University of Camerino

Knowledge Management and Business Intelligence

Learning From Observations

Partly based on Tan's and Steinbach's "Introduction to Data Mining"

Nadeem Qaisar Mehmood
Department of Computer Science
University of Camerino
nadeemqaisar.mehmood@unicam.it

Outline

- Learning
- Inductive Learning
- Machine Learning: Data Mining
- Predictive Learning
  - Classification
  - Different Classification Methods
    - Decision Tree Learning
      - Tree Representation
      - ID3 Algorithm
      - Tree Induction
        - Types of Attributes, their selection and splitting
        - Information gain and Impurity Measurement
        - When to stop
      - Rules Generation

Learning

- Improving behavior through diligent study of experience
- Learning modifies a software agent's decision mechanisms to improve performance

Definition:
A computer program is said to learn from experience E with respect to some class of tasks T and performance measure P, if its performance at tasks in T, as measured by P, improves with experience E (Tom Mitchel 1997)

For example, a computer program that learns to play checkers might improve its performance as ***measured by its ability to win*** at the class of tasks involving ***playing checkers games,*** through experience ***obtained*** by ***playing games against itself.***

Learning Element

Design of learning element is dictated by

- what type of **performance** element is used
- which functional component is to be learned
- how that functional component is represented
- what kind of **feedback** is available

Learning Types

- **Supervised Learning**
  - **Correct answers for each instance**
  - **Known Classes in advance**
  - **Prior knowledge**
- Unsupervised Learning: No training, unknown classes, no prior knowledge
- Reinforcement Learning: Occasional Rewards

Inductive Learning

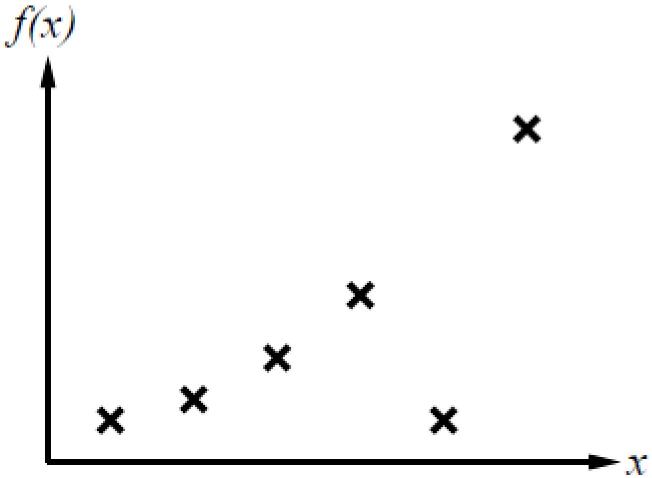
Simplest form: learn a function *f* from examples

f is the target function

Problem: Find a(n) hypothesis h
  such that h ≈ f
  given a training set of examples

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

For Example: Curve Fitting



[Russell & Norvig's Artificial Intelligence CH18]

Inductive Learning

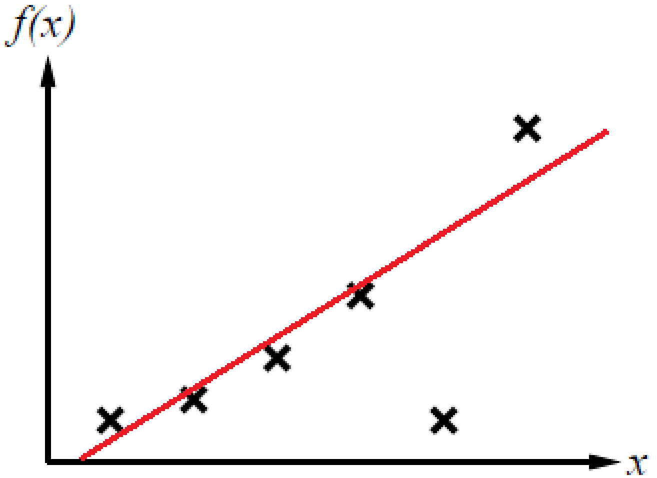
Simplest form: learn a function *f* from examples

f is the target function

Problem: Find a(n) hypothesis h
 such that h ≈ f
 given a training set of examples

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

For Example: Curve Fitting



[Russell & Norvig's Artificial Intelligence CH18]

Inductive Learning

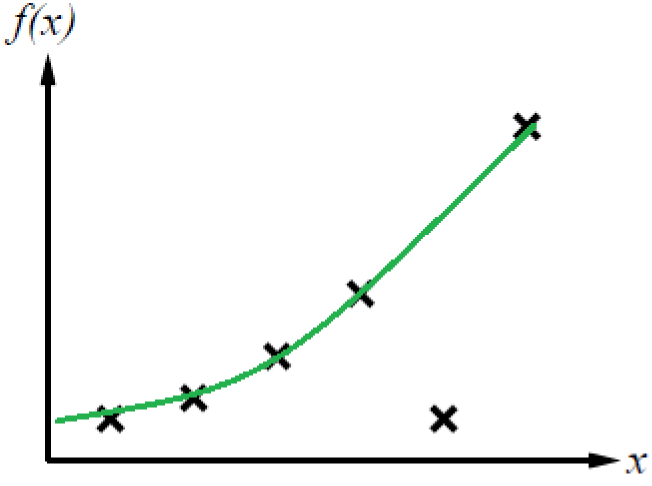
Simplest form: learn a function *f* from examples

f is the target function

Problem: Find a(n) hypothesis h
 such that h ≈ f
 given a training set of examples

