

České vysoké učení technické v Praze

Fakulta elektrotechnická

**Katedra kybernetiky
Katedra počítačů**

IDA

**Intelligent Data Analysis
RESEARCH GROUP**



**COMPUTATIONAL
INTELLIGENCE
GROUP**

Vytěžování dat – přednáška 12

Nové trendy ve vytěžování znalostí

Osnova přednášky

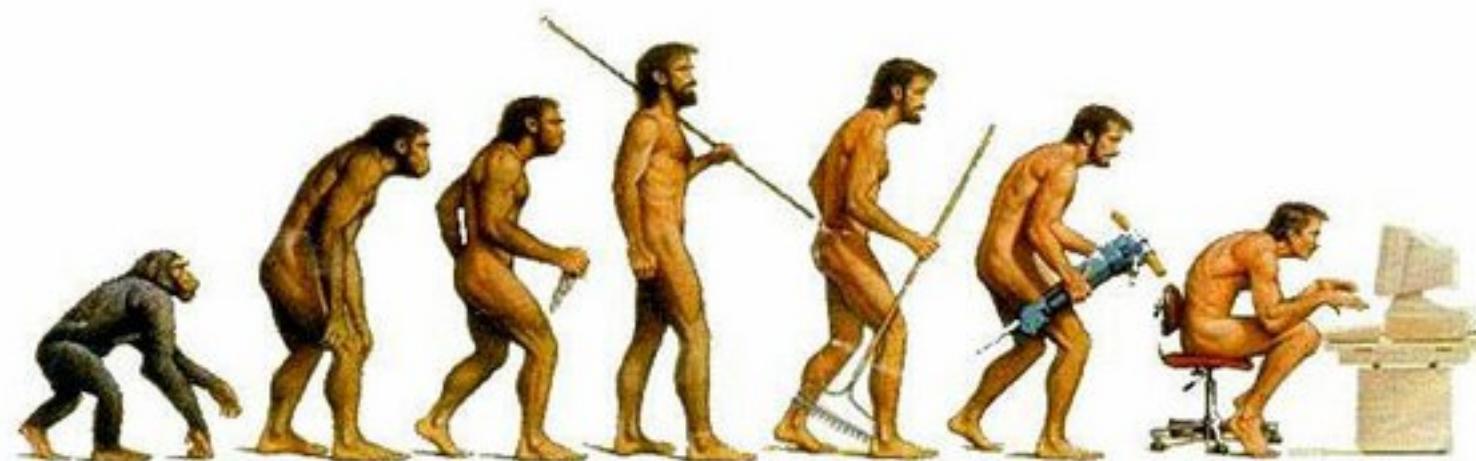
- Evoluční techniky
 - Úvod
 - Učení modelů pomocí evolučních technik
- Automatické těžení znalostí z dat
 - Trendy
 - Automatizace předzpracování dat
 - Automatizace vytěžování dat
 - Automatizace extrakce znalostí

Negradientní metody učení neuronových sítí

Zdeněk Buk, bukz1@fel.cvut.cz

2006

01001110 01100101
01110101 01110010
01101111 01101110
01101111 01110110
01100001 00100000
01110011 01101011
01110101 01110000
01101001 01101110
01100001 00100000
01101011 01100001
01110100 01100101
01100100 01110010
01111001 00100000
01110000 01101111
01100011 01101001
01110100 01100001
01100011 01110101
00101100 00100000
01000110 01000101
01001100 00100000
01000011 01010110
01010101 01010100
00101100 00100000
01010000 01110010
01100001 01101000
01100001 00000000





Genetický algoritmus – princip

- je založen na *reprodukci populace*,
- za účelem získání „kvalitnějších“ jedinců.

- Inspirace přírodou

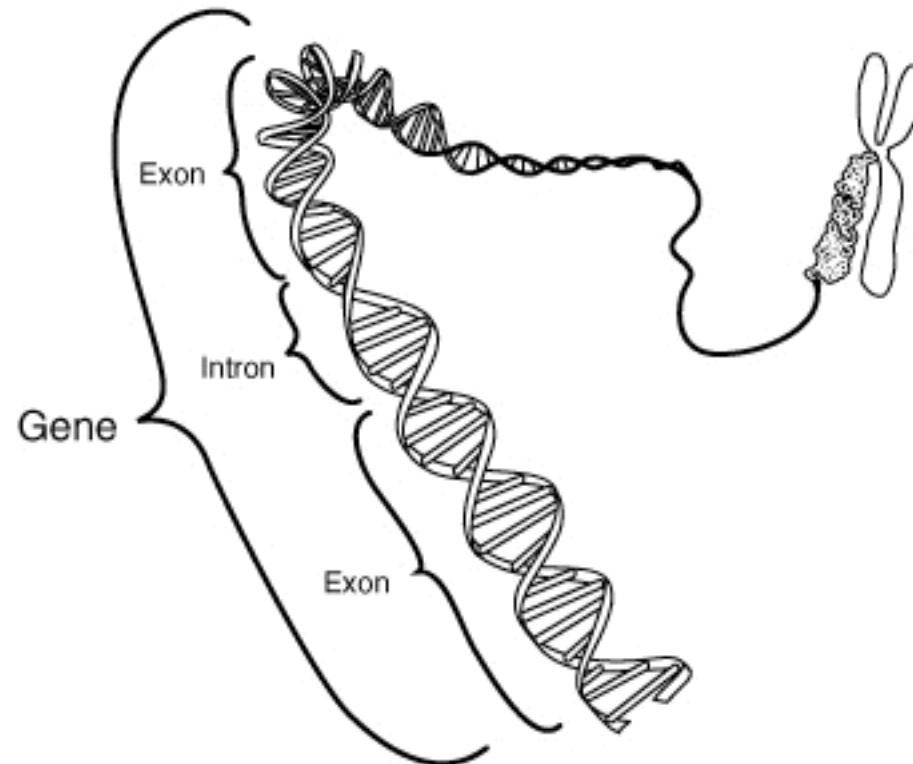
„*silnější přežije*“



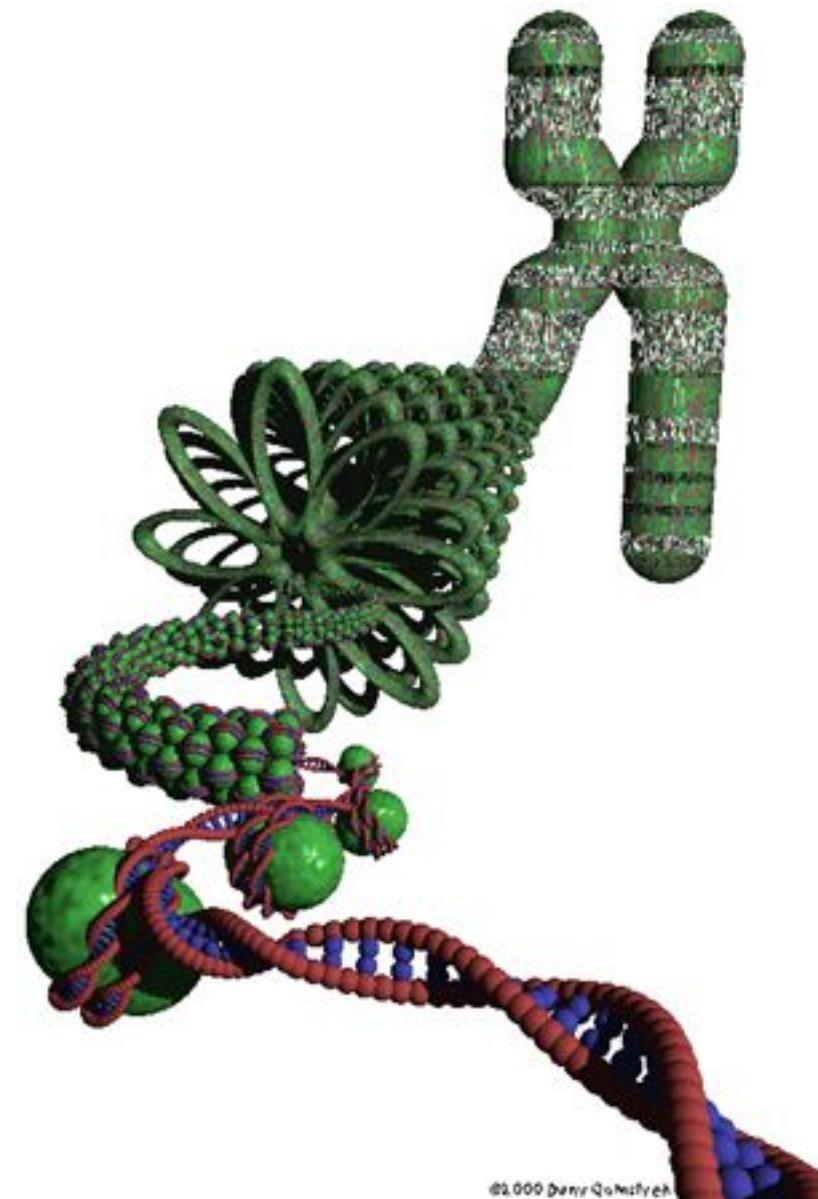
GA – Stručná historie

- 1954 – N. A. Barricelli, první simulace evoluce automatů pro jednoduchou karetní hru,
- 1970 – J. Holland, popis genetického algoritmu,
- 1992 – J. Koza, použití genetického algoritmu pro vývoj programů – *genetické programování.*

01001110 01100101
01110101 01110010
01101111 01101110
01101111 01110110
01100001 00100000
01110011 01101011
01110101 01110000
01101001 01101110
01100001 00100000
01101011 01100001
01110100 01100101
01100100 01110010
01111001 00100000
01100000 01101111
01100011 01101001
01110100 01100001
01100011 01110101
00101100 00100000
01000110 01000101
01001100 00100000
01000010 00100000
01010101 01010100
00101100 00100000
01010000 01110010
01100001 01101000
01100001 000000000



- Genom, chromosom, gen...





Genetický algoritmus – pojmy

- Kódování jedince → *chromosom*,
- množina chromosomů → *populace*.
- *Reprodukcií* populace vzniká nová *generace chromosomů*.
- Každý chromosom populace má přiřazenu *fitness hodnotu* – danou účelovou (fitness) funkcí, vyjadřující „kvalitu“ daného jedince.

Genetický algoritmus – pojmy

01001110 01100101

01110101 01110110

01101111 01101110

01100001 00100000

01110011 01101011

01101010 01110000

01101001 01101110

01100001 00100000

01101011 01100001

01110100 01100101

01100100 01110010

01111001 00100000

01110000 01101111

01100011 01101000

01110100 01100000

01100011 01110101

00101100 00100000

01000110 01000101

01001100 00100000

01000011 01010110

01010101 01010100

00101100 00100000

01010000 01110010

01100001 01101000

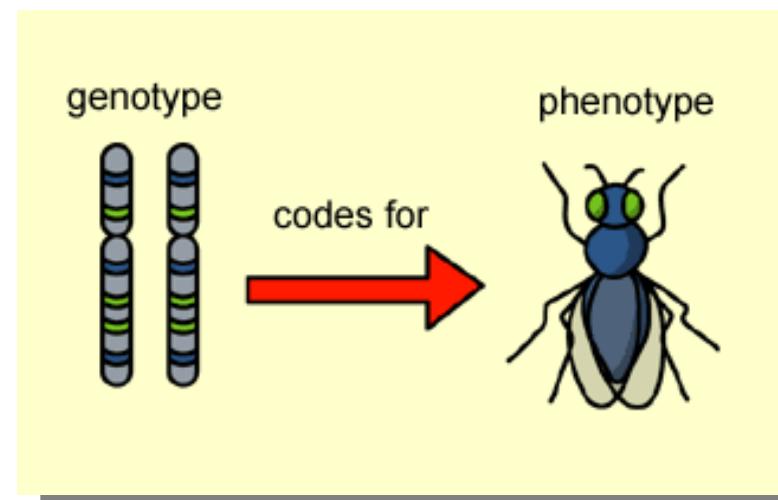
01100001 0000000000

- Genotyp

– soubor genetické informace jedince

- Fenotyp

– zobrazení genetické informace do konkrétního jedince



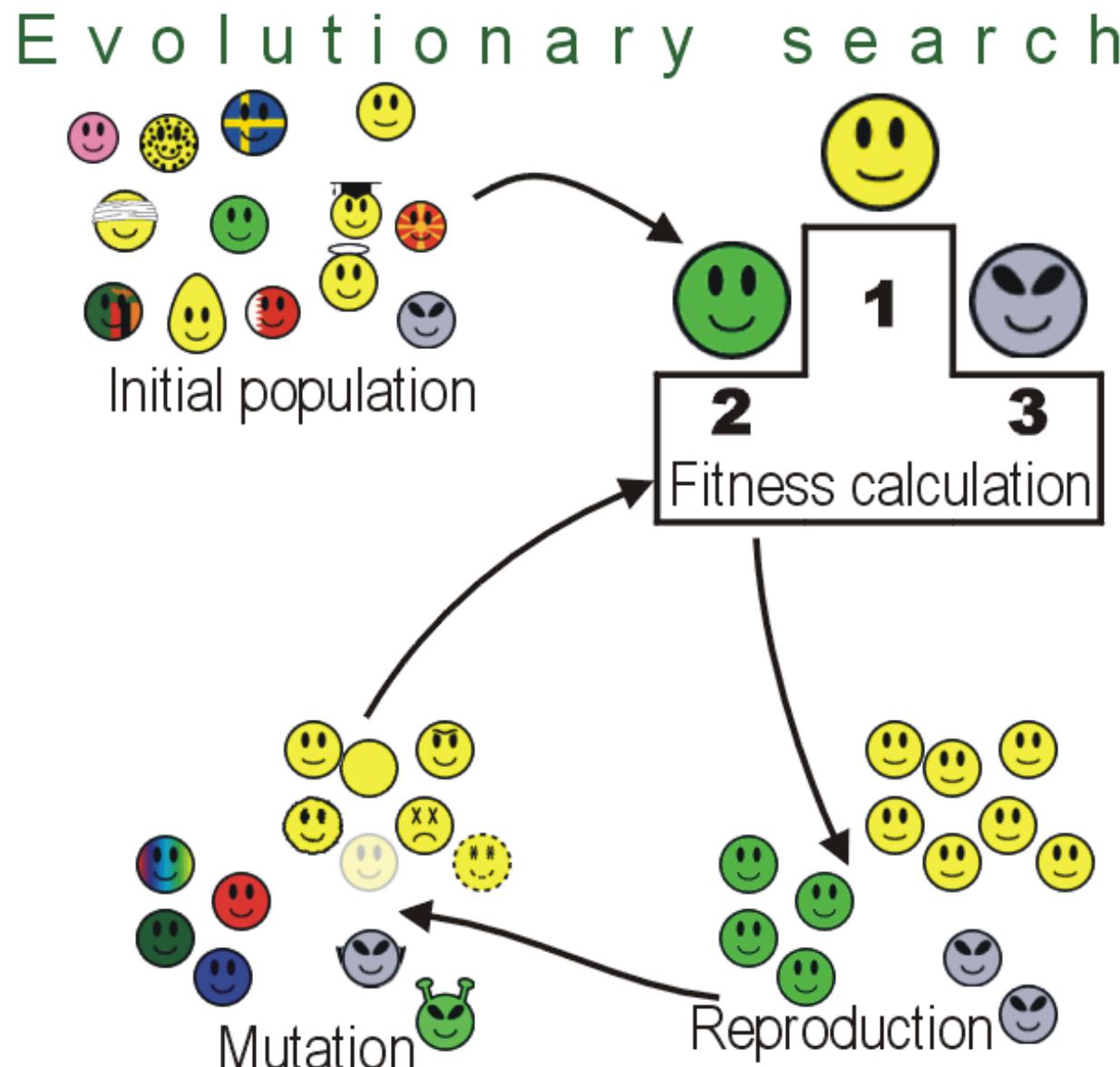


Genetický algoritmus

- Během *reprodukce* jsou jednotlivé chromosomy vybírány podle fitness hodnoty a podrobeny dalším operacím
 - *křížení* – rekombinace částí vybraných párů chromosomů,
 - *mutace* – náhodná modifikace hodnoty genů chromosomu.

Genetický algoritmus

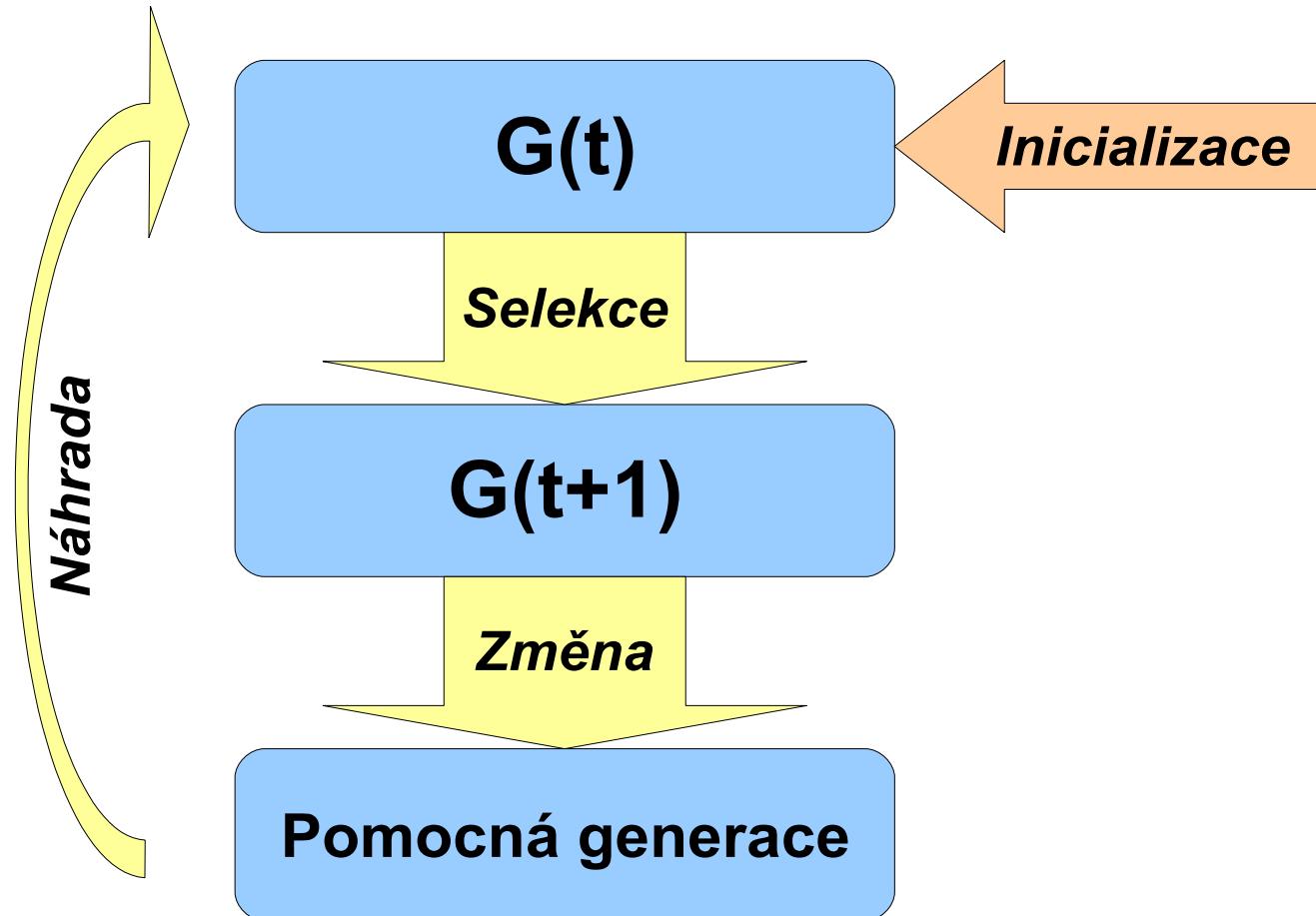
```
01001110 01100101  
01110101 01110010  
01101111 01101110  
01101111 01110110  
01100001 00100000  
01110011 01101011  
01110101 01110000  
01101001 01101110  
01100001 00100000  
01101011 01100001  
01110100 01100101  
01100100 01110010  
01111001 00100000  
01100000 01101111  
01100011 01101001  
01110100 01100001  
01100011 01110101  
00101100 00100000  
01000110 01000101  
01001100 00100000  
01000011 01010110  
01010101 01010100  
00101100 00100000  
01010000 01110010  
01100001 01101000  
01100001 000000000
```





Obecný genetický algoritmus

01001110 01100101
01110101 01110010
01101111 01101110
01101111 01110110
01100001 00100000
01110011 01101011
0110101 01110000
01101001 01101110
01100001 00100000
01101011 01100001
01110100 01100101
01100100 01110010
01111001 00100000
01100000 01101111
01100011 01101001
01110100 01100001
01100011 01110101
00101100 00100000
01000110 01000101
01001100 00100000
01000011 01010110
01010101 01010100
00101100 00100000
01010000 01110010
01100001 01101000
01100001 00000000





Obecný genetický algoritmus

- $G(t) = \{x_{t,1}, x_{t,2}, \dots, x_{t,N}\}$ – populace jedinců

$t=0;$

inicializace $G(t);$

vyhodnocení $G(t);$

while (not zastavovací_kritérium) **do**

$t=t+1;$

selekce z $G(t-1)$ **do** $G(t);$

modifikace $G(t);$

vyhodnocení $G(t);$

done



Tvorba genetického algoritmu

1. Způsob kódování chromosomů,

2. definice účelové (fitness) funkce,

3. metoda generování počáteční populace,

4. množina operátorů,

5. množina pracovních parametrů.

