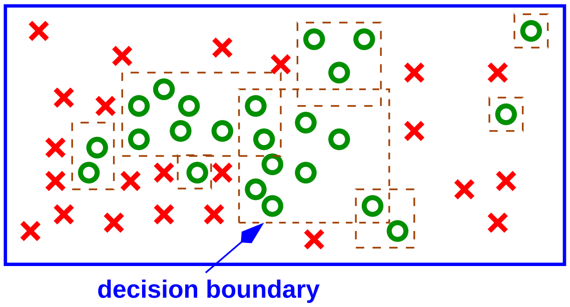
Learning Classifiers

- Instances $\mathbf{x}_i$ in dataset D mapped to feature space:



Classes associated with instances: ✕, ○

- **Classification:**

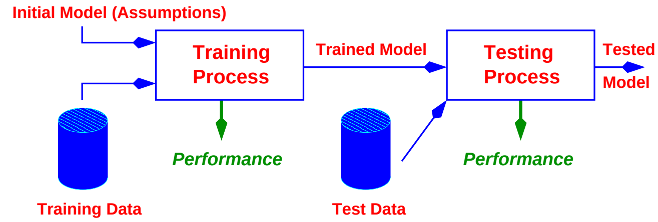
  $f(\mathbf{x}_i) = c \in \{✕, ○\}$

  – with $x_{i,j} \in \{\top, \bot\}$, and f *classifier*

  – dataset D is a multiset

- Objective: learn f (supervised)

Performance



Performance measures:

- TP: True Positive
- TN: True Negative
- FP: False Positive
- FN: False Negative

Note: if $N = |D|$, then N = TP+TN+FP+FN

Confusion matrix:

		Predicted class	
		yes	*no*
Actual class	*yes*	true positive	false negative
	no	false positive	true negative

Performance

Performance measures:

- Success rate σ:

  $$\sigma = \frac{\text{TP} + \text{TN}}{N}$$

- Error rate ϵ: $\epsilon = 1 - \sigma$
- TPR (= recall ρ) True Positive Rate

  TPR = TP/(TP + FN)

- FNR False Negative Rate: FNR = 1 − TPR
- FPR False Positive Rate:

  FPR = FP/(FP + TN)

- TNR True Negative Rate: TNR = 1 − FPR
- Precision π:

  π = TP/(TP + FP)

- F-measure:

  $$F = \frac{2 \cdot \rho \cdot \pi}{\rho + \pi} = \frac{2\text{TP}}{2\text{TP} + \text{FP} + \text{FN}}$$

Example: choosing contact lenses

Age	Spectacle prescription	Ast	Tear production rate	Lens
young	myope	no	reduced	none
young	myope	no	normal	soft
young	myope	yes	reduced	none
young	myope	yes	normal	hard
young	hypermetrope	no	reduced	none
young	hypermetrope	no	normal	soft
young	hypermetrope	yes	reduced	none
young	hypermetrope	yes	normal	hard
pre-presbyopic	myope	no	reduced	none
pre-presbyopic	myope	no	normal	soft
pre-presbyopic	myope	yes	reduced	none
pre-presbyopic	myope	yes	normal	hard
pre-presbyopic	hypermetrope	no	reduced	none
pre-presbyopic	hypermetrope	no	normal	soft
pre-presbyopic	hypermetrope	yes	reduced	none
pre-presbyopic	hypermetrope	yes	normal	none
presbyopic	myope	no	reduced	none
presbyopic	myope	no	normal	none
presbyopic	myope	yes	reduced	none
presbyopic	myope	yes	normal	hard
presbyopic	hypermetrope	no	reduced	none
presbyopic	hypermetrope	no	normal	soft
presbyopic	hypermetrope	yes	reduced	none
presbyopic	hypermetrope	yes	normal	none

Ast = Astigmatism

Rule representation and reasoning

Rule representation:

- Logical implication (= rules)

  LHS → RHS

  (LHS = left-hand side = antecedent; RHS = right-hand side = consequent)
- Literals in LHS and RHS are of the form:

  Variable ∘ *value* (or Attribute ∘ *value*)

  where $\circ \in \{<, \leq, =, >, \geq\}$

Rule-based reasoning:

$\mathcal{R} \cup F \vDash C$

where

- $\mathcal{R}$ is a set of rules $r \in \mathcal{R}$ (rule-base)
- F is a set of facts of the form