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

For Example: Curve Fitting



[Russell & Norvig's Artificial Intelligence CH18]

Inductive Learning

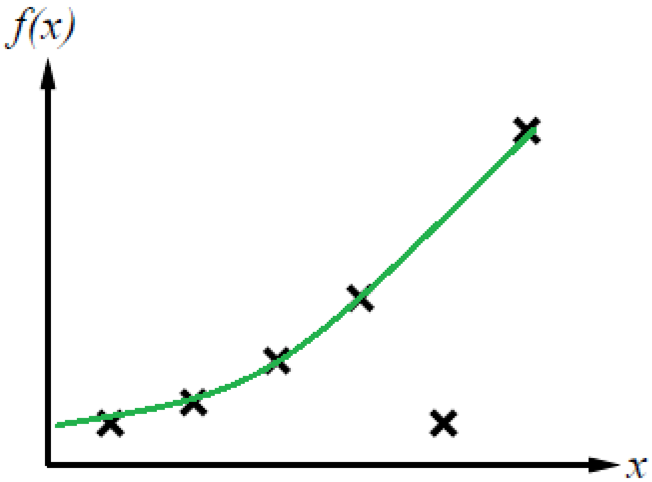
Simplest form: learn a function *f* from examples

f is the target function

Problem: Find a(n) hypothesis h
such that h ≈ f
given a training set of examples

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

For Example: Curve Fitting



[Russell & Norvig's Artificial Intelligence CH18]

Inductive Learning

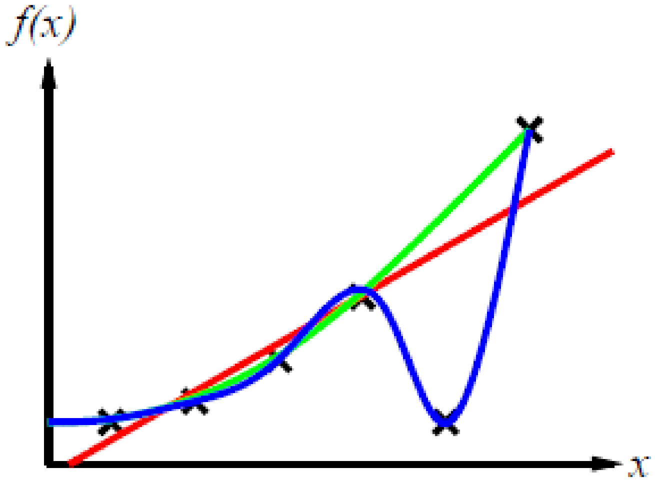
Simplest form: learn a function *f* from examples

f is the target function

Problem: Find a(n) hypothesis h
  such that h ≈ f
  given a training set of examples

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

For Example: Curve Fitting



[Russell & Norvig's Artificial Intelligence CH18]

Inductive Learning

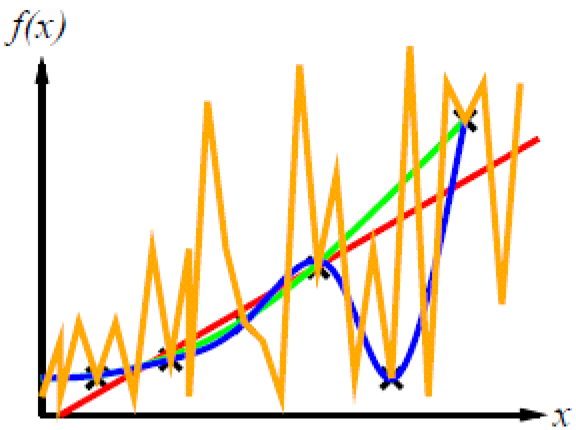
Simplest form: learn a function *f* from examples

f is the target function

Problem: Find a(n) hypothesis h
such that h ≈ f
given a training set of examples

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

For Example: Curve Fitting



[Russell & Norvig's Artificial Intelligence CH18]

