GECCO-2001 Tutorial on

Data Mining with Evolutionary Algorithms

Alex A. Freitas

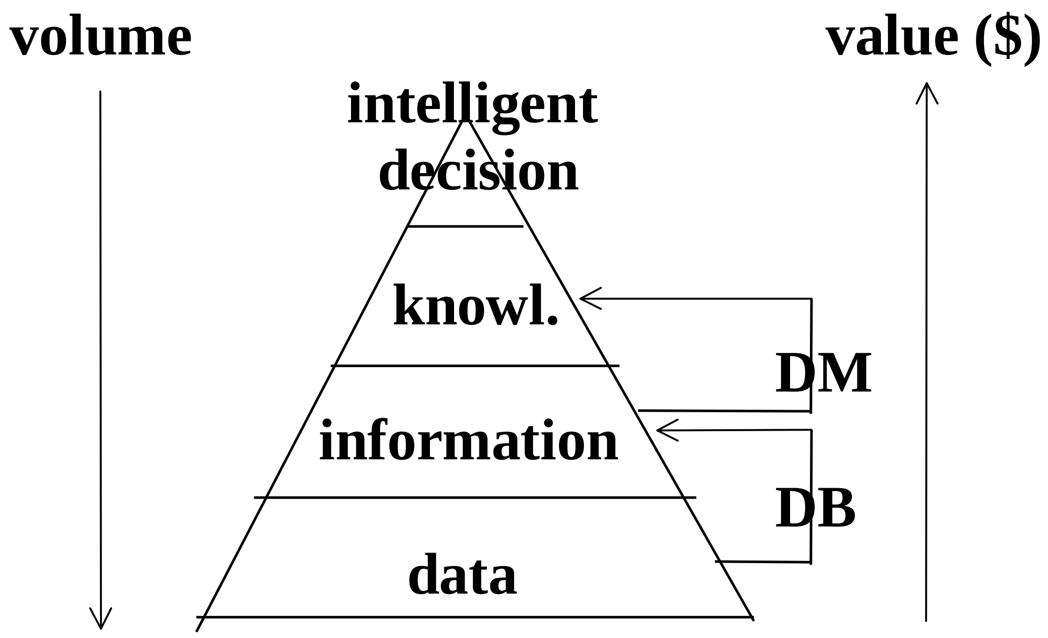
PUC-PR, Curitiba, Brazil

alex@ppgia.pucpr.br
http://www.ppgia.pucpr.br/~alex

Roadmap

- **Introduction**
  Classification and Clustering
- **Genetic Algorithms for Data Mining**
  Classification
  Clustering
  Attribute Selection
- **Genetic Programming for Classification**
  Constructive Induction
- **Conclusions**

Introduction



Information vs knowledge:
a simple example about a software house

Consulting low-level information in the DB:

How many videogames of type XYZ
were sold for customer ABC in 99/99/99?

Users - low managerial level

Now suppose we extract the following high-level knowledge from the database:

IF (Age < 18) AND (Job = student)
THEN (Buy = videogame) (prob.=90%)

We can ask: Which customers have a high probability of buying videogames?

Users - high managerial level

Desirable Properties of the discovered knowledge

*** Accurate (as much as possible)**

*** Comprehensible by the human user**

*** Interesting (useful / new / surprising)**

Data Mining Tasks

Types of problem to be solved:

Classification

Clustering

etc., etc.

Knowledge Discovery Paradigms

Type of method used to solve the task:

rule induction and decision trees

genetic algorithms

genetic programming

etc, etc;

Classification

Each example belongs to a predefined class

Each example consists of:

- **a class (or goal) attribute**
- **a set of predicting attributes**

The aim is to predict the class of an example, given its predicting attributes' values

[Hand 97], [Michie et al. 94]

Data partitioning for the classification task.

training data
(known class)

. . .		goal
		c
		b
		a
		a
		b
		c
		a

test data
(unknown class)

. . .		goal
		?
		?
		?
		?
		?
		?
		?

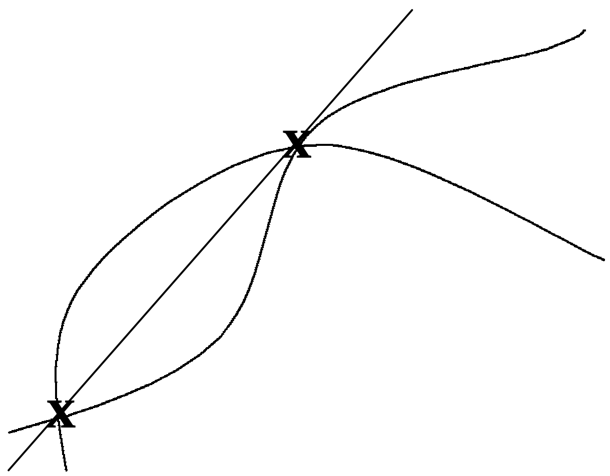
What is the next number in the sequence: [Bramer 96]

1, 4, 9, 16, ? (training data)

A possible answer is 20, based on the generator polynomium:

$$(-5n^4 + 50n^3 - 151n^2 + 250n - 120) / 24$$

Both n^2 and the complex polynomium are 100% consistent with the training data


x
x

Classification example [Freitas & Lavington 98]:

Goal is to predict whether or not a customer will buy a product, given a customer's Sex, Country and Age

Sex	Country	Age	Buy? (goal)
M	France	25	yes
M	England	21	yes
F	France	23	yes
F	England	34	yes
F	France	30	no
M	Germany	21	no
M	Germany	20	no
F	Germany	18	no
F	France	34	no
M	France	55	no