  Variable = *value*
- C is a set of conclusions of the same form as facts

ZeroR

Basic ideas:

- Construct rule that predicts the majority class
- Used as **baseline** performance

Example

Contact lenses recommendation rule:
→ Lens = *none*

- Total number of instances: 24
- Correctly classified instances: 15 (62.5%)
- Incorrectly classified instances: 9 (37.5%)

```
=== Detailed Accuracy By Class ===
TP Rate  FP Rate  Precision  Recall  F-Measure  Class
  0        0        0          0       0        soft
  0        0        0          0       0        hard
  1        1        0.625      1       0.769    none

=== Confusion Matrix ===
  a  b  c   <-- classified as
  0  0  5 |  a = soft
  0  0  4 |  b = hard
  0  0 15 |  c = none
```

OneR

- Construct a single-condition rule for each variable-value pair
- Select the rules defined for a single variable (in the condition) which perform best

OneR(classvar)
{
  $\mathcal{R} \leftarrow \varnothing$
  for each var ∈ Vars **do**
    for each value ∈ Domain(var) **do**
      classvar.most-freq-value ←
        MostFreq(var.value, classvar)
      rule ← MakeRule(var.value,
              classvar.most-freq-value)
      $\mathcal{R} \leftarrow \mathcal{R} \cup \{\text{rule}\}$
  for each $r \in \mathcal{R}$ **do**
    CalculateErrorRate(r)
  $\mathcal{R} \leftarrow$ SelectBestRulesForSingleVar($\mathcal{R}$)
}

OneR: Example

Rules for contact-lenses recommendation:

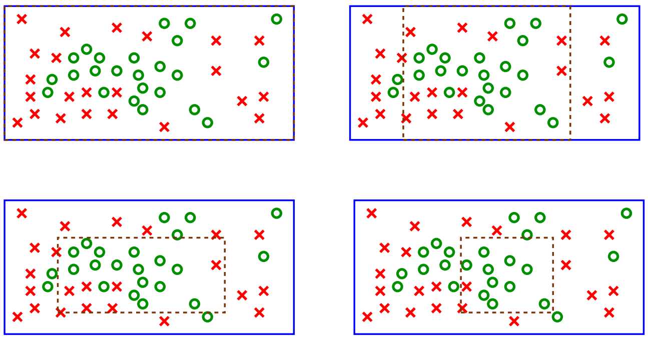
Tears = *reduced* → Lens = *none*
Tears = *normal* → Lens = *soft*

- 17/24 instances correct
- Correctly classified instances: 17 (70.83%)
- Incorrectly classified instances: 7 (29.16%)

```
=== Detailed Accuracy By Class ===
TP Rate FP Rate  Precision  Recall  F-Measure  Class
  1       0.368    0.417      1       0.588    soft
  0       0        0          0       0        hard
  0.8     0        1          0.8     0.889    none

=== Confusion Matrix ===
  a  b  c   <-- classified as
  5  0  0 |  a = soft
  4  0  0 |  b = hard
  3  0 12 |  c = none
```

Generalisation: separate-and-cover



Covering of classes:

- Rule-set generation for each class value separately
- Peeling: box compression – instances are peeled off (fall outside the box) one face at the time
- PRISM algorithm

Example: choosing contact lenses

Recommended contact lenses: none, soft, hard

General principles:

1. Choose class value, e.g. *hard*
2. Construct rule *Condition* → Lens = *hard*
3. Determine accuracy $\alpha = p/t$ for all possible *condition*s, where
   - t: *total* number of instances covered by the rule
   - p: covered instances with the right (*positive*) class value

Condition	$\alpha = p/t$
Age = *youg*	2/8
Age = *pre-presbyopic*	1/8
Age = *presbyopic*	1/8
Spectacles = *myope*	3/12
Spectacles = *hypermetrope*	3/12
Astigmatism = *no*	0/12
Astigmatism = *yes*	4/12
Tears = *reduced*	0/12
Tears = *normal*	4/12

