Chapter VI: Classification

Knowledge Discovery in Databases

Luciano Melodia M.A.
Evolutionary Data Management, Friedrich-Alexander University Erlangen-Nürnberg
Summer semester 2021

Chapter VI: Classification

Classification: basic concepts.

Decision-tree induction.

Bayes classification methods.

Rule-based classification.

Model evaluation and selection.

Techniques to improve classification accuracy: ensemble methods.

Summary.

Supervised vs. unsupervised learning

Supervised learning (classification).

Supervision:

The **training data** (observations, measurements, etc.) are accompanied by **labels** indicating the **class** of the observations.

New data is classified based on a **model** created from the training data.

Unsupervised learning (clustering).

The class labels of training data are unknown.

Or rather, there are no training data.

Given a set of measurements, observations, etc.,
the goal is to find classes or clusters in the data.

See next chapter.

Prediction problems: classification vs. numerical prediction

Classification:

- Predicts **categorical class labels** (discrete, nominal).
- Constructs a model based on the training set and the values (class labels) in a classifying attribute and uses it in classifying new data.

Numerical prediction:

- Models **continuous-valued functions**.
- I.e. predicts missing or unknown (future) values.

Typical applications of classification:

- Credit/loan approval: Will it be paid back?
- Medical diagnosis: Is a tumor cancerous or benign?
- Fraud detection: Is a transaction fraudulent or not?
- Web-page categorization: Which category is it?

Classification – a two-step process

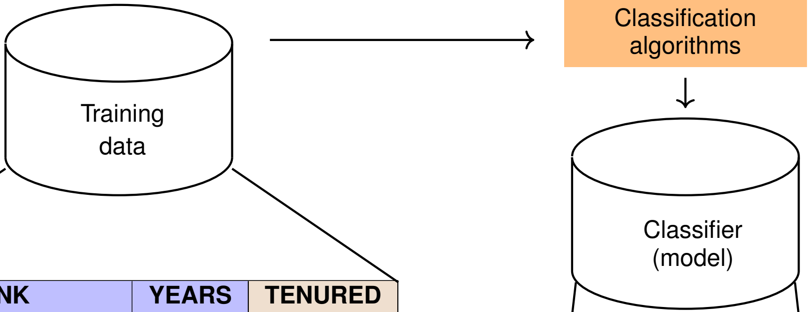
Model construction: describing a set of predetermined classes:

- Each tuple/sample is assumed to belong to a predefined class, as determined by the **class-label attribute**.
- The set of tuples used for model construction is the **training set**.
- The **model** is represented as classification rules, decision trees, or mathematical formulae.

Model usage, for classifying future or unknown objects:

- Estimate **accuracy** of the model:
  - The known label of **test samples** is compared with the result from the model.
  - **Accuracy rate** is the percentage of test-set samples that are correctly classified by the model.
  - Test set is independent of training set (otherwise overfitting).
- If the accuracy is acceptable, **use the model** to classify data tuples whose class labels are not known.

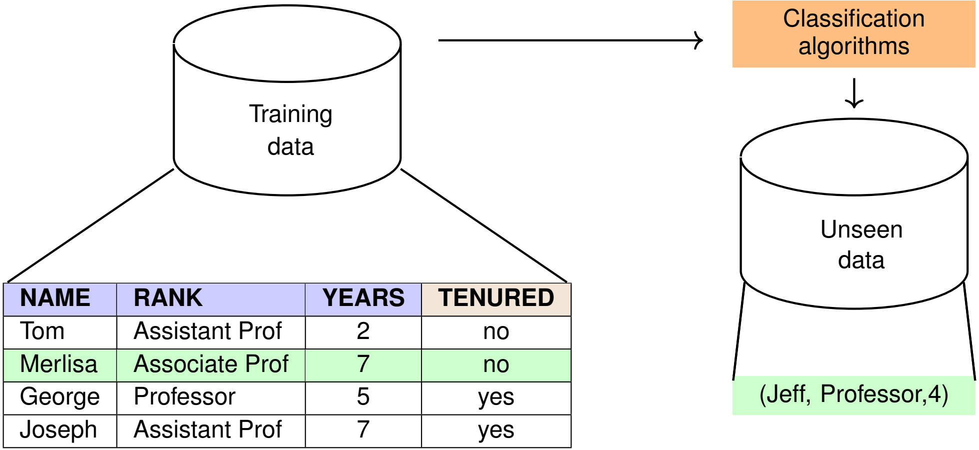
Classification – a two-step process



NAME	RANK	YEARS	TENURED
Mike	Assistant Prof	3	no
Mary	Assistant Prof	7	yes
Bill	Professor	2	yes
Jim	Associate Prof	7	yes
Dave	Assistant Prof	6	no
Anne	Associate Prof	3	no

if RANK ='Professor'
or YEARS >6
then TENURED ='yes'

Process (II): using the model in prediction



NAME	RANK	YEARS	TENURED
Tom	Assistant Prof	2	no
Merlisa	Associate Prof	7	no
George	Professor	5	yes
Joseph	Assistant Prof	7	yes

Chapter VI: Classification

Classification: basic concepts.

Decision-tree induction.

Bayes classification methods.

Rule-based classification.

Model evaluation and selection.

Techniques to improve classification accuracy: ensemble methods.

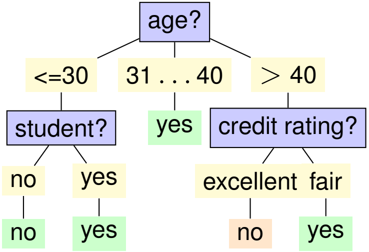
Summary.

Decision-tree induction: an example

Training dataset: buys_computer.

The dataset follows an example of Quinlan's ID3 (playing tennis).

Resulting tree:



age	income	student	credit_rating	buys_coputer
≤ 30	high	no	fair	no
≤ 30	high	no	excellent	no
$31 \ldots 40$	high	no	fair	yes
> 40	medium	no	fair	yes
> 40	low	yes	fair	yes
> 40	low	yes	excellent	no
$31 \ldots 40$	low	yes	excellent	yes
≤ 30	medium	no	fair	no
≤ 30	low	no	fair	yes
> 40	medium	yes	fair	yes
≤ 30	medium	yes	excellent	yes
$31 \ldots 40$	medium	no	excellent	yes
$31 \ldots 40$	high	yes	fair	yes
> 40	medium	no	excellent	no

Algorithm for decision-tree induction

Basic algorithm (a greedy algorithm):

- Tree is constructed in a **top-down recursive divide-and-conquer manner.**
- Attributes are categorical.
  - If not: discretize in advance.
- At start, all the training examples are at the root.
- Examples are **partitioned recursively** based on selected attributes.
- Test attributes are selected on the basis of a heuristic or statistical measure.
  - E.g. information gain – see on the next slide.

