DSA5101
Decision Trees Classification & Model Evaluation

Li Xiaoli

National University of Singapore

Information about Instructor

- Li Xiaoli, Principal Scientist & Dept Head at Machine Intellection Department, Institute for Infocomm Research (**I2R**), **A*STAR**
- Adj Prof at NTU
- Email: xlli@i2r.a-star.edu.sg (**I2R email**)

  xlli@ntu.edu.sg **(NTU email)**
- Webpage: Google Li Xiaoli

  https://personal.ntu.edu.sg/xlli/

Lecture Schedule

Lecture 2 (Li XL)	22 Aug	Introduction to ML (clustering and classification methods)
Lecture 3	29 Aug	ML methods and programming:
Lecture 4	5 Sept	Project 1
Lecture 5	12 Sept	

Project: will be announced in LumiNUS.

Outline

- Definition of Classification
- Decision Tree Techniques
- Model Evaluation

Classification: Definition

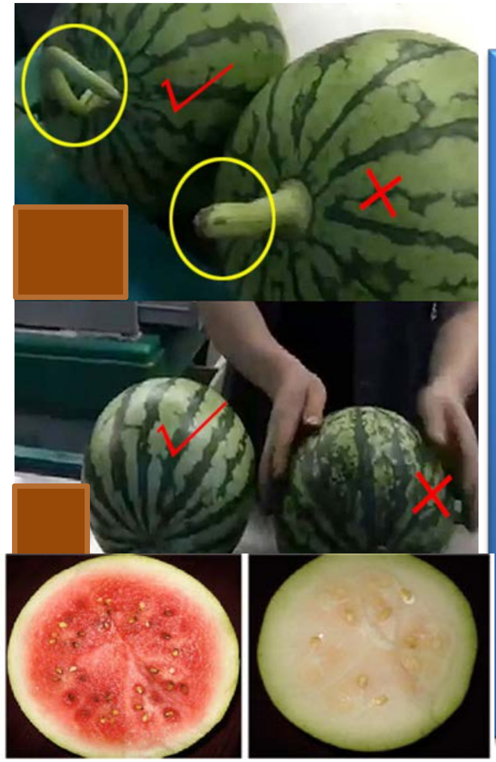
- Given a collection of records (*training set:* **History**)
  - Each record contains a set of *attributes/features/variables*, one of the key attributes is the *class label* (*target attribute/feature/variable*).
- Find a *model* for class variable as a function of the values of other attributes.
- Goal: previously unseen records should be assigned a class as **accurately** as possible (test set: consist of new/future examples).

Prediction based on the past examples (historical data).
Let History tell us the future!

Example: Watermelon Ripeness Determination

Watermelon Training set

ID	Color	Shape of Root	Sound	Ripeness
1	Green	Curl	Dull	Y
2	Black	Curl	Dull	Y
3	Green	Stiff	crisp	N
4	Black	Stiff	Dull	N



We go to supermarket to buy a watermelon. Which one is a good one? You use the model in your brain to predict. Perhaps you will use size, colour, shape root, and acoustic signal. If your model is not accurate, you cannot find good ones☺ The reason could be the quality of your sensors, less training data etc. The sales guy could be much more accurate.

There are 3 normal features and 1 target feature. This is a binary classification problem. Objective is to learn an accurate model to help us to pick good watermelon

If we have many training data and quality sensors, we can build a very accurate model using data mining. In addition, we could learn rules/insights, e.g. which feature is the most important one and which two could be used together to build a better model

Perhaps to build a mobile app to make $

Example in clinical diagnostic: Classification Data Set

Attributes

Target attribute

id number	Clump Thickness	Uniformity of Cell Size	Uniformity of Cell Shape	Marginal Adhesion	Single Epithelial Cell Size	Bare Nuclei	Bland Chromatin	Normal Nucleoli	Mitoses	Class: (2 for benign, 4 for malignant)
ID1	5	1	1	1	2	1	3	1	1	2
ID2	5	4	4	5	7	10	3	2	1	2
ID3	3	1	1	1	2	2	3	1	1	4
ID4	8	10	10	8	7	10	9	7	1	4

Training Examples

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

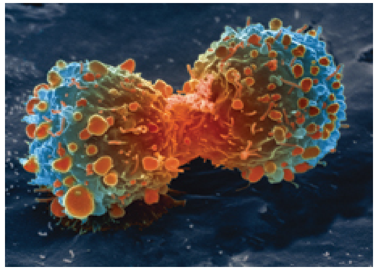
f (Clump Thickness, Uniformity of Cell Size, …, Mitoses) = **Class**

***f*(4, 6, 5, 6, 8, 9, 2, 4, 1)=?** A test example

More training data will usually lead to better classification performance

Examples of Classification Task

- Predicting if machines/tools in manufacturing/healthcare is in normal or faulty status
- Predict product quality/yield
- Predicting tumor cells as *benign* or *malignant*
- Classifying credit card transactions as *legitimate* or *fraudulent*
- *Classifying an email is spam or not*
- Reduce cost of mailing by *targeting* a set of consumers likely to buy a new cell-phone product (direct marketing).
- Categorizing news stories as *finance, weather, entertainment, sports*, etc
- Classifying overall opinions of a product as positive or negative
- ……



Classification Techniques

- K Nearest Neighbour
- Logistic Regression
- Tree based Methods (decision tree, random forest, Gradient Boost, XGBoost)
- Naïve Bayes Classifiers
- Support Vector Machines
- Neural Networks and Deep Learning
- Ensemble Classification
- ……

Outline

- Definition of Classification
- Decision Tree Techniques
- Model Evaluation

Decision Tree Induction

- Many Algorithms:
  - Hunt's Algorithm
  - CART
  - ID3
  - C4.5
  - C5
  - SLIQ
  - SPRINT

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

Learn a model for discriminating between examples between different classes (cheaters vs non-cheaters)

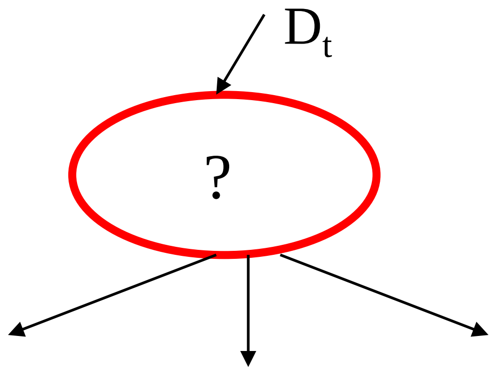
Objective: learn a classification tree model from training data and then apply it to predict future test data as accurate as possible.

