Genetic algorithm for variable selection

Jaakko Väyrynen
jaakko.j.vayrynen@tkk.fi

T-61.6040 Variable selection for regression
7.11.2006

References

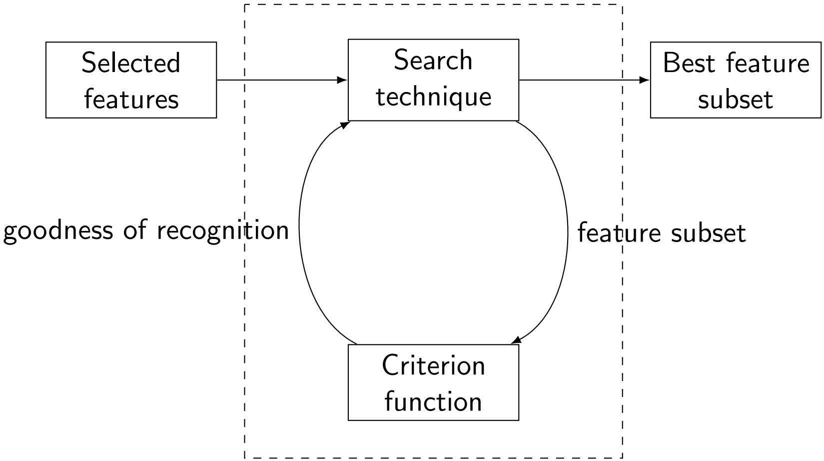
12 H. Vafaie. Robust feature selection algorithms. In Proc. 5th Intl. Conf. on Tools with Artificial Intelligence, 1993.

13 C. R. Houck, J. A. Joines and M. G. Kay. A genetic algorithm for function optimization: a matlab implementation. Technical report, North Carolina State University, 1996.

Feature selection techniques

- Tradeoff between accuracy and performance
- Search for "optimal" subset of features
- Problem specific heuristics (filter methods)
  - Domain knowledge is available
  - Prune search space based on domain knowledge
- General heuristics (wrapper methods)
  - Domain knowledge is unavailable or costly to exploit
  - Mainly hill-climbing algorithms

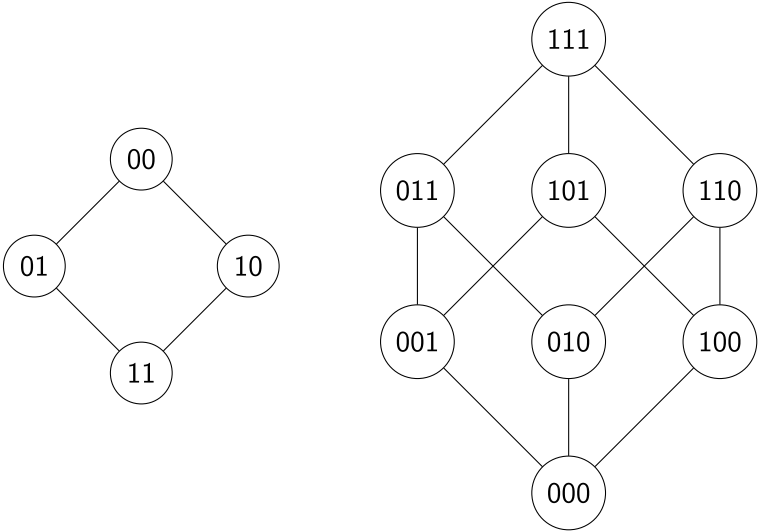
Feature selection process for wrapper approach



Sequential backward selection (SBS) algorithm

- Try removing each remaining feature one at a time
- Permanently remove the feature giving the least improvement
- Measure improvement with domain specific criterion function
- Greedy algorithm
  - Early mistakes cannot be canceled
  - Easily gets stuck at local optima
  - Does not consider effects of removing a set of features
- Running time $\mathcal{O}(n^2)$

Greedy feature selection lattice



Source of brittleness

- $f(x_0, x_1, x_2) = ax_0 + bx_1 + cx_2 + dx_0x_1 + ex_0x_2 + fx_1x_2 + gx_0x_1x_2$

