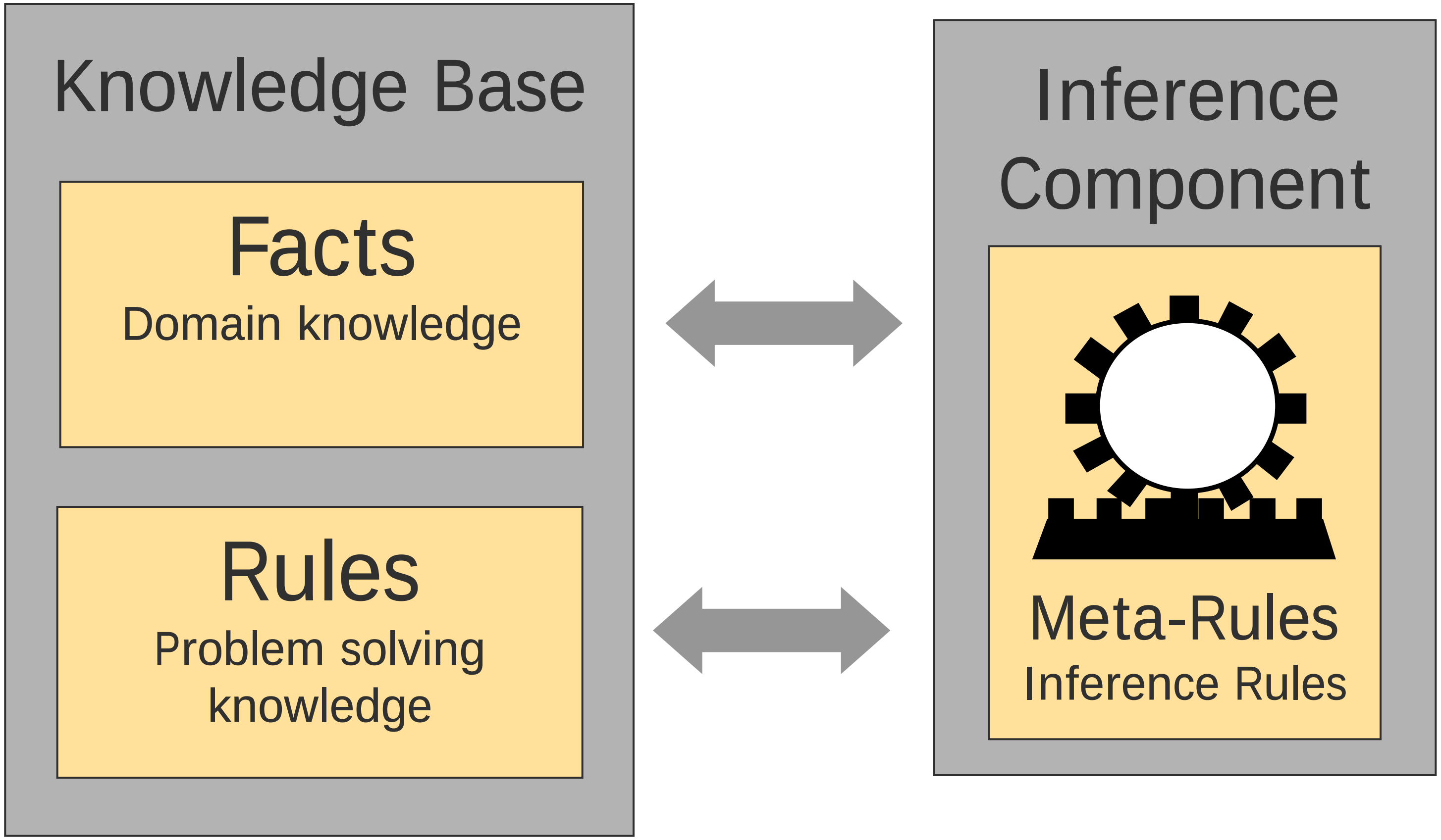
University of Applied Sciences and Arts Northwestern Switzerland
School of Business

Machine Learning:
Learning Decision Trees

Knut Hinkelmann

Knowledge-Based Systems (Rules & Facts)



Creating Knowledge Bases

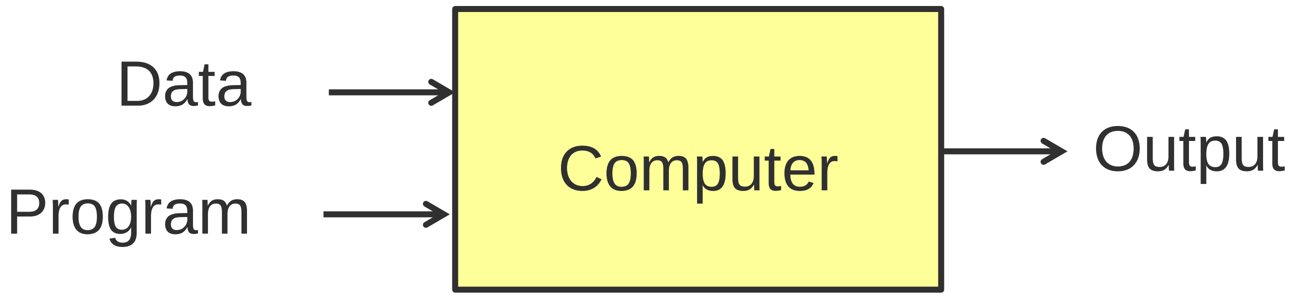
- Manual: Human expert builds rules
  - Rules are often redundant, incomplete, inconsistent or inefficient
  - This is also call Knowledge Engineering
- **Machine Learning**: automatic derivation of rules from example data
  - this is also called Data Mining or Rule Induction

What is Machine Learning?

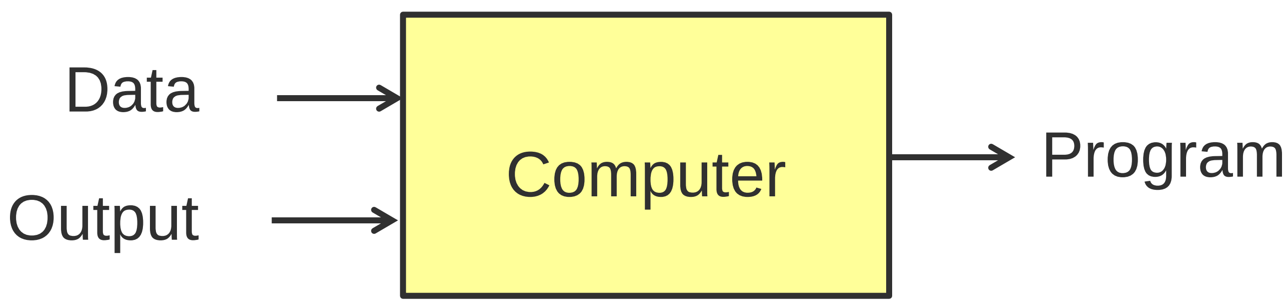
- Determine rules from data/facts
- Improve performance with experience
- Getting computers to program themselves
- …

