Genetic Algorithms
Simple Examples

Cse537
Artificial Intelligence
Professor Anita Wasilewska
Genetic Programming

A program in C

- int foo (int time)
  {
    int temp1, temp2;
    if (time > 10)
      temp1 = 3;
    else
      temp1 = 4;
    temp2 = temp1 + 1 + 2;
    return (temp2);
  }

- Equivalent expression (similar to a classification rule in data mining):
  
  
  (+ 1 2 (IF (> TIME 10) 3 4))
Program tree

\((+ 1 2 (IF (> \text{TIME} 10) 3 4))\)
Given data

<table>
<thead>
<tr>
<th>Input: Independent variable X</th>
<th>Output: Dependent variable Y</th>
</tr>
</thead>
<tbody>
<tr>
<td>-1.00</td>
<td>1.00</td>
</tr>
<tr>
<td>-0.80</td>
<td>0.84</td>
</tr>
<tr>
<td>-0.60</td>
<td>0.76</td>
</tr>
<tr>
<td>-0.40</td>
<td>0.76</td>
</tr>
<tr>
<td>-0.20</td>
<td>0.84</td>
</tr>
<tr>
<td>0.00</td>
<td>1.00</td>
</tr>
<tr>
<td>0.20</td>
<td>1.24</td>
</tr>
<tr>
<td>0.40</td>
<td>1.56</td>
</tr>
<tr>
<td>0.60</td>
<td>1.96</td>
</tr>
<tr>
<td>0.80</td>
<td>2.44</td>
</tr>
<tr>
<td>1.00</td>
<td>3.00</td>
</tr>
</tbody>
</table>

Citation: www.genetic-programming.com/c2003lecture1modified.ppt
## Problem description

<p>| | | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Objective:</strong></td>
<td>Find a computer program with one input (independent variable $X$) whose output $Y$ equals the given data</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td><strong>Terminal set:</strong></td>
<td>$T = {X, \text{Random-Constants}}$</td>
</tr>
<tr>
<td>2</td>
<td><strong>Function set:</strong></td>
<td>$F = {+, -, *, /}$</td>
</tr>
<tr>
<td>3</td>
<td><strong>Initial population:</strong></td>
<td>Randomly created individuals from elements in $T$ and $F$.</td>
</tr>
<tr>
<td>4</td>
<td><strong>Fitness:</strong></td>
<td>$</td>
</tr>
<tr>
<td>5</td>
<td><strong>Termination:</strong></td>
<td>An individual emerges whose sum of absolute errors (the value of its fitness function) is less than 0.1</td>
</tr>
</tbody>
</table>

Citation: [www.genetic-programming.com/c2003lecture1modified.ppt](http://www.genetic-programming.com/c2003lecture1modified.ppt)
Generation 0

Population of 4 randomly created individuals

(a) \(x + 1\)

(b) \(x^2 + 1\)

(c) 2

(d) \(x\)

Citation: examples taken from: www.genetic-programming.com/c2003lecture1modified.ppt
Mutation

Mutation:

picking “2” as mutation point

Citation: part of the pictures used as examples are taken from: www.genetic-programming.com/c2003lecture1modified.ppt
Crossover

Crossover:

picking “+” subtree and leftmost “x” as crossover points

Citation: example taken from: www.genetic-programming.com/c2003lecture1modified.ppt
Copy of (a) 

Mutant of (c) picking “2” as mutation point 

First offspring of crossover of (a) and (b) picking “+” of parent (a) and left-most “x” of parent (b) as crossover points 

Second offspring of crossover of (a) and (b) picking “+” of parent (a) and left-most “x” of parent (b) as crossover points 

Citation: part of the examples is taken from: www.genetic-programming.com/c2003lecture1modified.ppt
| X   | Y    | X+1 | |X+1-Y| |1| |1-Y| |X   | |X-Y| |X^2+X+1| |X^2+X+1-Y| |
|-----|------|-----|------|-----|-----|-----|-----|-----|-----|-----|-----|-----|-----|
| -1.00 | 1.00 | 0   | 1    |       | 1   | 0   | -1.00 | 2   | 1   | 0   |       |       |       |
| -0.80 | 0.84 | 0.20 | 0.64 |       | 1   | 0.16 | -0.80 | 1.64 | 0.84 | 0   |       |       |       |
| -0.60 | 0.76 | 0.40 | 0.36 |       | 1   | 0.24 | -0.60 | 1.36 | 0.76 | 0   |       |       |       |
| -0.40 | 0.76 | 0.60 | 0.16 |       | 1   | 0.24 | -0.40 | 1.16 | 0.76 | 0   |       |       |       |
| -0.20 | 0.84 | 0.80 | 0.04 |       | 1   | 0.16 | -0.20 | 1.04 | 0.84 | 0   |       |       |       |
| 0.00  | 1.00 | 1.00 | 0    |       | 1   | 0   | 0.00  | 1   | 1   | 0   |       |       |       |
| 0.20  | 1.24 | 1.20 | 0.04 |       | 1   | 0.24 | 0.20  | 1.04 | 1.24 | 0   |       |       |       |
| 0.40  | 1.56 | 1.40 | 0.16 |       | 1   | 0.56 | 0.40  | 1.16 | 1.56 | 0   |       |       |       |
| 0.60  | 1.96 | 1.60 | 0.36 |       | 1   | 0.96 | 0.60  | 1.36 | 1.96 | 0   |       |       |       |
| 0.80  | 2.44 | 1.80 | 0.64 |       | 1   | 1.44 | 0.80  | 1.64 | 2.44 | 0   |       |       |       |
| 1.00  | 3.00 | 2.00 | 1    |       | 1   | 2   | 1.00  | 2   | 3   | 0   |       |       |       |

