Vytěžování Dat
Přednáška 5 – Self Organizing Map

Miroslav Čepek

Katedra počítačů, Computational Intelligence Group

Evropský sociální fond Praha & EU:
Investujeme do vaší budoucnosti

21.10.2014

Shlukovací algoritmy, jejich omezení a cíle

- Jaké znáte shlukovací algoritmy?
  - KMeans,
  - hierarchické shlukování.
- Mají nějaká omezení?
  - KMeans – nedopadne pokaždé stejně, nejasný počet centroidů, zaujetí pro shluky ve tvaru hyperkoulí,
  - hierarchické shlukování – musím spočítat N^2 vzdáleností, pro větší N není jednoduché.
- Můžeme mít i jiné (příbuzné) cíle
  - vizualizace mnohorozměrných dat,
  - redukce dimenze, apod.

Další shlukovací algoritmy

- Existuje spousta dalších algoritmů pro shlukování dat.
- Ukáži vám ještě jeden – **samoorganizující se mapy**.

Kompetitivní učení

- Jedinci (reprezentanti, centroidy, neurony, jedinci) spolu soutěží
  - o něco :).
- Nepotřebuji žádného arbitra (učitele).
  - Učitel by jim říkal, kam se mají přesunout.
  - Každý jedinec to umí zjistit sám.
- Jedinci se učí z příkladů.
- Systém (populace jedinců) se v průběhu času **samoorganizuje**.
- A teď to zkusíme použít na shlukování.

Kompetitivní učení v pohádce

- Vzpomínáte si na Pohádku o králích z minulé přednášky?
- Do země s N domy přijelo K králů a někde se usídlili. Každý král zabral domy, které mu byly nejblíž a z nich vybíral daně. A protože lidé chtěli, aby jim byl král co nejblíž, král se přestěhoval do geometrického středu domů.
- Tím se ale některé domy ocitly blíže jinému králi a tak z nich daně začal vybírat jiný král. Králové se opět přesunuli a tak dále.
- Nejbližší král tedy získá všechny daně z domů, které jsou mu nejblíž.

Kompetitivní učení v KMeans

- KMeans také používá kompetitivní učení. Jak?
- KMeans je trochu skromnější.
  - Reprezentanti (centroidy) soutěží “jen” o data.
  - A nejbližší reprezentant vyhraje – zabere celou instanci.
  - Jiného reprezentanta k instanci nepustí. ”Bere vše”.

Kvantizační chyba

- Minule jsem v souvislosti s KMeans mluvil o optimalizaci (minimalizaci) chyby.
  - Této chybě se říká **kvantizační chyba**.
  - Vyjadřuje průměrnou vzdálenost mezi daty a odpovídajícími reprezentanty.
- Průměrná vzdálenost mezi králi a jejich poddanými:

$$\text{kvantizační chyba} = \frac{1}{\text{počet instancí}} \sum_{i=0}^{k} \sum_{x \in nearest(w_i)} dist(w_i, x)$$

  - w_i je i-tý reprezentant/král.
  - $nearest(w_i)$ je množina instancí/poddaných, které jsou mu nejblíž.
  - x je jedna z instancí.

Vektorová kvantizace

- Cílem (nejen) KMeans je minimalizovat tuto chybu.
- Tím, že minimalizuji kvantizační chybu, tlačím reprezentanty do míst, kde se nachází hodně instancí.
- Snažím se tím aproximovat hustotu instancí pomocí (menší) hustoty reprezentantů.
- Do míst, kde je vysoká hustota instancí, se snažím dostat hodně reprezentatnů a naopak – do míst s málo instancemi dávám málo reprezentatnů.
- Cílem kvantizace vektorů je aproximace hustoty pravděpodobnosti $p(x)$ výskytu instancí x pomocí konečného počtu reprezentantů w_i.
- Důsledkem je mj. komprese, instance lze nahradit odkazy na jejich reprezentanty.

Vítěz NEbere vše

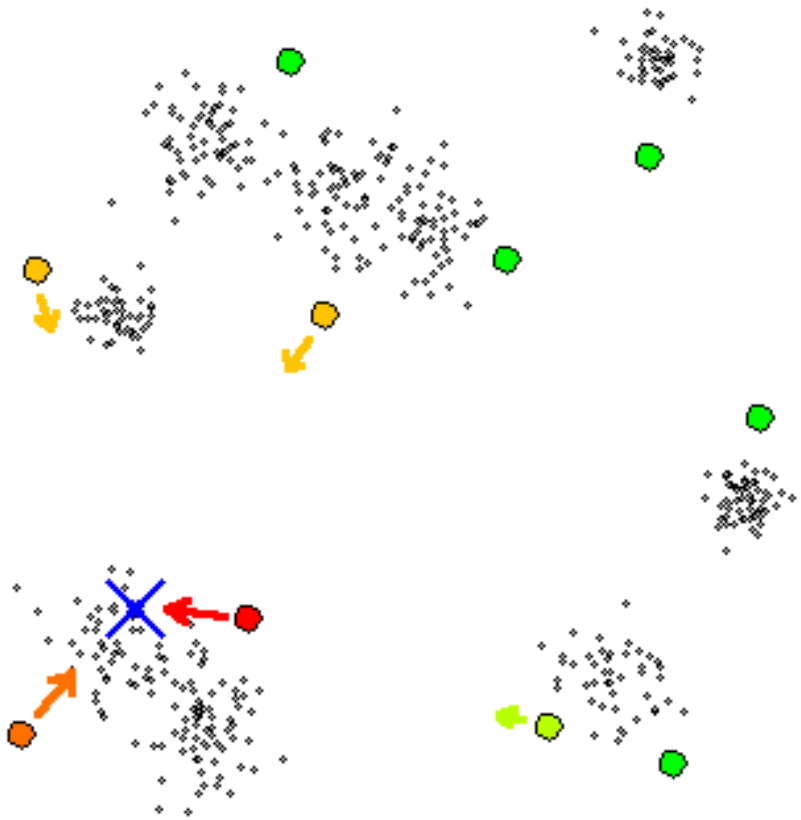
- KMeans
  - "Winner takes all",
  - z domu vybírá daně jen nejbližší král.
- Co když platí, že část daní může vybírat i jiný blízký král?
- Pak už neplatí, že vítěz bere vše a něco zbude i na ostatní.
- Pro míru dělení je důležité *okolí*.
  - Tj. jak daleko se králi ještě vyplatí jet pro svůj díl daní.
  - Malé okolí – vítěz bere vše – daně vybírá jen jeden král
  - Velké okolí – komunismus – každý král dostane z každého domu kousek.

Neuronový plyn

- Jiný způsob, jakým lze minimalizovat kvantizační chybu.
- Rozdíly od KMeans
  - používám okolí, hard $\rightarrow$ soft,
  - jinak počítám nové pozice středů.