Conditions for stopping partitioning:

- All samples for a given node belong to the same class.
- There are no remaining attributes for further partitioning.
  - Majority voting is employed for classifying the leaf.
- There are no samples left (i.e. partition for particular value is empty).

Attribute-selection measure: information gain (ID3/C4.5)

Select the attribute with the highest information gain.

Let p_i be the probability that an arbitrary tuple in D belings to class C_i, estimated by $\frac{|C_i|}{|D|}$, such that $1 \leq i \leq m$.

Expected information (entropy)needed to classify a tuple in D:

$$\text{Info}(D) = -\sum_{i=1}^{m} p_i \log_2(p_i). \tag{1}$$

Information needed (after using attribute A to split D into v partitions) to classify D:

$$\text{Info}_A(D) = \sum_{j=1}^{v} \left(\frac{|D_j|}{|D|} \text{Info}(D_j) \right). \tag{2}$$

Information gained by branching on A:

$$\text{Gain}(A) = \text{Info}(D) - \text{Info}_A(D). \tag{3}$$

Attribute selection: information gain

Class P: buys_computer = "yes"

Class N: buys_computer = "no"

$$\text{Info}(D) = I(9,5) = -\frac{9}{14}\log_2\left(\frac{9}{14}\right) - \frac{5}{14}\log_2\left(\frac{5}{14}\right) = 0.94$$

age	p	n	$I(p,n)$
≤ 30	2	3	0.971
$31\ldots40$	4	0	0
> 40	3	2	0.971

Similarly,

$$\begin{aligned} \text{Gain}(\texttt{income}) &= 0.029, \\ \text{Gain}(\texttt{student}) &= 0.151, \\ \text{Gain}(\texttt{credit\_rating}) &= 0.048. \end{aligned}$$

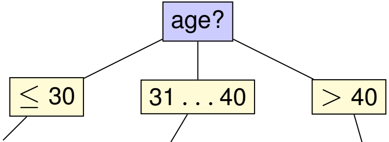
$$\text{Info}_{\text{age}}(D) = \frac{5}{14}I(2,3) + \frac{4}{14}I(4,0) + \frac{5}{14}I(3,2) = 0.694.$$

$\frac{5}{14}I(2,3)$ means "age ≤ 30" has 5 out of 14 samples, with 2 yes'es and 3 no's. Hence,

$$\text{Gain}(\text{age}) = \text{Info}(D) - \text{Info}_{\text{age}}(D) = 0.246.$$

age	income	student	credit_rating	buys_computer
≤ 30	high	no	fair	no
≤ 30	high	no	excellent	no
$31\ldots40$	high	no	fair	yes
> 40	medium	no	fair	yes
> 40	low	yes	fair	yes
> 40	low	yes	excellent	no
$31\ldots40$	low	yes	excellent	yes
≤ 30	medium	no	fair	no
> 40	medium	yes	fair	yes
≤ 30	medium	yes	excellent	yes
$31\ldots40$	medium	no	fair	yes
$31\ldots40$	high	yes	fair	yes
> 40	medium	no	excellent	no

Partitioning in the example



income	student	credit_rating	buys_computer
high	no	fair	no
high	no	excellent	no
medium	no	fair	no
low	yes	fair	yes
medium	yes	excellent	yes

income	student	credit_rating	
high	no	fair	
low	yes	excellent	yes
medium	no	excellent	yes
high	yes	fair	yes

income	student	credit_rating	buys_computer
medium	no	fair	yes
low	yes	fair	yes
low	yes	excellent	no
medium	yes	fair	yes
medium	no	excellent	no

Computing information gain for continuous-valued attributes

Let attribute A be a continuous-valued attribute.

Must determine the best split point for A.

- Sort the values of A in increasing order.
- Typically, the midpoint between each pair of adjacent values is considered as a possible split point.
  - $\frac{a_i+a_{i+1}}{2}$ is the midpoint between the values of a_i and a_{i+1}.
- The point with the minimum expected information requirement for A is selected as the split point for A.

Split:

- D_1 is the set of tuples in D satisfying $A \leq$ split point, and D_2 is the set of tuples in D satisfying $A >$ split point.

So to say: Discretization as you go along.

- For this particular purpose.

Gain ratio for attribute selection (C4.5)

Information-gain measure is biased towards attributes with a large number of values. C4.5 (a successor of ID3) uses gain ratio to overcome the problem (normalization to information gain):

$$\mathrm{SplitInfo}_A(D) = -\sum_{j=1}^{v} \frac{|D_j|}{|D|} \log_2\left(\frac{|D_j|}{|D|}\right), \tag{4}$$

$$\mathrm{GainRatio}(A) = \frac{\mathrm{Gain}(A)}{\mathrm{SplitInfo}_A(D)}. \tag{5}$$

Example:

$$\mathrm{SplitInfo}_{\mathtt{income}}(D) = -\frac{4}{14}\log_2\left(\frac{4}{14}\right) - \frac{6}{14}\log_2\left(\frac{6}{14}\right) - \frac{4}{14}\log_2\left(\frac{4}{14}\right) = 1.557, \tag{6}$$

$$\mathrm{GainRatio}(\mathtt{income}) = \frac{0.029}{1.557} = 0.019. \tag{7}$$

The attribute with the maximum gain ratio is selected as the splitting attribute.

Gini index

Corrado Gini (1884 – 1965).

- Italian statistician and sociologist.

Also called Gini coefficient.

Measures statistical dispersion.

- Zero expresses perfect equality where all values are the same.
- One expresses maximal inequality among values.

Based on the Lorenz curve.

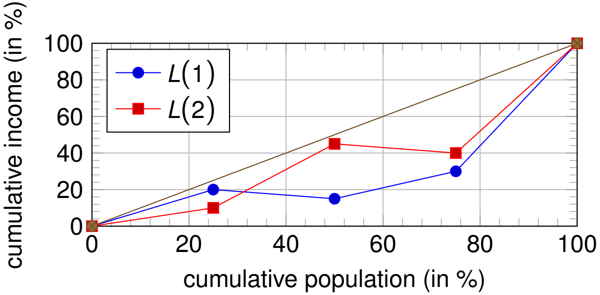
- Plots the proportion of the total sum of values (y-axis) that is cumulatively assigned to the bottom $x\%$ of the population.
- Line at 45 degrees thus represents perfect equality of value distribution.

Gini coefficient then is . . .

- . . . the ratio of the area that lies between the line of equality and the Lorenz curve over the total area under the line of equality.

Gini index (II)

Example: Distribution of incomes.