General Structure of Hunt's Algorithm

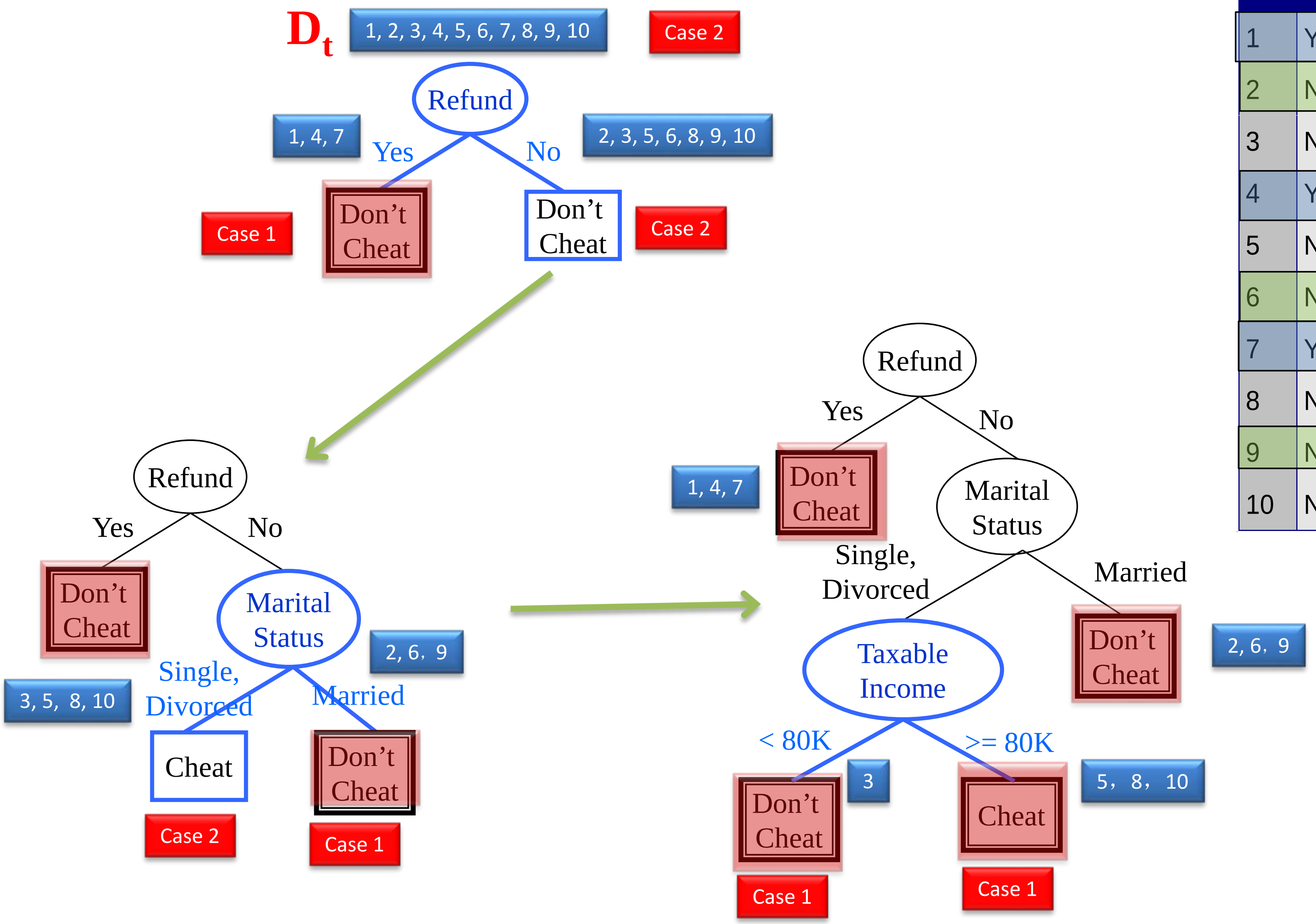
- Let D_t be the set of **training** records that reach a node **t**
- **General Procedure:**
  - Case 1: If D_t contains records that belong the same class y_t, then **t** is a **leaf** node labeled as y_t (pure, consistent)
  - Case 2: If D_t contains records that belong to more than one class, use an **attribute test** to **split** the data into smaller subsets.
  - **Recursively apply** the above procedure (case 1 and 2) to each subset.

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**



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

Tree Induction

- Greedy strategy
  - Split the records based on **an attribute test** that *optimizes certain criterion*.
- Unsolved Issues
  - Determine how to **split** the records
    - How to determine the **best split** (choose the best features/attributes to split the tree)?
    - How to specify the **attribute test condition** (attribute <=value, attribute>value)?
  - Determine when to **stop splitting**

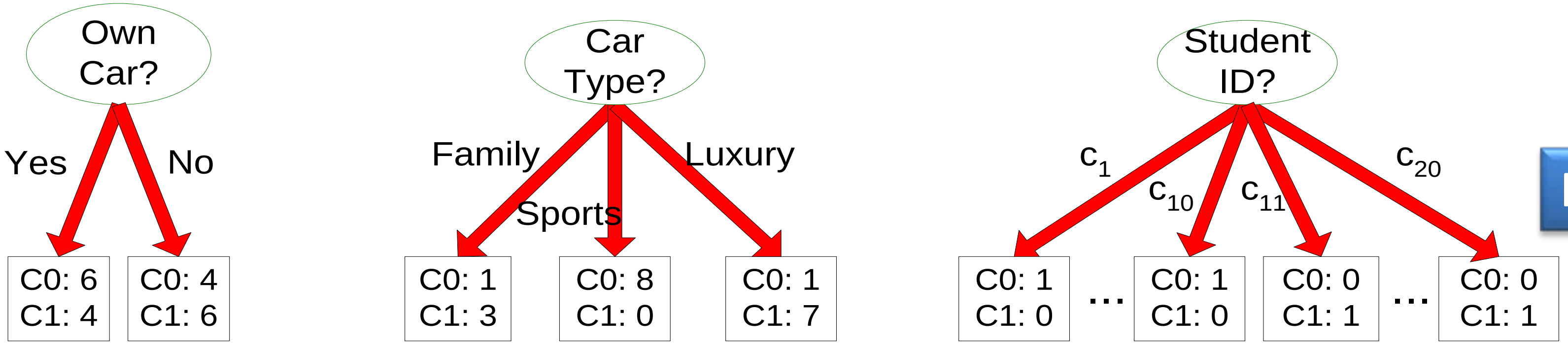
How to determine the Best Split? Intuition

	Student ID	Own Car	Car Type	…	Class
1	G4287203	Yes	Family	…	C0
2	G4287387	No	Sports	…	C0
3	G4287472	No	Luxury	…	C1
4	G4287593	Yes	Family	…	C1
…	…	…	…	…	…
20	…	…	…	…	…

Before Splitting:

10 records of class 0, i.e. C0: 10

10 records of class 1, i.e. C1: 10



Features

Feature values

Which features/attributes, i.e. Own Car, Car Type, Student ID, do you want to choose?

The split can help us to distinguish different classes. Note we will choose the best feature based on its values' *overall* differentiating capabilities

How to determine the Best Split

- Greedy approach:
  – Nodes with homogeneous class distribution are preferred
- Need a measure of node impurity:

C0: 5
C1: 5

Non-homogeneous,

High degree of impurity

Ambiguous : C0 or C1?

