Lecture 4:

Data classification (II)

Outline

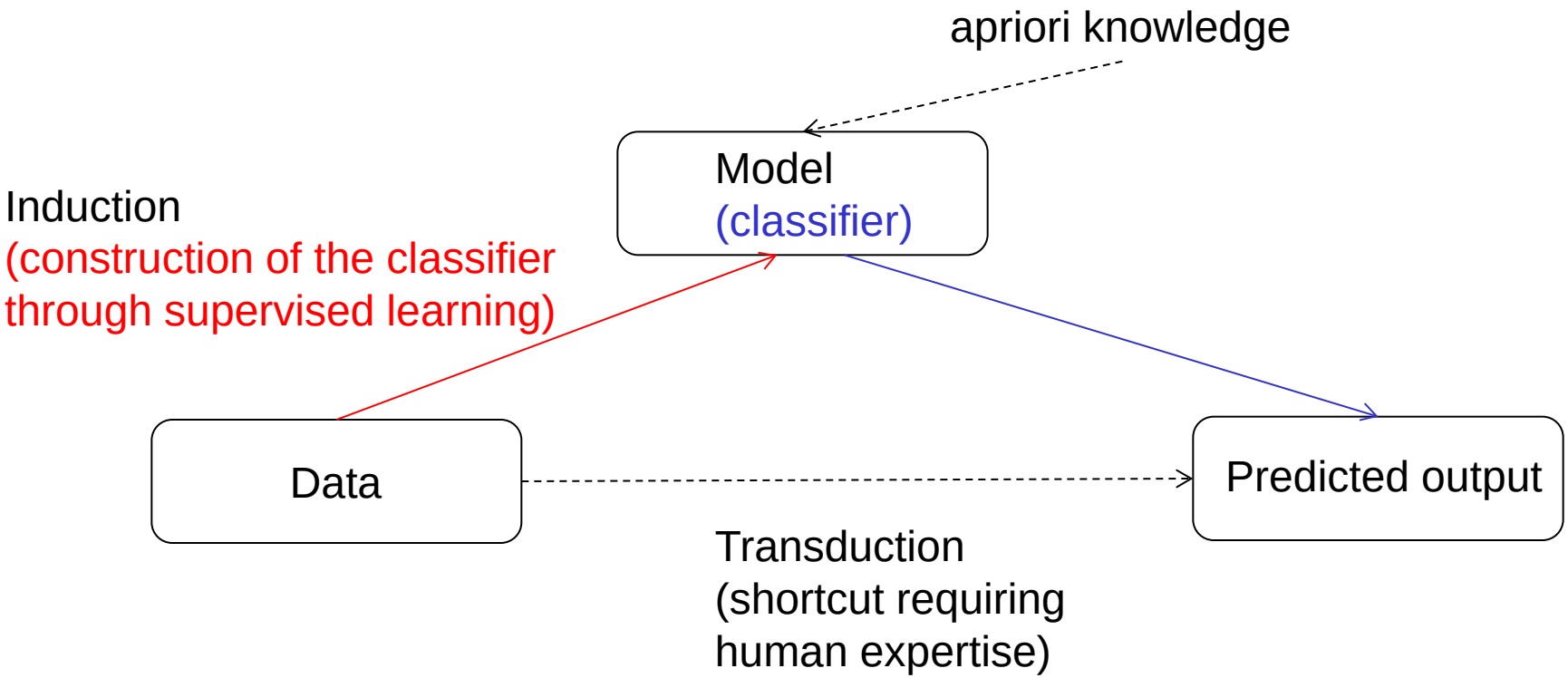
- Decision trees
  - Choice of the splitting attribute
  - ID3
  - C4.5
- Classification rules
  - Covering algorithms
- Naïve Bayes Classification

Reminder: classification models

Learning/ induction/ inference = construct a model starting from data (and some apriori knowledge specific to the domain)

Different ways of using data, models and knowledge:

induction vs deduction vs transduction



Reminder: classification models

A classification model is a "mapping" between attributes and class labels

Example of classification models:

- Decision trees
- Classification rules
- Prototypes (exemplars)
- Probabilistic models
- Neural networks etc.

The classification model should be:

- Accurate:
  - Identify the right class
- Compact / comprehensible
  - Easy to be understood/ interpreted by the user (it is preferable to not be a black box)
- Efficient in the
  - Learning/training step
  - Classification step

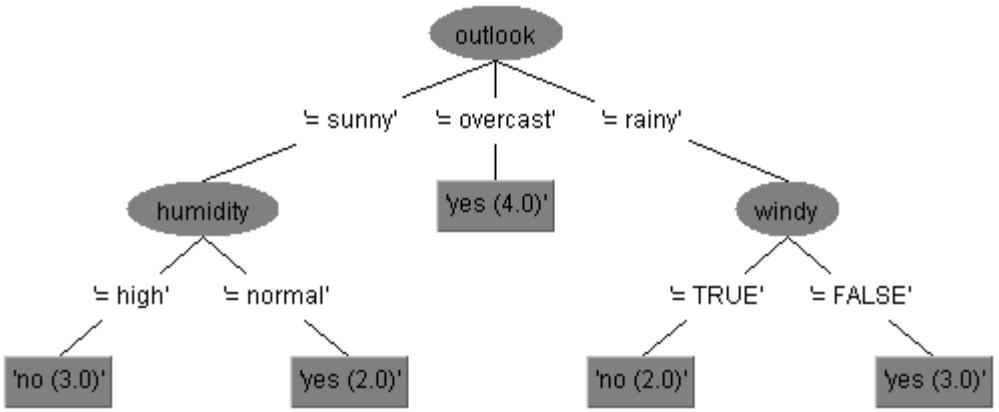
A simple example

Weather/play dataset

Relation: weather.symbolic

No.	outlook Nominal	temperature Nominal	humidity Nominal	windy Nominal	**play** Nominal
1	sunny	hot	high	FALSE	no
2	sunny	hot	high	TRUE	no
3	overcast	hot	high	FALSE	yes
4	rainy	mild	high	FALSE	yes
5	rainy	cool	normal	FALSE	yes
6	rainy	cool	normal	TRUE	no
7	overcast	cool	normal	TRUE	yes
8	sunny	mild	high	FALSE	no
9	sunny	cool	normal	FALSE	yes
10	rainy	mild	normal	FALSE	yes
11	sunny	mild	normal	TRUE	yes
12	overcast	mild	high	TRUE	yes
13	overcast	hot	normal	FALSE	yes
14	rainy	mild	high	TRUE	no

Decision tree (constructed using Weka)



How can be used?

Which class corresponds to a new instance:
(outlook=sunny, temperature=mild, humidity=normal, windy=False)?