Ockham's razor: maximize a combination of consistency and simplicity

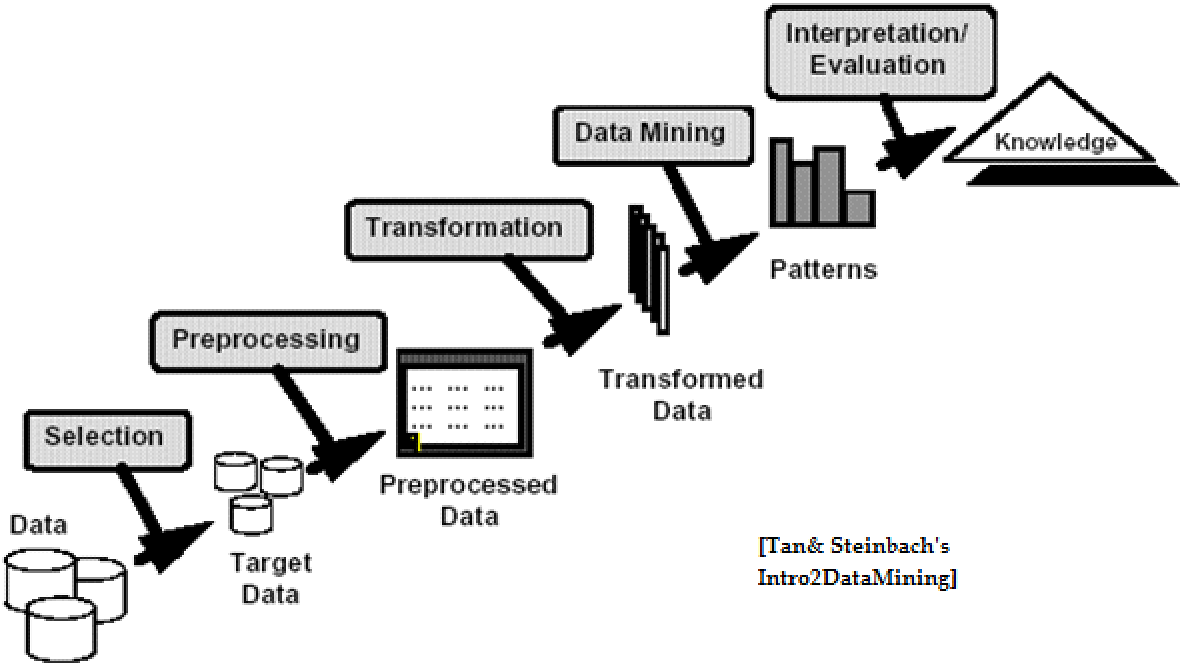
Machine Learning: Data Mining

Information is mostly hidden in data sets

Learn from data for prediction or description

A process how to extract or uncover hidden information to help in decisions or to identify patterns within the data.

Data Mining Process



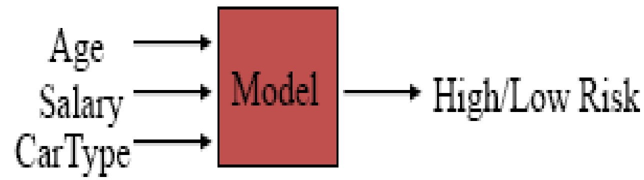
[Tan& Steinbach's Intro2DataMining]

Predictive Modeling: Classification

- Given a collection of training records (*training set*)
  - Each record consists 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: to assign a class to previously unseen records as accurately as possible.
  - A *test set* is used to determine the accuracy of the model.
  - Usually, the given data set is divided into
    1. training set (with training set used to build the model)
    2. test set (with test sets used to validate it)
- Supervised learning

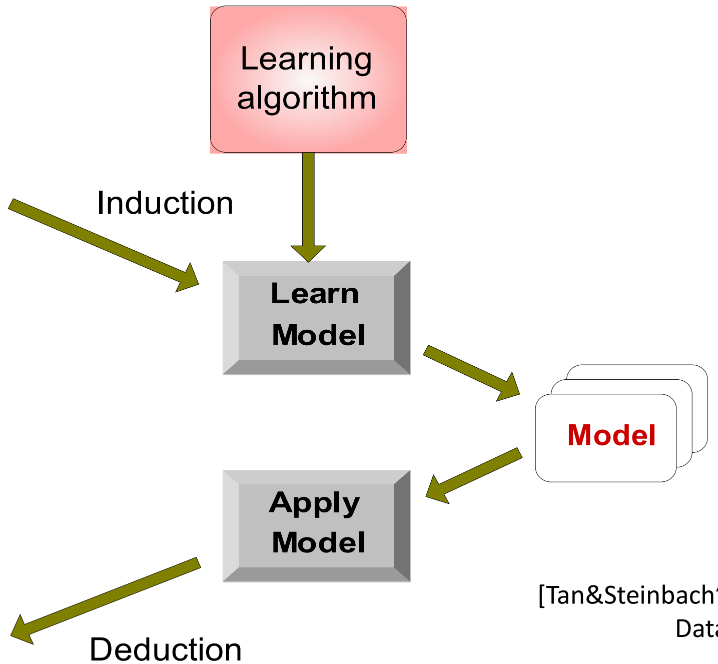
Predictive Learning Process

Learn to predict
Learn a model
Learn from instances/examples
Predict on un seen instances



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	?

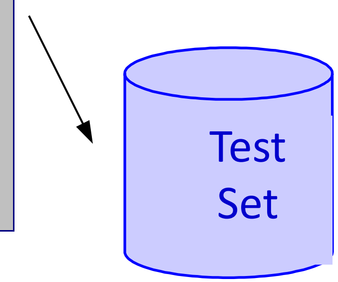
Test Set

[Tan&Steinbach's "Intro to Data Mining"]

Classification as Predictive Modeling

Tid	Refund (categorical)	Marital Status (categorical)	Taxable Income (continuous)	Cheat (class)
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

Refund	Marital Status	Taxable Income	Cheat
No	Single	75K	?
Yes	Married	50K	?
No	Married	150K	?
Yes	Divorced	90K	?
No	Single	40K	?
No	Married	80K	?



