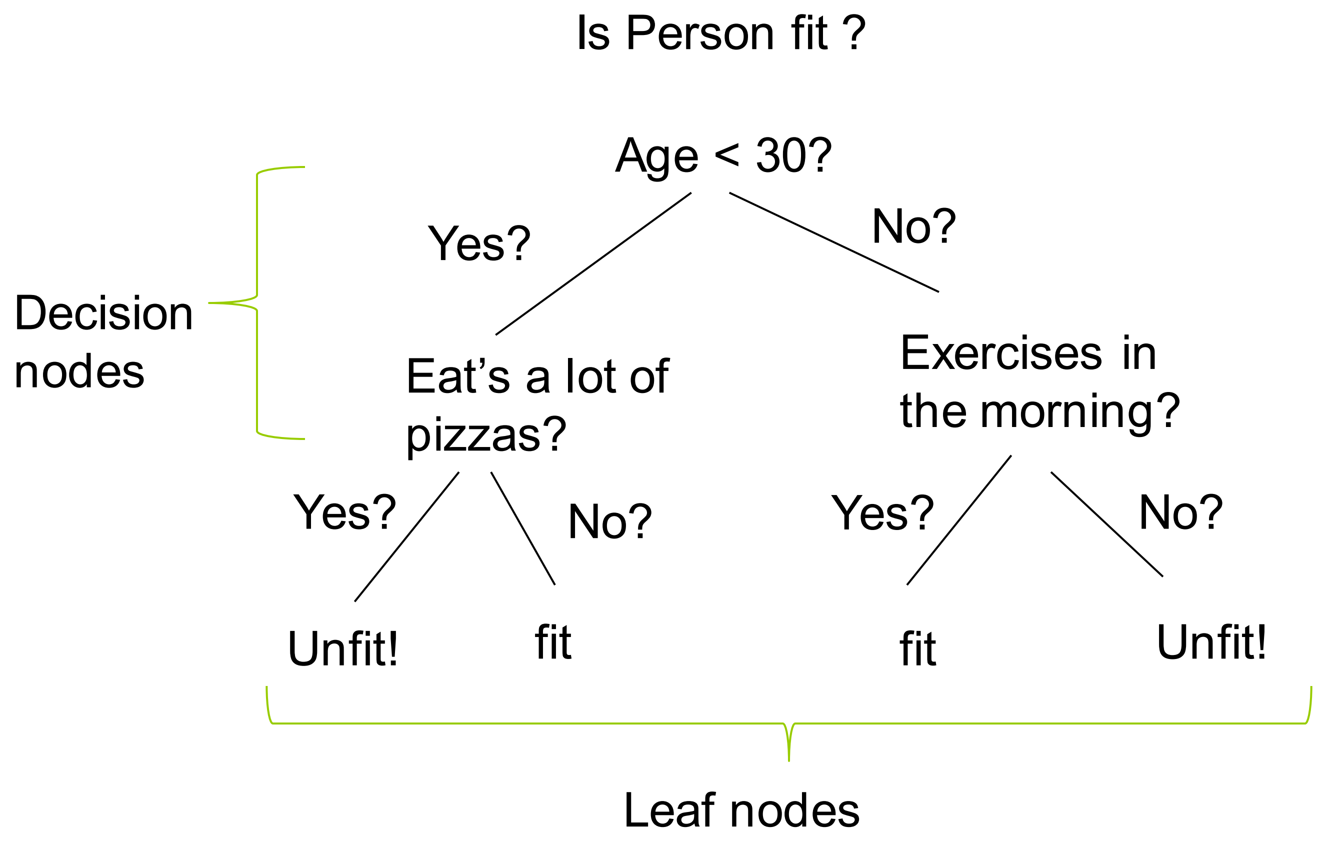
Decision Trees

Prof. G Panda, FNAE,FNASc,FIET(UK)
IIT Bhubaneswar

Introduction

- Decision Tree is a type of Supervised Machine Learning algorithm where decision is taken based on the tree built by data where it is continuously split according to certain parameter.
- The tree can be explained by two entities namely decision nodes and leaves.
- The leaves are the decisions or the final outcomes.
- The Decision nodes are where data is split.

Introduction



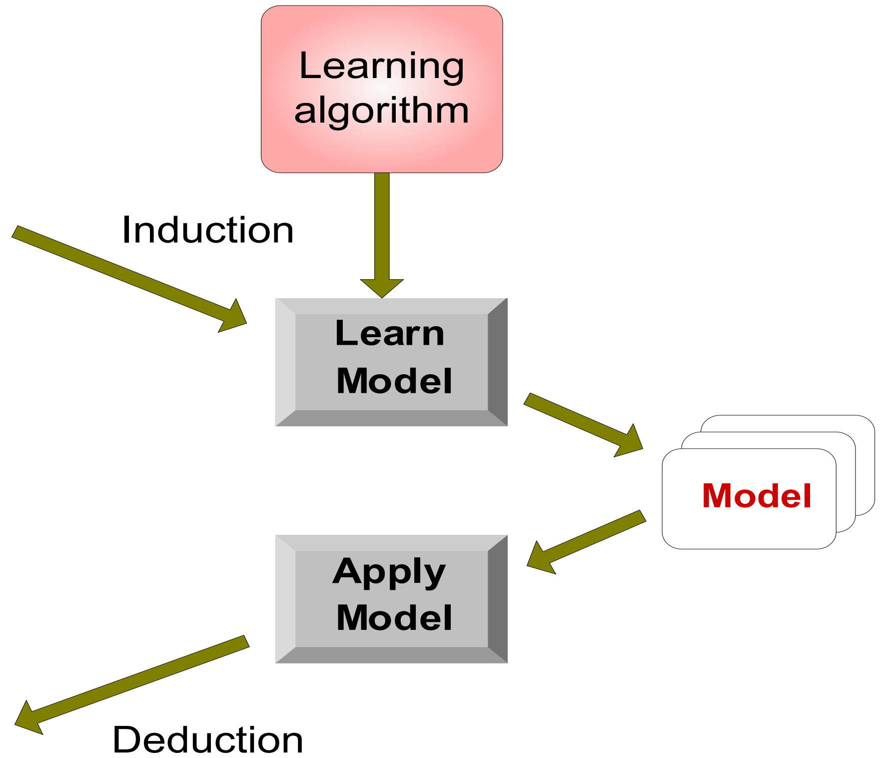
Illustrating Classification Task

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

Test Set



Examples of Classification Task

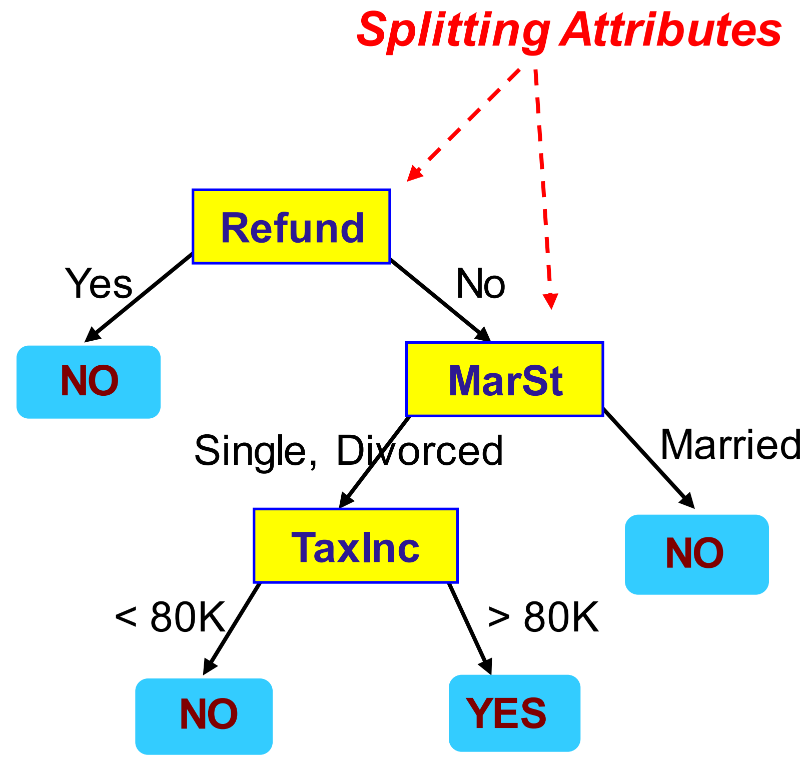
- Predicting tumor cells as benign or malignant

