BI-VZD přednáška 4

Alexander Kovalenko

FIT ČVUT

14. 03. 2022

Autoři: Karel Klouda, Juan Pablo Maldonado Lopez, Daniel Vašata.
Problémy, návrhy apod. hlaste v GitLabu.
Verze souboru: 14. března 2022 09:24.

Co bude v dnešní přednášce

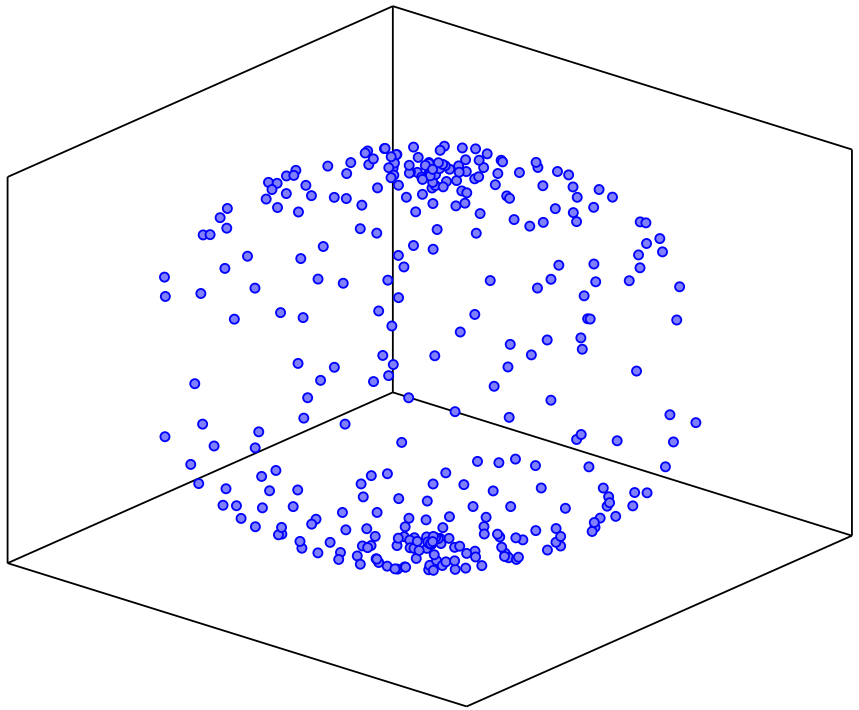
- Principy nesupervizovaného učení.
- Úvod do shlukové analýzy (clusterové analýzy).
- Hierarchické shlukování.
- Nehierarchické shlukování – algoritmus k-means.

Nesupervizované učení

- Nastává v situaci, kdy data nemáme nikterak označena. Tj. nemáme žádnou veličinu, kterou bychom u trénovacích dat znali a snažili se ji naučit predikovat.
- Cílem nesupervizovaného učení je porozumět struktuře dat pouze na základě jich samotných. To znamená bez nějakého vnějšího vodítka.
- Proto se nesupervizovanému učení také říká **učení bez učitele**.
- Porozuměním zde typicky myslíme nalezení co „nejmenších“ oblastí v prostoru příznaků, kde se data vyskytují nejčastěji.
- Obvykle totiž platí, že se naměřená skutečná data nevyskytují v celém prostoru stejně pravděpodobně, ale bývají více či méně lokalizována - tvoří nějaké shluky, vyskytují se v méně-dimenzionálních oblastech atd.
- Porozumění této lokalizaci přináší důležitou informaci o vnitřní struktuře dat!

Nesupervizované učení

Uvažujme data rozložená v třírozměrném prostoru podle následujícího obrázku.



Podaří-li se nám zjistit, že jsou všechny tyto body na kouli, můžeme jejich polohu popsat pomocí sférických souřadnic a získat tak ekvivalentní reprezentaci pomocí dvou spojitých příznaků (tzv. redukce dimenzionality).

Navíc je možné detekovat, že hustota bodů na pólech je vyšší než hustota bodů na rovníku.

Obecným problémem nesupervizovaného učení je, že vůbec není jasné, jak bychom měli vyhodnocovat úspěšnost získaného porozumění.

To je velký rozdíl oproti supervizovanému učení, kde je možné kvalitu naučeného modelu vyhodnocovat mnoha víceméně rovnocennými způsoby (např. přesností u klasifikace).

Nesupervizované učení z pohledu teorie pravděpodobnosti

Uvažujme situaci, kdy naše data obsahují p příznaků a označme $\mathcal{X}$ prostor ve kterém se nacházejí možné výsledky[1].