Classification rules for the above data:

IF (Country = ‘Germany’) THEN (Buy = ‘no’)

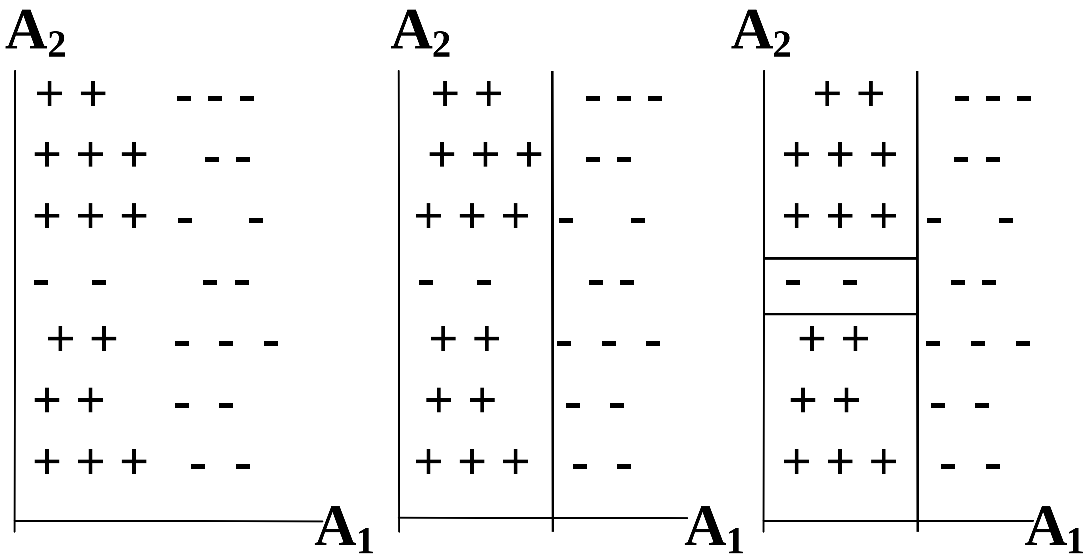
IF (Country = ‘England’) THEN (Buy = ‘yes’)

IF (Country = ‘France’ & Age ≤ 25)
THEN (Buy = ‘yes’)

IF (Country = ‘France’ & Age > 25)
THEN (Buy = ‘no’)

Classification regarded as data separation

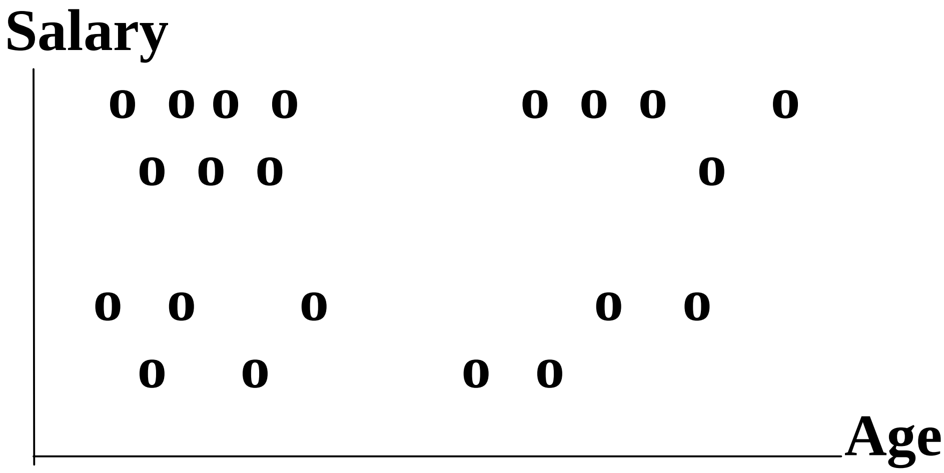
2 predicting attributes (A_1 and A_2)
2 classes ('+' and '-')



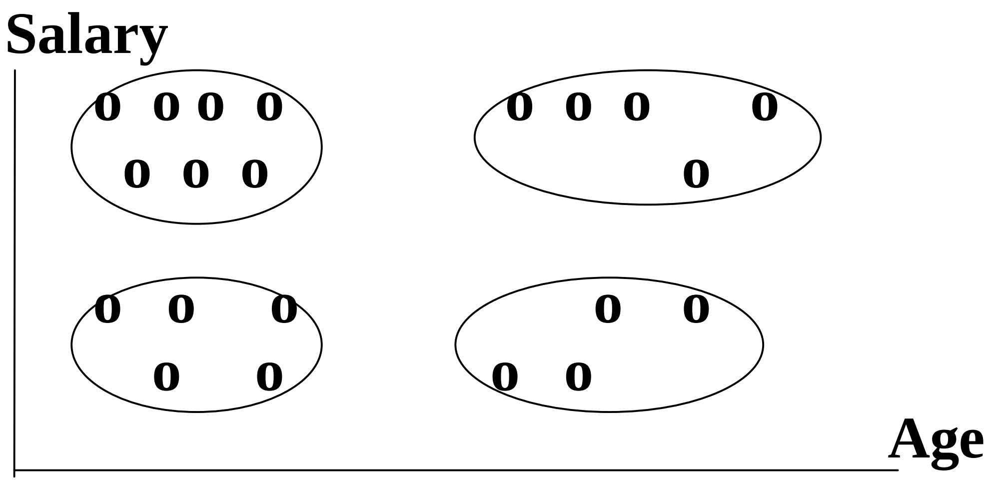
original data | **separating by A_1 values** | **separating by A_2 values**

Which classifier will be more accurate on unseen test data?