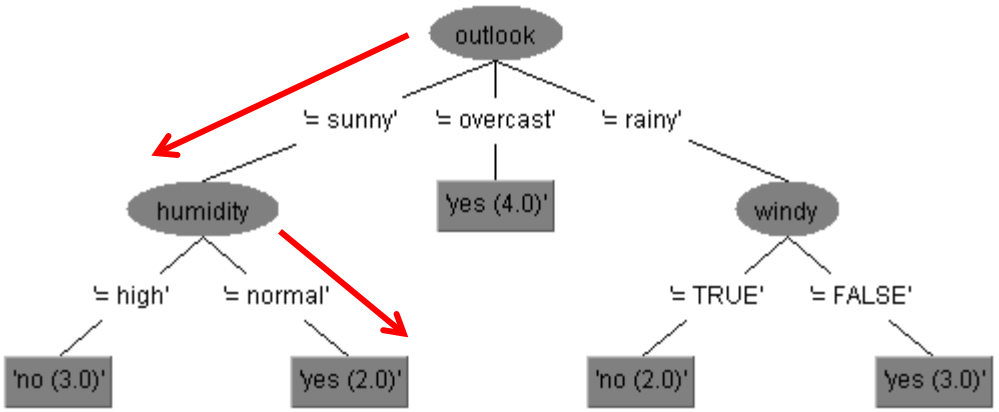
A simple example

Weather/play dataset

Relation: weather.symbolic

No.	outlook Nominal	temperature Nominal	humidity Nominal	windy Nominal	**play** Nominal
1	sunny	hot	high	FALSE	no
2	sunny	hot	high	TRUE	no
3	overcast	hot	high	FALSE	yes
4	rainy	mild	high	FALSE	yes
5	rainy	cool	normal	FALSE	yes
6	rainy	cool	normal	TRUE	no
7	overcast	cool	normal	TRUE	yes
8	sunny	mild	high	FALSE	no
9	sunny	cool	normal	FALSE	yes
10	rainy	mild	normal	FALSE	yes
11	sunny	mild	normal	TRUE	yes
12	overcast	mild	high	TRUE	yes
13	overcast	hot	normal	FALSE	yes
14	rainy	mild	high	TRUE	no

Decision tree (constructed using Weka)



How can be used?

Which class corresponds to a new instance
(outlook=sunny, temperature=mild, humidity=normal, windy=False)?

Class: Yes

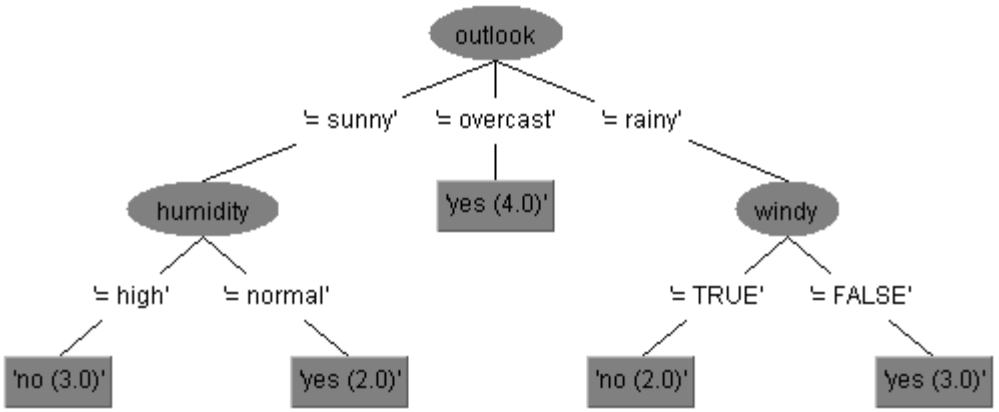
A simple example

Weather/play dataset

Relation: weather.symbolic

No.	outlook Nominal	temperature Nominal	humidity Nominal	windy Nominal	**play** Nominal
1	sunny	hot	high	FALSE	no
2	sunny	hot	high	TRUE	no
3	overcast	hot	high	FALSE	yes
4	rainy	mild	high	FALSE	yes
5	rainy	cool	normal	FALSE	yes
6	rainy	cool	normal	TRUE	no
7	overcast	cool	normal	TRUE	yes
8	sunny	mild	high	FALSE	no
9	sunny	cool	normal	FALSE	yes
10	rainy	mild	normal	FALSE	yes
11	sunny	mild	normal	TRUE	yes
12	overcast	mild	high	TRUE	yes
13	overcast	hot	normal	FALSE	yes
14	rainy	mild	high	TRUE	no

Decision tree (constructed using Weka)



How can be translated in a set of classification rules? Each branch leads to a rule

Rule 1: IF outlook=sunny and humidity=high THEN play=no
Rule 2: IF outlook=sunny and humidity=normal THEN play=yes
Rule 3: IF outlook=overcast THEN play=yes
Rule 4: IF outlook=rainy and windy=True THEN play=no
Rule 5: IF outlook=rainy and windy=False THEN play=yes

A simple example

How can a decision tree be constructed (inferred) from a dataset?

- Choose an attribute and put it as the root of the tree
- For each possible value of the attribute (present in the dataset) construct a branch (split the node)
- Split the dataset in subsets corresponding to each branch
  - If a subset contains data from only one class then it will correspond to a leaf node (no more splitting on that branch) – pure node
  - If in a subset there are data belonging to different classes then the splitting process is continued until
    - is arrived to a pure node
    - all attributes have been analyzed on that branch
    - the subset of data corresponding to that branch is empty

Weather/play dataset

Relation: weather.symbolic

No.	outlook Nominal	temperature Nominal	humidity Nominal	windy Nominal	**play** Nominal
1	sunny	hot	high	FALSE	no
2	sunny	hot	high	TRUE	no
3	overcast	hot	high	FALSE	yes
4	rainy	mild	high	FALSE	yes
5	rainy	cool	normal	FALSE	yes
6	rainy	cool	normal	TRUE	no
7	overcast	cool	normal	TRUE	yes
8	sunny	mild	high	FALSE	no
9	sunny	cool	normal	FALSE	yes
10	rainy	mild	normal	FALSE	yes
11	sunny	mild	normal	TRUE	yes
12	overcast	mild	high	TRUE	yes
13	overcast	hot	normal	FALSE	yes
14	rainy	mild	high	TRUE	no

Problem: in which order should be analyzed the attributes? which test condition should be assigned to a node?

A simple example

In which order should be analyzed the attributes?

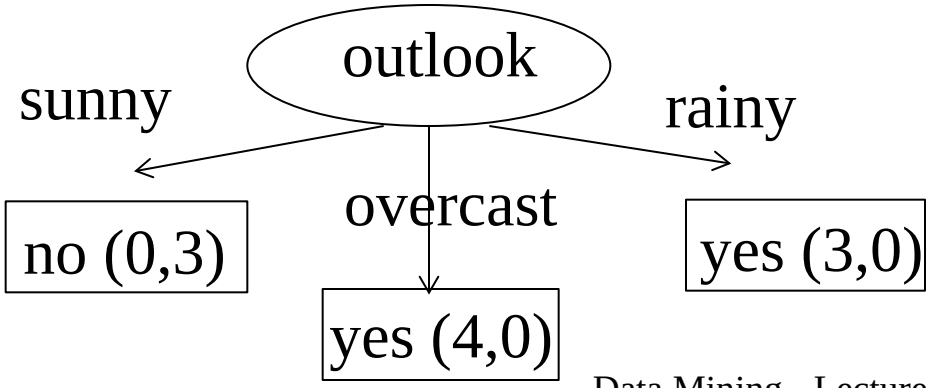
Main idea:

- Select the attribute which leads to a simple tree, i.e. an attribute with a high purity level (ideally, for each possible value of the attribute the corresponding data instances belong to the same class)

Weather/play dataset (selected instances)

No.	outlook Nominal	temperature Nominal	humidity Nominal	windy Nominal	**play** Nominal
1	sunny	hot	high	FALSE	no
2	sunny	hot	high	TRUE	no
3	overcast	hot	high	FALSE	yes
4	rainy	mild	high	FALSE	yes
5	rainy	cool	normal	FALSE	yes
6	overcast	cool	normal	TRUE	yes
7	sunny	mild	high	FALSE	no
8	rainy	mild	normal	FALSE	yes
9	overcast	mild	high	TRUE	yes
10	overcast	hot	normal	FALSE	yes

Example:



Remark:

- All leaves are “pure” (contain data belonging to the same class)
- Such a flat tree leads to classification rules involving only one attribute in the left-hand side
- This situation happens rarely for real-world data