Machine Learning vs. Programming

Computer Application

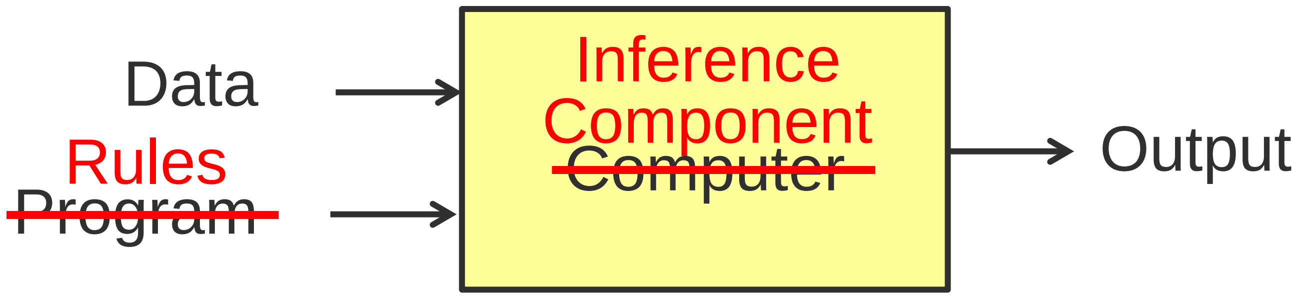


Machine Learning

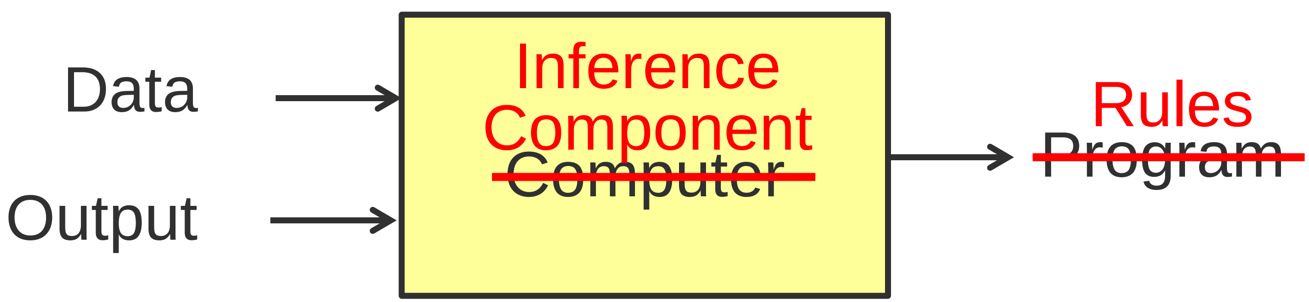


Machine Learning vs. Knowledge-based Systems

Knowledge-based System



Machine Learning



Source: Vibhav Gogate, UT Dallas

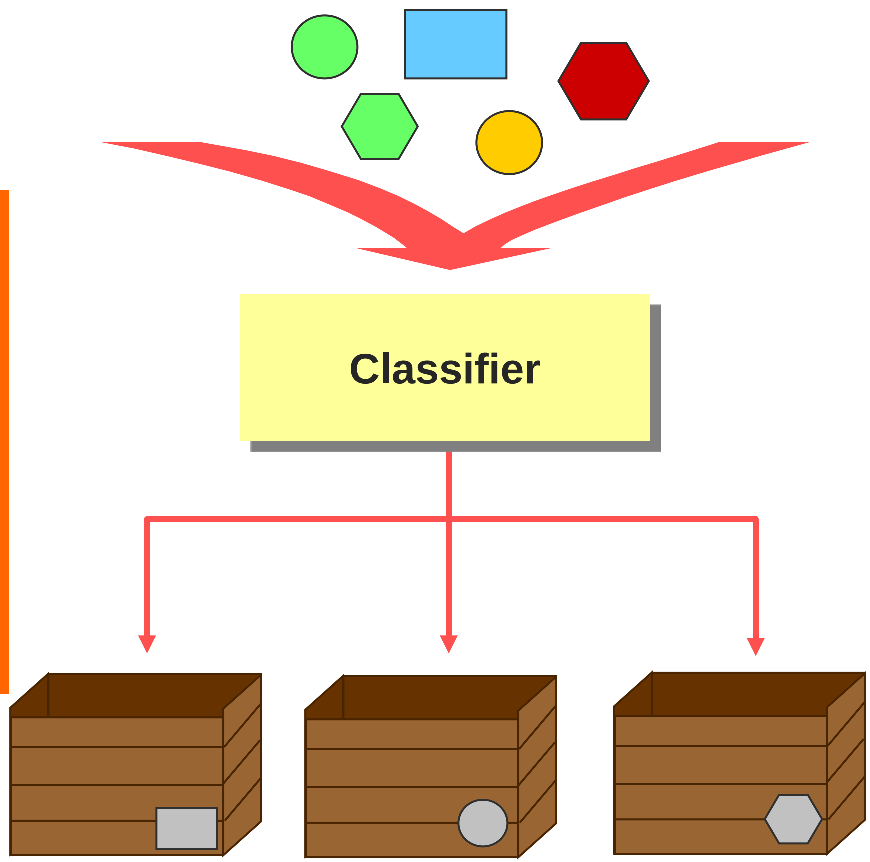
Types of Learning

- Supervised learning
  - Solutions/classes for examples are known
  - Criteria for classes are learned
- Unsupervised
  - No prior knowledge
  - classes have to be determined
- Reinforcement
  - occasional rewards

Classification

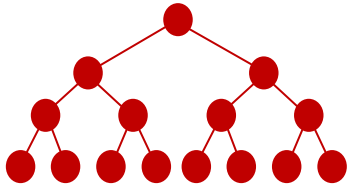
- Task
  - Assign individuals to known classes
- Examples:
  - credit assessment
    Individuals: customers of a bank
    Classes: credit worthy
    not credit worthy
  - quality chekc
    Individuals: products
    Classes: ok
    rework
    defective
  - optical character recognition (OCR)
    Individuals: scan (pixel image)
    Classes: ASCII characters

Supervised Learning: Classification Criteria



- The classifier decides, which individual belongs to which class
- The classifier is a model of the application
  - The classifier codifies the relevant criteria for the classification: class definitions
- Problems:
  - The criteria for the decision are not always obvious
  - The creation of a classifier requires knowledge and effort
- Learning:
  - Learn the classification criteria from known examples

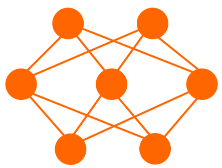
Classification Methods



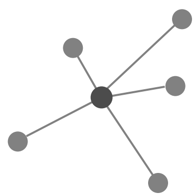
Decision Trees

IF ...
THEN ...

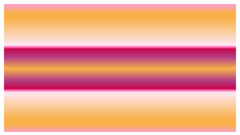
Rules



Neuronale Netze



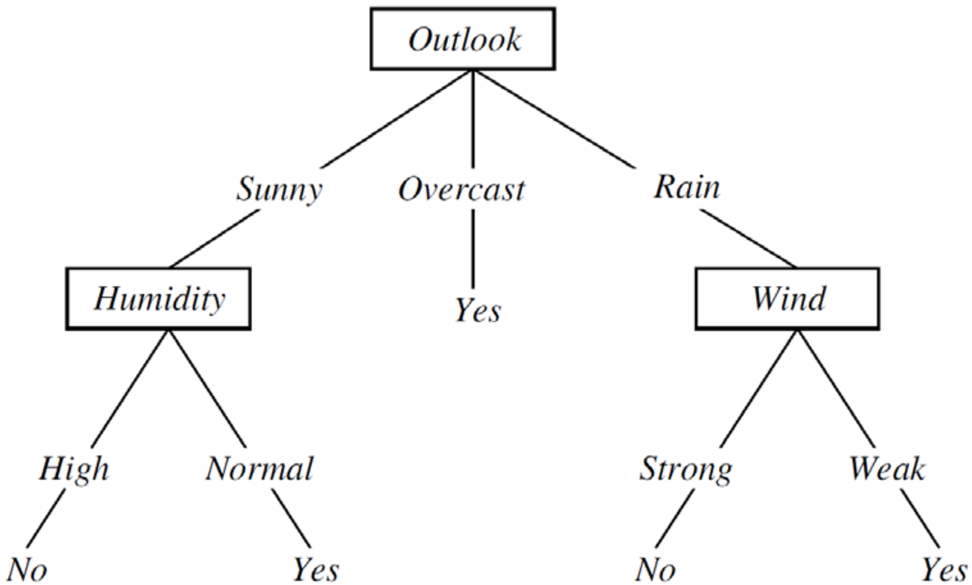
k-Nearest Neighbor



Genetic Algorithms

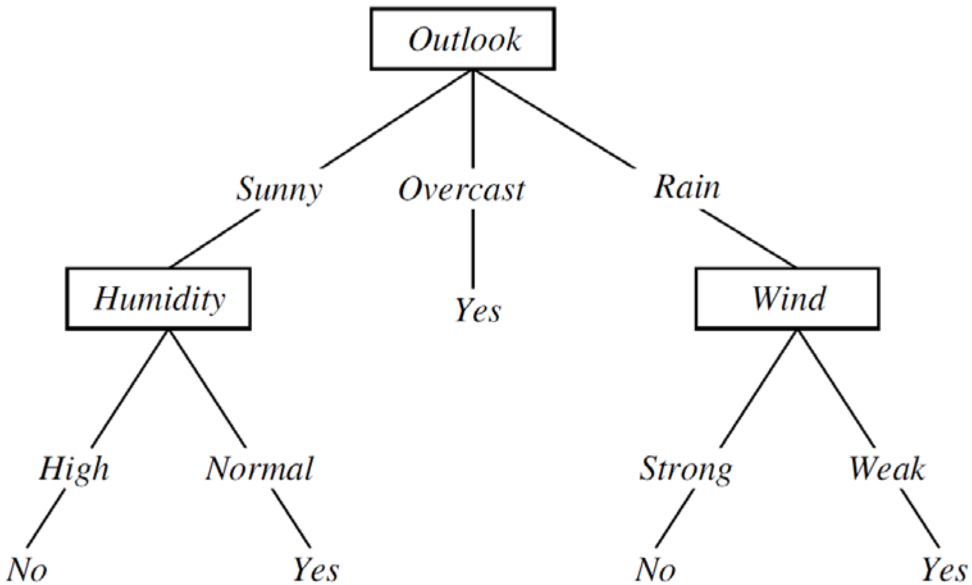
Decision Trees

Example: Decision tree for playing tennis:



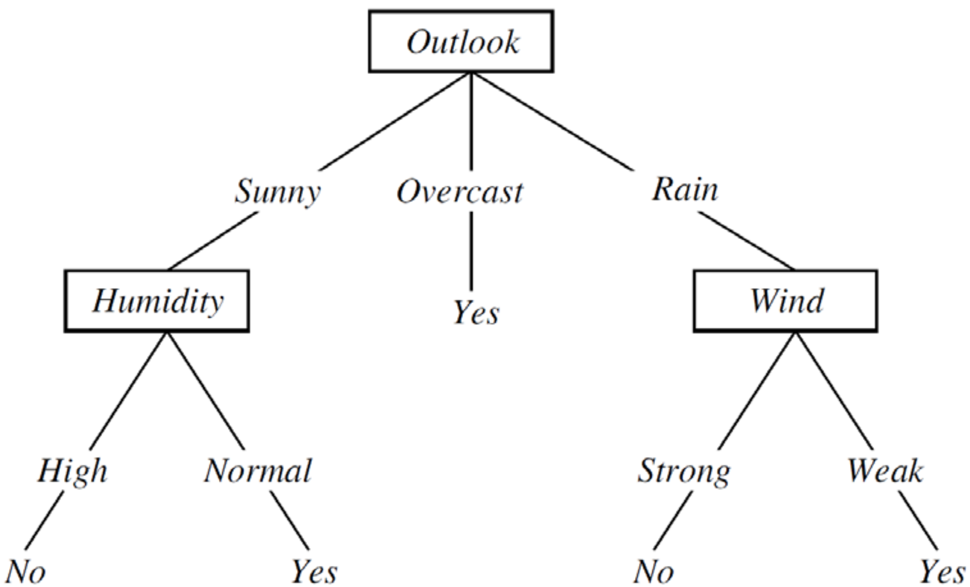
- Decision trees are primarily used for classification
- Decision trees represent classification rules
- Decision tree representation:
  - Each internal node tests an attribute
  - Each branch corresponds to attribute value
  - Each leaf node assigns a classification
- Decision trees classify instances by sorting them down the tree from the root to some leaf node,

