Data Mining & CMR Classification

Pisa KDD Lab, ISTI-CNR & Univ. Pisa

http://www-kdd.isti.cnr.it/

MAINS – Master in Management dell'Innovazione

Scuola S. Anna

Seminar 3 – Data Mining Technologies

Classification

Outline

- The classification task
- Main classification techniques
  - Bayesian classifiers
  - Decision trees
  - Hints to other methods
- Example on insurance fraud detection, using the tool Knowledge Studio®

The classification task

The classification task

- Input: a training set of tuples, each labelled with one class label
- Output: a model (classifier) which assigns a class label to each tuple based on the other attributes.
- The model can be used to predict the class of new tuples, for which the class label is missing or unknown
- Some natural applications
  - credit approval
  - medical diagnosis
  - treatment effectiveness analysis

Train & test

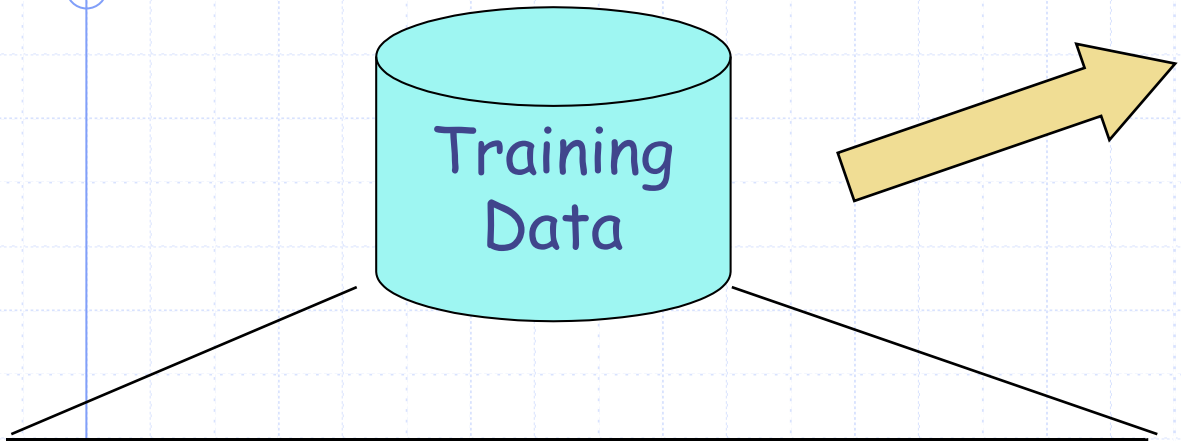
- The tuples (observations, samples) are partitioned in training set + test set.
- Classification is performed in two steps:

1. training - build the model from training set
2. test - check accuracy of the model using test set

Train & test

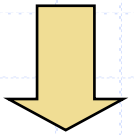
- Kind of models
  - IF-THEN rules
  - Other logical formulae
  - Decision trees
- Accuracy of models
  - The known class of test samples is matched against the class predicted by the model.
  - Accuracy rate = % of test set samples correctly classified by the model.

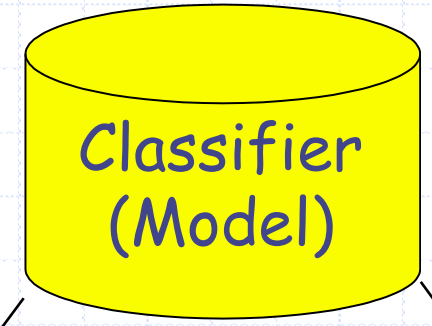
Training step



NAME	AGE	INCOME	CREDIT
Mary	20 - 30	low	poor
James	30 - 40	low	fair
Bill	30 - 40	high	good
John	20 - 30	med	fair
Marc	40 - 50	high	good
Annie	40 - 50	high	good

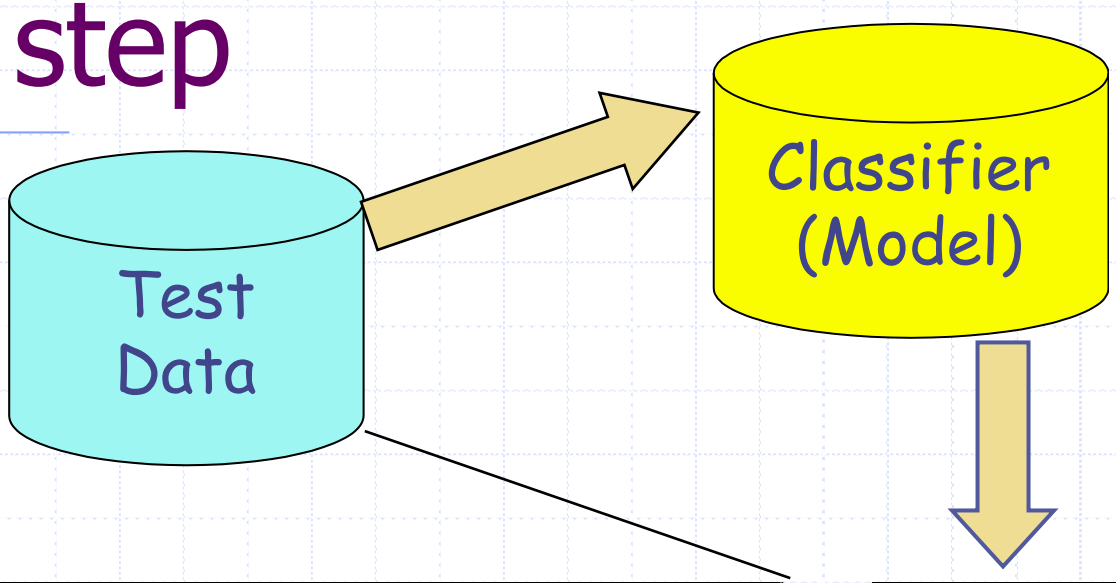
Classification Algorithms





IF age = 30 - 40
OR income = high
THEN credit = good

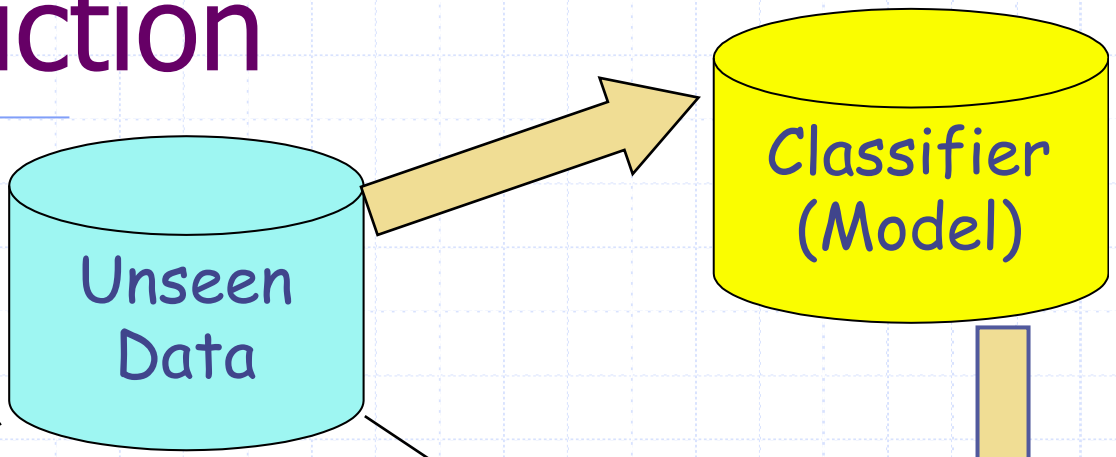
Test step



NAME	AGE	INCOME	CREDIT
Paul	20 - 30	high	good
Jenny	40 - 50	low	fair
Rick	30 - 40	high	fair