GA – kódování chromosomů

01001110 01100101

01110101 01101010

01101111 01101110

01101111 01110110

01100001 00100000

01110011 0110101

01110101 01110000

01101001 01101110

01100001 00100000

01101011 01100001

01110100 01100101

01100100 01110010

01111001 00100000

01110000 01101111

01100011 01101001

01110100 01100001

01100011 01110101

00101100 00100000

01000110 01000101

01001100 00100000

01000011 01010101

01010101 01010100

00101100 00100000

01010001 01000000

01100001 01101000

01100001 00000000

- Pomocí binárního kódu,

– příklad:

0	1	0	1	1	0	1	1

1	1	0	0	1	0	0	1

- řetězcem čísel – číselná reprezentace,

- stromem objektů – *genetické programování*.



GA – definice účelové funkce

- Účelová funkce vyjadřuje kvalitu daného jedince a tím i jeho šanci propracovat se do následující generace.
- Při učení neuronových sítí může být jako účelová funkce zvolena *chyba sítě* na trénovacích datech (potom hledáme minimum).



GA – počáteční populace

- Počáteční populace se většinou generuje náhodně,
 - ve stavovém prostoru se náhodně rozmístí jedinci.
- GA lze také použít pro vylepšení některého již známého řešení, potom se počáteční populace vytvoří na základě tohoto existujícího řešení resp. s ohledem na něj.



GA – operátor křížení

- Tzv. jednobodové křížení,

– dva jedinci (*rodiče*) generují dva nové jedince (*potomky*) následující generace (nová populace).

Rodič 1 (91)

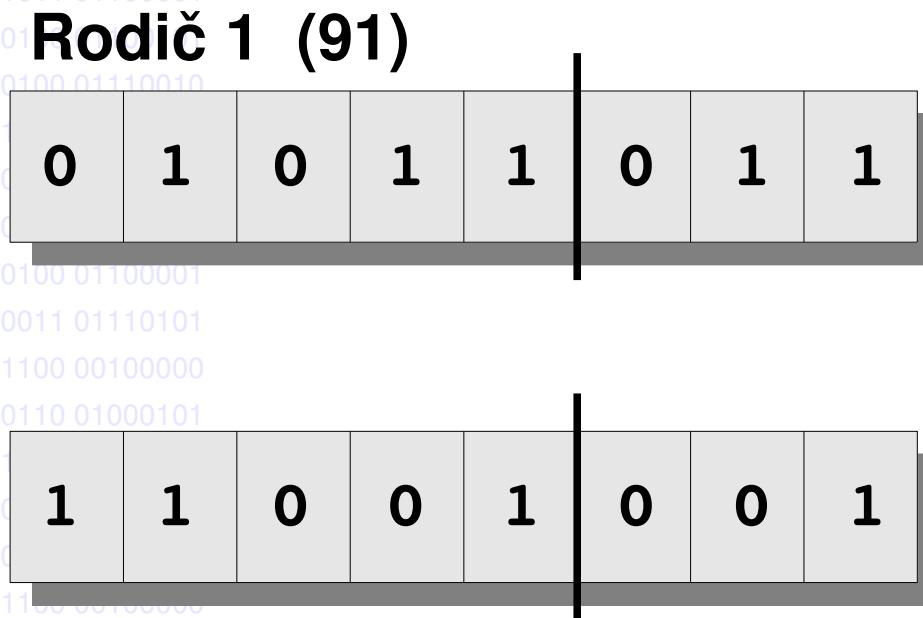
0	1	0	1	1	0	1	1
---	---	---	---	---	---	---	---

1	1	0	0	1	0	0	1
---	---	---	---	---	---	---	---

Rodič 2 (201)

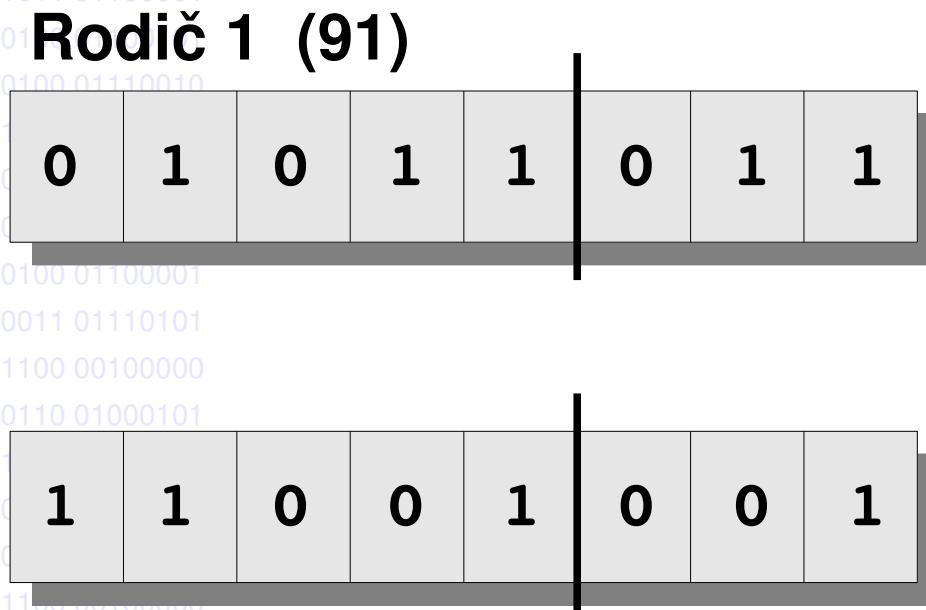
- Tzv. jednobodové křížení,

– dva jedinci (*rodiče*) generují dva nové jedince (*potomky*) následující generace (nová populace).

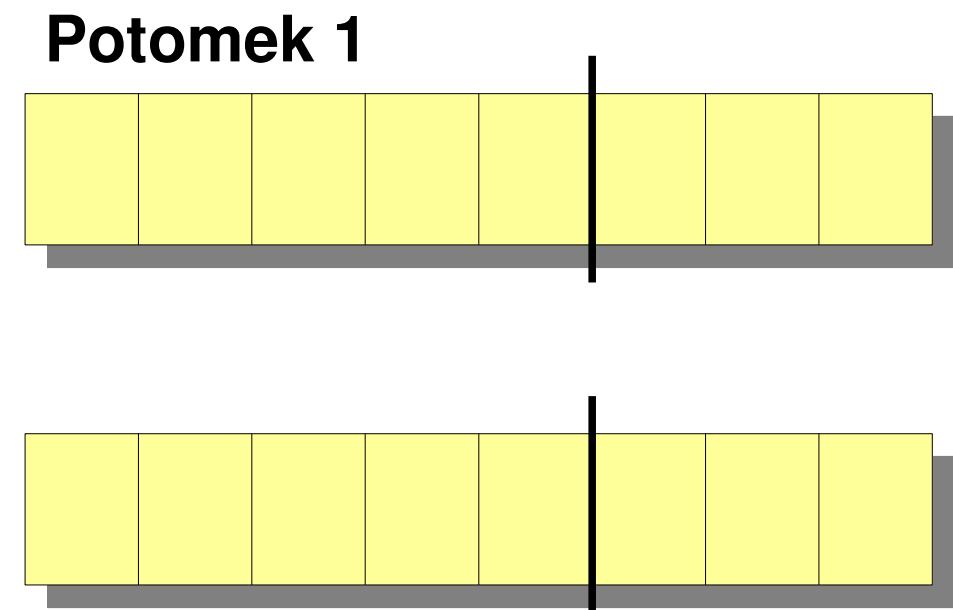


Rodič 2 (201)

- Tzv. jednobodové křížení,
 - dva jedinci (*rodiče*) generují dva nové jedince (*potomky*) následující generace (nová populace).



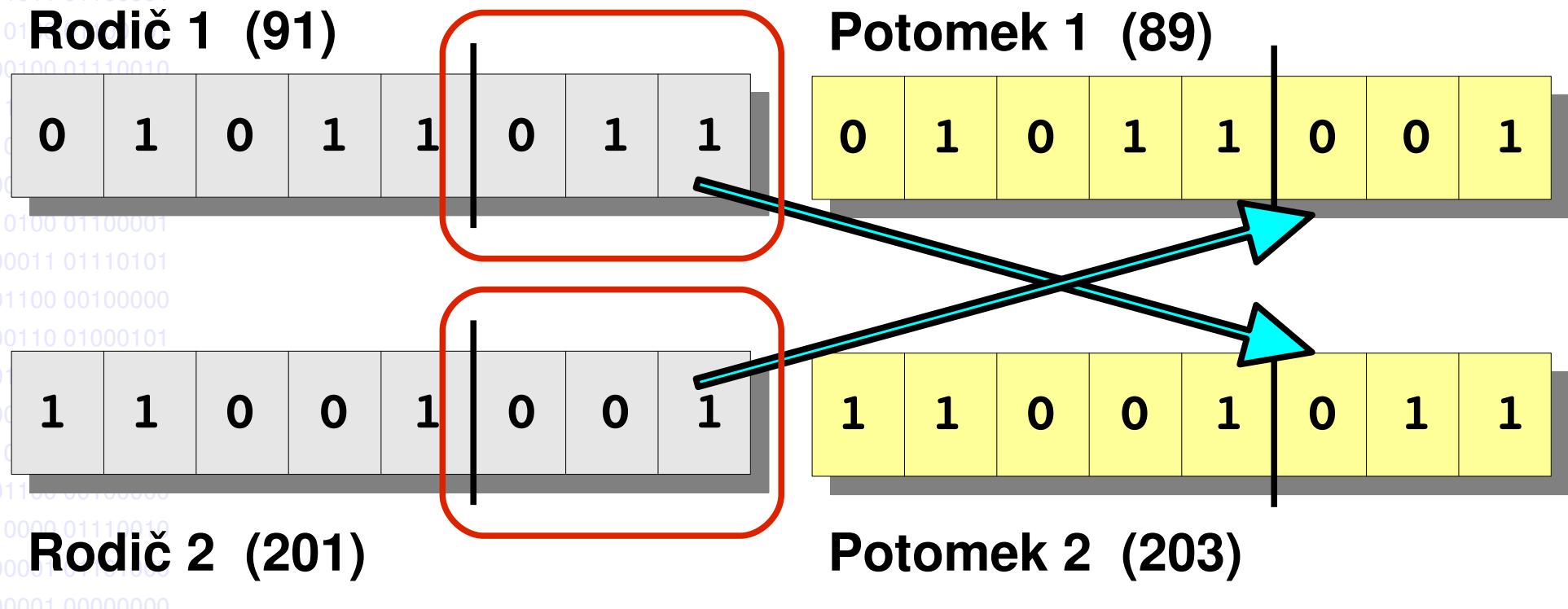
Rodič 2 (201)



Potomek 2

- Tzv. jednobodové křížení,

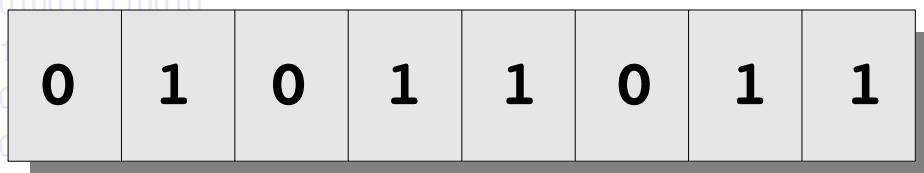
– dva jedinci (*rodiče*) generují dva nové jedince (*potomky*) následující generace (nová populace).



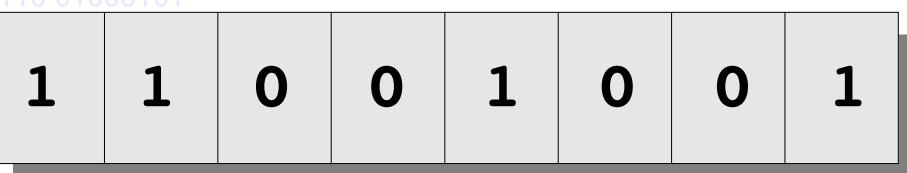
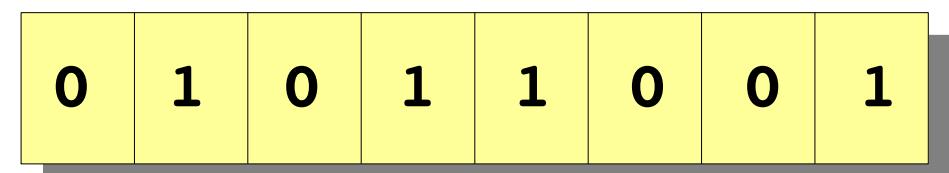
- Tzv. jednobodové křížení,

– dva jedinci (*rodiče*) generují dva nové jedince (*potomky*) následující generace (nová populace).

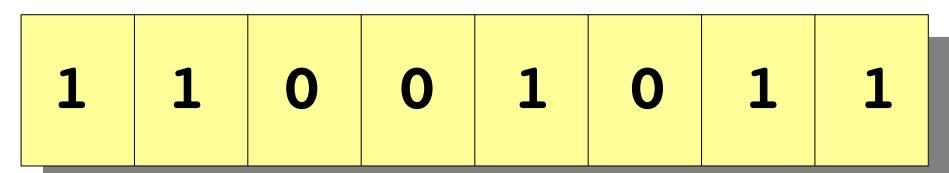
Rodič 1 (91)



Potomek 1 (89)



Rodič 2 (201)



Potomek 2 (203)

- Každý bit chromosomu se s pravděpodobností p při přechodu do další generace invertuje,
– pravděpodobnost p je v praxi velmi malá, jinak algoritmus přechází v náhodné prohledávání.

Původní jedinec (91)

0	1	0	1	1	0	1	1
---	---	---	---	---	---	---	---



1	1	0	0	1	0	0	0
---	---	---	---	---	---	---	---

Zmutovaný jedinec (90)



GA – pracovní parametry

- Velikost populace,
- pravděpodobnost mutace,
- způsob výběru jedinců z předchozí generace,
- definice *ukončovacího kritéria*
 - např. chyba sítě na validačních datech.



GA – Příklad

- Příklad aplikace genetického algoritmu na hledání minima 1D funkce.

Následující obrázky jsou výstupem appletu, jehož autorem je Marek Obitko

<http://www.cs.unibo.it/babaoglu/courses/cas/tutorials/ga/index.html>

Příklad – počáteční populace

01001110 01100101

01110101 01101010

01101111 01101110

01101111 01110110

01100001 00100000

01110011 01101011

01110101 01110000

01101001 01101110

01100001 00100000

01101011 01100001

01110100 01100101

01100100 01110010

01111001 00100000

01110000 01101111

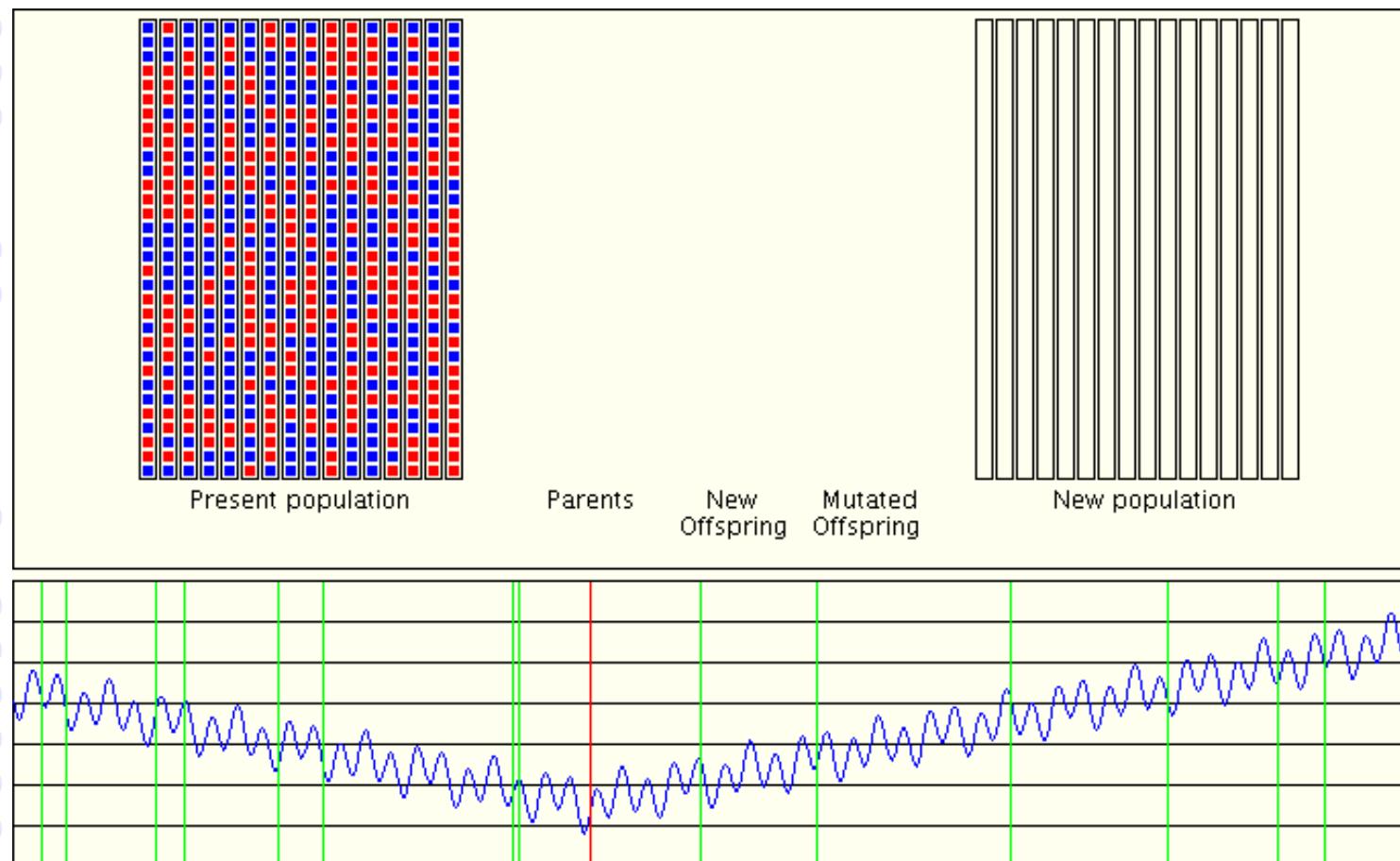
01100011 01101001

01110100 01100001

01100011 01110101

00101100 00100000

01000110 01000101



Příklad – elitářství

01001110 01100101

01110101 01101012

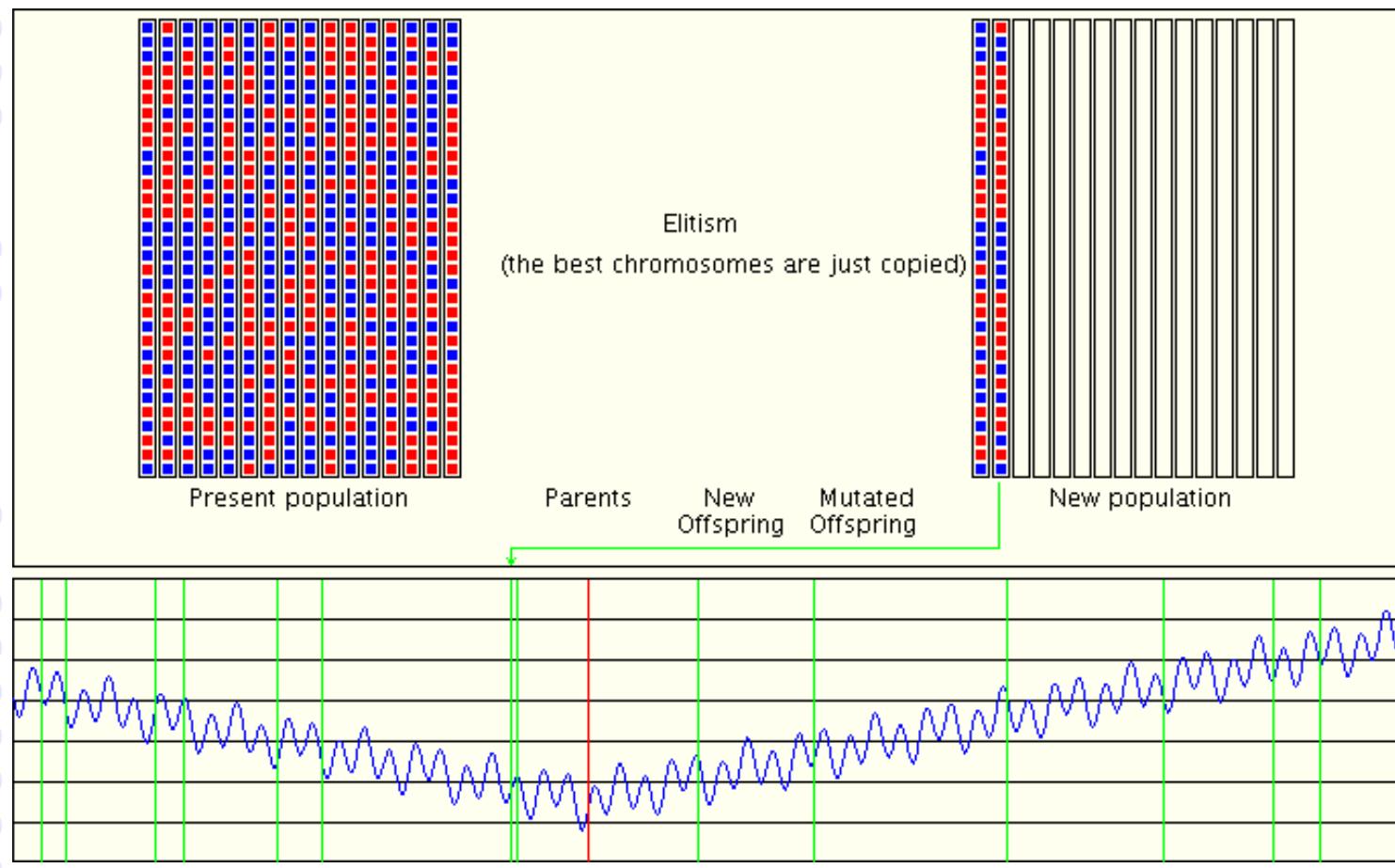
01101111 01101110

01101111 01110110

01100001 0010100

01110011 01101011

- Nejlepší jedinci přímo přecházejí do nové generace.