[Tan&Steinbach's "Intro to Data Mining"]

Different Classification Methods

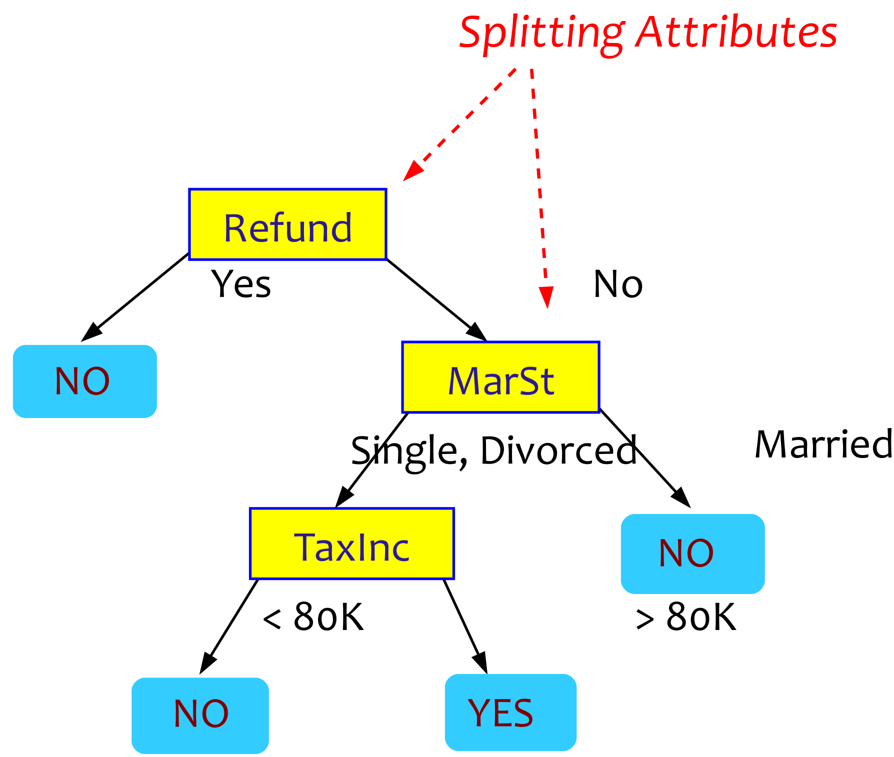
- Learn Decision Trees (our focus!)
- Instance based learning for rules
- Predict based on the nearest neighbors
- Predict based on probabilities
- Artificial Neural Networks

Decision Tree Representation (Example)

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

[Tan&Steinbach's "Intro to Data Mining"]

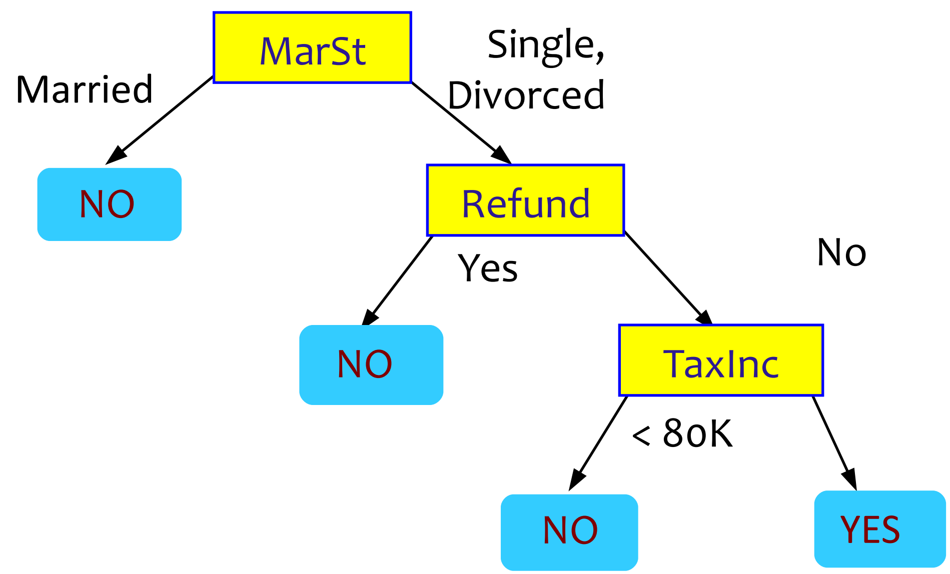
Decision Tree Representation (Another Example)