- Pro p binárních příznaků volíme $\mathcal{X} = \{0, 1\}^p$.
- Pro p spojitých příznaků typicky volíme $\mathcal{X} = \mathbb{R}^p$.

Z pohledu teorie pravděpodobnosti a statistiky (viz BI-PST) chápeme pozorovaná data jako **realizace náhodného vektoru** $\boldsymbol{X} = (X_1, \ldots, X_p)^T$.

Porozumění vnitřní struktuře pak znamená porozumění rozdělení $\boldsymbol{X}$. Chceme tedy získat **odhad pravděpodobnosti** $\mathrm{P}(\boldsymbol{X} \in O)$ pro každou (rozumnou) podmnožinu $O \subset \mathcal{X}$.

- V případě spojitých příznaků to ekvivalentně znamená odhadnout sdruženou **hustotu pravděpodobnosti** $f_{\boldsymbol{X}}(\boldsymbol{x}) \equiv f_{\boldsymbol{X}}(x_1, \ldots, x_p)$.
- V případě diskrétních příznaků to ekvivalentně znamená odhadnout sdruženou **pravděpodobnostní funkci** $p_{\boldsymbol{X}}(\boldsymbol{x}) \equiv \mathrm{P}(X_1 = x_1, \ldots, X_p = x_p)$.

Soustředíme se přitom zejména na nalezení oblastí v prostoru $\mathcal{X}$, které mají velkou pravděpodobnost a jsou při tom „co nejmenší“.

[1] Volba $\mathcal{X}$ není jednoznačná a je součástí volby modelu.

Úvod do shlukové analýzy

Jednou ze základních metod nesupervizovaného učení je **shlukování** nazývané také clusterování (angl. **clustering**).

Naším cílem je roztřídit data do skupin, které budeme nazývat **shluky**, tak, že platí dva přirozené požadavky:

- blízké body budou ve stejném shluku a
- vzdálené body budou v různých shlucích.

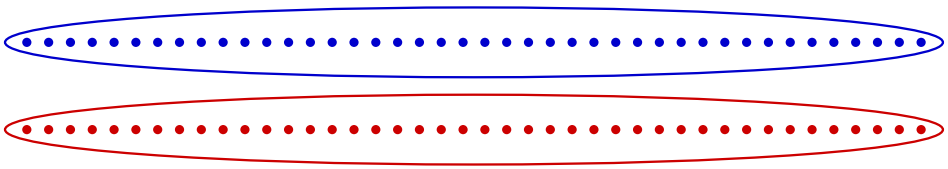
Tento popis je ovšem hodně vágní a není vůbec jasné, jak ho zformalizovat.

Jak uvidíme na následujícím slajdu, jedním z problémů je fakt, že tyto intuitivní požadavky obecně nejsou kompatibilní.

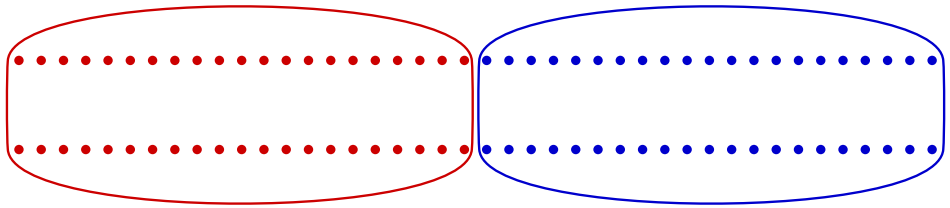
Dalším problémem je již dříve zmiňovaná neexistence hodnotícího kritéria kvality nalezeného shlukování.

V důsledku absence jednoznačnosti tudíž existuje více způsobů formalizace a následně také mnoho různých shlukovacích algoritmů, které mohou na stejných datech dávat velmi rozdílné výsledky.