Decision Trees represent Rules



- Each path from root to a leaf is a rule
- Each path/rule is a conjunction of attribute tests:
  - **IF** Outlook = Sunny AND
    Humidity = High
    THEN No
  - **IF** Outlook = Sunny AND
    Humidity = Normal
    THEN Yes
  - **IF** Outlook = Overcast
    THEN Yes
  - **IF** Outlook = Rain AND
    Wind = Strong
    THEN No
  - **IF** Outlook = Rain AND
    Wind = Weak
    THEN Yes

Decision Trees represent Rules



- If the classes are boolean, a path can be regarded as a conjunction of attribute tests.
- The tree itself is a disjunction of these conjunctions

$$(\text{Outlook} = \text{Sunny} \land \text{Humidity} = \text{Normal})$$
$$\lor$$
$$(\text{Outlook} = \text{Overcast})$$
$$\lor$$
$$(\text{Outlook} = \text{Rain} \land \text{Wind} = \text{Weak})$$

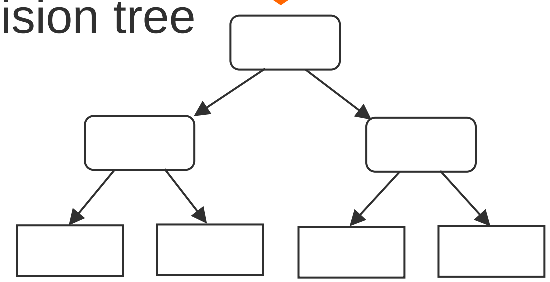
Learning Decision Trees by Induction

Training data

independent			*dependent*
...	...	...	...
...	...	...	...
...	...	...	...

Induction

Decision tree



Induction = Generalisation from examples

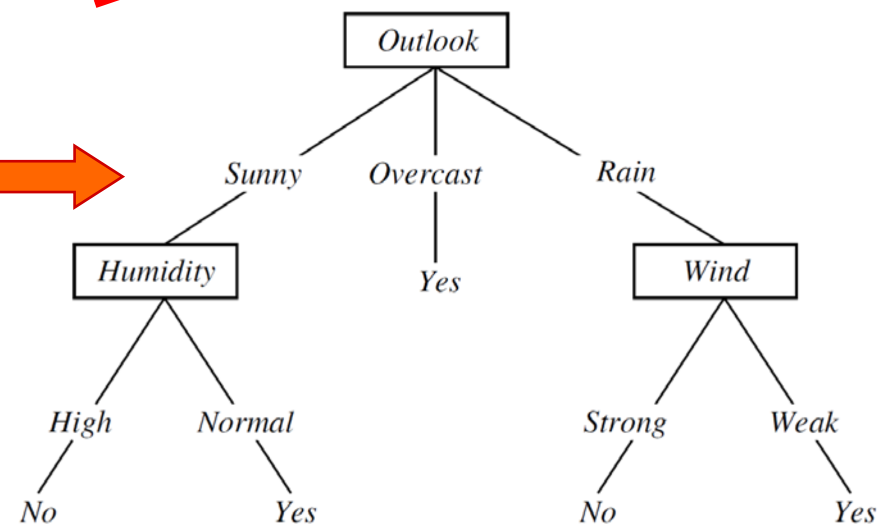
Example

The dependent variable „Tennis“ determines if the weather is good for tennis („Yes“) or not („No“).

Training Data

Element	Outlook	Temperature	Humidity	Wind	Tennis
1	Sunny	Hot	High	Weak	No
2	Sunny	Hot	High	Strong	No
3	Overcast	Hot	High	Weak	Yes
4	Rain	Mild	High	Weak	Yes
5	Rain	Cold	Normal	Weak	Yes
6	Rain	Cold	Normal	Strong	No
7	Overcast	Cold	Normal	Strong	Yes
8	Sunny	Mild	High	Weak	No
9	Sunny	Cold	Normal	Weak	Yes
10	Rain	Mild	Normal	Weak	Yes
11	Sunny	Mild	Normal	Strong	Yes
12	Overcast	Mild	High	Strong	Yes
13	Overcast	Hot	Normal	Weak	Yes
14	Rain	Mild	High	Strong	No

~~Naive approach: Each example data set represents a rule~~



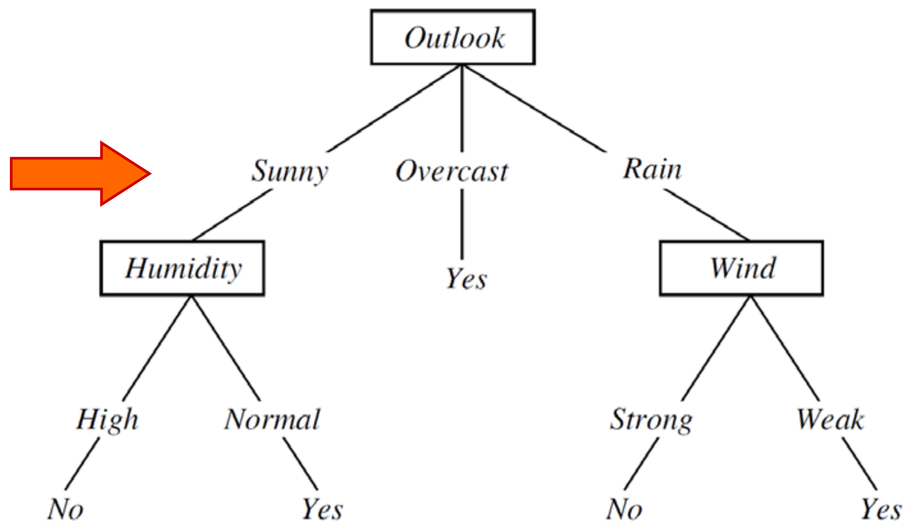
Induction generalizes the data set → prediction of future case

Example

The dependent variable „Tennis“ determines if the weather is good for tennis („Yes“) or not („No“).

Training Data

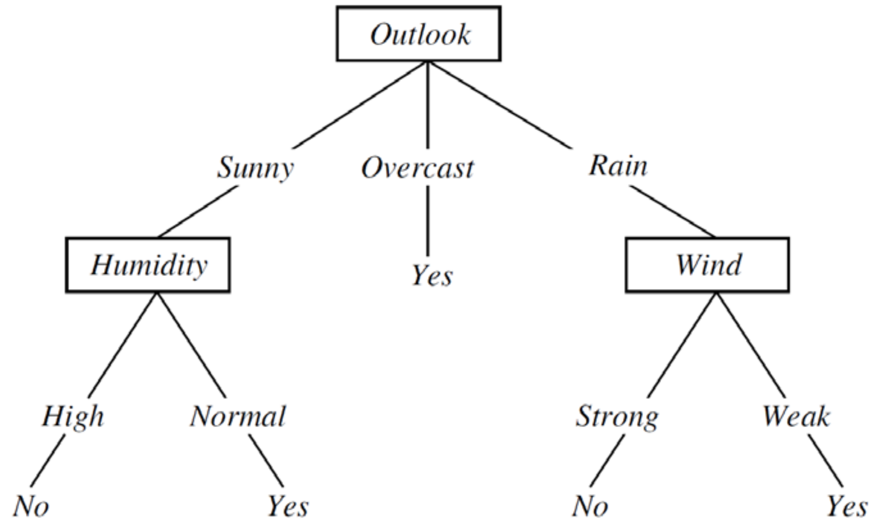
Element	Outlook	Temperature	Humidity	Wind	Tennis
1	Sunny	Hot	High	Weak	No
2	Sunny	Hot	High	Strong	No
3	Overcast	Hot	High	Weak	Yes
4	Rain	Mild	High	Weak	Yes
5	Rain	Cold	Normal	Weak	Yes
6	Rain	Cold	Normal	Strong	No
7	Overcast	Cold	Normal	Strong	Yes
8	Sunny	Mild	High	Weak	No
9	Sunny	Cold	Normal	Weak	Yes
10	Rain	Mild	Normal	Weak	Yes
11	Sunny	Mild	Normal	Strong	Yes
12	Overcast	Mild	High	Strong	Yes
13	Overcast	Hot	Normal	Weak	Yes
14	Rain	Mild	High	Strong	No



The result of the induction algorithms classifies the data with only three of the four attributes into the classes „Yes“ and „No“.

Example: Decision Tree – Decision Table

The decision tree can be represented as a decision table.



Playing Tennis

	Outlook	Humidity	Wind	Tennis
	Sunny,Overcast, Rain	*High, Normal*	*Strong,Weak*	*Yes, No*
1	Sunny	High		No
2	Sunny	Normal		Yes
3	Overcast			Yes
4	Rain		Strong	No
5	Rain		Weak	Yes

Example: Learning Decision Trees

	categorical	categorical	continuous	class
Tid	**Employed**	**Marital Status**	**Taxable Income**	**accept**
1	No	Single	125K	**No**
2	Yes	Married	160K	**Yes**
3	Yes	Single	70K	**No**
4	No	Married	120K	**No**
5	Yes	Divorced	95K	**Yes**
6	Yes	Married	60K	**No**
7	No	Divorced	220K	**No**
8	Yes	Single	85K	**Yes**
9	Yes	Married	95K	**No**
10	Yes	Single	90K	**Yes**

Training Data

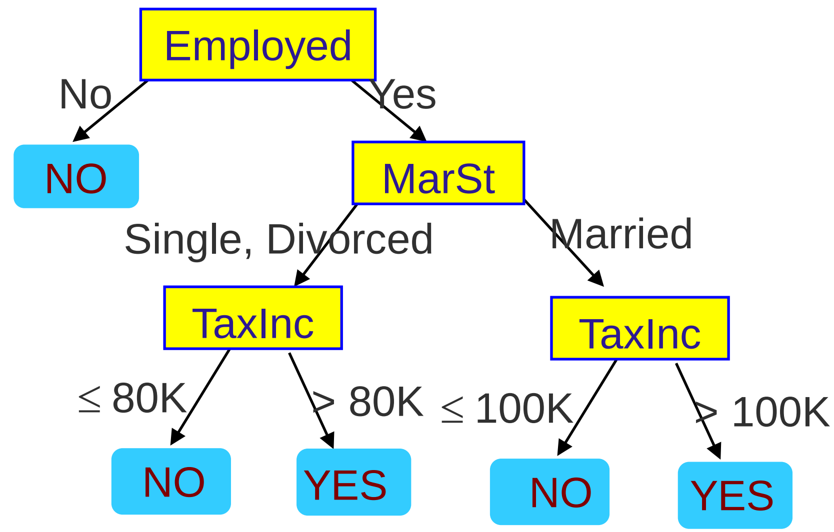
Decision Tree?