University of Camerino

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

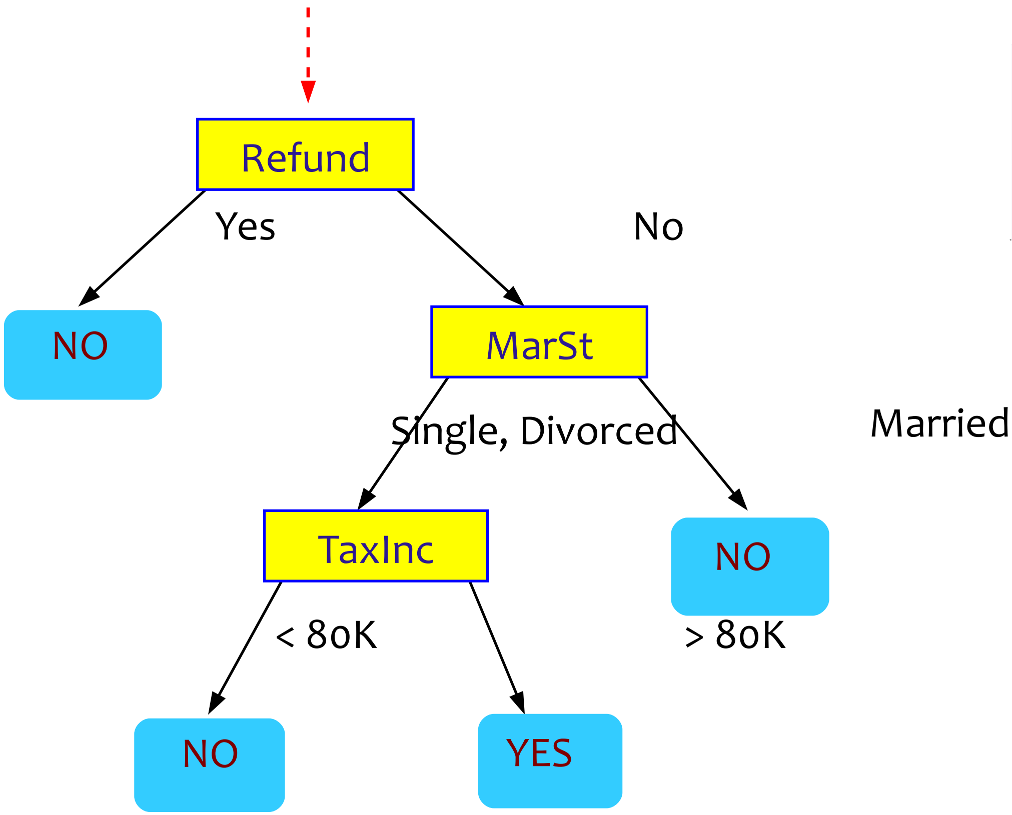


Model: Decision Tree

[Tan&Steinbach's "Intro to Data Mining"]

Applying Model to Test Data

Start from the root of the tree.



Test Data

Refund	Marital Status	Taxable Income	Cheat
No	Married	80K	?

Predict "Cheat" Attribute!

[Tan&Steinbach's "Intro to Data Mining"]

Basic Decision Tree Learning Algorithm:ID3

- Top down
- Many possible trees but only one is selected
- Greedy strategy
  - Which attribute should be tested at the root of the tree?
  - Split the records based on an attribute that split records and help classify the instances
  - We want to select the attribute that is most useful for classifying examples.

ID3(Examples, Target-attribute, Attributes)

/* Examples: The training examples; */
/* Target-attribute:The attribute whose value is to be predicted by the tree; */
/* Attributes: A list of other attributes that may be tested by the learned decision tree. */
/* Return a decision tree that correctly classifies the given Examples */

Step 1: Create a Root node for the tree
Step 2: If all *Examples* are positive, Return the single-node tree *Root*,with label = +
Step 3: If all E*Examples* are negative, Return the single-node tree *Root*,with label = -
Step 4: If *Attributes* is empty, Return the single-node tree *Root*, with label = most common value of *Target-attribute* in *Examples*
Step 5: Otherwise Begin

- A ← the attribute from *Attributes* that best (i.e., highest information gain) classifies *Examples*;
- The decision attribute for *Root* ← A;
- For each possible value, v_i , of A,
  - Add a new tree branch below *Root*, corresponding to the test A=v_i;
  - Let $Examples_{(v_i)}$ be the subset of *Examples* that have value v_i for A;
  - If $Examples_{(v_i)}$ is empty
    - Then below this new branch add a leaf node with label = most common value of *Target-attribute* in *Examples*
    - Else below this new branch add the subtree ID3($Examples_{(v_i)}$, *Target-attribute*, *Attributes*- A))

End

Return *Root*

- ❑ **Now how to measure worth of an Attribute?**
- ❑ **How to specify test conditions**
- ❑ **When to stop?**

[Mitchell, Tom M. Machine Learning. McGraw-Hill, 1997. pp. 55-58]

Tree Induction

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

How to measure attribute worth for best split

- Use a statistical property "Information Gain" to measure worth

$$GAIN(S,A) = Entropy(S) - \left(\sum_{v \in Values(A)} \frac{|S_v|}{|S|} Entropy(S_v) \right)$$

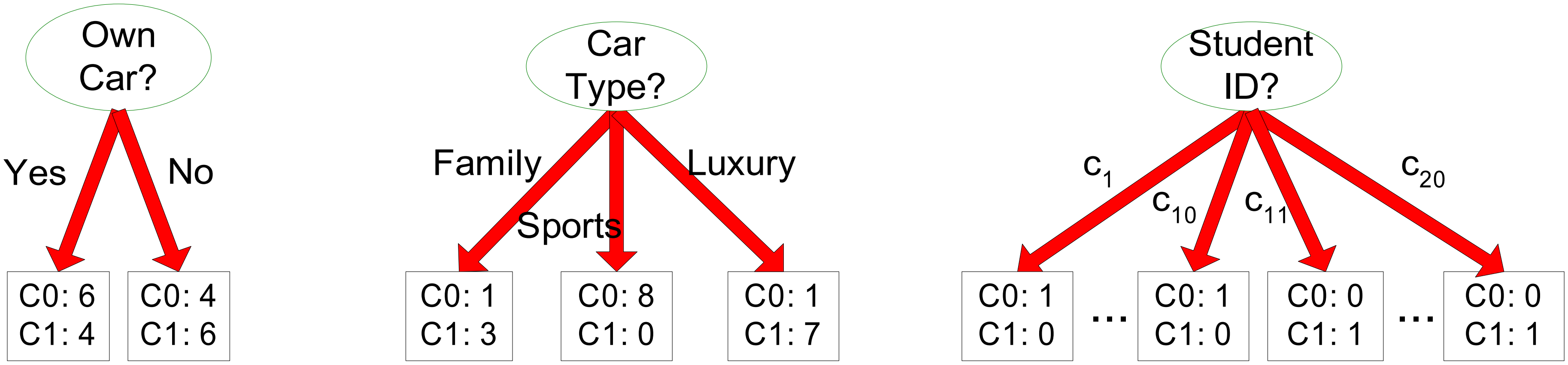
  - *Values(A)* is the set of all possible values for attribute A
  - S_v is the subset of S for which attribute A has value v
- Uses Entropy to generate the information gain