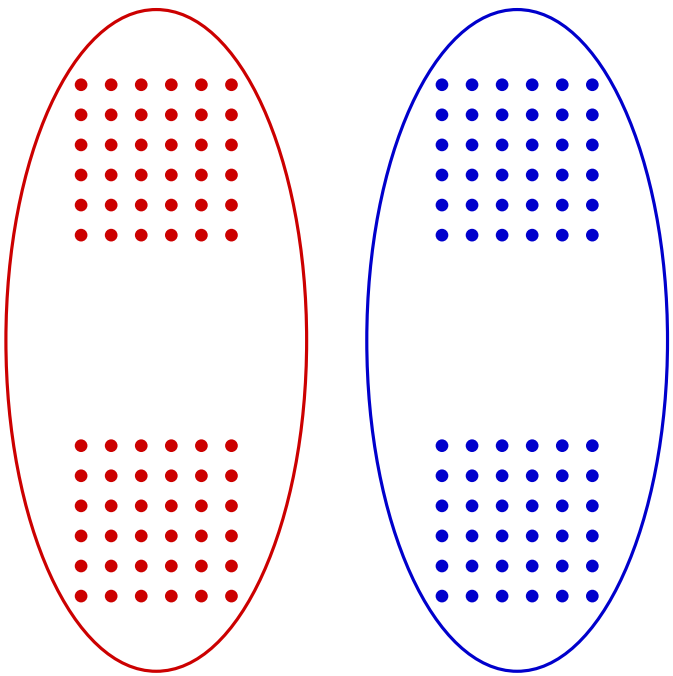
Shluková analýza - znázornění nejasných situací

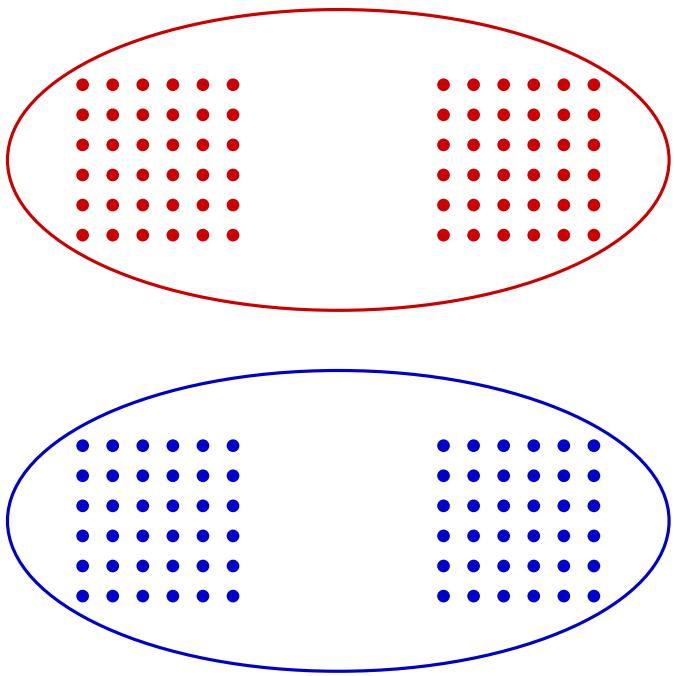


Shlukování, které se snaží udržet blízké body ve stejných shlucích.



Shlukování, které se snaží nemít v jednotlivých shlucích příliš vzdálené body.





Obě shlukování jsou stejně legitimní a není jasné, které vybrat.

Vzdálenost

Klíčovým pojmem, na kterém stojí shlukování, je vzdálenost.

Definice

Vzdálenost nebo také **metrika** na množině $\mathcal{X}$ je funkce $d : \mathcal{X} \times \mathcal{X} \to [0, +\infty)$ taková, že pro každé $x, y, z \in \mathcal{X}$ platí

i) $d(x, y) \geq 0$, a $d(x, y) = 0$ právě tehdy když $x = y$ - pozitivní definitnost,

ii) $d(x, y) = d(y, x)$ - symetrie,

iii) $d(x, y) \leq d(x, z) + d(z, y)$ - trojúhelníková nerovnost.

Dvojice $(\mathcal{X}, d)$ se potom nazývá **metrický prostor**.

Poznámky k vlastnostem vzdálenosti:

i) Vzdálenost různých bodů je vždy kladná. Nulová je pouze pro stejné body.

ii) Vzdálenost bodu x od bodu y je stejná jako vzdálenost y od x.

iii) Přímá vzdálenost mezi dvěma body je vždy menší nebo rovna vzdálenosti přes nějaký třetí bod.

Vzdálenost - příklady

Často používané vzdálenosti na $\mathbb{R}^p$:

- **Eukleidovská vzdálenost** nebo také L_2 vzdálenost,

$$d_2(\boldsymbol{x}, \boldsymbol{y}) = \sqrt{\sum_{i=1}^{p} (x_i - y_i)^2}.$$

- **Manhattanská vzdálenost** nebo také L_1 vzdálenost,

$$d_1(\boldsymbol{x}, \boldsymbol{y}) = \sum_{i=1}^{p} |x_i - y_i|.$$

- **Čebyševova vzdálenost** nebo také L_∞ vzdálenost,

$$d_\infty(\boldsymbol{x}, \boldsymbol{y}) = \max_i |x_i - y_i|.$$

Příkladem vzdálenosti na množině řetězců je pak **Levenštejnova vzdálenost** (angl. **Levenshtein distance**) definovaná jako minimální počet jednoznakových operací (vkládání, mazání, nahrazení), které převedou jeden řetězec na druhý.