CREDIT
fair
fair
good

Prediction



NAME	AGE	INCOME
Doc	20 - 30	high
Phil	30 - 40	low
Kate	40 - 50	med

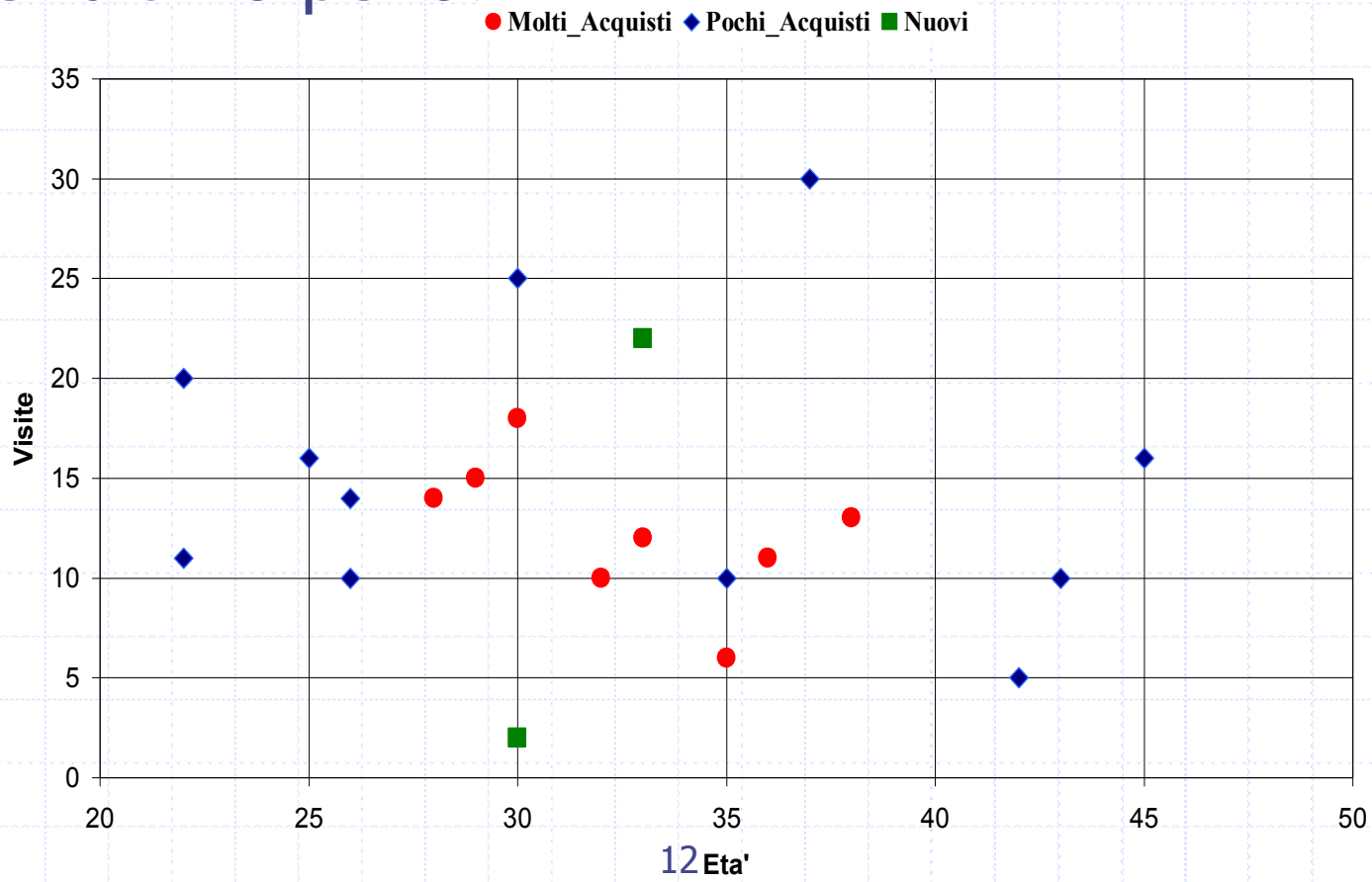
CREDIT
fair
poor
fair

Machine learning terminology

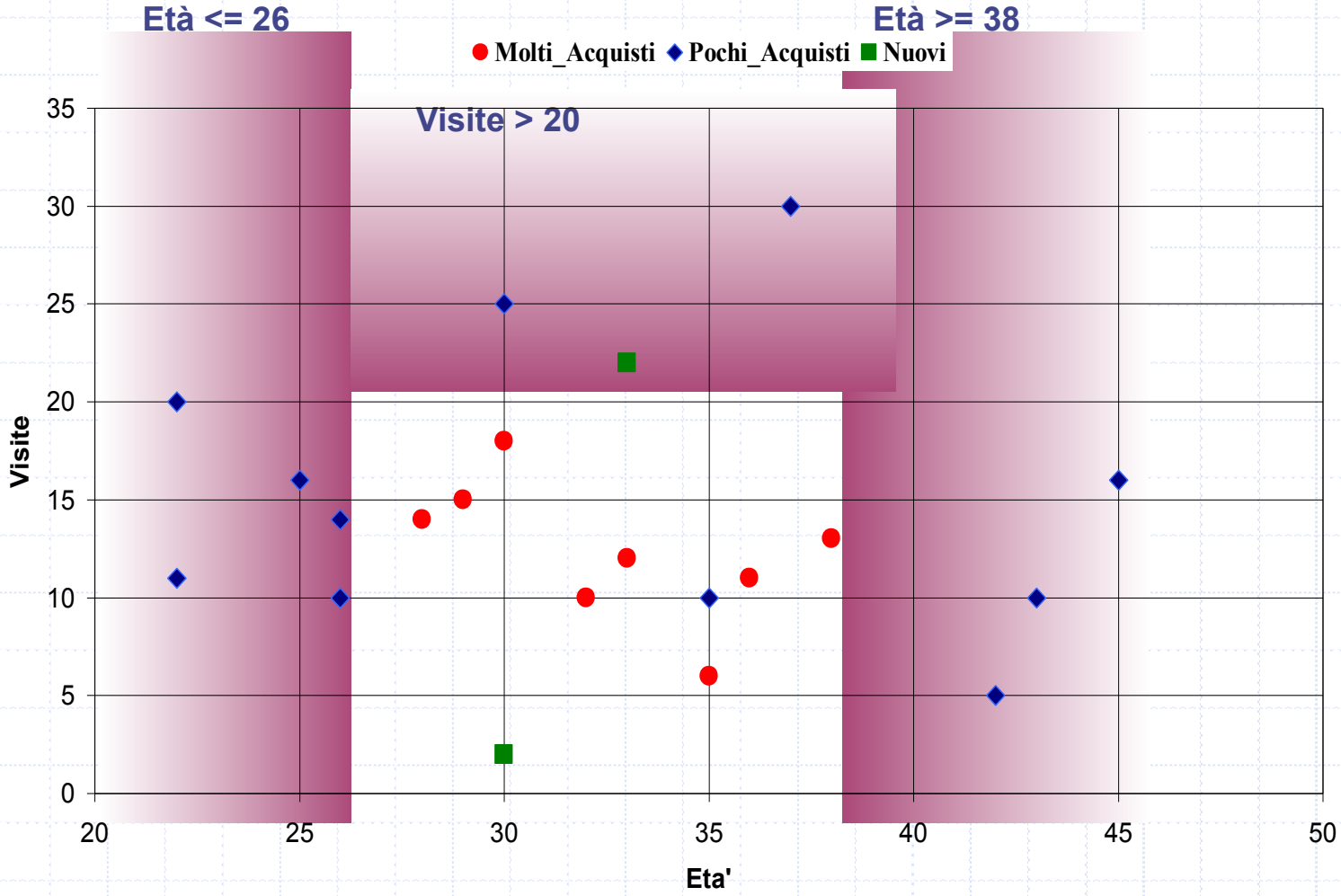
- Classification = supervised learning
  - use training samples with known classes to classify new data
- Clustering = unsupervised learning
  - training samples have no class information
  - guess classes or clusters in the data

Descrivete i clienti migliori ...

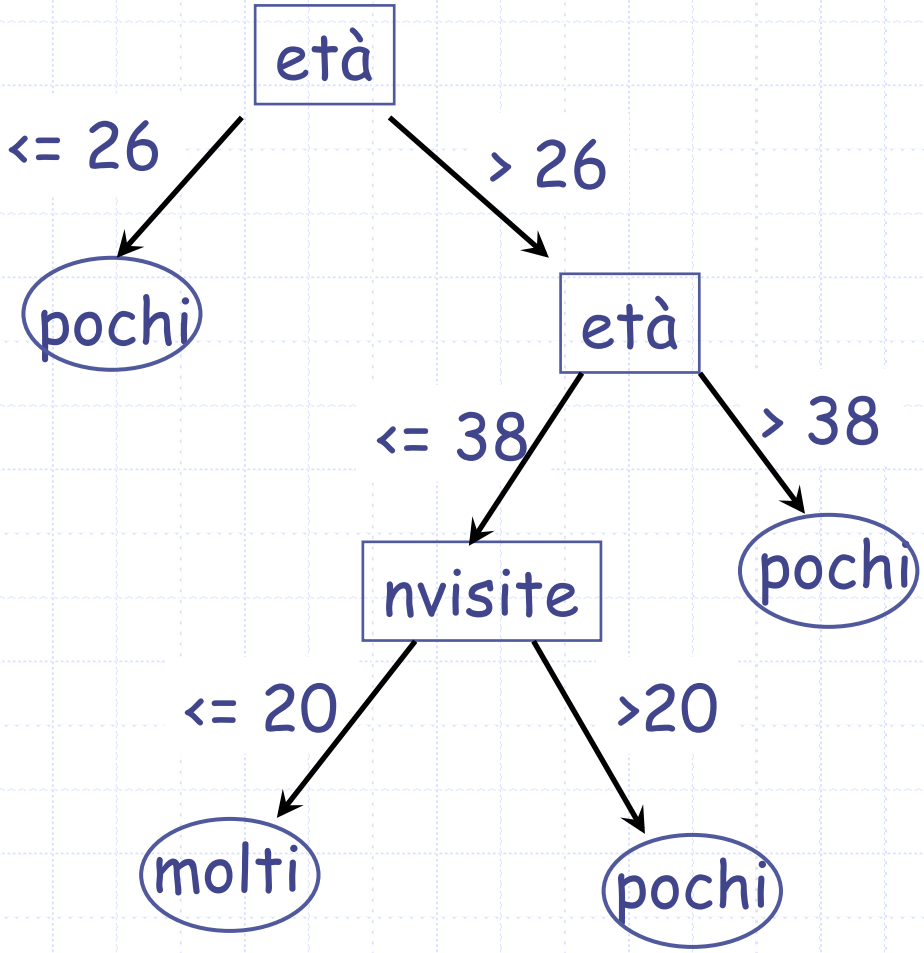
- ... in base ad età ed al numero di visite nell'ultimo periodo!



... una soluzione grafica ...



... e con un albero di decisione!



Il processo seguito

1. Abbiamo individuato una caratteristica "interessante" ma sconosciuta
   - Fare molti acquisti o pochi acquisti
2. Abbiamo individuato altre caratteristiche "conosciute" che riteniamo essere correlate a essa
   - Età, Numero di visite
3. Abbiamo raccolto dei dati storici
   - 19 clienti di cui si conoscono TUTTE le caratteristiche

Il processo seguito

4. Abbiamo indotto dai dati storici una descrizione della caratteristica "interessante" sulla base di quelle "conosciute"
   - estrazione del modello di CLASSIFICAZIONE
5. Abbiamo utilizzato il modello di classificazione per derivare la caratteristica "interessante" di un nuovo cliente a partire da quelle "conosciute"
   - predizione del modello di CLASSIFICAZIONE

Applicazioni della classificazione

- Market analysis & e-commerce
  - customer profiling, caratt. potenziali nuovi clienti, caratt. i clienti più appropriati per una campagna promozionale,
- Risk analysis
  - individuare i clienti fedeli, controllo di qualità dei processi,
- Fraud detection
  - individuare simulatori di incidenti stradali, movimenti bancari sospetti, acquisti con carta di credito sospetti, dichiarazioni dei redditi sospette,
- Web mining
  - categorizzare e-mail, personalizzazione di siti web,
- Altre applicazioni
  - medicina, astronomia, biologia, ecc.

Classificazione e predizione: input

- Un insieme di esempi (**istanze** o **casi**) che descrivono un concetto (**classe**) sulla base di **attributi** (o **features**)
  - La **classe** e gli **attributi** possono assume valori *continui o discreti*

outlook	temperature	humidity	windy	classe
sunny	85	85	false	Don't Play
sunny	80	90	true	Don't Play
overcast	83	78	false	Play
rain	70	96	false	Play
rain	68	80	false	Play
rain	65	70	true	Don't Play
overcast	64	65	true	Play
sunny	72	95	false	Don't Play
sunny	69	70	false	Play
rain	75	80	false	Play
sunny	75	70	true	Play
overcast	72	90	true	Play
overcast	81	75	false	Play
rain	71	80	true	Don't Play

Metadati: tipi di dato

- In aggiunta al tipo di dato per lo storage
  - Discreti
    - Assumono un insieme finito di valori
    - Nominali se identificatori univoci (es., Codice Fiscale)
    - Categorici se ripetibili (es., Città)
    - Binari se due soli valori (es., true – false)
    - Ordinali se definito un ordine totale (es., low < medium < high)
    - Ciclici (es., giorno della settimana)
  - Continui
    - Assumono valori su scala intera o reale
      - Numero acquisti, Importo, Reddito, …
    - L'operazione di divisione in fasce è detta *discretizzazione*

Discretizzazione

- Vantaggi
  - I dati originali possono avere valori continui estremamente sparsi
  - I dati discretizzati possono essere più semplici da interpretare
- Equal-width
  - Divide il range di valori in *k* intervalli della stessa ampiezza
- Equal-frequency
  - Divide il range di valori in k intervalli della stessa frequenza

Equal-width (k=4)

Bar	Beer	Price	PriceDisc
A	Bud	100	100-115
A	Becks	120	115-130
C	Bud	110	100-115
D	Bud	130	130-145
D	Becks	160	145-160
E	Becks	140	130-145
E	Bud	120	115-130
F	Bud	110	100-115
G	Bud	130	130-145
H	Bud	125	115-130
H	Becks	130	130-145
I	Bud	135	130-145

$\delta = (160-100)/4 = 15$

- Fascia 1: [100,115)
- Fascia 2: [115,130)
- Fascia 3: [130,145)
- Fascia 4: [145, 160]

Equal-frequency (k=4)

Bar	Beer	Price	PriceDisc
A	Bud	100	100-110
A	Becks	120	111-125
C	Bud	110	100-110
D	Bud	130	126-130
D	Becks	160	131-160
E	Becks	140	131-160
E	Bud	120	111-125
F	Bud	110	100-110
G	Bud	130	126-130
H	Bud	125	111-125
H	Becks	130	126-130
I	Bud	135	131-160

$\phi = 12/4 = 3$

- Fascia 1: [100,110]
- Fascia 2: [111,125]
- Fascia 3: [126,130]
- Fascia 4: [131, 160]

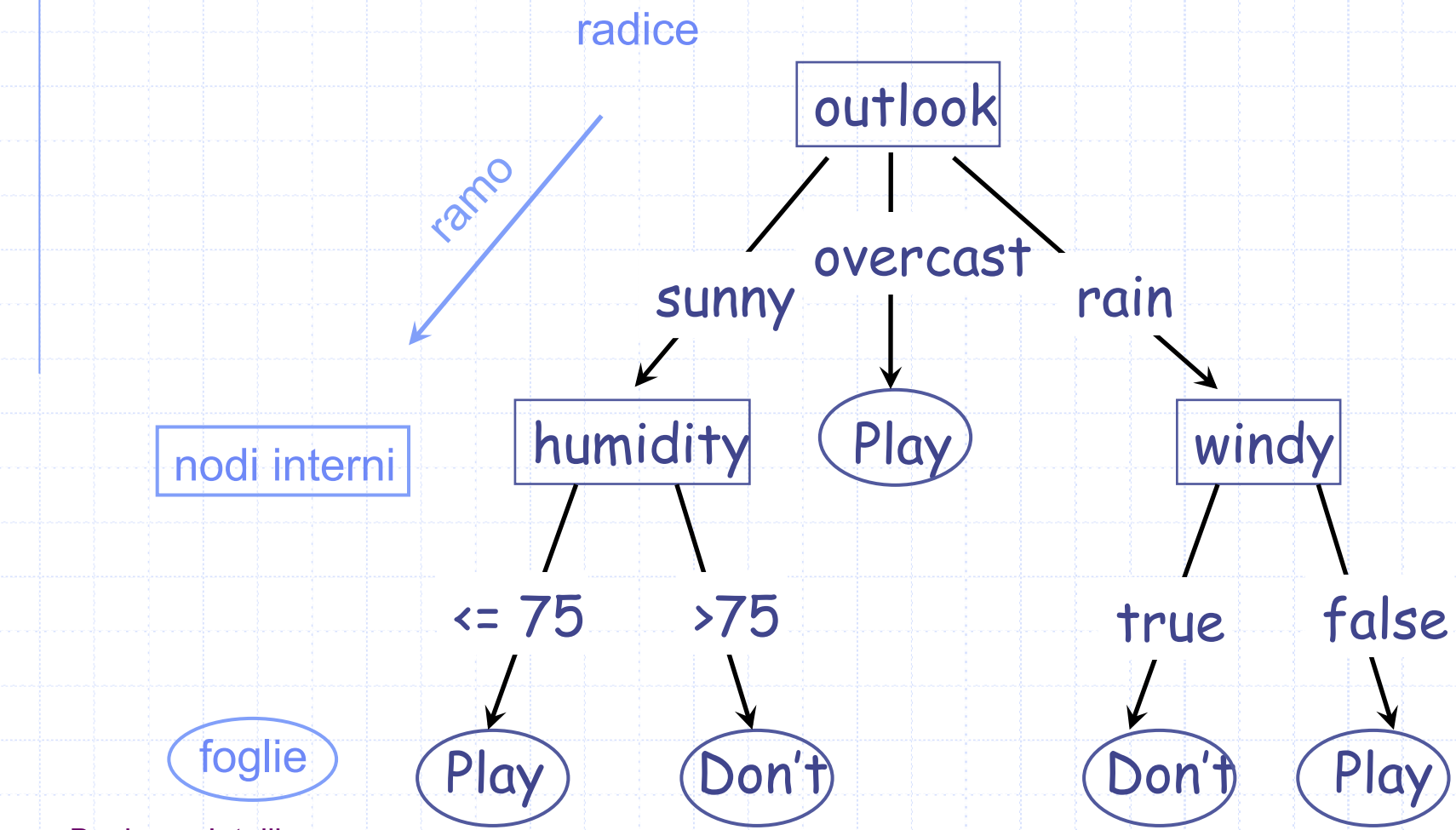
Classificazione e predizione: input

- Alcuni sistemi permettono di assegnare pesi differenti
  - a ciascun attributo
    - misurazione della temperatura VS radiografia
  - a ciascun esempio
    - Un cliente pesa in proporzione ai suoi ordini
  - a ciascun errore di classificazione
    - Perdere la possibilità di attrarre un cliente che fà molte spese vale di più del rischio di attrarre anche un cliente che fà poche spese