**Fitness**

: 4.40 6.00 15.40 0.00 10

**Found!**
GA and Classification

**Classify** customers based on number of children and salary:

<table>
<thead>
<tr>
<th>Parameter</th>
<th># of children (NOC)</th>
<th>Salary (S)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Domain</td>
<td>0…10</td>
<td>0…500000</td>
</tr>
<tr>
<td>Syntax of atomic expression</td>
<td>NOC = x</td>
<td>S = x</td>
</tr>
<tr>
<td></td>
<td>NOC &lt; x</td>
<td>S &lt; x</td>
</tr>
<tr>
<td></td>
<td>NOC &lt;= x</td>
<td></td>
</tr>
<tr>
<td></td>
<td>NOC &gt; x</td>
<td>S &gt; x</td>
</tr>
<tr>
<td></td>
<td>NOC &gt;= x</td>
<td></td>
</tr>
</tbody>
</table>
GA and Classification Rules

• A classification rule is of the form

\[
\text{IF description THEN class=c;}
\]

Antecedent    Consequence
Formula representation

- Possible rule:
  - If (NOC = 2) AND (S > 80000) then GOOD (customer)
## Initial data table

<table>
<thead>
<tr>
<th>Nr. Crt.</th>
<th>Number of children (NOC)</th>
<th>Salary (S)</th>
<th>Type of customer (C)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>2</td>
<td>&gt; 80000</td>
<td>GOOD</td>
</tr>
<tr>
<td>2</td>
<td>1</td>
<td>&gt; 30000</td>
<td>GOOD</td>
</tr>
<tr>
<td>3</td>
<td>0</td>
<td>= 50000</td>
<td>GOOD</td>
</tr>
<tr>
<td>4</td>
<td>&gt; 2</td>
<td>&lt; 10000</td>
<td>BAD</td>
</tr>
<tr>
<td>5</td>
<td>= 10</td>
<td>= 30000</td>
<td>BAD</td>
</tr>
<tr>
<td>6</td>
<td>= 5</td>
<td>&lt; 30000</td>
<td>BAD</td>
</tr>
</tbody>
</table>
Initial data represented as rules

- Rule 1: If (NOC = 2) AND (S > 80000) then C = GOOD
- Rule 2: If (NOC = 1) AND (S > 30000) then C = GOOD
- Rule 3: If (NOC = 0) AND (S = 50000) then C = GOOD
- Rule 4: If (NOC > 2) AND (S < 100000) then C = BAD
- Rule 5: If (NOC = 10) AND (S = 30000) then C = BAD
- Rule 6: If (NOC = 5) AND (S < 30000) then C = BAD
Generation 0

• Population of 3 randomly created individuals:
  
  – If (NOC > 3) AND (S > 10000) then C = GOOD
  – If (NOC > 1) AND (S > 30000) then C = GOOD
  – If (NOC >= 0) AND (S < 40000) then C = GOOD

• We want to find a more general (if it is possible the most general) characteristic description for class GOOD

• We want to assign predicted class GOOD for all individuals
Generation 0

Individual 1
>NOC > 3
> S > 10000

(NOC > 3) AND (S > 10000)

Individual 2
>NOC > 1
> S > 30000

(NOC > 1) AND (S > 30000)

Individual 3
>NOC >= 0
> S < 40000

(NOC >= 0) AND (S < 40000)
Fitness function

• For a rule IF A THEN C

CF (Confidence factor) = \frac{|A \cup C|}{|A|}

|A| = number of records that satisfy A
|A \cup C| = number of records that satisfy A and are in predicted class C

Citation: the confidence formula is taken from class slides: http://www.cs.sunysb.edu/~cse634/lecture_notes/07association.pdf
Fitness function – Generation 0

Rule 1: If (NOC = 2) AND ( S > 80000) then GOOD
Rule 2: If (NOC = 1) AND ( S > 30000) then GOOD
Rule 3: If (NOC = 0) AND ( S = 50000) then GOOD
Rule 4: If (NOC > 2) AND ( S < 10000) then BAD
Rule 5: If (NOC = 10) AND ( S = 30000) then BAD
Rule 6: If (NOC = 5) AND ( S < 30000) then BAD