Vstupy a výstupy shlukování

Než se zaměříme na dva konkrétní případy shlukovacích algoritmů, ujasněme si, co je vlastně vstupem a co výstupem shlukování.

Vstupy

- Metrický prostor $\mathcal{X}$ se vzdáleností d.
- Množina dat $\mathcal{D} \subset \mathcal{X}$.
- Obvykle také požadovaný počet shluků k.

Výstupy

- Rozklad množiny dat na jednotlivé shluky. To jest $C = (C_1, \ldots, C_k)$, kde $C_i \subset \mathcal{D}$ pro každé i a $C_i \cap C_j = \emptyset$ pro každé $i \neq j$, přičemž

$$\mathcal{D} = \bigcup_{i=1}^{k} C_i.$$

- Bod $x \in \mathcal{D}$ je tedy v i-tém shluku, jestliže $x \in C_i$.
- U hierarchického shlukování může být finálním výstupem dendrogram jakožto grafické znázornění hierarchické struktury shlukování, které podrobně představíme za chvíli.

Hierarchické shlukování

Začněme zcela přirozeným hladovým aglomerativním přístupem. Označme N počet prvků množiny dat $\mathcal{D}$.

- Na začátku uvažujme každý bod jako jeden shluk. Tj. máme právě N shluků.
- Nyní budeme opakovat následující kroky:
  - Najdeme dva shluky, které jsou k sobě nejblíže.
  - Tyto dva shluky spojíme do nového shluku.

Po $N - 1$ opakováních tedy skončíme s jediný velkým shlukem, ve kterém jsou všechny body.

- K tomu, abychom mohli tento postup provést, je třeba stanovit způsob měření vzdálenosti dvou shluků.
- Má-li být navíc výstupem shlukování, musíme určit nějaké zastavovací kritérium.
- To bývá nejčastěji počet shluků k, případně limitní hodnota vzdálenosti shluků, nad kterou už nebudeme shluky spojovat.[2]

[2] Vzdálenost spojovaných shluků v každém kroku roste (resp. neklesá).

Měření vzdálenosti shluků

Vstupem do shlukování je vzdálenost dvojice bodů $d(x, y)$. K měření vzdálenosti dvojic shluků $D(A, B)$ se obvykle používá jeden z následujících způsobů:

Metoda nejbližšího souseda (angl. single linkage) - generuje dlouhé řetězce

$$D(A, B) = \min_{x \in A, y \in B} d(x, y).$$

Metoda nejvzdálenějšího souseda (angl. complete linkage) - generuje kompaktní shluky

$$D(A, B) = \max_{x \in A, y \in B} d(x, y).$$

Párová vzdálenost (angl. average linkage) - kompromis předchozích dvou

$$D(A, B) = \frac{1}{|A||B|} \sum_{x \in A, y \in B} d(x, y).$$

Wardova metoda - v $\mathbb{R}^p$ - velmi účinná, minimalizuje nárust vnitřního rozptylu