Příklad – křížení

01001110 01100101

01110101 01100101

01101111 01101110

01101111 01110110

01100001 00100000

01110011 01101011

01110101 01110000

01101001 01101110

01100001 00100000

01101011 01100001

01110100 01100101

01100100 01110010

01111001 00100000

01110000 01101111

01100011 01101001

01110100 01100001

01100011 01110101

00101100 00100000

01000110 01000101

01001100 00100000

01000011 01010110

01010101 01010100

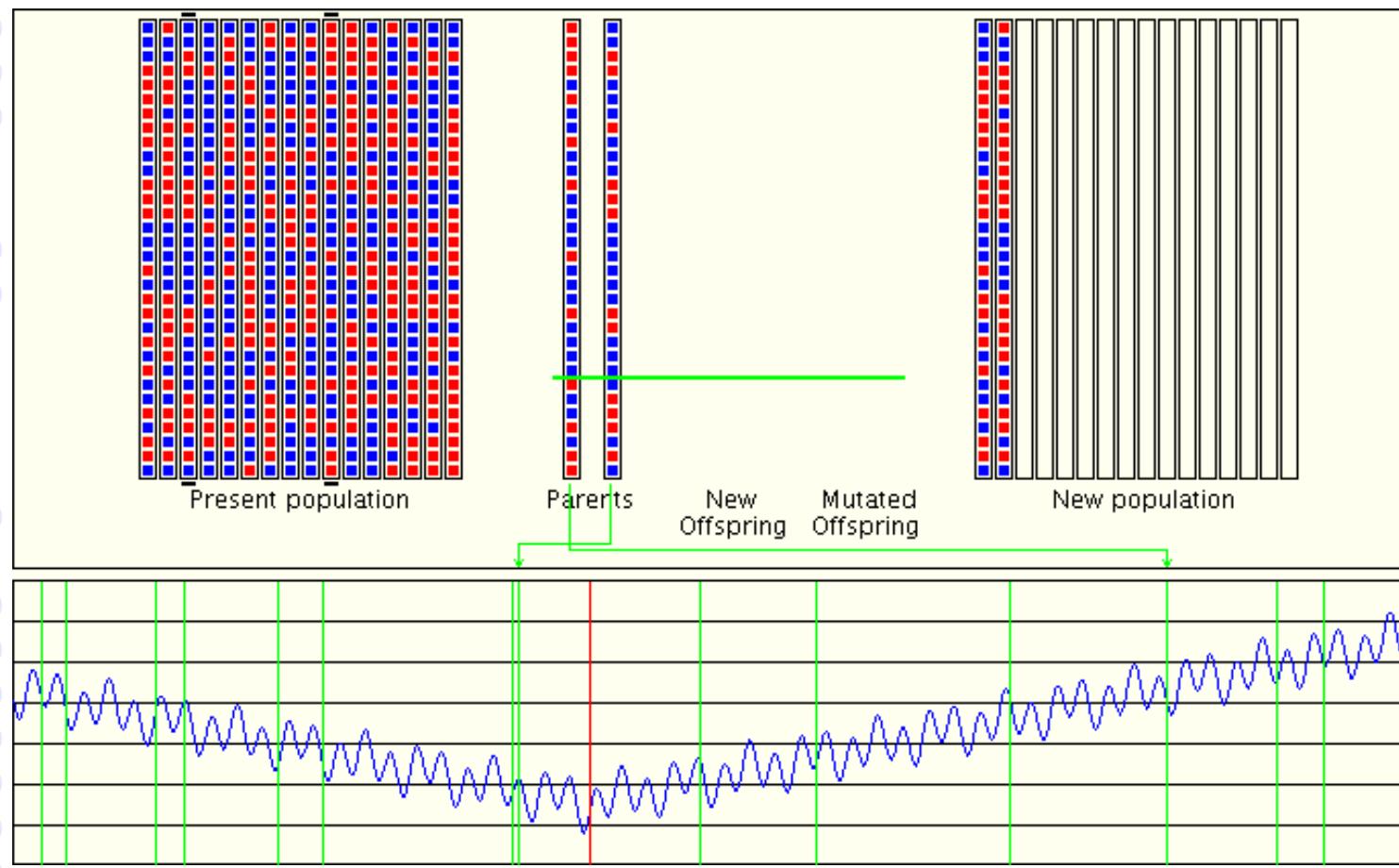
00101100 00100000

01010000 01110010

01100001 01101000

01100001 00000000

- Jsou vybráni dva jedinci pro křížení.



Příklad – křížení

01001110 01100101

01110101 01100101

01101111 01101110

01101111 01110110

01100001 00100000

01110011 01101011

01110101 01110000

01101001 01101110

01100001 00100000

01101011 01100001

01110100 01100101

01100100 01110010

01111001 00100000

01110000 01101111

01100011 01101001

01110100 01100001

01100011 01110101

00101100 00100000

01000110 01000101

01001100 00100000

01000011 01010110

01010101 01010100

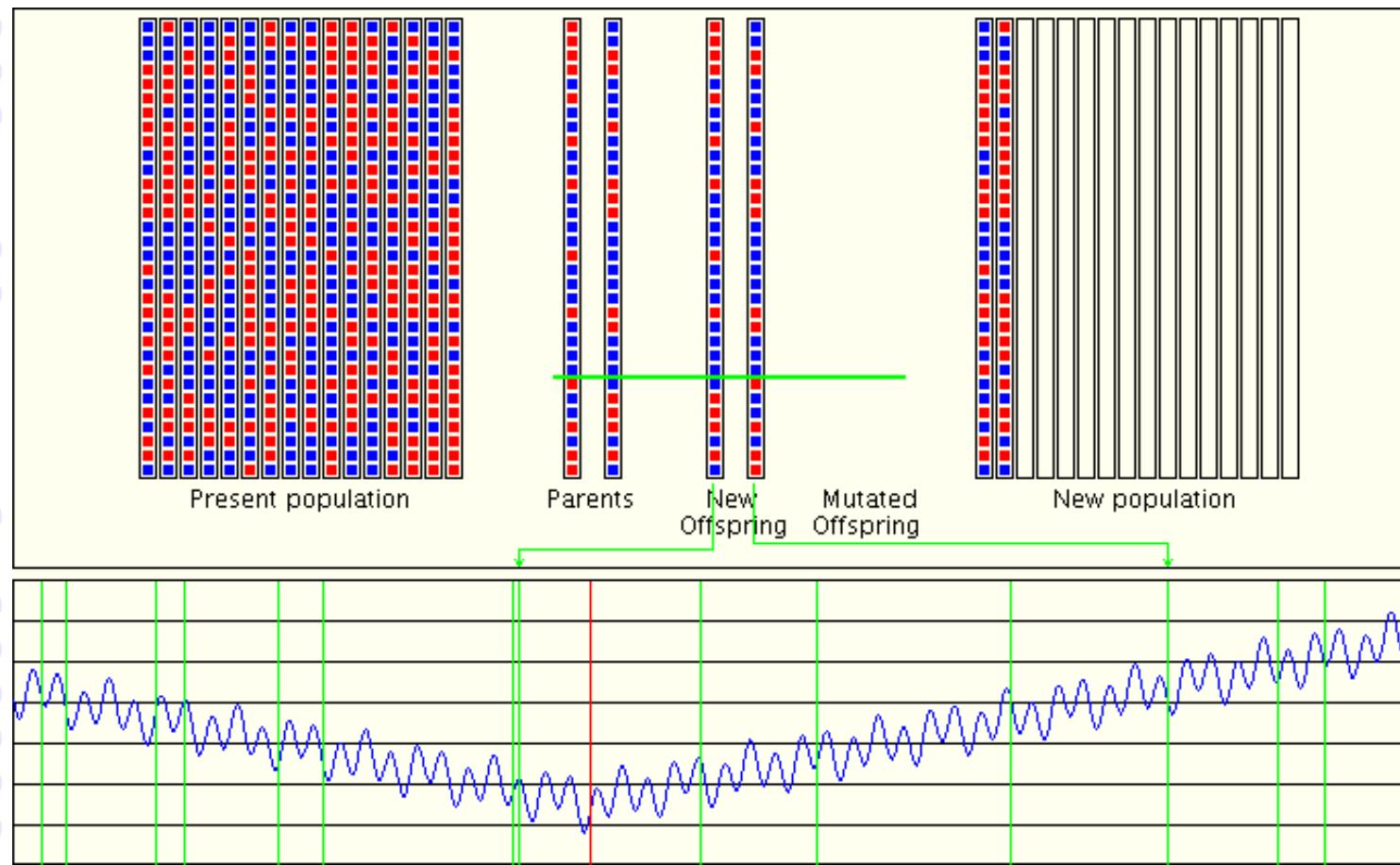
00101100 00100000

01010000 01110010

01100001 01101000

01100001 00000000

- Je provedeno jednobodové křížení.



01001110 01100101

01110101 01101010

01101111 01101110

01101111 01110110

01100001 00100000

01110011 01101011

01110101 01110000

01101001 01101110

01100001 00100000

01101011 01100001

01110100 01100101

01100100 01110010

01111001 00100000

01110000 01101111

01100011 01101001

01110100 01100001

01100011 01110101

00101100 00100000

01000110 01000101

01001100 00100000

01000011 01010110

01010101 01010100

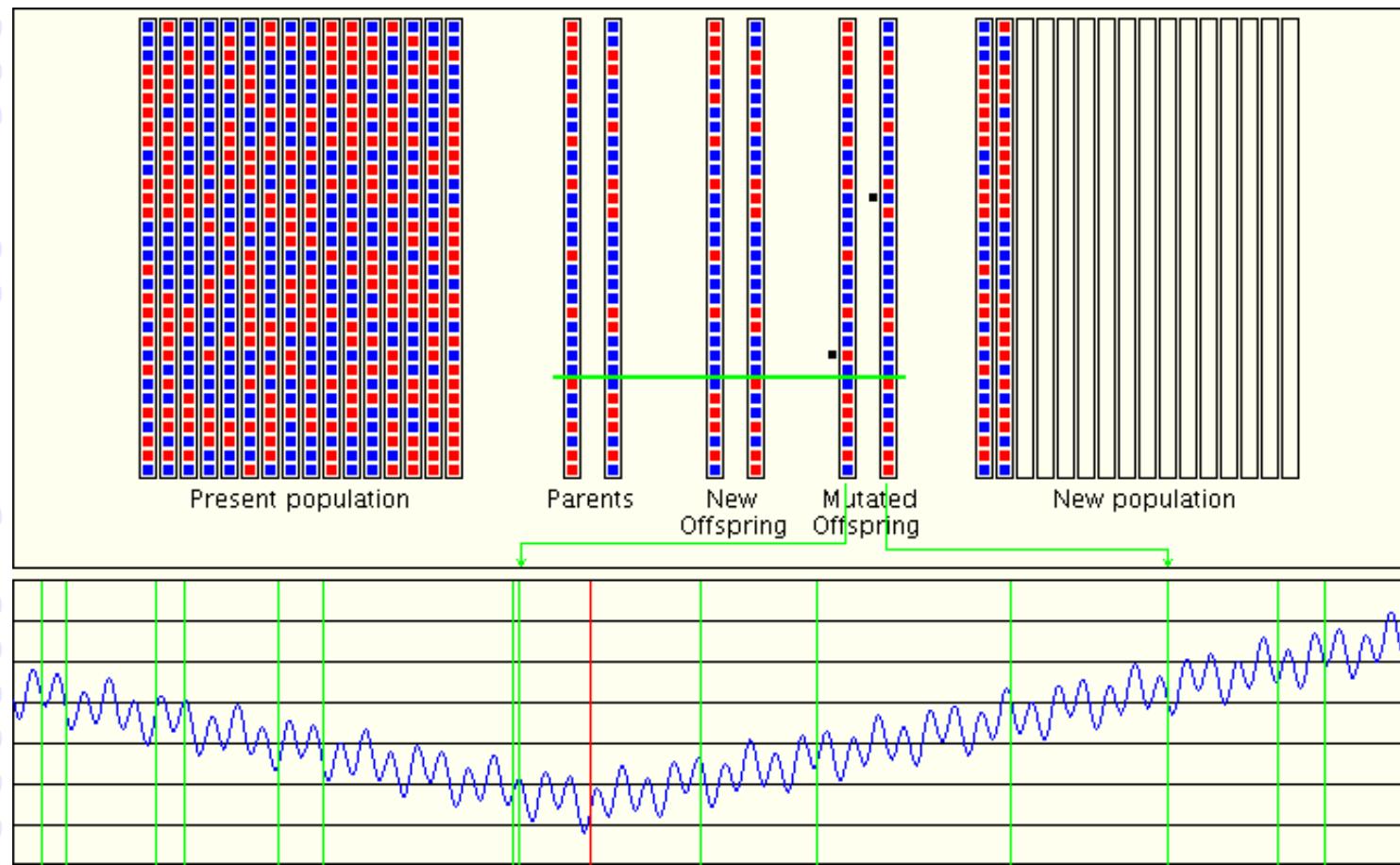
00101100 00100000

01010000 01110010

01100001 01101000

01100001 00000000

- Na potomky je aplikován operátor mutace.



Příklad – nová populace

01001110 01100101

01110101 01101010

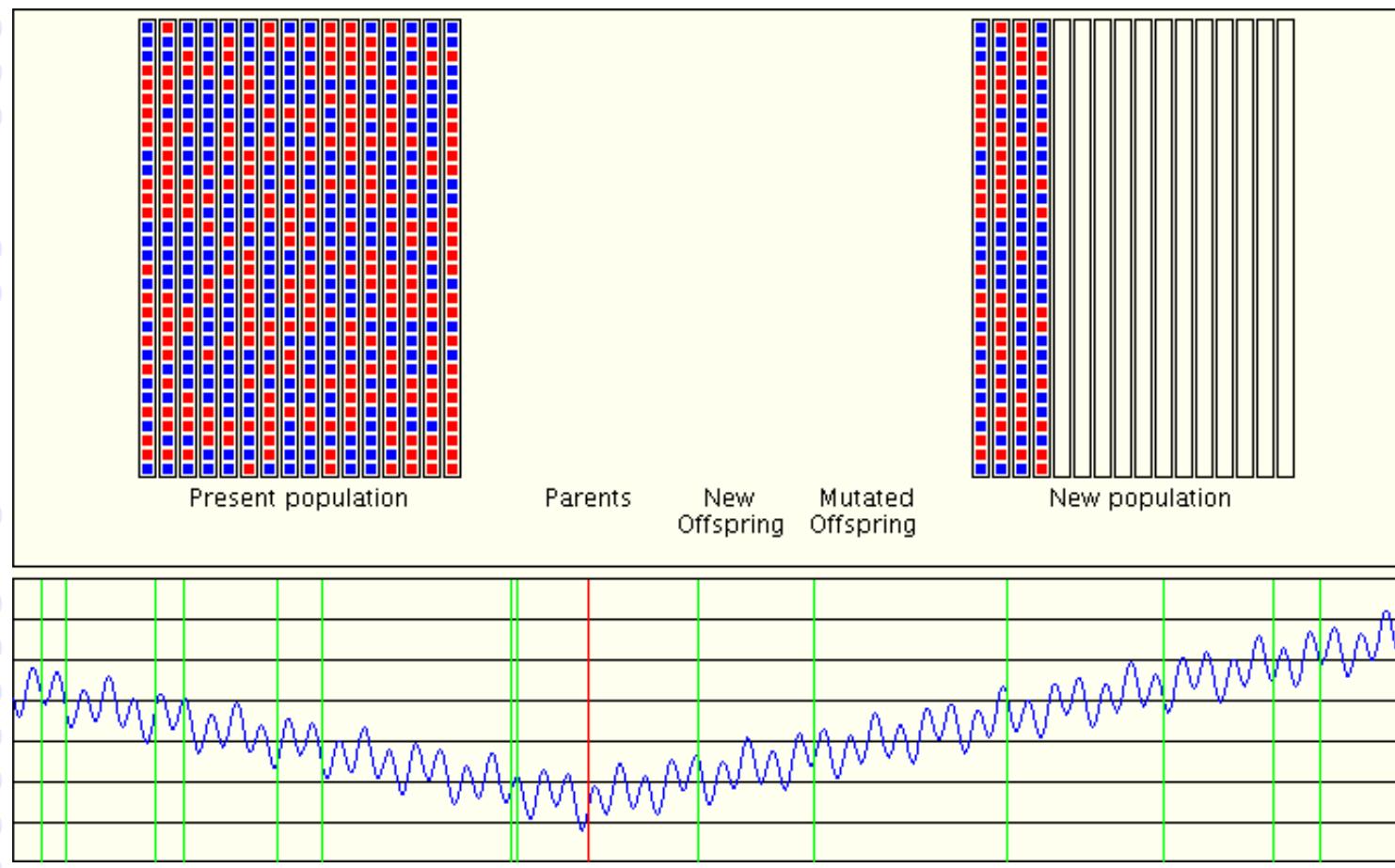
01101111 01101110

01101111 01110110

01100001 00100000

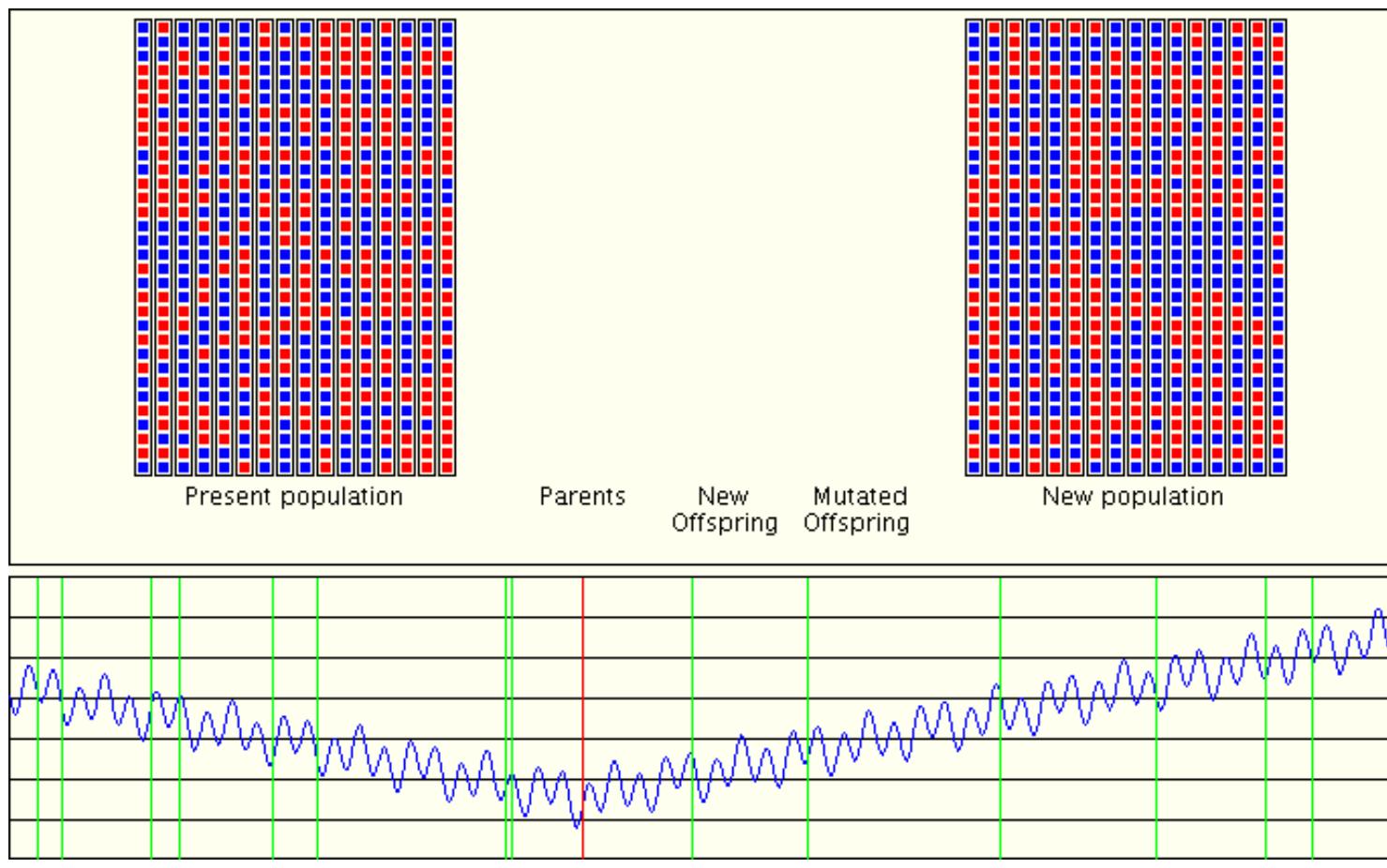
01110011 01101011

- Noví jedinci (*potomci*) jsou zařazeni do následující generace (*nové populace*).



Příklad – nová populace

- Opakováním celého procesu je postupně naplněna celá nová populace.