1. Náhodně rozmísti reprezentanty a zvol velké okolí.
2. Vyber *nějakou* vstupní instanci x_j.
3. Spočítej vzdálenost mezi x_j a všemi reprezentanty w_i $\forall i$.
4. Uprav pozice všech w_i v závislosti na vzdálenosti od instance a okolí.
5. Zmenši okolí.
6. Pokud ještě chceš pokračovat, pokračuj bodem 2.

Ilustrace iterace Neuronového plynu



Neuronový plyn (II)

- V algoritmu je několik stupňů volnosti.
- Inicializace vah
  - Náhodně nebo rovnoměrné rozprostření po prostoru.
- Výběr *nějaké* vstupní instance
  - Procházíme postupně jednotlivé instance postupně v pevném pořadí.
    - ★ Nevhodné, protože výstup může záviset na pořadí předkládání instancí.
  - Projde všechny instance jednou, pak v jiném pořadí podruhé, atd.
    - ★ Nejčastější volba, prochází se permutace vstupní množiny.
  - Vybírá skutečně náhodně. Čili nezaručuje, že počet předložení síti bude pro všechny instance stejný.
    - ★ Nepoužívá se. Není zaručeno, že nepředložím x_1 10x, pak x_2 12x, atd.

Neuronový plyn (III)

- "Uprav pozice všech středů v závislosti . . ."
- Čím vzdálenější reprezentant, tím se posouvá méně.

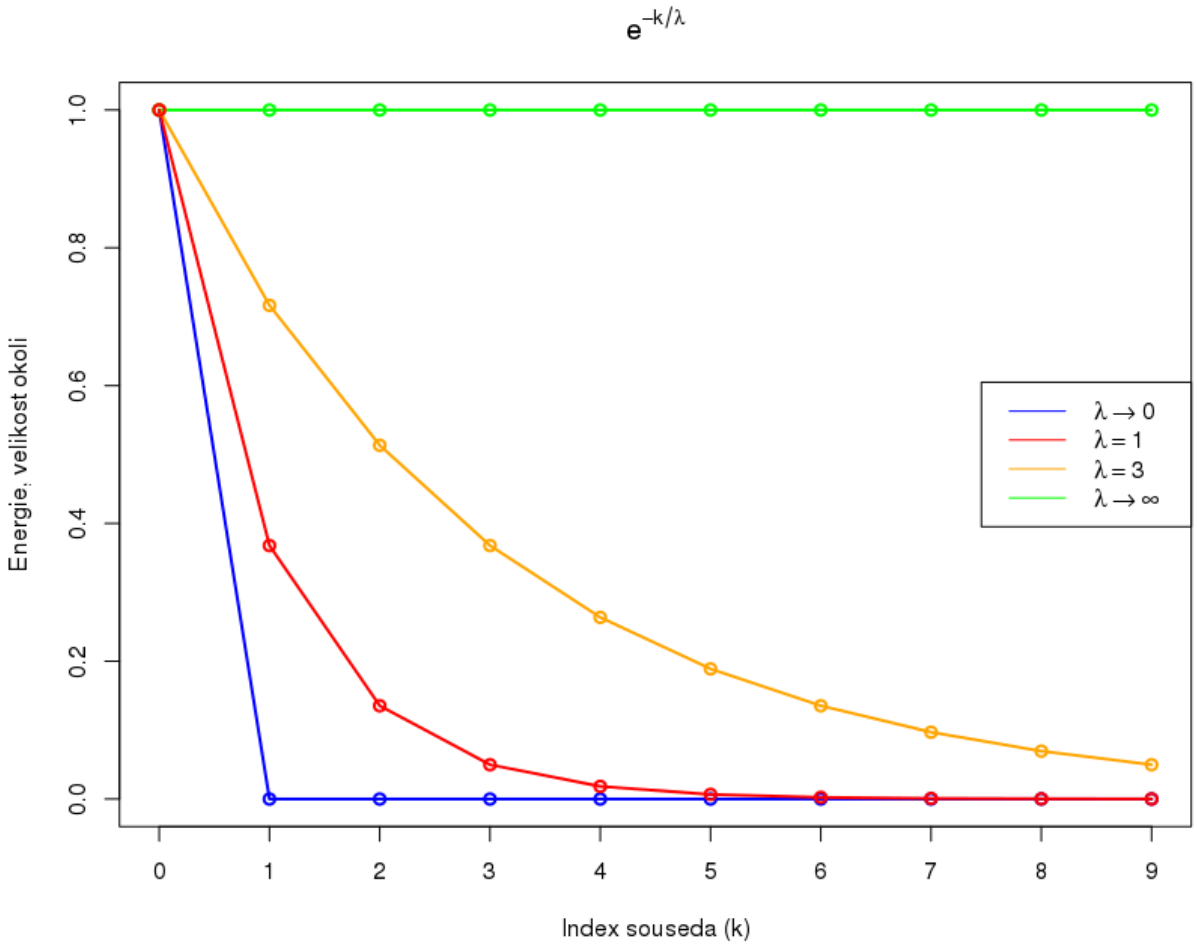
$$w_i^{t+1} = w_i^t + \alpha^t e^{-\frac{k}{\lambda^t}} (x_j - w_i^t)$$

  - α^t je adaptační krok v t-té iteraci a určuje o kolik se maximálně může reprezentant posunout (typicky $\alpha^t \ll 1$, s rostoucím t klesá k 0).
  - k je pořadí vzdálenosti reprezentanta od instance x_j.
  - λ^t definuje velikost okolí a s rostoucím t klesá, pokud položíme $\lambda^t = 2(\sigma^t)^2$ je zřejmé, že okolí má tvar gaussovky.

Neuronový plyn (IV)

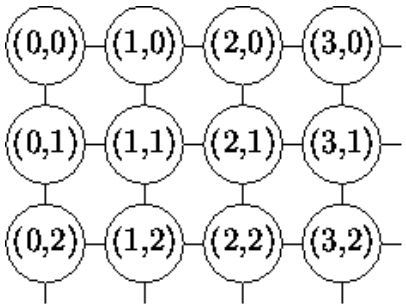
- "Zmenši okolí"
  - okolí λ^t se typicky postupně zmenšuje o nějaký násobek,
  - např. $\lambda^{t+1} = 0.95\,\lambda^t$,
  - s okolím se podobným způsobem zmenšuje i adaptační krok α^t,
  - např. $\alpha^{t+1} = 0.95\,\alpha^t$.
- "Pokud ještě chceš pokračovat . . ."
  - Dopředu určím, že chci pokračovat dokud je $\lambda > 0.05$.
  - Nebo skonči poté, co předložíš všechny instance 10x.
- Kontrolní otázka:
  - Za jakých podmínek se přesune nejbližší reprezentant na pozici právě předložené instance?