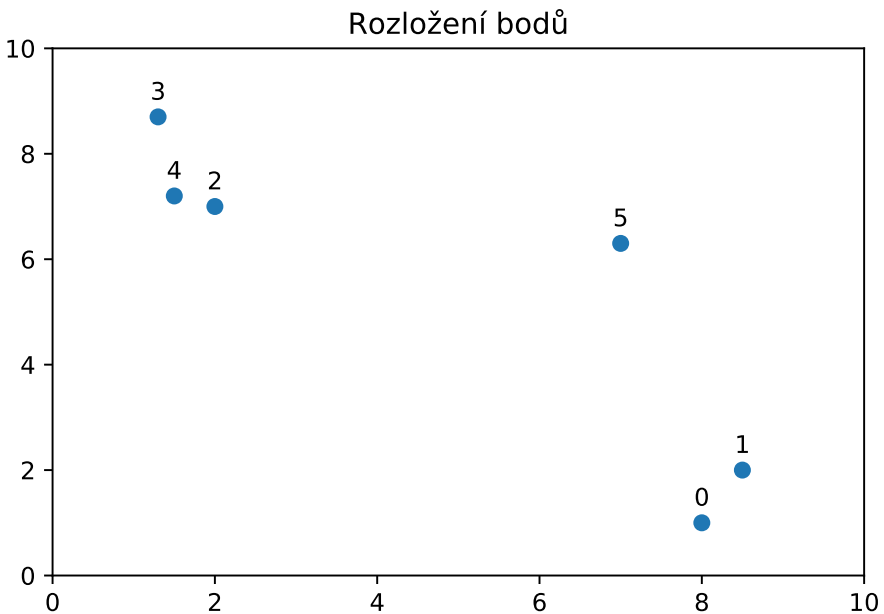
$$D(A, B) = \sum_{\boldsymbol{x} \in A \cup B} \|\boldsymbol{x} - \bar{\boldsymbol{x}}_{A \cup B}\|^2 - \sum_{\boldsymbol{x} \in A} \|\boldsymbol{x} - \bar{x}_A\|^2 - \sum_{\boldsymbol{x} \in B} \|\boldsymbol{x} - \bar{x}_B\|^2,$$

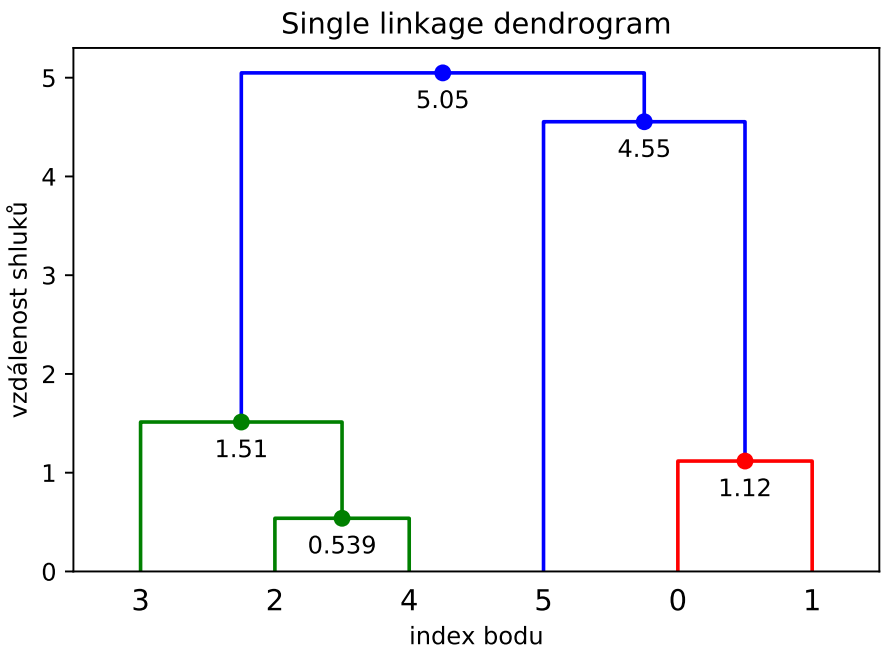
kde $\bar{\boldsymbol{x}}_A = \frac{1}{|A|} \sum_{\boldsymbol{x} \in A} \boldsymbol{x}$ je geometrický střed množiny A a $\bar{\boldsymbol{x}}_B$, $\bar{\boldsymbol{x}}_{A \cup B}$ analogicky.

Vizualizace hierarchického shlukování pomocí dendrogramu

Celý proces aglomerativního shlukování můžeme reprezentovat a graficky znázornit pomocí **dendrogramu**.