Source of brittleness

- $f(x_0, x_1, x_2) = ax_0 + bx_1 + cx_2 + dx_0x_1 + ex_0x_2 + fx_1x_2 + gx_0x_1x_2$
- With $b = c, d = e = 0$

Source of brittleness

- $f(x_0, x_1, x_2) = ax_0 + bx_1 + bx_2 + fx_1x_2 + gx_0x_1x_2$
- With $b = c, d = e = 0$
- Value assignments to all feature subsets

x_0, x_1, x_2	$f(x_0, x_1, x_2)$	constraint
000	0	
100	a	(global optimum)
001	b	
010	b	
011	$2b + f$	(local optimum)
101	$a + b$	
110	$a + b$	
111	$a + 2b + f + g$	

Source of brittleness

- $f(x_0, x_1, x_2) = ax_0 + bx_1 + bx_2 + fx_1x_2 + gx_0x_1x_2$
- Value assignments to all feature subsets

x_0, x_1, x_2	$f(x_0, x_1, x_2)$	constraint
000	0	
100	a	(global optimum)
001	b	
010	b	
011	$2b + f$	(local optimum)
101	$a + b$	
110	$a + b$	
111	$a + 2b + f + g$	

- Let $m = 2b + f$

Source of brittleness

- $f(x_0, x_1, x_2) = ax_0 + bx_1 + bx_2 + fx_1x_2 + gx_0x_1x_2$
- Value assignments to all feature subsets

x_0, x_1, x_2	$f(x_0, x_1, x_2)$	constraint
000	0	
100	a	(global optimum)
001	b	
010	b	
011	m	(local optimum)
101	$a + b$	
110	$a + b$	
111	$a + m + g$	

- Let $m = 2b + f$

Source of brittleness

- $f(x_0, x_1, x_2) = ax_0 + bx_1 + bx_2 + fx_1x_2 + gx_0x_1x_2$
- Value assignments to all feature subsets

x_0, x_1, x_2	$f(x_0, x_1, x_2)$	constraint
000	0	
100	a	(global optimum)
001	b	
010	b	
011	m	(local optimum)
101	$a + b$	
110	$a + b$	
111	$a + m + g$	

- Let $m = 2b + f$
- Global optimum implies that
  $a > 0, a > b, a > m, a > a + b, \dots$
  $a > 0 \wedge a > a + b \Rightarrow b < 0$

Source of brittleness

- $f(x_0, x_1, x_2) = ax_0 + bx_1 + bx_2 + fx_1x_2 + gx_0x_1x_2$
- Value assignments to all feature subsets

x_0, x_1, x_2	$f(x_0, x_1, x_2)$	constraint
000	0	
100	a	(global optimum)
001	b	
010	b	
011	m	(local optimum)
101	$a + b$	
110	$a + b$	
111	$a + m + g$	

- Let $m = 2b + f$
- Global optimum implies that
  $a > 0, a > b, a > m, a > a + b, \ldots$
  $a > 0 \wedge a > a + b \Rightarrow b < 0$
- Local optimum implies that $m > a + b$, $g < -a$

Source of brittleness

- $f(x_0, x_1, x_2) = ax_0 + bx_1 + bx_2 + f x_1 x_2 + g x_0 x_1 x_2$
- Value assignments to all feature subsets

x_0, x_1, x_2	$f(x_0, x_1, x_2)$	constraint
000	0	
100	a	(global optimum)
001	b	
010	b	
011	m	(local optimum)
101	$a + b$	
110	$a + b$	
111	$a + m + g$	

- Let $m = 2b + f$, $b = -4, f = 9$
- Global optimum implies that
  $a > 0, a > b, a > m, a > a + b, \ldots$
  $a > 0 \land a > a + b \Rightarrow b < 0$
- Local optimum implies that $m > a + b$, $g < -a$

Source of brittleness

- $f(x_0, x_1, x_2) = ax_0 - 4x_1 - 4x_2 + 9x_1x_2 + gx_0x_1x_2$
- Value assignments to all feature subsets

x_0, x_1, x_2	$f(x_0, x_1, x_2)$	constraint
000	0	
100	a	(global optimum)
001	-4	
010	-4	
011	m	(local optimum)
101	$a-4$	
110	$a-4$	
111	$a+m+g$	