Decision trees

The main decisions to be taken during the decision tree induction

- Which are the test conditions to be assigned to the branches corresponding to a node?
  - It depends on the attributes type
    - Nominal, ordinal, continuous
  - It depends on the desired degree of the splitting node:
    - 2-way split (the current subset is divided in 2 subsets)
    - Multi-way split (the current subset is divided in several subsets)

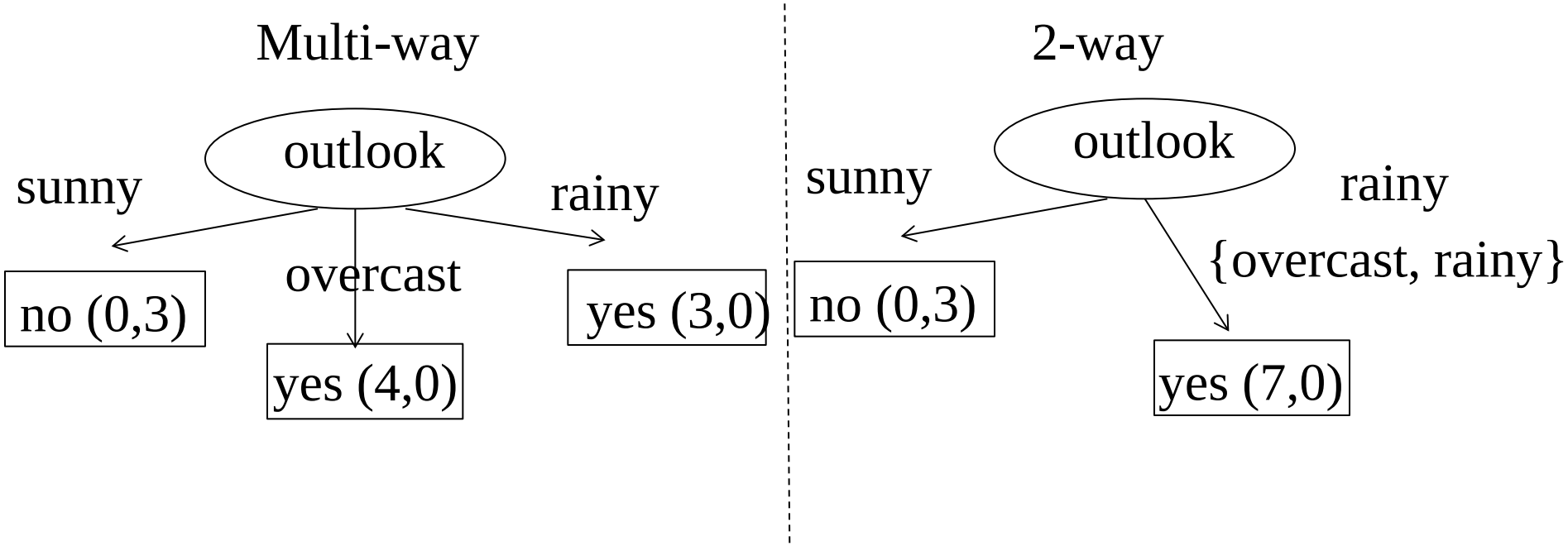
- Which attribute should be selected for splitting?
  - The most discriminative one – that which ensures a partition of the current dataset in subsets with a high degree of purity
  - There are several criteria which can be used:
    - Entropy (variants: information gain, gain ratio)
    - Gini index
    - Misclassification

Decision trees

- Which are the test conditions to be assigned to the branches corresponding to a node?

Nominal and ordinal attributes:

- Multi-way: as many branches as possible values
- 2-way: two branches



Decision trees

Which are the test conditions to be assigned to the branches corresponding to a node?

Numerical attributes:

- The numerical attributes are previously discretized and then is applied the approach which is specific to attributes with discrete values (using the mult-way or 2-way approaches)

Which attribute should be selected for splitting?

- That which leads to the maximal reduction in the information needed to take the right decision
  - Information gain = Entropy(distribution of data before splitting) – AveragedEntropy(distribution of data after splitting)

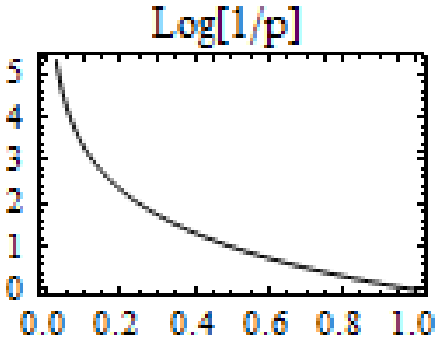
Reminder: entropy

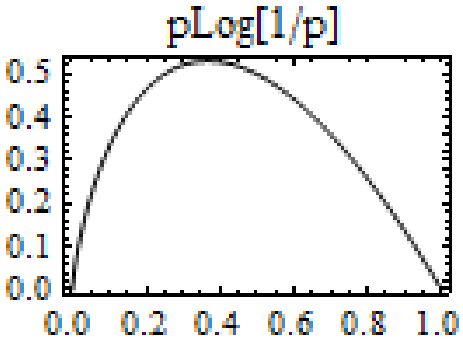
Let D=(p_1,p_2, …, p_k) be a distribution probability. The entropy associated to this distribution probability is defined as

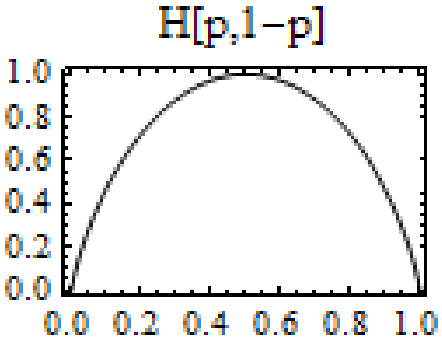
$$H(D) = H(p_1, p_2, ..., p_k) = -\sum_{i=1}^{k} p_i \log p_i$$

And can be interpreted as a measure of the amount of uncertainty (or surprise) when we sample data according to the distribution probability

Particular case: k=2 => p_1=p, p_2=1-p







Remark:

Log[1/p] interpretation: the surprise of seeing an event characterized by a small probability (unexpected event) is higher than in the case of an event of high probability (which is expected)

Reminder: entropy

In the context of a classification problem:

- $D=\{C_1,C_2,\ldots,C_k\}$ (dataset of instances belonging to k classes)
- Distribution probability $(p_1,p_2, \ldots, p_k)$, $p_i=card(C_i)/card(D)$
- Let A be an attribute and $v_1,v_2,\ldots,v_{mA}$ the set of values taken by this attribute
- Let D_j=set of instances from D for which attribute A has the value v_j and P_j the distribution of data of D_j in the k classes (C_{ji}=number of instances having the value v_j for attribute A which belong to class C_i)
- Information Gain obtained by splitting the dataset according to attribute A

$$IG(D,A) = H(D) - \sum_{j=1}^{m_A} P(D_j \mid A = v_j) H(D_j \mid A = v_j), \quad H(D) = -\sum_{i=1}^{k} p_i \log p_i$$

$$H(D_j \mid A = v_j) = -\sum_{i=1}^{k} p_{ij} \log p_{ij}, \quad p_{ij} = \frac{card(C_{ji})}{card(C_i)}$$

$$P(D_j \mid A = v_j) = \frac{card(D_j)}{card(D)}$$

Choosing the splitting attribute

Example

Relation: weather.symbolic

No.	outlook Nominal	temperature Nominal	humidity Nominal	windy Nominal	**play** Nominal
1	sunny	hot	high	FALSE	no
2	sunny	hot	high	TRUE	no
3	overcast	hot	high	FALSE	yes
4	rainy	mild	high	FALSE	yes
5	rainy	cool	normal	FALSE	yes
6	rainy	cool	normal	TRUE	no
7	overcast	cool	normal	TRUE	yes
8	sunny	mild	high	FALSE	no
9	sunny	cool	normal	FALSE	yes
10	rainy	mild	normal	FALSE	yes
11	sunny	mild	normal	TRUE	yes
12	overcast	mild	high	TRUE	yes
13	overcast	hot	normal	FALSE	yes
14	rainy	mild	high	TRUE	no