Classificazione e predizione: output

- Una funzione (modello) in grado di descrivere il valore della classe in base ai valori degli attributi, anche per casi non presenti nell'input
- Quando la classe assume valori discreti, si parla di modello di classificazione
  - Alberi di decisione, Regole di classificazione, SVM, Classificazione Bayesiana, K-nearest neighbors, Case-based reasoning, Regressione logistica, Association-based classification, …
- Quando la classe assume valori continui, si parla di modello di predizione numerica
  - Alberi di regressione, Reti neurali, regressione lineare e multipla, regressione non-lineare, regressione log-lineare, …

Modelli: Albero di Decisione



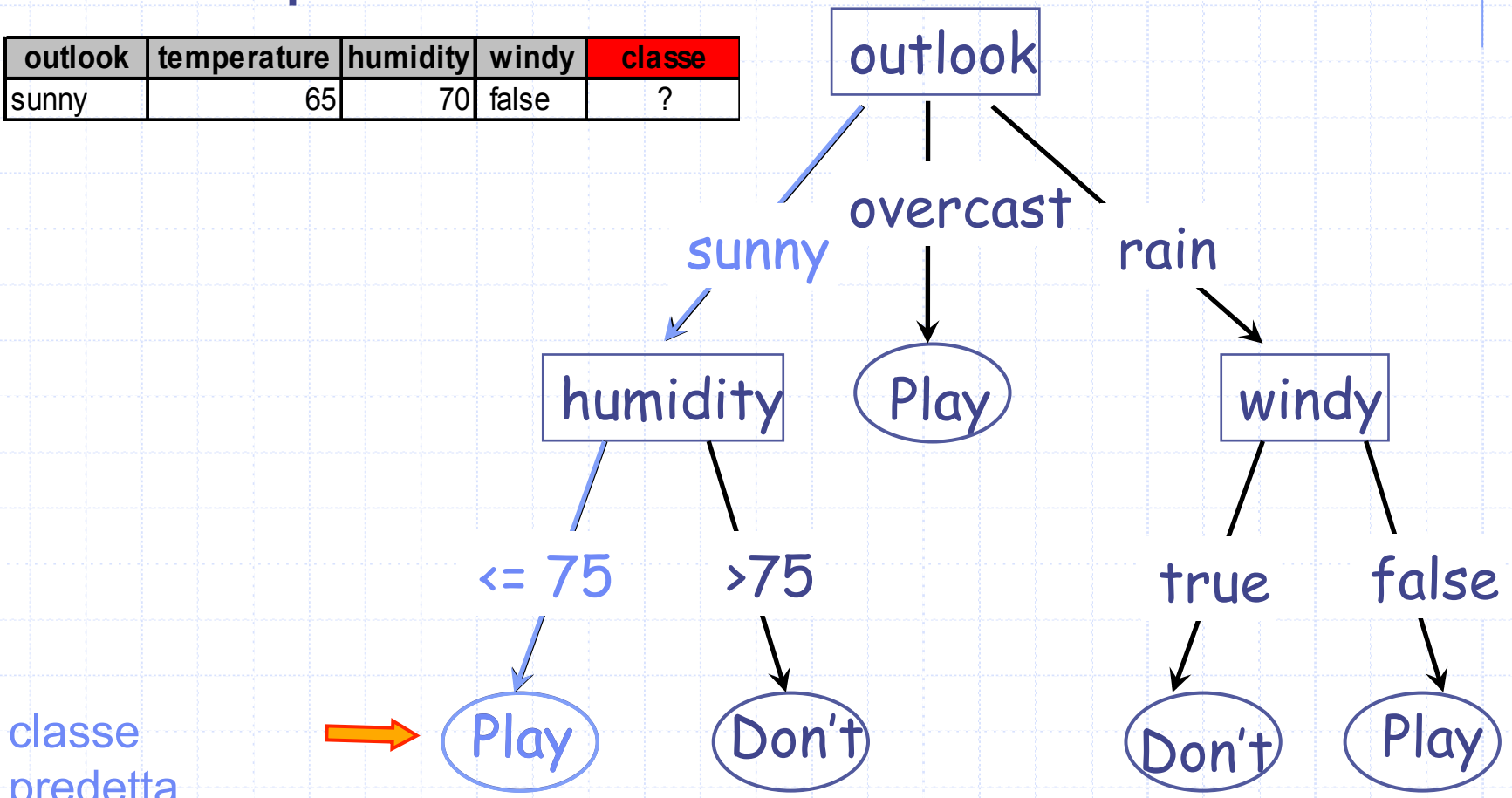
Uso di un modello e "score set"

- Un modello può essere usato:
  - a scopo descrittivo
    - descrivere l'appartenenza ad una classe in base agli attributi
      - descrivere le caratteristiche dei clienti che hanno una soglia minima di spesa
  - a scopo predittivo
    - predirre la classe di un nuovo esempio
    - validation o score set = esempi di cui si conoscono gli attributi *ma non la classe*
      - selezionare i clienti cui proporre una nuova campagna pubblicitaria
      - predirre la capacità di spesa di un cliente

“Usare” un albero per predizioni

- Classe predetta da un albero di decisione

outlook	temperature	humidity	windy	classe
sunny	65	70	false	?

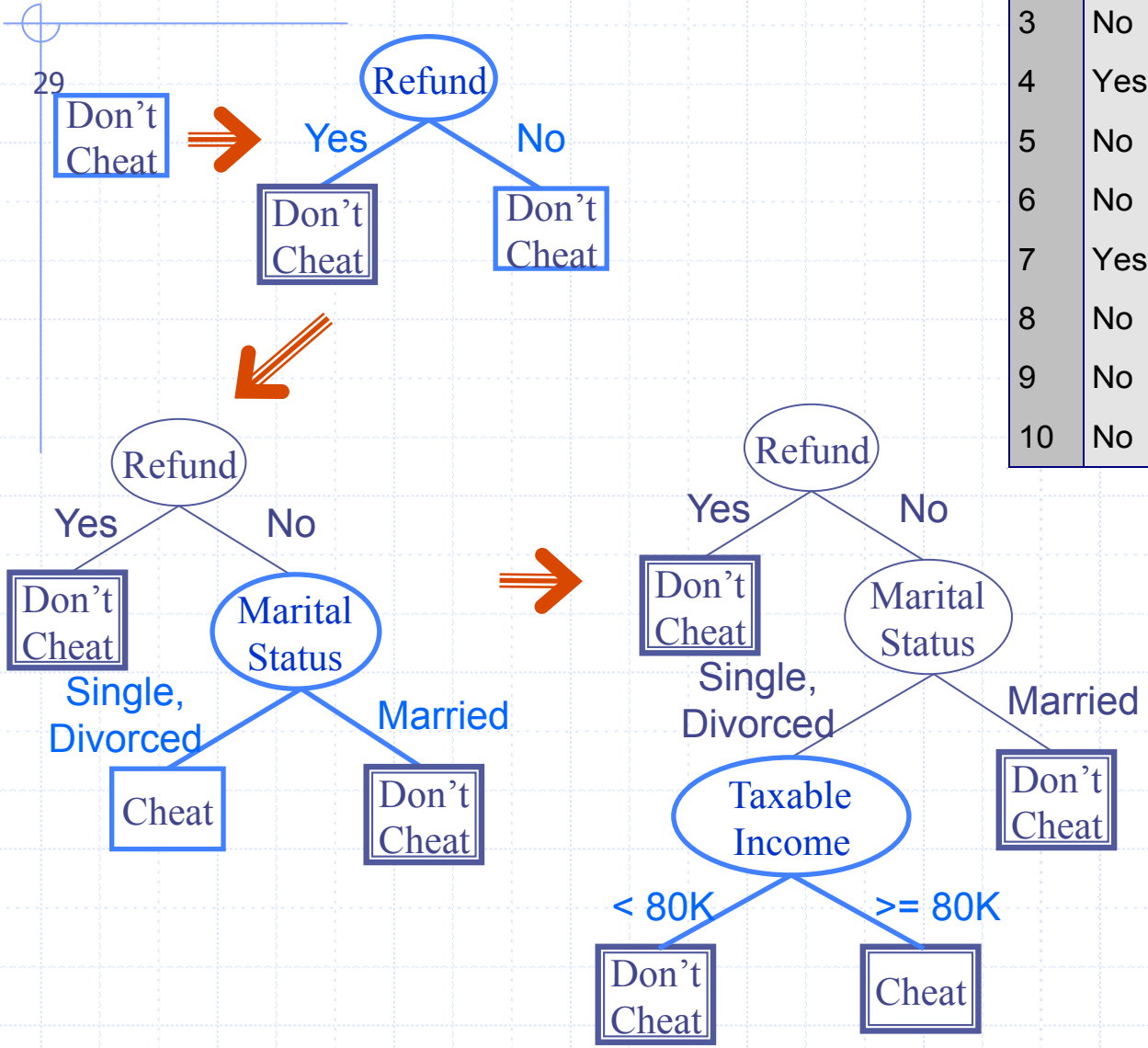


classe predetta

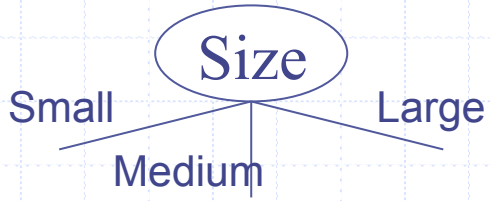
Costruire un albero di decisione

- Algoritmo di base
  - Costruzione top-down
    - Ciascun nodo ha associato un insieme di esempi (al nodo radice è associato tutto il training set)
    - In ogni nodo viene selezionato un singolo attributo
      - Per gli attributi discreti viene creato un figlio per ciascun possibile valore (variante con creazione di soli due figli)
      - Per gli attributi continui vengono creati due figli "<= threshold" e "> threshold" (variante con creazione di piu' di due figli)
    - Gli esempi del nodo vengono ripartiti tra i tra i figli del nodo
  - Terminazione
    - (**MinObjNum**) Numero di esempi nel nodo inferiore ad un valore fissato
    - (**Purity level**) percentuale di esempi appartenenti ad una classe maggiore di un valore fissato

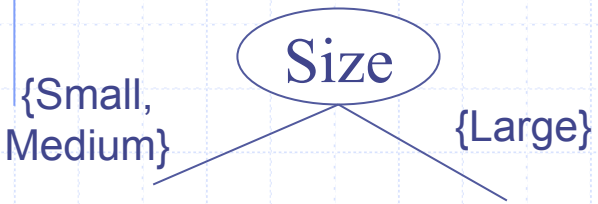
Tid	Refund	Marital Status	Taxable Income	Cheat
1	Yes	Single	125K	No
2	No	Married	100K	No
3	No	Single	70K	No
4	Yes	Married	120K	No
5	No	Divorced	95K	Yes
6	No	Married	60K	No
7	Yes	Divorced	220K	No
8	No	Single	85K	Yes
9	No	Married	75K	No
10	No	Single	90K	Yes



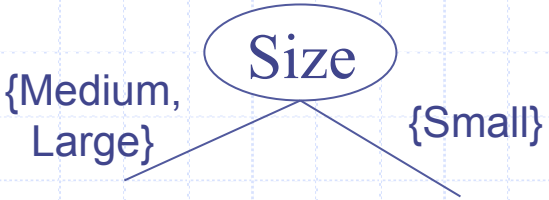
Split su attributi discreti



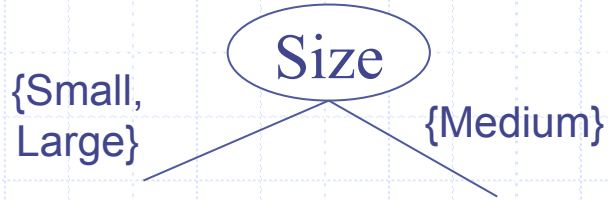
Multi-way



Binary

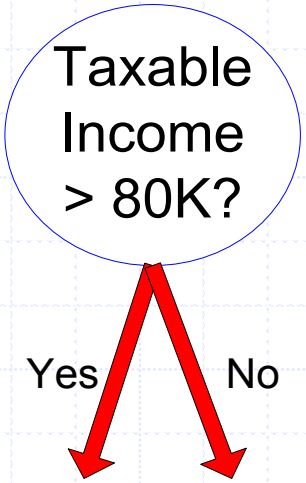


Binary

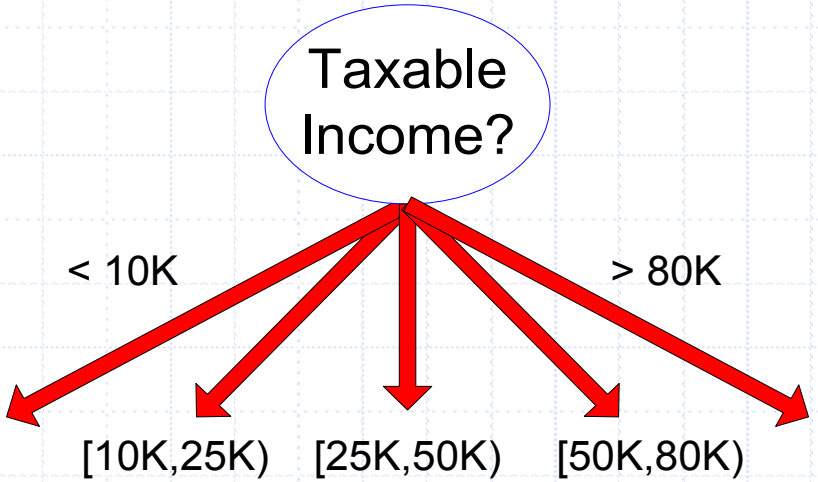


?