Funkce energie

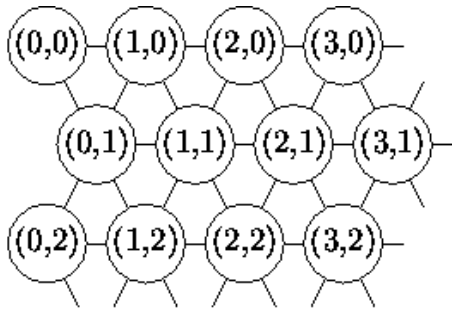


Vylepšení Neuronového plynu

- Jak by se dal neuronový plyn vylepšit dál?
- Co kdyby se neposouvali všechni reprezentanti blízko instance?
- Vytvoříme "přátelské" vztahy mezi reprezentanty. A budou se posouvat jen "kamarádi" vítězného reprezentanta.
- Když vizualizujeme "přátelství" mezi reprezentanty získáme pravidelnou mřížku (síť).
- Typicky se používá čtvercová nebo hexagonální síť.



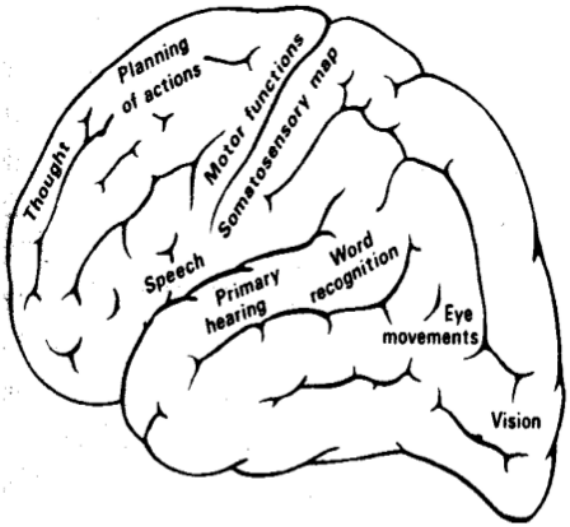
Rectangular

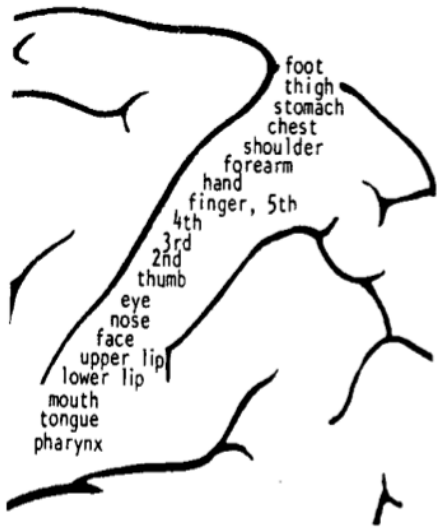


Hexagonal

Inspirace pro SOM

- Inspirací nejsou králové, ale oblasti v lidském mozku.
- Řídící centra jednotlivých končetin spolu souvisí a navzájem se ovlivňují.





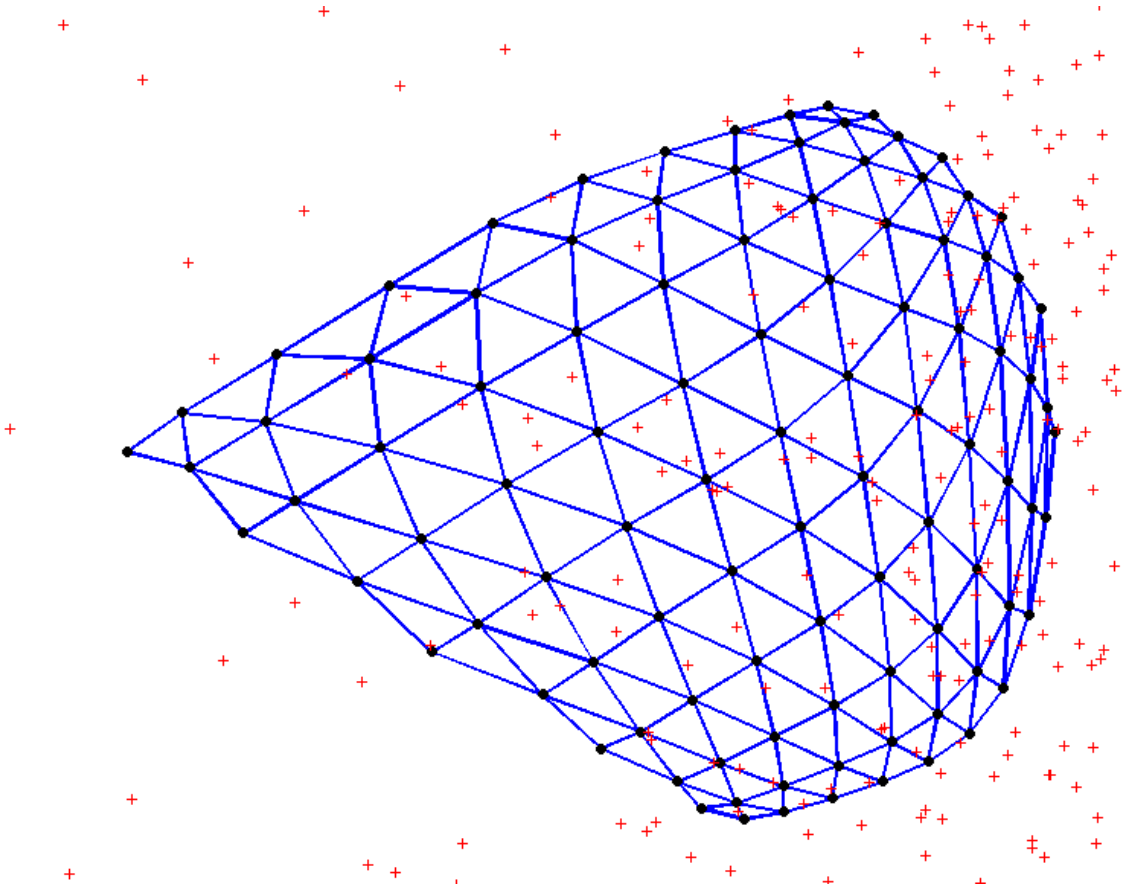
SOM

- Neuronová síť SOM je vynálezem prof. Kohonena z Finska.
- Původně vznikla jako model motorického cortexu a její první aplikace byl fonetický psací stroj.
- A protože se prof. Kohonen zabýval umělými neuronovými sítěmi, převzal i SOM jejich terminologii.

SOM - Pozice neuronů

- Každý reprezentant – v terminologii SOMu **neuron** – je opět reprezentován jeho souřadnicemi v prostoru.
- Souřadnice každého neuronu (reprezentanta) se označují jako váhy.
- Když si zkusím takovou síť vizualizovat, dostaneme například:

SOM – Pozice neuronů (II)



- Co se stalo? Ještě před chvílí byla ta síť přece pravidelná!
- To ano, ale to byla idealizovaná projekce, aby bylo názorně vidět vztahy!