- Class distribution (C_1="yes", C_2="no"):
- p_1=9/14, p_2=5/14
- $H(p_1,p_2)$=0.94

Outlook

	C1 (yes)	C2(no)	Frequency
Sunny	2/5	3/5	5/14
Overcast	4/4	0/4	4/14
Rainy	3/5	2/5	5/14

H(sunny)=-2/5*log(2/5)-3/5*log(3/5)=0.97
H(overcast)=-1*log(1)-0=0
H(rainy)=-3/5*log(3/5)-2/5*log(2/5)=0.97

IG(outlook)=0.94-5/14*0.97-4/14*0-5/14*0.97=0.94-0.69=0.25

Choosing the splitting attribute

Example

Relation: weather.symbolic

No.	outlook Nominal	temperature Nominal	humidity Nominal	windy Nominal	play Nominal
1	sunny	hot	high	FALSE	no
2	sunny	hot	high	TRUE	no
3	overcast	hot	high	FALSE	yes
4	rainy	mild	high	FALSE	yes
5	rainy	cool	normal	FALSE	yes
6	rainy	cool	normal	TRUE	no
7	overcast	cool	normal	TRUE	yes
8	sunny	mild	high	FALSE	no
9	sunny	cool	normal	FALSE	yes
10	rainy	mild	normal	FALSE	yes
11	sunny	mild	normal	TRUE	yes
12	overcast	mild	high	TRUE	yes
13	overcast	hot	normal	FALSE	yes
14	rainy	mild	high	TRUE	no

- Class distribution (C_1="yes", C_2="no"):
- p_1=9/14, p_2=5/14
- $H(p_1,p_2)$=0.94

Temperature

	C1 (yes)	C2(no)	Frequency
Hot	2/4	2/4	4/14
Mild	4/6	2/6	6/14
Cool	3/4	1/4	4/14

H(hot)=-2/4*log(2/4)-2/4*log(2/4)
H(mild)=-4/6*log(4/6)-2/6*log(2/6)
H(cool)=-3/4*log(3/4)-1/4*log(1/4)

IG(temperature)=0.03

Choosing the splitting attribute

Example

Relation: weather.symbolic

No.	outlook Nominal	temperature Nominal	humidity Nominal	windy Nominal	**play** Nominal
1	sunny	hot	high	FALSE	no
2	sunny	hot	high	TRUE	no
3	overcast	hot	high	FALSE	yes
4	rainy	mild	high	FALSE	yes
5	rainy	cool	normal	FALSE	yes
6	rainy	cool	normal	TRUE	no
7	overcast	cool	normal	TRUE	yes
8	sunny	mild	high	FALSE	no
9	sunny	cool	normal	FALSE	yes
10	rainy	mild	normal	FALSE	yes
11	sunny	mild	normal	TRUE	yes
12	overcast	mild	high	TRUE	yes
13	overcast	hot	normal	FALSE	yes
14	rainy	mild	high	TRUE	no

- Class distribution (C_1="yes", C_2="no"):
- p_1=9/14, p_2=5/14
- $H(p_1,p_2)$=0.94

Humidity

	C1 (yes)	C2(no)	Frequency
High	3/7	4/7	7/14
Normal	6/7	1/7	7/14

H(high)=-3/7*log(3/7)-4/7*log(4/7)
H(normal)=-6/7*log(6/7)-1/7*log(1/7)

IG(humidity)=0.15

Choosing the splitting attribute

Example

Relation: weather.symbolic

No.	outlook Nominal	temperature Nominal	humidity Nominal	windy Nominal	**play** Nominal
1	sunny	hot	high	FALSE	no
2	sunny	hot	high	TRUE	no
3	overcast	hot	high	FALSE	yes
4	rainy	mild	high	FALSE	yes
5	rainy	cool	normal	FALSE	yes
6	rainy	cool	normal	TRUE	no
7	overcast	cool	normal	TRUE	yes
8	sunny	mild	high	FALSE	no
9	sunny	cool	normal	FALSE	yes
10	rainy	mild	normal	FALSE	yes
11	sunny	mild	normal	TRUE	yes
12	overcast	mild	high	TRUE	yes
13	overcast	hot	normal	FALSE	yes
14	rainy	mild	high	TRUE	no

H(false)=-6/8*log(6/8)-2/8*log(2/8)
H(true)=-3/6*log(3/6)-3/6*log(3/6)

IG(windy)=0.05

- Class distribution (C_1="yes", C_2="no"):
- p_1=9/14, p_2=5/14
- $H(p_1,p_2)$=0.94

Windy

	C1 (yes)	C2(no)	Frequency
False	6/8	2/8	8/14
True	3/6	3/6	6/14

Choosing the splitting attribute

Example

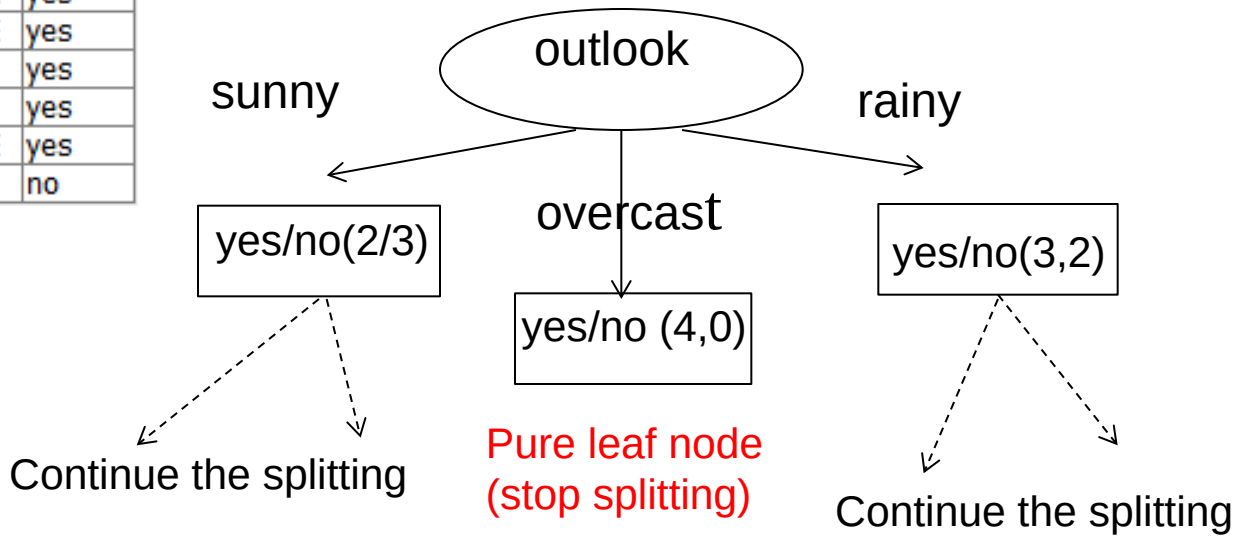
Relation: weather.symbolic

No.	outlook Nominal	temperature Nominal	humidity Nominal	windy Nominal	**play** Nominal
1	sunny	hot	high	FALSE	no
2	sunny	hot	high	TRUE	no
3	overcast	hot	high	FALSE	yes
4	rainy	mild	high	FALSE	yes
5	rainy	cool	normal	FALSE	yes
6	rainy	cool	normal	TRUE	no
7	overcast	cool	normal	TRUE	yes
8	sunny	mild	high	FALSE	no
9	sunny	cool	normal	FALSE	yes
10	rainy	mild	normal	FALSE	yes
11	sunny	mild	normal	TRUE	yes
12	overcast	mild	high	TRUE	yes
13	overcast	hot	normal	FALSE	yes
14	rainy	mild	high	TRUE	no