Gini index (CART, IBM IntelligentMiner)

If a dataset D contains examples from n classes, Gini index gini(D) is defined as:

$$\mathrm{gini}(D) = 1 - \sum_{j=1}^{n} p_j^2, \tag{8}$$

where p_j is the relative frequency of class j in D.

If a dataset D is split on A into two subsets D_1 and D_2, the Gini index gini(D) is defined as

$$\mathrm{gini}_A(D) = \frac{|D_1|}{|D|}\mathrm{gini}(D_1) + \frac{|D_2|}{|D|}\mathrm{gini}(D_2). \tag{9}$$

Reduction in impurity:

$$\Delta\mathrm{gini}_A(A) = \mathrm{gini}(D) - \mathrm{gini}_A(D). \tag{10}$$

The attribute A provides the smallest gini$_A(D)$ (or the largest reduction in impurity) is chosen to split the node.

Need to enumerate all the possible splitting points for each attribute.

Computation of Gini index

Example:

D has 9 tuples in buys_computer = "yes" and 5 in "no", thus

$$\text{gini}(D) = 1 - \left(\frac{9}{14}\right)^2 - \left(\frac{5}{14}\right) = 0.459. \tag{11}$$

Suppose the attribute `income` partitions D into 10 in D_1 : $\{\texttt{low},\texttt{medium}\}$ and 4 in D_2 : $\{\texttt{high}\}$:

$$\text{gini}(D|_{D[\texttt{income}]='' medium'','' low''}) \tag{12}$$

$$= \left(\frac{10}{14}\right)\text{gini}(D_1) + \frac{4}{14}\text{gini}(D_2) \tag{13}$$

$$= \frac{10}{14}\left(1 - \left(\frac{7}{10}\right)^2 - \left(\frac{3}{10}\right)^2\right) + \frac{4}{14}\left(1 - \left(\frac{2}{4}\right)^2 - \left(\frac{2}{4}\right)^2\right) = \tag{14}$$

$$= 0.443 = \text{gini}(D|_{D[\texttt{income}]='' high''}). \tag{15}$$

Computation of Gini index (II)

Example (cont.):

$\mathrm{gini}(D|_{D[\mathtt{income}]=''low'',''high''}) = 0.458,$
$\mathrm{gini}(D|_{D[\mathtt{income}]=''medium'',''high''}) = 0.450.$
Thus, split on the {"low","medium"} and {"high"}, since it has the lowest gini index.

All attributes are assumed continuous-valued.

May need other tools, e.g. clustering, to get the possible split values.

Can be modified for categorical attributes.

Computation of Gini index (II)

The three measures, in general, return good results, but

- **Information gain:**
  - Biased towards multi-valued attributes.
- **Gain ratio:**
  - Tends to prefer unbalanced splits in which one partition is much smaller than the others.
- **Gini index:**
  - Biased to multi-valued attributes.
  - Has difficulty when number of classes is large.
  - Tends to favor tests that result in equal-sized partitions and purity in both partitions.

Other attribute-selection measures

CHAID:

A popular decision-tree algorithm, measure based on χ^2 test for independence.

C-SEP:

Performs better than information gain and Gini index in certain cases.

G-statistic:

Has a close approximation to χ^2 distribution.

MDL (Minimal Description Length) principle:

I.e. the simplest solution is preferred.

The best tree is the one that requires the fewest number of bits to both (1) encode the tree and (2) encode the exceptions to the tree.

Multivariate splits:

Partitioning based on multiple variable combinations.

CART: finds multivariate splits based on a linear combination of attributes.

Which attribute-selection measure is the best?

Most give good results, none is significantly superior to others.

Overfitting and tree pruning

Overfitting: An induced tree may overfit the training data.

- Too many branches, some may reflect anomalies due to noise or outliers.
- Poor accuracy for unseen samples.

Two approaches to avoid overfitting:

- **Prepruning:**
  - Halt tree construction early.
  - Do not split a node, if this would result in the goodness measure falling below a threshold.
  - Difficult to choose an appropriate threshold.
- **Postpruning:**
  - Remove branches from a "fully grown" tree.
  - Get a sequence of progressively pruned trees.
  - Use a set of data different from the training data to decide which is the "best pruned tree."

Enhancements to Basic Decision-Tree Induction

Allow for continuous-valued attributes.

Dynamically define new discrete-valued attributes that partition the values of continuous-valued attributes into a discrete set of intervals.

Handle missing attribute values.

Assign the most common value of the attribute.

Assign probability to each of the possible values.

Attribute construction.

Create new attributes based on existing ones that are sparsely represented.

This reduces fragmentation, repetition, and replication.

Classification in large databases

Classification – a classical problem extensively studied by statisticians and machine-learning researchers.

Scalability:

- Classifying datasets with millions of examples and hundreds of attributes with reasonable speed.

Why is decision-tree induction popular?

- Relatively fast learning speed (compared to other classification methods).
- Convertible to simple and easy-to-understand classification rules.
- Can use SQL queries for accessing databases.
- Classification accuracy comparable with other methods.

RainForest (Gehrke, Ramakrishnan & Ganti, VLDB'98).

- Builds an AVC-list (attribute, value, class label).

Scalability framework for RainForest

Separates the scalability aspects from the criteria that determine the quality of the tree.

Builds an AVC-list:.

AVC (Attribute, Value, Class_label).

AVC-set (of an attribute X):

Projection of training dataset onto the attribute *X* and class label where counts of individual class label are aggregated.

AVC-group (of a node n):

Set of AVC-sets of all predictor attributes at the node *n*.

RainForest: training set and its AVC-sets

age	income	student	credit_rating	buys_computer
≤ 30	high	no	fair	no
≤ 30	high	no	excellent	no
$31 \ldots 40$	high	no	fair	yes
> 40	medium	no	fair	yes
> 40	low	yes	fair	yes
> 40	low	yes	excellent	no
$31 \ldots 40$	low	yes	excellent	yes
≤ 30	medium	no	fair	no
≤ 30	low	yes	fair	yes
> 40	medium	yes	fair	yes
≤ 30	medium	yes	excellent	yes
$31 \ldots 40$	medium	no	excellent	yes
$31 \ldots 40$	high	yes	fair	yes
> 40	medium	no	excellent	no

AVC-set on age:

age	yes	no
≤ 30	2	3
$31 \ldots 40$	4	0
> 40	3	2

AVC-set on income:

income	yes	no
high	2	2
medium	4	2
low	3	1

RainForest: training set and its AVC-sets (II)

age	income	student	credit_rating	buys_computer
≤ 30	high	no	fair	no
≤ 30	high	no	excellent	no
$31 \ldots 40$	high	no	fair	yes
> 40	medium	no	fair	yes
> 40	low	yes	fair	yes
> 40	low	yes	excellent	no
$31 \ldots 40$	low	yes	excellent	yes
≤ 30	medium	no	fair	no
≤ 30	low	yes	fair	yes
> 40	medium	yes	fair	yes
≤ 30	medium	yes	excellent	yes
$31 \ldots 40$	medium	no	excellent	yes
$31 \ldots 40$	high	yes	fair	yes
> 40	medium	no	excellent	no

AVC-set on student:

student	yes	no
yes	6	1
no	3	4

AVC-set on credit_rating:

credit_rating	yes	no
fair	6	2
excellent	3	3

BOAT (Bootstrapped optimistic algorithm for tree construction)

Use a statistical technique called bootstrapping to create several smaller samples (subsets), each fitting in memory.