- Classifying credit card transactions as legitimate or fraudulent

- Categorizing news stories as finance, weather, entertainment, sports, etc

Example of a Decision Tree

Tid categorical	Refund categorical	Marital Status continuous	Taxable Income class	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



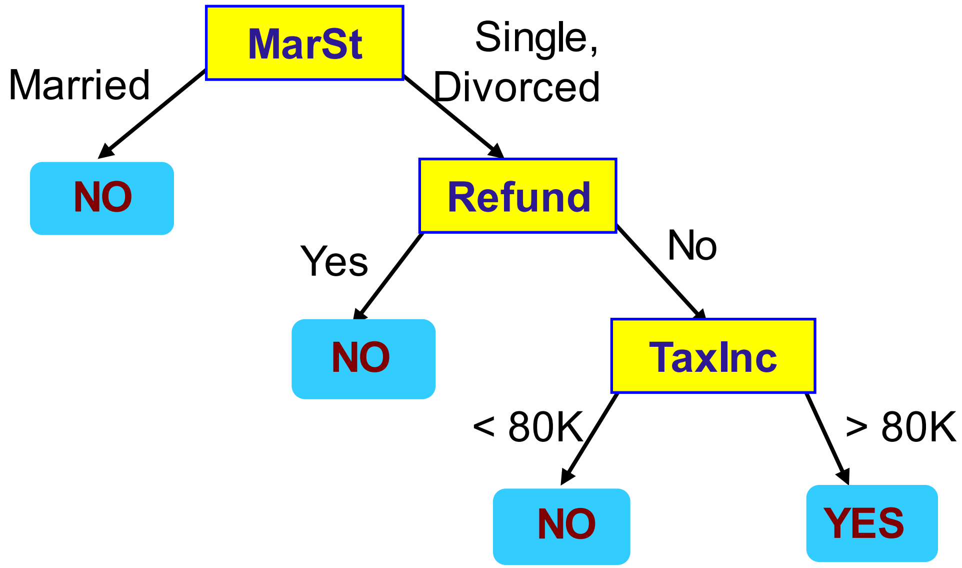
Training Data

Model: Decision Tree

Another Example of Decision Tree

categorical categorical continuous class

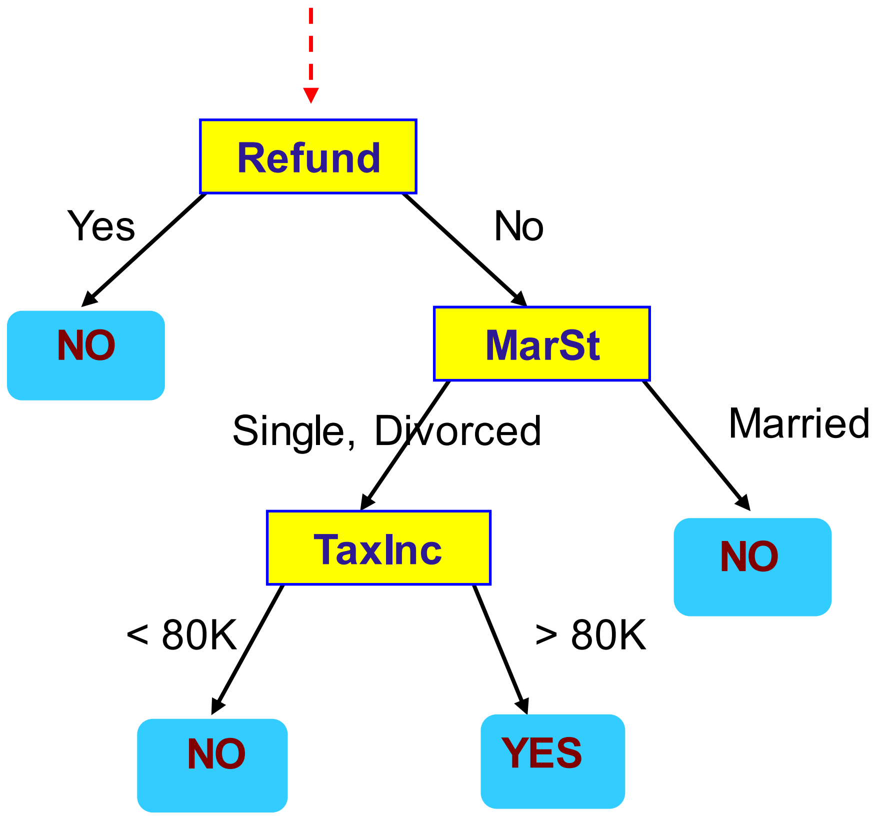
Tid	Refund	Marital Status	Taxable Income	Cheat
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes



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

Apply Model to Test Data

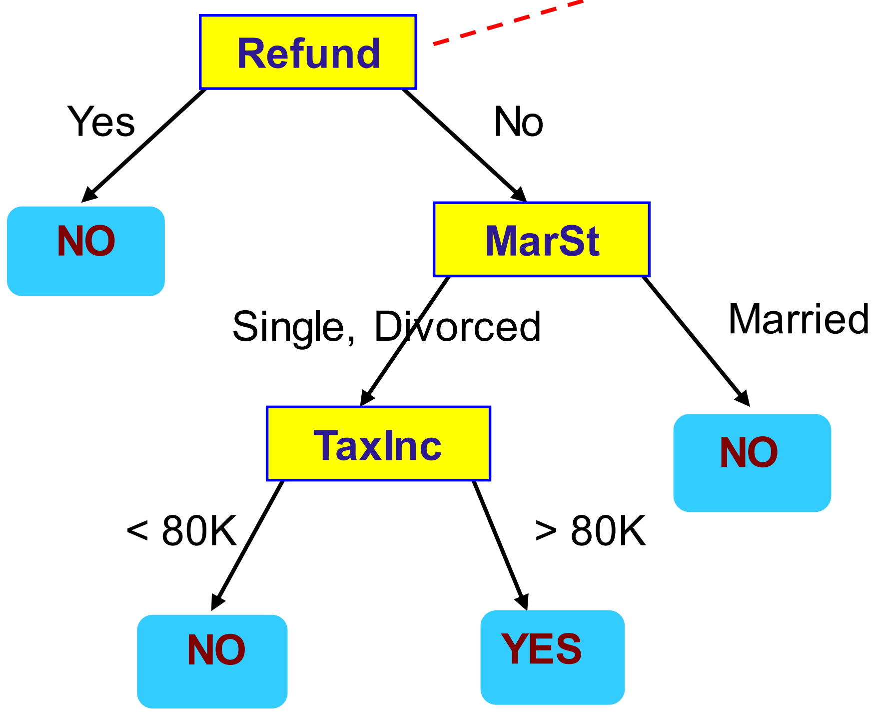
Start from the root of tree.



Test Data

Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?