Clustering



The system must "invent" classes, by grouping similar examples



After clustering, we can apply classification methods

Criteria for finding good clusters

Minimize within-cluster distance

Maximize between-cluster distance

Favor a small number of clusters

Induction of Classification Rules

Basic idea: improve candidate rules, via generalization and specialization operations

Example of specialization:

IF (Country = 'France') THEN ...

specialization ↓

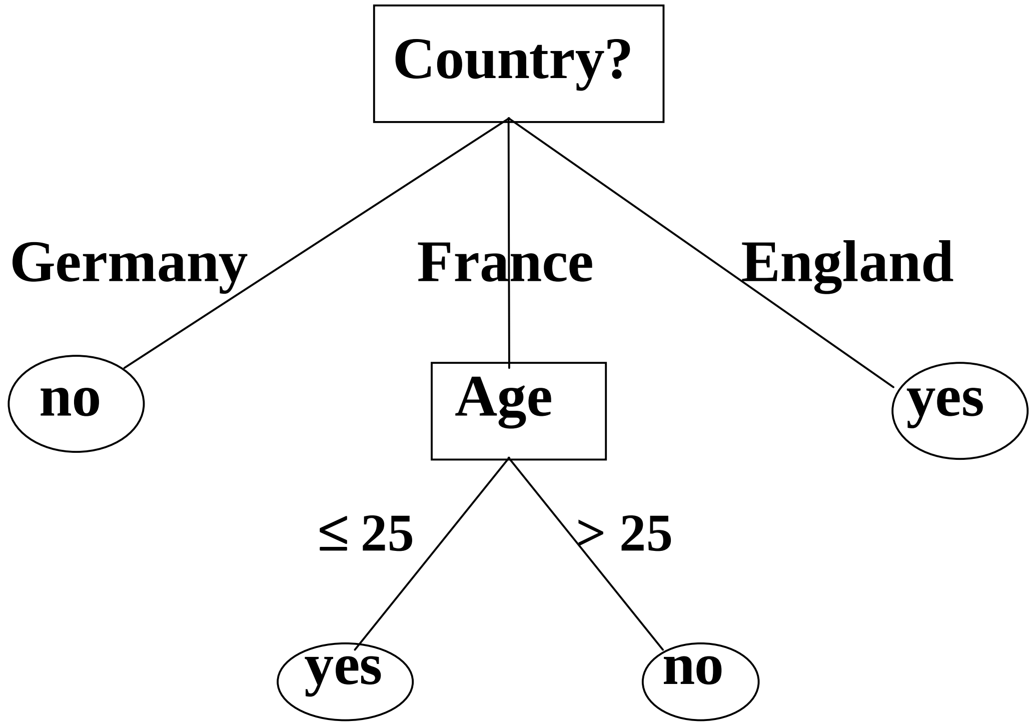
IF (Country = 'France' & Age ≤ 25) THEN ...

Generalization is the opposite operation

Decision Trees

internal nodes: predicting attributes
leaf nodes: predicted class

To classify a new example, push it down the tree, until reaching a leaf node



Tree is built by selecting one-attribute-at-a-time (local search)

LOOP

Select attribute that best separates classes;
Partition the set of examples in the current node according to selected attribute's values;

Repeat this process, recursively;

Drawback of Local search
(select one-attribute-at-a-time)

Problems with attribute interaction

Exclusive OR (XOR) problem:

A_1	A_2	XOR
0	0	0
0	1	1
1	0	1
1	1	0

Looking only at one attribute gives us no useful information for predicting XOR

Example of Simpson's Paradox

Renewal of magazine subscriptions, by month and subscription category
[Wagner 82], [Newson 91]

	subscription category					
month	gift	previous renewal	direct mail	sub. service	catalog agent	total
Jan						
total	3,954	18,364	2,986	20,862	149	45,955
renew	2,918	14,488	1,783	4,343	13	23,545
rate	0.812	0.789	0.597	0.208	0.087	0.512
Feb						
total	884	5,140	2,224	864	45	9,157
renew	704	3,907	1,134	122	2	5,869
rate	0.796	0.760	0.510	0.141	0.044	0.641

Genetic Algorithms for Data Mining

In data mining, GA can be used to:

(a) optimize parameters for other kinds of data mining algorithm

(b) discover knowledge by itself

Using GAs for parameter optimization

(a) finding a good set of attribute weights for nearest-neighbor algorithms

(b) finding a good set of weights and/or a good topology for a neural network

(c) selecting a good set of attributes to be given to another algorithm

GAs can also be used for rule discovery

Why should we consider using GAs rather than rule induction?

Both paradigms can discover high-level "IF-THEN" rules, but:

Most rule induction algorithms select one-attribute-at-a-time (*local* search)