- See on the subsequent slides.

Each subset is used to create a tree, resulting in several trees.

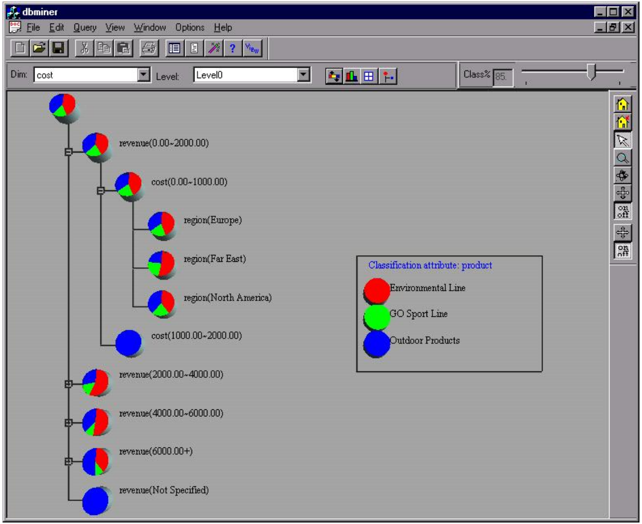
These trees are examined and used to construct a new tree T'.

- It turns out that T' is very close to the tree that would be generated using the whole data set together.

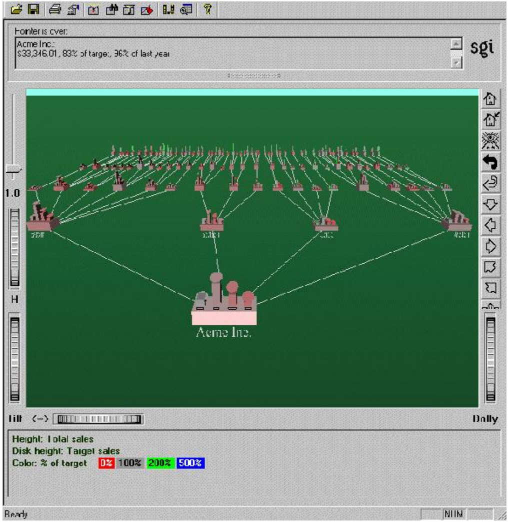
Advantages:

- Requires only two scans of DB.
- An incremental algorithm:
  - Take insertions and deletions of training data and update the decision tree.

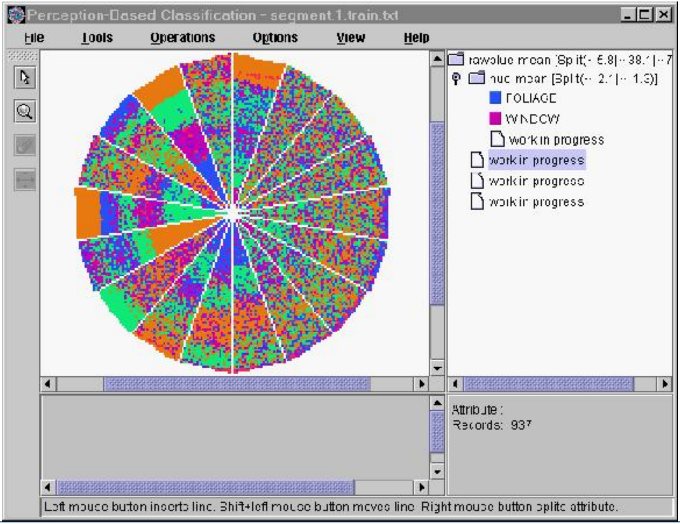
Presentation of classification results



Visualization of a decision tree in SGI/MineSet 3.0



Interactive visual mining by perception-based classification (PBC)



Chapter VI: Classification

Bayesian classification: why?

A statistical classifier:

Performs probabilistic prediction, i.e. predicts class-membership probabilities.

Foundation: Bayes' Theorem.

Performance:

A simple Bayesian classifier (naïve Bayesian classifier) has performance comparable with decision tree and selected neural-network classifiers.

Incremental:

Each training example can incrementally increase/decrease the probability that a hypothesis is correct–prior knowledge can be combined with observed data.

Standard:

Even when Bayesian methods are computationally intractable, they can provide a standard of optimal decision making against which other methods can be measured.

Bayes' Theorem: basics

Let X be a data sample ("evidence").

The class label shall be unknown.

Let C_i be the hypothesis that X belongs to class i.

Classification is to determine $P(C_i|X)$:

Posteriori probability: the probability that the hypothesis holds given the observed data sample X.

$P(C_i)$:

Prior probability: the initial probability.

E.g. X will buy computer, regardless of age, income, ...

$P(X)$:

Probability that sample data is observed.

$P(X|C_j)$:

Likelihood: the probability of observing the sample X given that the hypothesis holds.

E.g. given that X buys computer, the probability that X is 31 ... 40, medium income.

Bayes' Theorem (II)

Given training data X, the posteriori probability $P(C_i|X)$ of a hypothesis C_i follows from the Bayes' Theorem:

$$P(C_i|X) = \frac{P(X|C_i)P(C_i)}{P(X)}. \tag{16}$$

Predicts that X belongs to C_i iff the probability $P(C_i|X)$ is the highest among all the $P(C_k|X)$ for all k classes.

Practical difficulty:

- Requires initial knowledge of many probabilities.
- Significant computational cost.

Towards naïve Bayesian classifier

Let D be a training set of tuples and their associated class labels.

Each tuple is represented by an n-dimensional attribute $X = (x_1, x_2, \ldots, x_n)$.

Supose there are m classes $C_1, C_2, \ldots, C_m$.

Classification is to derive the maximum posteriori probability.

i.e. the maximal $P(C_i|X)$.

This can be derived from Bayes' Theorem:

$$P(C_i|X) = \frac{P(X|C_i)P(C_i)}{P(X)}. \quad (17)$$

Since $P(X)$ is constant for all classes, we must maximize only:

$$P(X|C_i)P(C_i). \quad (18)$$

Derivation of naïve Bayes classifier

A simplifying assumption: attributes are conditionally independent.