Příklad – nahrazení populace

01001110 01100101

01110101 01101000

01101111 01101110

01101111 01110110

01100001 00100000

01110011 01101011

01110101 01110000

01101001 01101110

01100001 00100000

01101011 01100001

01110100 01100101

01100100 01110010

01111001 00100000

01110000 01101111

01100011 01101001

01110100 01100001

01100011 01110101

00101100 00100000

01000110 01000101

01001100 00100000

01000011 01010110

01010101 01010100

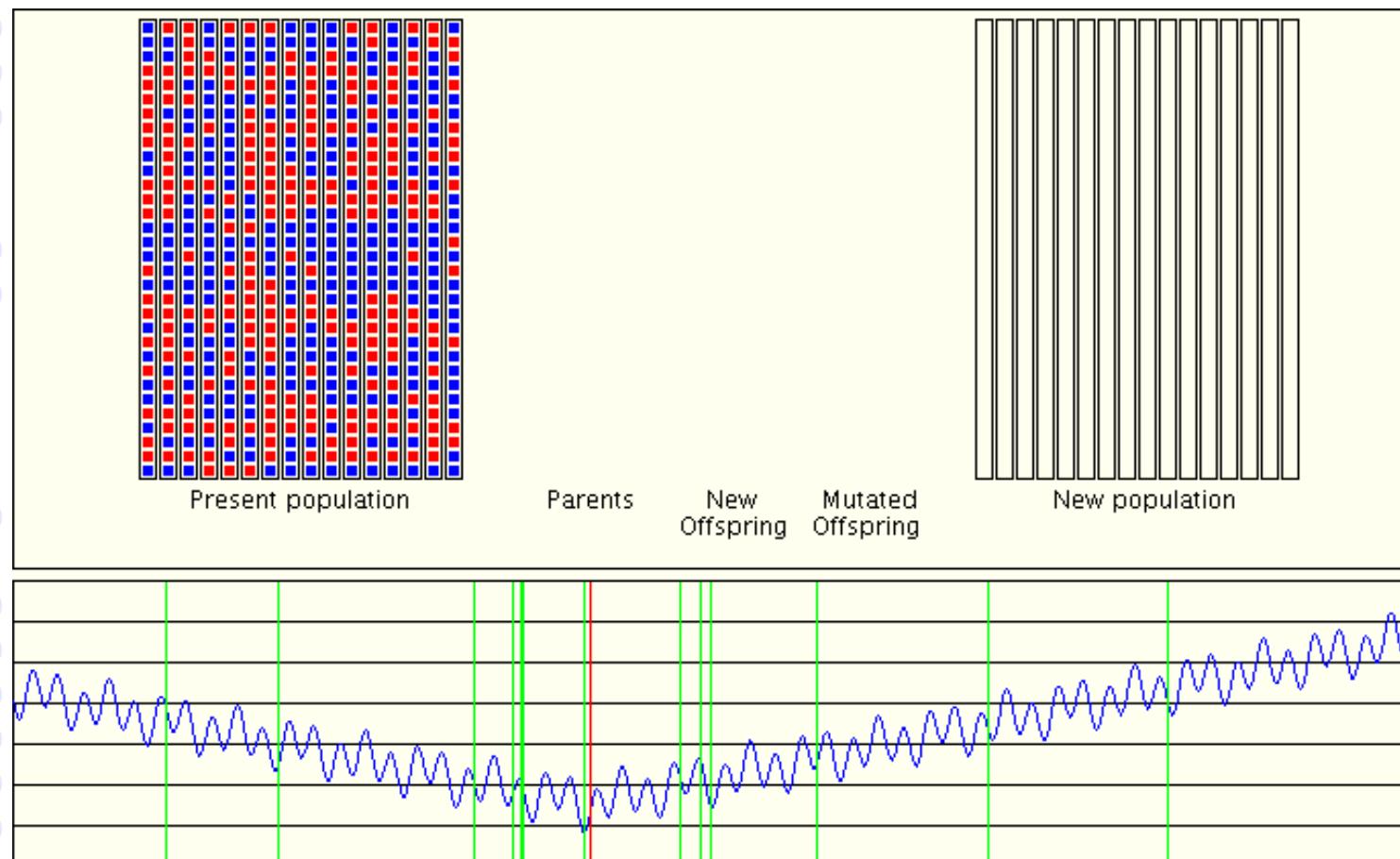
00101100 00100000

01010000 01110010

01100001 01101000

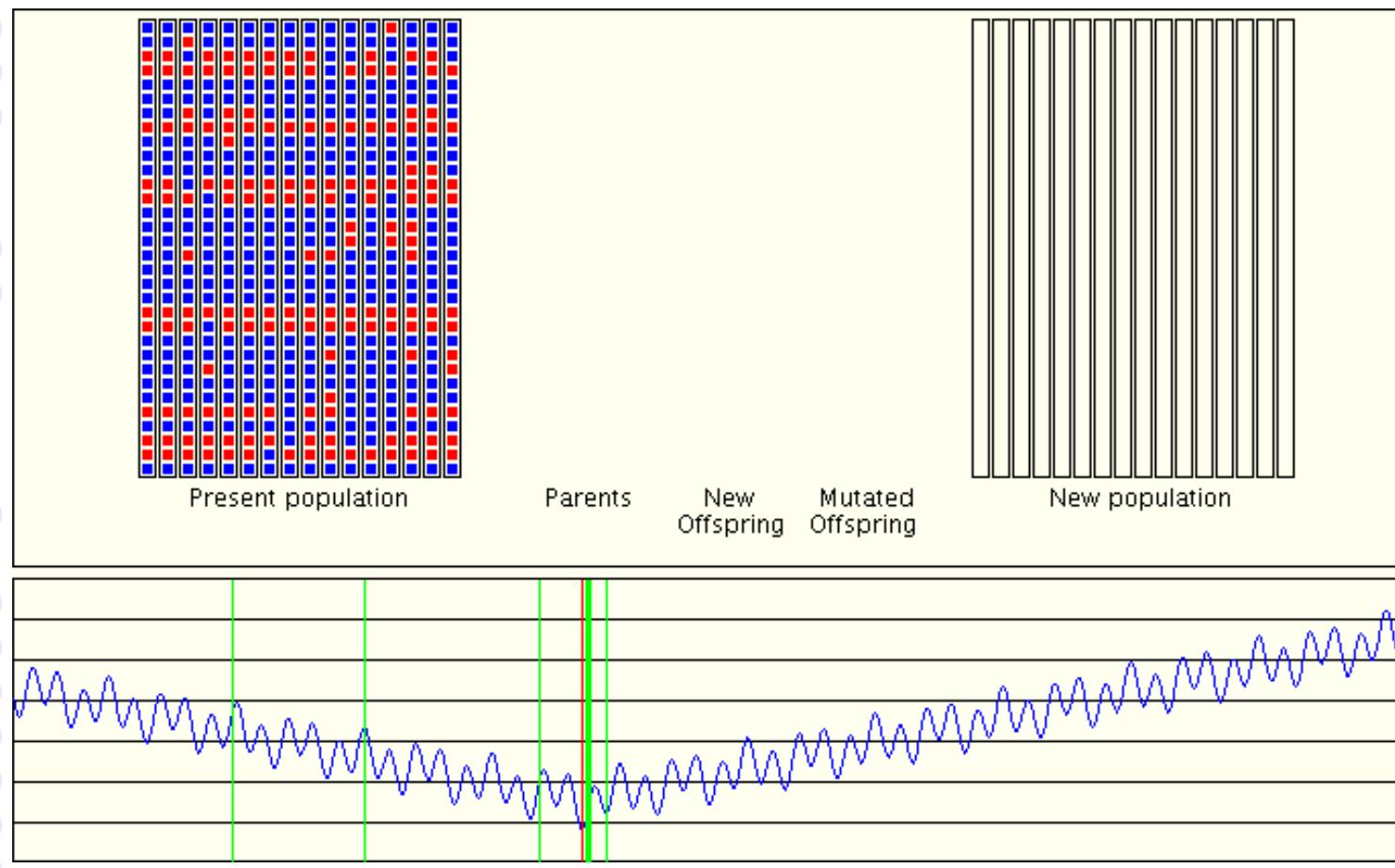
01100001 00000000

- Původní populace je nahrazena novou.



Příklad – finální populace

- Po několika generacích je na základě ukončovacího kritéria proces zastaven.





GA a neuronové sítě

- Obecně – optimalizační metoda.
- Použití genetického algoritmu v oblasti neuronových sítí:
 - pro modifikaci vah sítě,
 - konstrukci topologie sítě,
 - úpravě dalších parametrů, např. aktivačních funkcí,
 - optimalizaci parametrů jiného učicího algoritmu,
 - hledání parametrů samotného GA.

01001110 01100101

01110101 0111

01101111 01101110

01101111 01110110

01100001 00100000

01110011 01101011

01110101 01110000

01101000 01100000

01101011 01100001

01110100 01100101

01100100 01110010

01111001 00100000

01110000 01101111

01100011 01101001

01110100 01100001

01100011 01110101

00101100 00100000

01000110 01000101

01001100 00100000

01000011 01010110

01010101 01010100

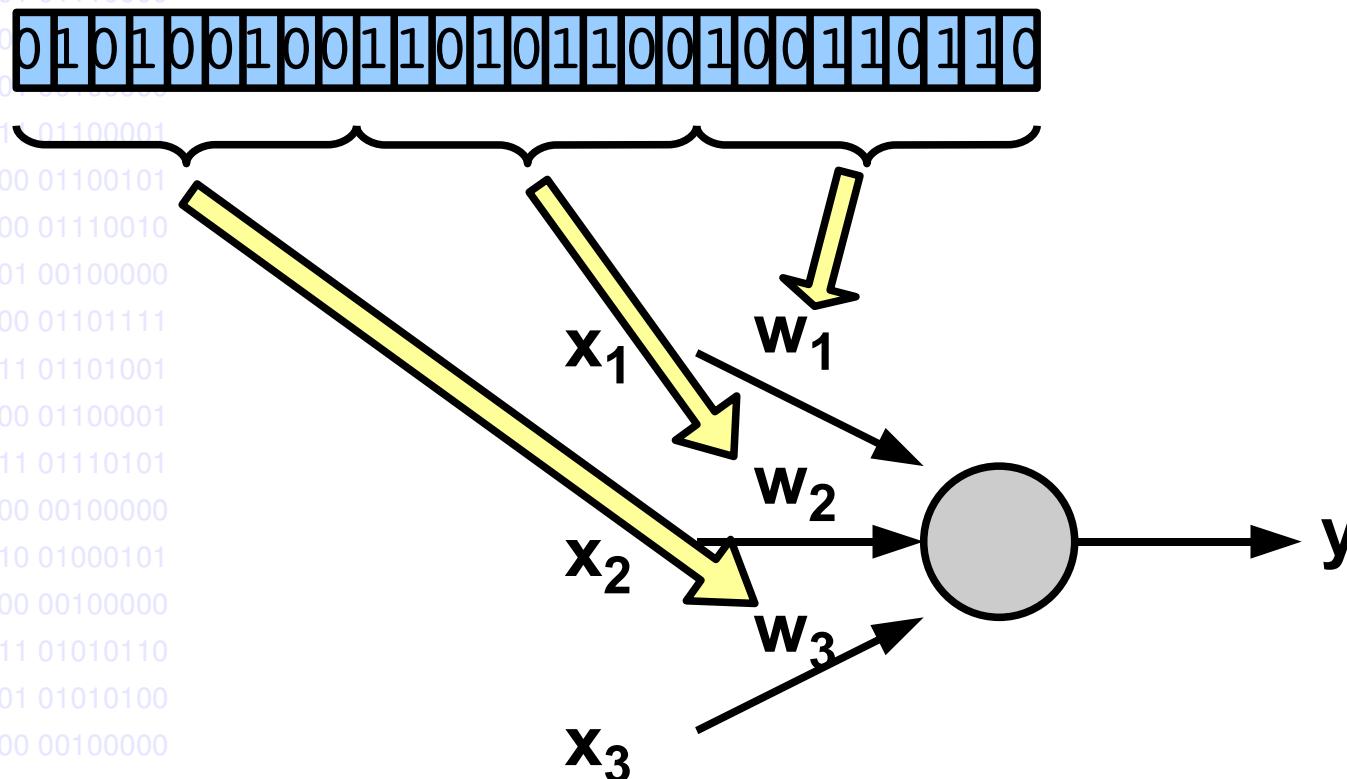
00101100 00100000

01010000 01110010

01100001 01101000

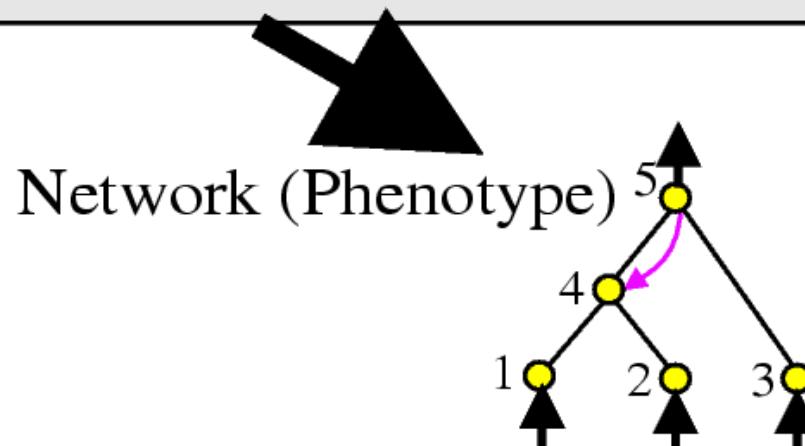
01100001 00000000

- Přímé kódování vah sítě



Příklad, modifikace topologie

Genome (Genotype)						
Node Genes	Node 1 Sensor Input	Node 2 Sensor Input	Node 3 Sensor Input	Node 4 Hidden Hidden	Node 5 Hidden Output	
Connect. Genes	In 1 Out 4 Weight 0.7 Enabled Innov 1	In 2 Out 4 Weight 0.5 Enabled Innov 3	In 2 Out 5 Weight 0.5 DISAB Innov 4	In 3 Out 5 Weight 0.2 Enabled Innov 5	In 4 Out 5 Weight 0.4 Enabled Innov 6	In 5 Out 4 Weight 0.6 Enabled Innov 10



Stanley, K., O., Miikkulainen, R.: Efficient Evolution of Neural Networks Topologies.

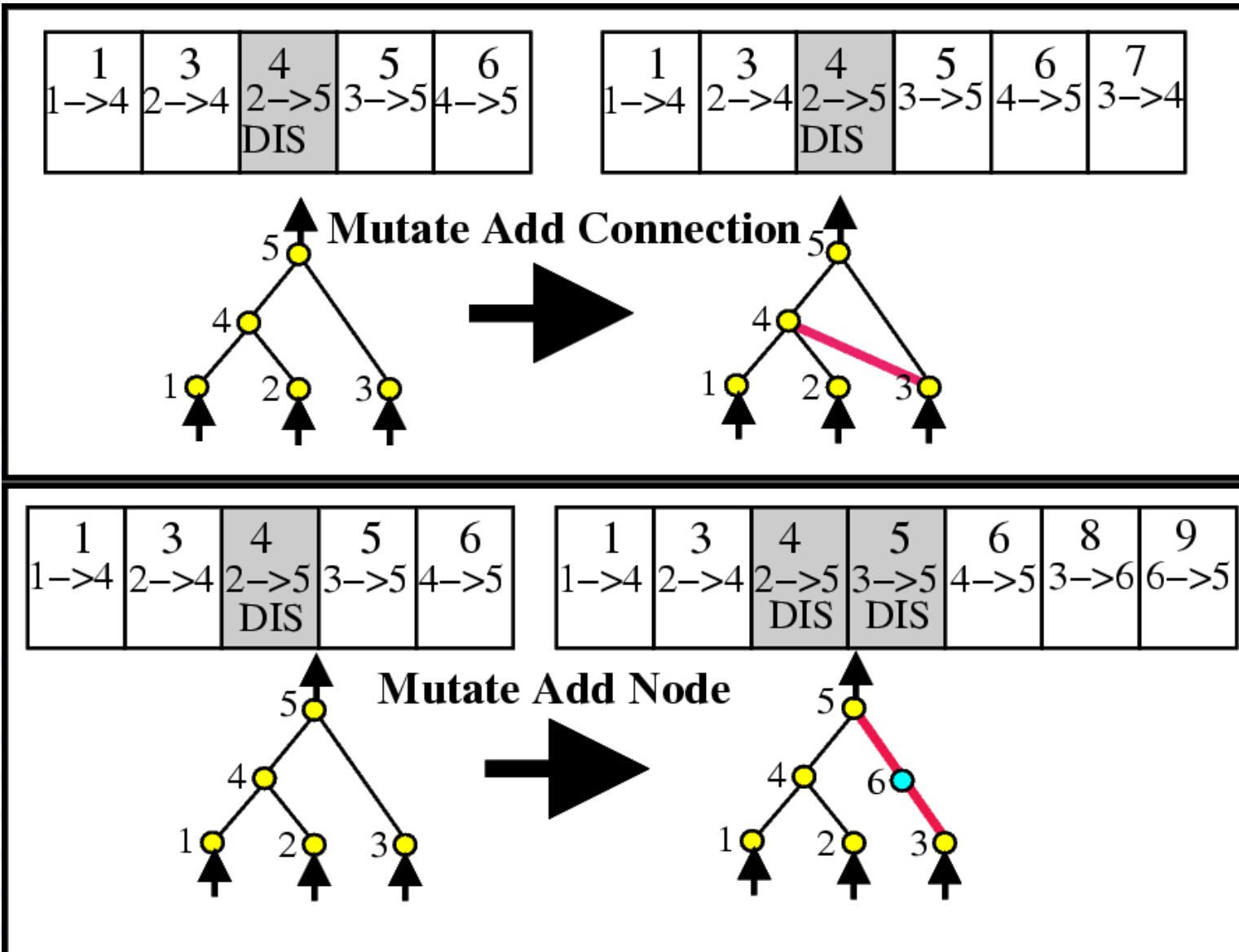
Proceedings of the 2002 Congress of Evolutionary Computation.

Modifikace topologie, mutace

```

01001110 0110010
01110101 0111001
01101111 0110111
01101111 0111011
01100001 0010000
01110011 0110101
0110101 0111000
01101001 0110111
01100001 0010000
01101011 0110000
01110100 0110010
01100100 0111001
01111001 0010000
01100000 0110111
01100011 0110100
01110100 0110000
01100011 0111010
00101100 0010000
01000110 0100010
01001100 0010000
01000011 0101011
01010101 0101010
00101100 0010000
01010000 0111001
01100001 0110100
01100001 000000000

```



Modifikace topologie, křížení 1/2

01001110 01100

01110101 01110

01101111 01100

01101111 01110

01100001 00100

01110011 01100

01101010 01110

01101001 01100

01100001 00100

01101011 01100

01110100 01100

01100100 01110

01111001 00100

01110000 01100

01100011 01100

01110100 01100

01100011 01110

00101100 00100

01000110 01000

01001100 00100

01000011 01010

01010101 01010

00101100 00100

01010000 01110

01100001 01100

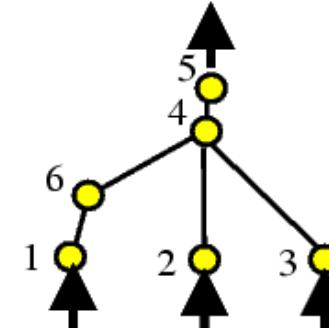
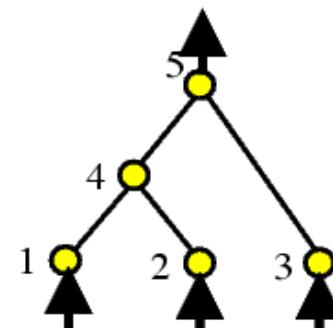
01100001 00000

Parent1

1	2	4	5	6
1->4	2->4	2->5 DISAB	3->5	4->5

Parent2

1	2	3	4	6	7	8
1->4 DISAB	2->4	3->4	2->5 DISAB	4->5	1->6	6->4



disjoint

Parent1

1	2
1->4	2->4

4	5	6
2->5 DISAB	3->5	4->5

Parent2

1	2	3	4
1->4 DISAB	2->4	3->4	2->5 DISAB

disjoint

6	7	8
4->5	1->6	6->4

excess excess

Modifikace topologie, křížení 2/2

01001110 01100101

01110101 01110010

01101111 01101110

01101

01100

01110

01110

01101

01100

01101

01110

01100

01111

01110

01100

01110

01100

00101

01000

01001

01000011 01010110

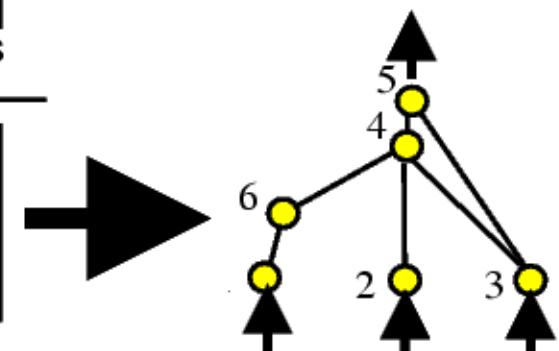
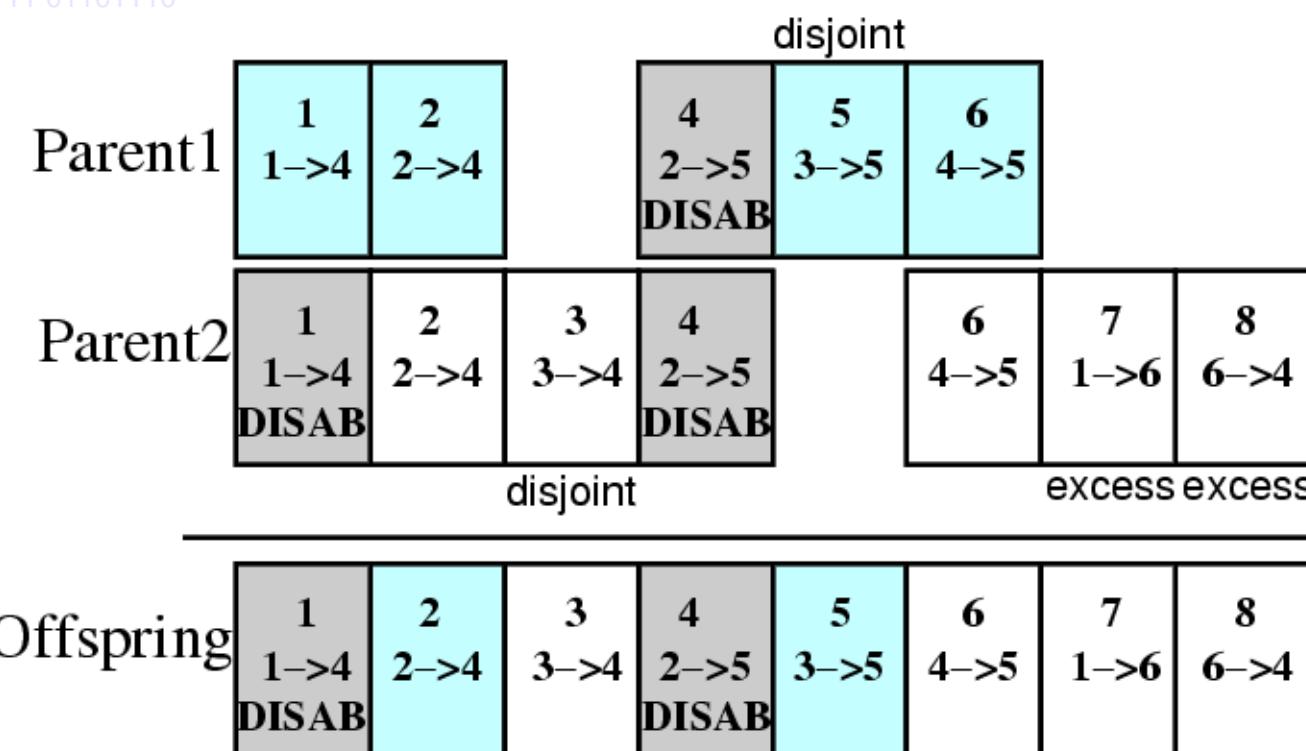
01010101 01010100