GAs perform a *global* search that copes better with attribute interaction

Rules are evaluated as a whole by the fitness function

Genetic operators can modify many-attributes-at-a-time

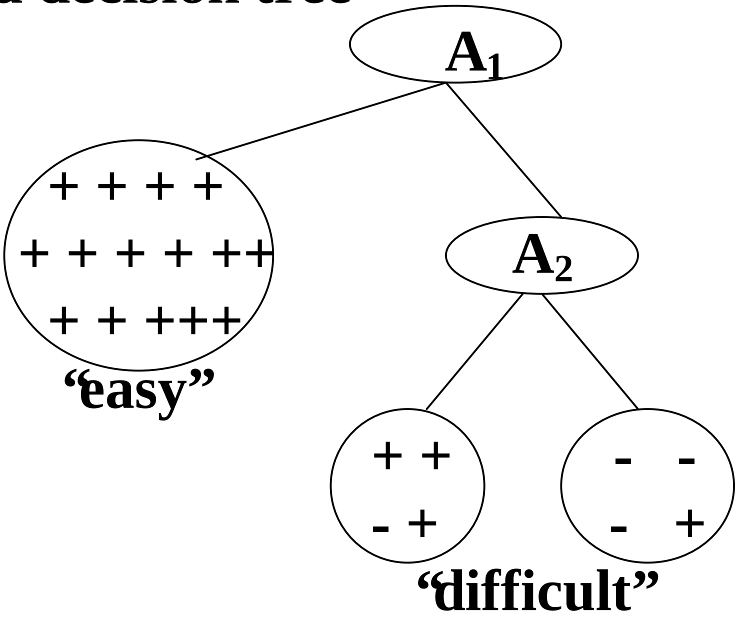
Hybrid decision tree/GA

[Carvalho & Freitas 2000a, 2000b]

- **decision tree used to classify "easy" examples (it exploits simplicity and efficiency of decision tree algorithms)**

- **GA used to classify "difficult" examples (it exploits GA ability to cope better with attribute interaction)**

Identifying easy/difficult examples in a decision tree



Basic ideas of GAs for rule discovery:

(a) Candidate rules are represented
as individuals of a population

(b) Rule quality is computed
by a fitness function

(c) Using task-specific knowledge

Classification with Genetic Algorithms

1) Each individual represents a rule set,
   i.e. an independent candidate solution

2) Each individual represents a single rule
   A set of individuals (or entire population) represents a candidate solution (rule set)

Individual Representation

In GABIL, an individual is a rule set, encoded as a bit string [DeJong et al. 93]

It uses k bits for the k values of a categorical attribute

If all k bits of an attribute are set to 1 the attribute is not used by the rule

Goal attribute: Buy furniture (y/n)
Marital_status: <u>S</u>ingle/<u>M</u>arried/<u>D</u>ivorced
House: <u>O</u>wn/<u>R</u>ented/<u>U</u>niversity

	Marital_status	House	Buy?
The string	**011**	**100**	**1**

represents the rule
IF (Marital_status = M or D) and (House = O)
THEN (Buy furniture = y)

An individual is a variable-length string representing a set of fixed-length rules

rule 1	rule 2
011 100 1	101 110 0

Mutation: traditional bit inversion

Crossover: corresponding crossover points in the two parents must semantically match

Example of crossover in GABIL

if a parent is "cut" in the second bit of a rule, the other parent must also be cut in the second bit of a rule, e.g.:

01|1 100 1 101 110 0
010 101 1 11|1 101 1 101 111 0

Individual representation

In GIL an individual is a set of rules,
using a high-level encoding [Janikow 93]

rule 1	rule 2
(A=1) and (B=2 or 3)	(C=2)

This kind of high-level encoding is more efficient for continuous attributes

Representing the predicted class

- included in the genome (and evolved)

- not included in the genome
  - all rules predict the same class
  - for a given rule antecedent, choose class maximizing rule quality

    [Greene & Smith 93], [Noda et al. 99]

Task-specific genetic operators

generalizing/specializing mutation
[Janikow 93], [Liu & Kwok 2000]

Example of specializing mutation:

IF (Age ≤ 30) ... AND ... THEN ...

$\Big\downarrow$ **specialization**

IF (Age ≤ 25) ... AND ... THEN ...

generalizing/specializing crossover

[Giordana et al 94], [Anglano et al. 1998]

generalizing crossover - logical OR

specializing crossover - logical AND

	generalizing	specializing
0 ǀ 1 0 ǀ 1	0 ǀ 1 1 ǀ 1	0 ǀ 0 0 ǀ 1
1 ǀ 0 1 ǀ 0	1 ǀ 1 1 ǀ 0	1 ǀ 0 0 ǀ 0

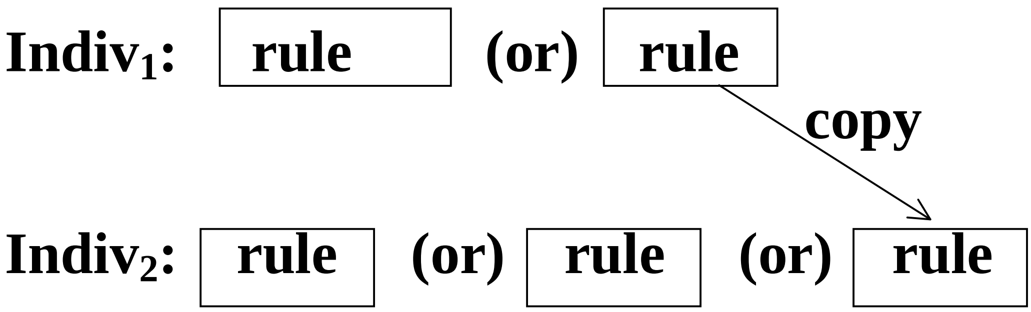
GIL has special genetic operators for handling: [Janikow 93]

- **rule sets**
- **rules**
- **rule conditions**

Operators can perform generalization, specialization or other operation

Generalization at the rule-set level in GIL:

Given two individuals, copy a rule from an individual to the other



Relevance-Based Rule Pruning

remove some conditions from a rule (simplifies and generalizes the rule)

Basic idea: the less relevant a rule condition is, the higher the probability of removing that condition

This basic idea was used e.g. in:
[Liu & Kwok 2000], [Carvalho & Freitas 2000a]

