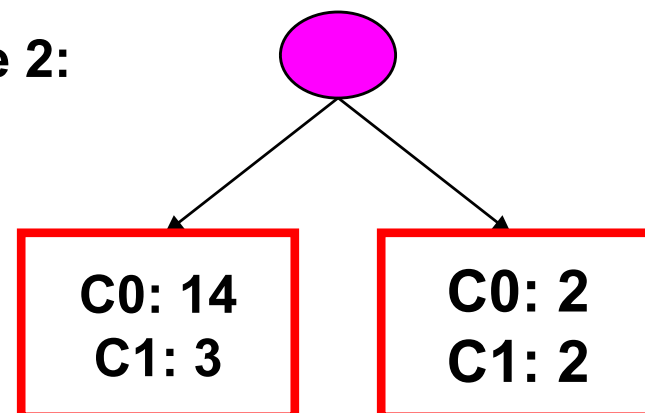
– ## Reduced error pruning?

Depends on validation set

Case 1:

```
        (  )
       /    \
C0: 11      C0: 2
C1: 3       C1: 4
```

Case 2:

```
        (  )
       /    \
C0: 14      C0: 2
C1: 3       C1: 2
```
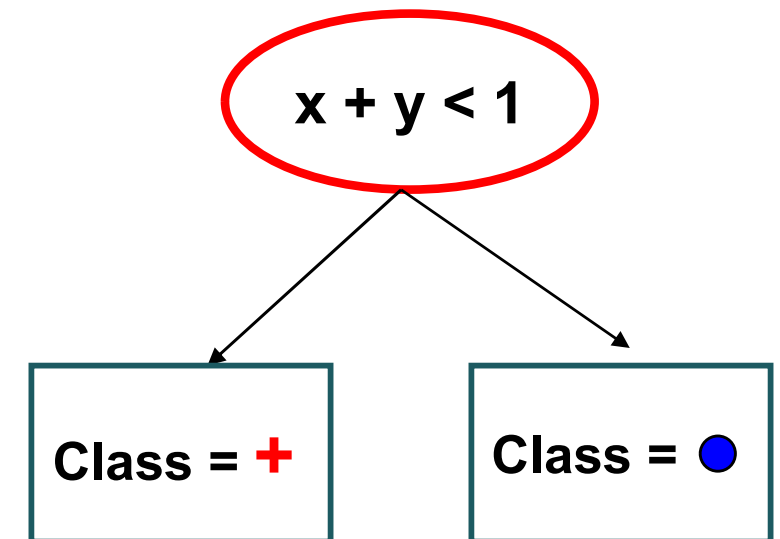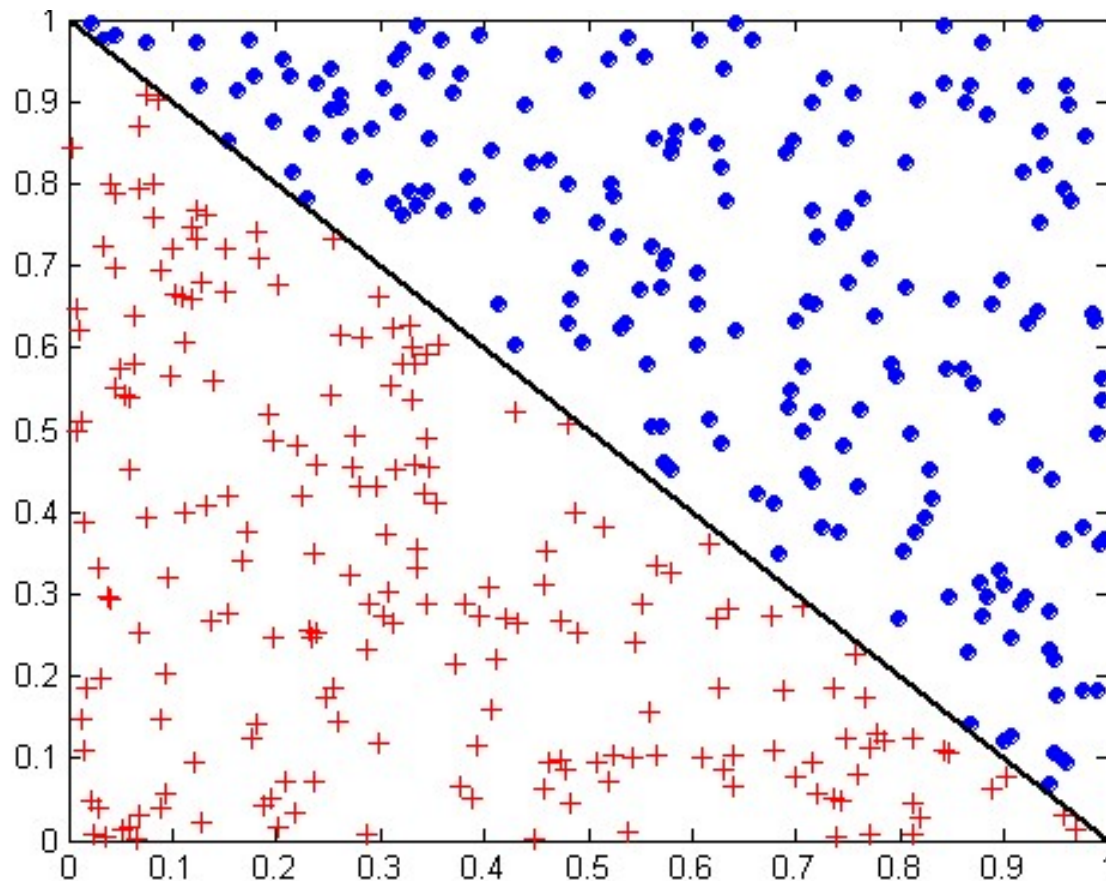
Expressiveness