4. Select best condition (4/12)

Separate-and-cover algorithm

```
SC(classvar, D)
{  R ← ∅
   for each val ∈ Domain(classvar) do
    E ← D
    while E contains instances with val do
      rule ← MakeRule(rhs(classvar.val), lhs(∅))
      IR ← ∅
      until rule is perfect do
        for each var ∈ Vars, ∀rule ∈ IR : var ∉ rule do
         for each value ∈ Domain(var) do
           inter-rule ← Add(rule, lhs(var.value))
           IR ← IR ∪ {inter-rule}
        rule ← SelectRule(IR)
      R ← R ∪ {rule}
      RC ← InstancesCoveredBy(rule, E)
      E ← E\RC
}
```

SelectRule: based on accuracy $\alpha = p/t$; if $\alpha = \alpha'$, for two rules, select the one with highest p

SelectRule example

Rule:

- RHS: Lens = *hard*
- LHS: Astigmatism = *yes*, with $\alpha = 4/12$

Not very accurate; **expanded rule:**
(Astigmatism = *yes* ∧ *New-condition*)
→ Lens = *hard*

Age	Spectacle prescription	Ast	Tear product rate	Lens
young	myope	yes	reduced	none
young	myope	yes	normal	hard
young	hypermetrope	yes	reduced	none
young	hypermetrope	yes	normal	hard
pre-presbyopic	myope	yes	reduced	none
pre-presbyopic	myope	yes	normal	hard
pre-presbyopic	hypermetrope	yes	reduced	none
pre-presbyopic	hypermetrope	yes	normal	none
presbyopic	myope	yes	reduced	none
presbyopic	myope	yes	normal	hard
presbyopic	hypermetrope	yes	reduced	none
presbyopic	hypermetrope	yes	normal	none

- Age = young (2/4); Age = pre-presbyopic (1/4); Age = presbyopic (1/4)
- Spectacles = myope (3/6); Spectacles = hypermetrope (1/6)
- Tears = reduced (0/6); Tears = normal (4/6)

SelectRule example (continued)

Rule:

(Astigmatism = *yes* $\wedge$ Tears = *normal*)
$\rightarrow$ Lens = *hard*

Expanded rule:

(Astigmatism = *yes* $\wedge$ Tears = *normal* $\wedge$
New-condition) $\rightarrow$ Lens = *hard*

Age	Spectacle prescription	Ast	Tear product rate	Lens
young	myope	yes	normal	hard
young	hypermetrope	yes	normal	hard
pre-presbyopic	myope	yes	normal	hard
pre-presbyopic	hypermetrope	yes	normal	none
presbyopic	myope	yes	normal	hard
presbyopic	hypermetrope	yes	normal	none

- Age = young (2/2); Age = pre-presbyopic (1/2); Age = presbyopic (1/2)
- Spectacles = myope (3/3); Spectacles = hypermetrope (1/3)

$\Rightarrow$ (Astigmatism = *yes* $\wedge$ Tears = *normal* $\wedge$
Spectacles = *myope*) $\rightarrow$ Lens = *hard*

SC example (continued)

Delete 3 instances from E; **new rule:**
New-condition $\rightarrow$ Lens = *hard*

Age	Spectacle prescription	Ast	Tear production rate	Lens
young	myope	no	reduced	none
young	myope	no	normal	soft
young	myope	yes	reduced	none
young	hypermetrope	no	reduced	none
young	hypermetrope	no	normal	soft
young	hypermetrope	yes	reduced	none
young	hypermetrope	yes	normal	hard
pre-presbyopic	myope	no	reduced	none
pre-presbyopic	myope	no	normal	soft
pre-presbyopic	myope	yes	reduced	none
pre-presbyopic	hypermetrope	no	reduced	none
pre-presbyopic	hypermetrope	no	normal	soft
pre-presbyopic	hypermetrope	yes	reduced	none
pre-presbyopic	hypermetrope	yes	normal	none
presbyopic	myope	no	reduced	none
presbyopic	myope	no	normal	none
presbyopic	myope	yes	reduced	none
presbyopic	hypermetrope	no	reduced	none
presbyopic	hypermetrope	no	normal	soft
presbyopic	hypermetrope	yes	reduced	none
presbyopic	hypermetrope	yes	normal	none