C0: 9
C1: 1

Homogeneous,

Low degree of impurity

More confident to infer C0

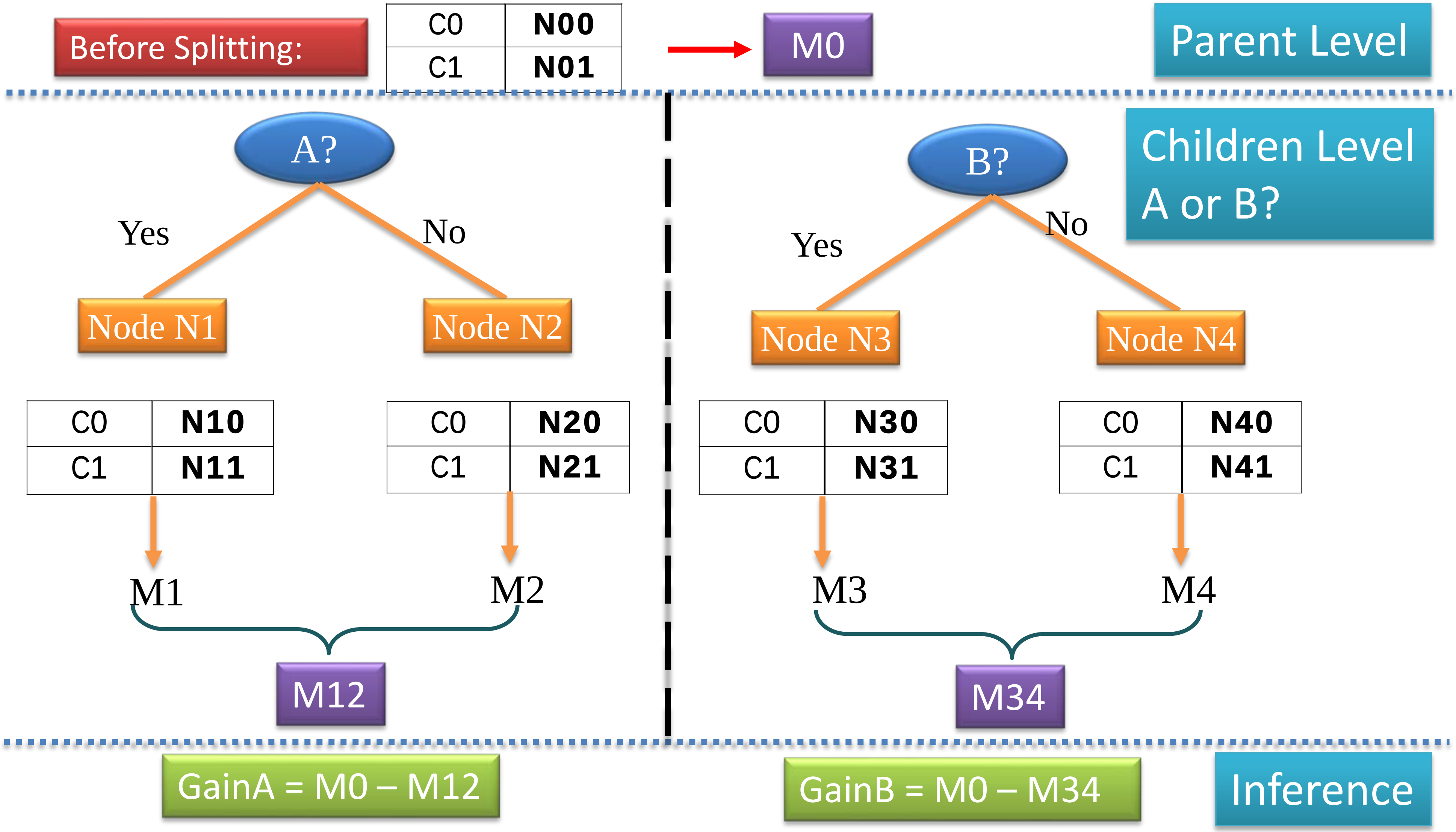
Should provide certainty and confidence for future test cases

- We also need to measure children impurity (each feature will split records into multiple child nodes)

Three Measures of Node Impurity

1. Gini Index
2. Entropy
3. Misclassification error

How to Find the Best Split (feature selection)?



Which attribute (A or B) should we choose? We choose the one that can reduce more impurity and bring more gain. If GainA is bigger, we choose A; B otherwise.

Three measures of Impurity:
1. GINI Index for a node

- Gini Index for a given node t:

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

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

(in the following example, class *j* is class c0 and c1).

C0	0
C1	6
Gini=0.000	

C0	1
C1	5
Gini=0.278	

C0	2
C1	4
Gini=0.444	

C0	3
C1	3
Gini=0.500	

p(c0|t)=0
p(c1|t)=1

p(c0|t)=1/6
p(c1|t)=5/6

p(c0|t)=2/6
p(c1|t)=4/6

p(c0|t)=3/6 =0.5
p(c1|t)=3/6 =0.5

Examples for computing node's GINI

How to compute Gini?

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

Same results (exchange)

C0	**0**
C1	**6**

P(C0) = 0/6 = 0 P(C1) = 6/6 = 1

Gini = 1 – $P(C0)^2$ – $P(C1)^2$ = 1 – 0 – 1 = 0

C0	**1**
C1	**5**

P(C0) = 1/6 P(C1) = 5/6

Gini = 1 – $(1/6)^2$ – $(5/6)^2$ = 0.278

C0	**2**
C1	**4**

P(C0) = 2/6 P(C1) = 4/6

Gini = 1 – $(2/6)^2$ – $(4/6)^2$ = 0.444

C0	**3**
C1	**3**

P(C0) = 3/6 P(C1) = 3/6

Gini = 1 – $(3/6)^2$ – $(3/6)^2$ = 0.5

C0	**6**
C1	**0**

P(C0) = 1 P(C1) = 0

Gini = 0

C0	**5**
C1	**1**

P(C0) = 5/6 P(C1) = 1/6

Gini = 0.278

C0	**4**
C1	**2**

C0	**3**
C1	**3**

Which nodes do you like?

What are the maximum and minimum GINIs for two classes (C0, C1)?

[0, 0.5]

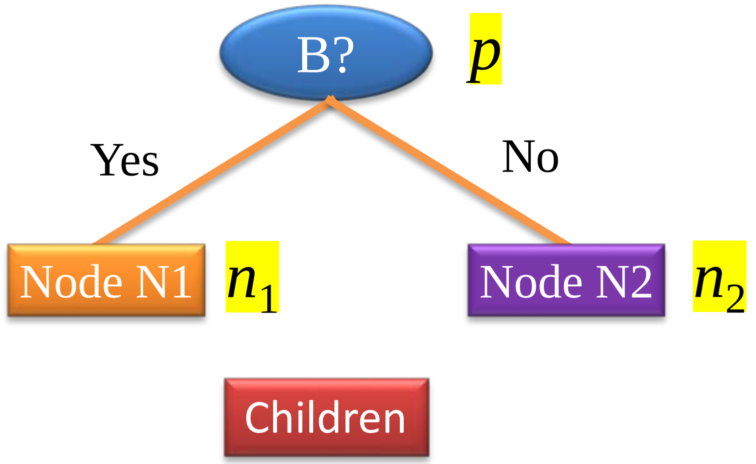
Splitting Based on **GINI**: How to assess the **quality of splitting For all children**

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



The quality of splitting GINI_{split} is the overall quality of all the children.
In this example, we first compute quality (Gini index) of Node 1 and Node 2, i.e., GINI(n1) and GINI(n2). Then merge them into GINI_{split} based on their record ratio n_i/n. *We get* $GINI_{split}=n_1/n*GINI(1)+n_2/n*GINI(2)$,

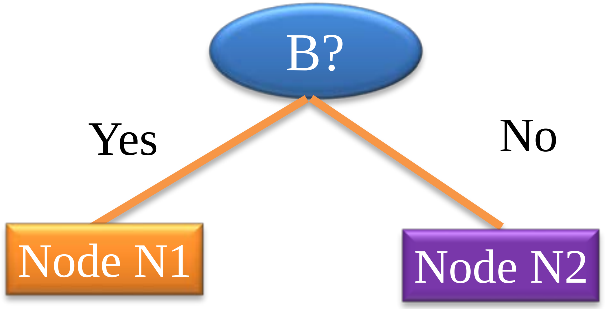
– *Larger* and *Purer* Partitions are sought for

Splitting Based on GINI for **Binary** Attributes

Assess splitting quality by computing gain

7 records

C0	5
C1	2



5 records

C0	1
C1	4

Parent Gini

	Parent
C0	6
C1	6
Gini = 0.500	

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

Gini(N1)
$= 1 - (5/7)^2 - (2/7)^2$
$= 0.4082$

Gini(N2)
$= 1 - (1/5)^2 - (4/5)^2$
$= 0.3200$

Count matrix for children

	N1	N2
C0	5	1
C1	2	4
Gini=0.3715		

After use attribute B

$GINI_{split}$=Gini(Children)
= 7/12 * 0.4082 +
5/12 * 0.3200
= 0.3715

What is the gain? Why children are better? What is the predicted label for node N1 and N2 respectively?

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
C0	1	2	1
C1	4	1	1
Gini	0.393		

Two-way split
(find best partition of values)

	CarType	
	{Sports, Luxury}	{Family}
C0	3	1
C1	2	4
Gini	0.400	

	CarType	
	{Sports}	{Family, Luxury}
C0	2	2
C1	1	5
Gini	0.419	

Which split do you want to choose?