☐ Decision tree provides expressive representation for learning discrete-valued function

 – But they do not generalize well to certain types of Boolean functions

 ◆ Example: parity function:

 – Class = 1 if there is an even number of Boolean attributes with truth value = True
 – Class = 0 if there is an odd number of Boolean attributes with truth value = True

 ◆ For accurate modeling, must have a complete tree

☐ Not expressive enough for modeling continuous variables

 – Particularly when test condition involves only a single attribute at-a-time

Decision Boundary
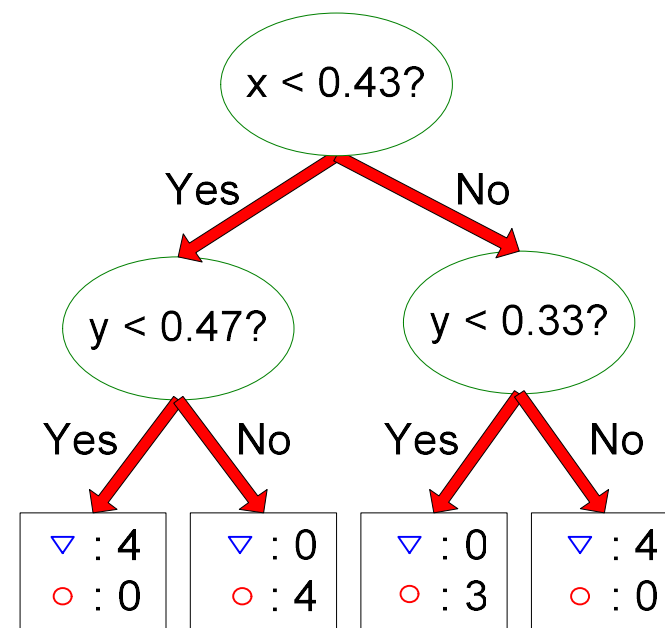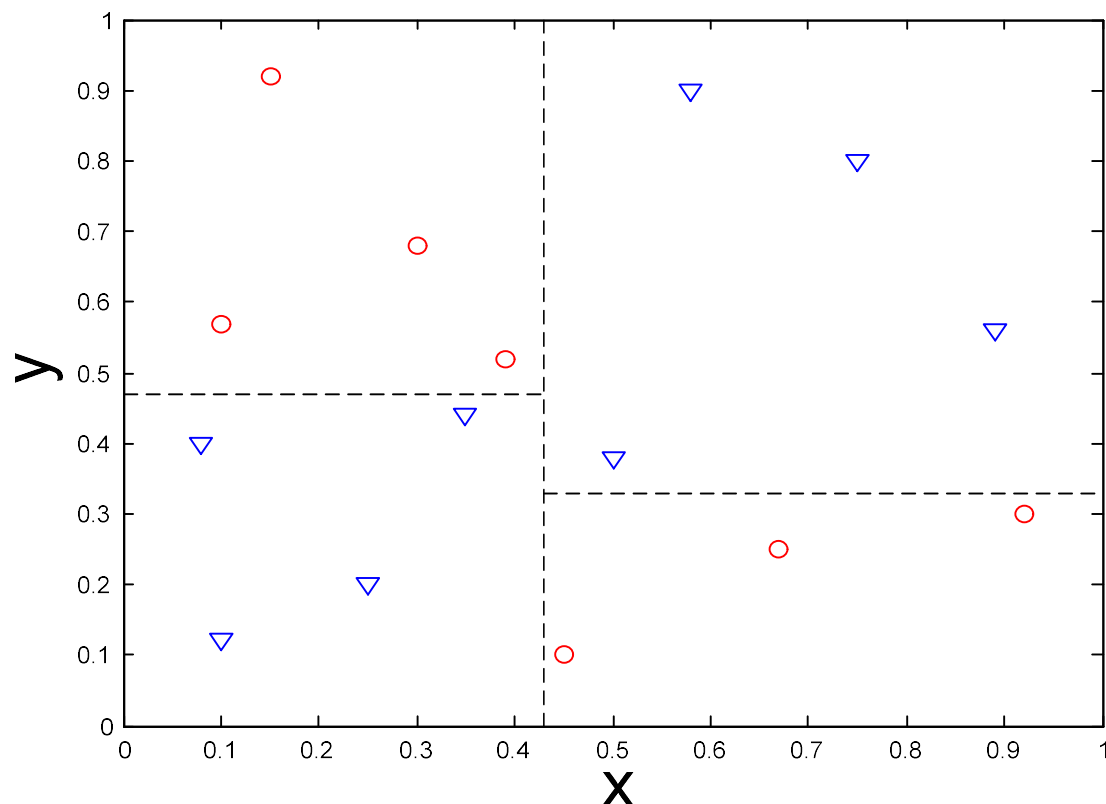
- Border line between two neighboring regions of different classes is known as decision boundary

- Decision boundary is parallel to axes because test condition involves a single attribute at-a-time

Oblique Decision Trees

- Test condition may involve multiple attributes

- More expressive representation

- Finding optimal test condition is computationally expensive

Model Evaluation

- Metrics for Performance Evaluation

    – How to evaluate the performance of a model?

- Methods for Performance Evaluation

    – How to obtain reliable estimates?

- Methods for Model Comparison

    – How to compare the relative performance among competing models?