WEKA Results

Rules:

```
If Astigmatism = no and Tears = normal
   and Spectacles = hypermetrope then Lens = soft
If Astigmatism = no and Tears = normal
   and age = young then  Lens = soft
If age = pre-presbyopic and Astigmatism = no
   and Tears = normal then Lens = soft
If Astigmatism = yes and Tears = normal
   and Spectacles = myope then Lens = hard
If age = young and Astigmatism = yes
   and Tears = normal then Lens = hard
If Tears = reduced then none
If age = presbyopic and Tears = normal
   and Spectacles = myope
   and Astigmatism = no then Lens = none
If Spectacles = hypermetrope
   and Astigmatism = yes
   and age = pre-presbyopic then Lens = none
If age = presbyopicand Spectacles = hypermetrope
   and Astigmatism = yes then Lens = none

=== Confusion Matrix ===
  a  b  c   <-- classified as
  5  0  0 |  a = soft
  0  4  0 |  b = hard
  0  0 15 |  c = none
```

Correctly classified instances: 24 (100%)

Limitations

- Adding one condition at the time is greedy search (‘optimal’ state may be missed)
- Accuracy $\alpha = p/t$: promotes overfitting: the more ‘correct’ (higher p compared to t) is, the higher α
- Resulting rules cover all instances perfectly

Example

Consider rule r_1 with accuracy $\alpha_1 = 1/1$ and rule r_2 with accuracy $\alpha_2 = 19/20$, then r_1 is considered superior to r_2

Alternative 1: information gain

Alternative 2: probabilistic measure

Information gain

$$I_D(r) = p' \left[\log \frac{p'}{t'} - \log \frac{p}{t} \right]$$

where

- $\alpha = p/t$ is the accuracy *before* adding a condition to r
- $\alpha' = p'/t'$ is the accuracy *after* a condition has been added to r

Example

Consider rule r' with $\alpha' = 1/1$ and rule r'' with accuracy $\alpha'' = 19/20$, both modifications of r with $\alpha = 20/200$. Then is r' considered superior to r'' according to accuracy, but

$I_D(r') = 1[\log(1/1) - \log(20/200)] = 1$
$I_D(r'') = 19[\log(19/20) - \log(20/200)] \approx 18.6$

hence r' is inferior to r'' according to information gain

Comparison accuracy versus information gain

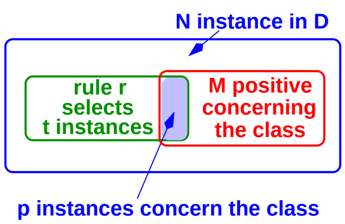
Information gain I:

- Emphasis is on large number of positive instances
- High coverage cases first, special cases later
- Resulting rules cover all instances perfectly

Accuracy α:

- Takes number of positive instances only into account if ties break
- Special cases first, high coverage cases later
- Resulting rules cover all instances perfectly

Probabilistic measure



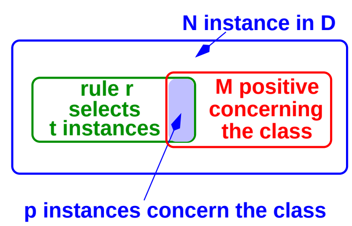
- $N = |D|$: # instances in dataset D
- M: # instances in D concerning a class
- t: # instances in D on which rule r succeeds
- p: # positive instances

Hypergeometric distribution:

$$f(k) = \frac{\binom{M}{k}\binom{N-M}{t-k}}{\binom{N}{t}}$$

sampling without replacement: probability that k instances out of t belong to the class

Probabilistic measure



Hypergeometric distribution:

$$f(k) = \frac{\binom{M}{k}\binom{N-M}{t-k}}{\binom{N}{t}}$$

- Rule r selects t instances, of which p are positive
- Probability that a randomly chosen rule r' does as well or better than r:

$$P(r') = \sum_{k=p}^{\min\{t,M\}} f(k) = \sum_{k=p}^{\min\{t,M\}} \frac{\binom{M}{k}\binom{N-M}{t-k}}{\binom{N}{t}}$$