Split su attributi continui

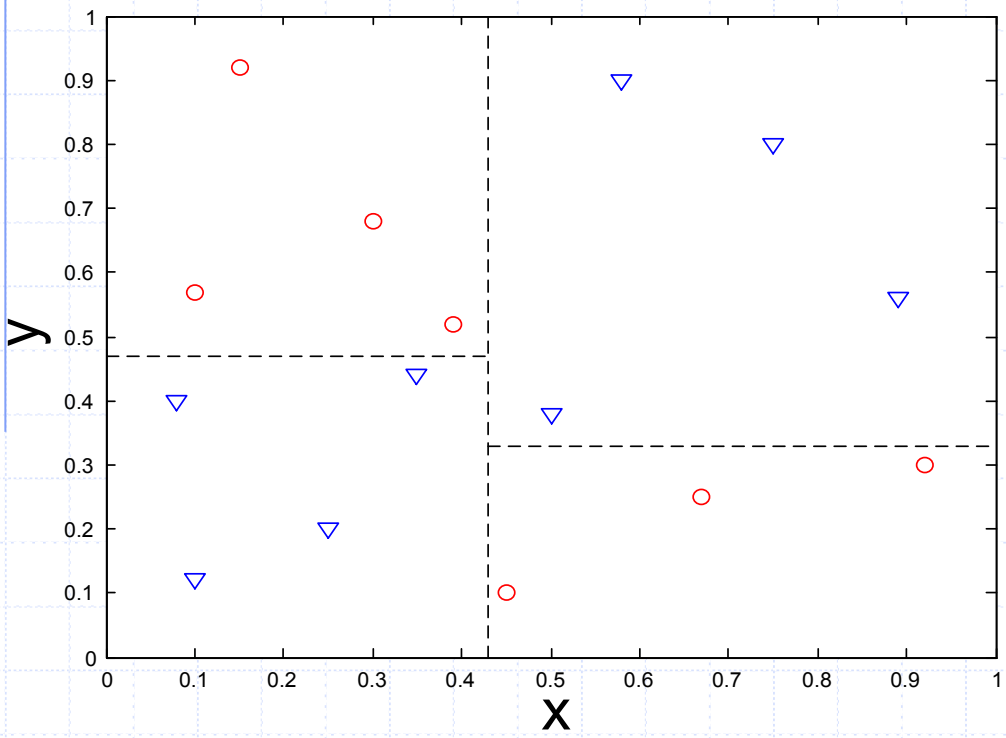


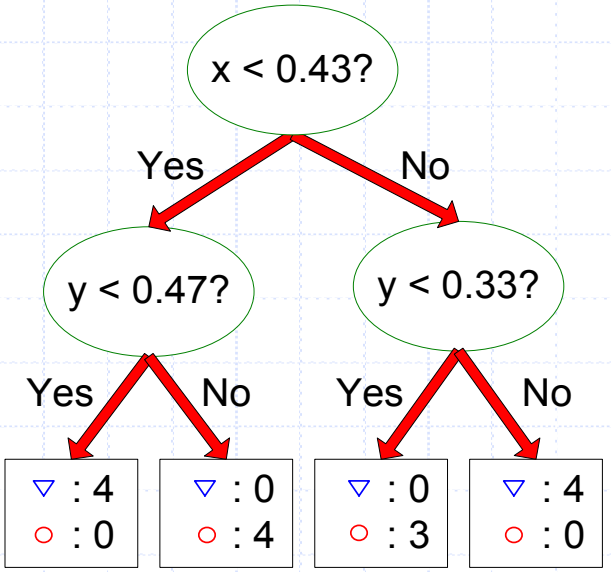
(i) Binary split



(ii) Multi-way split

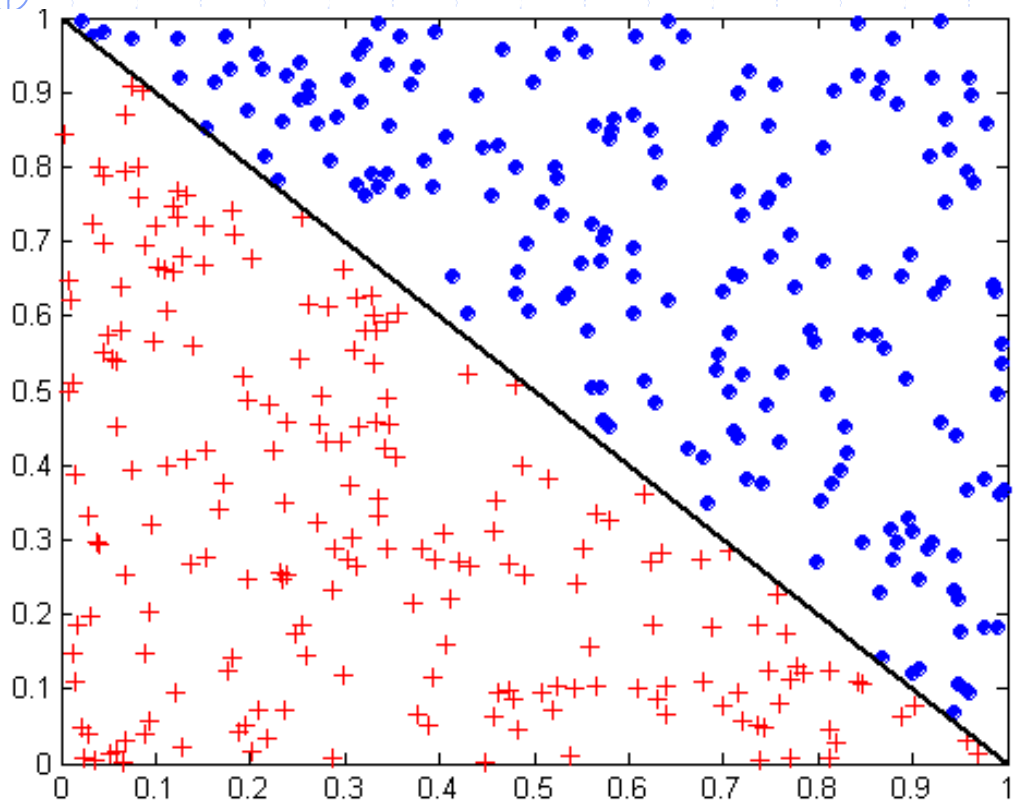
Split su attributi continui

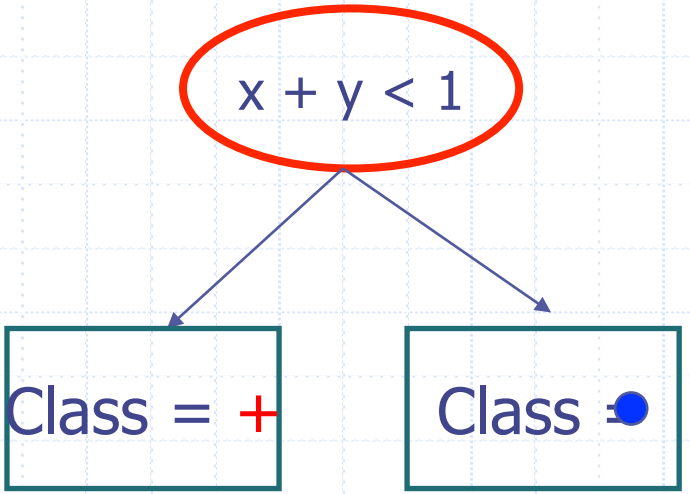




Split della forma "X < c" dividono lo spazio dei valori parallelamente agli assi.

Oblique Decision Trees





Split della forma "aX + bY < c" dividono lo spazio dei valori lungo rette qualsiasi. Più espressivi, ma meno efficienti.

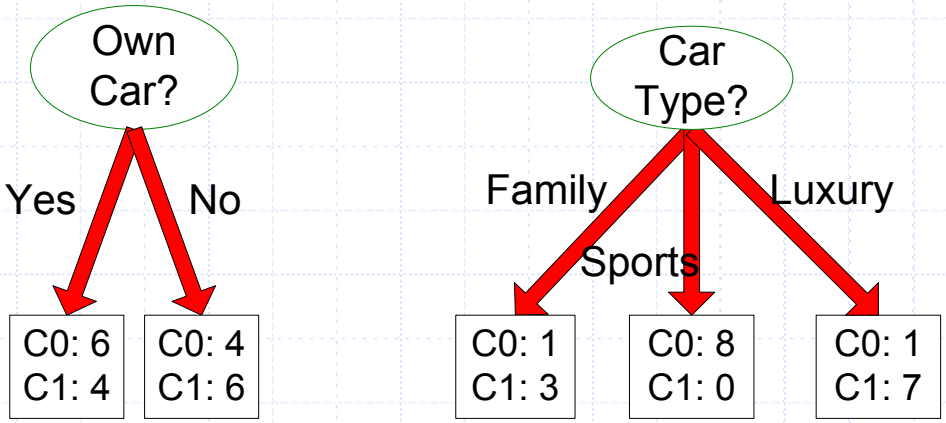
Costruire un albero di decisione

- Varianti dell'algoritmo di base:
  - dal **machine learning**: ID3 (Iterative Dichotomizer), C4.5
  - dalla **statistica**: CART (Classification and Regression Trees)
  - da **pattern recognition**: CHAID (Chi-squared Automated Interaction Detection)
- Differenze:
  - criterio di selezione dell'attributo ***(splitting criteria)***
    - Information gain/gain ration (ID3 – C4.5)
    - Gini index -- noto anche come Quadratic entropy (CART)
    - Twoing rule (CART)
    - $\chi 2$ contingency table statistic (CHAID)
    - Normalized distance measure