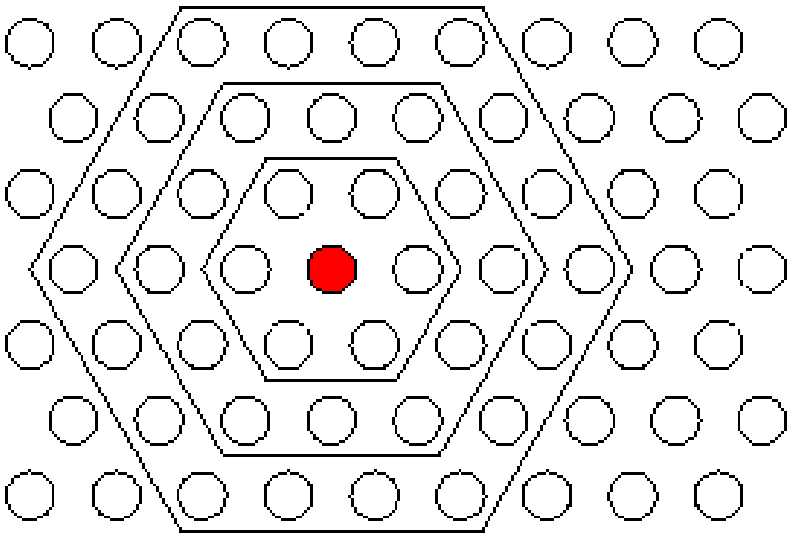
SOM – Algoritmus

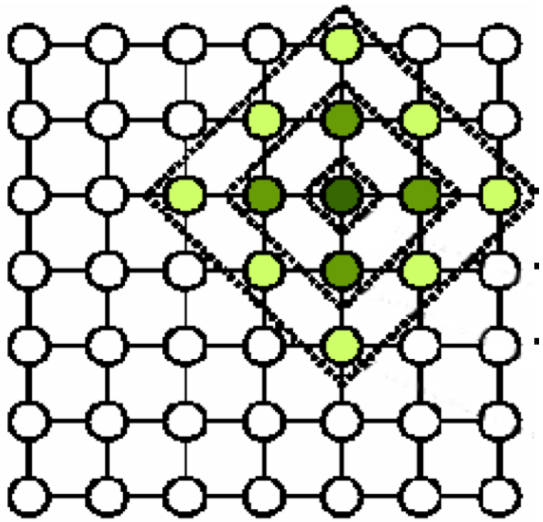
- Celý SOM algoritmus vypadá takto:

1. Inicializuj váhy všech neuronů (souřadnice všech reprezentantů).
2. Vyber *nějakou* vstupní instanci x_j.
3. Spočítej vzdálenost mezi x_j a všemi neurony w_i $\forall i$.
4. Urči nejbližší neuron – BMU (Best Matching Unit).
5. Uprav váhy (pozici) BMU a jeho okolí.
6. Pokud ještě chceš pokračovat, pokačuj bodem 2.

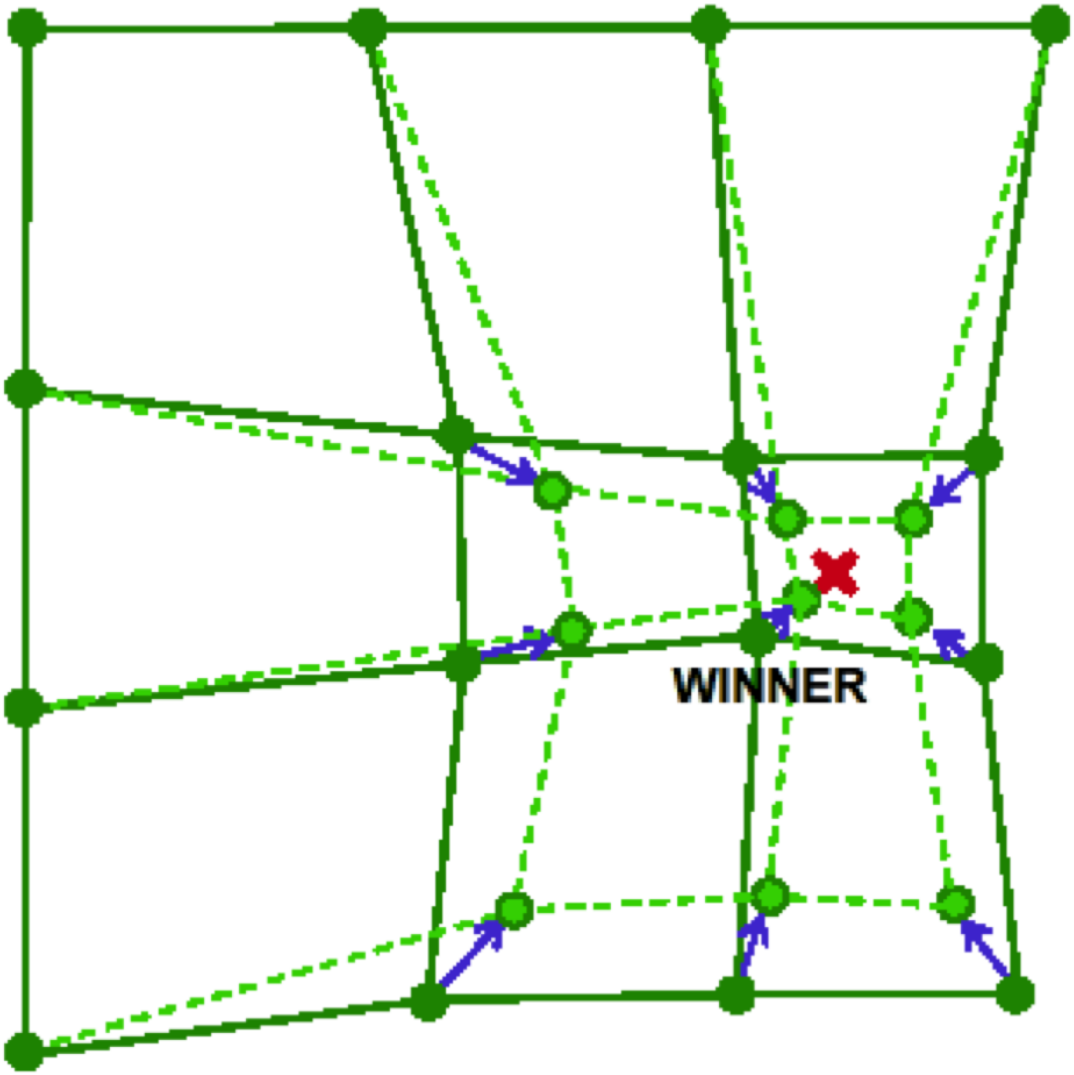
Co se změnilo ve srovnání s neuronovým plynem?

- Určení BMU je celkem jednoduchá záležitost.
  - Zvolím si metriku a aplikuji ji.
- Mnohem zajímavější je úprava pozice BMU a jeho okolí :).
- Jak vlastně určím neurony v okolí?