Metrics for Performance Evaluation

- Focus on the predictive capability of a model
  - Rather than how fast it takes to classify or build models, scalability, etc.
- Confusion Matrix:

	PREDICTED CLASS		
ACTUAL CLASS		Class=Yes	Class=No
	Class=Yes	a	b
	Class=No	c	d

a: TP (true positive)

b: FN (false negative)

c: FP (false positive)

d: TN (true negative)

Metrics for Performance Evaluation…

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

- Most widely-used metric:

$$\text{Đô chính xác} = \frac{a+d}{a+b+c+d} = \frac{TP+TN}{TP+TN+FP+FN}$$

Limitation of Accuracy

- Consider a 2-class problem
  - Number of Class 0 examples = 9990
  - Number of Class 1 examples = 10

- If model predicts everything to be class 0, accuracy is 9990/10000 = 99.9 %
  - Accuracy is misleading because model does not detect any class 1 example

Cost Matrix

	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)

C(i|j): Chi phí phân loại lỗi lớp j trong thực tế thành là lớp i

Computing Cost of Classification

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

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

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

Accuracy = 80%

Cost = 3910

Accuracy = 90%

Cost = 4255

Cost vs Accuracy

Count	PREDICTED CLASS	
	Class=Yes	Class=No
ACTUAL CLASS Class=Yes	a	b
ACTUAL CLASS Class=No	c	d