Fitness of Individual 1: If (NOC > 3) AND ( S > 10000) then GOOD
   |A| = 2 (Rule 5 & 6), |AUC| = 0, CF = 0 / 2 = 0
Fitness of Individual 2: If (NOC > 1) AND ( S > 30000) then GOOD
   |A| = 1 (Rule 1), |AUC| = 1, CF = 1 / 1 = 1  Best in Gen 0
Fitness of Individual 3: If (NOC >= 0) AND ( S < 40000) then GOOD
   |A| = 4 (Rule 2 & 4 & 5 & 6), |AUC| = 1, CF = 1 / 4 = 0.25
Mutation

\[(\text{NOC} \geq 0) \land (\text{S} < 40000)\]

\[(\text{NOC} > 0) \land (\text{S} < 90000)\]
Crossover

(NOC > 1) AND (S > 30000)

(NOC >= 0) AND (S > 30000)

(NOC > 1) AND (S < 40000)

(NOC >= 0) AND (S < 40000)

(NOC >= 0) AND (S > 30000)

(NOC >= 0) AND (S > 30000)
(NOC > 1) AND (S < 40000)
Fitness function – Generation 1

Rule 1: If (NOC = 2) AND (S > 80000) then GOOD
Rule 2: If (NOC = 1) AND (S > 30000) then GOOD
Rule 3: If (NOC = 0) AND (S = 50000) then GOOD
Rule 4: If (NOC > 2) AND (S < 10000) then BAD
Rule 5: If (NOC = 10) AND (S = 30000) then BAD
Rule 6: If (NOC = 5) AND (S < 30000) then BAD

Individual 1: If (NOC > 1) AND (S < 40000) then GOOD
|A| = 2 (Rule 4 & 5 & 6), |A&C| = 0, CF = 0 / 2 = 0

Individual 2: If (NOC >= 0) AND (S > 30000) then GOOD
|A| = 3 (Rule 1 & 2 & 3), |A&C| = 3, CF = 3 / 3 = 1

Individual 3: If (NOC > 0) AND (S < 90000) then GOOD
|A| = 5 (Rule 1 & 2 & 4 & 5 & 6), |A&C| = 1, CF = 1 / 5 = 0.2

Best in Gen 1
GA Rules Problem

- When GAs are used for optimization, the goal is typically to return a single value - the best solution found to date

- The entire population ultimately converges to the neighborhood of a single solution

- Sometimes GAs employ a special method called a niching method that makes them capable of finding and maintaining multiple rules
APPLICATION EXAMPLE

Technical Document of

LBS Capital Management, Inc., Clearwater, Florida

Link: http://nas.cl.uh.edu/boetticher/ML_DataMining/mahfoud96financial.pdf
Forecasting Individual Stock Performance

- **GOAL:** using historical data of a stock, *predict* relative return for a quarter

Example: If IBM stock is up 5% after one quarter and the S&P 500 index is up 3% over the same period, then IBM’s relative return is +2%

-The Implementation Example consists of 15 attributes of a stock at specific points in time and the relative return for the stock over the subsequent 12 week time period.

- **200 to 600 (records) examples** were utilized depending on the experiment and the data available for a particular stock

**GOAL:** Combination of rules is required to model relationships among financial variables

Example: **Rule-1:** IF [P/E > 30 ] THEN Sell

**Rule-2:** IF [P/E < 40 and Growth Rate > 40%] THEN Buy
Preliminary Experiments

• For Preliminary set of experiments, to predict the return, relative to the market, a Madcap stock was randomly selected from the S&P 400

• 331 examples (records) present in the database of examples of stock X

• 70% of examples (records) were used as a training set for the GA

• 20% of the examples (records) were used as a stopping set, to decide which population is best

• 10% of the examples (records) were used to measure performance

• A sample rule that the GA generated in one of the experiments:
  IF [Earning Surprise Expectation > 10% and Volatility > 7%] and […]
  THEN Prediction = Up

• Same set of experiments were used using Neural Network with one layer of hidden nodes using Backpropagation algorithm with the same training, stopping and test sets as that of GA experiment
Observations on the Results

• The **GA** correctly predicts the direction of stock relative to the market 47.6% of the time and incorrectly predicts the 6.6% of time and produces no prediction 45%

• Over **half of the time** (47.6% + 6.6%), the **GA** makes a prediction

• When it **does make a prediction**, **GA** is correct 87.8% of the time

• The **Neural Network correctly predicts** the direction relative to the market 79.2% of the time and incorrectly predicts direction 15.8% of the time.

• When it **does make a prediction**, the **NN** is correct 83.4%
Comparison with Neural Networks

• **Advantage** of GA’s over NN’s:
  1. GA has ability to output comprehensible rules
  2. GA provides rough explanation of the concepts learned by black-box approaches such as NN’s
  3. GA learns rules that are subsequently used in a formal expert system

• 3. GA makes no prediction when data is uncertain as opposed to Neural Network