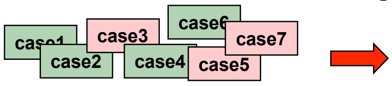
Supervised Learning of Concepts from Cases

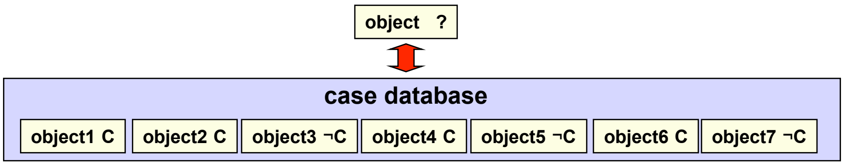
Given a set of positive and negative examples for an unknown class, what is a conceptual description for that class?

- Special kind of case-based reasoning (CBR)



concept description C

- Special kind of case-based problem solving where the problem is to determine class membership in a binary classification task.



1

Applications of Concept Formation

Compare to Neural Network (NN) learning tasks:
NNs are trained to recognize complex patterns such as
- handwritten characters
- earthquake threat in seismological signals
- faces
- etc.

Cases-based concept formation solves similar tasks on a logical basis instead of numerical values.

Examples:

Learning conceptual descriptions for
- market situations suitable for investment
- credit worthiness of customers
- complex visual structures

Basic idea of case-based concept formation: Look for combinations of properties present in positive examples and absent in negative examples.

2

History of Concept Formation

Early work in AI has dealt mostly with models of human concept formation.

Being able to generalize from past experiences and predict the future is considered one of the hallmarks of intelligence.

<u>Example</u>:
E.B. Hunt, C.I. Hovland: Programming a Model of Concept Formulation. In: Feigenbaum & Feldman (Eds.), Computers and Thought, McGraw-Hill 1963

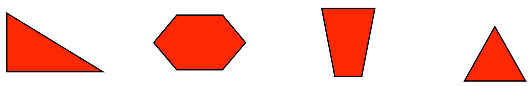
Typical approach of humans:

- **Consider individual attributes first, consider combinations later**
- **Use basic relations between attributes, e.g. EQUAL, GREATER**
- **Decide for one concept hypothesis at a time**

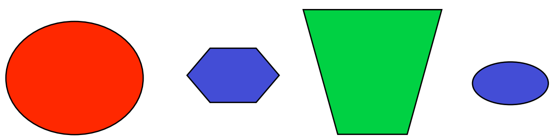
Experiment 1

What concepts describe the positive examples?

Positive examples:



Negative examples:

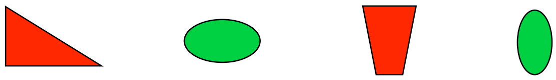


Three solutions:
{ (colour: red) (shape: polygon) }
{ (colour: red) (size: small) }
{ (colour: red) (shape: polygon) (size small) }

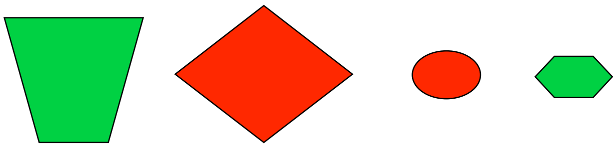
Experiment 2

What concept describes the positive examples?

Positive examples:



Negative examples:

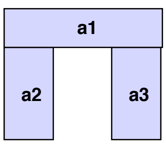


Solution: **{ (size: small) (colour&shape: red-polygon green-oval) }**

Learning Structures from Examples

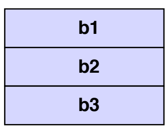
Winston, Learning Structural Descriptions from Examples (1975)

Example of an arch:



(on a1 a2) (on a1 a3)
(parts a [a1 a2 a3])
(inst a1 brick) (inst a2 brick) (inst a3 brick)
(inst a arch)
(not (touch a2 a3))

Example of a non-arch:



(on b1 b2) (on b2 b3)
(parts b [b1 b2 b3])
(inst b1 brick) (inst b2 brick) (inst b3 brick)
(inst b non-arch)
(not (touch b1 b3))

Obtain a general description of an arch from examples and counter-examples

Generalisation and Specialisation Examples

Generalisation:

(on a1 a2) (on a1 a3) (parts a [a1 a2 a3]) (inst a1 brick) (inst a2 brick) (inst a3 brick) (inst a arch) (not (touch a2 a3))	➡	(if (parts ?x [?x1 ?x2 ?x3]) (inst ?x arch))