Splitting criteria

10 casi di classe C0

10 casi di classe C1



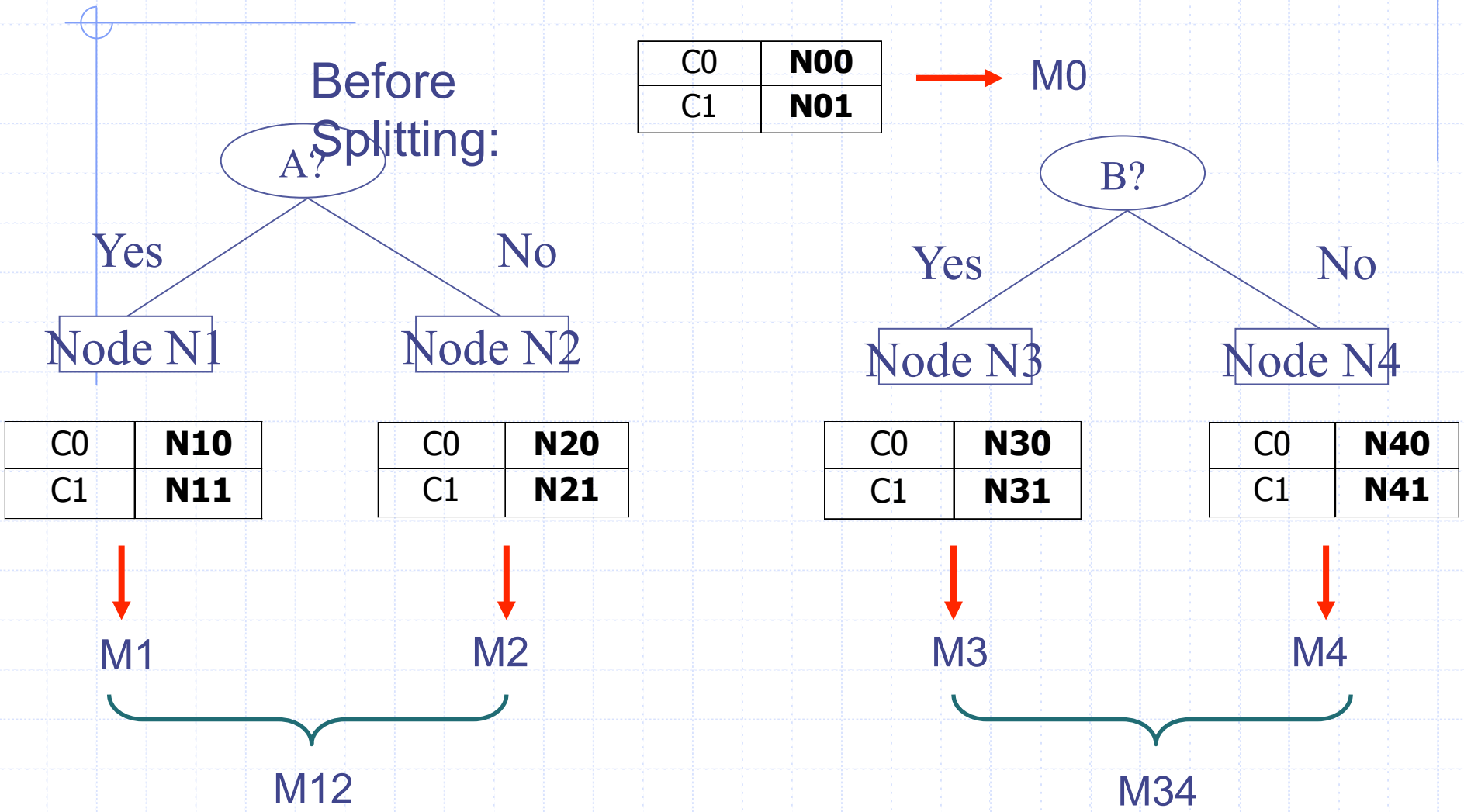
Quale attributo selezionare?

Quale attributo selezionare?

- Approccio greedy:
  - Si cerca di ottenere nodi con distribuzioni omogenee dei valori delle classi
  - Misura di impurità (GINI index, Information gain):

C0: 5 C1: 5	C0: 9 C1: 1
Non omogeneo	Omogeno
Alto grado di impurità	Basso grado di impurità

Quale attributo selezionare?



Gain = M0 – M12 vs M0 – M34

Information Gain

- Entropia ad un nodo 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).

  - Misura di omogeneità di un nodo
    - Massima (log n_c) quando i casi sono equamente distribuiti tra i valori delle classi
    - Minima (0.0) quando i casi appartengono tutti alla stessa classe

Information Gain

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

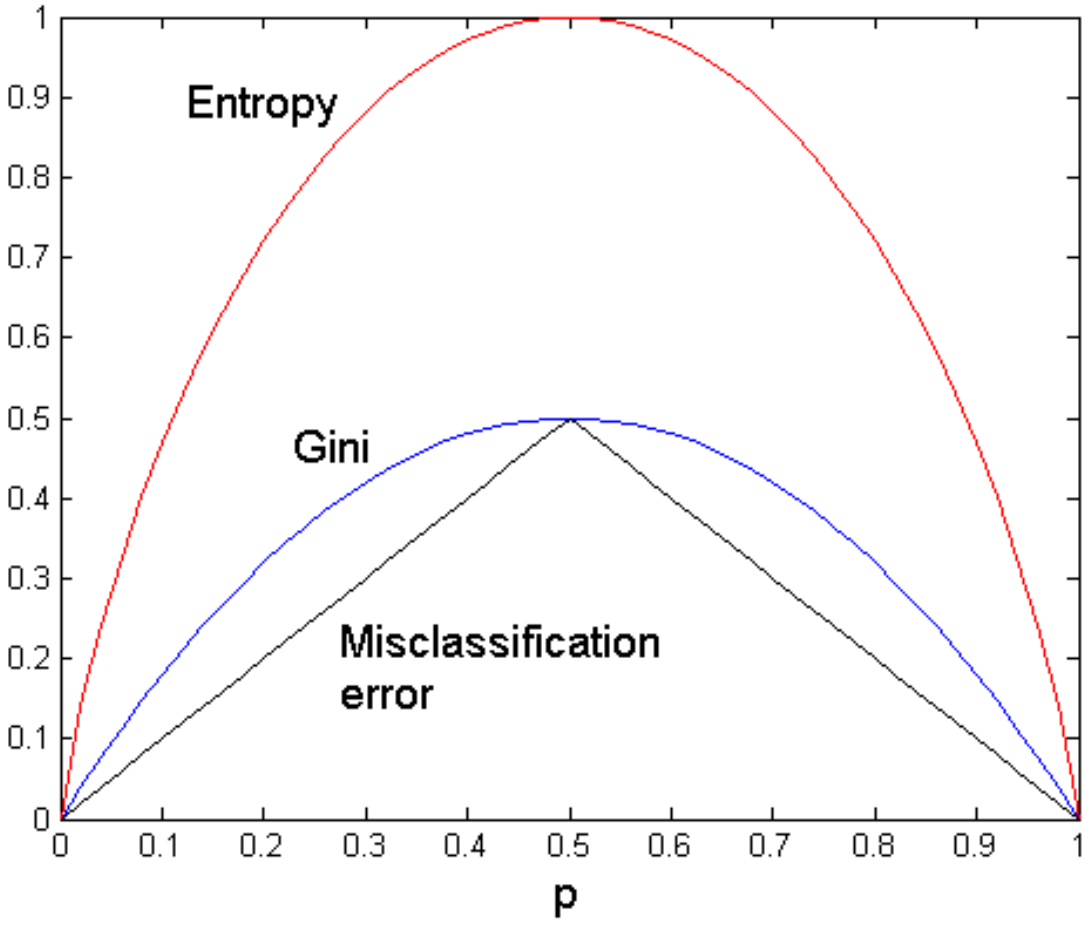
Parent Node, p is split into k partitions;

n_i is number of records in partition i

- Misura la riduzione di entropia ottenuta dopo lo split
- Viene scelto lo split che ottimene la massima riduzione

Esempio: 2 classi

p = frazione dei casi di una delle due classi



Costruire un albero di decisione

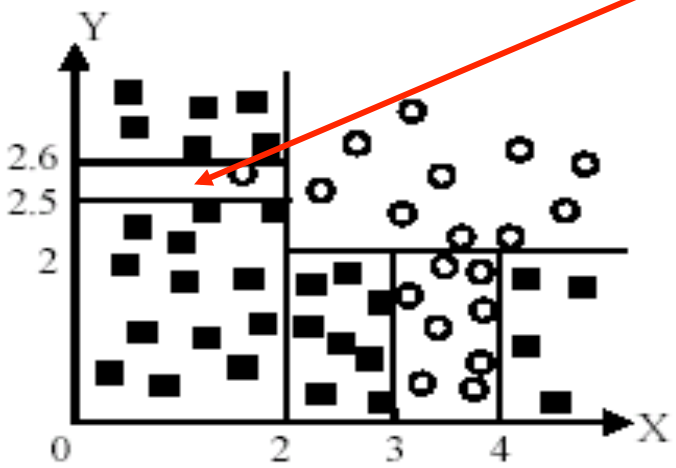
- Algoritmo di base
  - Costruzione top-down
    - Ciascun nodo ha associato un insieme di esempi (al nodo radice è associato tutto il training set)
    - In ogni nodo viene selezionato un singolo attributo
      - Per gli attributi discreti viene creato un figlio per ciascun possibile valore (variante con creazione di soli due figli)
      - Per gli attributi continui vengono creati due figli "<= threshold" e "> threshold" (variante con creazione di piu' di due figli)
    - Gli esempi del nodo vengono ripartiti tra i tra i figli del nodo
  - Terminazione
    - (**MinObjNum**) Numero di esempi nel nodo inferiore ad un valore fissato
    - (**Purity level**) percentuale di esempi appartenenti ad una classe maggiore di un valore fissato

Il problema dell'Overfitting

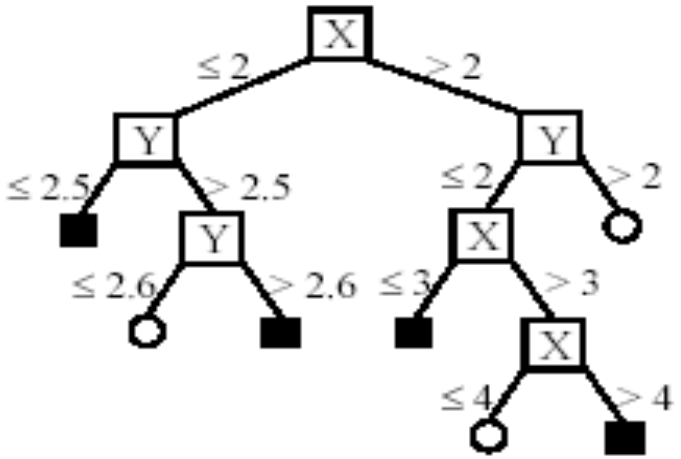
- Obiettivo della classificazione
  - trovare un modello che sia accurato su casi non visti
- Problema dell'overfitting
  - Un albero di decisione, però può descrivere bene aspetti **troppo specifici** degli esempi del training set e non generalizzabili al test e score set
  - Sorgenti di overfitting:
    - selezione random del training set
    - rumore nei dati (noise)
    - dati al limite (outliers)
    - pochi esempi di training
    - limitazioni (bias) dell'algoritmo di costruzione dell'albero