Změna vah graficky



Změna vah neuronu v SOM

- Novou pozici neuronu w_i v kroku $t+1$ (po předložení vzoru x_j) určím jako:

$$w_i^{t+1} = w_i^t + \alpha^t e^{-\frac{dist(w_i, BMU)}{2(\sigma^t)^2}} (x_j - w_i^t)$$

- ve srovnání s neuronovým plynem došlo k jediné změně
  - index k udávající pořadí neuronu ve vzdálenosti od instance x_j byl nahrazen jeho vzdáleností od BMU v neuronové mřížce $dist(w_i, BMU)$,
- podobně jako u neuronového plynu jdou parametry α^t a σ^t s rostoucím t k 0.

Příklad

Máme 3 neurony $w_1 = (0, 0)$, $w_2 = (2, 1)$ a $w_3 = (0, 3)$. Ty jsou na lince, w_1 je sousedem w_2 a w_2 je sousedem w_3. Vztahy sousednosti jsou symetrické.
Uvažujme instanci $x = (1, 1)$.
Který neuron je BMU? (Použijeme eukleidovskou metriku)

$$d(w_1, x) = \sqrt{(0-1)^2 + (0-1)^2} = \sqrt{2} \doteq 1.41$$

$$d(w_2, x) = \sqrt{(2-1)^2 + (1-1)^2} = \sqrt{1} = 1$$

$$d(w_3, x) = \sqrt{(0-1)^2 + (3-1)^2} = \sqrt{5} \doteq 2.23$$

Příklad

Máme 3 neurony $w_1 = (0, 0)$, $w_2 = (2, 1)$ a $w_3 = (0, 3)$. Ty jsou na lince, w_1 je sousedem w_2 a w_2 je sousedem w_3. Vztahy sousednosti jsou symetrické.
Uvažujme instanci $x = (1, 1)$.
Který neuron je BMU? (Použijeme eukleidovskou metriku)

$$d(w_1, x) = \sqrt{(0-1)^2 + (0-1)^2} = \sqrt{2} \doteq 1.41$$

$$d(w_2, x) = \sqrt{(2-1)^2 + (1-1)^2} = \sqrt{1} = 1$$

$$d(w_3, x) = \sqrt{(0-1)^2 + (3-1)^2} = \sqrt{5} \doteq 2.23$$

Příklad (II)

BMU je tedy w_2. Řekněme, že $\sigma(t) = 1$ a $\alpha(t) = 0.25$
Zkusme vypočítat novou pozici BMU (w_2):

$$\mu(t) = \alpha(t) e^{-\frac{dist(w_i, BMU)}{2\sigma^2(t)}} = 0.25\, e^{-\frac{dist(w_2, w_2)}{2\,1^2}} = 0.25\, e^0 = 0.25$$

$$\begin{aligned} w_2^{t+1} &= w_2^t + \mu(t)(x - w_2^t) = (2,1) + 0.25\,((1,1) - (2,1)) = \\ &= (2,1) + 0.25(-1,0) = (1.75, 1) \end{aligned}$$

w_1 se posune do pozice:

$$\mu(t) = \alpha(t) e^{-\frac{dist(w_1, w_2)}{2\sigma^2(t)}} = 0.25\, e^{-\frac{1}{2}} = 0.25 \times 0.606 = 0.151$$

$$\begin{aligned} w_1^{t+1} &= w_1^t + \mu(t)(x - w_1^t) = (0,0) + 0.151\,((1,1) - (0,0)) = \\ &= (0,0) + (0.151, 0.151) = (0.151, 0.151) \end{aligned}$$

Chyba SOM

- Stejně jako v Hierarchickém shlukování a KMeans potřebujeme nějakou míru "dobrého shluknutí".
- Kvantizační chyba
  - Tu už známe ze začátku přednášky!
  - Průměrná vzdálenost mezi instancemi a nejbližšími neurony.
- Topografická chyba
  - Popisuje kvalitu "natažení" mřížky sítě na vstupní data.
  - Procento instancí, pro které platí, že jejich BMU a druhý nebližší neuron nejsou sousedy v mřížce sítě.

$$err_{topo} = \frac{1}{n} \sum_{i=1}^{n} u(x_i)$$

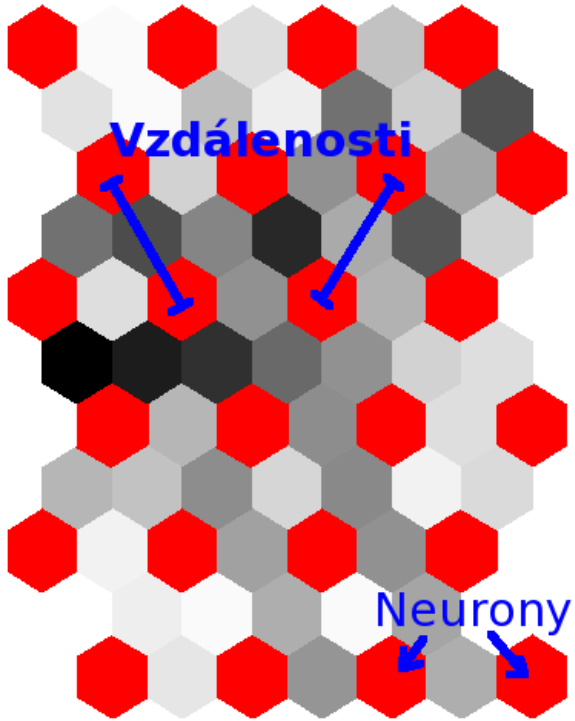
  - $u(x_i) = 1 \iff$ BMU a druhý nejbližší neuron pro x_i v mřížce nesousedí.

Vizualizace SOM

- Dokud máme jen 2D data, tak s vizualizací není problém.
- Co když máme více dimenzí?
  - U-Matice
  - Analýza hlavních komponent
  - Sammonova projekce

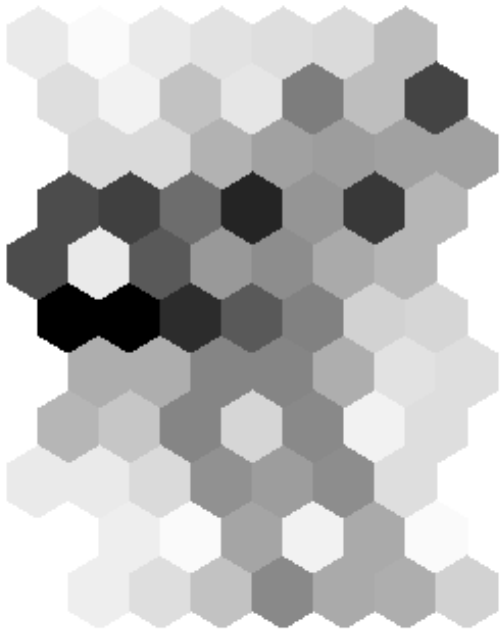
U-Matice

- Matice vzdáleností mezi váhovými vektory jednotlivých neuronů, typicky se vizualizuje, vzdálenosti vyjádřeny barvou – světlá barva = malá vzdálenost.
- Zobrazuje strukturu vzdáleností v prostoru dat.