I.e. no dependence relation between attributes (which is "naïve").

$$P(X|C_i) = \prod_{k=1}^{n} P(x_k|C_i) = P(x_1|C_i)P(x_2|C_i)\cdots P(x_k|C_i).$$

This greatly reduces the computation cost:

Only count the class distribution.

If A_k is categorical,

$P(x_k|C_i)$is the number of tuples in C_i having value x_k for A_k divided by $|C_{i,D}|$ (the number of tuples of C_i in D).

If A_k is continuous-valued,

$P(x_k|C_i)$ is usually computed based on Gaussian distribution with a mean μ and standard deviation σ:

$$G(x, \mu, \sigma) = \frac{1}{\sqrt{2\pi}\sigma} e^{\frac{(x-\mu)^2}{2\sigma^2}},$$

and $P(x_k|C_i)$ is $P(x_k|C_i) = G(x_k, \mu_{C_i}, \sigma_{C_i})$.

Naïve Bayesian cellcolormake dataset

Classes:

C_1: `buys_computer` = "yes".

C_2: `buys_computer` = "no".

Data sample:

$X = (\texttt{age} \leq 30,$
$\texttt{income} = ''\textit{medium}'',$
$\texttt{student} = ''\textit{yes}'',$
$\texttt{credit\_rating} = ''\textit{fair}'').$

age	income	student	credit_rating	buys_computer
≤ 30	high	no	fair	no
≤ 30	high	no	excellent	no
$31 \ldots 40$	high	no	fair	yes
> 40	medium	no	fair	yes
> 40	low	yes	fair	yes
> 40	low	yes	excellent	no
$31 \ldots 40$	low	yes	excellent	yes
≤ 30	medium	no	fair	no
> 40	medium	yes	fair	yes
≤ 30	medium	yes	excellent	yes
$31 \ldots 40$	medium	no	fair	yes
$31 \ldots 40$	high	yes	fair	yes
> 40	medium	no	excellent	no

Naïve Bayesian classifier: an example

$P(C_i)$:

$$P(\texttt{buys\_computer} = "yes") = \frac{9}{14} = 0.643.$$
$$P(\texttt{buys\_computer} = "no") = \frac{5}{14} = 0.357.$$

$$X = (\texttt{age} \leq 30, \texttt{income} = "medium", \texttt{student} = "yes", \texttt{credit\_rating} = "fair").$$

Compute $P(X|C_i)$ for each class:

$$P(\texttt{age} \leq 30 | \texttt{buys\_computer} = "yes") = \frac{2}{9} = 0.222.$$
$$P(\texttt{age} \leq 30 | \texttt{buys\_computer} = "no") = \frac{3}{5} = 0.6.$$
$$P(\texttt{income} = "medium" | \texttt{buys\_computer} = "yes") = \frac{4}{9} = 0.444.$$
$$P(\texttt{income} = "medium" | \texttt{buys\_computer} = "no") = \frac{2}{5} = 0.4.$$
$$P(\texttt{student} = "yes" | \texttt{buys\_computer} = "yes") = \frac{6}{9} = 0.667.$$
$$P(\texttt{student} = "yes" | \texttt{buys\_computer} = "no") = \frac{1}{5} = 0.2.$$
$$P(\texttt{credit\_rating} = "fair" | \texttt{buys\_computer} = "yes") = \frac{6}{9} = 0.667.$$
$$P(\texttt{credit\_rating} = "fair" | \texttt{buys\_computer} = "no") = \frac{2}{5} = 0.4.$$

Naïve Bayesian classifier: an example (II)

$P(C_i)$:

$$P(X|\texttt{buys\_computer} = \text{"}yes\text{"}) = 0.222 \cdot 0.444 \cdot 0.667 \cdot 0.667 = 0.044.$$
$$P(X|\texttt{buys\_computer} = \text{"}no\text{"}) = 0.6 \cdot 0.4 \cdot 0.2 \cdot 0.4 = 0.019.$$

$P(X|C_i) \cdot P(C_i)$:

$$P(X|\texttt{buys\_computer} = \text{"}yes\text{"}) \cdot P(\texttt{buys\_computer} = \text{"}yes\text{"}) = 0.028.$$
$$P(X|\texttt{buys\_computer} = \text{"}no\text{"}) \cdot P(\texttt{buys\_computer} = \text{"}no\text{"}) = 0.007.$$

Therefore, X belongs to class C_1 (buys_computer = "yes").

Avoiding the zero-probability problem

Naïve Bayesian prediction requires each conditional probability to be non-zero.

Otherwise, the predicted probability will be zero.

$$P(X|C_i) = \prod_{k=1}^{n} P(x_k|C_i). \tag{19}$$

Example:

Suppose a dataset with 1000 tuples, `income` = "low" (0), `income` = "medium" (990), and `income` = "high" (10).

Use Laplacian correction (or Laplacian estimator):

Add 1 to each case:

$$P(\texttt{income} = "low") = \frac{1}{1003}.$$
$$P(\texttt{income} = "medium") = \frac{991}{1003}.$$
$$P(\texttt{income} = "high") = \frac{11}{1003}.$$

The "corrected" probability estimates are close to their "uncorrected" counterparts.

Naïve Bayesian classifier: comments

Advantages:

- Easy to implement.
- Good results obtained in most of the cases.

Disadvantages:

- Assumption: class conditional independence, therefore loss of accuracy.
- Practically, **dependencies** exist among variables.
  - E.g. hospital patients:
    - Profile: age, family history, etc.
    - Symptoms: fever, cough, etc.
    - Disease: lung cancer, diabetes, etc.
  - Cannot be modeled by naïve Bayesian classifier.

How to deal with these dependencies?

- Bayesian belief networks (see textbook).

Chapter VI: Classification

Classification: basic concepts.

Decision-tree induction.

Bayes classification methods.

Rule-based classification.

Model evaluation and selection.

Techniques to improve classification accuracy: ensemble methods.

Summary.

Using IF-THEN rules for classification

Represent the knowledge in the form of IF-THEN rules.

E.g. if `age` $\leq$ 30 AND `student` = "yes" THEN buys_computer = "yes".
Readable.

Rule antecedent/precondition vs. rule consequent.

Assessment of a rule R: coverage and accuracy.

$n_{\text{covers}} = \#$ of tuples covered by R (antecedent if true).
$n_{\text{correct}} = \#$ of tuples correctly classified by R.
$\text{coverage}(R) = \frac{n_{\text{covers}}}{|D|}$ with D training data set.
$\text{accuracy}(R) = \frac{n_{\text{correct}}}{n_{\text{covers}}}$.

Using IF-THEN rules for classification (II)

If more than one rule are triggered, need conflict resolution.