Il problema dell' Overfitting

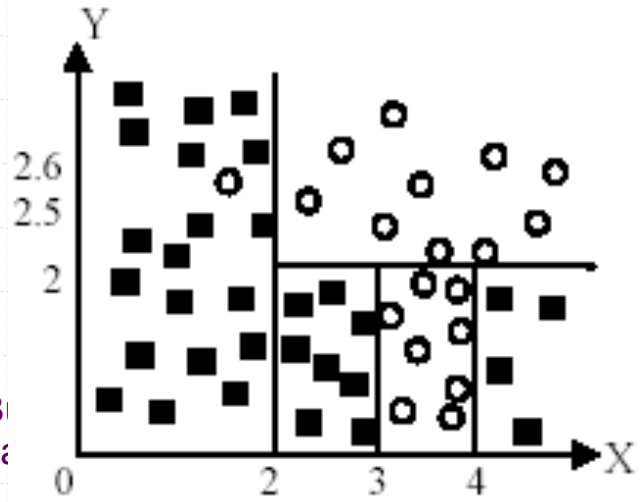
Probabile overfitting

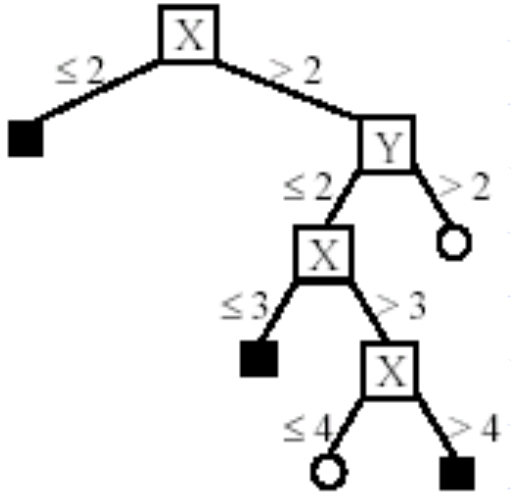


(A) A partition of the data space

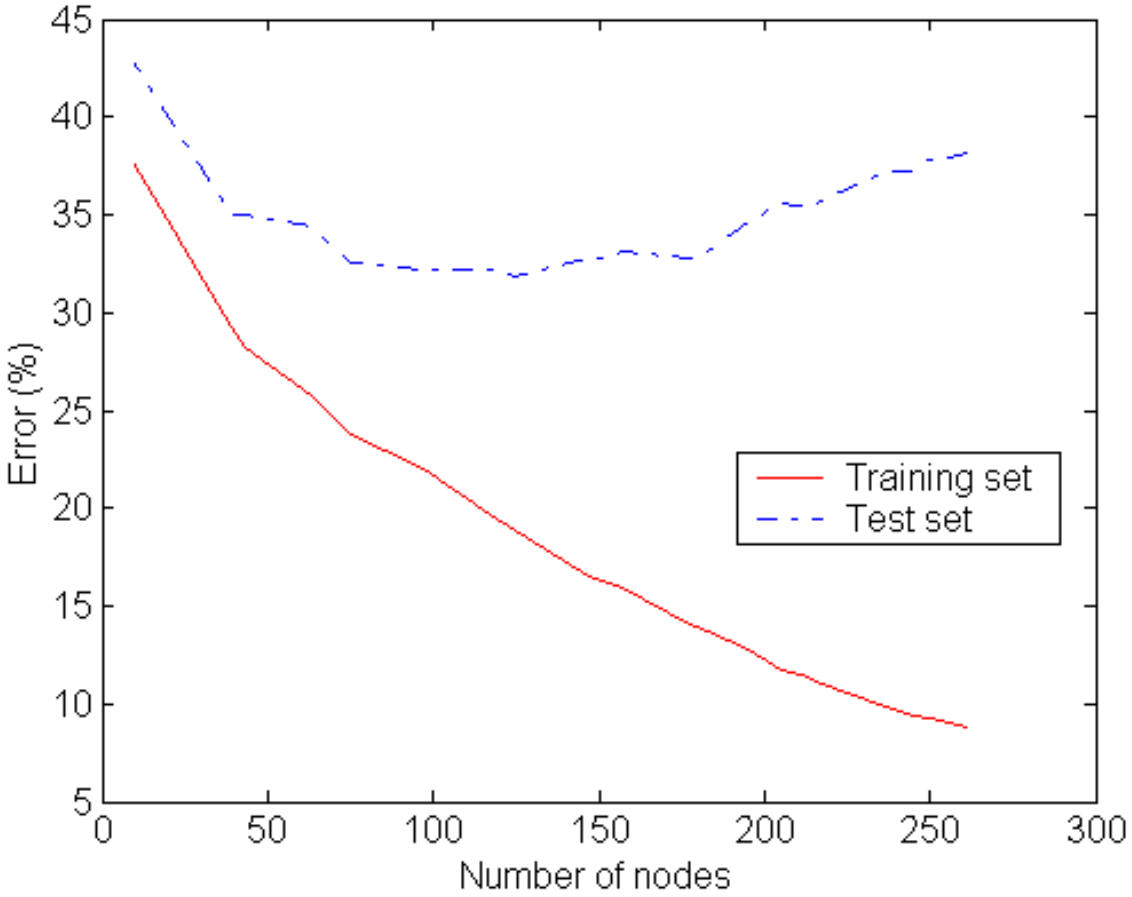


(B). The decision tree





Il problema dell'Overfitting



Euristiche per l'Overfitting

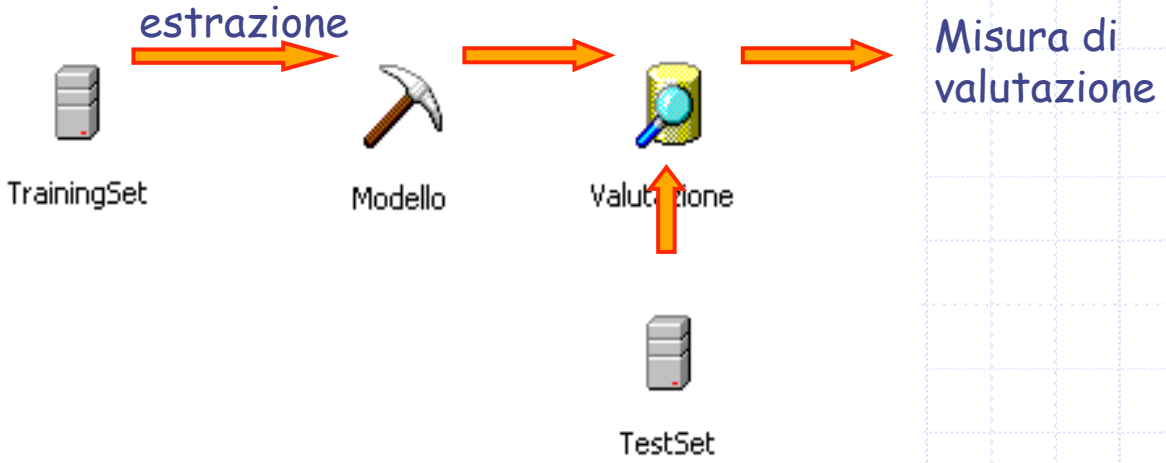
- Rasoio di Occam
  - A parità di accuratezza, meglio un classificatore "semplice" di uno complicato
- Due euristiche per implementare il rasoio di Occam:
  - **Stop earlier**: fermare la crescita dell'albero durante la costruzione
  - **Post prune** (o semplificazione): costruire interamente l'albero e poi **semplificarlo** tagliando alcuni rami
- Il **post-pruning** produce risultati migliori
  - Richiede un maggior costo computazionale

Valutazione (e scelta) dei modelli

- Valutazione
  - Accuratezza e altre misure di bontà
- Efficienza di estrazione e di predizione
- Robustezza
  - rispetto a rumore e outliers
- Scalabilità
  - applicabilità a grandi volumi di dati
- Interpretabilità del modello
- Semplicità
  - dimensione dell'albero, numero di regole, ...
- Indicatori dipendenti dal dominio
  - best practices, confronti con letteratura, benchmarks, ...

Valutazione: metodo "holdout"

- Test set
  - insieme di esempi indipendenti dal training set
  - di ciascun esempio si conoscono gli attributi e la classe reale
- Metodo "holdout"
  - valutare la bontà del modello sul test set
  - dato un insieme di esempi lo si divide (casualmente) in una percentuale per il training ed una per il test set (in genere 2/3 per il training e 1/3 per il test set)



Misure: accuratezza

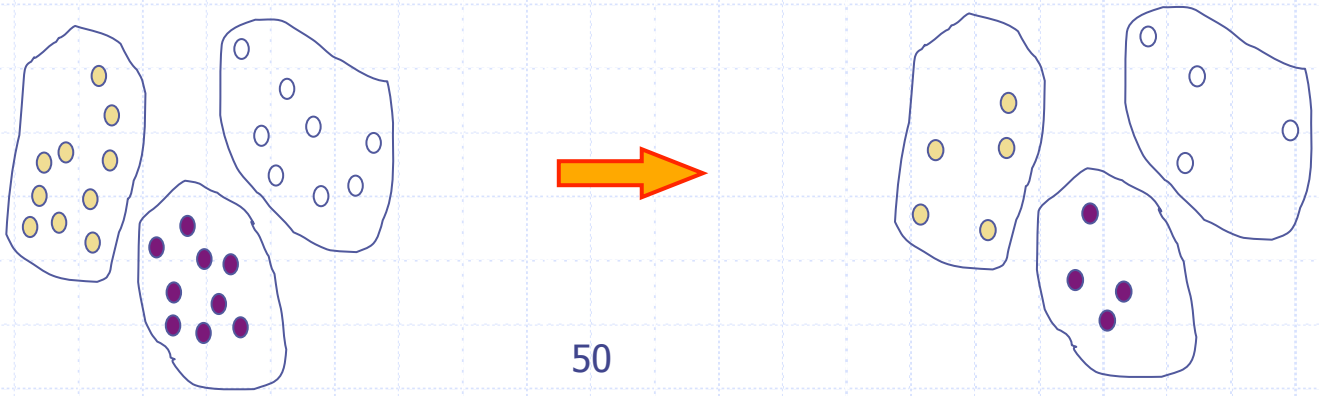
- Accuratezza di un modello M su un insieme T
  - percentuale di casi di T la cui classe predetta da M coincide con la classe reale
- Accuratezza di un modello
  - accuratezza del modello sul test set
    - L'accuratezza sugli esempi del training set è **ottimistica**
    - È facile costruire un modello accurato al **100%** sul training set
- % di misclassificazione = (100 – accuratezza)

Accuratezza: limiti inferiori

- Accuratezza di un classificatore casuale
  - k valori per la classe
  - accuratezza = 1 / k * 100
  - esempio: per k = 2 l'accuratezza è del 50%
- Accuratezza di un classificatore a maggioranza
  - classificatore che classifica sempre come la classe più frequente nel training set
  - esempio: con due classi rappresentate al 98% e al 2% nel training set, l'accuratezza è del 98% (se la stessa proporzione viene mantenuta anche nel test set)
- Qualsiasi classificatore deve fare almeno meglio del classificatore a maggioranza!!

Valutazione: "stratified holdout"

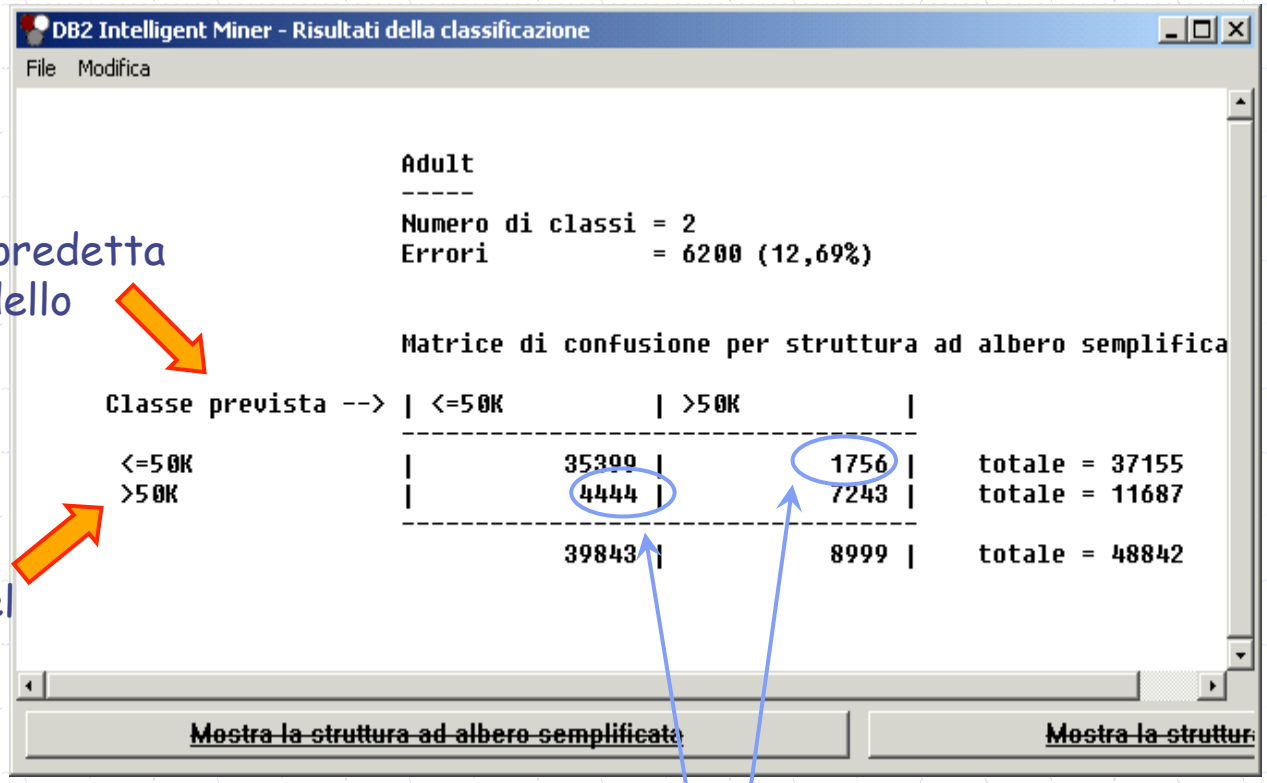
- Metodo "stratified holdout"
  - la divisione **casuale** di esempi in training e test set potrebbe generare un training set non rappresentativo della popolazione
    - proporzione delle classi diversa nel training e nel test set
  - "stratified holdout" consiste in una divisione che mantenga nel training e test set la stessa distribuzione delle classi dell'insieme di partenza



Valutazione: "cross-validation"

- Metodo "(stratified) n-fold cross-validation"
  - Si dividono gli esempi in n insiemi S_1, …, S_n delle stesse dimensioni e con distribuzione delle classi uguale a quella dei dati originari (se stratified)
  - Per ciascuno insieme S_i
    - Si costruisce un classificatore con training set pari altri n-1 insiemi
    - Si usa l'insieme S_i come insieme di test del classificatore
  - La valutazione finale è data dalla **media** delle valutazioni degli n classificatori
    - Metodo standard: n = 10, stratified 10-fold cross validation
  - Quando i dati sono molto pochi oppure quando si vuole ridurre la probabilità che il training/test set siano non rappresentativi della distribuzione

Misure: matrice di confusione



Classe predetta dal modello

Classe reale del training/test set

Esempi misclassificati

Misure: matrice di confusione

	Classified Positive	Classified Negative
Actual Positive	TP	FN
Actual Negative	FP	TN

where

TP: the number of correct classifications of the positive examples (**true positive**),

FN: the number of incorrect classifications of positive examples (**false negative**),

FP: the number of incorrect classifications of negative examples (**false positive**), and

TN: the number of correct classifications of negative examples (**true negative**).