Apply Model to Test Data



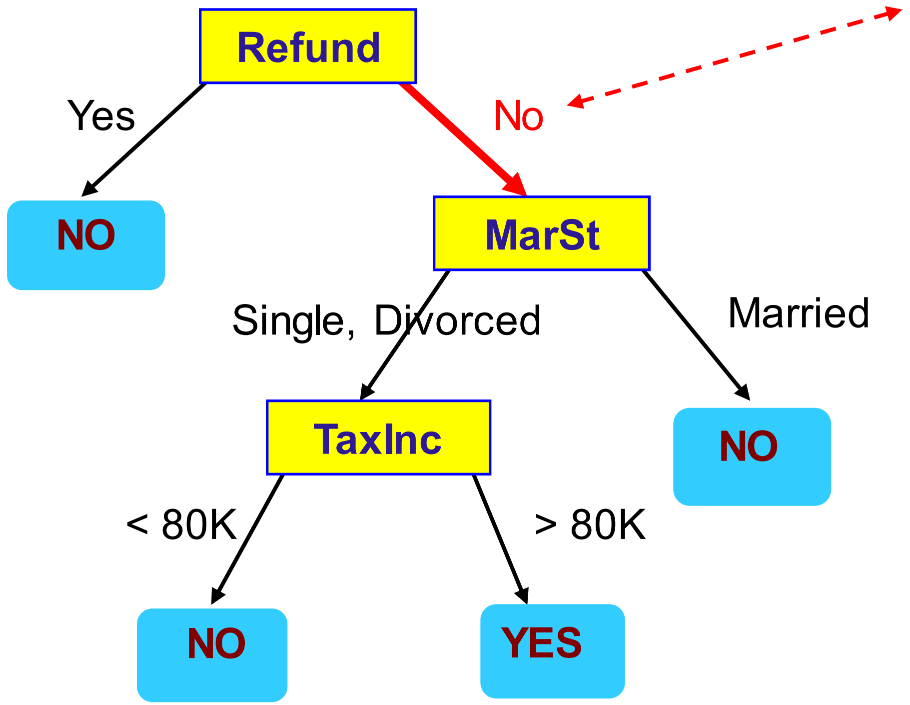
Test Data

Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?

Apply Model to Test Data

Test Data

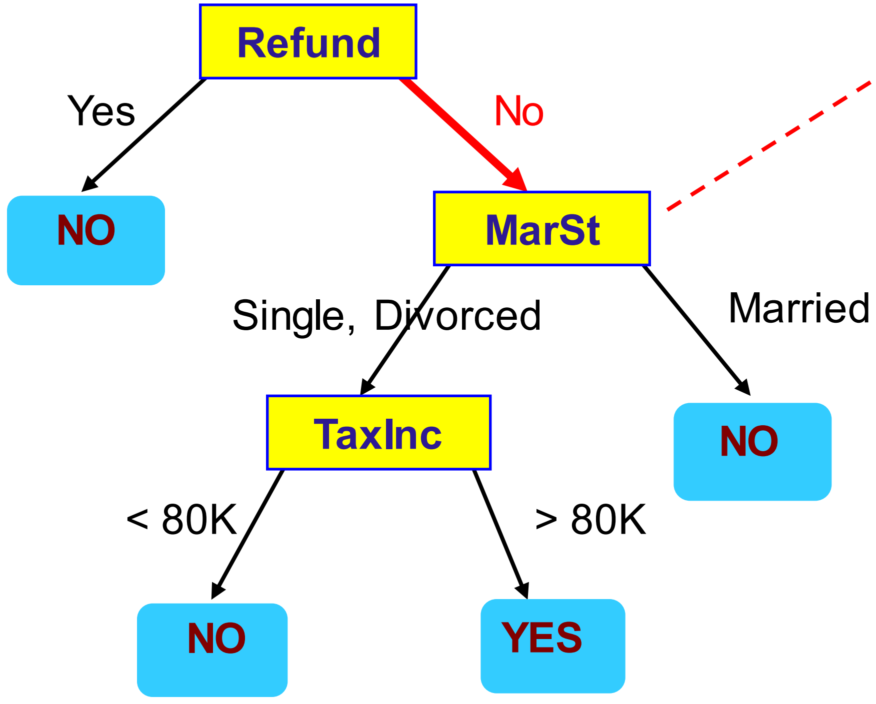
Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?



Apply Model to Test Data

Test Data

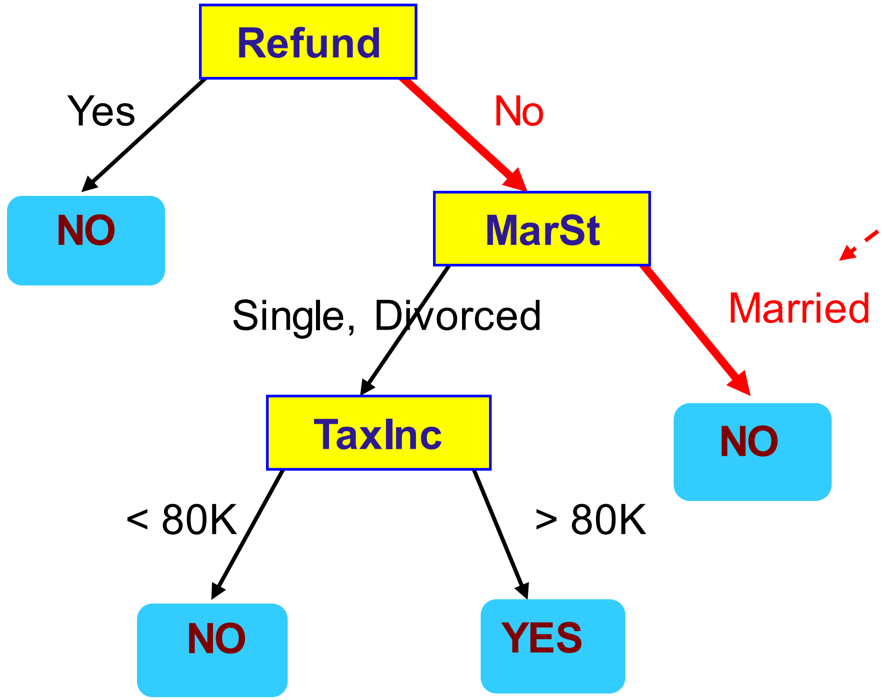
Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?



Apply Model to Test Data

Test Data

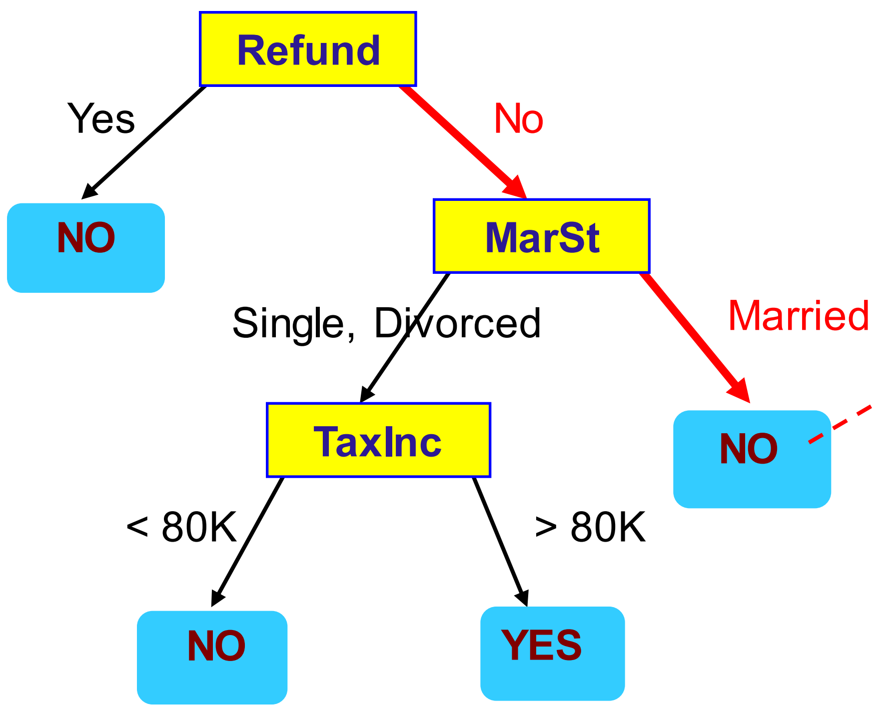
Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?