- **Size ordering:**
  - Assign the highest priority to the triggered rule that has the "toughest" requirement (i.e., the most attribute tests).
- **Class-based ordering:**
  - Decreasing order of prevalence or misclassification cost per class.
- **Rule-based ordering** (decision list):
  - Rules are organized into one long priority list, according to some measure of rule quality, or by experts.

Rule extraction from a decision tree

Rules are easier to understand than large trees.

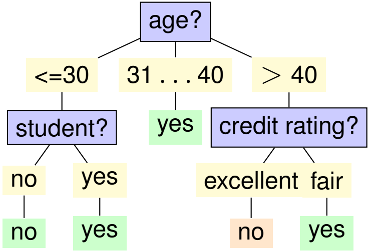
Rule can be created for each path from the root to a leaf.

The leaf holds the class prediction.

Each attribute-value pair along the path forms a conjunction:

Example:

IF $\texttt{age} \leq 30$ AND $\texttt{student}$ = "no"
THEN $\texttt{buys\_computer}$ = "no".
IF $\texttt{age} \leq 30$ AND $\texttt{student}$ = "yes"
THEN $\texttt{buys\_computer}$ = "yes".
IF $\texttt{age}$ 31 . . . 40 THEN
$\texttt{buys\_computer}$ = "yes".



Rule induction: sequential covering method

Sequential covering algorithm:

Extracts rules directly from training data.

Typical sequential covering algorithms:

FOIL, AQ, CN2, RIPPER.

Rules are learned sequentially.

Each rule for a given class C_i will cover many tuples of C_i, but none (or few) of the tuples of other classes.

Steps:

Rules are learned one at a time.

Each time a rule is learned, the tuples covered by the rule are removed.

The process repeats on the remaining tuples unless termination condition, e.g. when no more training examples left or when the quality of a rule returned is below a user-specified threshold.

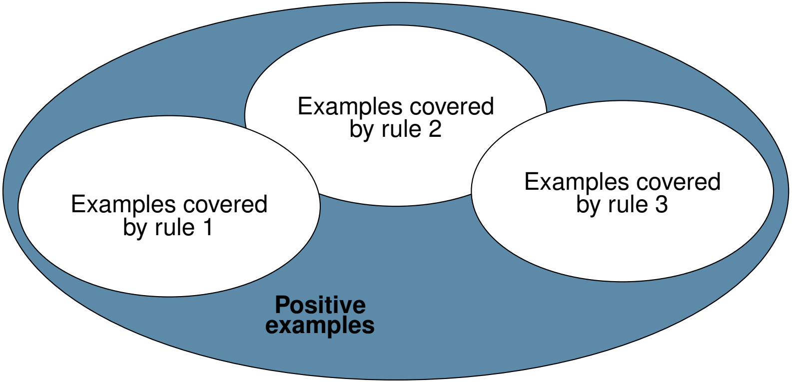
Compare with decision-tree induction:

That was learning a set of rules simultaneously.

Sequential covering algorithm

While (enough target tuples left):

generate a rule;
remove positive target tuples satisfying this rule;



Sequential covering algorithm

To generate a rule:

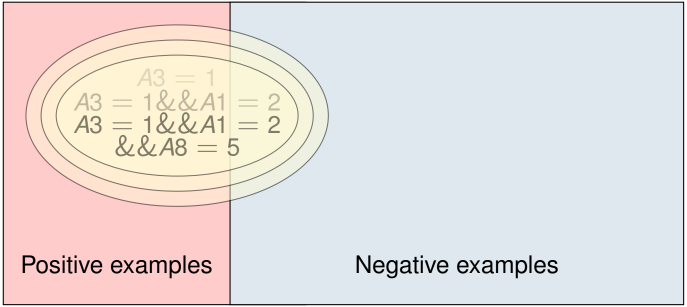
while(true:)

find the best predicate *p* (attribute = value);

if `FOIL_Gain`(p) > threshold

then add *p* to current rule;

else break;



Sequential covering algorithm

Start with the most general rule possible:

Condition = empty.

Add new attributes by adopting a greedy depth-first strategy.

Pick the one that improves the rule quality most.

Current rule R: IF condition THEN class = c.

New rule R': IF condition' THEN class = c,

pos/neg are $\#$ of positive/negative tuples covered by R.

Rule-quality measures.

Must consider both coverage and accuracy.

`FOIL_Gain` (from `FOIL` – First-Order Inductive Learner):

$$\mathrm{FOIL\_Gain} = \mathrm{pos}' \left(\log_2 \frac{\mathrm{pos}'}{\mathrm{pos}' + \mathrm{neg}'} - \log_2 \frac{\mathrm{pos}}{\mathrm{pos} + \mathrm{neg}} \right). \tag{20}$$

Favors rules that have high accuracy and cover many positive tuples.

Rule pruning

Danger of overfitting.

Removing a conjunct (attribute test),

if pruned version of rule has greater quality,
assessed on an independent set of test tuples (called "pruning set").

FOIL uses:

$$\text{FOIL\_Prune}(R) = \frac{\text{pos} - \text{neg}}{\text{pos} + \text{neg}}. \tag{21}$$

If FOIL_Prune is higher for the pruned version of R, prune R.

Chapter VI: Classification

Model evaluation and selection

Evaluation metrics:

- How can we measure accuracy?
- Other metrics to consider?

Use test set of class-labeled tuples instead of training set when assessing accuracy.

Methods for estimating a classifier's accuracy:

- Holdout method, random subsampling.
- Cross-validation.
- Bootstrap.

Comparing classifiers:

- Confidence intervals.
- Cost-benefit analysis and ROC curves.

Model evaluation and selection

Confusion Matrix:

Actual class/predicted class:	C_1	$\neg C_1$
C_1	**True positives (TP)**	**True negatives (TN)**
$\neg C_1$	**False positives (FP)**	**False negatives (FN)**

Example:

Actual class/predicted class:	buys_computer = yes	buys_computer = no	Total
buys_computer = yes	**6954**	**46**	7000
buys_computer = no	**412**	**2588**	3000
Total	7366	2634	10000

Given M classes, an entry $C_{ij}^{(m)}$ in an $M \times M$ confusion matrix indicates # of tuples in class i that were labeled by the classifier as class j.

May have extra rows/columns to provide totals.

Classifier-evaluation metrics: accuracy, error rate, sensitivity and specificity

A/P	C	¬ C	
C	**TP**	**FN**	**P**
¬ C	**FP**	**TN**	**N**
	P'	**N'**	**All**

Classifier accuracy, or recognition rate:

Percentage of test set tuples that are correctly classified.

Accuracy $= \frac{\text{TP}+\text{TN}}{\text{All}}$.

Error rate:

1 $-$ accuracy, or

Error rate $= \frac{\text{FP}+\text{FN}}{\text{All}}$.