Given we have same parent Gini index, the split with smallest Gini index (i.e. largest gain) will be preferred

2. Alternative Splitting Criteria based on **Information gain**

- Entropy at a given node t:

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

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

- Measures homogeneity of a node.
  - Maximum ($\log_2 n_c$) when records are **equally distributed** among all classes implying the **least** information (n_c is the number of class)
  - Minimum (0.0) when all records belong to one class, implying the **most** information
- Entropy based computations are similar to the GINI index computations

Examples for computing Entropy

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

C0	**0**
C1	**6**

P(C0) = 0/6 = 0   P(C1) = 6/6 = 1

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

C0	**1**
C1	**5**

P(C0) = 1/6   P(C1) = 5/6

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

C0	**2**
C1	**4**

P(C0) = 2/6   P(C1) = 4/6

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

C0	**3**
C1	**3**

P(C0) = 3/6   P(C1) = 3/6

Entropy = – (3/6) $\log_2$ (3/6) – (3/6) $\log_2$ (3/6) = 1

What are the maximum and minimum entropies for two classes?

[0, 1]

Splitting Based on **Information gain**

- Information Gain: **Parent quality** **Children quality**

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

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, i.e. maximizes ***$GAIN_{split}$***
  – Used in ID3 and C4.5
  – Disadvantage:
    - Tends to prefer splits that result in large number of partitions, each being small but pure (like student ID). Imaging when each child is small and pure, the overall split quality will be high, leading to high ***$GAIN_{split}$***

Splitting Based on Information gain

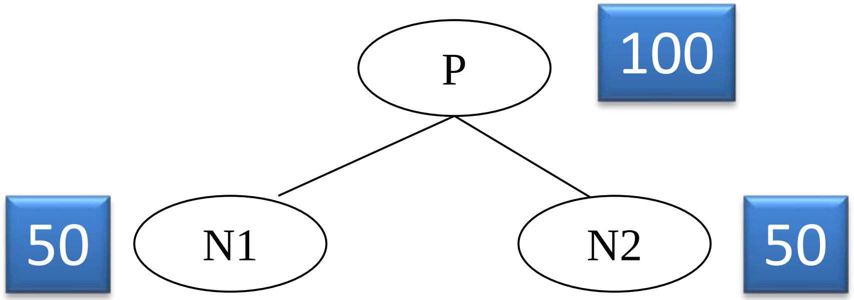
- Gain Ratio: How about we normalize ***GAIN**$_{split}$* by SplitINFO

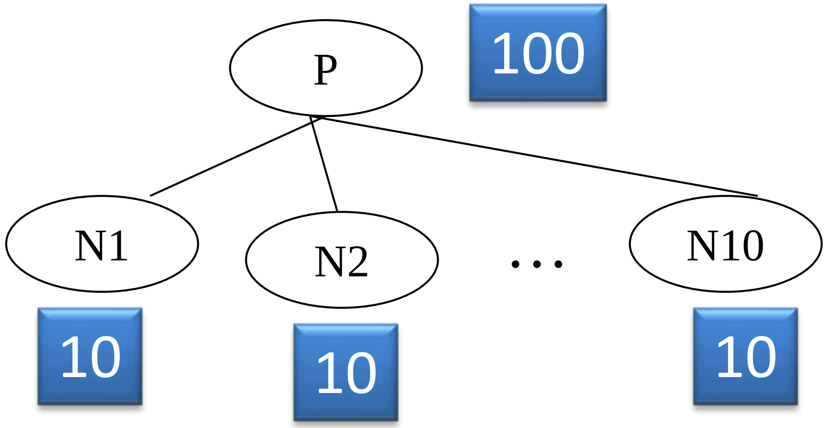
$$GainRATIO_{split} = \frac{GAIN_{Split}}{SplitINFO}$$

- How to compute 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



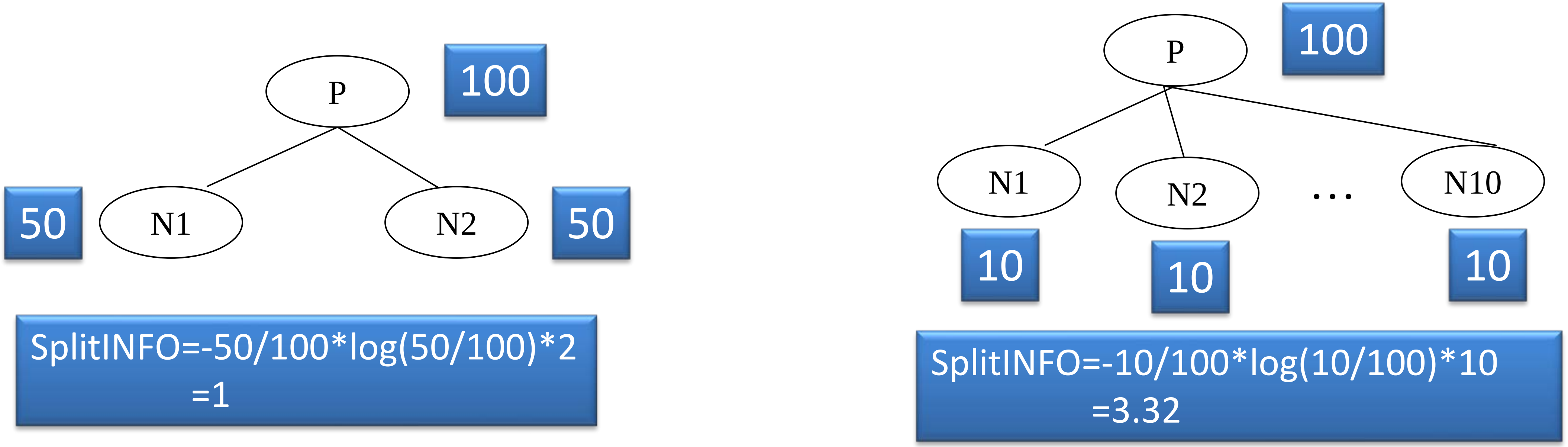


SplitINFO=-50/100*log(50/100)*2
=1

SplitINFO=-10/100*log(10/100)*10
=3.32

Just for illustration, each node could have different number of records

Splitting Based on INFO...



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

– 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

3. Splitting Criteria based on **Classification Error**

- Classification error at a node t :

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

- Measures misclassification error made by a node.
  - Maximum (1 - 1/n_c) when records are equally distributed among all classes, implying least interesting information
  - Minimum (0.0) when all records belong to one class, implying most interesting information

Why it is called classification error? It is an error estimation when classifying the node into majority class and all the other class counts are errors

Examples for Computing Error

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

C0	**0**
C1	**6**

P(C0) = 0/6 = 0   P(C1) = 6/6 = 1

Error = 1 – max (0, 1) = 1 – 1 = 0

C0	**1**
C1	**5**

P(C0) = 1/6   P(C1) = 5/6

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

We classify into C2 class, so 1/6 error

C0	**2**
C1	**4**

P(C0) = 2/6   P(C1) = 4/6

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

C0	**3**
C1	**3**

P(C0) = 3/6   P(C1) = 3/6

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

What are the maximum and minimum classification errors for two classes?