Fitness Functions for Classification

- **at least a measure of predictive accuracy**
- **possibly, also a measure of comprehensibility**
  (the fewer the number of rules and
  rule conditions, the better)

  E.g.: complexity = 2 #rules + #conditions

  [Janikow 93]

- **possibly, also a measure of rule interestingness [Noda et al. 1999]**

motivation for interestingness measures:
IF (pregnant) THEN (sex = ´female´)
(accurate, comprehensible, *uninteresting*)

- **Standard approach to combine accuracy, comprehensibility and interestingness: weighted fitness functions**

- **Problem: non-commensurate objectives**

- **Solution: multi-objective EAs**
  [Bhattacharyya 2000a, 2000b], [Kim et al. 2000]
  [Emmanouilidis et al. 2000]

Parallel GA

- **Parallelize fitness computation**
- **Fitness of many individuals can be computed in parallel (each processor evaluates a subset of individuals)**
- **Fitness of a single individual can be computed in parallel by p processors (data distributed across p processors)**

See [Freitas & Lavington 98], [Flockhart & Radcliffe 95], [Giordana & Neri 95],[Anglano et al 97],[Araujo et al 99]

Clustering with GA

Simple representation:

- **one gene per object to be clustered**
- **each gene = id of the cluster to which the object belongs**

E.g. - 10 objects, 4 clusters:

B C B B A D D C A D

Advantage: fixed-length individual

Disadvantages [Falkenauer 98]:

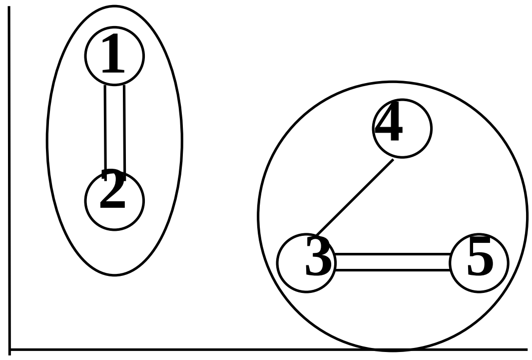
- **high redundancy**
  e.g. (A B B A) and (B A A B)

- **crossover and mutation problems**

- **is not scalable w.r.t. No. of objects**

Graph-based clustering [Park & Song 98]

Each data instance is a node in a graph
A cluster is a group of connected nodes



Adjacency-based representation:
A vector of N integer elements, where
N = No. of instances to be clustered.

i-th gene with value j means that the nodes i and j are connected by a link in the graph

E.g. the above clustering could be represented by: I = < 2, 1, 5, 3, 3 >

Search space size: $(N-1)^N$

Extending adjacency representation with task-specific knowledge:

The i-th gene can take on a value j only if j is one of the k nearest neighbours of the i-th instance

search space size: k^N

Advantages of this representation:

- **does not require prespecified number of clusters**
- **does not require special genetic operators to produce valid offspring**
- **knowledge-based representation reduces the size of the search space**

Disadvantages of this representation:

- **not scalable for large data sets (genome length is N, where N is the No. of instances)**

- **redundancy - several genotypes correspond to the same phenotype**

Clustering with hybrid GA/K-means
[Hall et al. 99]

GA optimizes location of cluster centroids

Individual representation:
a matrix of c x n cluster centers
(c = No. of clusters, f = No. of features)

M_{11} . . . . . . M_{1f}	**fitness based on**
. . . .	**distances from**
. . . .	**centroids**
M_{c1} . . . . . . M_{cf}	**(K-means)**

Attribute selection with GA

Simple Individual Representation:
One gene for each predicting attribute

Each gene can take on 2 values:
0 (attrib. ignored), or 1 (attrib. selected)

Example: 0 1 1 0 1 0 0 0
(attributes 2, 3, and 5 are selected)

See [Vafaie & DeJong 93], [Bala et al. 95]

Fitness depends on the performance of a data mining algorithm with the selected attributes (GA is a wrapper)

Fitness function can include a penalty for individuals with many attributes

More elaborated individual encoding: [Cherkauer & Shavlik 97]

Each gene can contain an attribute (A_i) or be empty (0)

Ex.: 0 0 A_7 A_7 A_2 0 A_7 A_5 0 0
(selected attributes: A_2, A_5, A_7)

Advantages of the elaborated encoding:

Repeated attributes reduces loss of genetic diversity

Individual´ s length does not depend on the number of attributes being mined

Individual contains info about relative importance of selected attributes

GA for selecting attributes for an ensemble of classifiers

[Guerra-Salcedo & Whitley 99]

Each GA run selects an attribute subset

Each selected attribute subset is used to build one classifier of the ensemble

Genetic Programming for Classification

Standard approach:

terminal set: predicting attributes, random constant generator

function set: mathematical, comparison and logical operators

each individual is a "rule"

Classification: compare the output of the root node against a threshold

For an m class problem, run GP m times: each time we solve a 2-class problem

(one class is the “+” class and all other classes are grouped into a “-” class)