Class-imbalance problem:

One class may be rare e.g. fraud, or HIV-positive.

Significant majority of the negative class and minority of the positive class

Sensitivity: True-positive recognition rate. Sensitivity $= \frac{TP}{P}$.

Specificity: False-negative recognition rate. Specificity $= \frac{TN}{N}$.

Classifier-evaluation metrics: precision, recall, and F-measures

Precision:

Exactness – the % of tuples that are actually positive in those that the classifier labeled as positive: $\frac{\text{TP}}{\text{TP}+\text{FP}}$.

Recall:

Completeness – the % of tuples that the classifier labeled as positive in all positive tuples: $\frac{\text{TP}}{\text{TP}+\text{FN}}$.
Perfect score is 1.0.

Inverse relationship between precision and recall.

F-measure (F_1 or F-score):

Gives equal weight to precision and recall: $F = \frac{2\cdot\text{precision}\cdot\text{recall}}{\text{precision}+\text{recall}}$.

F_β weighted measure of precision and recall:

Assigns β times as much weight to recall as to precision: $F_\beta = \frac{(1+\beta)^2\cdot\text{precision}\cdot\text{recall}}{\beta^2\cdot\text{precision}+\text{recall}}$.

Classifier-evaluation metrics: precision, recall, and F-measures

Actual class/predicted class	cancer = yes	cancer = no	Total	Recognition (%)
cancer = yes	**90**	**210**	300	30.00 (sensitivity)
cancer = no	**140**	**9560**	9700	98.56 (specificity)
Total	230	9770	10000	96.40 (accuracy)

Precision $= \frac{90}{230} = 39.13\%.$

Recall $= \frac{90}{300} = 30.00\%.$

F-measure = $2 \cdot 0.3913 \cdot \frac{0.3}{0.3913+0.3} = 33.96\%.$

Classifier-evaluation metrics: holdout & cross-validation methods

Holdout method.

Given data is randomly partitioned into two independent sets:

- **Training set** (e.g. $2/3$) for model construction.
- **Test set** (e.g. $1/3$) for accuracy estimation.

Random sampling: a variation of holdout.

- Repeat holdout k times, accuracy = avg. of the accuracies obtained.

Cross-validation (k-fold, where $k = 10$ is most popular).

Randomly partition the data into k mutually exclusive subsets, each approximately equal size.
At i-th iteration, use D_i as test set and the others as training set.
Leave-one-out: k folds, where $k = \#$ of tuples; for small-sized data.
Stratified cross-validation: Folds are stratified so that class distribution in each fold is approx.
The same as that in the initial data.

Evaluating classifier accuracy: bootstrap

Bootstrap.

Works well with small data sets.

Samples the given training tuples uniformly with replacement.

I.e. each time a tuple is selected, it is equally likely to be selected again and re-added to the training set.

Several bootstrap methods, and a common one is .632 **bootstrap.**

Data set with d tuples sampled d times, with replacement, resulting in a training set of d samples.

The data tuples that did not make it into the training set end up forming the test set.

About 63.2% of the original data end up in the bootstrap, and the remaining 36.8% form the test set (since $(1 - \frac{1}{d})^d \approx e^{-1} = 0.368$).

Repeat the sampling procedure k times; overall accuracy of the model:

$$\mathrm{Acc}(M) = \frac{1}{k}\sum_{i=1}^{k} 0.632 \cdot \mathrm{Acc}(M_i)_{\text{test\_set}} + 0.368 \cdot \mathrm{Acc}(M_i)_{\text{train\_set}}. \tag{22}$$

Evaluating classifier accuracy: bootstrap

Suppose we have 2 classifiers, M_1 and M_2, which one is better?

Use 10-fold cross-validation to obtain $\overline{\text{err}}(M_1)$ and $\overline{\text{err}}(M_2)$.

Recall: error rate is $1 - \text{accuracy}(M)$.

Mean error rates:

Just estimates of error on the true population of future data cases.

What if the difference between the 2 error rates is just attributed to chance?

Use a test of statistical significance.
Obtain confidence limits for our error estimates.

Evaluating classifier accuracy: null hypothesis

Perform 10-fold cross-validation.

10 times.

Assume samples follow a t-distribution with $k - 1$ degrees of freedom.

Here, $k = 10$.

Use t-test

Student's t-test.

Null hypothesis:

$M_1 \& M_2$ are the same.

If we can reject the null hypothesis, then

Conclude that difference between $M_1 \& M_2$ is statistically significant.
Obtain confidence limits for our error estimates.

Estimating confidence intervals: *t*-test

If only one test set available: pairwise comparison:

For i-th round of 10-fold cross-validation, the same cross partitioning is used to obtain $\text{err}(M_1)_i$ and $\text{err}(M_2)_i$.

Average over 10 rounds to get $\overline{\text{err}}(M_1)_i$ and $\overline{\text{err}}(M_2)_i$.

t-test computes t-statistic with $k-1$ degrees of freedom:

$$t = \frac{\overline{\text{err}}(M_1) - \overline{\text{err}}(M_2)}{\sqrt{\frac{\text{var}(M_1 - M_2)}{k}}}, \tag{23}$$

where

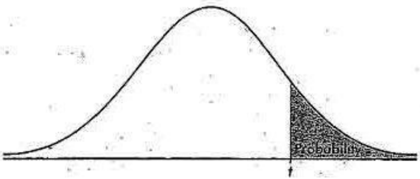
$$\text{var}(M_1 - M_2) = \frac{1}{k}\sum_{i=1}^{k}\left[\overline{\text{err}}(M_1)_i - \overline{\text{err}}(M_2)_i - (\overline{\text{err}}(M_1)_i - \overline{\text{err}}(M_2)_i)\right]^2. \tag{24}$$

If two test sets available: use nonpaired *t*-test:

$$\text{var}(M_1 - M_2) = \sqrt{\frac{\text{var}(M_1)}{k_1} + \frac{\text{var}(M_2)}{k_2}}, \tag{25}$$

where k_1 & k_2 are $\#$ of cross-validation samples used for M_1 & M_2, respectively.

Estimating confidence intervals: table for t-distribution



Symmetrical.

Significance level:

E.g. sig $= 0.05$ or 5% means M_1 & M_2 are significantly different for 95% of population.

Confidence limit: $z = \frac{\text{sig}}{2}$.