Learning Decision Trees: Generalisation of Data

categorical categorical continuous class

Tid	Employed	Marital Status	Taxable Income	accept
1	No	Single	125K	No
2	Yes	Married	160K	Yes
3	Yes	Single	70K	No
4	No	Married	120K	No
5	Yes	Divorced	95K	Yes
6	Yes	Married	60K	No
7	No	Divorced	220K	No
8	Yes	Single	85K	Yes
9	Yes	Married	95K	No
10	Yes	Single	90K	Yes

Training Data



Credit Worthiness

	Employed	Marital Status	Taxable Income	Accept
	Yes, No	*Single, Divorced, Married*	*Integer*	*Yes, No*
1	No			No
2	Yes	Single	> 80K	Yes
3	Yes	Divorced	> 80K	Yes
4	Yes	Single	≤ 80K	No
5	Yes	Divorced	≤ 80K	No
6	Yes	Married	> 100K	Yes
7	Yes	Married	≤ 100K	No

Model: Decision Tree/Table

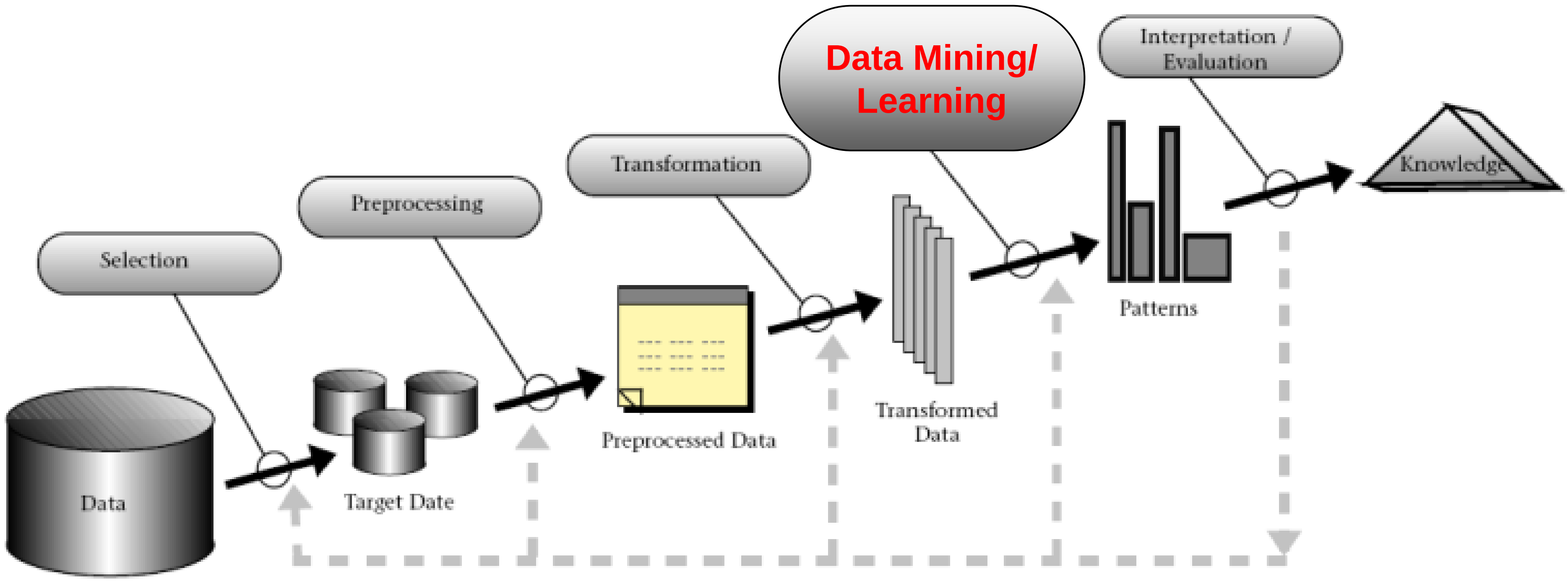
The model uses intervals instead of concrete numerical data

Predictive Model for Classification

- Given a collection of training records (*training set*)
  - Each record consists of *attributes*, one of the attributes is the *class*
  - The class is the dependent attribute, the other attributes are the independent attributes
- Find a *model* for the class attribute as a function of the values of the other attributes.
- Goal: to assign a class to **previously unseen records** as accurately as possible.
- Generalisation of data if training set does not cover all possible cases or data are too specific
  - → Induction

Knowledge Discovery in Data

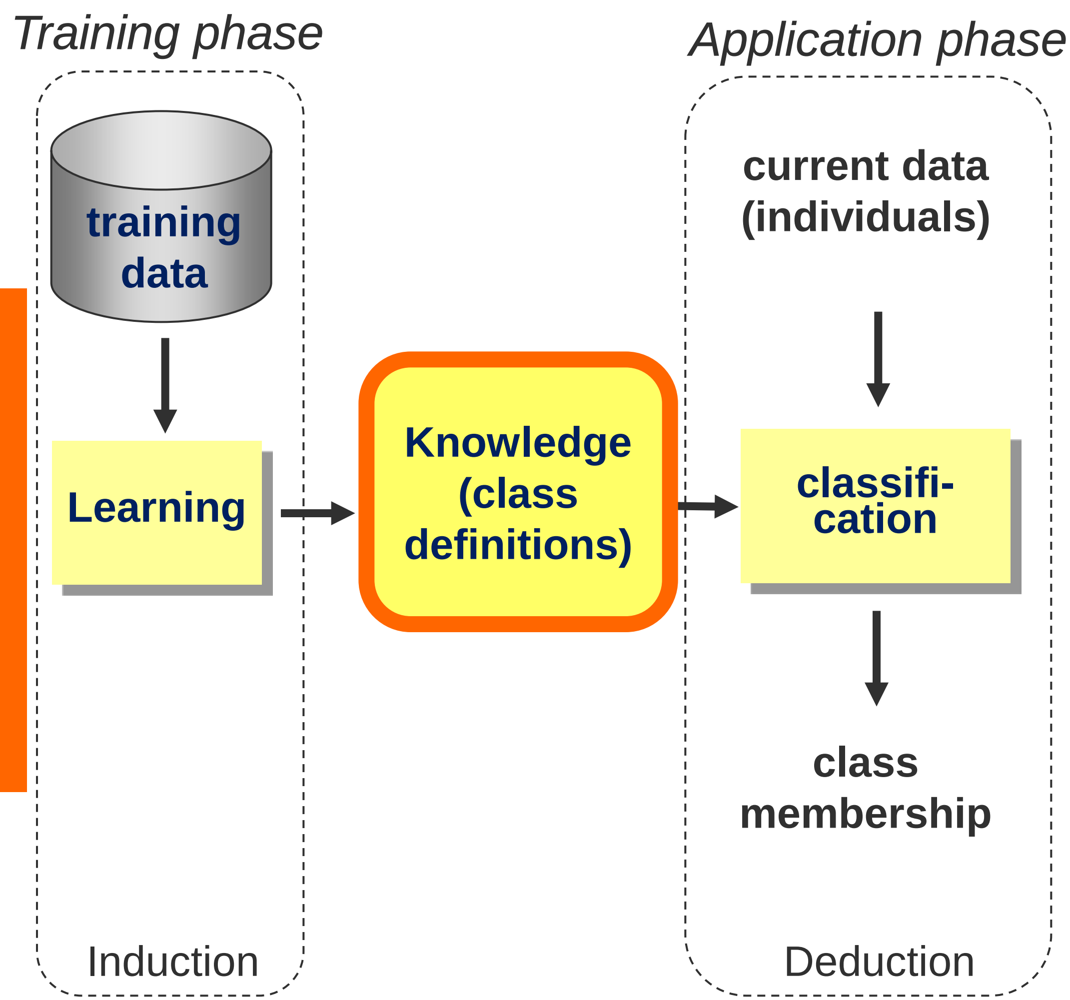
- Data Mining/Machine Learning is a step to discover knowledge in data



Knowledge is then used in processes and applications.

(Fayyad et al., 1996)

Training and Application Phase



- **Training:** Learning the criteria for the assignment of individuals to classes
  - Given: a sample set of individuals
    - described by attribute values (independent variables)
    - class membership (dependent variables)
  - Result: class definitions (decision tree, rules, model)
- **Application:** Assignment of individuals to classes (classification)
  - Given: Description of individuals by attribute values
  - Result: class membership
- During training the class membership is given, during application the class membership is calculated

Induction of Decision Tree

- Enumerative approach
  - Create all possible decision trees
  - Choose the tree with the least number of questions

  This approach finds the best classifying tree, but it is inefficient.

- Heuristic approach:
  - Start with an empty root and extend the tree step by step with new decision nodes
  - Stop, if the desired homogenity is achieved

  This approach is efficient, but does not necessariy find the best classifying tree.