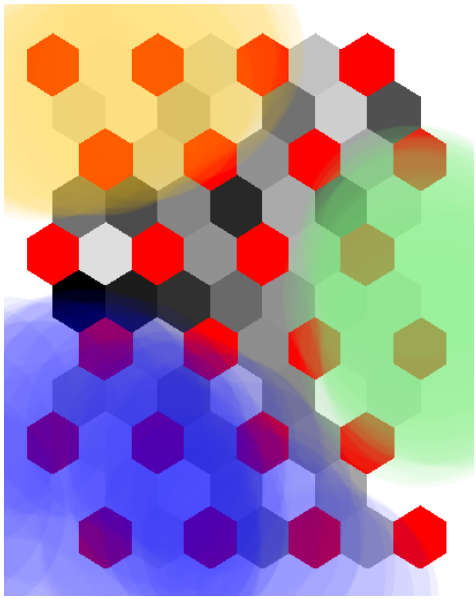
U-Matice (II)

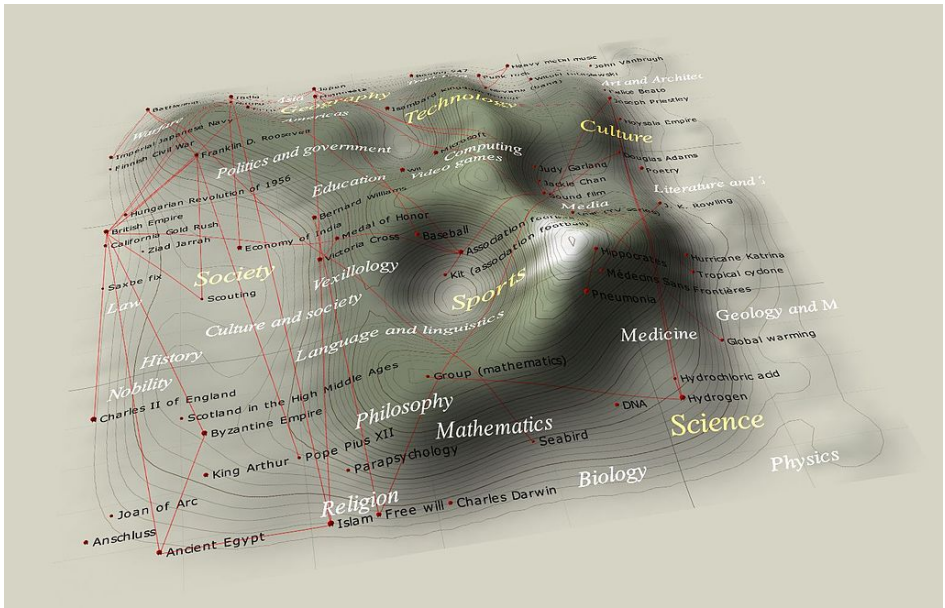
- Barva neuronu je vzdálenost jeho váhového vektoru od váhových vektorů okolních neuronů
  - Tmavé váhové vektory jsou vzdáleny od ostatních datových vektorů ve vstupním prostoru.
  - Světlé váhové vektory jsou obklopeny cizími vektory ve vstupním prostoru.



U-Matice (III)

- Jak z U-Matice poznám shluky?
- Ze vzdáleností mezi neurony.
- Kopce oddělují clustery (údolí).





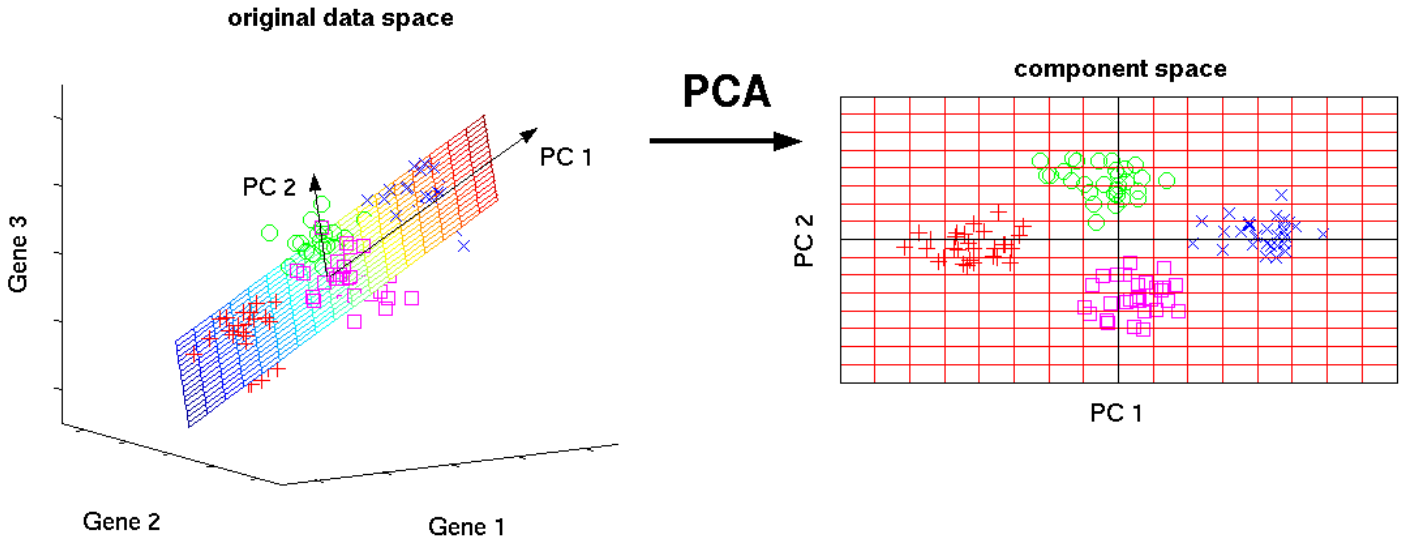
Wikipedia – Self-organizing map

Analýza hlavních komponent

- Jde o statistickou metodu pro redukci dimenzionality.
- Označuje se jako PCA z anglického Principal Component Analysis.
- Snaží se najít nové osy, které lépe popisují data s minimální ztrátou informace.
- První osa vede směrem, který má největší rozptyl hodnot, druhá osa směrem, kde je druhý největší rozptyl, atd.
- Osy jsou ortogonální, tedy vzájemně pravoúhlé.
- Vždy mi vrátí stejný počet nových os, jako mají původní data dimenzí, ale já se mohu rozhodnout některé nepoužít.