Specialisation:

(if (parts ?x [?x1 ?x2 ?x3]) (inst ?x arch))	➡	(if (and (parts ?x [?x1 ?x2 ?x3]) (on ?x1 ?x2) (on ?x2 ?x3) (not (touch ?x2 ?x3)) (inst ?x arch))

7

Generalisation and Specialisation Rules

GENERALISATION

Variabilisation:

constants => variables brick1 => ?x

is-a hierarchy generalisation:

class => parent class brick => polyeder => object
glued(x, y) => attached(x, y)

Disjunctive generalisation:

expr1 => expr1 V expr2 (on ?x ?y) => (on ?x ?y) v (above ?x ?y)

Conjunctive generalisation:

expr1 $\wedge$ expr2 => expr1 (on ?x table) $\wedge$ (red ?x) => (on ?x table)

SPECIALISATION

Inverse of generalisation operations

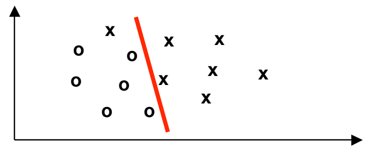
8

Characteristics of Concept Formation

So far, the examples have shown several characteristics of concept formation:

- Few examples may suffice
- There may be several solutions
- Relevant attributes may not be obvious
- It may be necessary to consider combinations of attributes
- The complexity of the task depends on the description language

Compare to Pattern Recognition and classification in feature space:



Version Space Learning (VSL)

M. Genesereth, N. Nilsson: Logical Foundations of Artificial Intelligence
Morgan Kaufmann, 1987
T.M. Mitchell: Generalization as Search. Artificial Intelligence 18, 1982, 203-226

Basic idea:
For given positive and negative examples, represent the space of consistent concepts by two boundaries:

- The <u>general boundary</u> contains all concepts which cannot be further generalized without becoming inconsistent (i.e. including negative examples).
- The <u>specific boundary</u> contains all concepts which cannot be further specialized without becoming inconsistent (i.e. excluding positive examples).

For each example, adjust the general and specific boundary accordingly.

Example: Learning to Classify Mushrooms

Learn from positive and negative examples to distinguish poisonous and nonpoisonous mushrooms.

Mushroom description:

Colour	{Red, Grey}
Size	{Small, Large}
Shape	{rOund, Elongated}
Environment	{Humid, Dry}
Height	{loW, hIgh}
Texture	{sMooth, roUgh}
Class	{Poisonous, Nonpoisonous}

Note simple attribute language for the sake of an easy example. VSL can deal with much richer languages.

Initialization of Mushroom Version Space

Initially, the general boundary GB contains the concept hypothesis which includes all possible examples:

GB = { [{ R, G}, {S, L}, {O, E}, {H, D}, {W, I}, {M, U}] }

Initially, the specific boundary SB contains the concept hypothesis which excludes all possible examples:

SB = { [{ }, { }, { }, { }, { }, { }] }

Training data presented incrementally:

	Colour	Size	Shape	Environ.	Height	Texture	Class
Example1	{R}	{S}	{O}	{H}	{W}	{M}	P
Example2	{R}	{S}	{E}	{H}	{W}	{M}	P
Example3	{G}	{L}	{E}	{H}	{W}	{U}	¬P
Example4	{R}	{S}	{E}	{H}	{I}	{U}	P

Learning Procedure for Mushrooms

Example1: [{R}, {S}, {O}, {H}, {W}, {M}] positiv
Example2: [{R}, {S}, {E}, {H}, {W}, {M}] positiv
Example3: [{G}, {L}, {E}, {H}, {W}, {U}] negativ
Example4: [{R}, {S}, {E}, {H}, {I}, {U}] positiv

initial version space

GB(0) = { [{ R, G}, {S, L}, {O, E}, {H, D}, {W, I}, {M, U}] }
GB(1) = { [{ R, G}, {S, L}, {O, E}, {H, D}, {W, I}, {M, U}] }
GB(2) = { [{ R, G}, {S, L}, {O, E}, {H, D}, {W, I}, {M, U}] }
GB(3) = { [{ R}, {S, L}, {O, E}, {H, D}, {W, I}, {M, U}],
[{ R, G}, {S}, {O, E}, {H, D}, {W, I}, {M, U}],
[{ R, G}, {S, L}, {O, E}, {H, D}, {W, I}, {M}] }