Val. rispetto ad un valore della classe

- Quando interessi la bontà del modello sulla descrizione/ predizione del valore "C"
  - Classe = {nuovoCliente, vecchioCliente }, C = nuovoCliente
- Precision: accuratezza nella predizione di "C"

$$\frac{\#\text{ Casi predetti Classe=C e realmente Classe=C}}{\#\text{ Casi predetti Classe=C}}$$

- p = TP/(TP+FP)
- Una precision dell' 85%
  - Se il modello risponde "nuovoCliente" allora 85 volte su 100 si tratta effettivamente di un nuovo cliente

Val. rispetto ad un valore della classe

- Recall: copertura rispetto al totale dei "C"

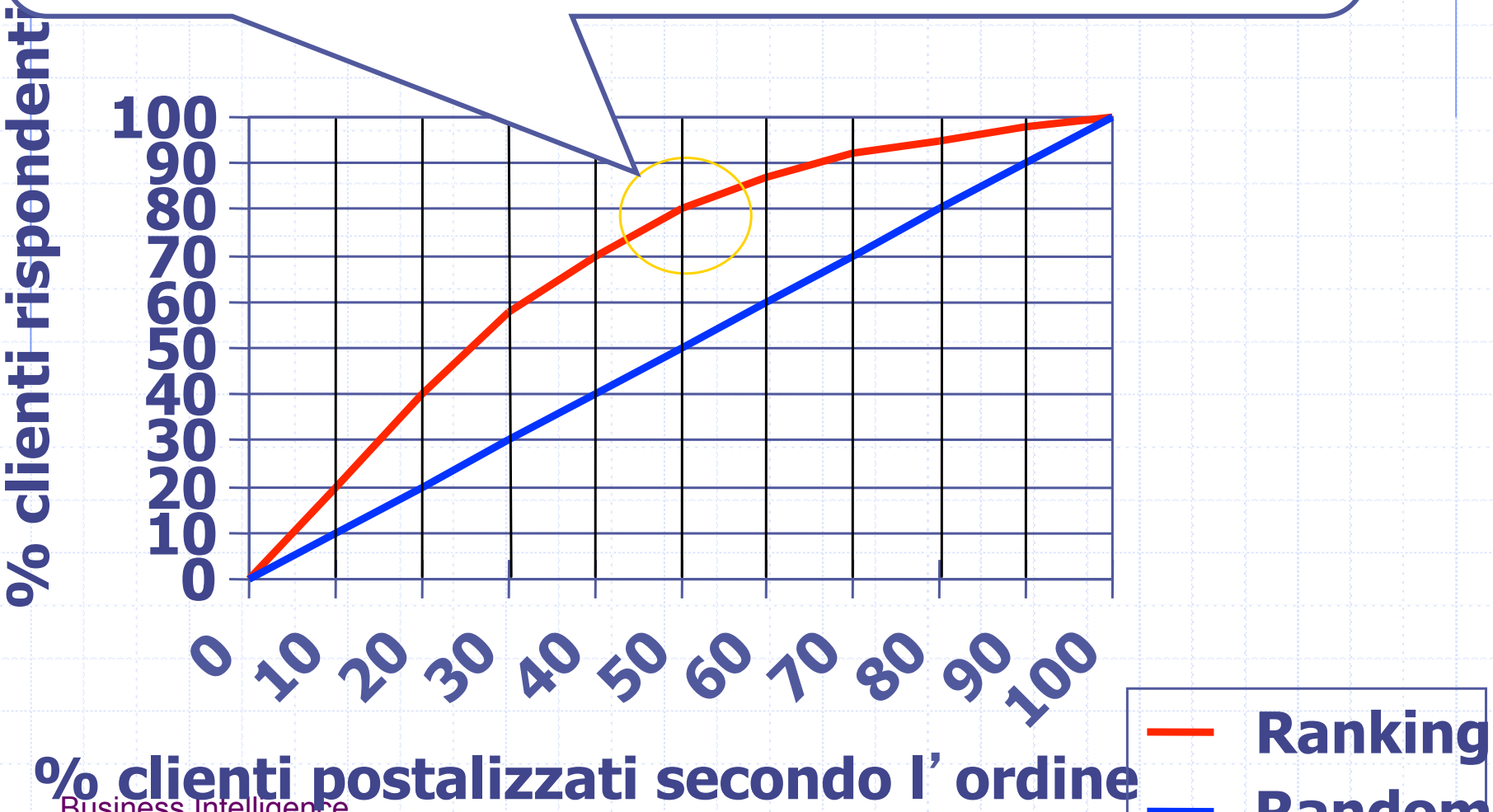
$$\frac{\#\text{ Casi predetti Classe=C e realmente Classe=C}}{\#\text{ Casi reali Classe=C}}$$

- Un recall dell' 85%
  - Il modello risponde "nuovoCliente" per 85 nuovi clienti su 100
- r = TP/(TP+FN)

Lift Chart

- Lift Chart
  - Un classificatore produce una confidenza per la classe “C”
  - Asse X = casi ordinati per confidenza descrescente
  - Asse Y = recall ottenuti considerando solo i casi in X
- Lift chart con valori di lift
  - Sulle Y sono rappresentati i lift invece dei recall
- Profit chart
  - Dato un modello di guadagno, sulle Y sono rappresentati i guadagni invece dei recall

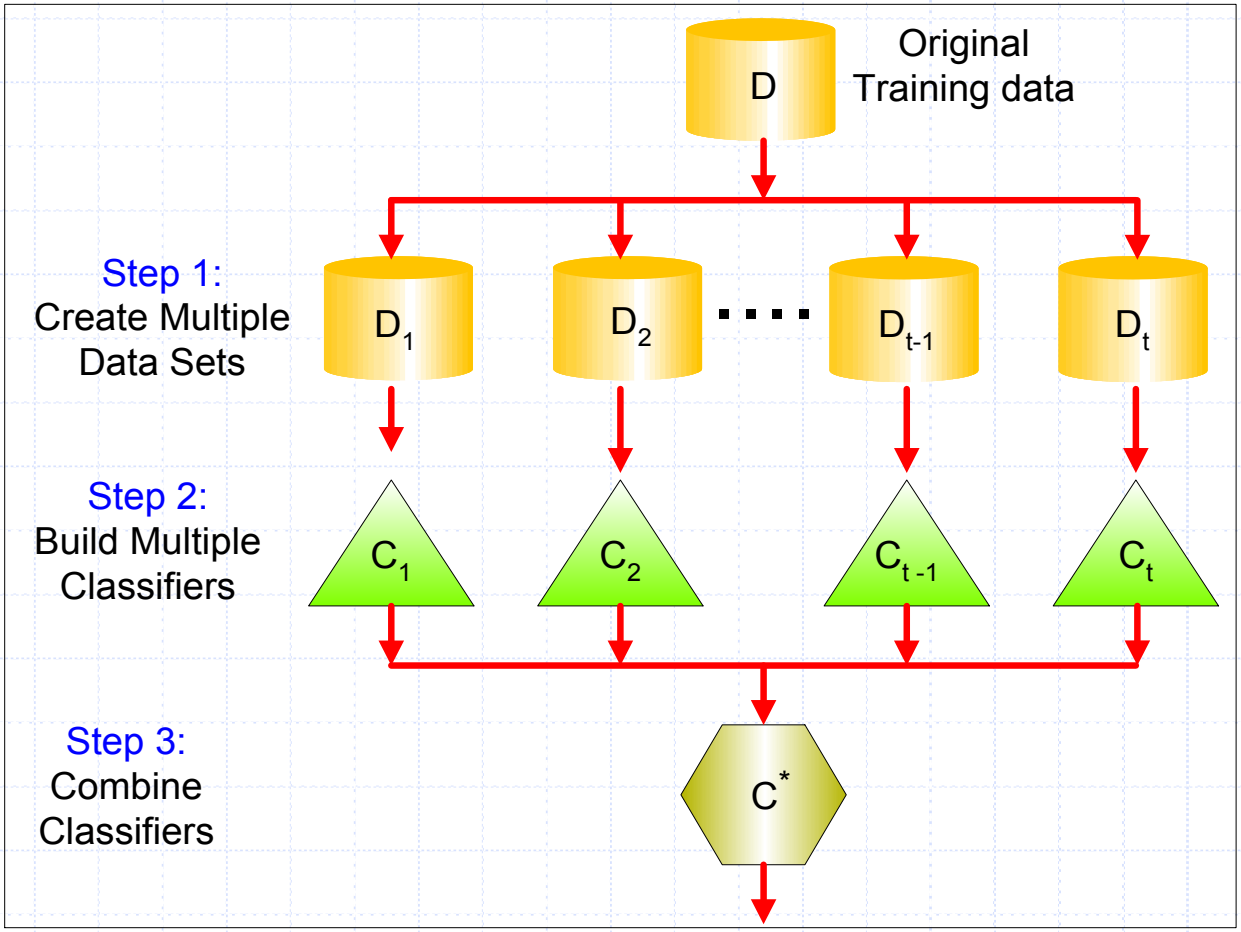
Con la metà delle postalizzazioni si **raggiunge** l'80% dei clienti che risponderanno. Lift = 80/50 = 1.6



Ottimizzare l'accuratezza

- Quando le classi sono molto sbilanciate, il modello inferito può essere poco informativo
  - Evasori = 2% Non evasori = 98%
    - Un classificatore che risponde sempre 'Non evasori' sbaglia solo nel 2% dei casi, ma non è di alcuna utilità
- Oversampling e Undersampling
  - Selezionare un **training** set con una distribuzione delle classi A e B maggiormente equilibrata
    - 60-70% per la classe A e 30-40% per la classe B
    - Aumentando il numero di esempi di classe B (oversampling) o diminuendo quelli di classe A (undersampling)
  - La costruzione del classificatore *ha maggiori esempi* per distinguere B da A
    - Il test set rimane con la distribuzione originaria

Ensembling methods



Ottimizzare l'accuratezza

- Costruendo due o più modelli: esembling
  - Su training set diversi
    - **Bagging**
      - costruisce alberi su training set della stessa dimensione ottenuti mediante selezione (con rimpiazzamento) da un insieme di esempi di partenza
      - classificazione mediante votazione a maggioranza
  - Sullo stesso training set, ma ogni volta con pesi diversi
    - **Boosting**
      - costruisce alberi rimodellando i pesi degli esempi misclassificati dagli alberi già costruiti
      - classificazione mediante votazione pesata rispetto all'accuratezza
  - Meta-learning
    - **Stacking**
      - costruisce un albero in grado di predirre quale predizione sia migliore tra quelle di due o più altri alberi

Other classification methods

Backpropagation

- Is a neural network algorithm, performing on multilayer feed-forward networks (Rumelhart et al. 86).
- A network is a set of connected input/ output units where each connection has an associated weight.
- The weights are adjusted during the training phase, in order to correctly predict the class label for samples.

Backpropagation

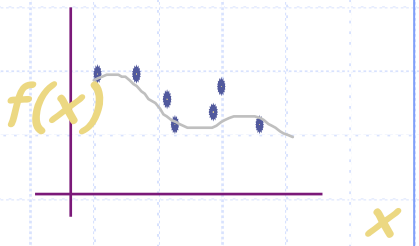
PROS

- High accuracy
- Robustness w.r.t. noise and outliers

CONS

- Long training time
- Network topology to be chosen empirically
- Poor interpretability of learned weights

Prediction and (statistical) regression



- Regression = construction of models of continuous attributes as functions of other attributes
- The constructed model can be used for prediction.
- E.g., a model to predict the sales of a product given its price
- Many problems solvable by linear regression, where attribute Y (response variable) is modeled as a linear function of other attribute(s) X (predictor variable(s)):

$$Y = a + b \cdot X$$

- Coefficients a and b are computed from the samples using the least square method.

Other methods (not covered)

- *K*-nearest neighbors algorithms
- Case-based reasoning
- Genetic algorithms
- Rough sets
- Fuzzy logic
- Association-based classification (Liu et al 98)

Classification: references

References - classification

- J. R. Quinlan. *C4.5: Programs for Machine Learning*. Morgan Kaufman, 1993.
- T. M. Mitchell. *Machine Learning*. McGraw-Hill, 1997.
- J. R. Quinlan. Induction of decision trees. *Machine Learning*, 1:81-106, 1986.
- L. Breiman, J. Friedman, R. Olshen, and C. Stone. *Classification and Regression Trees*. Wadsworth International Group, 1984.
- C. Apte and S. Weiss. Data mining with decision trees and decision rules. *Future Generation Computer Systems*, 13, 1997.