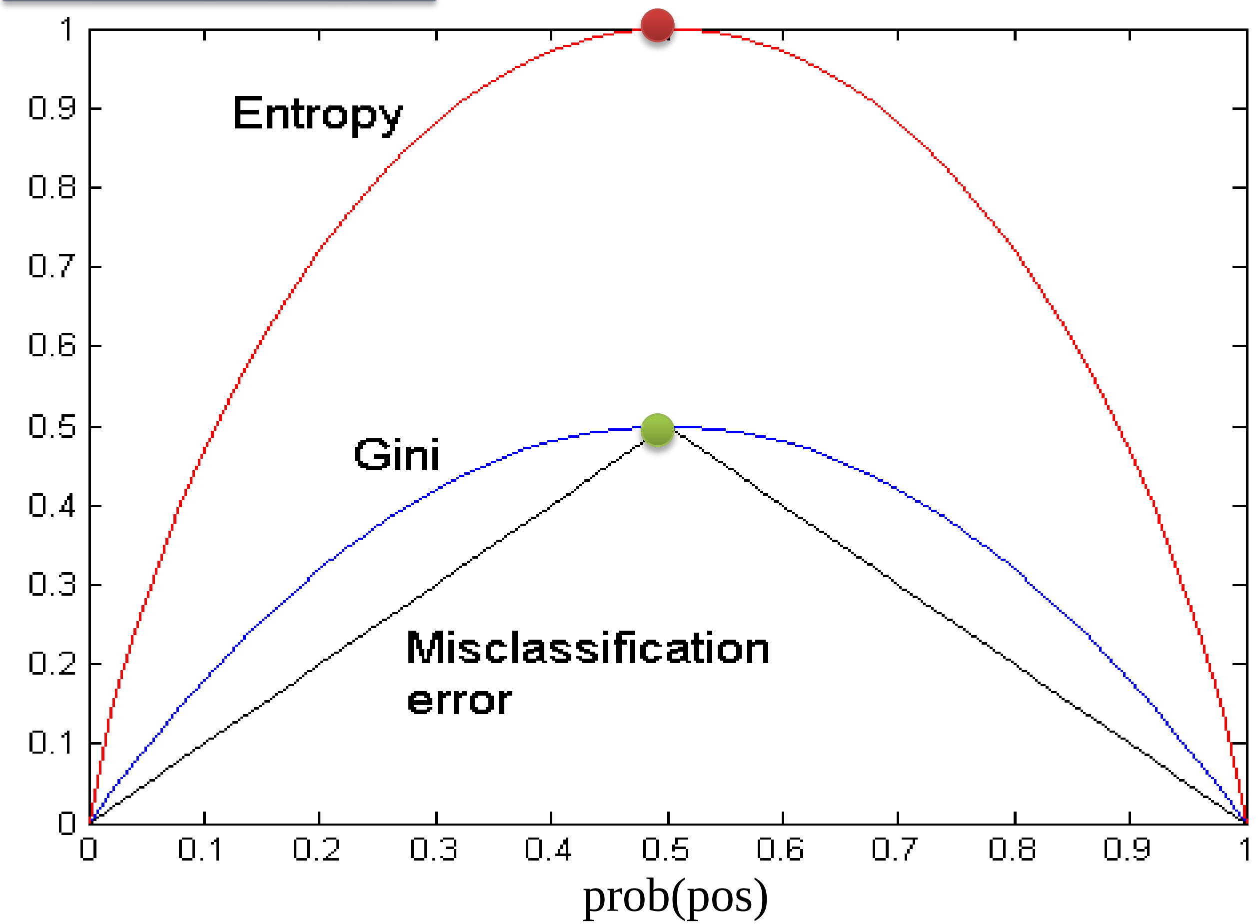
[0, 0.5]

Comparison among Splitting Criteria

For a 2-class classification problem: we only have pos and neg class.

And Pro(neg) =1-prob(pos) You can treat pos=C0, while neg=C1

3 criteria's values



The scores among the three criteria could be different

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

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

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

When prob(pos)=0.5, then prob(neg)=0.5

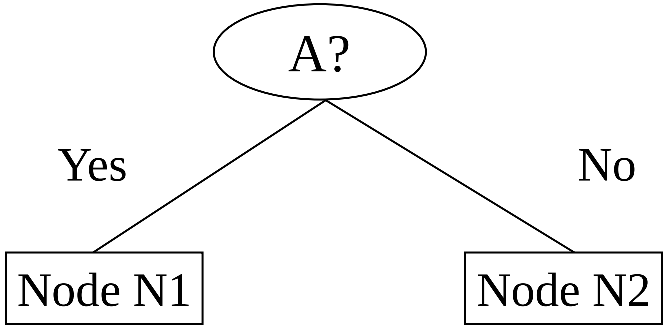
Entropy = – (0.5) $\log_2$ (0.5) – (0.5) $\log_2$ (0.5) = 1

Gini = 1 – $(0.5)^2$ – $(0.5)^2$ = 0.5

Error = 1 – max (0.5, 0.5) = 1 – 0.5 =0.5

Let us show difference: Misclassification Error vs Gini

1. Misclassification Error



	Parent
C1	7
C2	3

Error=1-max(7/10, 3/10)=**3/10**

Error (N1)
= 1 – max (3/3, 0/3)
= 0

Error(N2)
= 1 – max(4/7,3/7)
= 3/7

	N1	N2
C1	3	4
C2	0	3
Missclassification error=0.3		

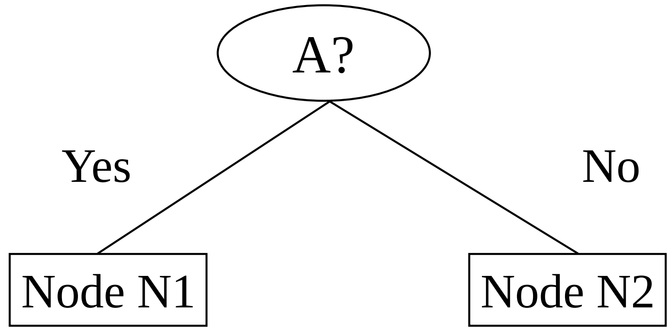
Error(Children)
= 3/10 * 0
+ 7/10 * 3/7
= 3/10=0.3

As there is no gain, we will stop at the parent level, which may not be suitable

Let us show difference: Misclassification Error vs Gini

2. Let us try Gini index

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



	Parent
C1	7
C2	3
Gini = 0.42	

Gini(Parent)
$= 1 - (7/10)^2 - (3/10)^2$
$= 0.42$

Gini(N1)
$= 1 - (3/3)^2 - (0/3)^2$
$= 0$

Gini(N2)
$= 1 - (4/7)^2 - (3/7)^2$
$= 0.489$

	N1	N2
C1	3	4
C2	0	3
Gini=0.342		

Gini(Children)
$= 3/10 * 0$
$+ 7/10 * 0.489$
$= 0.342$

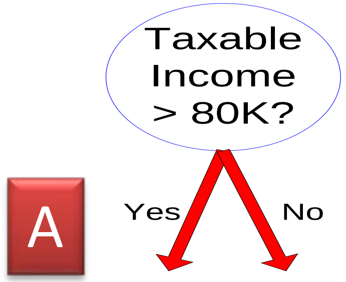
Gini score improves !

The gain is 0.42-0.342>0, we will continue partitioning the tree using feature A

In other words, different criteria could lead to different trees

Continuous Attributes: Computing Gini Index

- Use Binary Decisions based on one value
- Several choices for the splitting value v
  - Number of possible splitting values = Number of distinct values
- Each splitting value v has a count matrix associated with it
  - Class counts in each of the partitions, A <= v and A > v
  - v could be any value (e.g., 80)
- 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

Computing Gini Index for continuous Attributes

- For efficient computation: for each continuous 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 (largest gain)

Sorted Values

Split Positions

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	

Where do these splitting positions come from? E.g. 172

They are the *floor* of the *average* of the two neighboring sorted values, i.e. floor((125+220)/2)=floor(172.5)=172

Why 97 is best 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 determine the best split?
    - How to specify the attribute test condition?
  - Determine when to stop splitting