Entropy : $E(S) = -(p_+)*\log_2(p_+) - (p_-)*\log_2(p_-)$

  - ***E(S)*** *is the information entropy of the sample training examples* ***S***
  - *Where* ***p+*** *is the proportion of positive samples in S*
  - *Where* ***p-*** *is the proportion of negative samples in S*

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

Non-homogeneous,

High degree of impurity

C0: 9
C1: 1

Homogeneous,

Low degree of impurity

Computing Entropy (example)

University of Camerino

Entropy: $E(S) = -(p_+)*\log_2(p_+) - (p_-)*\log_2(p_-)$

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 = $-(1/6)\log_2(1/6) - (5/6)\log_2(5/6) = 0.65$

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

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

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

Entropy: $E(S) = -(p_+)*\log_2(p_+) - (p_-)*\log_2(p_-)$

Computing Entropy (example)

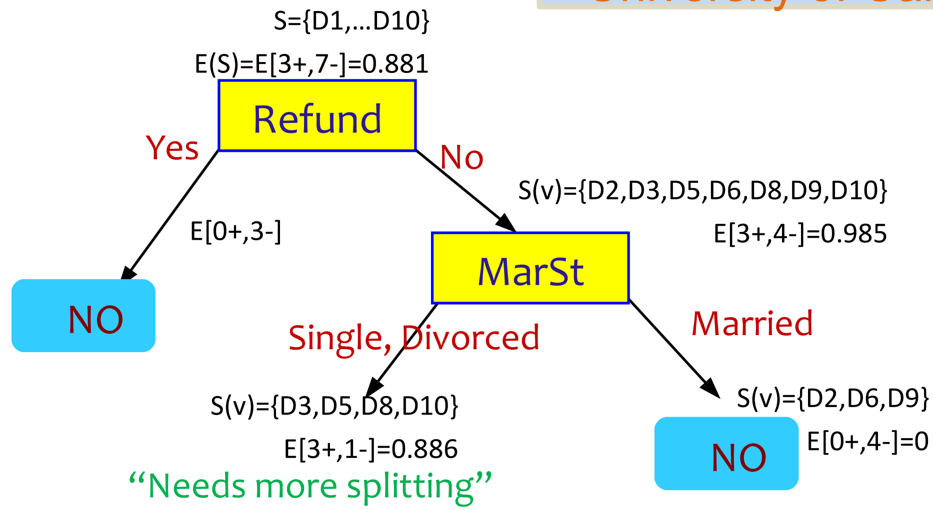
Tid	**Refund**	**Marital Status**	**Taxable Income**	**Cheat**
D1	Yes	Single	125K	**No**
D2	No	Married	100K	**No**
D3	No	Single	70K	**No**
D4	Yes	Married	120K	**No**
D5	No	Divorced	95K	**Yes**
D6	No	Married	60K	**No**
D7	Yes	Divorced	220K	**No**
D8	No	Single	85K	**Yes**
D9	No	Married	75K	**No**
D10	No	Single	90K	**Yes**

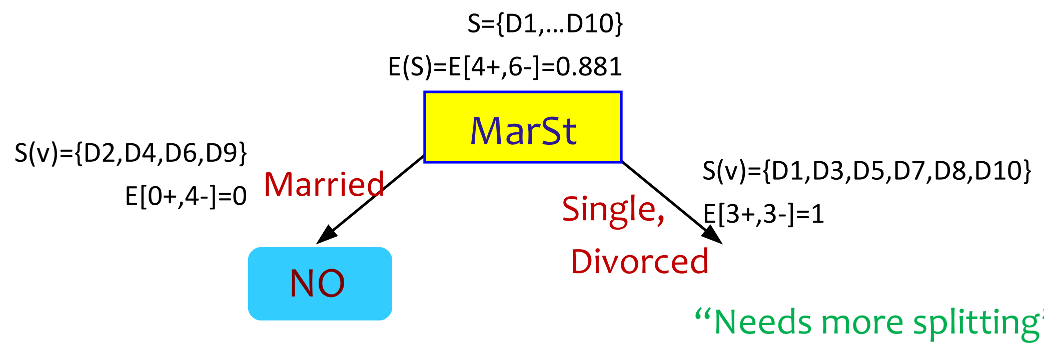
Training Data (S)

E(S)=E[3+,7-]= -(3/10)log(3/10)-(7/10)log(7/10)

=-(0.3)log(0.3)-(0.7)log(0.7)