00101100 00100000

01010000 01110010

01100001 01101000

01100001 000000000





Reference

- Mařík, V., Štěpánková, O., Lažanský, J.: Umělá inteligence 3,
Academia, Praha 2000, ISBN-80-200-0472-6
- Mařík, V., Štěpánková, O., Lažanský, J.: Umělá inteligence 4,
Academia, Praha 2003, ISBN-80-200-1044-0
- Implementace algoritmu diferenciální evoluce a další informace
<http://www.ft.utb.cz/people/zelinka/soma/>
- Pokorný, M.: Umělá inteligence v modelování a řízení,
BEN Technická literatura, Praha 1996, ISBN 80-901984-4-9
- Williams, R. J., Zipser, D.: A Learning Algorithm for Continually
Running Fully Recurrent Neural Networks, Neural computation 1

Automatické těžení znalostí z dat

Trendy:

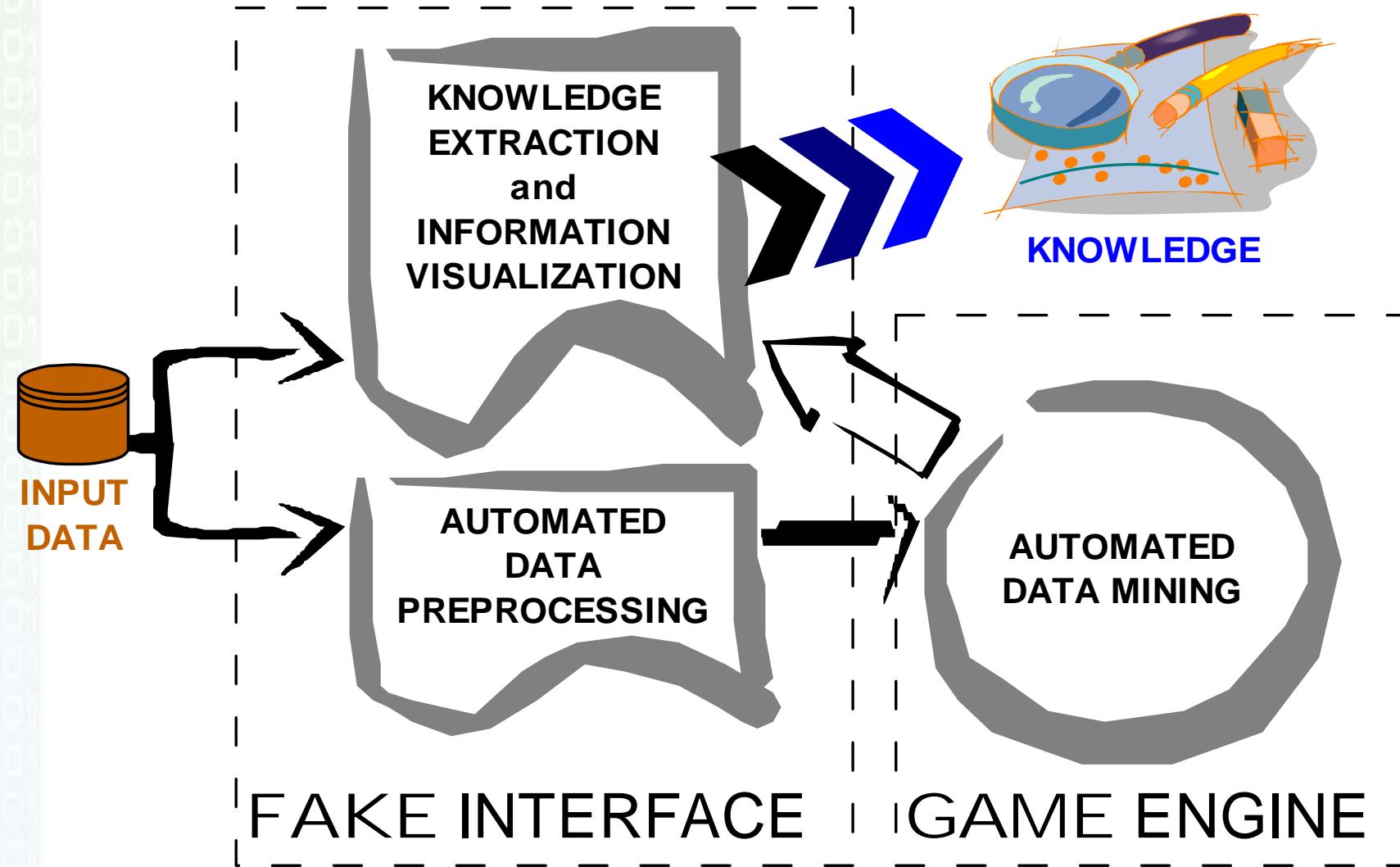
- Je těžké se stát DM specialistou (a drahé si takového specialistu najmout)
- Přesto hodně firem potřebuje analyzovat data a vytěžit znalosti
- Řešením je specializovaný software, který uživatele odstíní od miliónů konfiguračních nastavení, kterým nerozumí, a přesto poskytne použitelný výsledek.

- Co musí takový software umět?

Automatizace předzpracování dat

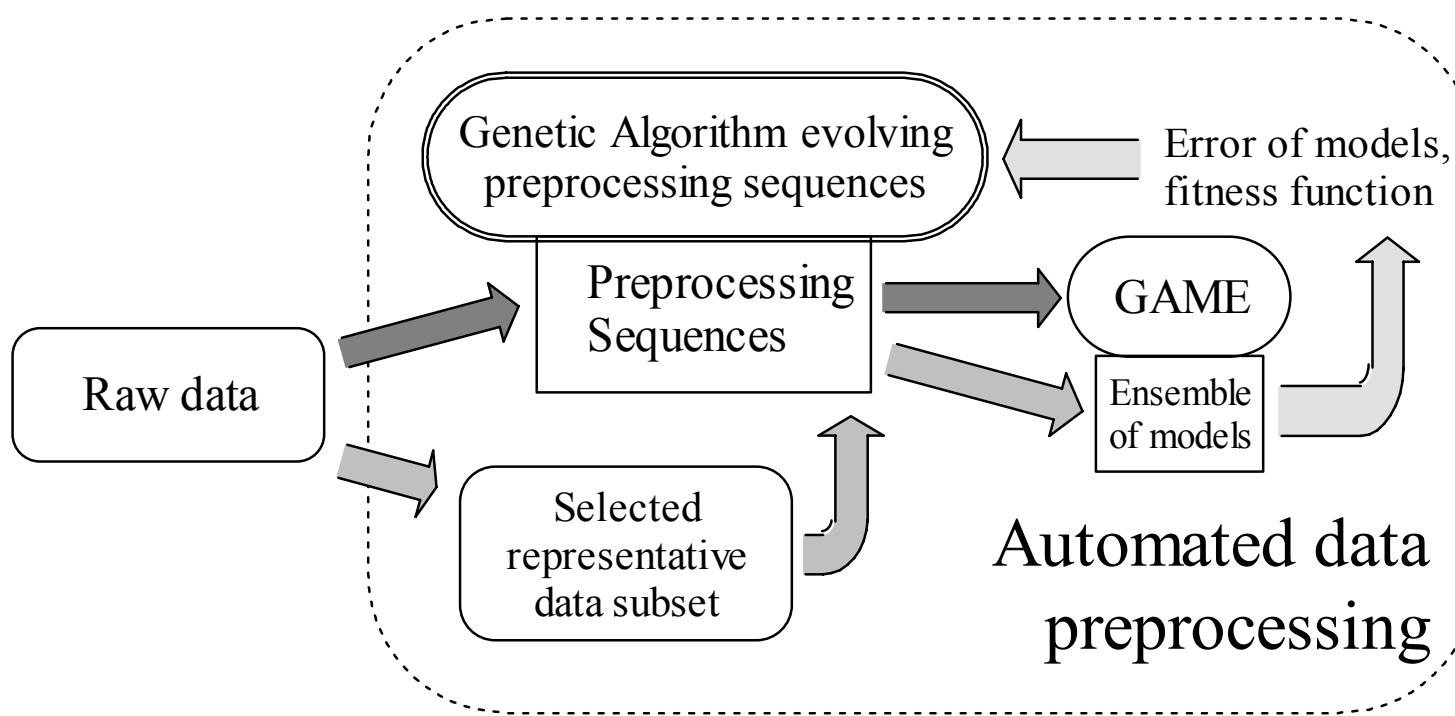
- **Znalost následujících slajdů nebude vyžadována u zkoušky**
- Předzpracování dat jde obecně automatizovat velmi těžko – nutná velká opatrnost
- Ukázka, jak se o to snažíme v naší výzkumné skupině:

FAKE GAME software



Automated data preprocessing

- Pro každý vstupní atribut vyšlechtíme genetickým algoritmem posloupnost předzpracovacích metod:

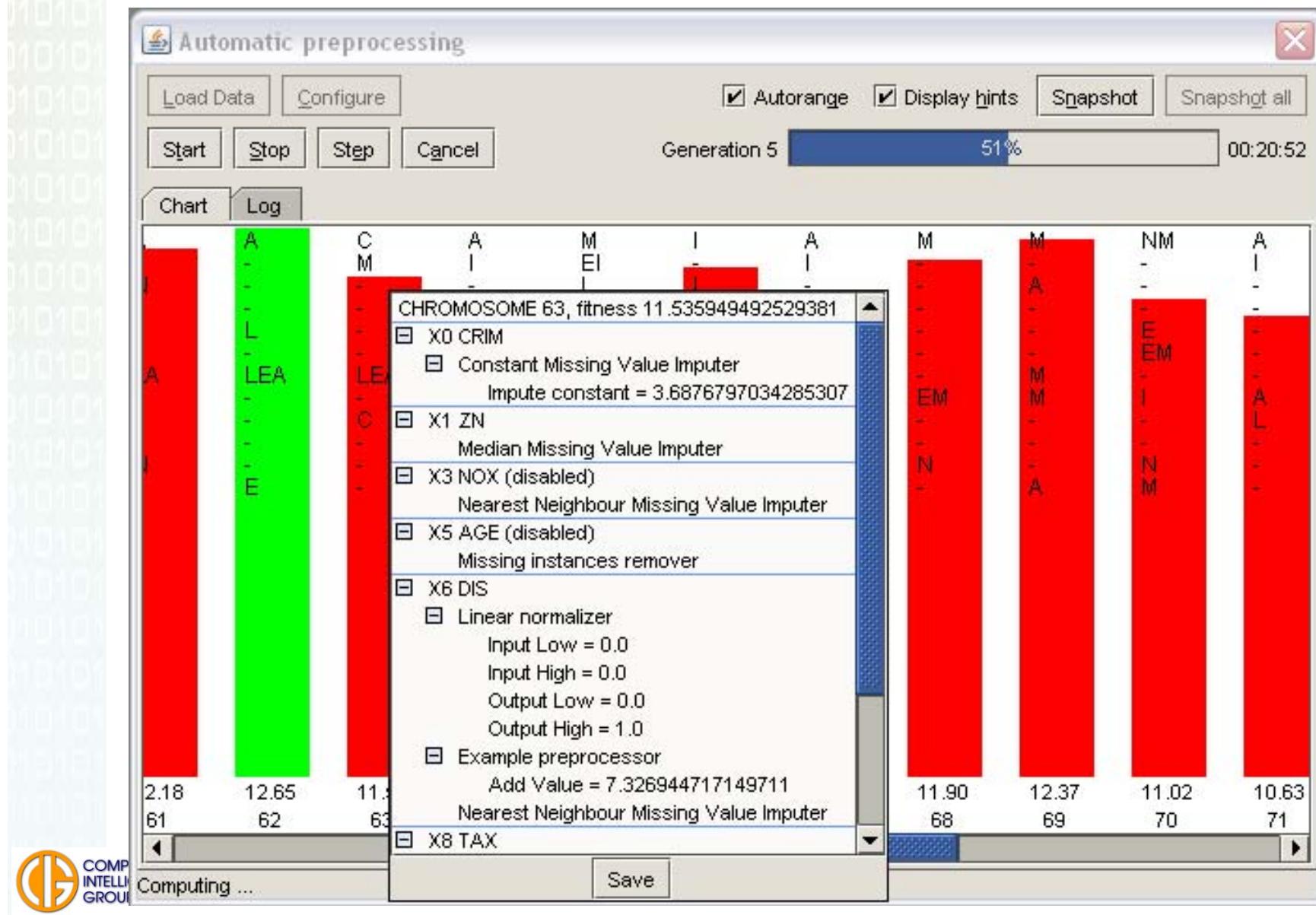


Metody které máme implementovány

- Preprocessing methods
 - Experiments
 - Example preprocessor
 - Noise Adder
 - Imports
 - Load RAW Data
 - Load GAME Data
 - Test attribute types
 - Decode nominals to 1-of-N
 - Mark Missing Values
 - Missing values
 - Constant Missing Value Imputer
 - Median Missing Value Imputer
 - Nearest Neighbour Missing Value Imputer
 - Missing instances remover
 - Another instance value data imputer
 - Normalization
 - Example normalizer
 - Linear normalizer
 - SoftMax normalizer
 - Mean value normalizer
 - Z-score normalizer
 - Custom JS normalizer
 - Custom Octave normalizer
 - Data reduction
 - Random data reducer
 - Outlayer remover
 - Leave-out neighbours
 - KMeans data replacer
 - Principal Component Analysis
 - KD-Tree cell replacer
 - Discretisation
 - Adaptive binning
 - Clustering
 - K-Means Clustering
 - K-Means Clustering with Radius
 - K-Means Clustering Auto
 - X-Means Clustering

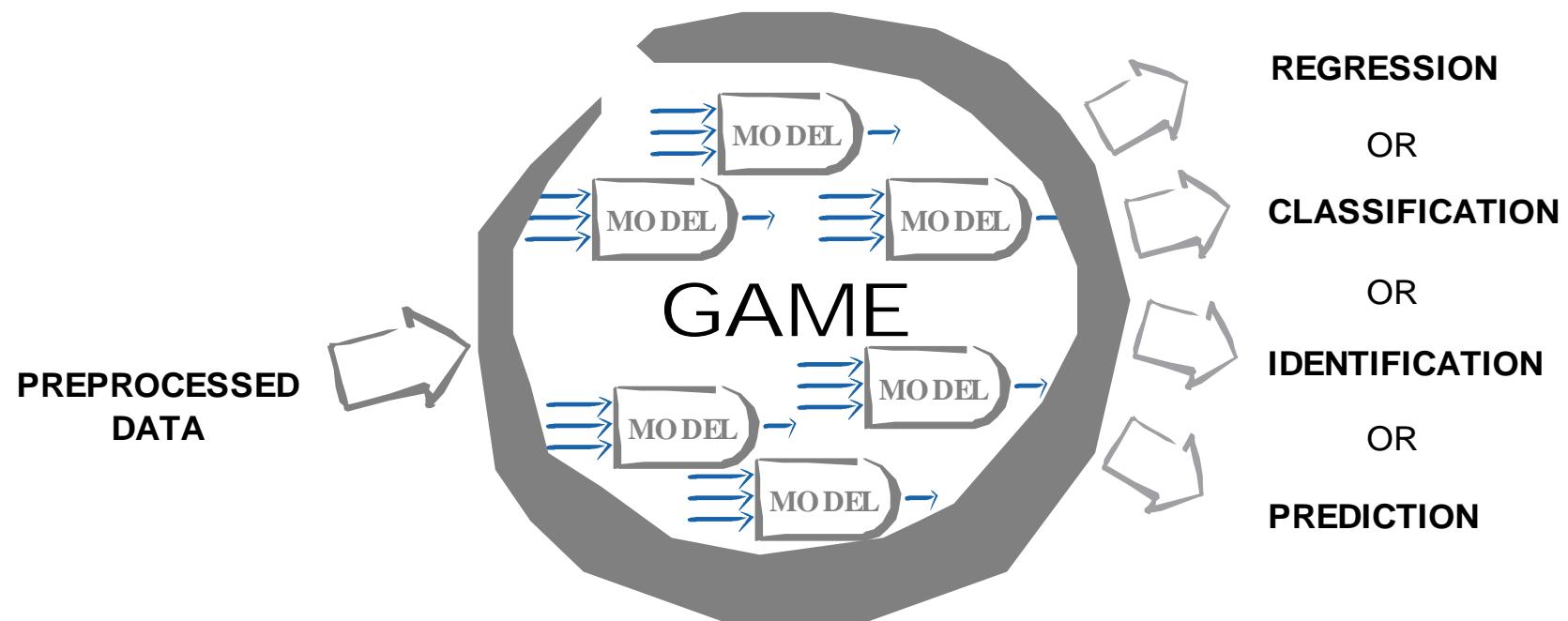
	sepal_length	sepal_width	petal_length	petal_width	Iris-setosa	Iris-versico...	Iris-virginica	
Input/Output...	<Input attrib...	<Input attrib...	<Input attrib...	<Input attrib...	<Output attri...	<Output attri...	<Output attri...	▲
Manually sel...	<Numeric ty...							
Instance 0	5.1	3.5	1.4	0.2	1.0	0.0	0.0	
Instance 1	4.9	3.0	1.4	0.2	1.0	0.0	0.0	
Instance 2	4.7	3.2	1.3	0.2	1.0	0.0	0.0	
Instance 3	4.6	3.1	1.5	0.2	1.0	0.0	0.0	
Instance 4	5.0	3.6	1.4	0.2	1.0	0.0	0.0	
Instance 5	5.4	3.9	1.7	0.4	1.0	0.0	0.0	
Instance 6	4.6	3.4	1.4	0.3	1.0	0.0	0.0	
Instance 7	5.0	3.4	1.5	0.2	1.0	0.0	0.0	
Instance 8	4.4	2.9	1.4	0.2	1.0	0.0	0.0	
Instance 9	4.9	3.1	1.5	0.1	1.0	0.0	0.0	
Instance 10	5.4	3.7	1.5	0.2	1.0	0.0	0.0	
Instance 11	4.8	3.4	1.6	0.2	1.0	0.0	0.0	
Instance 12	4.8	3.0	1.4	0.1	1.0	0.0	0.0	
Instance 13	4.3	3.0	1.1	0.1	1.0	0.0	0.0	
Instance 14	5.8	4.0	1.2	0.2	1.0	0.0	0.0	
Instance 15	5.7	4.4	1.5	0.4	1.0	0.0	0.0	
Instance 16	5.4	3.9	1.3	0.4	1.0	0.0	0.0	
Instance 17	5.1	3.5	1.4	0.3	1.0	0.0	0.0	
Instance 18	5.7	3.8	1.7	0.3	1.0	0.0	0.0	
Instance 19	5.1	3.8	1.5	0.3	1.0	0.0	0.0	
Instance 20	5.4	3.4	1.7	0.2	1.0	0.0	0.0	
Instance 21	5.1	3.7	1.5	0.4	1.0	0.0	0.0	
Instance 22	4.6	3.6	1.0	0.2	1.0	0.0	0.0	
Instance 23	5.1	3.3	1.7	0.5	1.0	0.0	0.0	
Instance 24	4.8	3.4	1.9	0.2	1.0	0.0	0.0	
Instance 25	5.0	3.0	1.6	0.2	1.0	0.0	0.0	
Instance 26	5.0	3.4	1.6	0.4	1.0	0.0	0.0	
Instance 27	5.2	3.5	1.5	0.2	1.0	0.0	0.0	
Instance 28	5.2	3.4	1.4	0.2	1.0	0.0	0.0	
Instance 29	4.7	3.2	1.6	0.2	1.0	0.0	0.0	
Instance 30	4.8	3.1	1.6	0.2	1.0	0.0	0.0	
Instance 31	5.4	3.4	1.5	0.4	1.0	0.0	0.0	
Instance 32	5.2	4.1	1.5	0.1	1.0	0.0	0.0	