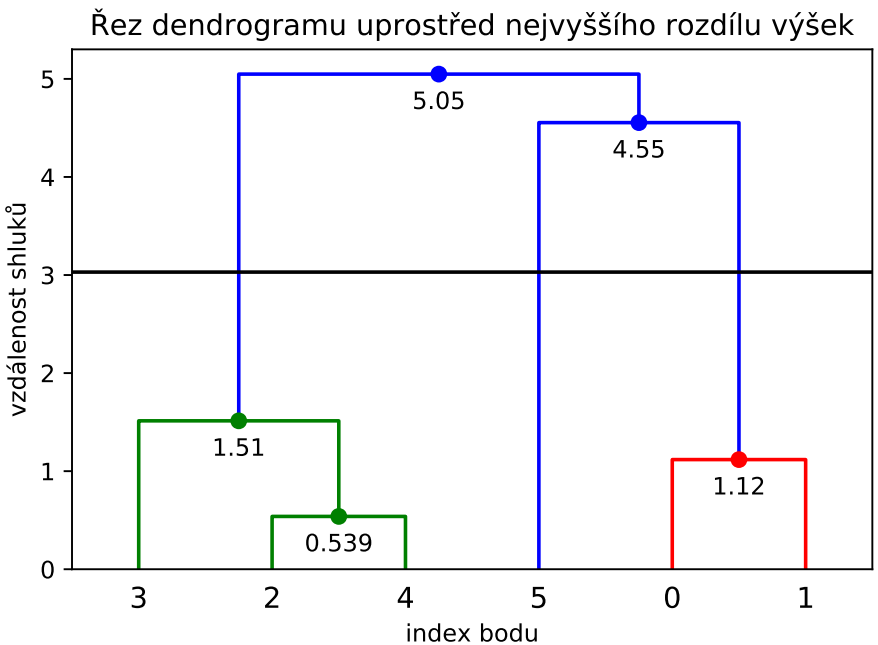
- Jedná se o strom, jehož vrcholy představují shluky, které při běhu vznikly.
- V listech jsou počáteční jednoprvkové shluky a kořen reprezentuje finální shluk všech bodů.
- Strom je navíc nakreslený tak, že jsou všechny listy ve výšce 0 a výška ostatních vrcholů odpovídá vzdálenosti podřazených shluků, které se v tomto vrcholu spojily.

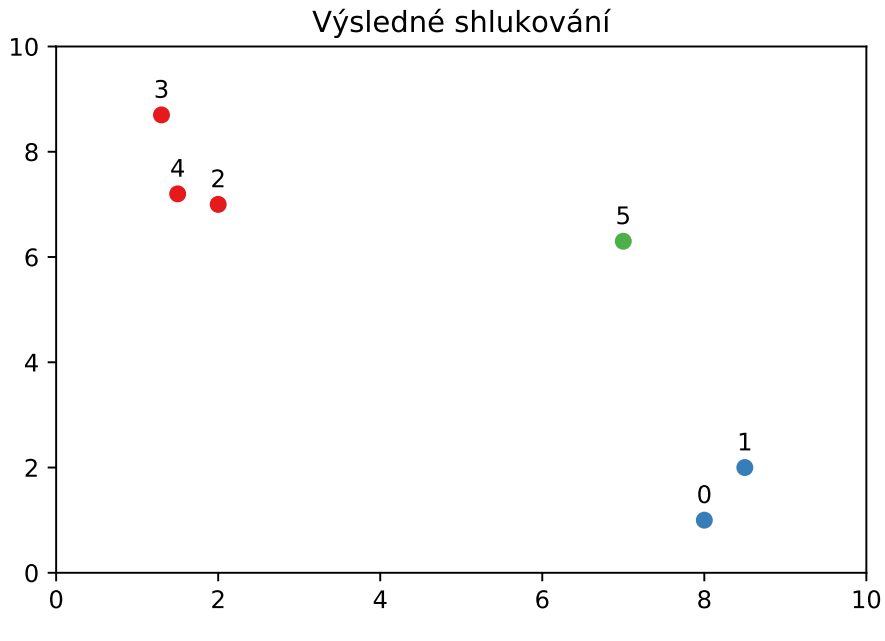




Jak získáme shlukování z dendrogramu

- Známe-li požadovaný počet k shluků, „rozřízneme“ dendrogram mezi k-tým a $(k-1)$-tým nejvyšším vrcholem. Hrany procházející říznutím pak odpovídají výsledným shlukům.
- Nebo můžeme stanovit limitní hranici vzdálenosti shluků a rozříznout dendrogram v dané výšce.
- Případně můžeme určovat místo rozříznutí a tím potažmo i počet shluků pomocí rozdílů výšek sousedních vrcholů.





Finální poznámky k hierarchickému shlukování

- Ukázali jsme si aglomerativní hierarchické shlukování.
- Existuje také divizní hierarchické shlukování při kterém naopak vycházíme z jednoho shluku a opakovaně provádíme dělení.
- Výhodou hierarchického shlukování je možnost získat ucelený pohled pomocí dendrogramu a teprve na jeho základě vybírat vhodný počet shluků.
- Další výhodou je hierarchická struktura. Při zvýšení počtu shluků o jedničku dojde pouze k rozdělení některého ze stávajících, přičemž ostatní se nemění.
- Kvalita vzniklého dendrogramu z pohledu zachování párových vzdáleností originálních dat může například být měřena pomocí cophenetic correlation.
- Hierarchické shlukování se příliš nehodí pro velké datové soubory, protože je výpočetně náročné.
  - V jednom kroku aglomerativního algoritmu musíme provést $n(n-1)/2$ porovnání, kde n je aktuální počet shluků.
  - V jednom kroku divizního algoritmu musíme provést 2^{n-1} porovnání, kde n je velikost děleného shluku.