Sketch of an Induction Algorithmus

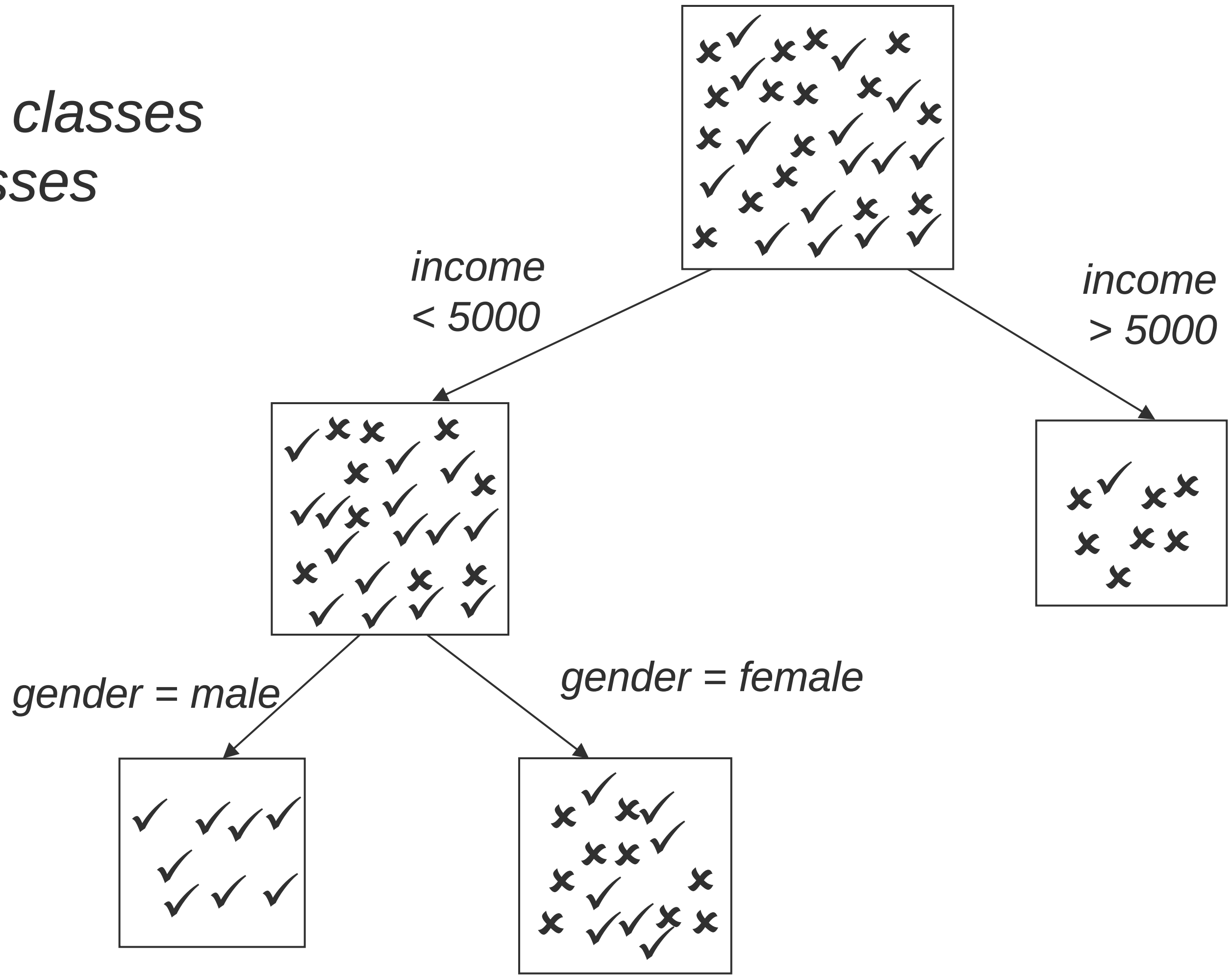
Heuristic Approach

Learning a Decision Tree

- Calculate for each attribute, how *good* it classifies the elements of the training set
- Classify with the *best* attribute
- *Repeat* for each resutling subtree the first two steps
- Stop this recursive process as soon as a *termination condition* is satisfied

Learning of Decision Tree

Principle:
From heterogeous classes
to homogeous classes

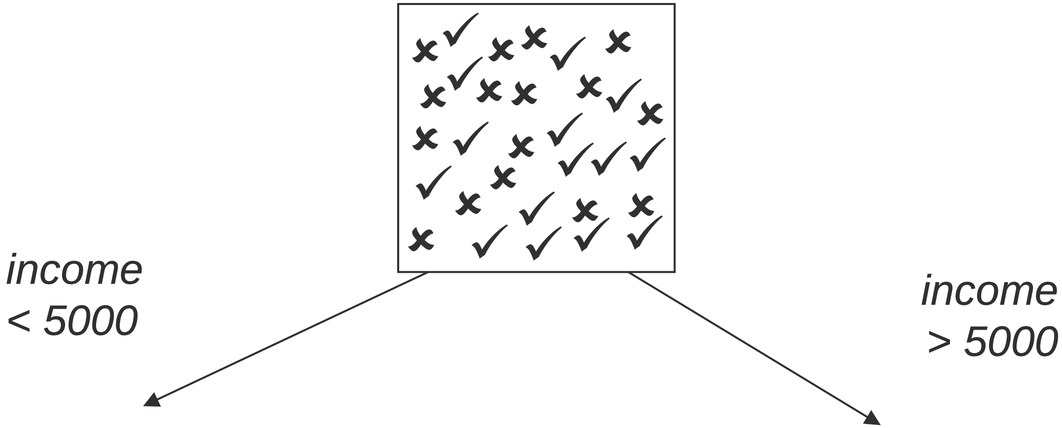


Types of Data

- Discrete: endliche Zahl möglicher Werte
  - Examples: marital status, gender
  - Splitting: selection of values or groups of values
- Numeric infinite number of values on which an order is defined
  - Examples: age, income
  - Splitting: determine interval boundaries

For which kind of attributes is splitting easier?

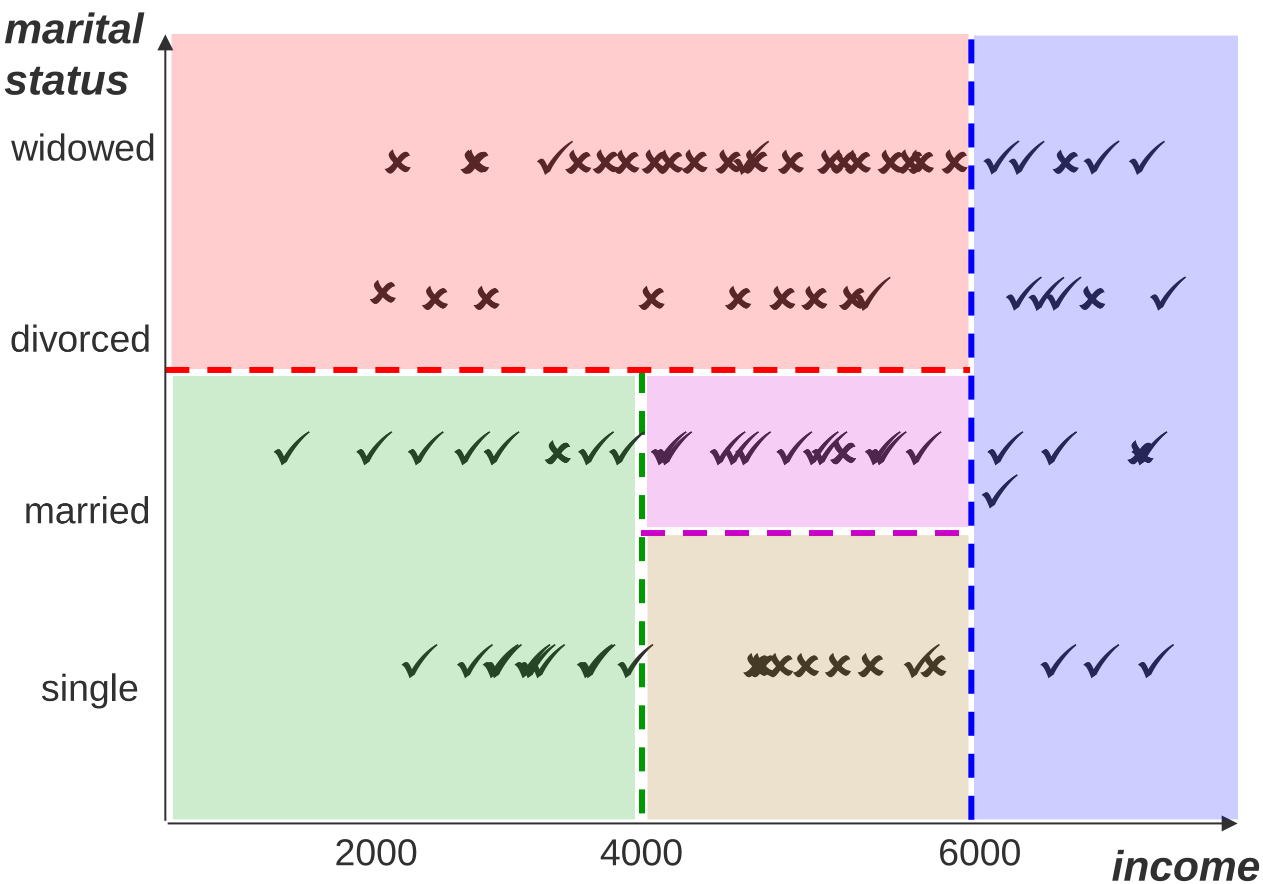
Determine how to split the Records in a Decision Tree



- **Attribute selection**
  - Which **attributes** separate best in which order?
    - e.g. income before marital status
- **Test condition**
  - Which **values** separate best?
    - select value, e.g. single or married
    - determine number: e.g. income < 5000 instead of < 6000?