Accuracy is proportional to cost if
1. C(Yes|No)=C(No|Yes) = q
2. C(Yes|Yes)=C(No|No) = p

$$N = a + b + c + d$$

$$Accuracy = (a + d)/N$$

Cost	PREDICTED CLASS	
	Class=Yes	Class=No
ACTUAL CLASS Class=Yes	p	q
ACTUAL CLASS Class=No	q	p

$$Cost = p \, (a + d) + q \, (b + c)$$
$$= p \, (a + d) + q \, (N - a - d)$$
$$= q \, N - (q - p)(a + d)$$
$$= N \, [q - (q-p) \times Accuracy]$$

Cost-Sensitive Measures

$$\text{Precision (p)} = \frac{a}{a+c}$$

$$\text{Recall (r)} = \frac{a}{a+b}$$

$$\text{F-measure (F)} = \frac{2rp}{r+p} = \frac{2a}{2a+b+c}$$

- Precision is biased towards C(Yes|Yes) & C(Yes|No)
- Recall is biased towards C(Yes|Yes) & C(No|Yes)
- F-measure is biased towards all except C(No|No)

$$\text{Weighted Accuracy} = \frac{w_1 a + w_4 d}{w_1 a + w_2 b + w_3 c + w_4 d}$$

Model Evaluation

- Metrics for Performance Evaluation
  - How to evaluate the performance of a model?

- Methods for Performance Evaluation
  - How to obtain reliable estimates?

- Methods for Model Comparison
  - How to compare the relative performance among competing models?

Methods for Performance Evaluation

- How to obtain a reliable estimate of performance?

- Performance of a model may depend on other factors besides the learning algorithm:
  - Class distribution
  - Cost of misclassification
  - Size of training and test sets

Learning Curve
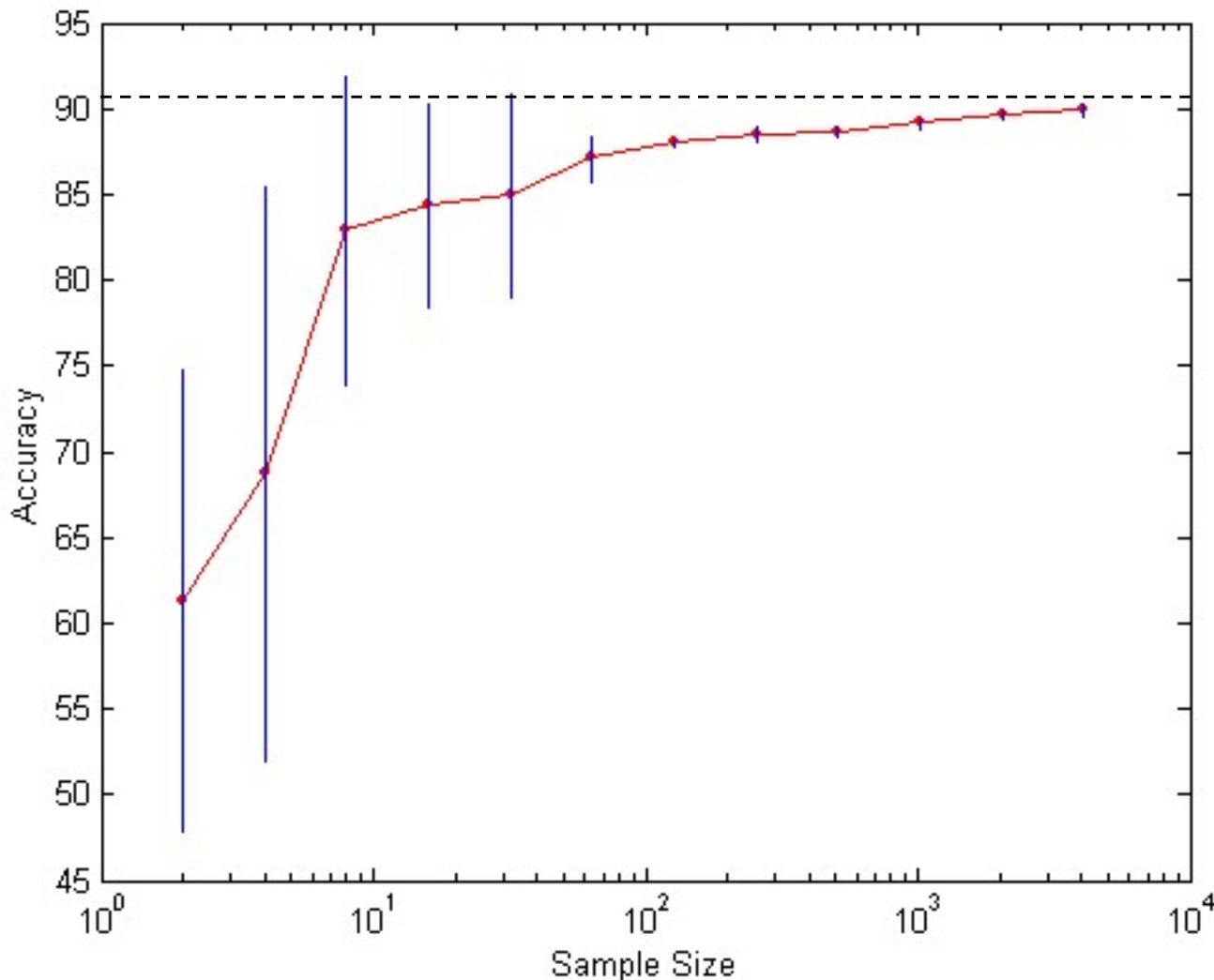
- Learning curve shows how accuracy changes with varying sample size