more specific

GB(4) = { [{ R}, {S, L}, {O, E}, {H, D}, {W, I}, {M, U}],
[{ R, G}, {S}, {O, E}, {H, D}, {W, I}, {M, U}] }

SB(4) = { [{R}, {S}, {O, E}, {H}, {W, I}, {M, U}] }

version space
after 4 examples

SB(3) = { [{R}, {S}, {O, E}, {H}, {W}, {M}] }
SB(2) = { [{R}, {S}, {O, E}, {H}, {W}, {M}] }
SB(1) = { [{R}, {S}, {O}, {H}, {W}, {M}] }
SB(0) = { [{ }, { }, { }, { }, { }, { }] }

more general

13

Candidate Elimination Algorithm

Initialize SB to the empty concept and SG to the most general concept
For each training example E do:
If E is a positive example then:
  Remove from GB any hypothesis inconsistent with E
  For each hypothesis H in SB that is not consistent with E do:
    Remove H from SB
    Add to SB all minimal generalizations H´ of H such that
      H´ is consistent with E and
      some member of GB is more general than H´
    Remove from SB any hypothesis that is more general
    than another hypothesis in SB
else if E is a negative example:
  Remove from SB any hypothesis consistent with E
  For each hypothesis H in GB that is consistent with E do:
    Remove H from G
    Add to GB all minimal specializations H´ of H such that
      H´ is inconsistent with E and
      some member of SB is more specific than H´ .
    Remove from GB any hypothesis that is less general
    than another hypothesis in GB

14

Properties of the Version Space

- The version space consists of all hypotheses (potential concept descriptions) equal to or less general than the concepts of the general boundary and equal to or more general than the concepts of the specific boundary.
- A consistent version space contains all hypotheses which subsume all positive examples and do not subsume any negative examples.
- To establish a version space, hypotheses must be partially ordered in a specialization lattice.
- After learning, all hypotheses of the version space are candidates for a concept definition of the class in question.
- If learning causes the concepts of the general boundary to become more special than the specific boundary, the version space collapses: no possible concept descriptions exist for the examples.
- A single "outlier" may cause a version space to collaps.

15

Learning Structured Objects

Example:

Your house-keeping robot learns to lay a table

Example:

An image analysis program learns to recognize balconies or window arrays

Examples are presented in terms of sets of components in annotated images

Learning conceptual descriptions of structured objects requires an expressive description language

16

Learning "Entrances"

Annotated training image with four positive examples of "Entrance"

17

Concept Description with an Expressive Representation Language

Example of a learnt concept description of "Entrance"

ENTRANCE
Aggregate Width = [184..216] cm
Aggregate Height = [299..366] cm

Attributes with ranges of real values

Shape = { Quadratic }
Colour = {gray, green, yellow}

Attributes with sets of symbolical values

Has-Parts = [3..4]
door = [1..1]
stairs = [1..1]
canopy = [0..1]
railing = [0..1]
sign = [0..1]

Attributes with ranges of integer values

BelowNeighbourOf (stairs011) [0..2] (door012)
AboveNeighbourOf (door012) [0..2] (stairs011)

Symbolic spatial relations between parts

distance range

18

General-Specific Ordering

There must be a (partial) generalization order between concept hypotheses in order to determine the general boundary GB and the specific boundary SB of the version space.

$C_1 = (A_{11}\ V_{11}) \wedge .. \wedge (A_{1K}\ V_{1K}) \wedge (R_{11}\ V(A_{1r1})\ V(A_{2r1})) \wedge .. \wedge (R_{1L}\ V(A_{1rL})\ V(A_{2rL}))$

$C_2 = (A_{21}\ V_{21}) \wedge .. \wedge (A_{2M}\ V_{2M}) \wedge (R_{21}\ V(A_{2s1})\ V(A_{2s1})) \wedge .. \wedge (R_{2N}\ V(A_{2rN})\ V(A_{2rN}))$

C_1 and C_2 are two concepts with attribute-value pairs (A V) and relations (R V(A) V(A´)). C_1 is more general than C_2 (written $C_1 \geq C_2$) if for each (A_{2m} V_{1k}) and (R_{2n} $V(A_{2n})$ $V(A´_{2n})$) in C_2 there is a corresponding (A_{1k} V_{1k}) and (R_{1l} $V(A_{1l})$ $V(A´_{1l})$) in C_1 such that

$A_{1k} \geq A_{2m}$ and $V_{1k} \geq V_{1k}$, and

$R_{1l} \geq R_{2n}$ and $V(A_{1l}) \geq V(A_{2n})$ and $V(A´_{2n}) \geq V(A´_{1l})$

If neither $C_1 \geq C_2$ nor $C_2 \geq C_1$, there is no generalization order between C_1 and C_2.