Stopping Criteria for Tree Induction

Stop expanding a node when all the records belong to the same class

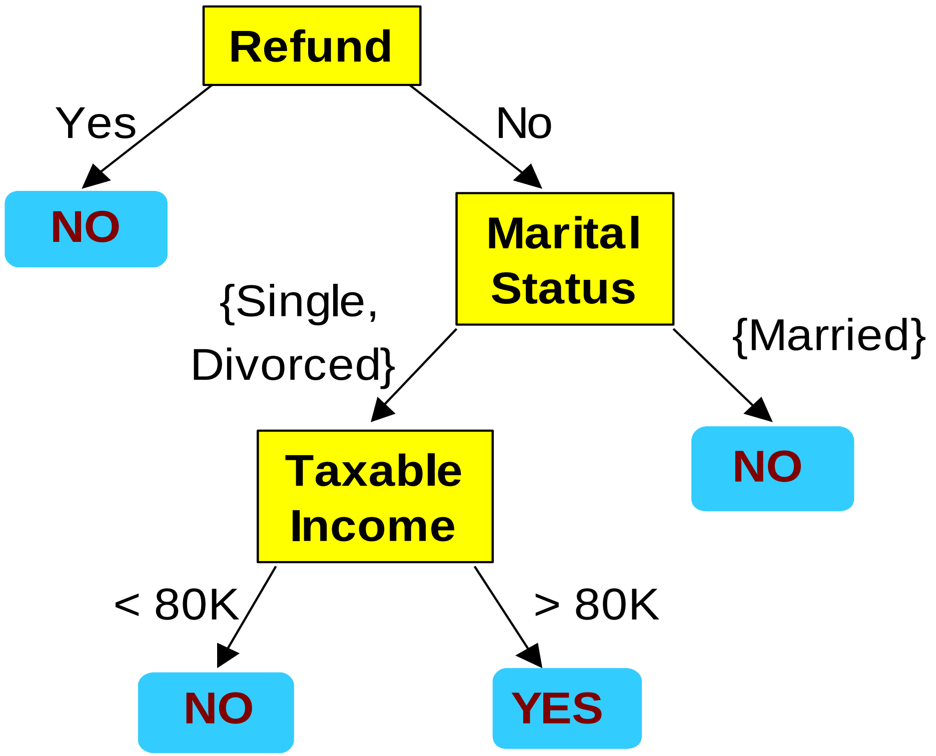
Stop expanding a node when all the records have same/similar attribute values – you have to stop

Early termination when nodes contain small number of records, to avoid overfitting – you fit the training data well, but not necessarily perform well in the test set – we care more about *generalization* performance

Decision Tree Based Classification

- Advantages:
  - Inexpensive to construct
  - Extremely fast at classifying unknown records
  - Easy to interpret
  - Accuracy is comparable to other classification techniques for many data sets

From Decision Trees To Rules



Classification Rules

(Refund=Yes) ==> No

(Refund=No, Marital Status={Single,Divorced}, Taxable Income<80K) ==> No

(Refund=No, Marital Status={Single,Divorced}, Taxable Income>80K) ==> Yes

(Refund=No, Marital Status={Married}) ==> No

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

- Transparent
- Explainable by providing examples in the corresponding nodes
- Adjustable – refine the rules

Outline

- Definition of Classification
- Decision Tree Techniques
- Model Evaluation

Model Evaluation

All models are wrong, but some are useful!

- Wrong because it is a simplification of reality
- Useful if it may reach certain prediction accuracy.
  Many real-world problem needs prediction

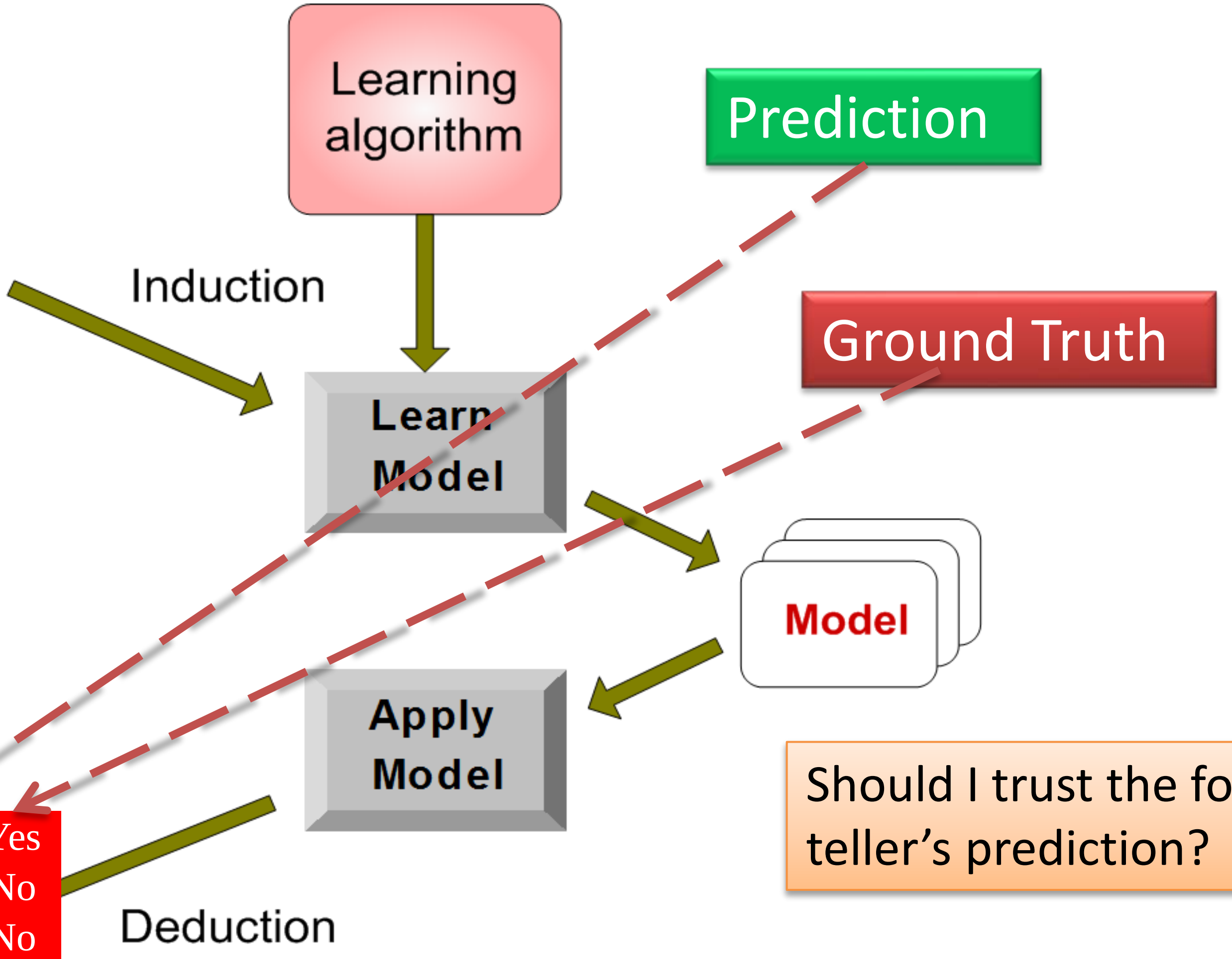
Model Evaluation

- Metrics for Performance Evaluation
  - How to evaluate the performance of a model?
- Methods for Performance Evaluation
  - How to obtain reliable estimates?

Metrics for Performance Evaluation

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

Test Set

Should I trust the fortune teller's prediction?

Test set should not be too small

Need ground truth to compare with the prediction results to do fair evaluation.