TABLE B: t-DISTRIBUTION CRITICAL VALUES

	Tail probability p											
df	.25	.20	.15	.10	.05	.025	.02	.01	.005	.0025	.001	.0005
1	1.000	1.376	1.963	3.078	6.314	12.71	15.89	31.82	63.66	127.3	318.3	636.6
2	.816	1.061	1.386	1.886	2.920	4.303	4.849	6.965	9.925	14.09	22.33	31.60
3	.765	.978	1.250	1.638	2.353	3.182	3.482	4.541	5.841	7.453	10.21	12.92
4	.741	.941	1.190	1.533	2.132	2.776	2.999	3.747	4.604	5.598	7.173	8.610
5	.727	.920	1.156	1.476	2.015	2.571	2.757	3.365	4.032	4.773	5.893	6.869
6	.718	.906	1.134	1.440	1.943	2.447	2.612	3.143	3.707	4.317	5.208	5.959
7	.711	.896	1.119	1.415	1.895	2.365	2.517	2.998	3.499	4.029	4.785	5.408
8	.706	.889	1.108	1.397	1.860	2.306	2.449	2.896	3.355	3.833	4.501	5.041
9	.703	.883	1.100	1.383	1.833	2.262	2.398	2.821	3.250	3.690	4.297	4.781
10	.700	.879	1.093	1.372	1.812	2.228	2.359	2.764	3.169	3.581	4.144	4.587
11	.697	.876	1.088	1.363	1.796	2.201	2.328	2.718	3.106	3.497	4.025	4.437
12	.695	.873	1.083	1.356	1.782	2.179	2.303	2.681	3.055	3.428	3.930	4.318
13	.694	.870	1.079	1.350	1.771	2.160	2.282	2.650	3.012	3.372	3.852	4.221
14	.692	.868	1.076	1.345	1.761	2.145	2.264	2.624	2.977	3.326	3.787	4.140
15	.691	.866	1.074	1.341	1.753	2.131	2.249	2.602	2.947	3.286	3.733	4.073
16	.690	.865	1.071	1.337	1.746	2.120	2.235	2.583	2.921	3.252	3.686	4.015
17	.689	.863	1.069	1.333	1.740	2.110	2.224	2.567	2.898	3.222	3.646	3.965
18	.688	.862	1.067	1.330	1.734	2.101	2.214	2.552	2.878	3.197	3.611	3.922
19	.688	.861	1.066	1.328	1.729	2.093	2.205	2.539	2.861	3.174	3.579	3.883
20	.687	.860	1.064	1.325	1.725	2.086	2.197	2.528	2.845	3.153	3.552	3.850
21	.686	.859	1.063	1.323	1.721	2.080	2.189	2.518	2.831	3.135	3.527	3.819
22	.686	.858	1.061	1.321	1.717	2.074	2.183	2.508	2.819	3.119	3.505	3.792
23	.685	.858	1.060	1.319	1.714	2.069	2.177	2.500	2.807	3.104	3.485	3.768
24	.685	.857	1.059	1.318	1.711	2.064	2.172	2.492	2.797	3.091	3.467	3.745
25	.684	.856	1.058	1.316	1.708	2.060	2.167	2.485	2.787	3.078	3.450	3.725
26	.684	.856	1.058	1.315	1.706	2.056	2.162	2.479	2.779	3.067	3.435	3.707
27	.684	.855	1.057	1.314	1.703	2.052	2.158	2.473	2.771	3.057	3.421	3.690
28	.683	.855	1.056	1.313	1.701	2.048	2.154	2.467	2.763	3.047	3.408	3.674
29	.683	.854	1.055	1.311	1.699	2.045	2.150	2.462	2.756	3.038	3.396	3.659
30	.683	.854	1.055	1.310	1.697	2.042	2.147	2.457	2.750	3.030	3.385	3.646
40	.681	.851	1.050	1.303	1.684	2.021	2.123	2.423	2.704	2.971	3.307	3.551
50	.679	.849	1.047	1.299	1.676	2.009	2.109	2.403	2.678	2.937	3.261	3.496
60	.679	.848	1.045	1.296	1.671	2.000	2.099	2.390	2.660	2.915	3.232	3.460
80	.678	.846	1.043	1.292	1.664	1.990	2.088	2.374	2.639	2.887	3.195	3.416
100	.677	.845	1.042	1.290	1.660	1.984	2.081	2.364	2.626	2.871	3.174	3.390
1000	.675	.842	1.037	1.282	1.646	1.962	2.056	2.330	2.581	2.813	3.098	3.300
∞	.674	.841	1.036	1.282	1.645	1.960	2.054	2.326	2.576	2.807	3.091	3.291
	50%	60%	70%	80%	90%	95%	96%	98%	99%	99.5%	99.8%	99.9%
	Confidence level C											

Estimating confidence intervals: statistical significance

Are M_1 & M_2 significantly different?

- Compute t. Select significance level (e.g. sig = 5%).
- Consult table for t-distribution:
  - Find t value corresponding to $k - 1$ degrees of freedom (here, 9).
- t-distribution is symmetrical:
  - Typically upper % points of distribution shown
  - $\Longrightarrow$ look up value for confidence limit $z = \frac{\text{sig}}{2}$ (here, 0.025).
- If $t > z$ or $t < -z$, then t value lies in rejection region:
  - **Reject null hypothesis** that mean error rates of M_1 & M_2 are equal.
  - Conclude: **statistically significant difference** between M_1 & M_2.
- Otherwise, conclude that any difference is chance.

Model selection: ROC curves

ROC (Receiver Operating Characteristics) curves:

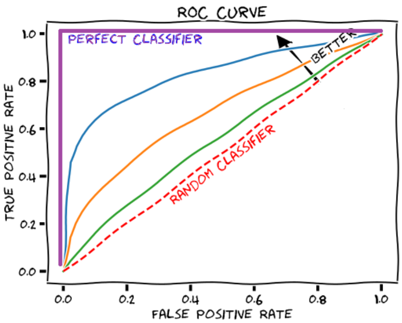
For visual comparison of classification models.
Originated from signal-detection theory.
Shows the trade-off between the true-positive rate and the false-positive rate.

The area under the ROC curve is a measure of the accuracy of the model.

Rank the test tuples in decreasing order:

The one that is most likely to belong to the positive class appears at the top of the list.

The closer to the diagonal line (i.e. the closer the area is to 0.5), the less accurate is the model.



Vertical axis represents TP.

Horizontal axis repr. the FP.

The plot also shows a diagonal line.

A model with perfect accuracy will have an area of 1.0.

Issues after model selection

Accuracy.

Classifier accuracy: predicting class label.

Speed.

Time to construct the model (training time).

Time to use the model (classification/prediction time).

Robustness.

Handling noise and missing values.

Scalability.

Efficiency in disk-resident databases.

Interpretability.

Understanding and insight provided by the model.

Other measures.

E.g. goodness of rules, such as decision-tree size or compactness of classification rules.

Chapter VI: Classification

Thank you for your attention.
Any questions about the sixt chapter?

Ask them now, or again, drop me a line:
`luciano.melodia@fau.de`.