Creation of Decision Trees

Each decision divides the area in sections



IF income > 6000
THEN accept

IF income < 6000 and
marital status = widowed or
marital status = divorced
THEN reject

IF income < 4000 and
marital status = single or
marital status = married
THEN accept

IF income > 4000 and
income < 6000 and
marital status = married
THEN accept

IF income > 4000 and
income < 6000 and
marital status = single
THEN reject

Generating Decision Trees

- ID3 is a basic decision learning algorithm.
- It recursively selects test attributes and begins with the question "*which attribute should be tested at the root of the tree?* "
- ID3 selects the attribute with the highest
  - **Information Gain**
- To calculate the information gain of an attribute one needs
  - the **Entropy** of a classification
  - the **Expectation Value** of the attribute

ID3: Induction Algorithm for Decision Trees

- **Create a decision from all data**

  Randomly choose a training set from the whole data set

  Create a decision tree from the training set

  UNTIL the decision tree correctly classifies all data

      Add selected wrongly classified elements to the training set

      Create a decision tree from the new training set

- **Create** a decision tree

  FOR EACH attribute

      Calculate the Information Gain

          create a decision tree for the partition

      IF the partition contains only positive instances

          Mark the note as +

      IF the partition contains only negative instances

          Mark the note as –

A basic Decision Tree Learning Algorithm

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 *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*

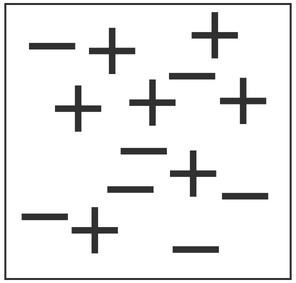
Entropy („disorder“)

- Entropy is a measure of *unpredictability* of *information content*.
  - The higher the information content, the lower the entropy
- The more classification information a decision tree contains, the smaller the entropy
- The goal of ID3 is to create a tree with minimal entropy

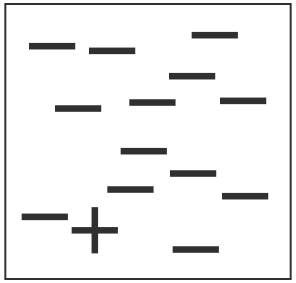
Entropy increases with increasing Equality of Distribution

- Assume there are two classes + und –
- An uninformed classifier will assign the individuals randomly to the classes + and –
- It is thus plausible, that the entropy is smaller the more the frequencies p (of +) and n (of –) for each class are different from equal distribution.
- The more unequal p and n, the smaller is the entropy

high entropy



low entropy



Calculation of the Entropy for binary Classification

- Assume a decision tree which classifies the training set into to classes + (positive) and – (negative)
- The entropy is calculated by

Entropy (S) = $- p_{+} * \log_2(p_{+}) - p_{-} * \log_2(p_{-})$

S = p + n is the number of all elements

p frequency of elements of class +

n frequency of elements of class –

p+ = p / S and p- = n / S are the relative frequencies, i.e. the proportions of values of classes + and –

Entropy Calculation for different Distributions

- The more different p and n, the lower is the entropy

p	*n*	*p+*	*ld(p+)*	*p-*	*ld(p-)*	*Entropy(p+n)*
7	7	0.5	-1	0.5	-1	1
6	8	0.43	-1.22	0.57	-0.81	0.99
5	9	0.36	-1.49	0.64	-0.64	0.94
4	10	0.29	-1.81	0.71	-0.49	0.86
3	11	0.21	-2.22	0.79	-0.35	0.75
2	12	0.14	-2.81	0.86	-0.22	0.59
1	13	0.07	-3.81	0.93	-0.11	0.37

ld = $\log_2$ (logarithmus dualis)
ld(0) cannot be calculated, but for p = 0 or n = 0 no classification is necessary

Expectation Value

The expectation value measures the information, which is needed for classification with attribute A

Expectation Value

- Let A be an attribute with m possible values $v_1, \ldots, v_i, \ldots v_m$
  - *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*
- The attribute A divides the elements into m partitions (subtrees)
- Entropy(S_v) = entropy of the subtree created by the attribute value v
- **Expectation Value EV_A** of the required information for the classification of the root attribute A

$$\mathrm{EV(A)} := \sum_{v \in Values(A)} \frac{|S_v|}{|S|} Entropy(S_v)$$

- The expectation value is the weighted average of the entropies of the subtrees created by v_i

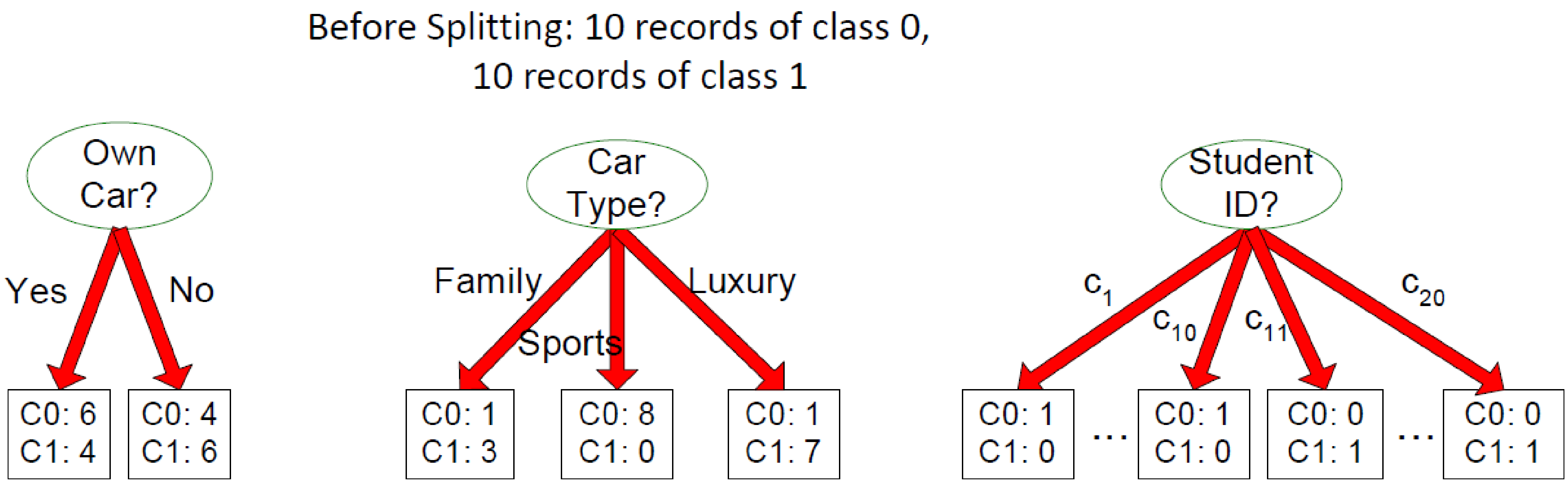
Information Gain

- The information gain measures how well a given attribute A separates the training examples according to their target classification.
- The information gain is calculated by subtracting the expectation value of the subtrees created by A from the entropy of the tree with root A

$$GAIN\ (S, A) = Entropy\ (S) - EV(A)$$

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

Exercise



- Which test condition is the best?

Thanks to Nadeem Qaisar Mehmood

ID3: Information Gain for Attribute Selection

- ID3 uses the Information Gain to select the test attribute

On each level of the tree select the attribute with the highest information gain

- The recursive calculation of the attributes stops when either
  - all partitions contain only positive or only negative elements or
  - a user-defined threshold is achieved

Computing Entropy: Example

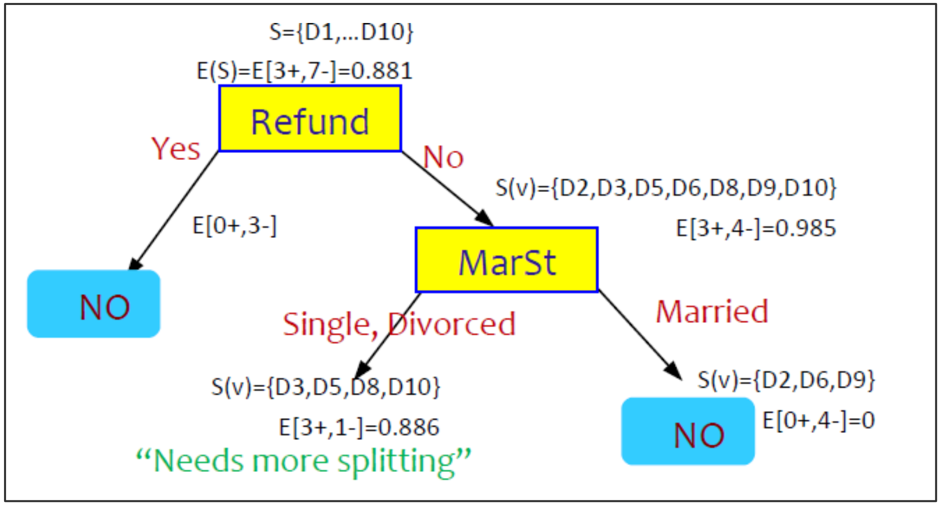
Tid	Refund	Marital Status	Taxable Income	Cheat
D1	Yes	Single	125K	No
D2	No	Married	100K	No
D3	No	Single	70K	No
D4	Yes	Married	120K	No
D5	No	Divorced	95K	Yes
D6	No	Married	60K	No
D7	Yes	Divorced	220K	No
D8	No	Single	85K	Yes
D9	No	Married	75K	No
D10	No	Single	90K	Yes

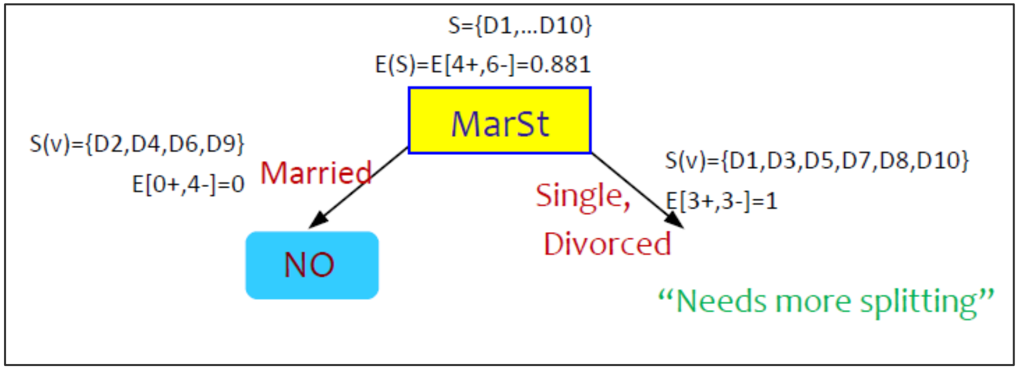
Training Data (S)