- Requires a sampling schedule for creating learning curve:

  - Arithmetic sampling (Langley, et al)

  - Geometric sampling (Provost et al)

Effect of small sample size:

- Bias in the estimate

- Variance of estimate

Methods of Estimation

- Holdout
  - Reserve 2/3 for training and 1/3 for testing
- Random subsampling
  - Repeated holdout
- Cross validation
  - Partition data into k disjoint subsets
  - k-fold: train on k-1 partitions, test on the remaining one
  - Leave-one-out: k=n
- Stratified sampling
  - oversampling vs undersampling
- Bootstrap
  - Sampling with replacement

Model Evaluation

- Metrics for Performance Evaluation
  - How to evaluate the performance of a model?

- Methods for Performance Evaluation
  - How to obtain reliable estimates?

- Methods for Model Comparison
  - How to compare the relative performance among competing models?

Test of Significance

- Given two models:
  - Model M1: accuracy = 85%, tested on 30 instances
  - Model M2: accuracy = 75%, tested on 5000 instances

- Can we say M1 is better than M2?
  - How much confidence can we place on accuracy of M1 and M2?
  - Can the difference in performance measure be explained as a result of random fluctuations in the test set?
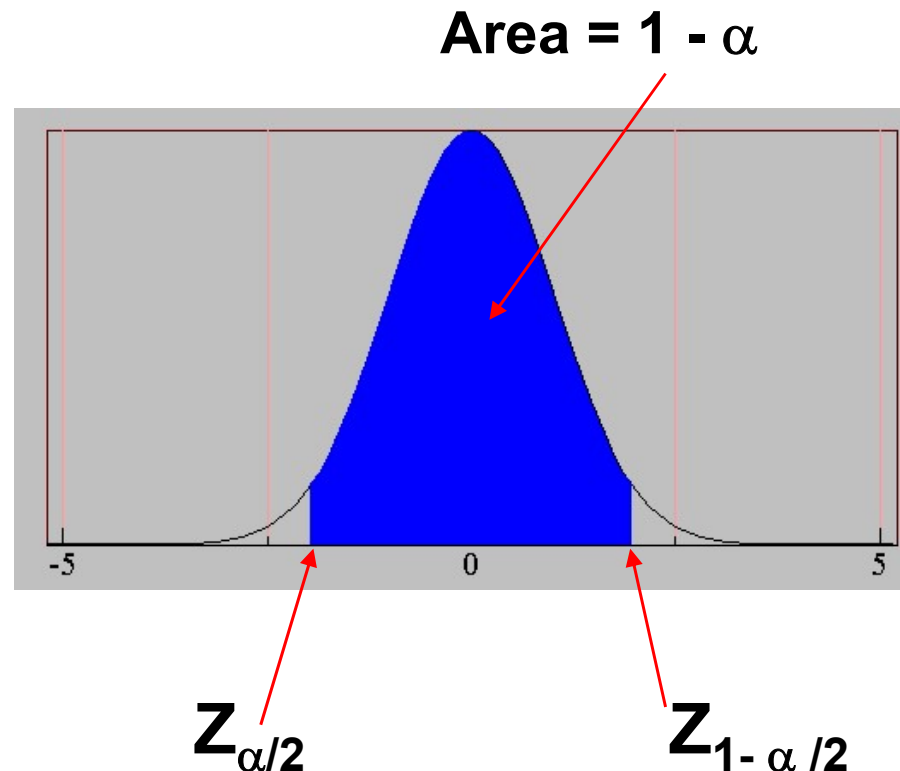
Confidence Interval for Accuracy