Výstup genetického algoritmu

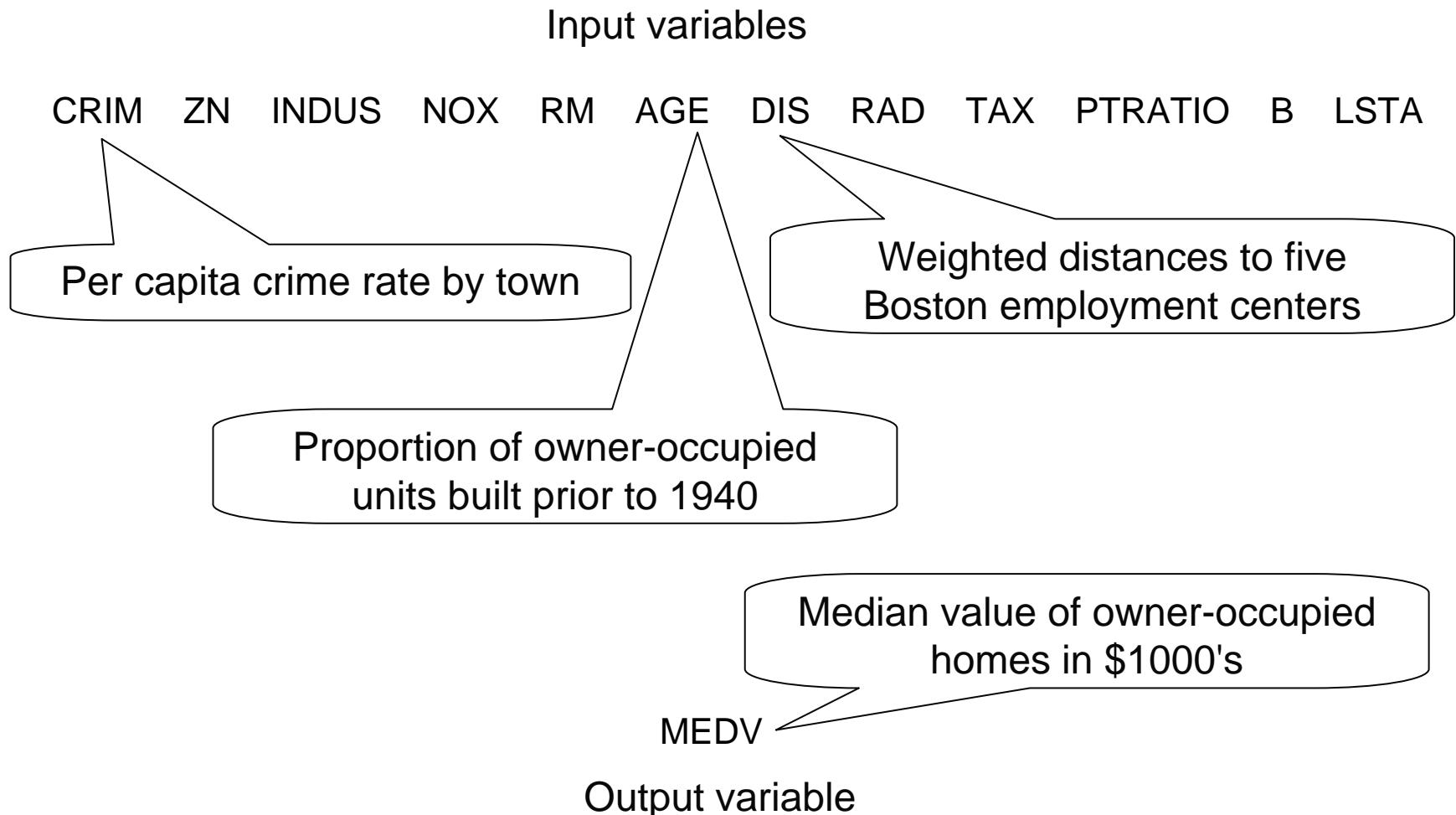


Automatizace vytěžování dat

- Algoritmy se musejí adaptovat na data



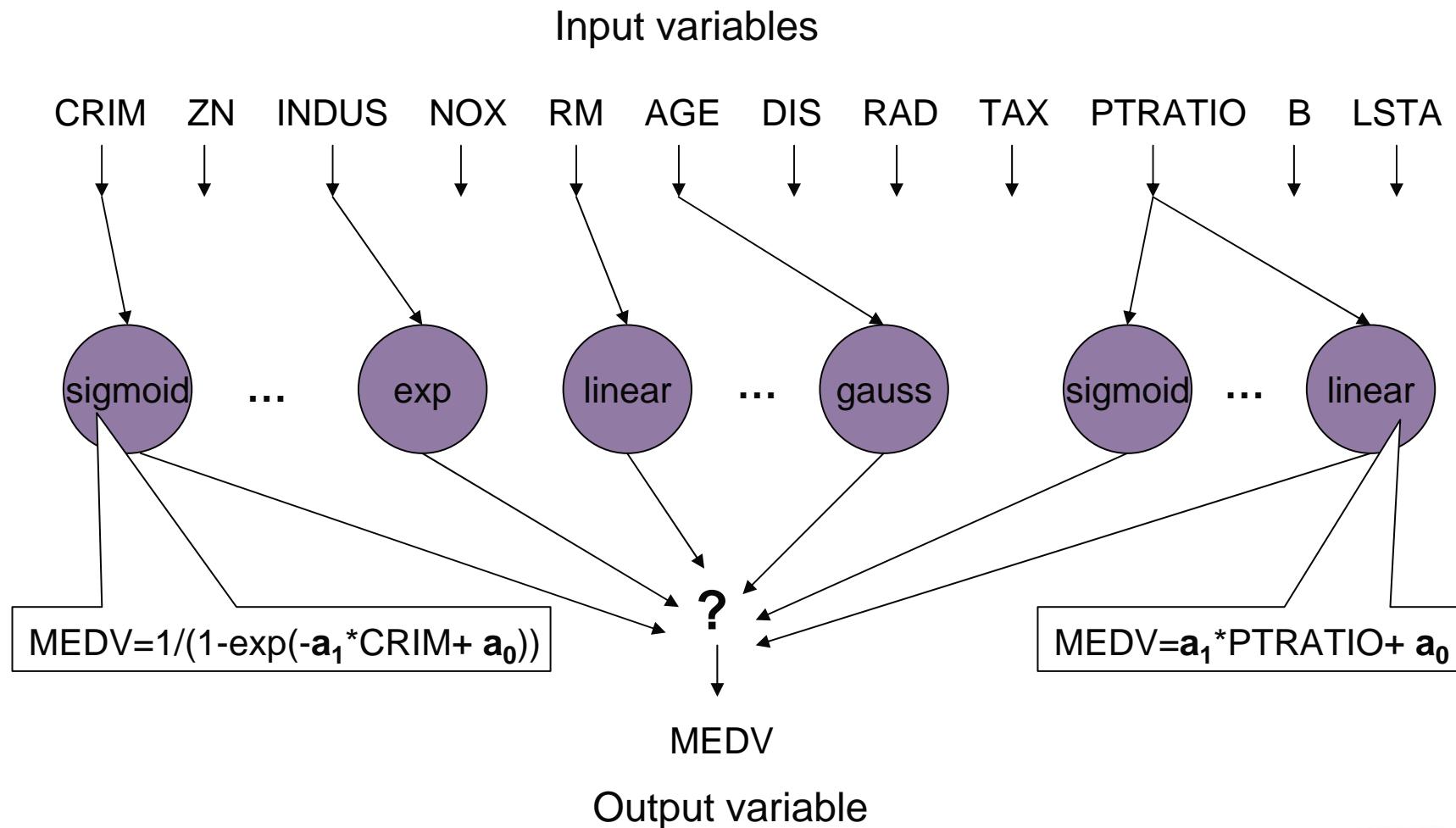
Příklad: Housing data



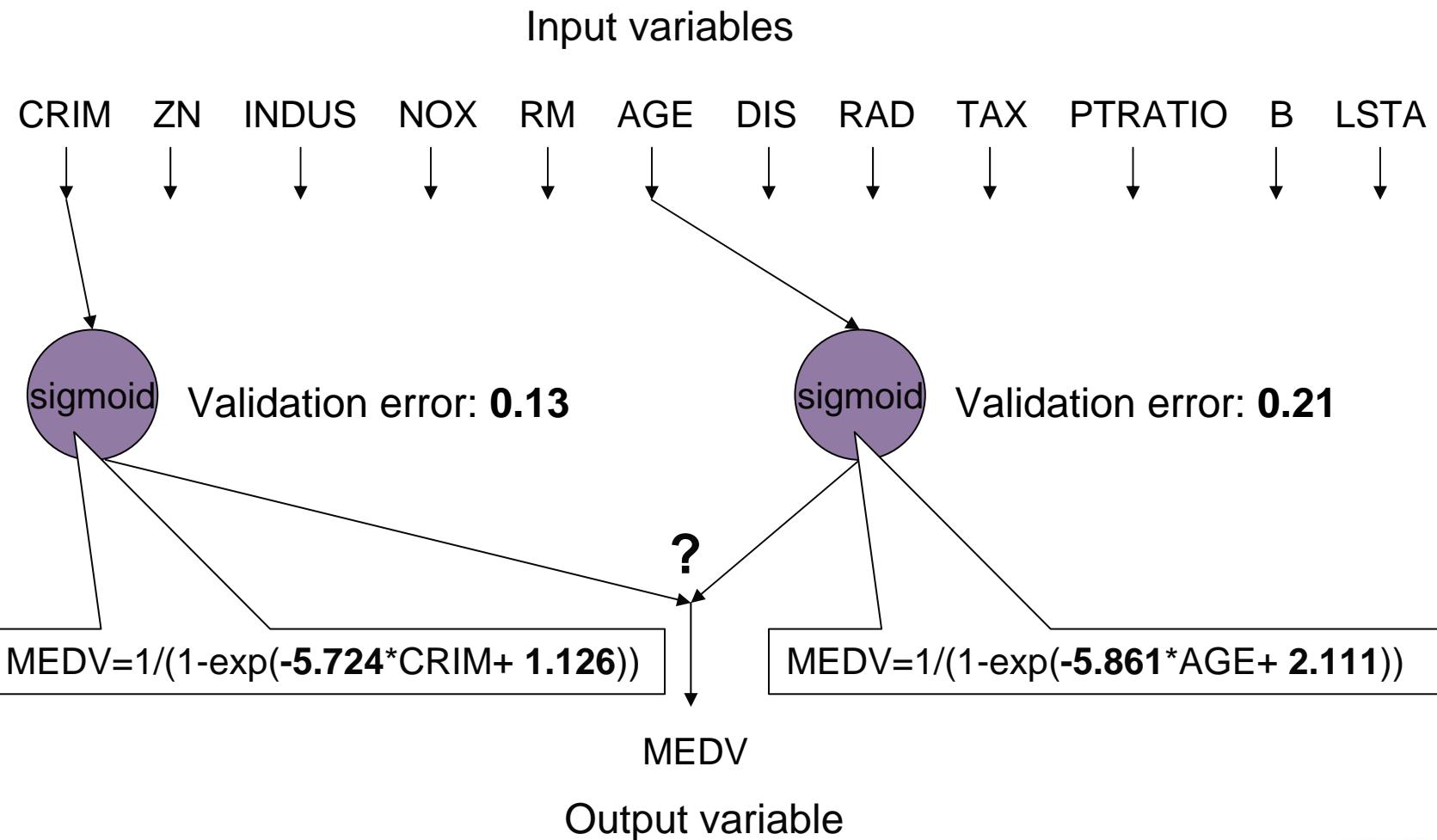
Housing data – records

	Input variables												Output variable	
	CRIM	ZN	INDUS	NOX	RM	AGE	DIS	RAD	TAX	PTRATIO	B	LSTA	MEDV	
A	24	0.00632	18	2.31	53.8	6.575	65.2	4.09	1	296	15.3	396.9	4.98	
	21.6	0.02731	0	7.07	46.9	6.421	78.9	4.9671	2	242	17.8	396.9	9.14	
											
B	A = Training set ... to adjust weights and coefficients of neurons													
	B = Validation set ... to select neurons with the best generalization													
C	C = Test set ... not used during training													