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





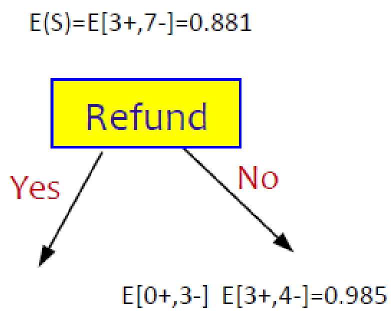
[Tan&Steinbach's "Intro to Data Mining"]

Thanks to Nadeem Qaisar Mehmood

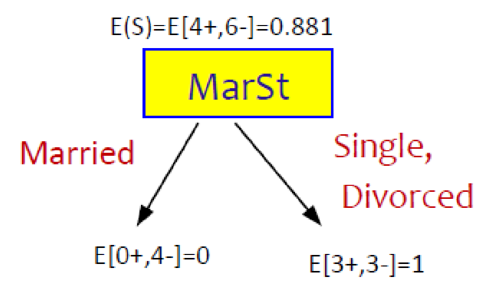
Get Information Gain using Entropy

- 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.

Thanks to Nadeem Qaisar Mehmood

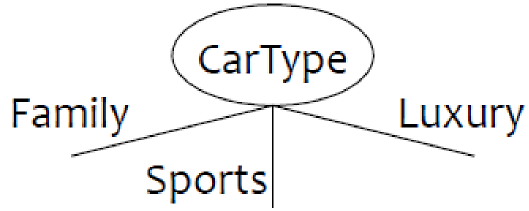
How to specify Attribute Test Conditions

Specification of the test condition depends on

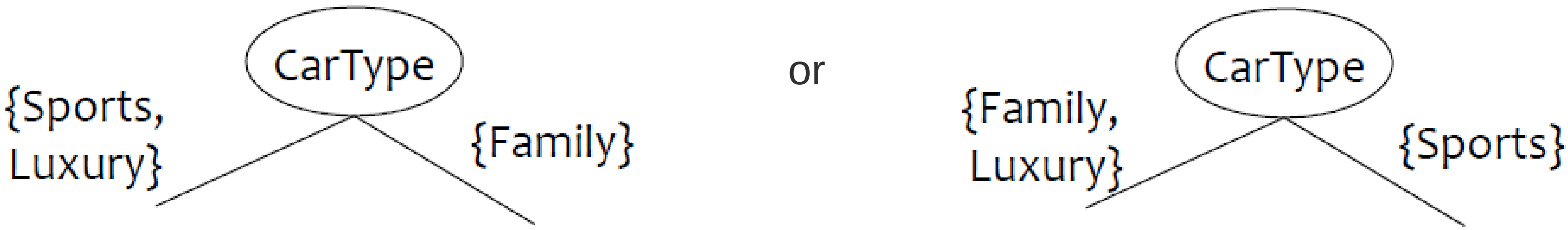
- attribute types
  - Nominal
  - Ordinal
  - Continuous
- number of ways to split
  - 2-way split
  - Multi-way split

Splitting for Nominal Attributes

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

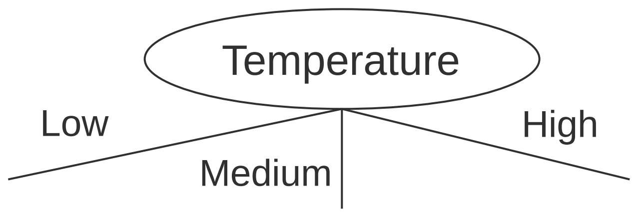


- Binary split: Divides values into two subsets. Need to find optimal partitioning.

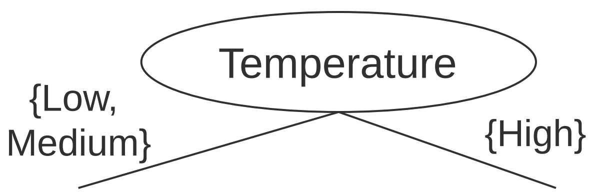


Splitting for Ordinal Attributes

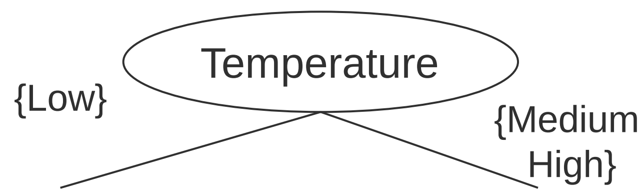
- Multi-way split: Use as many partitions as distinct values.



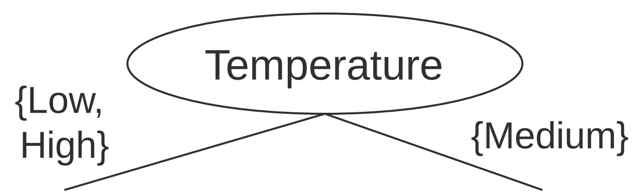
- Binary split: Divides values into two subsets. Need to find optimal partitioning.



OR

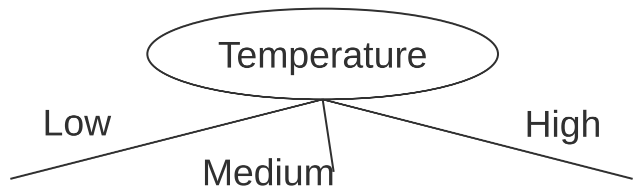


What about this split?

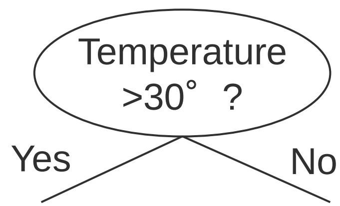


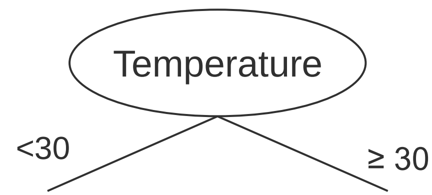
Splitting for Continuous Attributes

- Different ways of handling
  - Discretization to form an ordinal categorical attribute

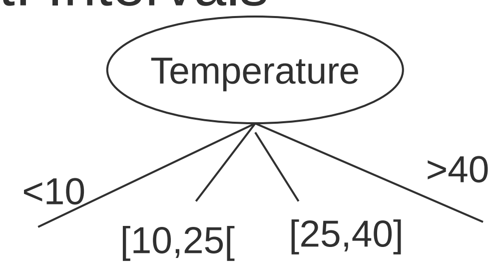


  - Binary Decision: (A < v) or (A ≥ v)



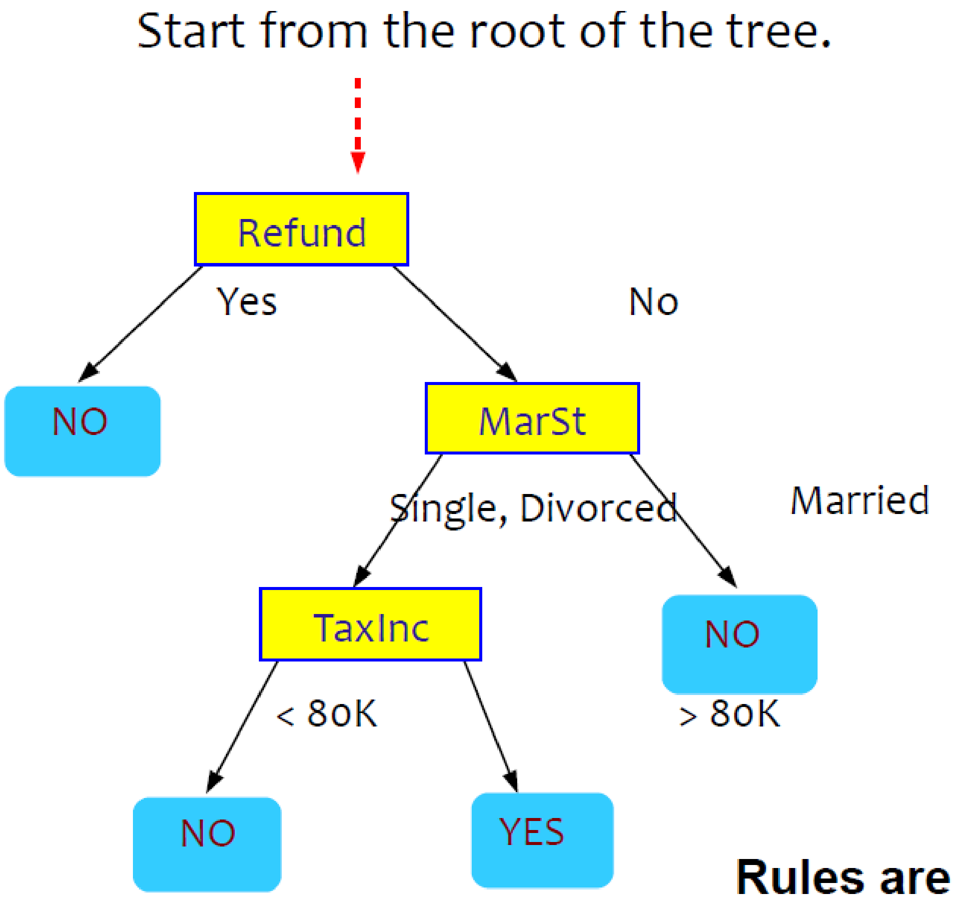


  - Multi-way Split: Intervals



considering all possible splits and finding the best cut can be computing intensive