Genetic Programming for Classification

Main problems with standard approach

Closure property requires that all tree nodes return the same data type

Size and complexity of GP trees make them difficult to understand

Non-standard approaches for Classification with GP

"Booleanize" attributes and function set

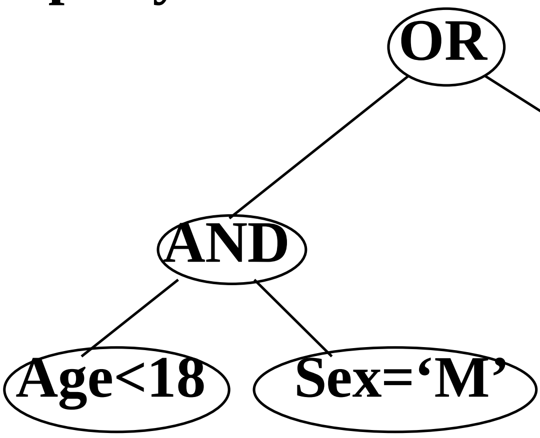
Constrained-syntax GP

Grammar-based GP

Booleanize attributes and use logical operators (and, or) in the function set

[Hu 98], [Eggermont et al 99], [Bojarczuk et al 99]

Property of closure is satisfied



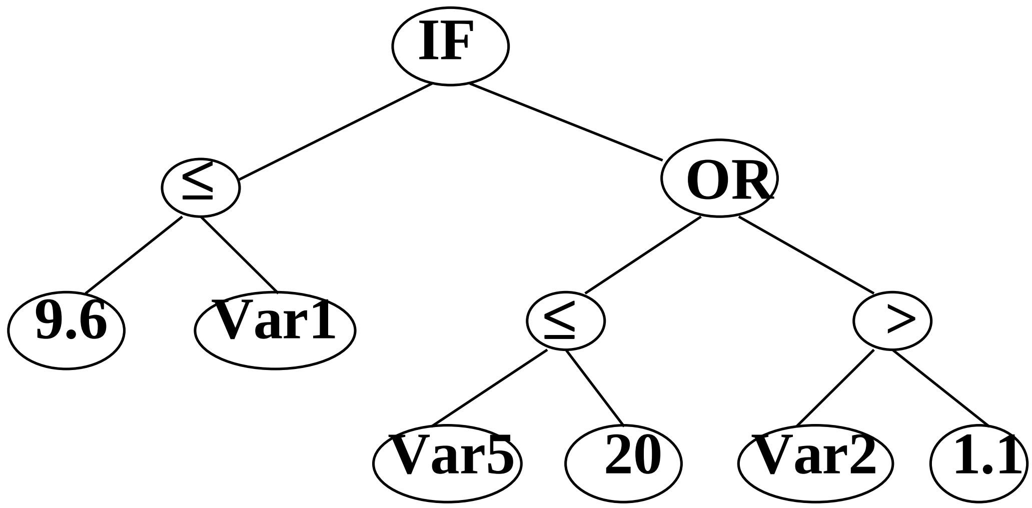
Constrained-syntax GP

For each function used in the function set, specify the type of its arguments and the type of its result

Crossover and mutation are modified to respect the defined restrictions

See e.g. [Bhattacharyya et al. 98]

Functions	datatype of input arguments	datatype of output
+, -, *, /	(real, real)	real
≤, >	(real, real)	boolean
AND, OR	(boolean, boolean)	boolean
IF	(boolean, boolean or real, boolean or real)	boolean



Grammar-based GP

Basic idea: use a grammar to define the format of rules [Wong & Leung 2000]

The placement of a symbol in the tree must be allowed by the grammar

To create an individual, a complete derivation is performed from the start symbol of the grammar

Ex.: predicting attributes: sex, age, x_ray
goal attributes: disease_1, disease_2

Rule → Rule1 | Rule2
Rule1 → if Antec1 then Cons1
Rule2 → if Antec2 then Cons2
Antec1 → Sex_cond and Age_cond
Antec2 → Age_cond and X_ray_cond
Sex_cond → ∅ | "sex = M" | "sex = F"

.

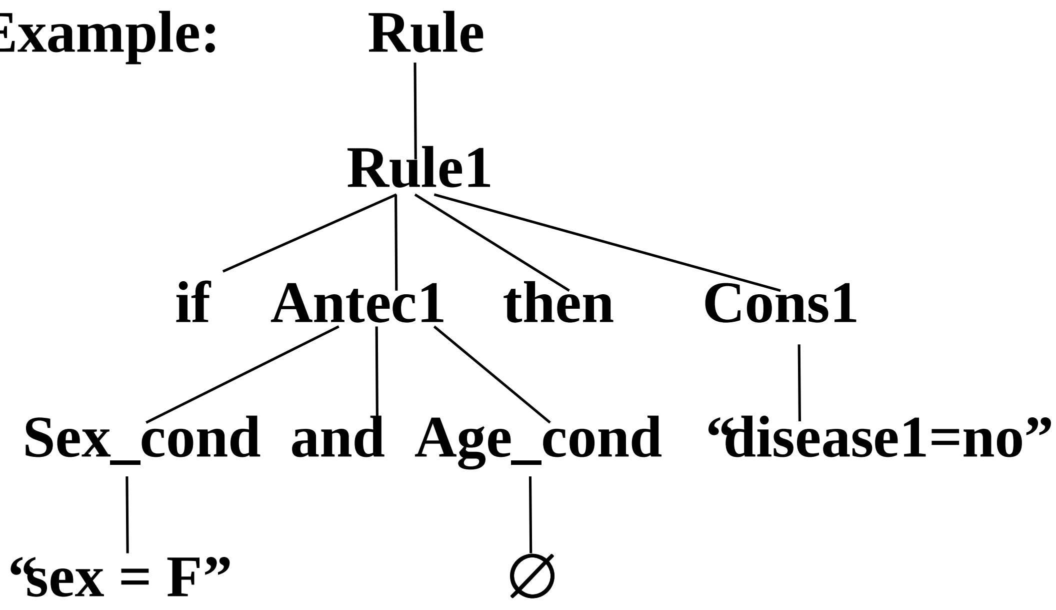
.