Formulace shlukování jako optimalizační úlohy

- Oblíbeným přístupem ke shlukování je formulace ve tvaru optimalizační úlohy.
- Při tomto přístupu je třeba definovat **účelovou funkci** (angl. **objective function**), která nějakým způsobem ohodnocuje daný rozklad množiny na jednotlivé shluky.
- Cílem je pak nalézt takový rozklad, který účelovou funkci minimalizuje.
- Nyní se budeme zabývat situací, kdy pro dané k hledáme takový rozklad $C = (C_1, \ldots, C_k)$ na prostoru $\mathcal{X} = \mathbb{R}^p$ vybaveném Eukleidovskou vzdáleností, který minimalizuje účelovou funkci

$$G(C) = \sum_{i=1}^{k} \frac{1}{2|C_i|} \sum_{\boldsymbol{x},\boldsymbol{y} \in C_i} \|\boldsymbol{x} - \boldsymbol{y}\|^2.$$

- V účelové funkci se pro každý shluk sečtou průměrné kvadráty vzdáleností všech bodů daného shluku od jeho ostatních bodů.
- Ukažme si, že tuto účelovou funkci můžeme elegantně vyjádřit pomocí součtů kvadrátů vzdáleností od geometrických středů (těžišť) příslušných shluků.

Souvislost účelové funkce s geometrickým středem

Tvrzení

Pro konečnou množinu bodů $A \subset \mathbb{R}^p$ platí

$$\frac{1}{2|A|} \sum_{\boldsymbol{x},\boldsymbol{y} \in A} \|\boldsymbol{x}-\boldsymbol{y}\|^2 = \sum_{\boldsymbol{x} \in A} \|\boldsymbol{x}-\bar{\boldsymbol{x}}\|^2 = \min_{\boldsymbol{\mu} \in \mathbb{R}^p} \sum_{\boldsymbol{x} \in A} \|\boldsymbol{x}-\boldsymbol{\mu}\|^2,$$

*kde $\bar{\boldsymbol{x}} = \frac{1}{|A|} \sum_{\boldsymbol{x} \in A} \boldsymbol{x}$ je **geometrický střed** (angl. **centroid**) množiny A.*

Důkaz.

Pro každé $\boldsymbol{a}, \boldsymbol{b} \in \mathbb{R}^p$ platí $\|\boldsymbol{a}-\boldsymbol{b}\|^2 = (\boldsymbol{a}-\boldsymbol{b})^T(\boldsymbol{a}-\boldsymbol{b}) = \|\boldsymbol{a}\|^2 - 2\boldsymbol{a}^T\boldsymbol{b} + \|\boldsymbol{b}\|^2$, protože $\boldsymbol{b}^T\boldsymbol{a} = \boldsymbol{a}^T\boldsymbol{b}$. Pro $\boldsymbol{a} = \boldsymbol{x}-\boldsymbol{\mu}$ a $\boldsymbol{b} = \boldsymbol{y}-\boldsymbol{\mu}$ z toho plyne

$$\frac{1}{2|A|} \sum_{\boldsymbol{x},\boldsymbol{y} \in A} \|\boldsymbol{x}-\boldsymbol{y}\|^2 = \frac{1}{2|A|} \sum_{\boldsymbol{x},\boldsymbol{y} \in A} \|\boldsymbol{x}-\boldsymbol{\mu}\|^2 + \frac{1}{2|A|} \sum_{\boldsymbol{x},\boldsymbol{y} \in A} \|\boldsymbol{y}-\boldsymbol{\mu}\|^2 - \frac{1}{|A|} \sum_{\boldsymbol{x},\boldsymbol{y} \in A} (\boldsymbol{x}-\boldsymbol{\mu})^T(\boldsymbol{y}-\boldsymbol{\mu}).$$