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 (element -> #cases in test set):

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

a: TP (true positive)

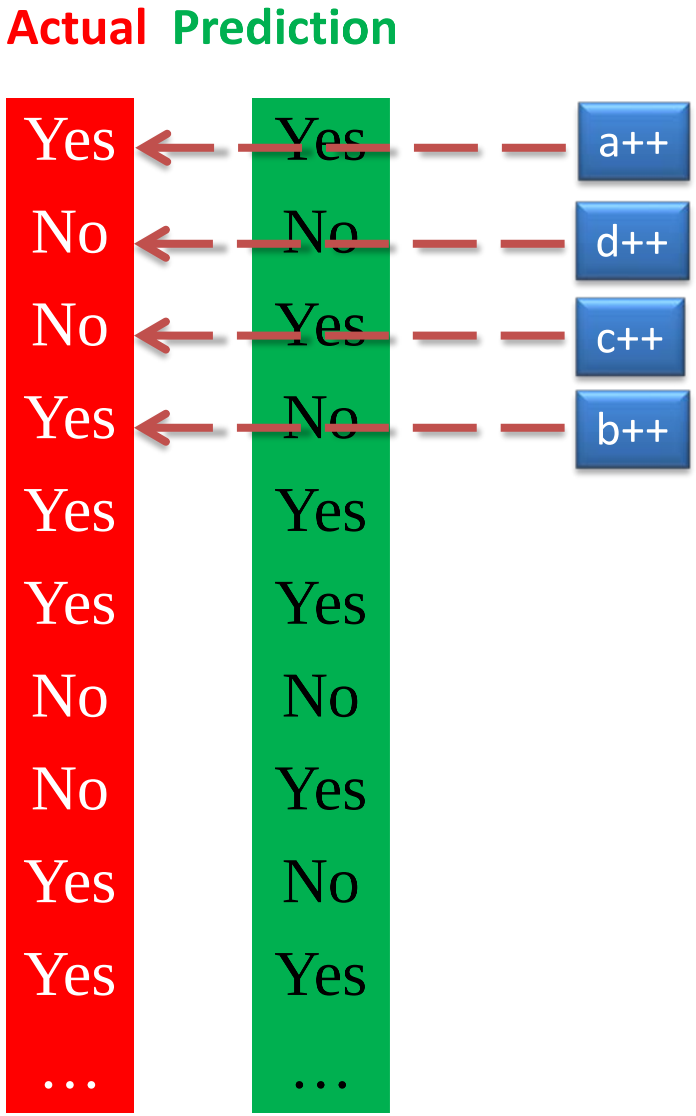
b: FN (false negative)

c: FP (false positive)

d: TN (true negative)

Metrics for Performance Evaluation

- In the test set



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

- Most widely-used metric:

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

- Error Rate = 1- Accuracy

Limitation of Accuracy

- Consider a 2-class problem
  - spam detection
  - fraud detection
  - COVID-9 diagnostic
- Usually negative class = OK class

  positive class = not-OK class
- Assume in the test set
  - Number of negative examples = 9990
  - Number of positive examples = 10

Limitation of Accuracy

- Number of negative examples = 9990
- Number of positive examples = 10
- If model predicts everything to be negative class, accuracy is 9990/(9990+10) = 99.9 %

  (TP=0, TN=9990, FP=0, FN=10)

  – Accuracy is misleading because model does not detect any positive class example
- In the imbalanced cases, accuracy is not really a reliable metric

Cost Matrix

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

C(i|j): Cost of misclassifying class j example as class i

Cost/penalty means how much you need to pay if you suffer misclassification

Cost Matrix in Medical Domain

	PREDICTED CLASS		
ACTUAL CLASS	C(i|j)	**Class=Cancer**	**Class=Normal**
	Class=Cancer	C(**Cancer|Cancer**)	C(**Normal|Cancer)** **99999?**
	Class=Normal	C(**Cancer|Normal**) **100?**	C(**Normal|Normal**)

It is not acceptable to misclassify cancer patients into normal, as it could delay the treatment

C(i|j): Cost of misclassifying class j example as class i

Cost/penalty means how much you need to pay if you suffer misclassification

Computing Cost Performance of Classification

Award

spam emails
fraud cases
disease cases

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

Big penalty

Small penalty

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

Accuracy = 400/500=80%
Cost Performance =
-1*150+100*40+60*1+0*250=3910

Model M_2	PREDICTED CLASS		
ACTUAL CLASS		+	-
	+	160	30
	-	110	200

Accuracy = 360/500=72%
Cost Performance
= -1*160+100*30+1*110+0*200=2950

Precision, Recall and F-measure

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

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

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

Precision: We predict *a+c* cases as positives, out of which *a* cases are correct

Recall: There are a+b positive cases, out of which *a* cases are classified as positive correctly.

	PREDICTED CLASS	
	Class=spam	Class=Normal
ACTUAL CLASS Class=spam	40	60
ACTUAL CLASS Class=Normal	100	5000

spam emails

fraud cases

disease cases

What is the precision and recall (by default wrt positive/spam class)?

Precision, Recall and F-measure

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

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

	PREDICTED CLASS		
		Class=spam	Class=Normal
ACTUAL CLASS	Class=spam	40 (a)	60 (b)
	Class=Normal	160 (c)	5000 (d)

- **p= a/(a+c)=40/(40+160)=20%**
- **r=a/(a+b)=40/(40+60)=40%**

$$\text{F - measure (F)} = \frac{2rp}{r+p} = \frac{2*0.4*0.2}{0.4+0.2} = 0.267$$

For imbalanced cases, precision/recall/F-measure wrt minority class are good metrics

Matthews correlation coefficient

- MCC is a measure of the quality of binary (two-class) classifications, introduced by biochemist Brian W. Matthews in 1975.

$$\mathrm{MCC} = \frac{TP \times TN - FP \times FN}{\sqrt{(TP+FP)(TP+FN)(TN+FP)(TN+FN)}}$$

- If any of the *four sums* in the denominator is 0, then MCC=0. MCC is useful when the two classes are of very different sizes

Matthews, B. W. (1975). "Comparison of the predicted and observed secondary structure of T4 phage lysozyme". Biochimica et Biophysica Acta (BBA) - Protein Structure. ***405*** *(2): 442–451.*

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

Model Evaluation

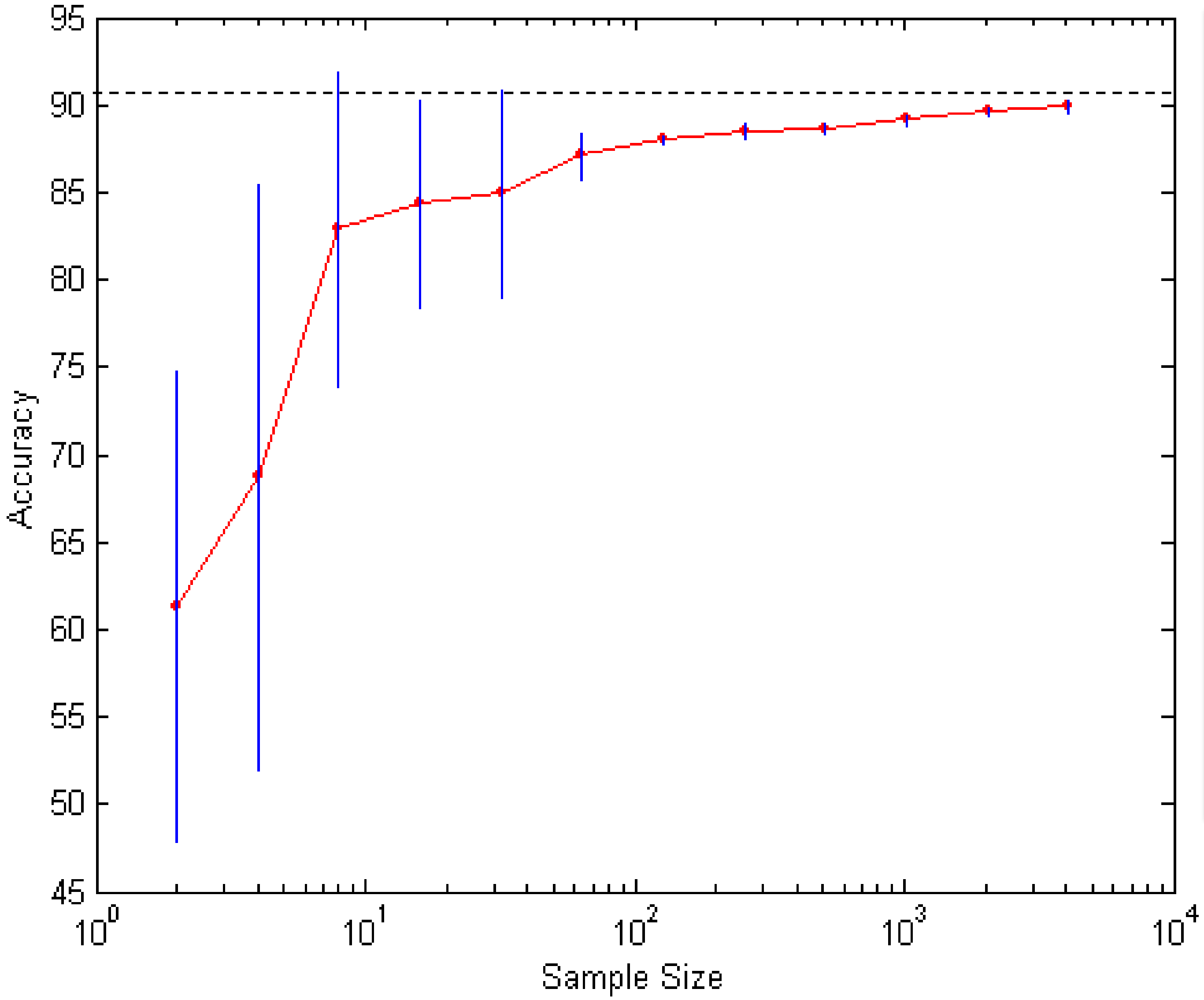
- Metrics for Performance Evaluation
  - How to evaluate the performance of a model?