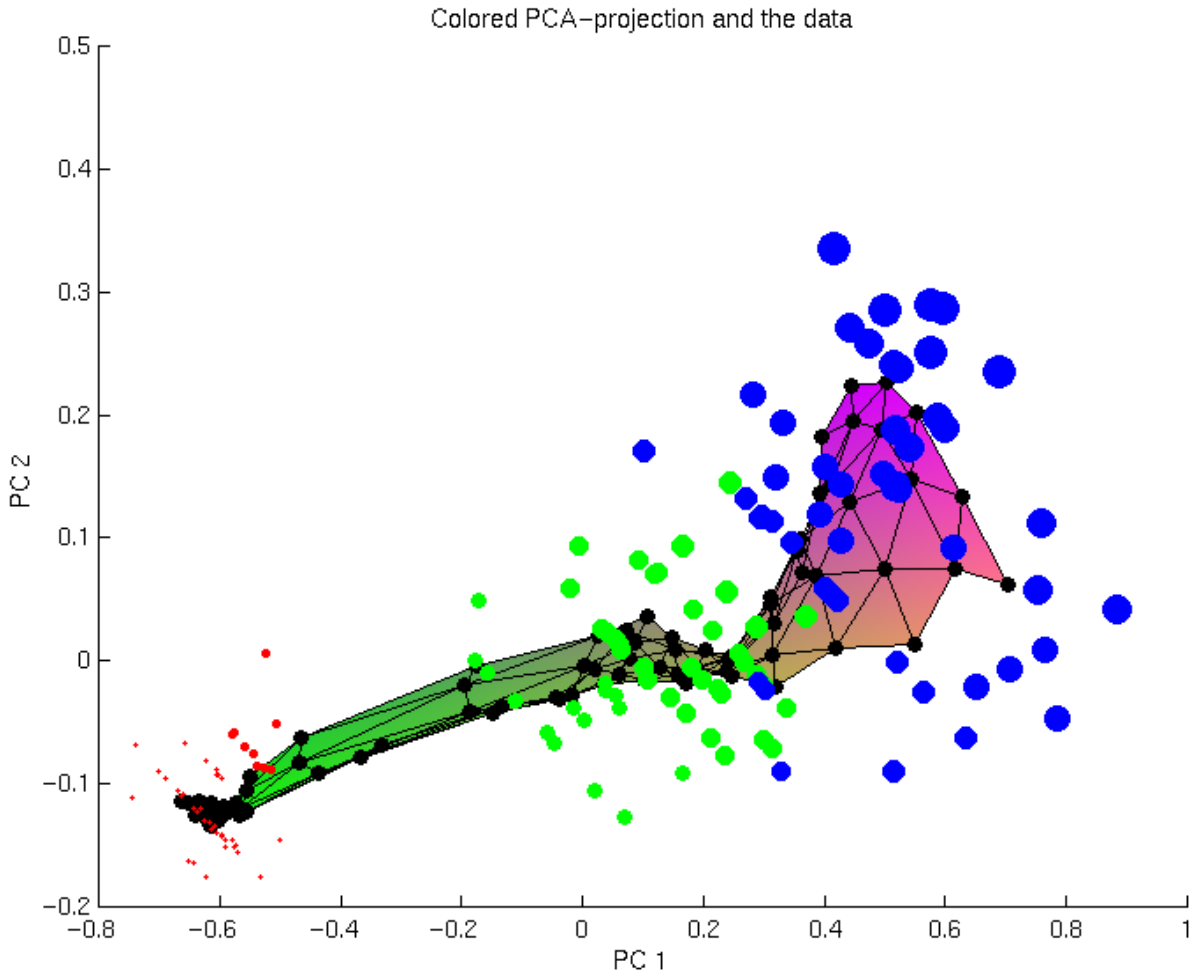
Analýza hlavních komponent (II)

- Výpočet nových souřadnic pomocí
  - kovariance, vlastních čísel a vlastních vektorů.
- `http://www.cs.otago.ac.nz/cosc453/student_tutorials/principal_components.pdf`



Využití PCA v SOM

- Nyní můžu provést PCA projekci SOM sítě do 2D a zobrazit si ji.



Využití PCA mimo SOM

- PCA není limitována jen na použití v SOM, ale můžu ji použít například pro průzkum dat.
- Stejně tak, některé metody vytěžování dat nemají rády příliš mnoho dimenzí a PCA jim můžete pomoci k lepším výsledkům.
- Nevýhodou je umělost nových os, která znesnadňuje interpretaci získaných výsledků:
  - $0.125\, petal_length + 0.578\, petal_width + 0.934\, sepal_length - 0.346\, sepal_width$.

Sammonova projekce

- Jinou možností je Sammonova projekce.
- Ta netransformuje osy, ale znovu umísťuje objekty v novém (méně dimenzionálním) prostoru.
- Při umisťování se snaží zachovat vztahy v datech (data, která byla blízko v původním prostoru, budou blízko i v novém prostoru).

Sammonova projekce (2)

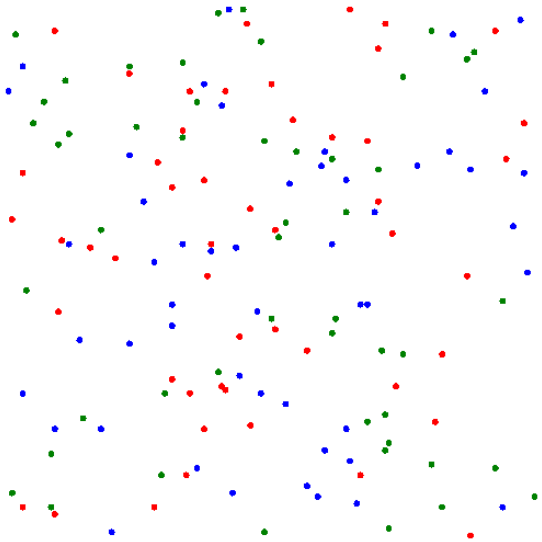
- Sammonova projekce minimalizuje stresovou funkci:

$$E = \frac{1}{\sum_{i<j} dist^*(x_i, x_j)} \sum_{i<j} \frac{(dist^*(x_i, x_j) - dist(x_i, x_j))^2}{dist^*(x_i, x_j)}$$

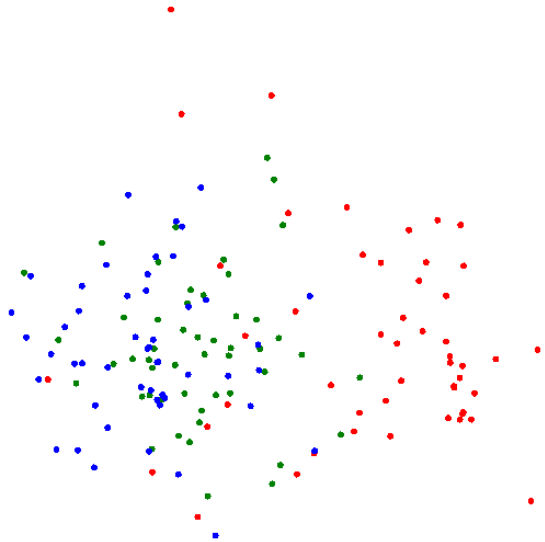
- $dist^*(x_i, x_j)$ je vzdálenost x_i a x_j v původním prostoru.
- $dist(x_i, x_j)$ je vzdálenost x_i a x_j v novém prostoru (v projekci).
- Pro minimalizaci se používají standardní optimalizační metody
  - pro tuto úlohu typicky iterační metody.
- Při minimalizaci se pohybuje body v novém prostoru (v projekci). Tím ovlivníte $dist(x_i, x_j)$ a můžete dosáhnout zmenšení E.