Apply Model to Test Data

Test Data

Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?



Assign Cheat to “No”

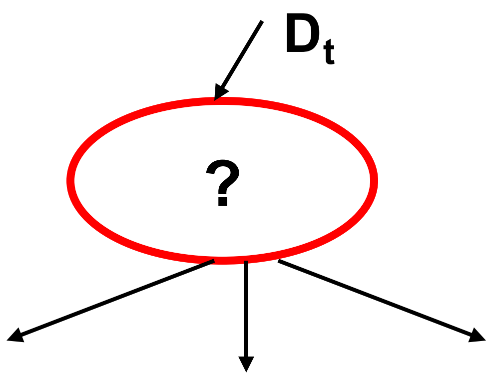
Decision Tree Induction

- Many Algorithms:
  - Hunt’s Algorithm (one of the earliest)
  - CART(Classification and Regression tree)
  - ID3(Iterative Dichotomiser 3)

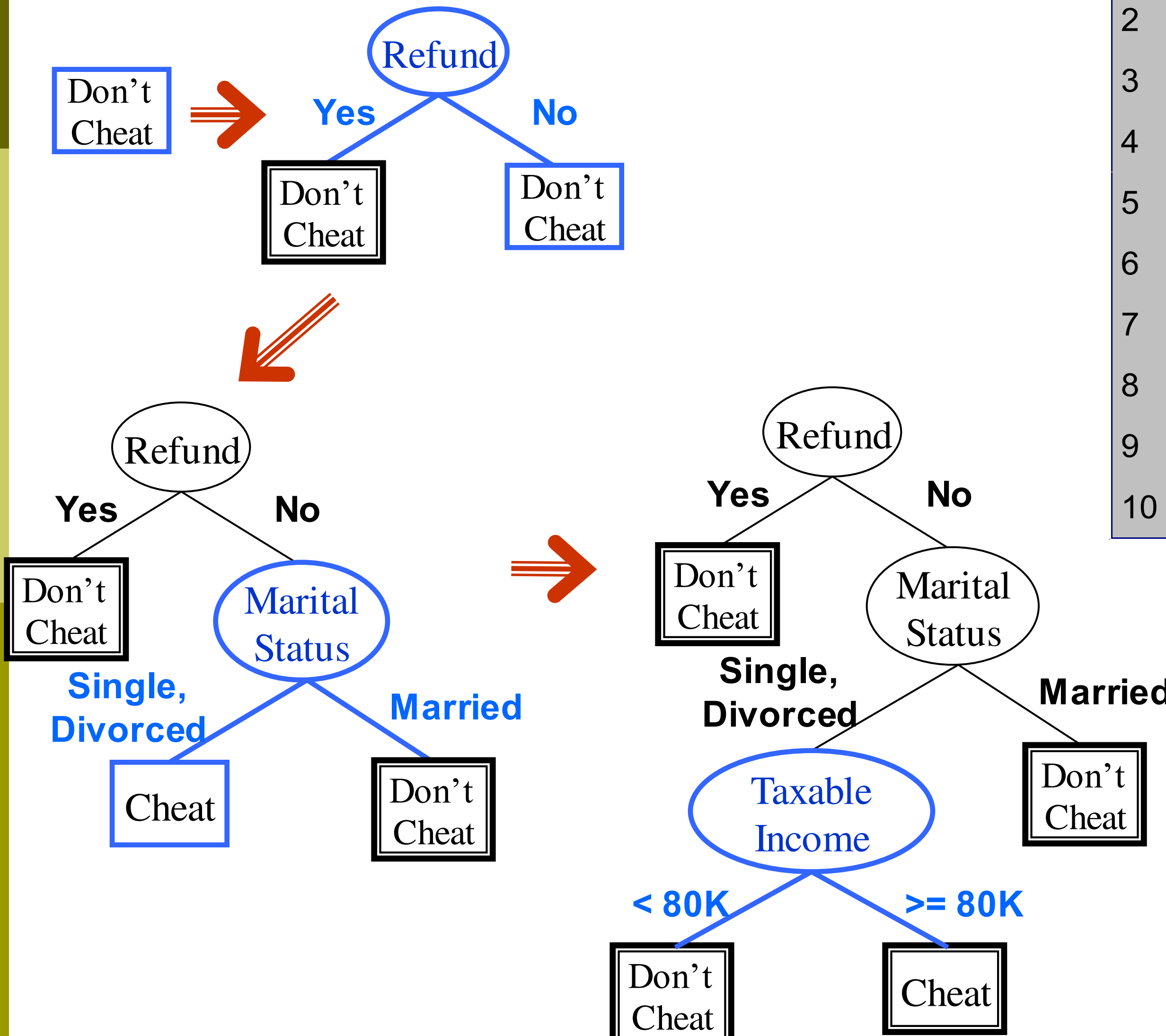
General Structure of Hunt's Algorithm

- Let D_t be the set of training records that reach a node t
- General Procedure:
  - If D_t contains records that 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

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

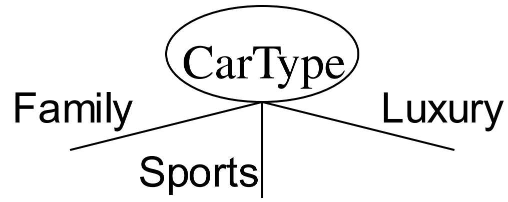
How to Specify Test Condition?

- Depends on attribute types
  - Nominal
  - Ordinal
  - Continuous

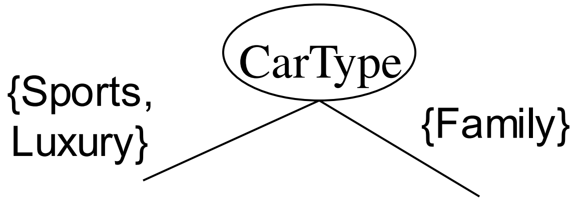
- Depends on number of ways to split
  - 2-way split
  - Multi-way split

Splitting Based on Nominal Attributes

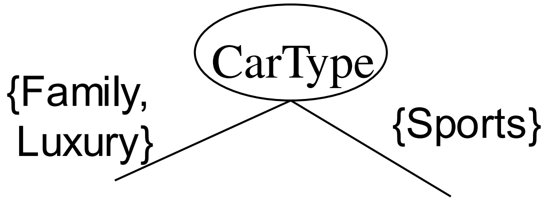
- Multi-way split: Use as many partitions as distinct values.