When is one attribute more general than another attribute?
When is one value more general than another value?
When is one relation more general than another relation?

General-Specific Ordering of Attribute Values (1)

Set-valued attributes:

$V_1 \geq V_2$ iff $V_1 \supseteq V_2$

Example: {green, gray, yellow} > {gray, yellow}

Generalize V_1 to V_3 for inclusion of V_2:

$V_3 = V_1 \cup V_2$

Example: V_1 = {green, gray, yellow} V_2 = {blue} V_3 = {green, gray, yellow, blue}

Specialize V_1 to V_3 for exclusion of V_2:

$V_3 = V_1 \setminus V_2$

Example: V_1 = {green, yellow} V_2 = {green, blue} V_3 = {yellow}

General-Specific Ordering of Attribute Values (2)

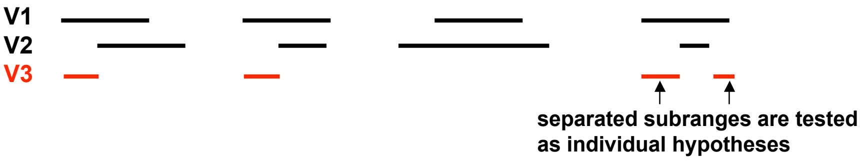
<u>Real-valued range attributes</u>:

$V_1 \geq V_2$ iff $V_1 \supseteq V_2$

Example: [1.2 ,, 2.0] > [1.3 .. 1.8]

Generalize $V_1 = [l_1 .. u_1]$ to $V_3 = [l_3 .. u_3]$ for inclusion of $V_2 = [l_2 .. u_2]$

Specialize $V_1 = [l_1 .. u_1]$ to $V_3 = [l_3 .. u_3]$ for exclusion of $V_2 = [l_2 .. u_2]$



<u>Scalar-valued range attributes</u> are treated analogously.

21

General-Specific Ordering of Relational Values

Relations are treated similar to attributes, except that values are pairs of attribute values.

<u>Example</u>:

(R V(A1) V(A2)) = (EQUAL V(Width) V(Height))
(Width [120 .. 200])
(Height [180 .. 300])

The relation constrains possible values of width and height to equal values in the range [180 .. 200]

Relations may have a generalization order. Let R1 and R2 be two relations between the same pair of attributes.

$R_1 \geq R_2$ iff $R_1 \supseteq R_2$

<u>Example</u>:

R1 = ALMOST-EQUAL
R2 = EQUAL
=> R1 > R2

22

Concept Selection from Version Space

VSL does not offer a criterion for selecting a final concept from version space.

Concepts from the GB are as permissive as possible while excluding all negative examples.

Concepts from the SB are as restrictive as possible while including all positive examples.

A good compromise seems to be the conjunction of all concepts of the GB.

23

Learnt Concept for Window Array

Version space learnt from 13 positive and 200 negative examples.
Concept hypothesis constructed by conjunction of all hypotheses in the general boundary:

Aggregate Width = [549..INF] cm
Aggregate Height = [0..199] cm
Parts Width = [0..INF] cm
Parts Height = [0..INF] cm
Parts Top-Left-X Variance = [132..INF] cm
Parts Top-Left-Y Variance = [0..32] cm
Parts Bottom-Right-X Variance = [116..INF] cm
Parts Bottom-Right-Y Variance = [0..8] cm
Has-Parts = [3..INF] window = [3..INF] door = [0..0]
Part-Of = [1..1] facade = [0..1] roof = [0..1]
Fuzzy-Equal (top-left-y)
Fuzzy-Equal (bottom-right-y)
Fuzzy-Equal (parts-height)
Fuzzy-Equal (parts-distance-x)
Value-Equal (parts-type)

24

Summary of Version Space Learning

- VSL is a logic-based method for determining all possible concept descriptions based on positive and negative examples.
- Concepts must be described in a description language which allows to establish a generalization hierarchy.
- VSL may be extended for structured objects.
- The version space may collapse if erroneous examples are introduced.
- Probabilistic learning models compete.