Information gain for each attribute:

- IG(outlook)=0.25
- IG(temperature)=0.03
- IG(humidity)=0.15
- IG(windy)=0.05

First splitting attribute: outlook

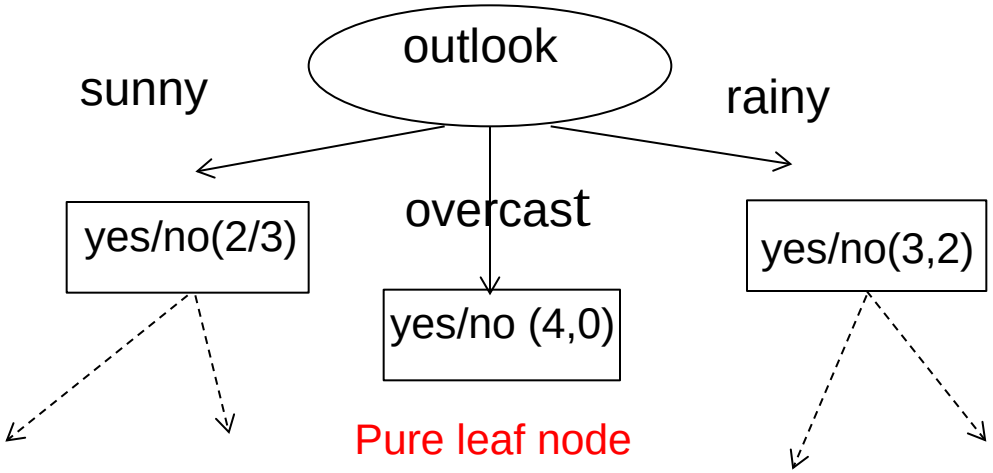


Choosing the splitting attribute

Example

Relation: weather.symbolic

No.	outlook Nominal	temperature Nominal	humidity Nominal	windy Nominal	play Nominal
1	sunny	hot	high	FALSE	no
2	sunny	hot	high	TRUE	no
3	overcast	hot	high	FALSE	yes
4	rainy	mild	high	FALSE	yes
5	rainy	cool	normal	FALSE	yes
6	rainy	cool	normal	TRUE	no
7	overcast	cool	normal	TRUE	yes
8	sunny	mild	high	FALSE	no
9	sunny	cool	normal	FALSE	yes
10	rainy	mild	normal	FALSE	yes
11	sunny	mild	normal	TRUE	yes
12	overcast	mild	high	TRUE	yes
13	overcast	hot	normal	FALSE	yes
14	rainy	mild	high	TRUE	no



Pure leaf node (stop splitting)

Information gain for each remaining attribute:

- Entropy for class distribution on “sunny” subset: H(D(sunny))=-2/5*log(2/5)-3/5*log(3/5)=0.97
- H(hot)=0, H(mild)=1, H(cool)=0
- IG(temperature)=0.97-2/5=0.57

Temperature

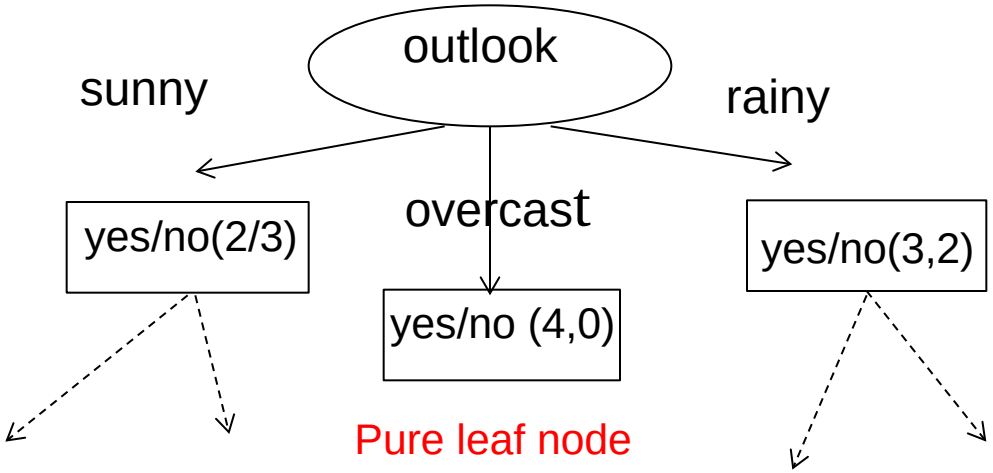
	C1 (yes)	C2(no)	Freq.
Hot	0/2	2/2	2/5
Mild	1/2	1/2	2/5
Cool	1/1	0/1	1/5

Choosing the splitting attribute

Example

Relation: weather.symbolic

No.	outlook Nominal	temperature Nominal	humidity Nominal	windy Nominal	play Nominal
1	sunny	hot	high	FALSE	no
2	sunny	hot	high	TRUE	no
3	overcast	hot	high	FALSE	yes
4	rainy	mild	high	FALSE	yes
5	rainy	cool	normal	FALSE	yes
6	rainy	cool	normal	TRUE	no
7	overcast	cool	normal	TRUE	yes
8	sunny	mild	high	FALSE	no
9	sunny	cool	normal	FALSE	yes
10	rainy	mild	normal	FALSE	yes
11	sunny	mild	normal	TRUE	yes
12	overcast	mild	high	TRUE	yes
13	overcast	hot	normal	FALSE	yes
14	rainy	mild	high	TRUE	no



Pure leaf node (stop splitting)

Information gain for each remaining attribute:

- Entropy for class distribution on “sunny” subset: H(D(sunny))=-2/5*log(2/5)-3/5*log(3/5)=0.97
- H(high)=0, H(normal)=0
- IG(humidity)=0.97-0=0.97

Humidity

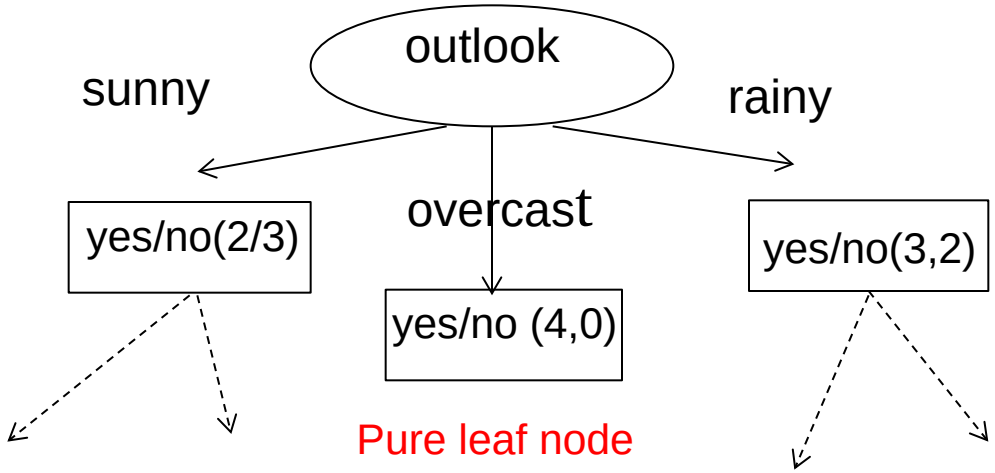
	C1 (yes)	C2(no)	Freq.
High	0/3	3/3	3/5
Normal	2/2	0/2	2/5