=0.881



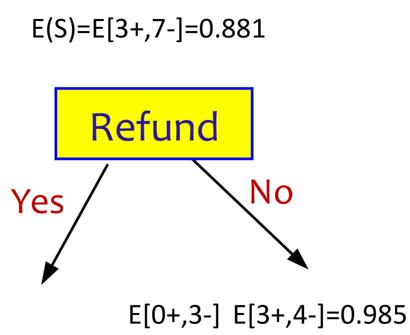


[Tan&Steinbach's "Intro to Data Mining"]

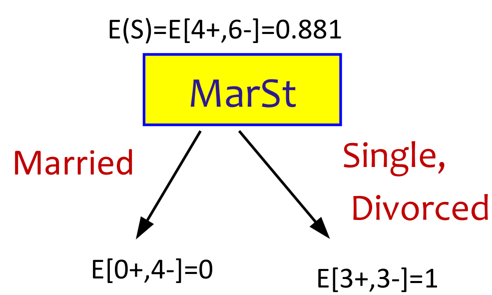
Get Information Gain using Entropy

- Measures Reduction in Entropy achieved because of the split.
- Choose the split that achieves most reduction in entropy

$$GAIN(S,A) = Entropy(S) - \left(\sum_{v \in Values(A)} \frac{|S_v|}{|S|} Entropy(S_v) \right)$$



- ❑ Gain (S,Refund)
  =0.881-{ (3/10)(0)+(7/10)(0.985) }
  =0.1915
- ❑ Gain(S, MarStatus)
  =0.881-{(4/10)(0)+(6/10)(1)}
  =0.281



❖ Since with marital status provides more gain, therefore in this case it will be the root node.

Tree Induction

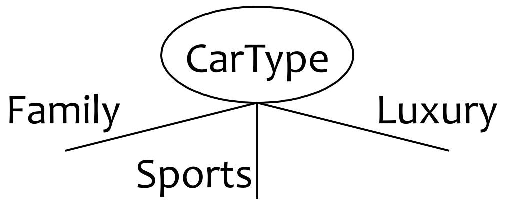
- Greedy strategy.
  - Split the records based on an attribute test that optimizes certain criterion.
- Issues
  - Determine how to split the records
    - How to determine the best split?
    - How to specify the attribute test condition?
  - Determine when to stop splitting

How to specify Attribute Test Conditions?

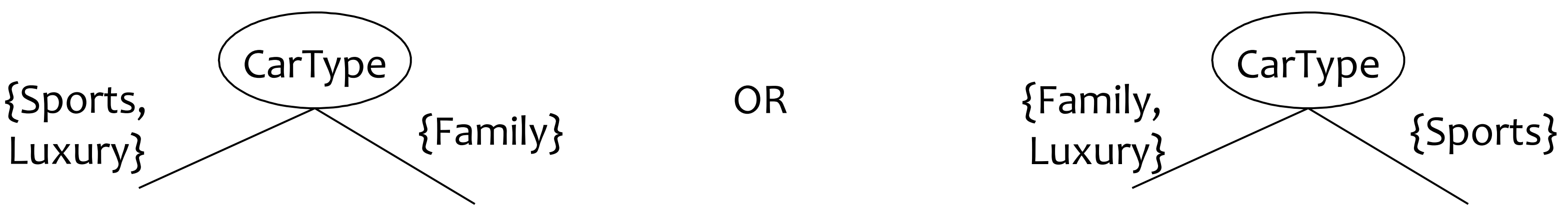
- Depends on attribute types
  - Nominal
  - Ordinal
  - Continuous
  -
- Depends on number of ways to split
  - 2-way split
  - Multi-way split

Splitting based on Nominal Attributes

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

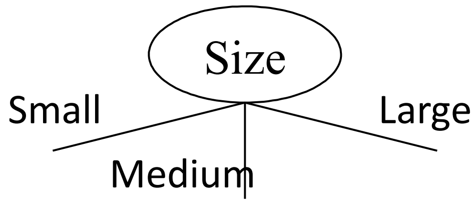


- Binary split: Divides values into two subsets. Need to find optimal partitioning.

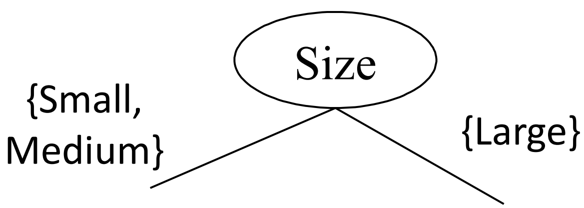


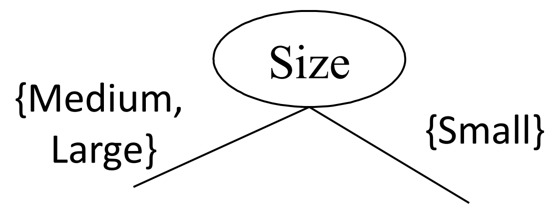
Splitting based on Ordinal Attributes

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

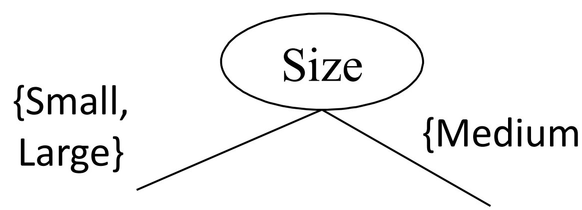


- Binary split: Divides values into two subsets. Need to find optimal partitioning.