- Let $m = 1$, $b = -4, f = 9$
- Global optimum implies that
  $a > 0, a > b, a > m, a > a + b, \ldots$
  $a > 0 \wedge a > a + b \Rightarrow b < 0$
- Local optimum implies that $m > a + b$, $g < -a$

Source of brittleness

- $f(x_0, x_1, x_2) = ax_0 - 4x_1 - 4x_2 + 9x_1x_2 + gx_0x_1x_2$
- Value assignments to all feature subsets

x_0, x_1, x_2	$f(x_0, x_1, x_2)$	constraint
000	0	
100	a	(global optimum)
001	-4	
010	-4	
011	m	(local optimum)
101	$a-4$	
110	$a-4$	
111	$a+m+g$	

- Let $m = 1$, $b = -4, f = 9$
- Global optimum implies that
  $a > 0, a > b, a > m, a > a + b, \ldots$
  $a > 0 \wedge a > a + b \Rightarrow b < 0$
- Local optimum implies that $m > a + b$, $g < -a$

Source of brittleness

- $f(x_0, x_1, x_2) = ax_0 - 4x_1 - 4x_2 + 9x_1x_2 + gx_0x_1x_2$
- Value assignments to all feature subsets

x_0, x_1, x_2	$f(x_0, x_1, x_2)$	constraint
000	0	
100	a	(global optimum)
001	-4	
010	-4	
011	1	(local optimum)
101	$a - 4$	
110	$a - 4$	
111	$a + 1 + g$	

- Let $m = 1$, $b = -4, f = 9$
- Global optimum implies that
  $a > 0, a > b, a > m, a > a + b, \ldots$
  $a > 0 \wedge a > a + b \Rightarrow b < 0$
- Local optimum implies that $m > a + b$, $g < -a$

Source of brittleness

- $f(x_0, x_1, x_2) = ax_0 - 4x_1 - 4x_2 + 9x_1x_2 + gx_0x_1x_2$
- Value assignments to all feature subsets

x_0, x_1, x_2	$f(x_0, x_1, x_2)$	constraint
000	0	
100	a	(global optimum)
001	-4	
010	-4	
011	1	(local optimum)
101	$a-4$	
110	$a-4$	
111	$a+1+g$	

- Let $m = 1$, $b = -4, f = 9$, $g = -5$
- Global optimum implies that
  $a > 0, a > b, a > m, a > a + b, \ldots$
  $a > 0 \wedge a > a + b \Rightarrow b < 0$
- Local optimum implies that $m > a + b$, $g < -a$

Source of brittleness

- $f(x_0, x_1, x_2) = ax_0 - 4x_1 - 4x_2 + 9x_1x_2 - 5x_0x_1x_2$
- Value assignments to all feature subsets

x_0, x_1, x_2	$f(x_0, x_1, x_2)$	constraint
000	0	
100	a	(global optimum)
001	-4	
010	-4	
011	1	(local optimum)
101	$a - 4$	
110	$a - 4$	
111	$a + 1 - 5$	

- Let $m = 1$, $b = -4, f = 9$, $g = -5$
- Global optimum implies that
  $a > 0, a > b, a > m, a > a + b, \dots$
  $a > 0 \wedge a > a + b \Rightarrow b < 0$
- Local optimum implies that $m > a + b,\ g < -a$

Source of brittleness

- $f(x_0, x_1, x_2) = ax_0 - 4x_1 - 4x_2 + 9x_1x_2 - 5x_0x_1x_2$
- Value assignments to all feature subsets

x_0, x_1, x_2	$f(x_0, x_1, x_2)$	constraint
000	0	
100	a	(global optimum)
001	-4	
010	-4	
011	1	(local optimum)
101	$a-4$	
110	$a-4$	
111	$a-4$	