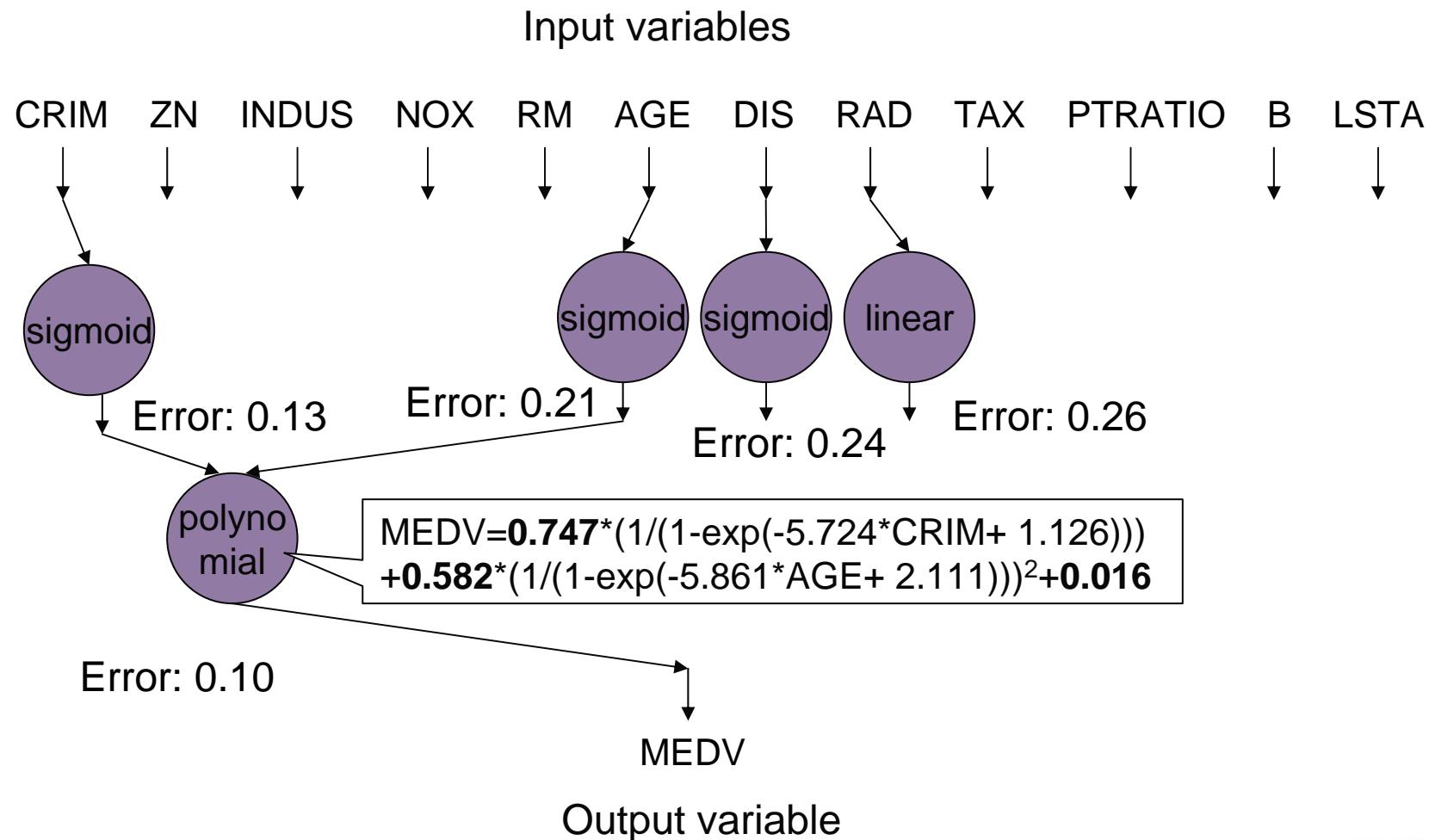
Housing data – inductive model



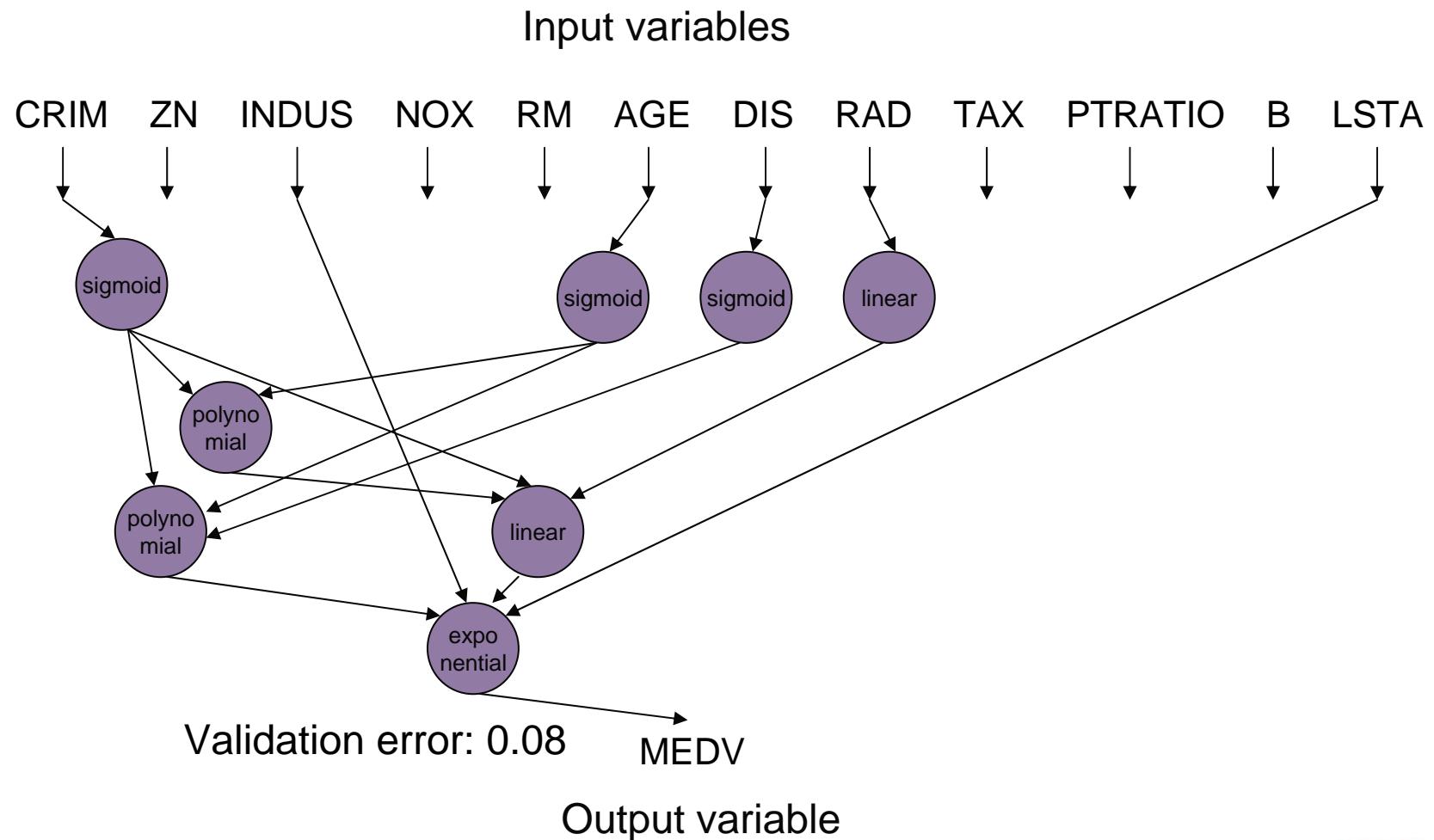
Housing data – inductive model



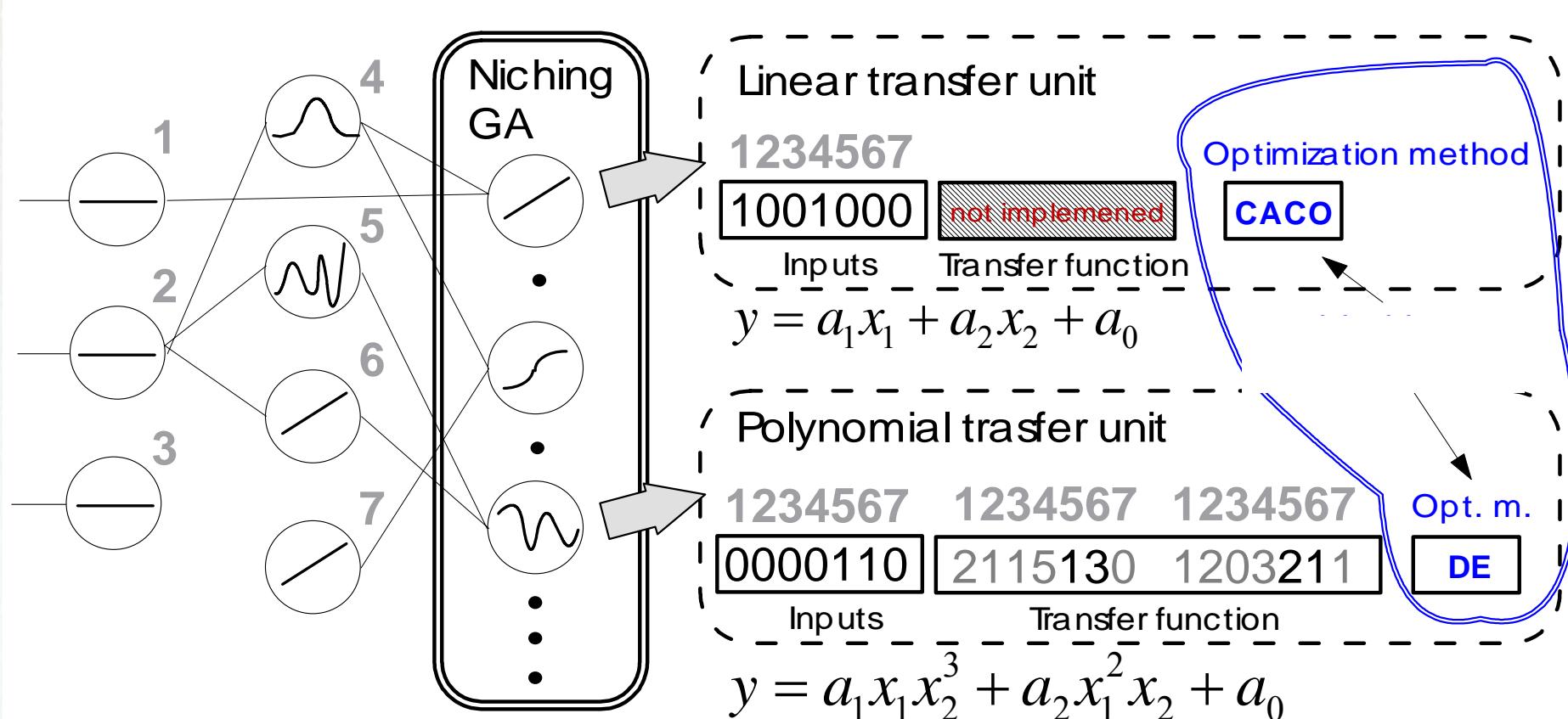
Housing data – inductive model



Housing data – inductive model



Modely jsou šlechtěny speciálním genetickým algoritmem



Fitness of unit: inverse of its error on the validation data set

Další krok – šlechtění ensemblů

■ Hierarchie modelů

Bagging

Boosting

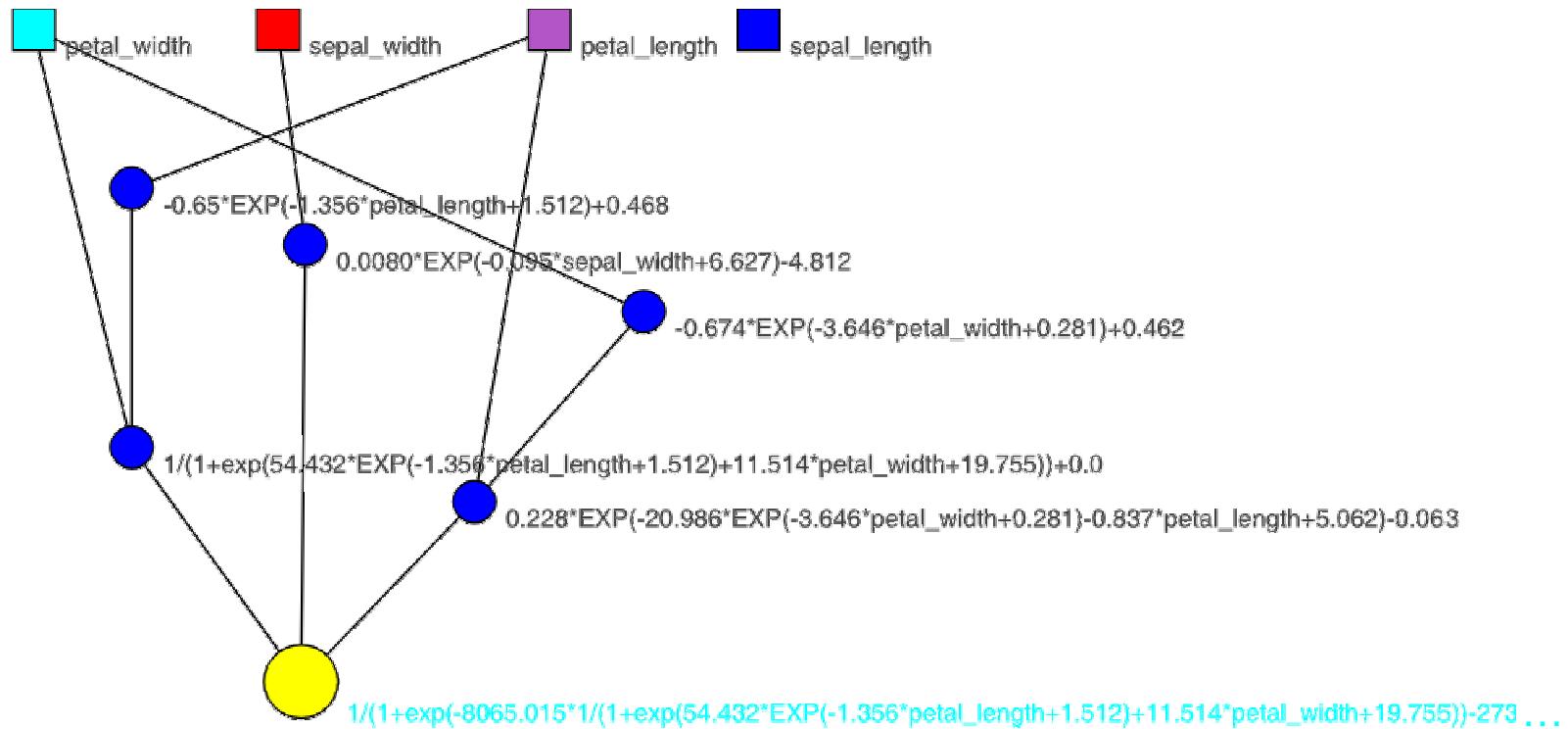
4x Gaussian

Stacking

5x Sigmoid using Sine

■ <http://sourceforge.net/projects/fakegame/files/fakegame/automated-model-selection/automatedMS.zip/download>

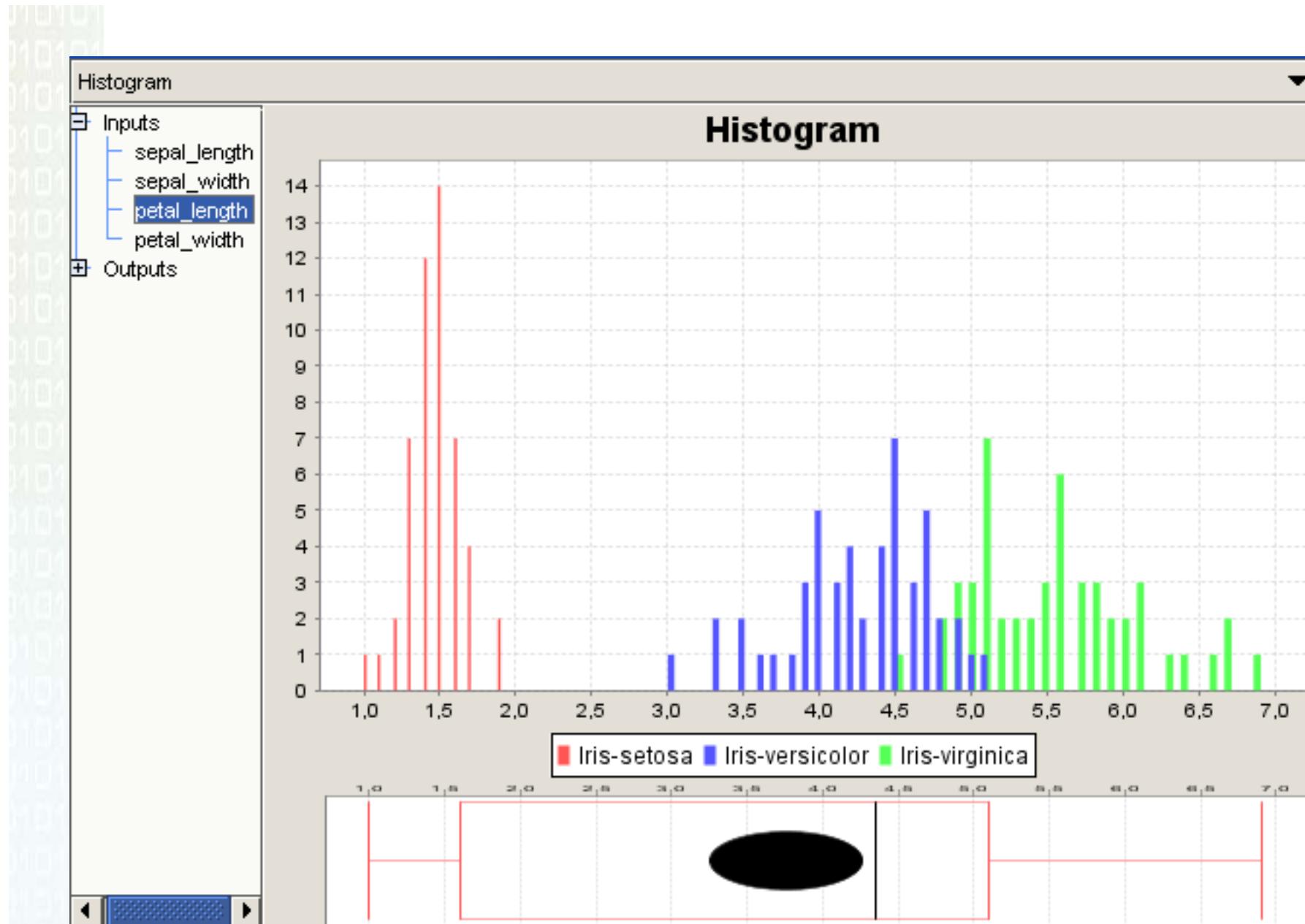
Naroste model a co s ním?



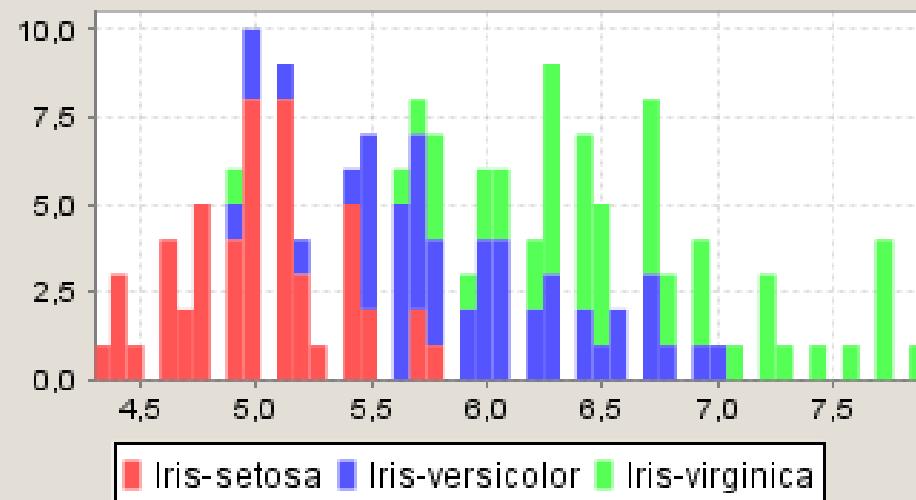
Automaticky extrahované informace

- Co všechno může být užitečné?
- Jak to udělat automaticky?

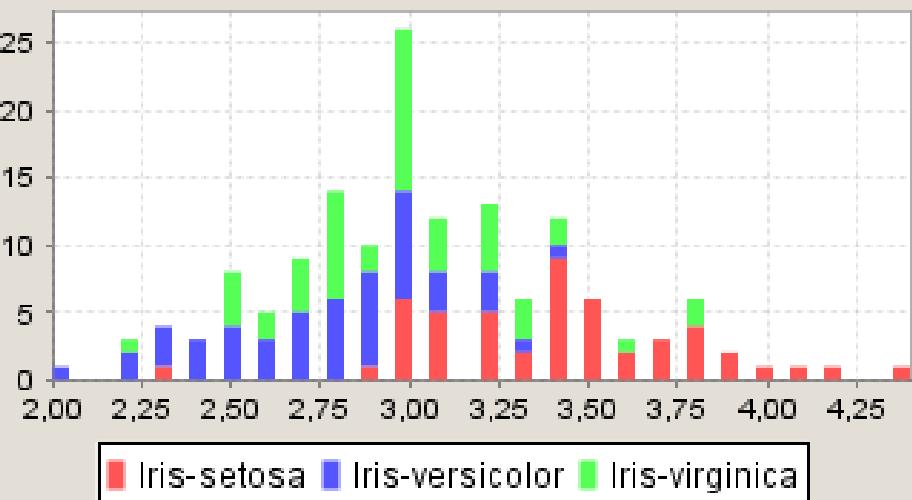




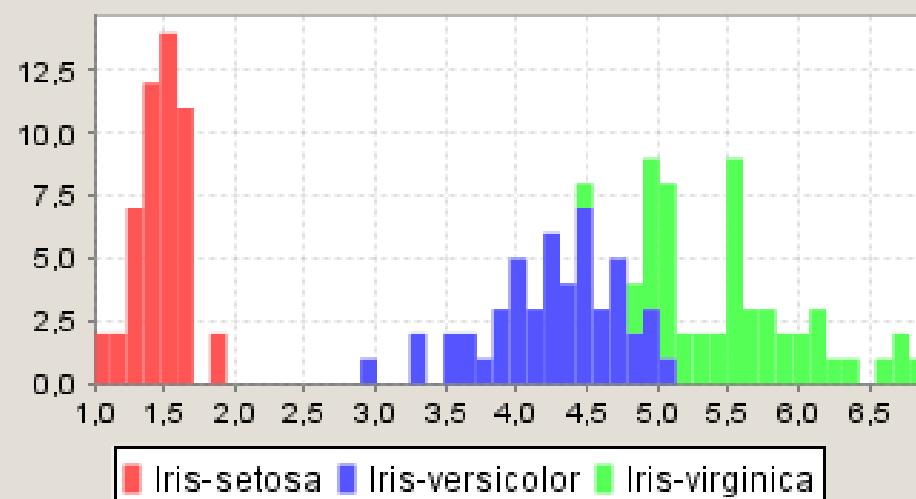
sepal_length



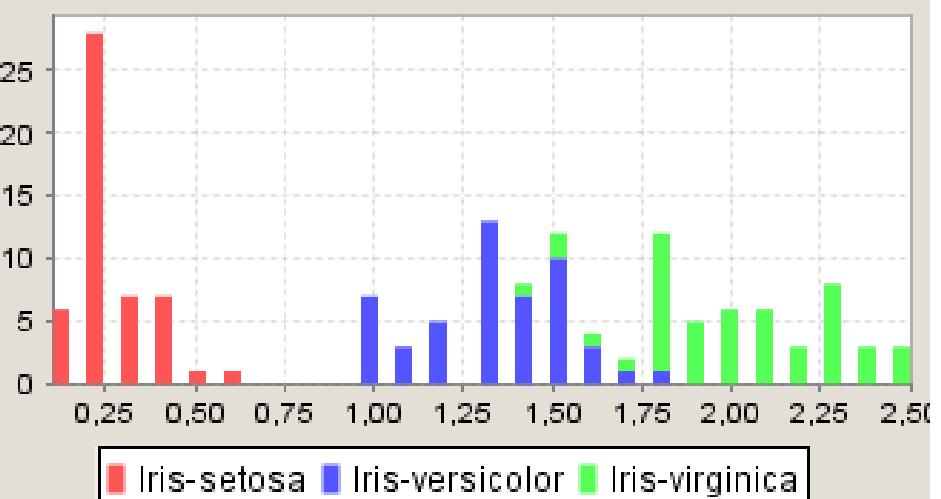
sepal_width



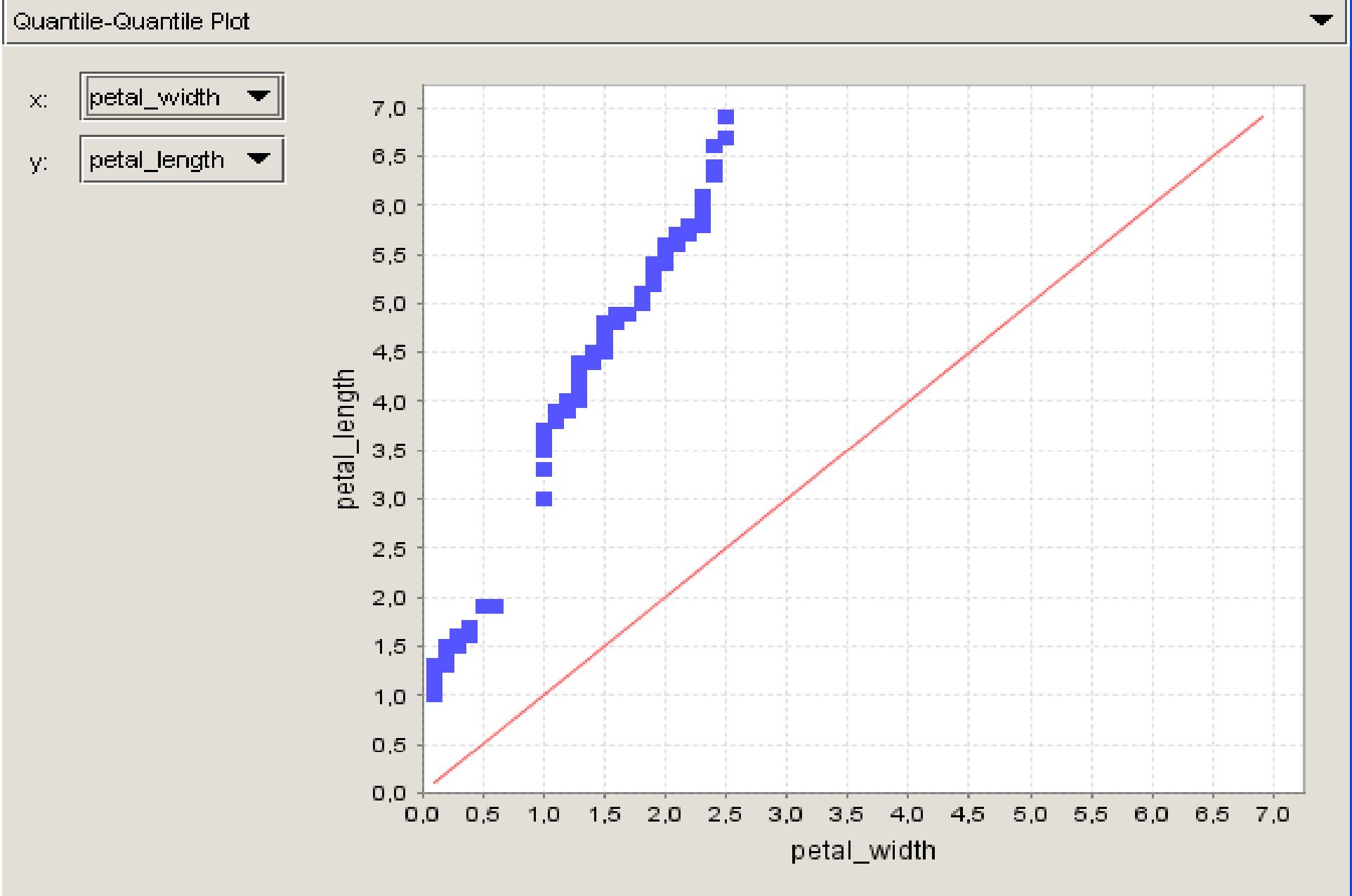
petal_length



petal_width



	Input	Min	25%	50%	75%	Max	Average	Variance
Inputs	sepal_length	4.3	5.1	5.8	6.4	7.9	5.843333333...	0.68112222...
Outputs	sepal_width	2.0	2.8	3.0	3.3	4.4	3.053999999...	0.18675066...
	petal_length	1.0	1.5	4.3	5.1	6.9	3.75866666...	3.09242488...
	petal_width	0.1	0.3	1.3	1.8	2.5	1.19866666...	0.57853155...



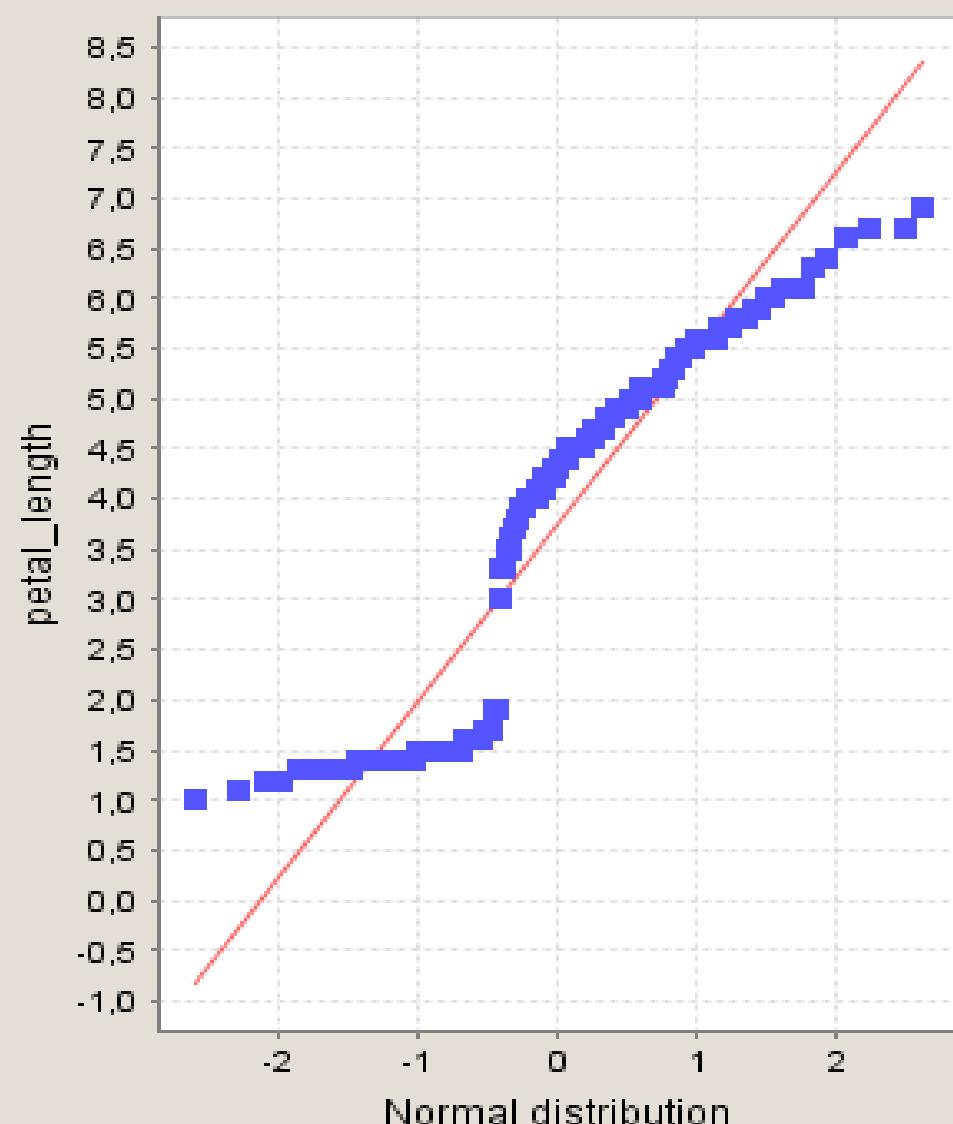
Probability Plot

x: Normal distribution

y: petal_length

EX (3.256383807786332; 4.260949525547006)