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

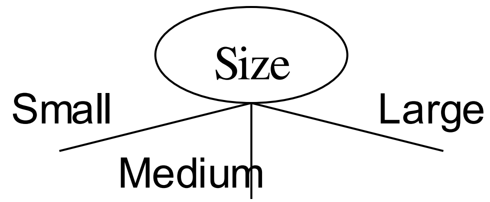


OR

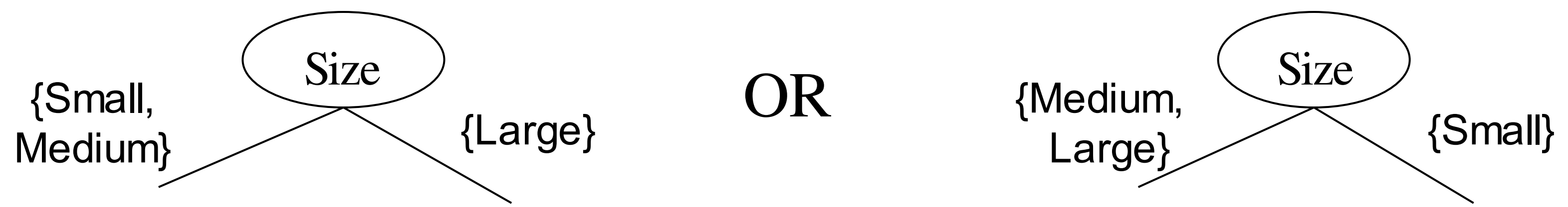


Splitting Based on Ordinal Attributes

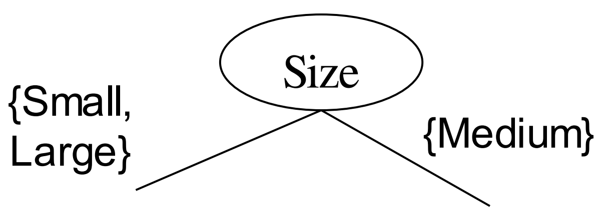
- Multi-way split: Use as many partitions as distinct values.



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



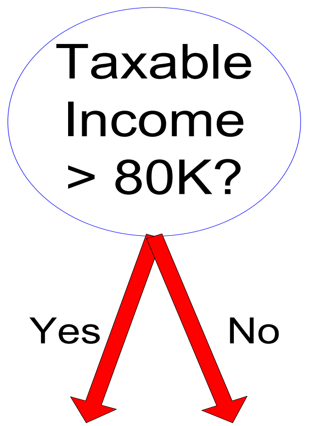
- What about this split?



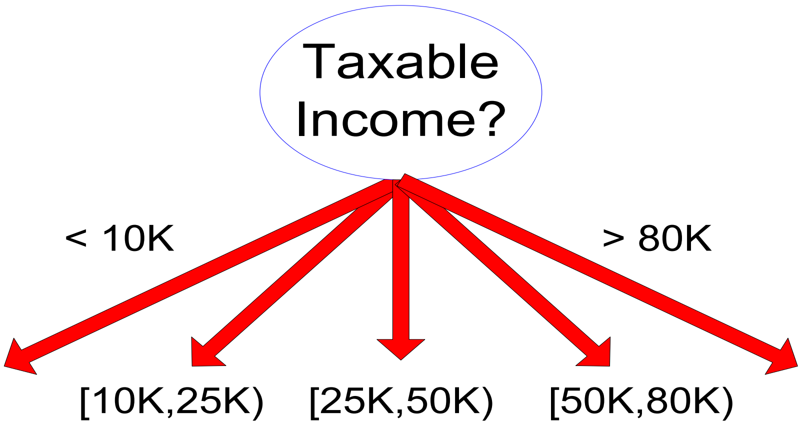
Splitting Based on Continuous Attributes

- 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 $(A \geq v)$
    - consider all possible splits and finds the best cut
    - can be more compute intensive

Splitting Based on Continuous Attributes



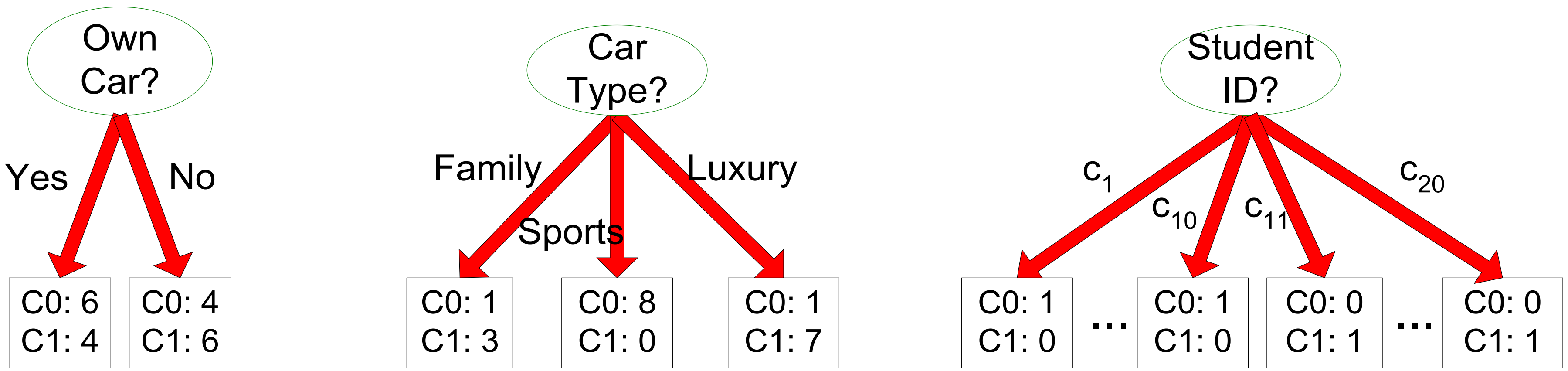
(i) Binary split



(ii) Multi-way split

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

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

Measures of Node Impurity

- Gini Index(used in CART algorithm)
- Entropy(used in ID3 algorithm)

Measure of Impurity: GINI

- Gini Index for a given node t :

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

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

  - 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

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

Splitting Based on GINI

- When a parent node p is split into k partitions (children), the quality of split is computed using information gain,

$$\boxed{Gain_{split} = GINI(p) - \sum_{i=1}^{k} \frac{n_i}{n} GINI(i)}$$

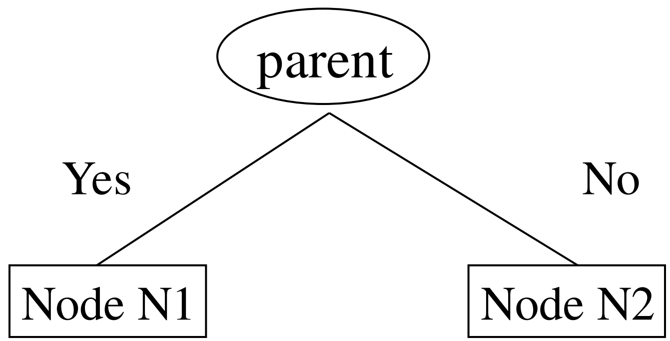
where, n_i = number of records at child node i,

n = number of records at parent node p.

Let Gini = $\boxed{\sum_{i=1}^{k} \frac{n_i}{n} GINI(i)}$ indicate GINI at children nodes smaller the value of Gini, larger is the gain.

Binary Attributes: Computing GINI Index

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



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

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

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

Gini(Children)
= 7/12 * 0.194 +
5/12 * 0.528
= 0.333

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)

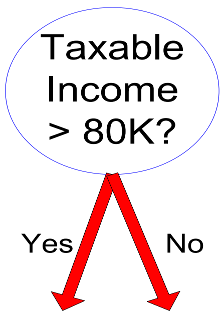
	CarType	
	{Sports, Luxury}	{Family}
C1	3	1
C2	2	4
Gini	0.400	

	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 ≥ 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



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

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	

Alternative Splitting Criteria based on Entropy

- Entropy at a given node t:

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

(NOTE: $p(j \mid t)$ 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
- Entropy based computations are similar to the GINI index computations

Examples for computing Entropy

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

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

Splitting Based on Entropy

- 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

Let, $$H(p) = \left(\sum_{i=1}^{k} \frac{n_i}{n} Entropy(i) \right)$$

Where H(p) indicates uncertainty at parent node p

So, smaller the uncertainty(H(p)) larger is the Gain.