- Let $m = 1$, $b = -4, f = 9$, $g = -5$
- Global optimum implies that
  $a > 0, a > b, a > m, a > a + b, \ldots$
  $a > 0 \wedge a > a + b \Rightarrow b < 0$
- Local optimum implies that $m > a + b$, $g < -a$
- Constraints apply when $1 < a < 5$

Genetic algorithm (GA)

- Motivation from genetics
- Representations for solutions (chromosomes)
- Maintains a number of solutions (population)
- Performs search in the space of solutions (evolution)
  - Combines solutions (chromosome mixing)
  - Inserts/modifies random properties (mutation)
  - Better solutions reproduce more likely ("survival of the fittest")
- Does not require continuity etc.
- Has been shown to find good solutions to hard problems
- No proof of convergence
- Is usually slow
- Heuristic implementation for each problem

Basic genetic algorithm

1. Initial population P_0 of N individuals with fitness values
2. $G \leftarrow 1$
3. $P'_G \leftarrow selection\_function(P_G - 1)$
4. $P_G \leftarrow reproduction\_function(P'_G)$
5. $F_G \leftarrow evaluate\_fitness(P_G)$
6. $G \leftarrow G + 1$
7. Repeat step 3 until termination
8. Print out best solution found

- How to represent solutions?
- How to initialize?
- How to evaluate fitness of each individual?
- How to select individuals for reproduction?
- How to accomplish reproduction?
- When to terminate?

Solution representation

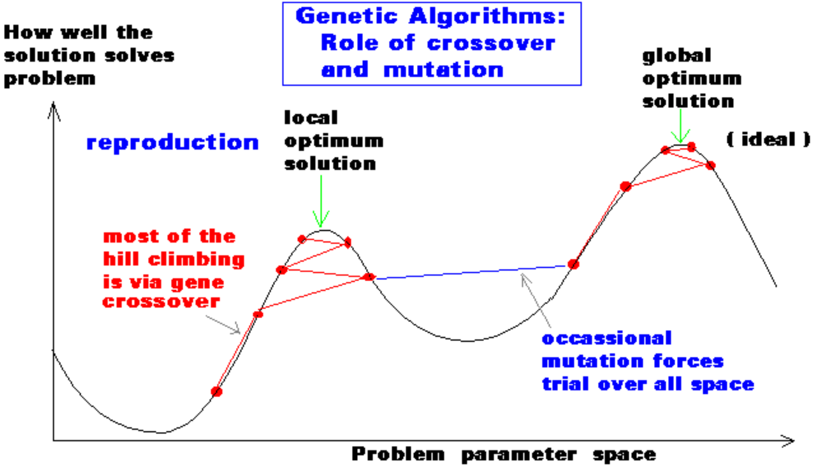
- Structure of the solution
- Linked to reproduction (genetic operations)
- An individual is represented by (a sequence of) genes from an alphabet
- Alphabet: binary, (bounded) floats, integers, sets, symbols, matrices, graphs, model parameters, etc.

Selection function

- Defines which individuals produce next generations
- Good individuals have better chance to get selected
- Techniques: roulette wheel, scaling techniques, tournament, elitist models, ranking models
- Probability P_i for the selection of i:th individual
  - Roulette wheel $P_i = \frac{F_i}{\sum_j F_j}$, with fitness F_i
  - Windowing and scaling allows minimization and negativity
  - Normalized ranking $P_i = q'(1-q)^{r-1}$
    - $q =$ probability of selecting the best individual
    - $r =$ the rank of the individual, where 1 is the best
    - $N =$ the population size
    - $q' = \frac{q}{1-(1-q)^N}$
- Tournament selection
  - Select j items randomly (with replacement)
  - Insert the best of j into the new population
  - Repeat until N individuals have been selected