DX (1.5848096213641532; 1.9903089802139846)



Covariance matrix

Covariance:

	sepal_length	sepal_width	petal_length	petal_width
sepal_length	0.6811222222222222	-0.03900666666666667	1.2651911111111114	0.5134577777777779
sepal_width	-0.03900666666666667	0.18675066666666667	-0.319568000000000013	-0.11719466666666661
petal_length	1.2651911111111114	-0.319568000000000013	3.0924248888888854	1.2877448888888892
petal_width	0.5134577777777779	-0.11719466666666661	1.2877448888888892	0.57853155555555559

	Iris-setosa	Iris-versicolor	Iris-virginica	
Iris-setosa	0.2222222222222221	-0.11111111111111092	-0.11111111111111098	▲
Iris-versicolor	-0.11111111111111092	0.2222222222222221	-0.11111111111111098	■
Iris-virginica	-0.11111111111111098	-0.11111111111111098	0.22222222222222168	■
sepal_length	-0.2791111111111111	0.03088888888888896	0.2482222222222218	▼

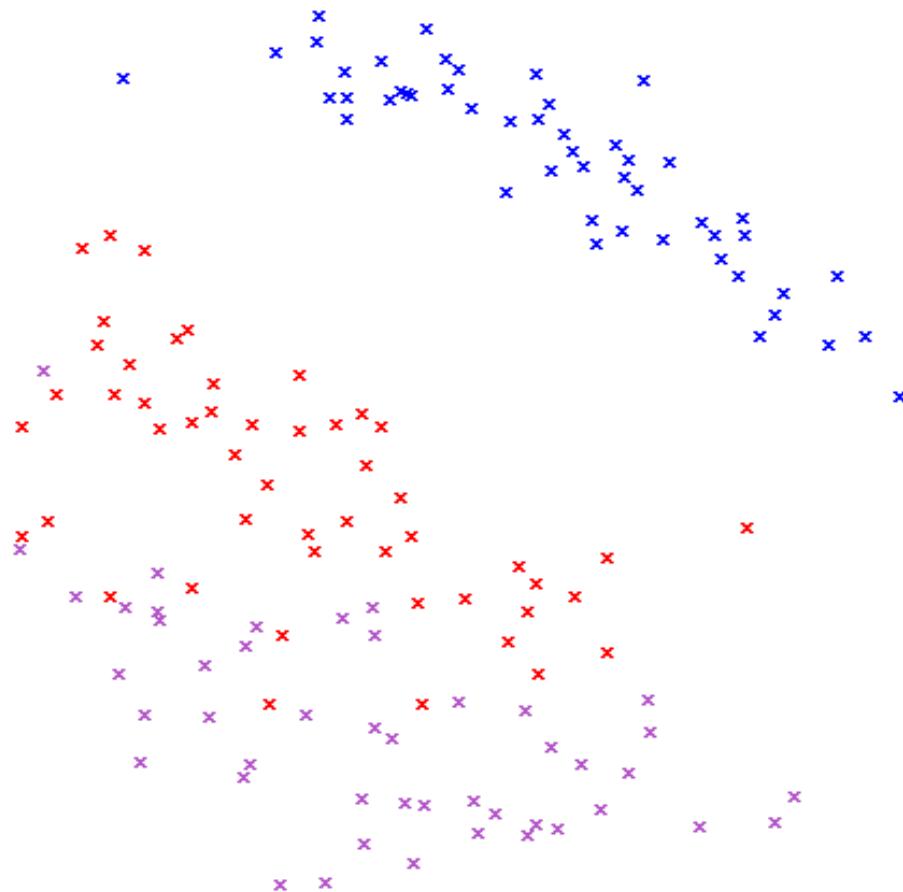
Corelation:

	sepal_length	sepal_width	petal_length	petal_width
sepal_length	0.99333333333338	-0.10864012161765108	0.8659424629228616	0.8125006091467221
sepal_width	-0.10864012161765108	0.9933333333333974	-0.41771265575849487	-0.35416712901639186
petal_length	0.8659424629228616	-0.41771265575849487	0.993333333333375	0.9563387164039631
petal_width	0.8125006091467221	-0.35416712901639186	0.9563387164039631	0.993333333333335

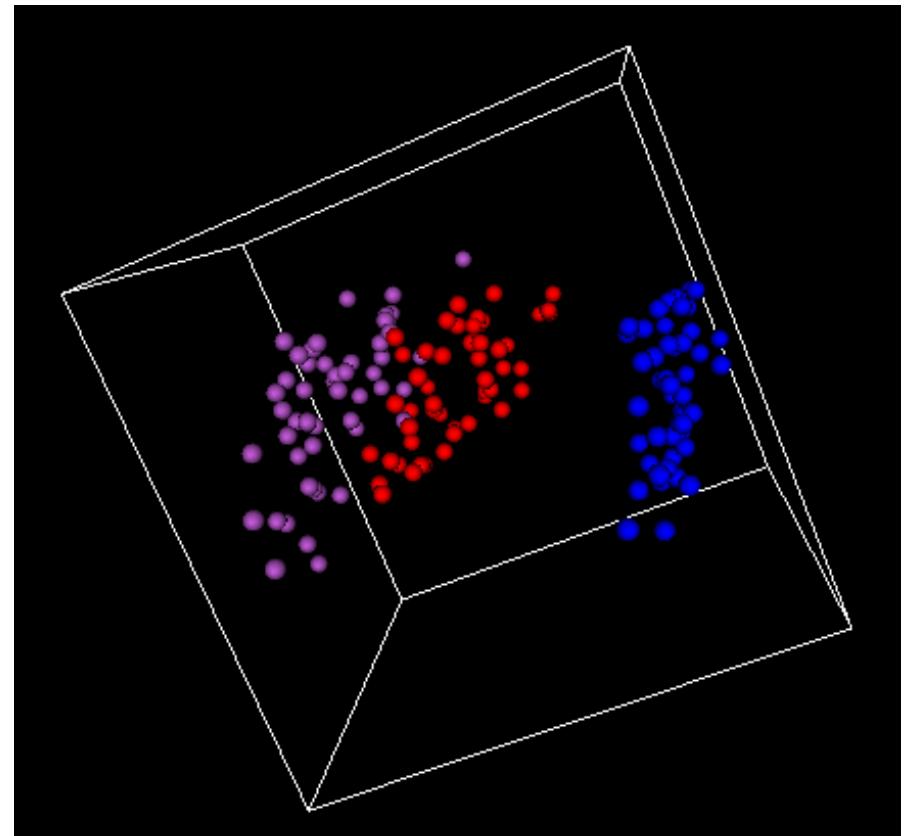
	Iris-setosa	Iris-versicolor	Iris-virginica	
Iris-setosa	0.99333333333333	-0.4966666666666659	-0.4966666666666615	▲
Iris-versicolor	-0.4966666666666659	0.99333333333333	-0.4966666666666615	■
Iris-virginica	-0.4966666666666615	-0.4966666666666615	0.993333333333311	■
sepal_length	-0.7126328975615389	0.07886622035115759	0.6337666772103813	▼

Projekce dat

2D



3D

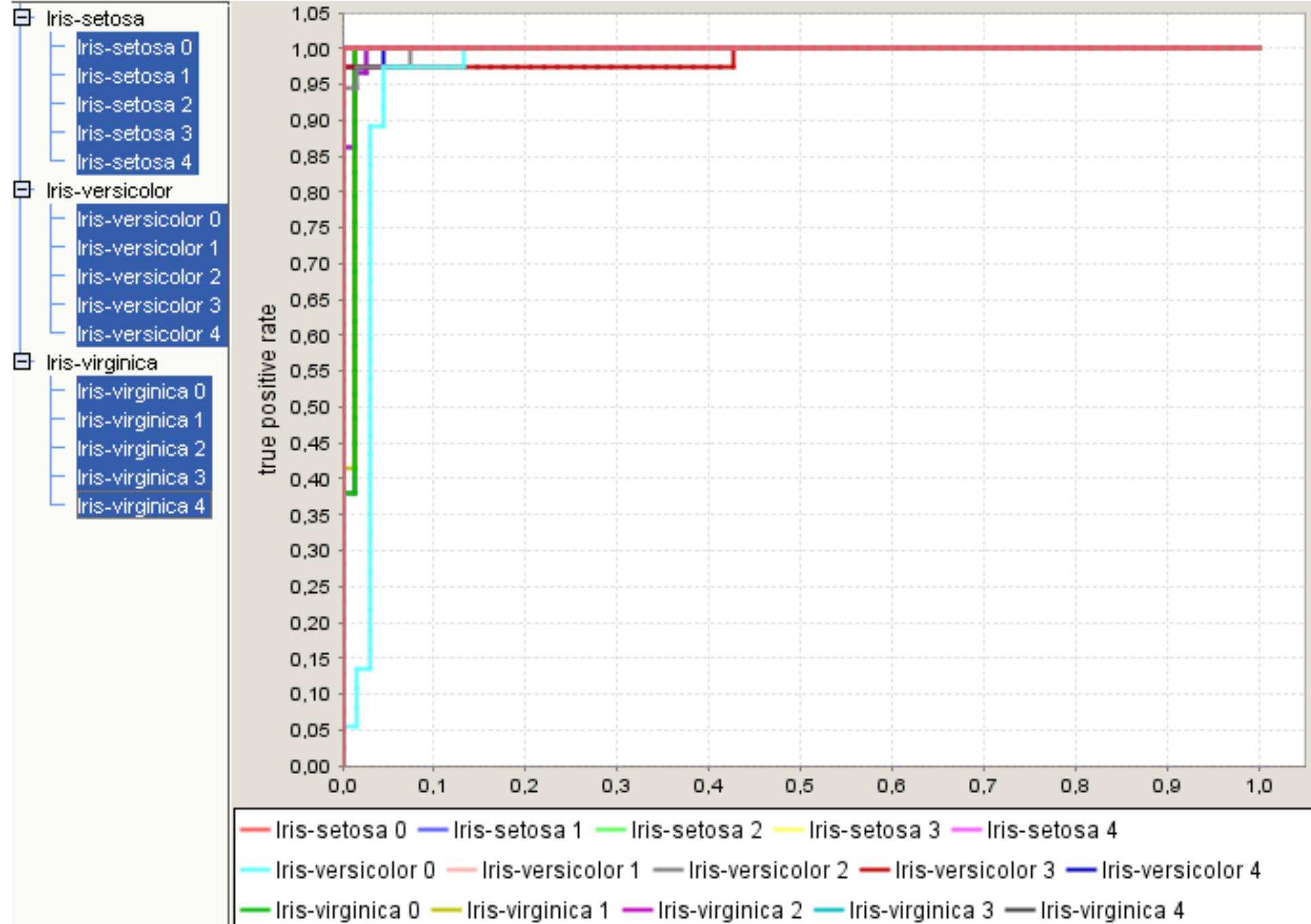


Významnost vstupních atributů

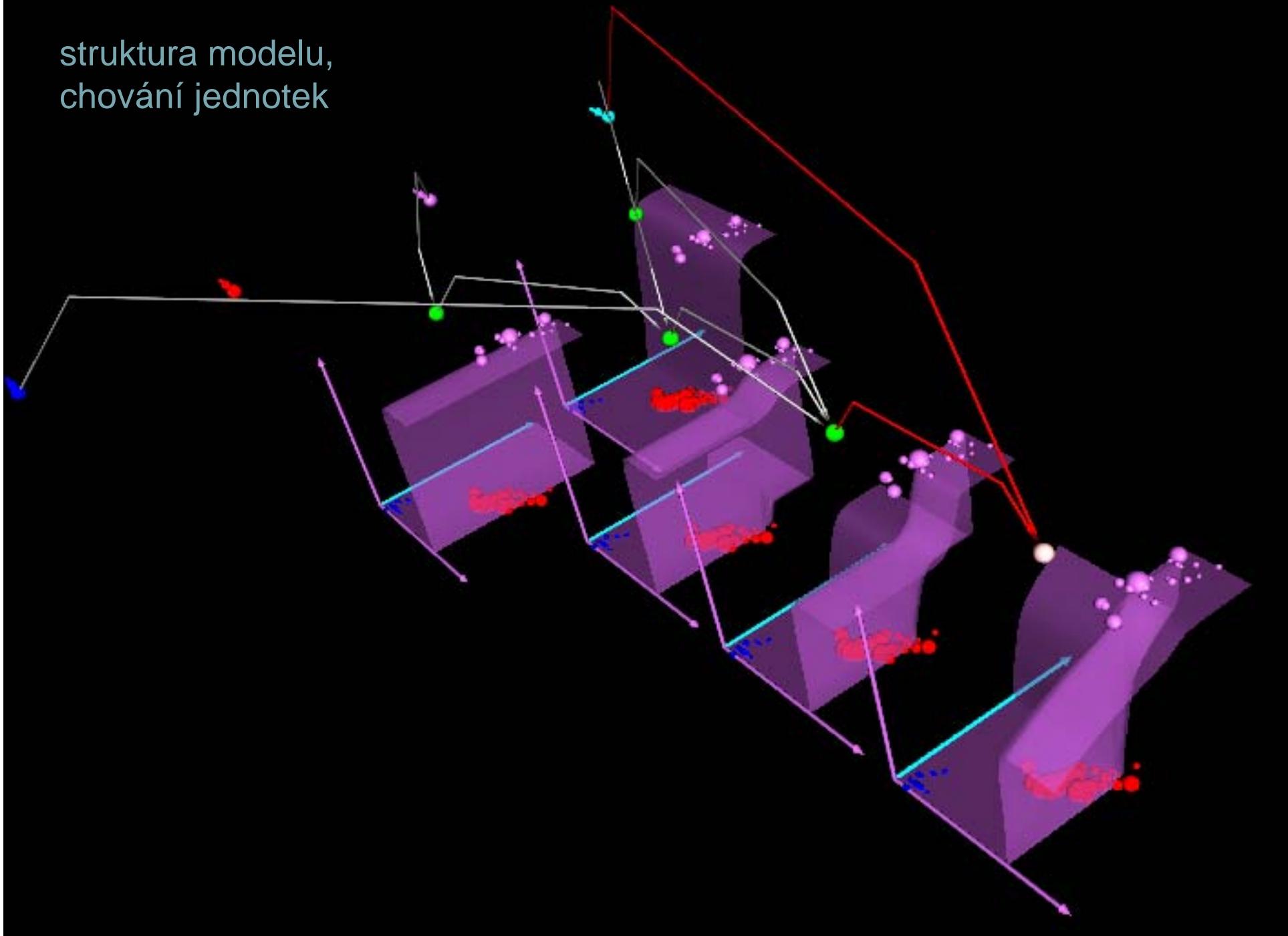
	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	-	28	-	33	34	35	-	40	41	42	43
ChiSquare	7	6	9	10	8	2	1	3	5	4	38	47	28	26	21	48	50	33	29	40	49	27	22	42	32	34	30	17	16	20			
GainRatio	10	6	5	9	7	1	3	2	4	8	38	50	21	49	27	47	29	36	33	22	37	48	24	39	43	28	34	18	20	19			
InfoGain	7	6	8	9	2	10	1	3	5	4	38	47	28	26	29	48	50	40	21	33	46	27	24	22	31	32	43	17	16	20			
OneR	7	9	10	8	2	3	1	6	4	5	38	47	48	26	25	40	29	28	50	21	24	33	35	43	30	34	27	16	11	12			
ReliefF	6	9	7	4	10	3	5	2	1	8	26	37	24	36	27	50	21	46	30	49	28	31	23	45	41	47	48	18	15	16			
SVM	2	4	6	3	7	9	1	10	8	23	47	24	26	39	30	37	42	22	34	21	38	44	5	17	41	14	28	20	45	18			
SU	10	6	7	9	5	1	3	2	4	8	38	50	21	47	49	27	29	33	36	48	37	22	24	44	39	40	30	18	20	19			
GAME 1	7	8	2	6	26	23	9	35	28	4	1	32	25	33	38	40	43	20	42	47	3	5	15	21	22	24	34	36	37	39			
GAME 2	10	7	3	9	37	27	24	50	8	5	23	26	32	46	44	1	2	4	6	11	12	13	19	28	29	30	36	38	39	40			
GAME 3	10	2	5	8	7	6	3	29	26	4	37	42	44	47	36	46	16	34	43	1	9	11	18	23	24	25	32	33	35	38			
GAME 4	3	9	10	41	6	7	5	8	27	25	31	38	49	48	16	39	20	34	44	28	35	1	14	21	22	23	32	33	36	37			
GAME 5	9	6	3	10	28	50	4	38	29	8	13	32	35	36	48	27	20	15	18	23	42	1	14	22	24	25	34	37	39	40			

Nr.Ggroups	Most – least Ranked Features																													
All	1	2	3	4	5	6	46	15	24	37	48	13	17	21	28	29	30	41	43	45	49	50	7	8	9	10	11	12	14	16
1 / 2	1	2	3	4	26	12	35	46	5	6	7	8	9	10	11	13	14	15	16	17	18	19	20	21	22	23	24	25	27	28
1 / 3	1	2	37	21	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	22	23	24	25	26	27	28	29
1 / 4	1	2	41	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29
3	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30
2	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30
1	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30

Receiver operating characteristic



struktura modelu,
chování jednotek



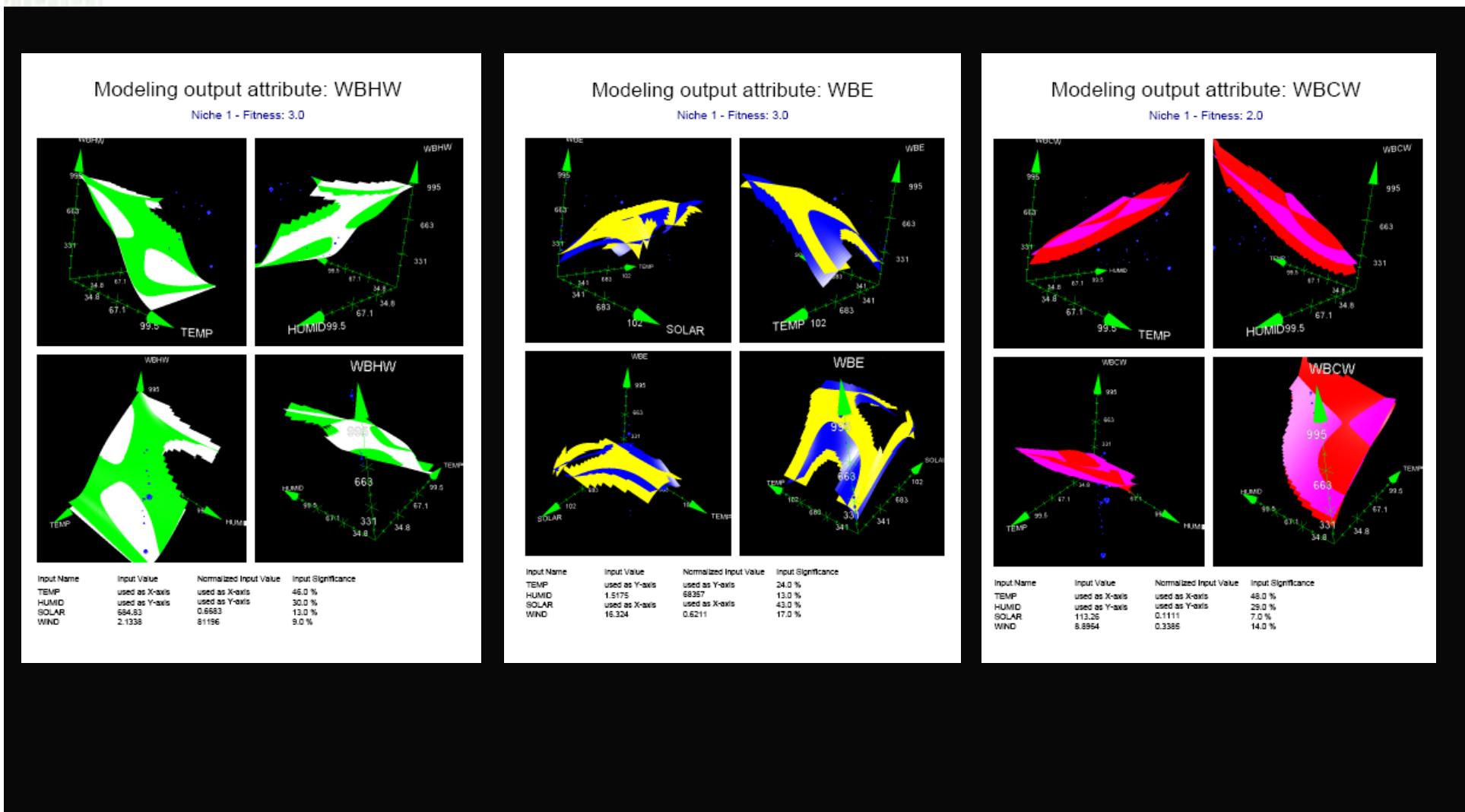
Play FAKE GAME with your data



Log messages

- Evolving preprocessing sequences ...
 - Evolving ensemble of inductive models ...
 - Evolving “interesting” visualizations ...
 - Generating report ...
 - Done ... almost ☺

Toto už umíme generovat automaticky:



Co to znamená?

- Part of the FAKE GAME project (fully automated knowledge extraction from data)
- Ensemble of models is generated on a data set – in this case Building data [proben1]
- “Interesting and credible” areas of model behavior are located in multidimensional input space by means of the niching genetic algorithm.
- These areas are visualized in the 3D graph and the report is produced.
- More:

<http://neuron.felk.cvut.cz/game/doc/fake-game.pdf>

Chcete se na projektu podílet?

- Computational Intelligence Group, Dept. of Computer Science, FEE and FIT, Czech Technical University in Prague, Czech Republic
- logo:

- website: <http://cig.felk.cvut.cz/>
- FAKE GAME project:
<http://sourceforge.net/projects/fakegame>
- Contact:
- Pavel Kordík, kordikp@fit.cvut.cz