Stopping Criteria for Tree Induction

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

Example

Day	Outlook	Temperature	Humidity	Wind	Play cricket
D1	Sunny	Hot	High	Weak	No
D2	Sunny	Hot	High	Strong	No
D3	Overcast	Hot	High	Weak	Yes
D4	Rain	Mild	High	Weak	Yes
D5	Rain	Cool	Normal	Weak	Yes
D6	Rain	Cool	Normal	Strong	No
D7	Overcast	Cool	Normal	Strong	Yes
D8	Sunny	Mild	High	Weak	No
D9	Sunny	Cool	Normal	Weak	Yes
D10	Rain	Mild	Normal	Weak	Yes
D11	Sunny	Mild	Normal	Strong	Yes
D12	Overcast	Mild	High	Strong	Yes
D13	Overcast	Hot	Normal	Weak	Yes
D14	Rain	Mild	High	Strong	No

Example

- How to choose best attribute?
  - Choose the attribute with highest information gain which is equivalent to finding minimum uncertainty(in ID3 algorithm)

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

$$\boxed{H(p) = \left(\sum_{i=1}^{k} \frac{n_i}{n} Entropy(i)\right)}$$

Compute H(p) for each attribute and consider the one with Minimum value as the root.

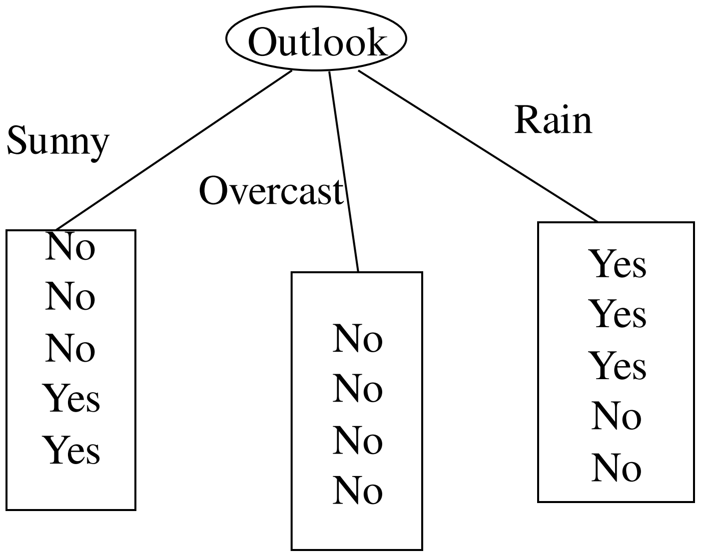
Example

Attribute Outlook

Entropy(outlook = sunny)
= -2/5xlog(2/5) - 3/5xlog(3/5) =0.971

Entropy(outlook = Overcast)
= -1xlog(1) – 0xlog(0) = 0

Entropy(outlook = Rain)
= -3/5xlog(3/5) – 2/5xlog(2/5)
= 0.971



Uncertainty of outlook is
H(Outlook) = 5/14xEntropy(outlook = sunny) + 4/14xEntropy(outlook = Overcast)
+ 5/14xEntropy(outlook = Rain)

= 5/14x0.971 + 4/14x0 + 5/14x0.971

= 0.693

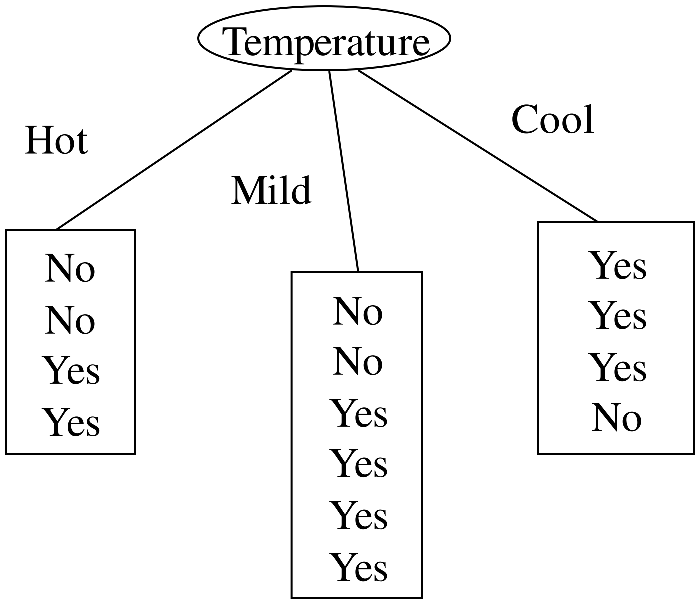
Example

Attribute Temperature

Entropy(Temperature = Hot)
= -2/4xlog(2/4) - 2/4xlog(2/4) = 1

Entropy(Temperature = Mild)
= -2/6xlog(2/6) – 4/6xlog(4/6) = 0.918

Entropy(Temperature = Cool)
= -1/4xlog(1/4) – 3/4xlog(3/4) = 0.811



Uncertainty of Temperature is
H(Temperature) = 4/14xEntropy(Temperature = Hot) + 6/14xEntropy(Temperature = Mild)
+ 4/14xEntropy(Temperature = Cool)

= 4/14x1 + 6/14x0.918 + 4/14x0.811

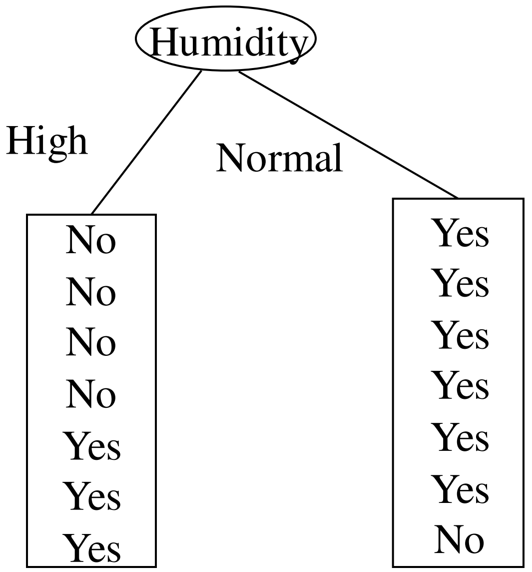
= 0.911

Example

Attribute Humidity

Entropy(Humidity = High)
= -3/7xlog(3/7) - 4/7xlog(4/7) =0.985

Entropy(Humidity = Normal)
= -1/7xlog(1/7) – 6/7xlog(6/7) = 0.591



Uncertainty of Humidity is
H(Humidity) = 7/14xEntropy(Humidity = High) + 7/14xEntropy(Humidity = Normal)