Genetic operators

- Search mechanism in GA
- Creates new solutions based on existing solutions
- Implementation depends on chromosome representation
- Crossover takes two solutions and produces two new solutions
- Mutation takes one solution and produces one new solution



from http://www.sussex.ac.uk/space-science/ga.html

Crossover and mutation for binary valued vectors

- Let $\bar{X} = (x_1, ..., x_m)$ and $\bar{Y} = (y_1, ..., y_m)$ be two selected m-dimensional binary-valued row vectors
- Mutation flips each bit with probability p_m:
$$x_i' = \begin{cases} 1 - x_i, & \text{if } U(0,1) < p_m \\ x_i, & \text{otherwise} \end{cases}$$
- Inversion is mutation with probability $p_m = 1$
- Crossover splits and joins the parents from bit $r = U(1, m)$:
$$x_i' = \begin{cases} x_i, & \text{if } i < r \\ y_i, & \text{otherwise} \end{cases} \text{ and } y_i' = \begin{cases} y_i, & \text{if } i < r \\ x_i, & \text{otherwise} \end{cases}$$
- Variations: single split, multiple splits, different splits

Crossover and mutation for bounded floats

- Let $\bar{X}$ and $\bar{Y}$ be two selected m-dimensional float-valued row vectors with lower bound a_i and upper bound b_i for each variable, and j is a randomly selected variable, $r = U(0, 1)$
- Arithmetic crossover: $\begin{array}{l} \bar{X}' = r\bar{X} + (1-r)\bar{Y} \\ \bar{Y}' = (1-r)\bar{X} + r\bar{Y} \end{array}$
- Heuristic crossover, with $F(\bar{X}) > F(\bar{Y})$: $\begin{array}{l} \bar{X}' = \bar{X} + r(\bar{X} - \bar{Y}) \\ \bar{Y}' = \bar{X} \end{array}$

  if $x_i' \geq a_i, x_i' \leq b_i \quad \forall i$, otherwise recalculate at most t times
- Uniform mutation: $x_i' = \begin{cases} U(a_i, b_i), & \text{if } i = j \\ x_i, & \text{otherwise} \end{cases}$

- Boundary mutation changes value to its upper or lower bound:

$$x_i' = \begin{cases} a_i, & \text{if } i = j, r < 0.5 \\ b_i, & \text{if } i = j, r \geq 0.5 \\ x_i, & \text{otherwise} \end{cases}$$

- Non-uniform mutation:

$$x_i' = \begin{cases} x_i + (b_i - x_i)f(G), & \text{if } i = j, r_1 < 0.5 \\ x_i - (x_i + a_i)f(G), & \text{if } i = j, r_1 \geq 0.5 \\ x_i, & \text{otherwise} \end{cases}$$

$$f(G) = (r_2(1 - \frac{G}{G_{max}}))^b$$

where

$$\begin{aligned} r_k &= U(0,1),\ k \in \{0,1\} \\ G &= \text{the current generation number} \\ G_{max} &= \text{the maximum number of generations} \\ b &= \text{a shape parameter} \end{aligned}$$

- Multi-non-uniform mutation

Initialization, termination and evaluation

- Selection of initial population P_0
  - Random solutions from the search space
  - Approximate solutions found with other methods
  - Solutions from experts
  - Mixtures of these
- Termination criterion
  - Maximum number of generations G_{max}
  - Population convergence criteria
  - Lack of improvement within a number of generations
  - External evaluation measure
- Fitness evaluation functions
  - Minimal requirement: map the population into a partially ordered set

Genetic algorithm for variable selection

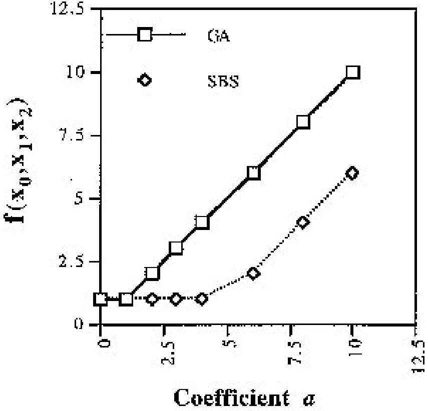
- Representation: Binary string where 1 is for selected and 0 for unselect variable
- Initialization: Random
- Evaluation: Cost function with selected variables
- Selection: Any binary selection method
- Genetic operators: Binary mutation and crossover (and inversion)
- Termination: Any