Choosing the splitting attribute

Example

Relation: weather.symbolic

No.	outlook Nominal	temperature Nominal	humidity Nominal	windy Nominal	play Nominal
1	sunny	hot	high	FALSE	no
2	sunny	hot	high	TRUE	no
3	overcast	hot	high	FALSE	yes
4	rainy	mild	high	FALSE	yes
5	rainy	cool	normal	FALSE	yes
6	rainy	cool	normal	TRUE	no
7	overcast	cool	normal	TRUE	yes
8	sunny	mild	high	FALSE	no
9	sunny	cool	normal	FALSE	yes
10	rainy	mild	normal	FALSE	yes
11	sunny	mild	normal	TRUE	yes
12	overcast	mild	high	TRUE	yes
13	overcast	hot	normal	FALSE	yes
14	rainy	mild	high	TRUE	no



Pure leaf node (stop splitting)

Information gain for each remaining attribute:

- Entropy for class distribution on “sunny” subset: H(D(sunny))=-2/5*log(2/5)-3/5*log(3/5)=0.97
- H(false)=0, H(true)=1
- IG(windy)=0.97-0.95=0.02

Windy

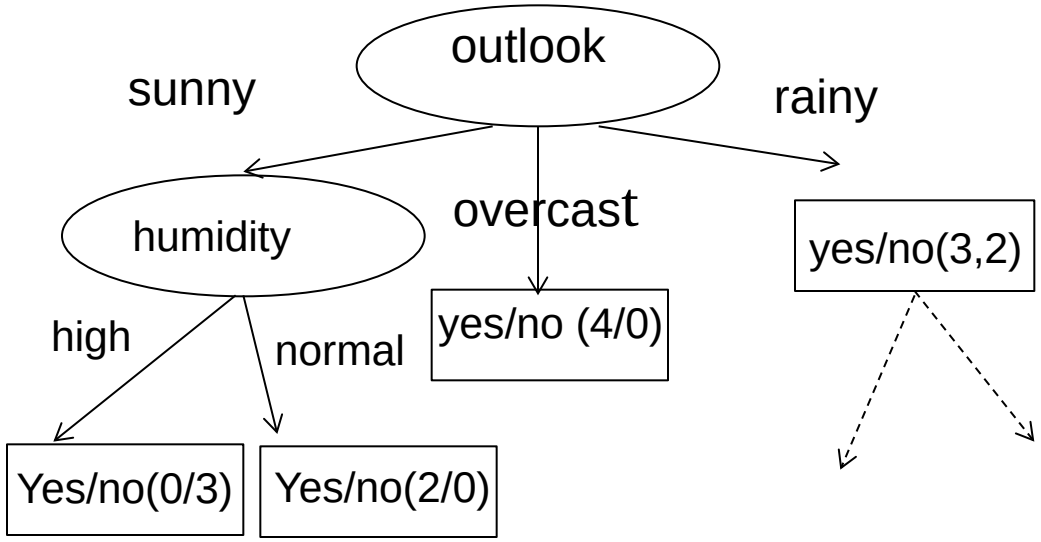
	C1 (yes)	C2(no)	Freq.
false	1/3	2/3	3/5
true	1/2	1/2	2/5

Choosing the splitting attribute

Example

Relation: weather.symbolic

No.	outlook Nominal	temperature Nominal	humidity Nominal	windy Nominal	**play** Nominal
1	sunny	hot	high	FALSE	no
2	sunny	hot	high	TRUE	no
3	overcast	hot	high	FALSE	yes
4	rainy	mild	high	FALSE	yes
5	rainy	cool	normal	FALSE	yes
6	rainy	cool	normal	TRUE	no
7	overcast	cool	normal	TRUE	yes
8	sunny	mild	high	FALSE	no
9	sunny	cool	normal	FALSE	yes
10	rainy	mild	normal	FALSE	yes
11	sunny	mild	normal	TRUE	yes
12	overcast	mild	high	TRUE	yes
13	overcast	hot	normal	FALSE	yes
14	rainy	mild	high	TRUE	no



Pure leaf nodes
(stop splitting)

Information gain for each remaining attribute:

- IG(temperature)=0.97-2/5=0.57
- IG(humidity)=0.97-0=0.97
- IG(windy)=0.97-0.95=0.02

Choosing the splitting attribute

Remarks:

- Information Gain favors the attributes characterized by a larger number of values
- In order to avoid this bias it can be used the Gain Ratio:

$$GainRatio(D, A) = \frac{IG(D, A)}{H(p_1^A, p_2^A, ..., p_{m_A}^A)}$$

$$p_j^A = \frac{card(D, A = v_j)}{card(D)} \text{ (ratio of data having value } v_j \text{ for attr. } A)$$

- The splitting attribute can be selected by using the Gini impurity measure = how often a randomly chosen element from the set would be incorrectly labeled if it were randomly labeled according to the distribution of labels in the subset corresponding to a branch (smaller values are better)

$$Gini(p_1, p_2, ..., p_n) = 1 - \sum_{i=1}^{n} p_i^2$$

Algorithms for Decision Tree Induction

ID3:

- Input: dataset D
- Output: decision tree (input nodes labelled with attributes, leaf nodes labelled with classes, edges labelled with attribute values)

DTinduction (D, DT, N) /* D=dataset, DT=decision tree, N=node */
    find the best splitting attribute A
    label node N with A
    construct the splitting predicates (branches) for N
    FOR each branch i from N DO
        construct the corresponding data set D_i
        create a new childe node N_i
        IF <stopping condition>
            THEN label N_i with the dominant class in D_i (N_i is a leaf node)
            ELSE DTinduction(D_i,DT, N_i)

Algorithms for Decision Tree Induction

C4.5 = improvement of ID3 with respect to

- Continuous attributes:
  - incorporates a discretization procedure for continuous attributes
- Missing values:
  - During the induction process the instances with missing values are ignored
  - During the classification the missing values of the instance to be classified are imputed
- Splitting attribute:
  - It uses the Gain Ratio as attribute selection (in order to be more robust with respect to the number of values)
- Pruning:
  - Some subtrees are replaced with leaf nodes (if the classification error is not increased significantly) – bottom-up approach

Remark:

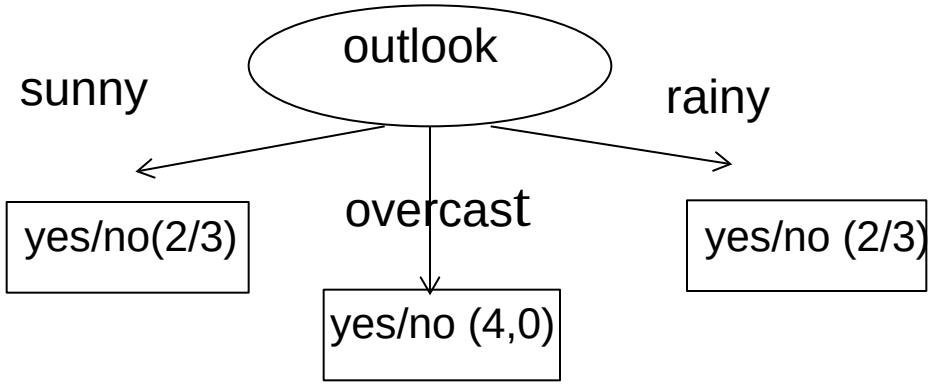
C5.0 – commercial variant of C4.5

J48 – Weka implementation of C4.5

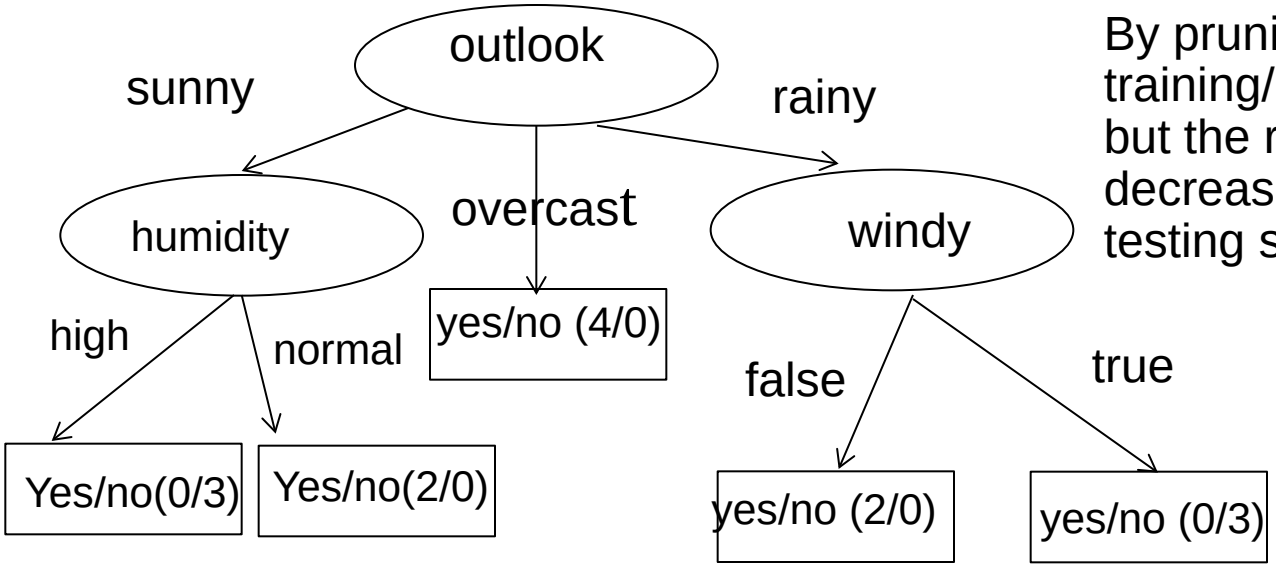
Algorithms for Decision Tree Induction

Pruning:

- Some subtrees are replaced with leaf nodes (if the classification error is not increased significantly) – bottom-up approach



Unpruned tree: error = 0



Pruned tree: error = 4/14

By pruning the error on the training/ validation set is increased but the risk of overfitting could be decreased (error of an unseen testing set could be smaller)

Classification rules induction

Reminder: classification rules are IF … THEN statements containing:

- In the antecedent part (left hand side): conditions on the attribute values (it could be a condition concerning only one attribute or logical expressions involving several attributes)
- In the consequent part (right hand side): a class label

Example:
IF outlook=sunny THEN play=no
IF outlook=rainy THEN play=no
IF outlook=overcast THEN play=yes

Remark:
- These rules are extracted from a decision tree – each branch of the tree leads to a rule
- The conditions related to nodes on the same branch should be combined by AND : IF (outlook=sunny) and (humidity=high) THEN play=no
- Rules corresponding to different branches but leading to the same consequent part (same class label) can by merged by using OR between the antecedent parts:

IF (outlook=sunny) OR (outlook=rainy) THEN play=no

Classification rules induction

The classification rules can be extracted directly from the data during a learning process by using covering algorithms

Notions:

- A rule covers a data instance if the values of the attributes' values match the antecedent part of the rule
- Similarly, a data instance triggers a rule if the values of the attributes' values match the antecedent part of the rule
- Ruleset = set of rules
- Support of a rule = fraction of the dataset which is covered by the rule and belong to the same class = |cover(R) ∩class(R) |/|D|
- Confidence of a rule = fraction of data instances covered by a rule which have the same class as the rule = |cover(R) ∩class(R)|/|cover(R)|

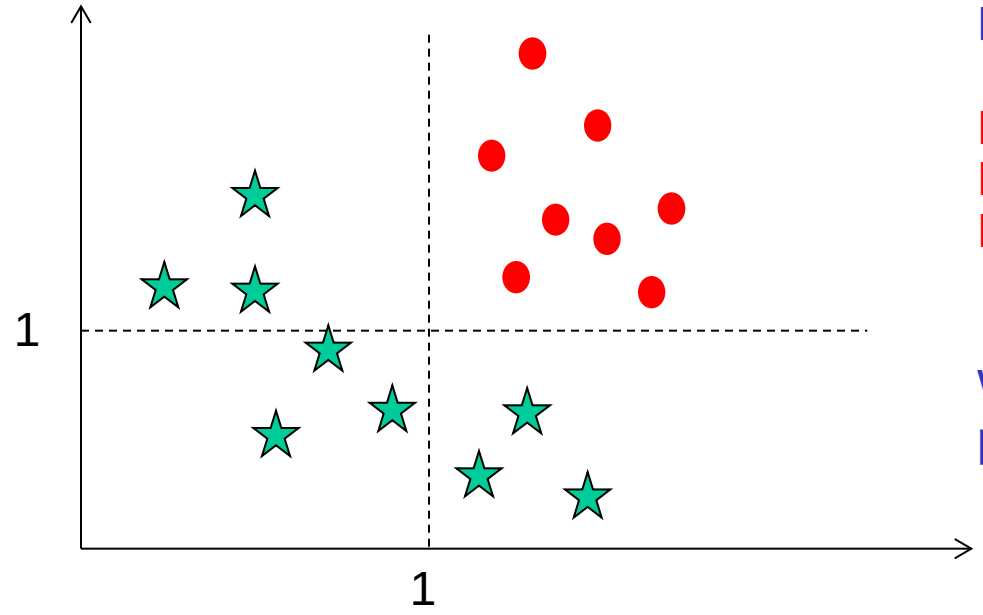
cover(R) = set of the instances covered by R
class(R) = set of instances having the same class as R
D= dataset

Classification rules induction

Notions:

- Mutually exclusive rules = the regions covered by rules are disjoint (an instance triggers only one rule)
- Exhaustive ruleset = each instance triggers at least one rule

Remark: if a ruleset is both exhaustive and the rules are mutually exclusive then it is easy to take a decision for a given instance



Example:

R1: IF x>1 and y>1 THEN C0
R2: IF x<=1 THEN C*
R3: IF x>1 and y<=1 THEN C*

What about the case when these properties are not satisfied?