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

Methods for 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 (easy or challenging problems?)
  - Cost of misclassification (which class do you care more?)
  - **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.), e.g. 10, 20, 30
  - Geometric sampling (Provost et al,), e.g. 2, 4, 8, 16, 32,…

Effect of small sample size:

- Bias in the estimate
- Variance of estimate

Methods of Estimation

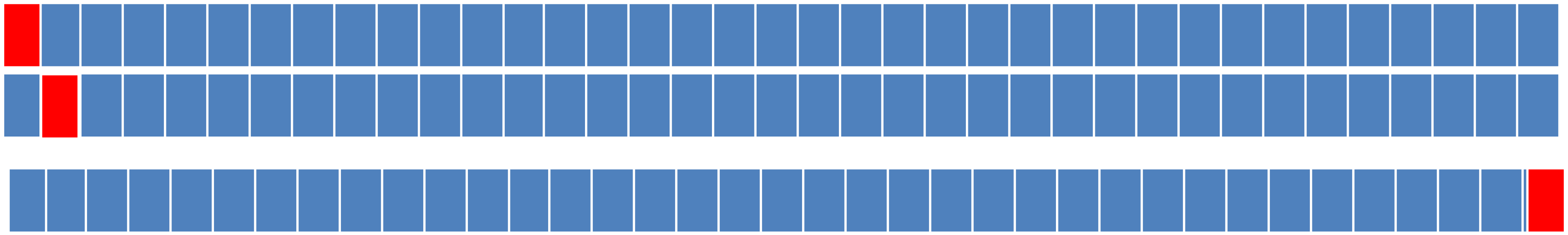
- Holdout
  - Reserve 2/3 for training and 1/3 for testing
- Random subsampling
  - Repeated holdout
- Cross validation (preferred way for estimation, used it together with metrics)
  - Partition data into k disjoint subsets
  - k-fold: train on k-1 partitions, test on the remaining one
  - Leave-one-out: k=n
- Stratified sampling (dividing members of the population into homogeneous subgroups before sampling; every group will have good representation)

Cross Validation

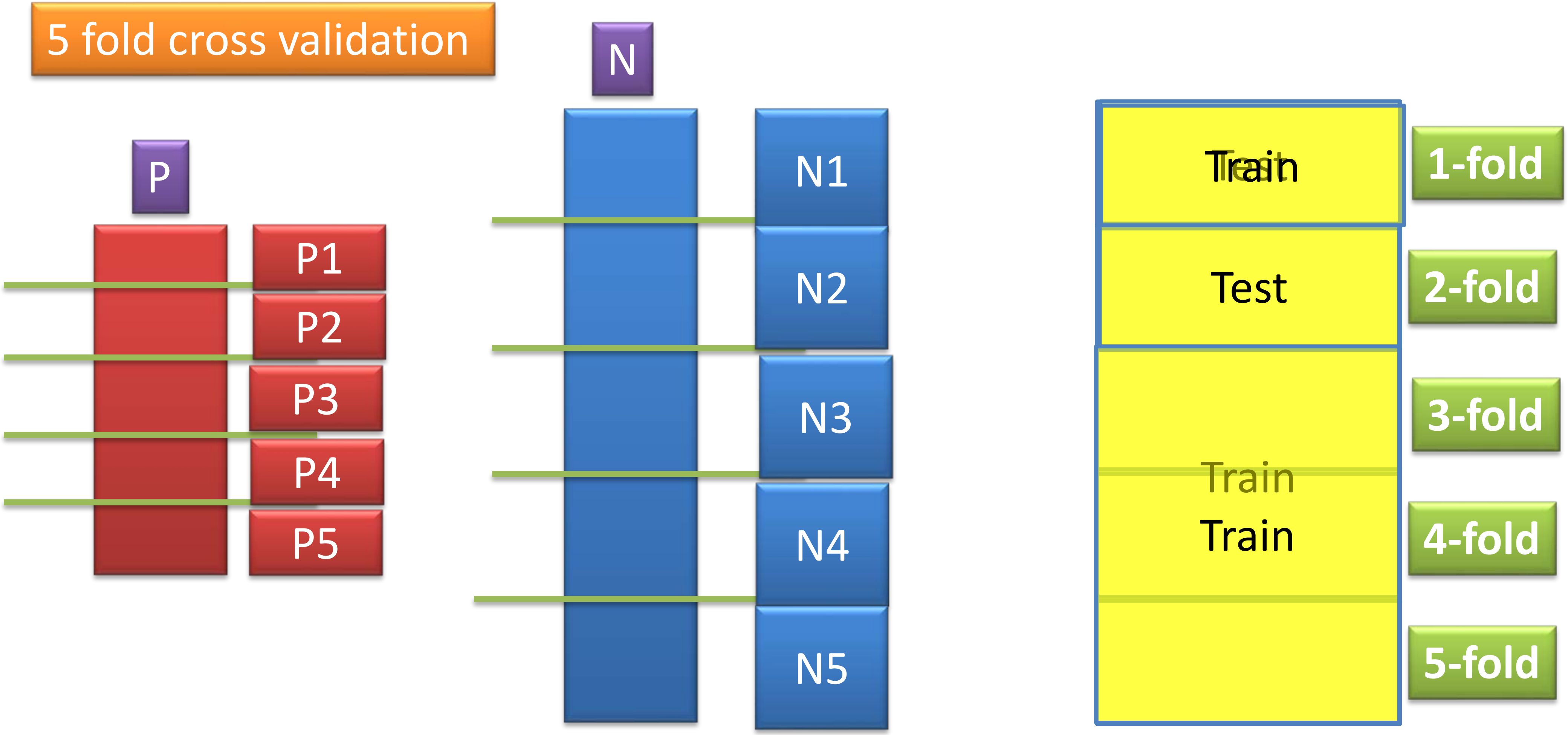
5-fold cross validation

- Divide samples into k roughly equal disjoint parts
- Each part has similar proportion of samples from different classes
- Use each part to test other parts
- Average accuracy and F-measure etc

Leave-one-out cross validation



Each part has similar proportion of samples from different classes



We use binary classification– this applies to multi-class classification

Summary

- Definition of Classification

  Learn from *past* experience/labels, and use the learned knowledge to classify *new* data

- Decision Tree Techniques

  A decision tree is a flowchart-like structure in which each *internal node* represents a "test" on an attribute, each *branch* represents the outcome of the test, and each *leaf node* represents a class label.

- Model Evaluation (Metrics for Performance Evaluation)

  - Confusion Matrix
  - Accuracy/Error Rate
  - Cost Performance
  - Precision, Recall and F-measure

- Model Evaluation (Methods for Performance Evaluation)

Thank You

Contact: xlli@i2r.a-star.edu.sg if you have questions