= 7/14x0.985 + 7/14x0.591

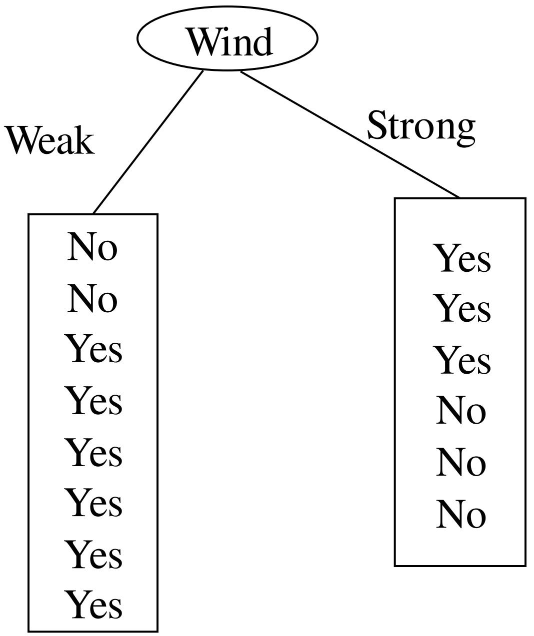
= 0.788

Example

Attribute Wind

Entropy(Wind = Weak)
= -2/8xlog(2/8) - 6/8xlog(6/8) =0.811

Entropy(Wind = Strong)
= -3/6xlog(3/6) – 3/6xlog(3/6) = 1



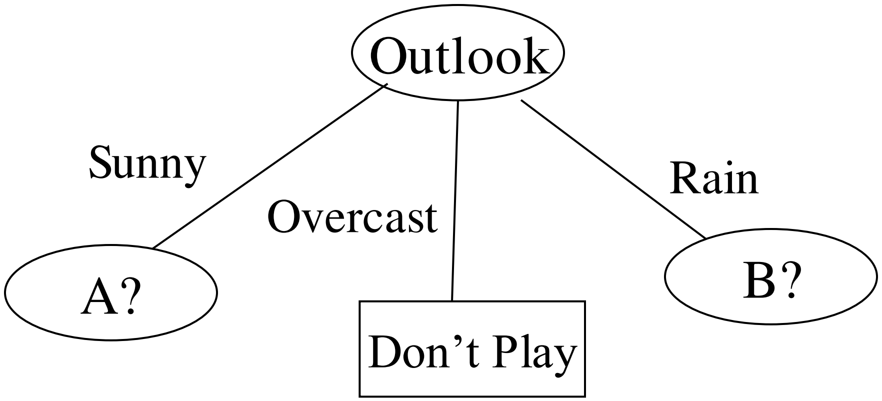
Uncertainty of Wind is
H(Wind) = 8/14xEntropy(Wind = Weak) + 6/14xEntropy(Wind = Strong)

= 8/14x0.811 + 4/14x0 + 6/14x1

= 0.892

Example

- Since Outlook has less uncertainty, choose it as root



- Since outlook = overcast has all class labels ‘No’, it is classified as Don’t play.
- To find attributes A and B , repeat same procedure

Example

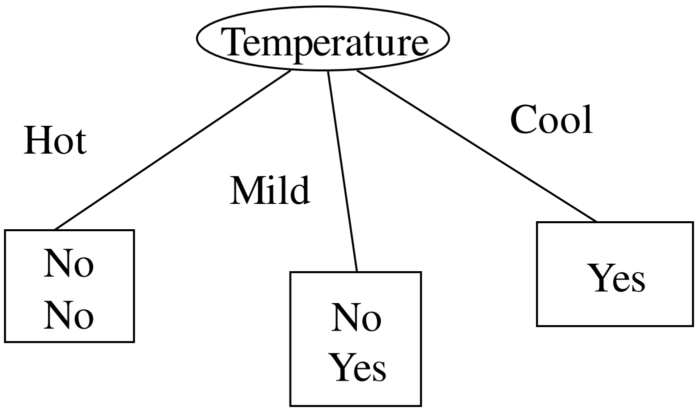
Attribute Temperature

Given Outlook = Sunny

Entropy(Temperature = Hot)
= -1xlog(1) - 0xlog(0) = 0

Entropy(Temperature = Mild)
= -1/2xlog(1/2) – 1/2xlog(1/2) = 1

Entropy(Temperature = Cool)
= -1xlog(1) – 0xlog(0) = 0



Uncertainty of Temperature is
H(Temperature) = 2/5xEntropy(Temperature = Hot) + 2/5xEntropy(Temperature = Mild)
+ 1/5xEntropy(Temperature = Cool)

=2/5x0 + 2/5x1 + 1/5x0

= 0.4

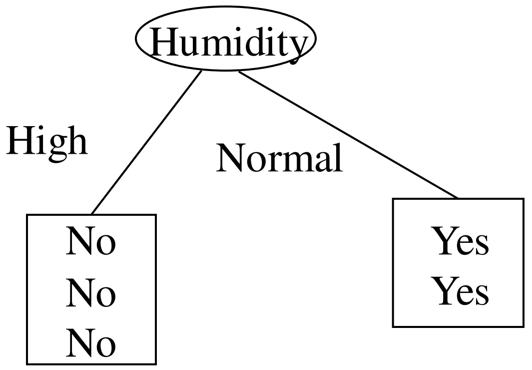
Example

Attribute Humidity

Given Outlook = Sunny

Entropy(Humidity = High)
= -1xlog(1) - 0xlog(0) =0

Entropy(Humidity = Normal)
= -0xlog(0) – 1xlog(1) = 0



Uncertainty of Humidity is
H(Humidity) = 3/5xEntropy(Humidity = High) + 2/5xEntropy(Humidity = Normal)

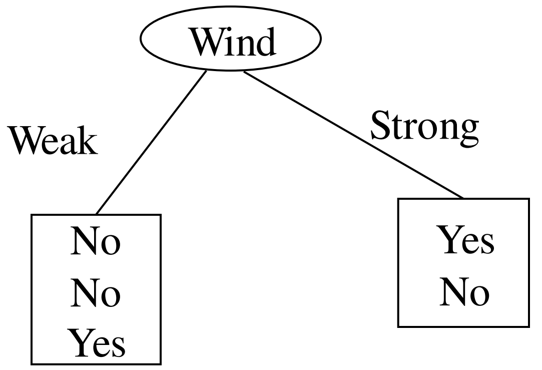
= 3/5x0 + 2/5x0

= 0

Example

Attribute Wind

Given Outlook = Sunny



Entropy(Wind = Weak)
= -2/3xlog(2/3) - 1/3xlog(1/3) =0.918

Entropy(Wind = Strong)
= -1/2xlog(1/2) – 1/2xlog(1/2) = 1

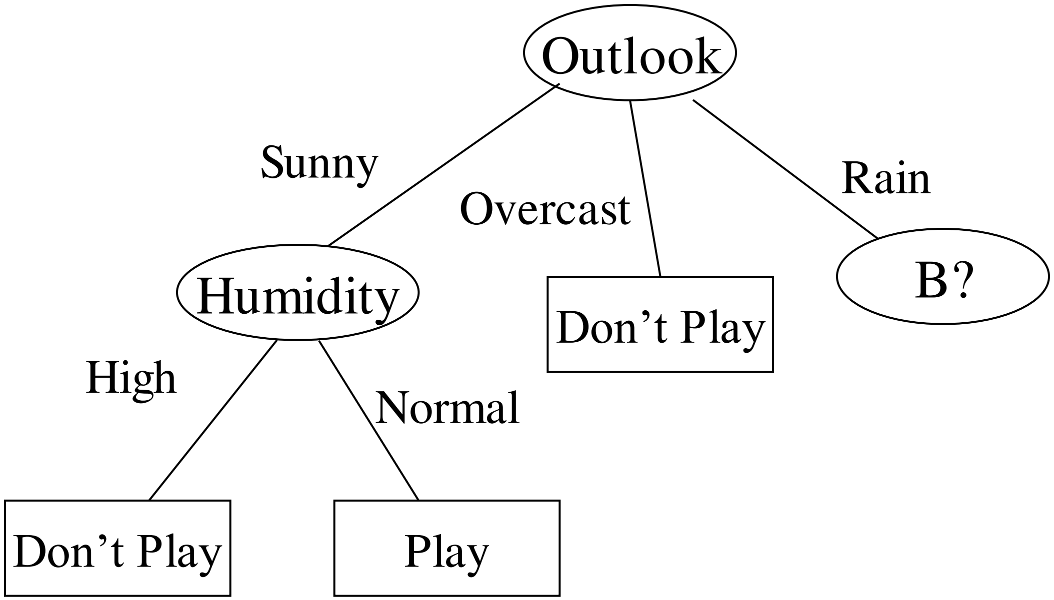
Uncertainty of outlook is
H(Outlook) = 3/5xEntropy(Wind = Weak) + 2/5xEntropy(Wind = Strong)

=3/5x0.918 + 2/5x1

= 0.951

Example

- Since for Outlook = Sunny, Humidity has less uncertainty we can choose it as child