Approximation

$$P(r') = \sum_{k=p}^{\min\{t,M\}} \frac{\binom{M}{k}\binom{N-M}{t-k}}{\binom{N}{t}}$$
$$\approx \sum_{k=p}^{\min\{t,M\}} \binom{t}{k}\left(\frac{M}{N}\right)^k \left(1-\frac{M}{N}\right)^{t-k}$$

i.e. hypergeometric distribution approximated by a binomial distribution

$$= I_{M/N}(p, t-p+1)$$

where $I_x(\alpha,\beta)$ is the incomplete beta function:

$$I_x(\alpha,\beta) = \frac{1}{\mathrm{B}(\alpha,\beta)}\int_0^x z^{\alpha-1}(1-z)^{\beta-1}\mathrm{d}z$$

where $\mathrm{B}(\alpha,\beta)$ is the beta function, defined as

$$\mathrm{B}(\alpha,\beta) = \int_0^1 z^{\alpha-1}(1-z)^{\beta-1}\mathrm{d}z$$

Reduced-error pruning

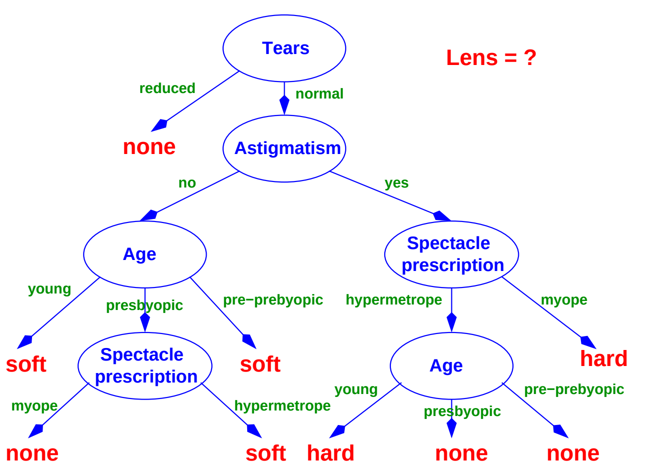
Danger of overfitting to training set can be reduced by splitting this into:

- a growing set (GS) (2/3 of training set)
- a pruning set (PS) (1/3 of training set)

REP(classvar, D)
{ $\mathcal{R} \leftarrow \varnothing$; $E \leftarrow D$
  (GS, PS) $\leftarrow$ Split(E)
  while $E \neq \varnothing$ **do**
    IR $\leftarrow \varnothing$
    for each val $\in$ Domain(classvar) **do**
      if GS and PS contain a val-instance **then**
        rule $\leftarrow$ BSC(classvar.val, GS)
        while P(rule | PS) $>$ P(rule$^-$ | PS) **do**
          rule $\leftarrow$ rule$^-$
        IR $\leftarrow$ IR $\cup$ {rule}
    rule $\leftarrow$ SelectRule(IR); $\mathcal{R} \leftarrow \mathcal{R} \cup$ {rule}
    RC $\leftarrow$ InstancesCoveredBy(rule, E)
    $E \leftarrow E \backslash$RC
    (GS, PS) $\leftarrow$ Split(E)
}

BSC is basic separate-and-cover algorithm, and rule$^-$ is a rule with last condition removed

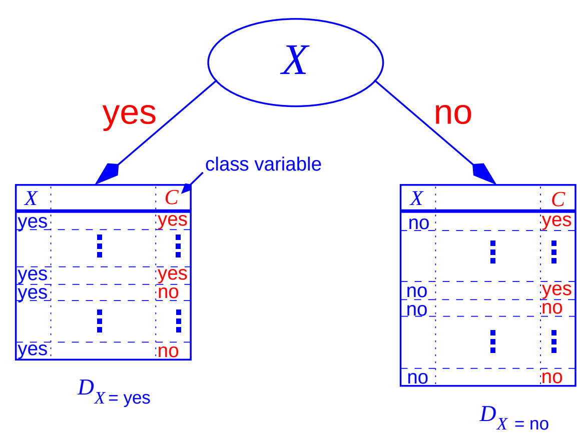
Divide-and-conquer: decision trees



Learning **decision trees**:

- R. Quinlan: ID3, C4.5 and C5.0
- L. Breiman: CART (Classification and Regression Trees)

Which variable/attribute is best?