Cons1 → Disease1
Disease1 → "disease1=yes" | "disease1=no"

.

.

Example:


Rule
Rule1
if
Antec1
then
Cons1
Sex_cond
and
Age_cond
“disease1=no”
“sex = F”
∅

Advantages of grammar-based GP:

- **uses domain knowledge**
- **avoids the need for closed function set**

Disadvantages of grammar-based GP:

- **grammar is domain-specific**
- **reduces the autonomy of the algorithm to discover novel, surprising knowledge**

Constructive Induction

Motivation - generate "higher-level" attributes, such as:

"Income > Expenditure?"

which can be used to generate rules such as:

IF ("Income > Expenditure?" = "yes") ...

Note that the condition:
(“Income > Expenditure?” = “yes”)

corresponds to an “infinite” number
of conditions of the form:
(Income > value) AND (Expenditure < value)

Constructive induction with GP [Hu 98]

1st step: "booleanize" *all* the attributes

E.g.: values of attribute Age can be divided into 2 groups: Age ≤ v, Age > v
(v is automatically chosen)

2nd step:
apply GP to construct new attributes

Each individual represents an attribute

Terminals: booleanized attributes
Functions: logical operators (and, not)

All terminal and function symbols return boolean values
(meets the closure requirement)

Conclusions

- **Motivation for data mining with GA/GP: to cope with attribute interaction better than local, greedy rule induction methods**

- **Need for task-specific genetic operators**

- **GP's representational power is more useful to construct new attributes and discover novel knowledge**

- **Fitness function should consider accuracy, comprehensibility, and interestigness – which suggests multi-objective algorithms**

References

[Anglano et al. 97] C. Anglano, A. Giordana, G. Lo Bello, L. Saitta. A network genetic algorithm for concept learning. *Proc. 7th Int. Conf. Genetic Algorithms*, 434-441. Morgan Kaufmann, 1997.

[Anglano et al. 98] C. Anglano, A. Giordana, G. Lo Bello, L. Saitta. An experimental evaluation of coevolution learning. *Machine Learning: Proc. 15th Int. Conf. (ICML-98)*, 19-27. Morgan Kaufmann, 1998.

[Araujo et al. 99] D.L.A. Araujo, H.S. Lopes and A.A. Freitas. A parallel genetic algorithm for rule discovery in large databases. *Proc. 1999 IEEE Systems, Man and Cybernetics Conf., v.III*, 940-945. Tokyo, Oct. 1999.

[Bala et al. 95a] J. Bala, J. Huang, H. Vafaie, K. DeJong and H. Wechsler. Hybrid learning using genetic algorithms and decision trees for pattern classification. *Proc. 14th Int. Joint Conf. AI (IJCAI-95)*, 719-724. 1995.

[Bala et al. 95b] J. Bala, K. De Jong, J. Huang, H. Vafaie, H. Wechsler. Using learning to facilitate the evolution of features for recognizing visual concepts. *Special Issue of Evolutionary Computation – Evolution, Learning and Instinct: 100 years of the Baldwin effect.* 1995.

[Bhattacharyya et al. 98] S. Bhattacharyya, O. Pictet, G. Zumbach. Representational semantics for genetic programming based learning in high-frequency financial data. *Genetic Programming 1998: Proc. 3rd Annual Conf.*, 11-16. Morgan Kaufmann, 1998.

[Bhattacharyya 2000a] S. Bhattacharyya. Evolutionary algorithms in data mining: multi-objective performance modeling for direct marketing. *Proc. 6th ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining (KDD-2000)*, 465-473. ACM, 2000.

[Bhattacharyya 2000b] S. Bhattacharyya. Multi-objective data mining using genetic algorithms. In: A.S. Wu (Ed.) *Proc. 2000 Genetic and Evolutionary Computation Conf. Workshop Program – Worshop on Data Mining with Evolutionary Algorithms*, 76-79. 2000.

[Bojarczuk et al. 99] C.C. Bojarczuk, H.S. Lopes and A.A. Freitas. Discovering comprehensible classification rules using genetic programming: a case study in a medical domain. *To appear in Proc. Genetic and Evolutionary Computation Conf. (GECCO-99).* Orlando, FL, USA. July 1999.

[Bramer 96] M. Bramer. Induction of classification rules from examples: a critical review. *Proc. Data Mining'96 Unicom Seminar*, 139-166. London: Unicom, 1996.

[Carvalho & Freitas 2000a] D.R. Carvalho and A.A. Freitas. A hybrid decision tree/genetic algorithm for coping with the problem of small disjuncts in data mining. *Proc. 2000 Genetic and Evolutionary Computation Conf. (GECCO-2000)*, 1061-1068. Morgan Kaufmann, 2000.

[Carvalho & Freitas 2000b] D.R. Carvalho and A.A. Freitas. A genetic algorithm-based solution for the problem of small disjuncts. *Principles of Data Mining and Knowledge Discovery (Proc. 4th European Conf., PKDD-2000). Lecture Notes in Artificial Intelligence 1910*, 345-352. Springer-Verlag, 2000.

[Cherkauer & Shavlik 97] K.J. Cherkauer and J.W. Shavlik. Selecting salient features for machine learning from large candidate pools through parallel decision tree constructions. In: H. Kitano & J.A. Hendler. (Eds.) *Massively Parallel Artificial Intelligence*, 102-136. AAAI Press, 1994.

[DeJong et al. 93] K.A. DeJong, W.M. Spears and D.F. Gordon. Using genetic algorithms for concept learning. *Machine Learning* 13, 161-188, 1993.