Example

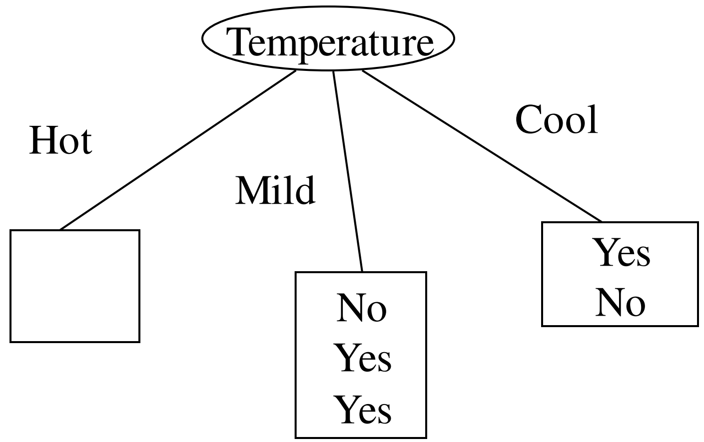
Attribute Temperature

Given Outlook = Rain

Entropy(Temperature = Hot)
= -0xlog(0) - 0xlog(0) = 0

Entropy(Temperature = Mild)
= -1/3xlog(1/3) – 2/3xlog(2/3) = 0.918

Entropy(Temperature = Cool)
= -1/2xlog(1/2) – 1/2xlog(1/2) = 1



Uncertainty of Temperature is
H(Temperature) = 0/5xEntropy(Temperature = Hot) + 3/5xEntropy(Temperature = Mild)
+ 2/5xEntropy(Temperature = Cool)

= 0/5x0 + 3/5x0.918. + 2/5x1

= 0.951

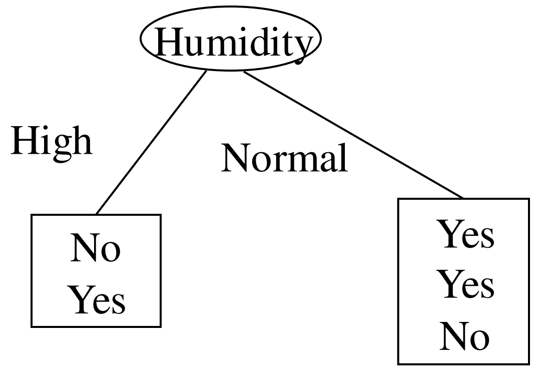
Example

Attribute Humidity

Given Outlook = Rain

Entropy(Humidity = High)
= -1/2xlog(1/2) – 1/2xlog(1/2) =1

Entropy(Humidity = Normal)
= -1/3xlog(1/3) – 2/3xlog(2/3) = 0.918



Uncertainty of Humidity is
H(Humidity) = 2/5xEntropy(Humidity = High) + 3/5xEntropy(Humidity = Normal)

= 2/5x1 + 3/5x0.918

= 0.951

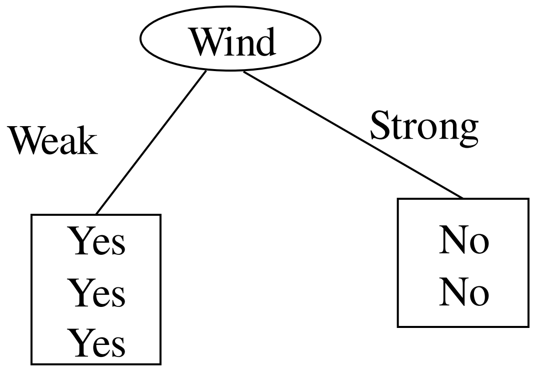
Example

Attribute Wind

Given Outlook = Rain

Entropy(Wind = Weak)
= -1xlog(1) - 0xlog(0) =0

Entropy(Wind = Strong)
= -0xlog(0) – 1xlog(1) = 0



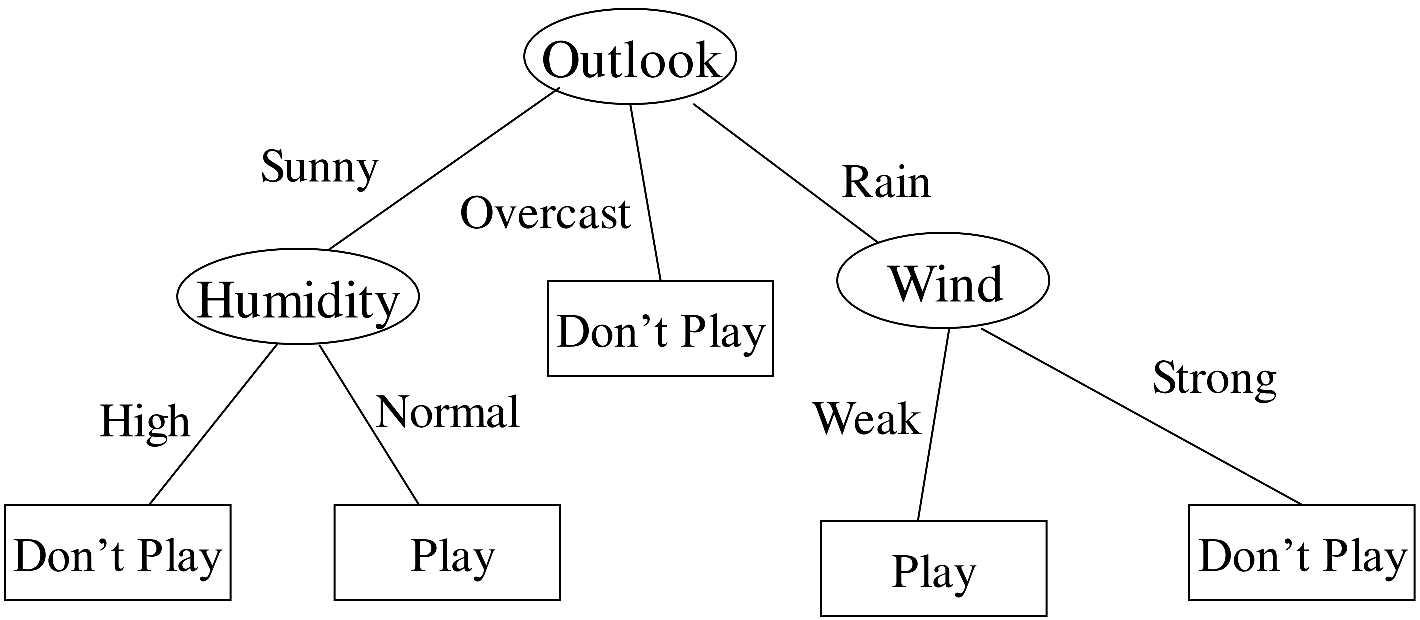
Uncertainty of Wind is
H(Wind) = 3/5xEntropy(Wind = Weak) + 2/5xEntropy(Wind = Strong)

= 3/5x0 + 2/5x0

= 0

Example

- Since for Outlook = Rain, Wind has less uncertainty we can choose it as child



This is the final Decision tree, since all leaf nodes indicate the class label.

Example

- Testing :
- Given outlook = Sunny, Temperature = cool, Humidity = Normal and wind = weak whether to play cricket or not.

- From the Decision tree, it is found that if outlook = Sunny and Humidity = Normal then **play Cricket**.

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

References

- Data Mining with Decision Trees: Theory and Applications
  Book by Lior Rokach, 17 December 2007

- Decision Tree and Ensemble Learning Based on Ant Colony Optimization Book by Jan Kozak, 20 June 2018

- Decision Trees and Random Forests: A Visual Introduction for Beginners Book by Chris Smith and Mark Koning,
- 4 October 2017

Thank You