Decision Tree represented in Rules form



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

Preference for Short Trees

- Preference for short trees over larger trees, and for those with high information gain attributes near the root
- **Occam's Razor:** Prefer the simplest hypothesis that fits the data.
- Arguments in favor:
  - a short hypothesis that itts data is unlikely to be a coincidence – compared to long hypothesis
- Arguments opposed:
  - There are many ways to define small sets of hypotheses

Overfitting

- When there is ***noise in the data***, or when the number of training ***examples is too small*** to produce a representative sample of the true target function, the rule set (hypothesis) ***overfits*** the training examples!!
- Consider error of hypothesis *h* over
  - training data: *errortrain*(h)
  - entire distribution D of data: *errorD*(h)
- Hypothesis *h* OVERFITS training data if there is an alternative hypothesis h*0* such that
  - *errortrain*(*h*) < *errortrain*(*h0*)
  - *errorD*(*h*) > *errorD*(*h0*)

Avoiding Overfitting by Pruning

- The classification quality of a tree can be improved by cutting weak branches
- Reduced error pruning
  - remove the subtree rooted at that node,
  - make it a leaf,
  - assign it the most common classification of the training examples afiliated with that node.
- To test accuracy, the data are separated in training set and valication set. Do until further pruning is harmful:
  - Evaluate impact on *validation* set of pruning each possible node
  - Greedily remove the one that most improves *validation* set accuracy

Training and Validation

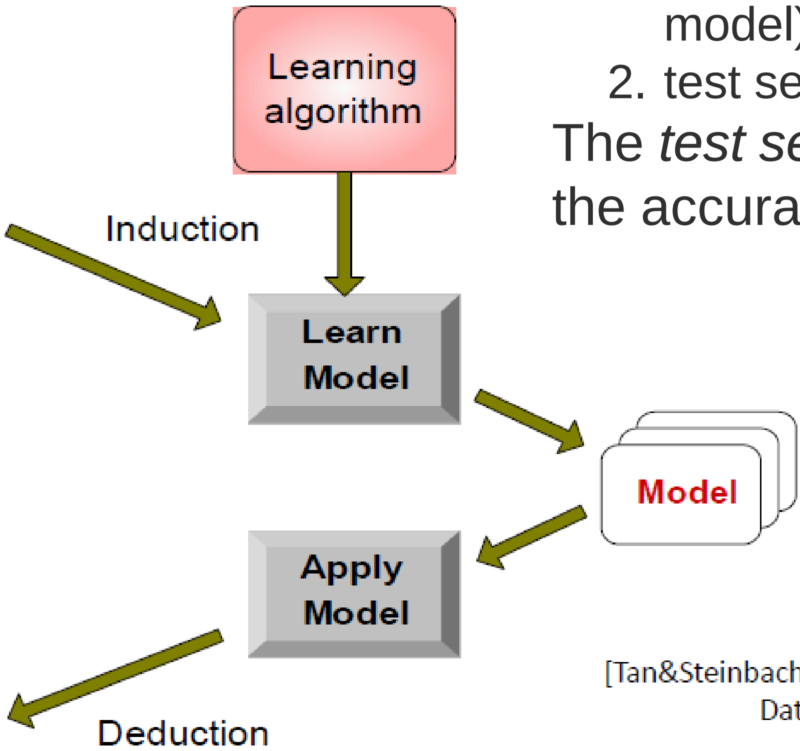
Usually, the given data set is divided into

1. training set (used to build the model)
2. test set (used to validate it)

The *test set* is used to determine the accuracy of the model.

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



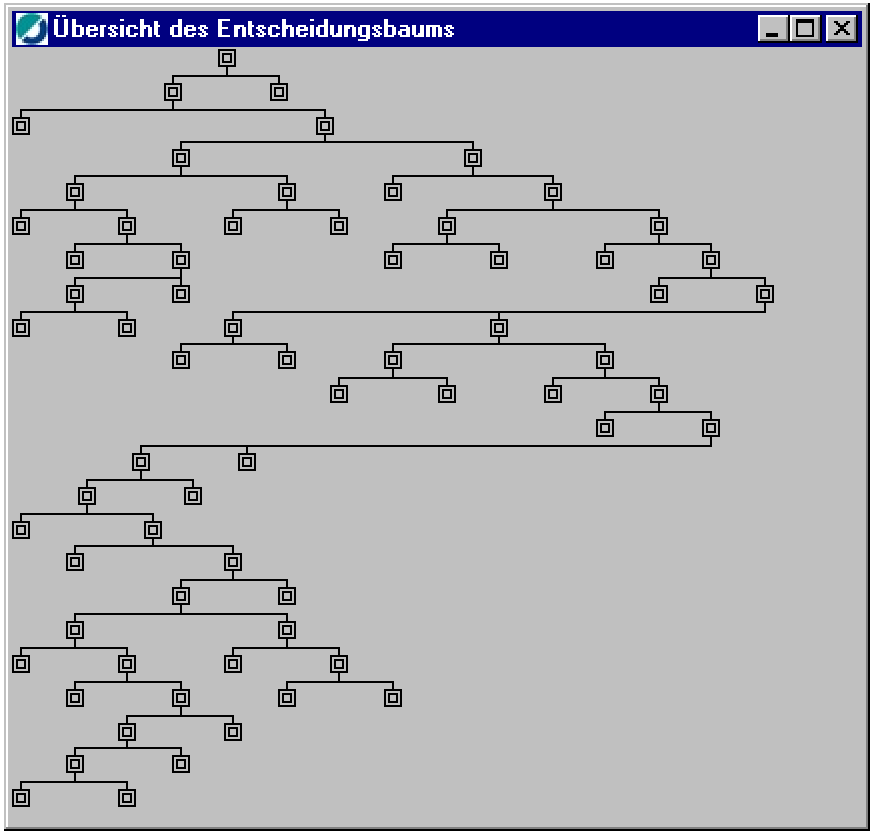
Tid	Attrib1	Attrib2	Attrib3	Class
11	No	Small	55K	?
12	Yes	Medium	80K	?
13	Yes	Large	110K	?
14	No	Small	95K	?
15	No	Large	67K	?

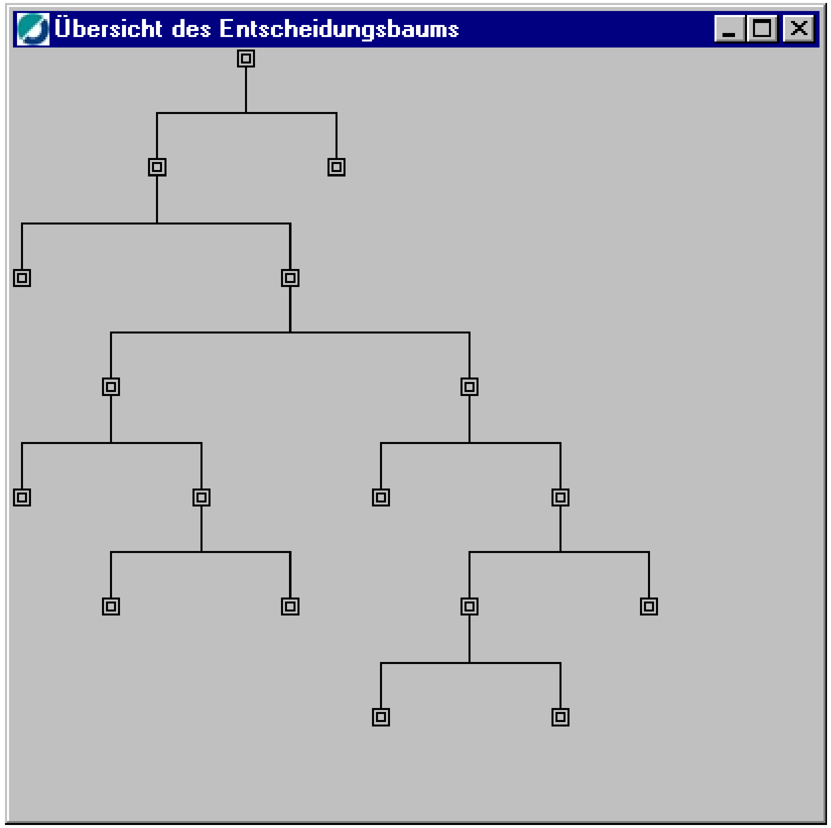
Test Set

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

Pruning

These figures shoe the structure of a decision tree before and after pruning





Generalisations

- Multiple Classes
  - Although the examples had only two classes, decision tree learning can be done also for more than two classes
  - Example: Quality
    - okay, rework, defective
- Probability
  - The examples only had Boolean decisions
    - Example: **IF** income > 5000 and age > 30
      THEN creditworthy
  - Generalisation: Probabilties for classification
    - Example: **IF** income > 5000 and age > 30
      THEN creditworthy with probability 0.92

Algorithms for Decision Tree Learning

- Examples os algorithms for learning decision trees
  - C4.5 (successor of ID3, predecessor of C5.0)
  - CART (Classification and Regression Trees)
  - CHAID (CHI-squared Automatic Interaction Detection)
- A comparison [1)] of various algorithms showed that
  - the algorithms are similar with respect to classification performance
  - pruning increases the performance
  - performance depends on the data and the problem.

1) D. Michie, D.J. Spiegelhalter und C.C. Taylor: Machine Learning, Neural and Statistical Classificaiton, Ellis Horwood 199

- Entroy (S) = 1
- EV(OC) = 0.5 * E(S1) + 0.5 *E(S0)
- =0.5 *(-0.6 log(0.6) – 0.4 log(0.4) + 0.5 (*(-0.4 log(0.4) – 0.6 log(0.6) = 0.97