Compare SBS and GA with toy data

- $f(x_0, x_1, x_2) = ax_0 - 4x_1 - 4x_2 + 9x_1x_2 - 5x_0x_1x_2$
- If $0 < a < 1$, 011 is the global optimum
- If $a > 1$, global optimum is 100 with value a
- If $1 < a < 5$, 011 is the local optimum
- If $a \geq 5$, 111 is a local optimum
- GENESIS program was used with parameters
  - Population size 50
  - Crossover rate 0.6
  - Mutation rate 0.001 for the new generation

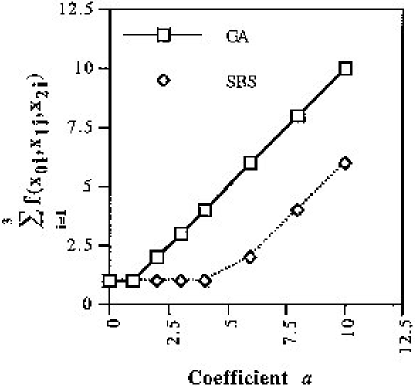
Toy data: add undepended features

- Add additional 27 features to create search space of 2^{30}
- Vary a



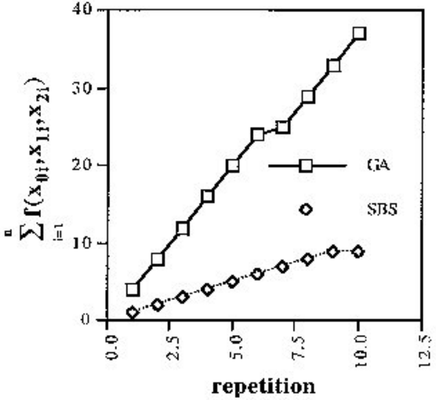
Toy data: replicate depended features

- Replicate depended features 3 times
- Add additional 21 features to create search space of 2^{30}
- Vary a



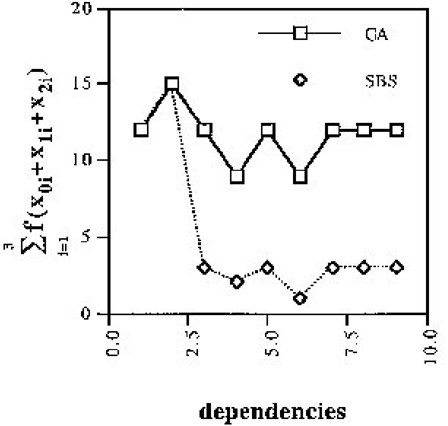
Toy data: number or replications

- Let $a = 4$
- Vary number of replications n of depended features
- Add additional features to create search space of 2^{30}



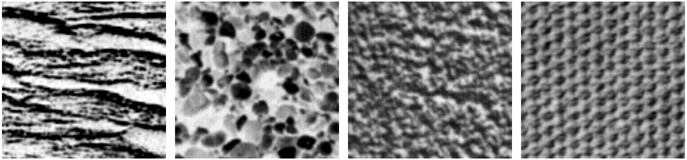
Toy data: number or dependencies

- Let $a = 4$
- Vary number of m-feature dependencies by overlapping 3-feature dependencies

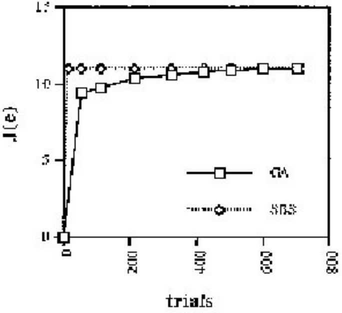


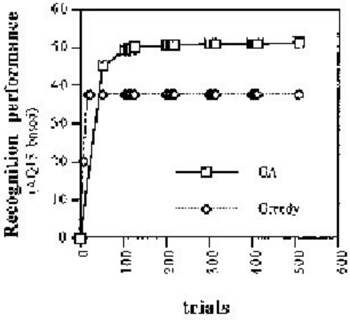
Compare SBS and GA with texture classification task

- Four textures from Brodatz album of textures
- 18 features
- 100 samples of size 30×30 pixels from each image
- Goal: optimal feature set for AQ15 for inducing texture classification rules
- Two evaluation functions
  - Heuristic evaluation function as the maximal separation of classes
  - Exact evaluation by first training AQ15 and testing classification accuracy on test data