Sammonova projekce - ukázka

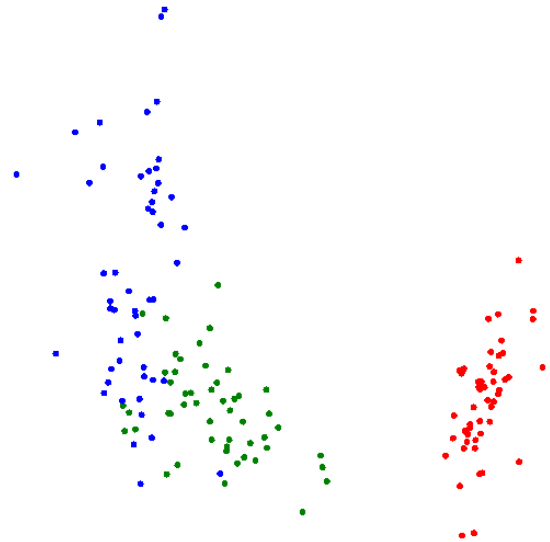
Ukázka několika iterací Sammonovy projekce na Iris datech.



Počáteční stav



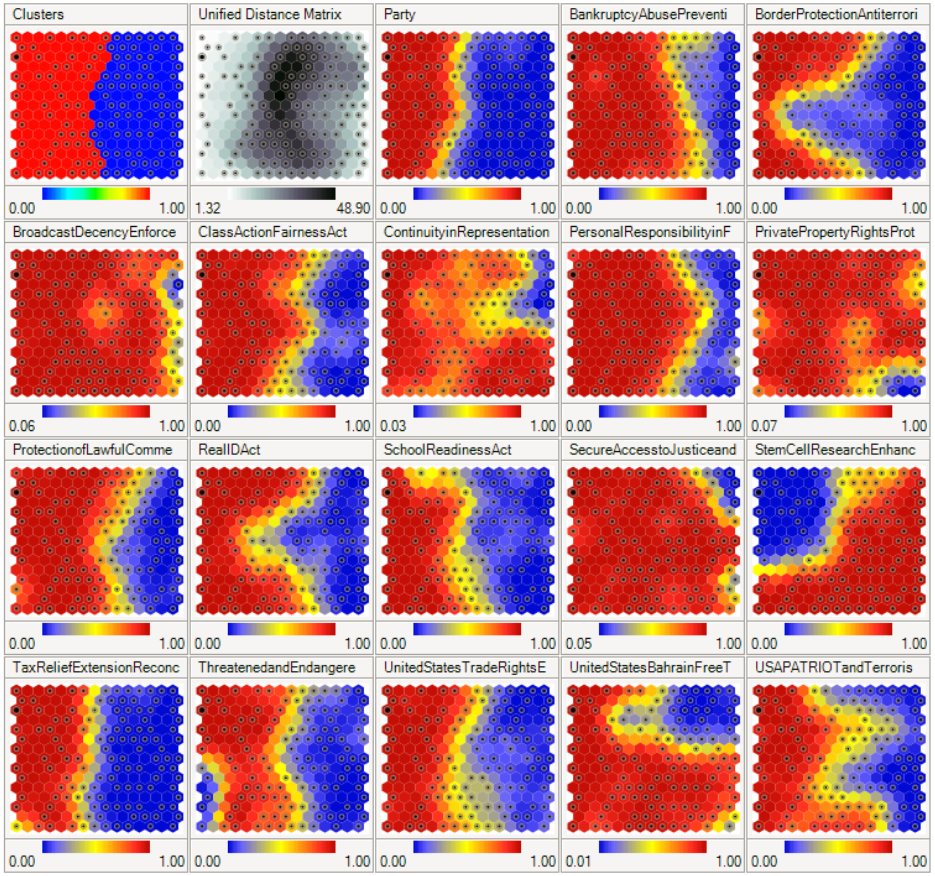
1. iterace



10. iterace

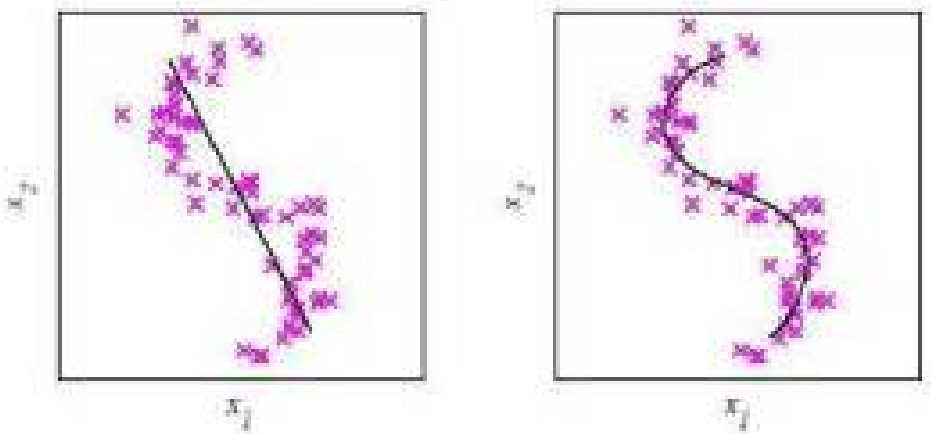
Další vizualizace – Příznakové grafy

- Vychází z U-Matice, ale místo vzdálenosti jednotlivých vektorů se do šestiúhelníčků kreslí hodnoty vybrané proměnné.



Shrnutí, souvislosti

- Samoorganizující se mapy
  - více než shlukovací algoritmus = třídění objektů,
  - vizualizace – porozumění struktuře dat,
  - redukce dimenze – mapováním objektů na souřadnice mřížky.
- SOM vs PCA
  - SOM je nelineárním zobecněním PCA.



Shrnutí, souvislosti (II)

- SOM vs KMeans
  - pro malý počet neuronů/středů velmi podobné chování,
  - oba přístupy kvantizují data, aproximují hustotu vstupních dat,
  - při více neuronech odlišnosti plynoucí z okolí a mřížky neuronů,
  - K by mělo odpovídat skutečnému počtu shluků,
  - počet neuronů je vyšší, shluky zřejmé mj. z U-Matice.

Další čtení

- http://www.cis.hut.fi/somtoolbox/theory/somalgorithm.shtml
- http://www.cis.hut.fi/research/som-research/
- http://www.ai-junkie.com/ann/som/som1.html
- www.cs.bham.ac.uk/~jxb/NN/l16.pdf