OR

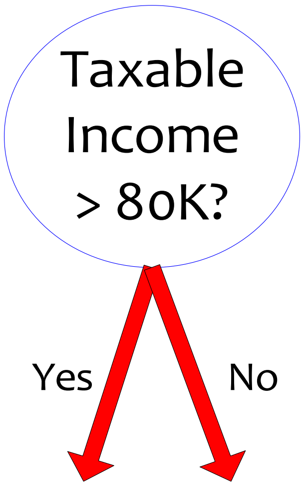


What about this split?

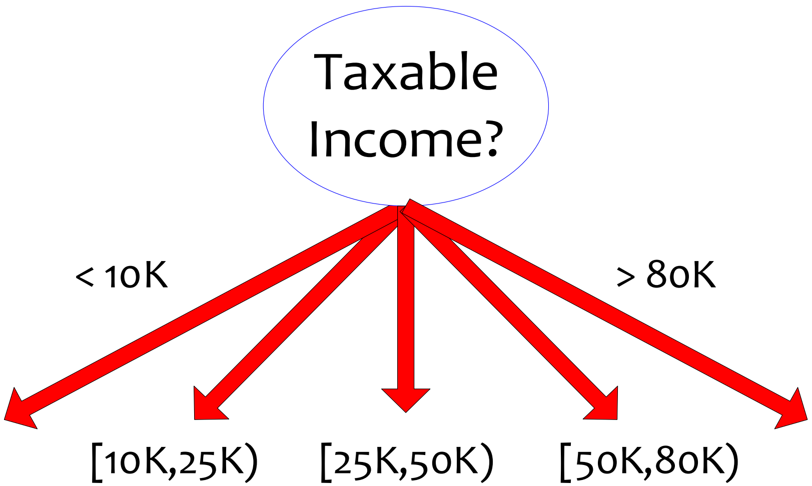
Splitting based on Continuous Attributes

- 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

Tree Induction

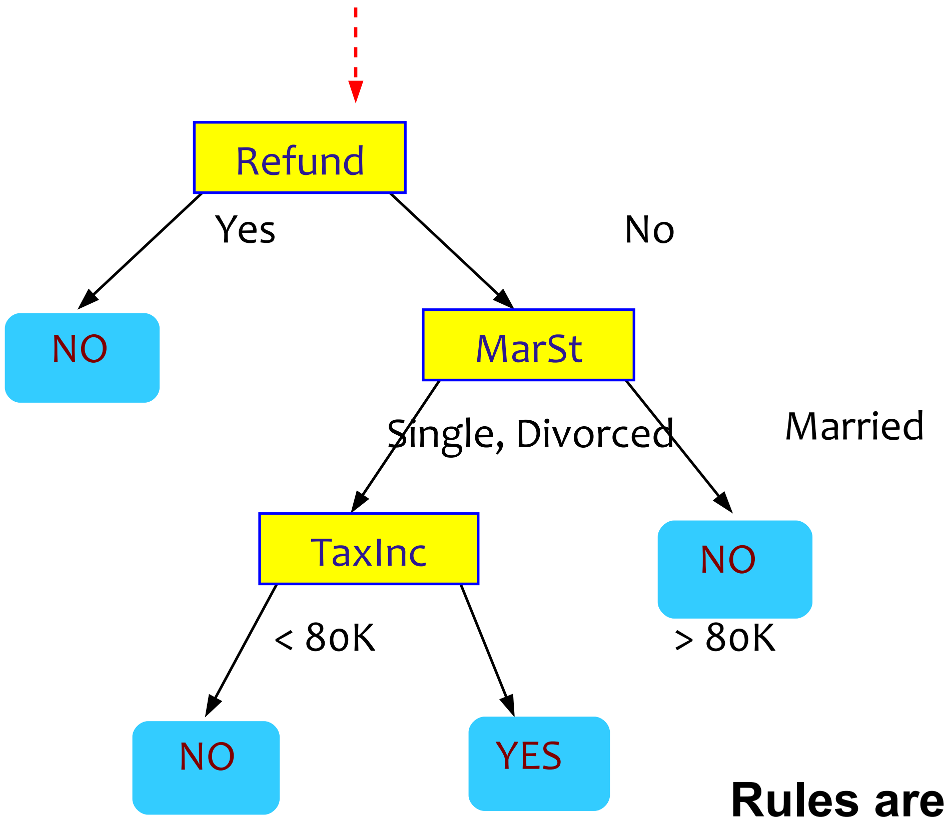
- Greedy strategy.
  - Split the records based on an attribute test that optimizes certain criterion.
- Issues
  - Determine how to split the records
    - How to specify the attribute test condition?
    - How to determine the best split?
  - Determine when to stop splitting

When to stop Splitting during Tree Induction

- Stop expanding a node when all the records belong to the same class
- Stop expanding a node when all the records have similar attribute values
- Early termination (to be discussed later)

Decision Tree representation in Rules form

Start from the root of the tree.



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

Rules are mutually exclusive and exhaustive

Rule set contains as much information as the tree

References

University of Camerino

- Tan and Steinbach's "Introduction to Data Mining"
- Peter Norvig's "Artificial Intelligence"
- Tom Mitchel's "Machine Learning"
- Quinlan, J. R. 1986. Induction of Decision Trees. Mach. Learn. 1, 1 (Mar. 1986), 81-106.