Dataset D:

$$D = D_{X=\text{yes}} \cup D_{X=\text{no}}$$

Entropy:

$$H_C(X=x) = -\sum_c P(C=c|X=x)\ln P(C=c|X=x)$$

Expected entropy:

$$E_{H_C}(X) = \sum_x P(X=x)H_C(X=x)$$

Information gain (again)

Dataset D:

$$D = D_{X=\text{yes}} \cup D_{X=\text{no}}$$

Entropy:

$$H_C(X=x) = -\sum_c P(C=c|X=x) \ln P(C=c|X=x)$$

Expected entropy:

$$E_{H_C}(X) = \sum_x P(X=x) H_C(X=x)$$

Without the split of the dataset D on variable X, the entropy is:

$$H_C(\top) = -\sum_c P(C=c) \ln P(C=c)$$

Information gain G_C from X:

$$G_C(X) = H_C(\top) - E_{H_C}(X)$$

Example: contact lenses recommendation

Class variable is Lens:

$$P(\text{Lens}) = \begin{cases} 5/24 & \text{if Lens} = \textit{soft} \\ 4/24 & \text{if Lens} = \textit{hard} \\ 15/24 & \text{if Lens} = \textit{none} \end{cases}$$

$$\begin{aligned} H(\top) &= -\frac{5}{24}\ln\frac{5}{24} - \frac{4}{24}\ln\frac{4}{24} - \frac{15}{24}\ln\frac{15}{24} \\ &\approx 0.92 \end{aligned}$$

For variable Ast (Astigmatism):

$$P(\text{Lens}|\text{Ast} = \text{no}) = \begin{cases} 5/12 & \text{if Lens} = \textit{soft} \\ 0/12 & \text{if Lens} = \textit{hard} \\ 7/12 & \text{if Lens} = \textit{none} \end{cases}$$

Therefore:

$$\begin{aligned} H(\text{Ast}=\textit{no}) &= -\frac{5}{12}\ln\frac{5}{12} - \frac{0}{12}\ln\frac{0}{12} - \frac{7}{12}\ln\frac{7}{12} \\ &\approx 0.68 \end{aligned}$$

Example (continued)

For variable Ast (Astigmatism):

$$P(\text{Lens}|\text{Ast} = \text{yes}) = \begin{cases} 0/12 & \text{if Lens} = \textit{soft} \\ 4/12 & \text{if Lens} = \textit{hard} \\ 8/12 & \text{if Lens} = \textit{none} \end{cases}$$

Therefore:

$$\begin{aligned} H(\text{Ast}=\textit{yes}) &= -\frac{0}{12}\ln\frac{0}{12} - \frac{4}{12}\ln\frac{4}{12} - \frac{8}{12}\ln\frac{8}{12} \\ &\approx 0.64 \end{aligned}$$

$$\begin{aligned} \Rightarrow E_H(\text{Ast}) &= \frac{1}{2}H(\text{Ast}=\textit{no}) + \frac{1}{2}H(\text{Ast}=\textit{yes}) \\ &= 1/2(0.68 + 0.64) = 0.66 \end{aligned}$$

Information gain:

$$\begin{aligned} \Rightarrow G(\text{Ast}) &= H(\top) - E_H(\text{Ast}) \\ &= 0.92 - 0.66 = 0.26 \end{aligned}$$

Example (continued)

For variable Tears:

$$\begin{aligned} E_H(\text{Tears}) &= \frac{1}{2}H(\text{Tears}=\textit{red}) + \frac{1}{2}H(\text{Tears}=\textit{norm}) \\ &\approx 1/2(0.0 + 1.1) = 0.55 \end{aligned}$$

Information gain:

$$\begin{aligned} \Rightarrow G(\text{Tears}) &= H(\top) - E_H(\text{Tears}) \\ &= 0.92 - 0.55 = 0.37 \end{aligned}$$

Comparison:

$$G(\text{Tears}) > G(\text{Ast})$$

Select Tears as first splitting variable

Final remarks I

A node with too many branches causes the information gain measure to break down

Example
Suppose that with each branch of a node a dataset with exactly one instance is associated:

$$E_{H_C}(X) = n \cdot 1/n \cdot (1 \log 1 + 0 \log 0) = 0$$

if X has n values. Hence, $G_C(X) = H_C(\top) - 0 = H_C(\top)$ attains a maximum

Solution: Gain ratio R_C:

- Split information

$$H_X(\top) = -\sum_x P(X = x) \ln P(X = x)$$

- Gain ratio:

$$R_C(X) = G_C(X)/H_X(\top)$$

Final remarks II

- Variable selection is myopic: it does not look beyond the effects of its own values; a resulting decision tree is therefore likely to be *suboptimal*

- Decision trees may grow unwieldy, and may need to be pruned (ID3 ⇒ C4.5)
  - subtree replacement
  - subtree raising

- Decision trees can also be used for numerical variables: regression trees