[Eggermont et al. 99a] J. Eggermont, A.E. Eiben, and J.I. van Hemert. Adapting the fitness function in GP for data mining. *Proc. EuroGP-99.*

[Eggermont et al. 99b] J. Eggermont, A.E. Eiben, and J.I. van Hemert. A comparison of genetic programming variants for data variants. *Proc. IDA-99 (Intelligent Data Analysis*).

[Emmanouilidis et al. 2000] C. Emmanouilidis, A. Hunter and J. MacIntyre. A multiobjective evolutionary setting for feature selection and a commonality-based crossover operator. *Proc. 2000 Congress on Evolutionary Computation (CEC-2000),* 309-316. IEEE, 2000.

[Falkenauer 98] E. Falkenauer. *Genetic Algorithms and Grouping Problems.* John Wiley & Sons, 1998.

[Flockhart & Radcliffe 95] I.W. Flockhart and N.J. Radcliffe. GA-MINER: parallel data mining with hierarchical genetic algorithms - final report. *EPCC-AIKMS-GA-MINER-Report 1.0*. University of Edinburgh, 1995.

[Freitas & Lavington 98] A.A. Freitas & S.H. Lavington. *Mining Very Large Databases with Parallel Processing.* Kluwer, 1998.

[Freitas 99] A.A. Freitas. (Ed.) *Data Mining with Evolutionary Algorithms: Research Directions – Papers from the AAAI Workshop*. (Proc. AAAI-99 & GECCO-99 workshop.) Technical Report WS-99-06. AAAI Press, 1999.

[Gilbert et al. 98] R.J. Gilbert, R. Goodacre, B. Shann, D.B. Kell, J. Taylor and J.J. Rowland. *Genetic Programming 1998: Proc. 3rd Annual Conf.*, 109-115. Morgan Kaufmann, 1998.

[Giordana et al. 94] A. Giordana, L. Saitta, F. Zini. Learning disjunctive concepts by means of genetic algorithms. *Proc. 11th Int. Conf. Machine Learning (ICML-94)*, 96-104. Morgan Kaufmann, 1994.

[Giordana & Neri 95] A. Giordana and F. Neri. Search-intensive concept induction. *Evolutionary Computation 3(4)*: 375-416, Winter 1995.

[Greene & Smith 93] D.P. Greene & S.F. Smith. Competition-based induction of decision models from examples. *Machine Learning 13,* 229-257. 1993.

[Guerra-Salcedo & Whitley 99] C. Guerra-Salcedo & D. Whitley. Feature selection mechanisms for ensemble creation: a genetic search perspective. *Proc. AAAI-99 & GECCO-99 Workshop on Data Mining with Evolutionary Algorithms: Research Directions. (To be published as an AAAI Technical Report.)*

[Hall et al. 99] L.O. Hall, I.B. Ozyurt, and J.C. Bezdek. Clustering with a genetically optimized approach. *IEEE Trans. on Evolutionary Comp. 3(2)*, 103-112. July 1999.

[Hand 97] D. Hand. *Construction and Assessment of Classification Rules.* John Wiley & Sons, 1997.

[Hu 98] Y.-J. Hu. A genetic programming approach to constructive induction. *Genetic Programming 1998: Proc. 3rd Annual Conf.*, 146-151. Morgan Kaufmann, 1998.

[Janikow 93] C.Z. Janikow. A knowledge-intensive genetic algorithm for supervised learning. *Machine Learning* 13, 189-228. 1993.

[Kim et al. 2000] Y. Kim, W.N. Street and F. Menczer. Feature selection in unsupervised learning via evolutionary search. *Proc. 6th ACM SIGKDD Int. Conf. on Knowledge Discovery and Data Mining (KDD-2000)*, 365-369. ACM, 2000.

[Liu & Kwok 2000] J.J. Liu and J.T.-Y. Kwok. An extended genetic rule induction algorithm. *Proc. 2000 Congress on Evolutionary Computation (CEC-2000)*. 2000.

[Michie et al. 94] D. Michie, D.J. Spiegelhalter and C.C. Taylor. *Machine Learning, Neural and Statistical Classification*. New York: Ellis Horwood, 1994.

[Newson 91] G. Newson. Simpson's paradox revisited. *The Mathematical Gazette*, 75(473), Oct./91, 290-293.

[Noda et al. 99] E. Noda, A.A. Freitas and H.S. Lopes. Discovering interesting prediction rules with a genetic algorithm. *To appear in Proc. Conf. on Evolutionary Computation – 1999 (CEC-99)*. Washington, D.C., USA, July 1999.

[Park & Song 98] Y-J. Park & M-S. Song. A genetic algorithm for clustering problems. *Genetic Programming 1998: Proc. 3rd Annual Conf.*, 568-575. Morgan Kaufmann, 1998.

[Vafaie & De Jong 93] H. Vafaie & K. De Jong. Robust feature selection algorithms. *Proc. 1993 IEEE Int. Conf. on Tools with AI*, 356-363. Nov. 1993.

[Wagner 82] C.H. Wagner. Simpson's Paradox in Real Life. *The American Statistician*, 36(1), Feb./82, 46-48.

[Wong & Leung 2000] M.L. Wong and K.S. Leung. *Data Mining Using Grammar Based Genetic Programming and Applications*. Kluwer, 2000.

[Wu 2000] A. Wu. (Eds.) Proc. 2000 Genetic and Evolutionary Computation Conf. (GECCO-2000) Workshop Program – Workshop on Data Mining with Evolutionary Algorithms, pp. 71-92. 2000.