Texture classification experiment results





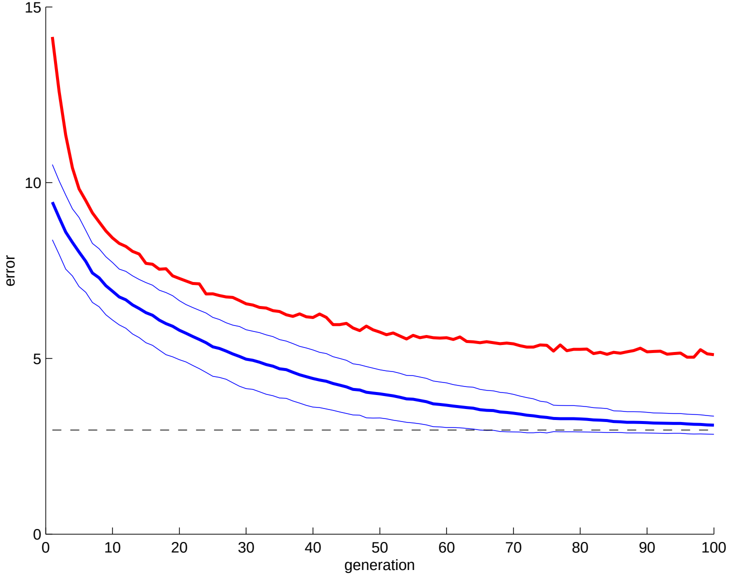
Implementation of GA for variable selection

- Implemented in Matlab with
  - Binary representation
  - Different cost (fitness) functions
  - Roulette wheel selection
  - Survival of the elite
  - Reproduction: binary mutation and crossover
  - Termination after N generations
- Problems
  - Slow with KNN and lots of samples and large population

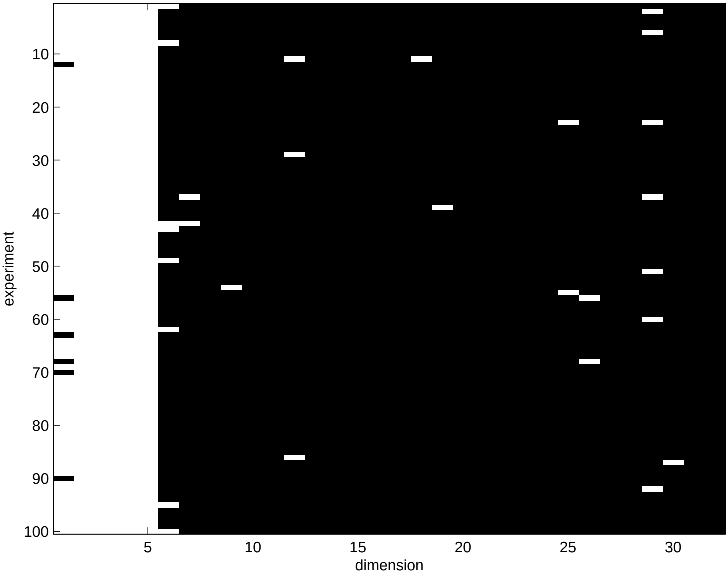
Experiment with implementation

- $y = x_1 - 2x_2 - 3x_3 + 2x_4x_5 + 0.1z$
  $x_6 = x_1x_2$
  $x_7 = x_2x_9$
  $x_8 = x_3x_{10}$
  $z, x_i \sim \mathcal{N}(0,1),\ i \in [1,32]$
- 300 samples, $2^{32} \approx 10^9$ input combinations
- Population size 10 of which 2 was elite
- Random initialization with equal chances for each bit
- KNN cost function
- Roulette wheel selection proportional to costs
- Reproduction functions with equal chance
  - Crossover with equal chance
  - Mutation with 0.1 chance for flip for each bit
- Termination after 100 generations

Best solution and average population development



Best individuals for the experiments



Summary

- Genetic algorithms are very heuristic
- They can find good solutions to hard problems
- Can do variable selection