- Prediction can be regarded as a Bernoulli trial
  - A Bernoulli trial has 2 possible outcomes
  - Possible outcomes for prediction: correct or wrong
  - Collection of Bernoulli trials has a Binomial distribution:
    - $x \sim Bin(N, p)$ x: number of correct predictions
    - e.g: Toss a fair coin 50 times, how many heads would turn up?
      Expected number of heads = $N \times p = 50 \times 0.5 = 25$

- Given x (# of correct predictions) or equivalently, acc=x/N, and N (# of test instances),

  Can we predict p (true accuracy of model)?

Confidence Interval for Accuracy

☐ For large test sets (N > 30),

– acc has a normal distribution with mean p and variance p(1-p)/N

$$P(Z_{\alpha/2} < \frac{acc - p}{\sqrt{p(1-p)/N}} < Z_{1-\alpha/2})$$

$$= 1 - \alpha$$

Area = 1 - α

$Z_{\alpha/2}$ $Z_{1-\alpha/2}$

☐ Confidence Interval for p:

$$p = \frac{2 \times N \times acc + Z_{\alpha/2}^2 \pm \sqrt{Z_{\alpha/2}^2 + 4 \times N \times acc - 4 \times N \times acc^2}}{2(N + Z_{\alpha/2}^2)}$$

Confidence Interval for Accuracy

☐ Consider a model that produces an accuracy of 80% when evaluated on 100 test instances:

– N=100, acc = 0.8

– Let 1-α = 0.95 (95% confidence)

– From probability table, $Z_{\alpha/2}$=1.96

1-α	Z
0.99	2.58
0.98	2.33
0.95	1.96
0.90	1.65

N	50	100	500	1000	5000
p(lower)	0.670	0.711	0.763	0.774	0.789
p(upper)	0.888	0.866	0.833	0.824	0.811

Comparing Performance of 2 Models

☐ Given two models, say M1 and M2, which is better?

- M1 is tested on D1 (size=n1), found error rate = e_1
- M2 is tested on D2 (size=n2), found error rate = e_2
- Assume D1 and D2 are independent
- If n1 and n2 are sufficiently large, then

$$e_1 \sim N(\mu_1, \sigma_1)$$
$$e_2 \sim N(\mu_2, \sigma_2)$$

- Approximate: $\hat{\sigma}_i = \dfrac{e_i(1-e_i)}{n_i}$

Comparing Performance of 2 Models

- To test if performance difference is statistically significant: d = e1 – e2
  - d ~ $N(d_t, \sigma_t)$ where d_t is the true difference
  - Since D1 and D2 are independent, their variance adds up:

$$\sigma_t^2 = \sigma_1^2 + \sigma_2^2 \cong \hat{\sigma}_1^2 + \hat{\sigma}_2^2$$

$$= \frac{e1(1-e1)}{n1} + \frac{e2(1-e2)}{n2}$$

  - At (1-α) confidence level, $d_t = d \pm Z_{\alpha/2} \hat{\sigma}_t$

An Illustrative Example

- Given: M1: n1 = 30, e1 = 0.15

      M2: n2 = 5000, e2 = 0.25

- d = |e2 – e1| = 0.1   (2-sided test)

$$\sigma_t^2 = \frac{0.15(1-0.15)}{30} + \frac{0.25(1-0.25)}{5000} = 0.0043$$

- At 95% confidence level, $Z_{\alpha/2}$=1.96

$$d_t = 0.100 \pm 1.96 \times \sqrt{0.0043} = 0.100 \pm 0.128$$

=> Interval contains 0 => difference may not be
                  statistically significant
(2 mô hình không khác nhau nhiều)