Example: an application in fiscal fraud detection

Fraud detection and audit planning

- A major task in fraud detection is constructing *models* of fraudulent behavior, for:
  - preventing future frauds (*on-line* fraud detection)
  - discovering past frauds (*a posteriori* fraud detection)
- Focus on a posteriori FD: analyze historical audit data to plan effective future audits
- Audit planning is a key factor, e.g. in the fiscal and insurance domain:
  - tax evasion (from incorrect/fraudulent tax declarations) estimated in Italy between 3% and 10% of GNP

Case study

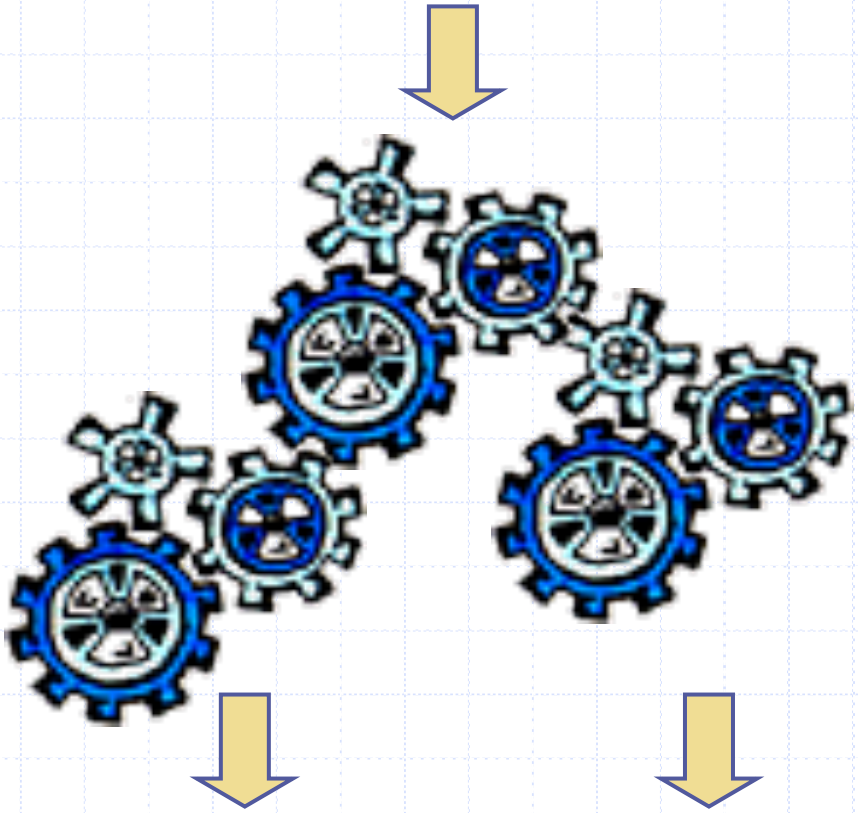
- Conducted by our Pisa KDD Lab (Bonchi et al 99)
- A data mining project at the Italian Ministry of Finance, with the aim of assessing:
  - the potential of a KDD process oriented to planning audit strategies;
  - a methodology which supports this process;
  - an integrated logic-based environment which supports its development.

Audit planning

- Need to face a trade-off between conflicting issues:
  - *maximize audit benefits*: select subjects to be audited to maximize the recovery of evaded tax
  - *minimize audit costs*: select subjects to be audited to minimize the resources needed to carry out the audits.
- Is there a KDD methodology which may be tuned according to these options?
- How extracted knowledge may be combined with domain knowledge to obtain useful audit models?

Autofocus data mining

policy options, business rules



fine parameter tuning of mining tools

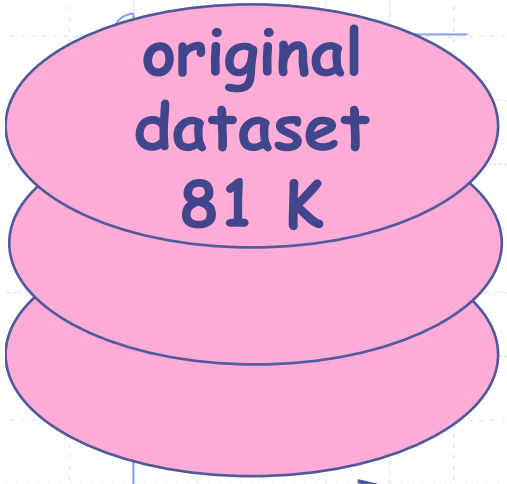
Classification with decision trees

- Reference technique:
  - Quinlan's C4.5, and its evolution C5.0
- Advanced mechanisms used:
  - pruning factor
  - misclassification weights
  - boosting factor

Available data sources

- Dataset: tax declarations, concerning a targeted class of Italian companies, integrated with other sources:
  - social benefits to employees, official budget documents, electricity and telephone bills.
- Size: 80 K tuples, 175 numeric attributes.
- A subset of 4 K tuples corresponds to the *audited* companies:
  - outcome of audits recorded as the *recovery* attribute (= *amount of evaded tax ascertained*)

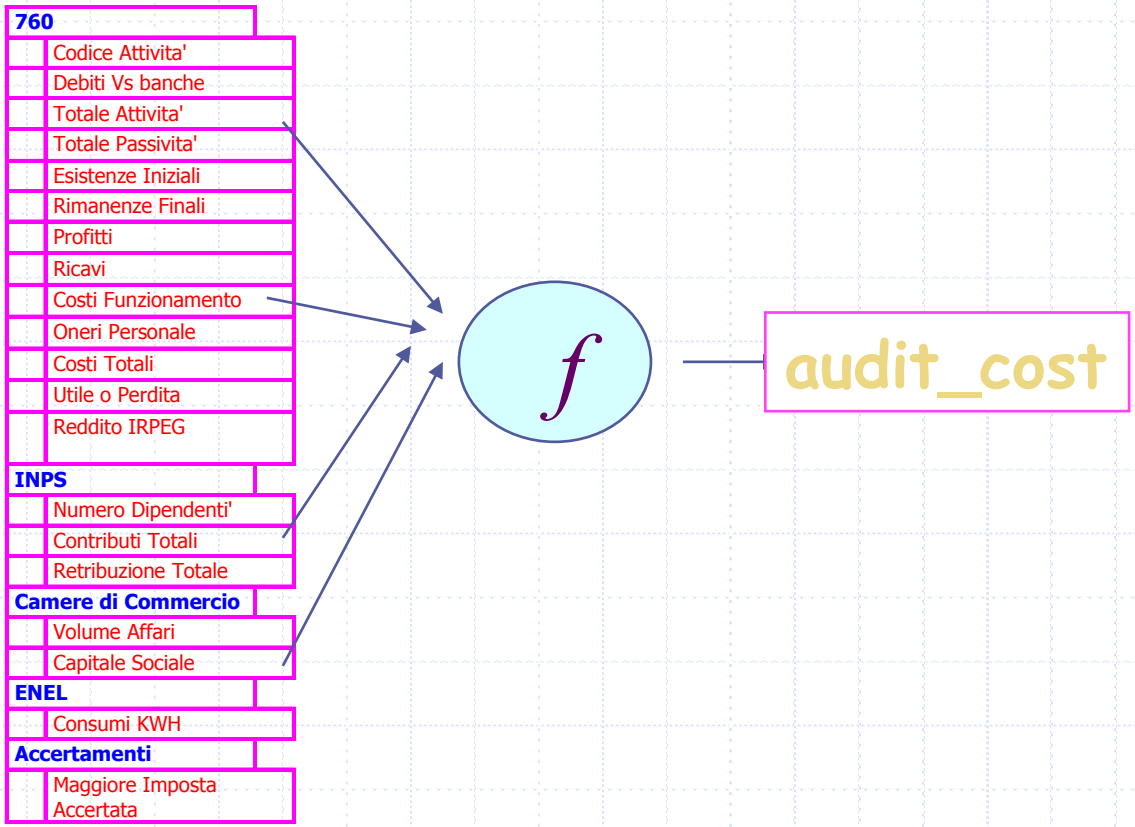
Data preparation



TAX DECLARATION
Codice Attivita'
Debiti Vs banche
Totale Attivita'
Totale Passivita'
Esistenze Iniziali
Rimanenze Finali
Profitti
Ricavi
Costi Funzionamento
Oneri Personale
Costi Totali
Utile o Perdita
Reddito IRPEG
SOCIAL BENEFITS
Numero Dipendenti'
Contributi Totali
Retribuzione Totale
OFFICIAL BUDGET
Volume Affari
Capitale Sociale
ELECTRICITY BILLS
Consumi KWH
AUDIT
Recovery

Cost model

- A derived attribute audit_cost is defined as a function of other attributes



Cost model and the target variable

- recovery of an audit after the audit cost
  actual_recovery = recovery - audit_cost

- target variable (class label) of our analysis is set as the Class of Actual Recovery (c.a.r.):

- *c.a.r.* =

$$\begin{cases} \textit{negative} & \textbf{\textit{if}}\ \ \textit{actual\_recovery} \leq 0 \\ \textit{positive} & \textbf{\textit{if}}\ \ \textit{actual\_recovery} > 0. \end{cases}$$

Training set & test set

- Aim: build a binary classifier with c.a.r. as target variable, and evaluate it
- Dataset is partitioned into:
  - training set, to build the classifier
  - test set, to evaluate it
- Relative size: incremental samples approach
- In our case, the resulting classifiers improve with larger training sets.
- Accuracy test with 10-fold cross-validation

Quality assessment indicators

- The obtained classifiers are evaluated according to several indicators, or metrics
- Domain-independent indicators
  - confusion matrix
  - misclassification rate
- Domain-dependent indicators
  - audit #
  - actual recovery
  - profitability
  - relevance

Domain-independent quality indicators

- confusion matrix (of a given classifier)

negative	positive	← classified as
TN	FP	actual class negative
FN	TP	actual class positive

TN (TP): true negative (positive) tuples
FN (FP): false negative (positive) tuples

misclassification rate =
(FN ∪ FP) / # test-set

Domain-dependent quality indicators

- audit # (of a given classifier): number of tuples classified as positive = # (FP ∪ TP)
- actual recovery: total amount of actual recovery for all tuples classified as positive
- profitability: average actual recovery per audit
- relevance: ratio between profitability and misclassification rate

The REAL case

- Classifiers can be compared with the REAL case, consisting of the whole test-set:

- audit # (REAL) = 366
- actual recovery(REAL) = 159.6 M euro

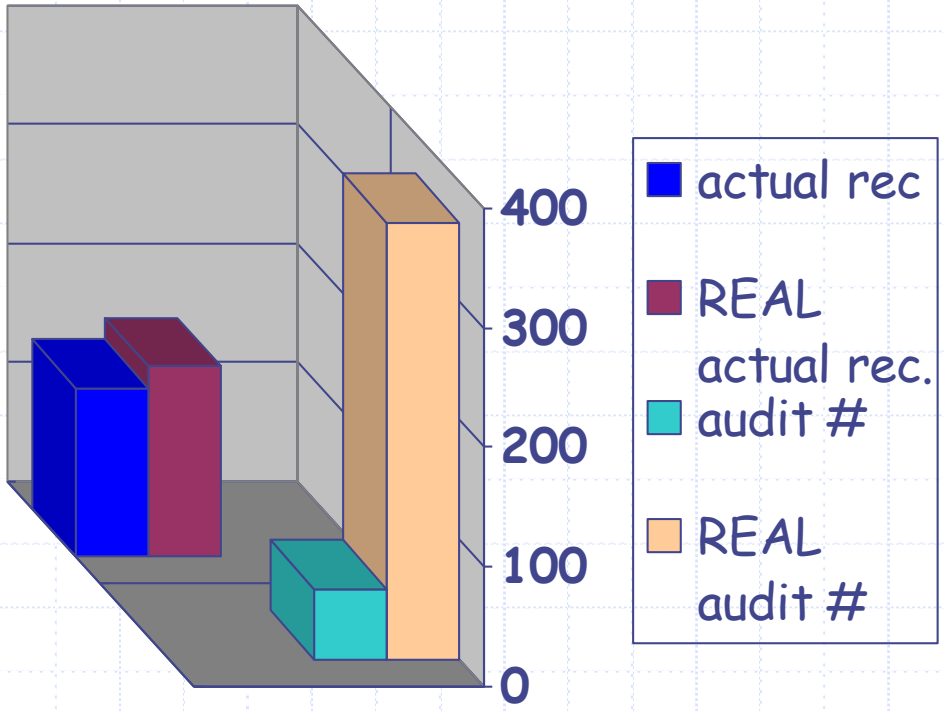
Controlling classifier construction

- *maximize audit benefits*: minimize FN
- *minimize audit costs*: minimize FP
- hard to get both!
  - unbalance tree construction towards eiher negatives or positives
- which parameters may be tuned?
  - misclassification weights, e.g., trade 1 FN for 10 FP
  - replication of minority class
  - boosting and pruning level

Model evaluation: classifier 1 (min FP)

- no replication in training-set (unbalance towards negative)
- 10-trees adaptive boosting

- *misc. rate* = 22%
- *audit #* = 59 (11 FP)
- *actual rec.*= 141.7 Meuro
- *profitability* = 2.401



Model evaluation: classifier 2 (min FN)

- replication in training-set (balanced neg/pos)
- misc. weights (trade 3 FP for 1 FN)
- 3-trees adaptive boosting

- *misc. rate* = 34%
- *audit #* = 188 (98 FP)
- *actual rec.*= 165.2 Meuro
- *profitability* = 0.878