Classification rules induction

Remark: if the rules are not mutually exclusive then may appear conflicts (one data instance trigger several rules which have different classes)

The conflicts can be solved in one of the following ways:

- The rules are ordered (based on a quality measure) and the decision is taken according to the first rule triggered by the data instance (rule which matches to the instance. The sorting criteria can be related to:
  - the rule quality (e.g. high confidence) – higher confidence is better
  - the rule specificity – the rules are considered better if they are more specific (e.g. those corresponding to rare classes)
  - the rule complexity (e.g. number of conditions in the antecedent part) – simpler rules are better

- The result is the dominant class from the set of rules triggered by the data instance

Classification rules induction

Sequential covering algorithm:

Input: data set
Output: ordered set of rules

Step 1: Select a class label and determine the “best” rule which cover the data instances from D having the selected class label. Add this rule to the bottom of the ordered rule list

Step 2: Remove all data from D which match to the antecedent of the added rule. If there are still class labels to select and data in D go to Step 1

Remark:

- This is the general structure of sequential covering algorithms
- Particular algorithms differ with respect to the ordering strategy

Classification rules induction

Example: RIPPER

Particularities:

- Class-based ordering: the classes are selected in order of their size (the rare classes are selected first)
- The rules corresponding to one class are placed contiguously in the ordered list of rules
- The addition of a new rule corresponding to one class is stopped:
  - when the rule becomes too complex
  - when the new rule has a classification error (on a validation set) which is larger than a predefined threshold
- If at the end remain some uncovered data then is defined a “catch all” rule to which is assigned the dominant class

Probabilistic Classification

Idea: construct a model which capture the relationship between the probability of a data instance to belong to a given class

Aim: estimate $P(C_k|D_i)$ = probability that the class of data instance D_i is C_k
Remark: P(B|A) is a conditional probability = probability of event B given that (by assumption, presumption, assertion or evidence) event A has occurred

Reminder on probability theory:

$$\text{Conditional probability}: \; P(B \mid A) = \frac{P(A,B)}{P(A)}$$

$$P(A,B) = \text{probability that both events A and B occured}$$

$$\text{Bayes rule}: \; P(B \mid A) = \frac{P(A \mid B)P(B)}{P(A)}$$

Bayes rule is useful to estimate the posterior probability P(B|A) when the prior probability P(B) and the other probabilities P(A|B), P(A) can be estimated easier.

Probabilistic Classification

Example: Let us suppose that we are interested to estimate the probability that a patient having some symptom S has the illness T

- We want to estimate P(T|S)
- Let us suppose that we know:
  - P(S) = 1 – the symptom exists (they correspond to an event which is sure
  - P(T) – estimated based on population studies (how frequent is the illness)
  - P(S|T) – estimated based on prior medical knowledge (how often is the symptom present in the case of illness T)
- P(T|S)=P(S|T)P(T)/P(S)=P(S|T)P(T)
- What about the case when there is not only one symptom S, but a list of symptoms $S_1,S_2,\ldots,S_n$?

Probabilistic Classification

Example: Let us suppose that we are interested to estimate the probability that a patient having some symptom S has the illness T

- What about the case when there is not only one symptom S, but a list of symptoms $S_1,S_2,\ldots,S_n$?
- In this case one have to estimate $P(T| S_1,S_2,\ldots,S_n)$
- Based on the Bayes rule:
  - $P(T| S_1,S_2,\ldots,S_n)=P(S_1,S_2,\ldots,S_n|T)P(T)/P(S_1,S_2,\ldots,S_n)$
- Simplifying assumption: the symptoms $(S_1,S_2,\ldots,S_n)$ are independent events (this is not always true but many practical situations this assumption can be accepted)
- Considering that $P(S_1,S_2,\ldots,S_n)=1$ (the symptoms are all real)

$$P(T| S_1,S_2,\ldots,S_n)=P(S_1|T)\,P(S_2|T)\ldots P(S_n|T)P(T)$$

Naïve Bayes Classifier

Classification problem:

- For a data instance $D_i=(a_{i1},a_{i2},\ldots,a_{in})$ find the class to which it belongs

Main idea

- Estimate $P(C_k| D_i)=P(C_k|a_{i1},a_{i2},\ldots,a_{in})\ P(C_k)$ for all k in {1,2,…,K} and select the maximal probability; it will indicate the class to which the data instance most probably belongs
- Simplifying assumption: the attributes are independent (this is why the method is called “naive”)
- $P(C_k| D_i)= P(a_{i1}|C_k)\ P(a_{i2}|C_k)\ldots P(a_{in}|C_k)P(C_k)$
- This requires the knowledge of $P(a_{i1}|C_k)$, $P(a_{i2}|C_k)$, …, $P(a_{in}|C_k)$ and $P(C_k)$
- These probabilities can be estimated from the dataset (as relative frequencies) – this is the learning process corresponding to the Naïve Bayes

Naïve Bayes Classifier

Example:

Relation: weather.symbolic

No.	outlook Nominal	temperature Nominal	humidity Nominal	windy Nominal	**play** Nominal
1	sunny	hot	high	FALSE	no
2	sunny	hot	high	TRUE	no
3	overcast	hot	high	FALSE	yes
4	rainy	mild	high	FALSE	yes
5	rainy	cool	normal	FALSE	yes
6	rainy	cool	normal	TRUE	no
7	overcast	cool	normal	TRUE	yes
8	sunny	mild	high	FALSE	no
9	sunny	cool	normal	FALSE	yes
10	rainy	mild	normal	FALSE	yes
11	sunny	mild	normal	TRUE	yes
12	overcast	mild	high	TRUE	yes
13	overcast	hot	normal	FALSE	yes
14	rainy	mild	high	TRUE	no

P(C1)=P(no)=5/14 P(C2)= P(yes)=9/14

A1: outlook

P(sunny|C1)=P(sunny,C1)/P(C1)
=(3/14)/(5/14)=3/5

P(sunny|C2)=P(sunny,C2)/P(C2)
=(2/14)/(9/14)=2/9

P(overcast|C1)=P(overcast,C1)/P(C1)
=0

P(overcast|C2)=P(overcast,C2)/P(C2)
=(4/14)/(9/14)=4/9

P(rainy|C1)=P(rainy,C1)/P(C1)
=(2/14)/(5/14)=2/5

P(rainy|C2)=P(rainy,C2)/P(C2)
=(3/14)/(9/14)=3/9

Naïve Bayes Classifier

Example:

Relation: weather.symbolic

No.	outlook Nominal	temperature Nominal	humidity Nominal	windy Nominal	**play** Nominal
1	sunny	hot	high	FALSE	no
2	sunny	hot	high	TRUE	no
3	overcast	hot	high	FALSE	yes
4	rainy	mild	high	FALSE	yes
5	rainy	cool	normal	FALSE	yes
6	rainy	cool	normal	TRUE	no
7	overcast	cool	normal	TRUE	yes
8	sunny	mild	high	FALSE	no
9	sunny	cool	normal	FALSE	yes
10	rainy	mild	normal	FALSE	yes
11	sunny	mild	normal	TRUE	yes
12	overcast	mild	high	TRUE	yes
13	overcast	hot	normal	FALSE	yes
14	rainy	mild	high	TRUE	no

Same computations should be done for A2 (temperature), A3 (humidity) and A4 (windy)

Remark: if for a given attribute value a_{ij} and a given class C_k there is no example in the training set, then $P(a_{ij}| C_k)=0$ and (because of the independency assumption) for any instance having the value a_{ij} for attribute A_i the probability to belong to C_k is 0.

This situation might appear especially in the case of small classes

Laplace smoothing:

$P(a_{ij}| C_k)=(count(a_{ij},C_k)+alpha)/(count(C_k)+m_i*alpha)$
alpha= Laplace smoothing parameter
m_i= number of distinct values of attribute A_i

Naïve Bayes Classifier

Remarks:

- This classifier can be directly applied for discrete attributes and it is based on the following probabilistic models:
  - Binomial model
  - Multinomial model

- In the case of real attributes there are two main approaches:
  - The attributes are discretized before using the classifier (the classifier performance is depends on the discretization process
  - The attributes are modeled through continuous probabilistic models (e.g. Gaussian) with parameters estimated based on the training data

Next lecture

- Neural Networks
- Support Vector Machines