Poslední člen upravíme:

$$\frac{1}{|A|} \sum_{\boldsymbol{x},\boldsymbol{y} \in A} (\boldsymbol{x}-\boldsymbol{\mu})^T(\boldsymbol{y}-\boldsymbol{\mu}) = \frac{1}{|A|} \sum_{\boldsymbol{x} \in A} (\boldsymbol{x}-\boldsymbol{\mu})^T \sum_{\boldsymbol{y} \in A} (\boldsymbol{y}-\boldsymbol{\mu}) = \frac{1}{|A|} \Big\| \sum_{\boldsymbol{x} \in A} (\boldsymbol{x}-\boldsymbol{\mu}) \Big\|^2.$$

Poslední člen je tedy vždy nezáporný. Navíc je rovný nule právě, když $\boldsymbol{\mu} = \bar{\boldsymbol{x}}$, což plyne z definice $\bar{\boldsymbol{x}}$. Prohodíme-li ve druhém členu $\boldsymbol{x}$ a $\boldsymbol{y}$ dostaneme první člen a po vyščítání přes $\boldsymbol{y}$ dostaneme

$$\frac{1}{2|A|} \sum_{\boldsymbol{x},\boldsymbol{y} \in A} \|\boldsymbol{x}-\boldsymbol{y}\|^2 \leq \sum_{\boldsymbol{x} \in A} \|\boldsymbol{x}-\boldsymbol{\mu}\|^2, \quad \text{s rovností pouze pokud } \boldsymbol{\mu} = \bar{\boldsymbol{x}}.$$

□

Algoritmus k-means - úvodní představení

- Problém nalezení globálního minima uvedené účelové funkce je výpočetně obtížný, ve skutečnosti **NP-těžký** (viz [Aloise et al, (2009)]).
- Představíme si heuristický iterativní algoritmus nazývaný **algoritmus k-means**, který bude v každém kroku zmenšovat hodnotu účelové funkce a konvergovat tak k jejímu lokálnímu minimu.

Algoritmus k-means - slovní popis

Nejprve zvolíme k středových bodů.
Iterativně opakujeme:

i. Vytvoříme shluky odpovídající středovým bodům tak, že pro každý bod $\boldsymbol{x}$ najdeme k němu nejbližší středový bod a podle něj $\boldsymbol{x}$ zařadíme do shluku, který tomuto středovému bodu odpovídá.

ii. Spočítáme nové středové body jako geometrické středy těchto shluků.

Algoritmus k-means - důkaz lokální optimalizace

Na základě předchozího tvrzení můžeme účelovou funkci vyjádřit jako

$$G(C) = \sum_{i=1}^{k} \sum_{\boldsymbol{x} \in C_i} \|\boldsymbol{x} - \bar{\boldsymbol{x}}_i\|^2,$$

kde $\bar{\boldsymbol{x}}_i$ je geometrický střed i-tého shluku.

Zafixujme nyní $\boldsymbol{\mu}_i = \bar{\boldsymbol{x}}_i$. Vytvořme nové shluky $\tilde{C} = \{\tilde{C}_1, \ldots, \tilde{C}_k\}$, tak že bod $\boldsymbol{x}$ přesuneme do takového shluku $\tilde{C}_i$, ve kterém je vzdálenost $\|\boldsymbol{x} - \boldsymbol{\mu}_i\|$ nejmenší.

Tím zcela jistě dojde ke zmenšení součtů kvadrátů vzdáleností,

$$\sum_{i=1}^{k} \sum_{\boldsymbol{x} \in \tilde{C}_i} \|\boldsymbol{x} - \boldsymbol{\mu}_i\|^2 \leq \sum_{i=1}^{k} \sum_{\boldsymbol{x} \in C_i} \|\boldsymbol{x} - \boldsymbol{\mu}_i\|^2.$$

Z druhé rovnosti předchozího tvrzení pak plyne, že

$$G(\tilde{C}) = \sum_{i=1}^{k} \sum_{\boldsymbol{x} \in \tilde{C}_i} \|\boldsymbol{x} - \bar{\tilde{\boldsymbol{x}}}_i\|^2 \leq \sum_{i=1}^{k} \sum_{\boldsymbol{x} \in \tilde{C}_i} \|\boldsymbol{x} - \boldsymbol{\mu}_i\|^2,$$

kde $\bar{\tilde{\boldsymbol{x}}}_i$ je geometrický střed shluku $\tilde{C}_i$.
Dohromady tedy máme $G(\tilde{C}) \leq G(C)$.Tento postup můžeme opakovat a postupně tak klesat k nějakému lokálnímu minimu účelové funkce.

Algoritmus k-means - formalizace

Algoritmus k-means

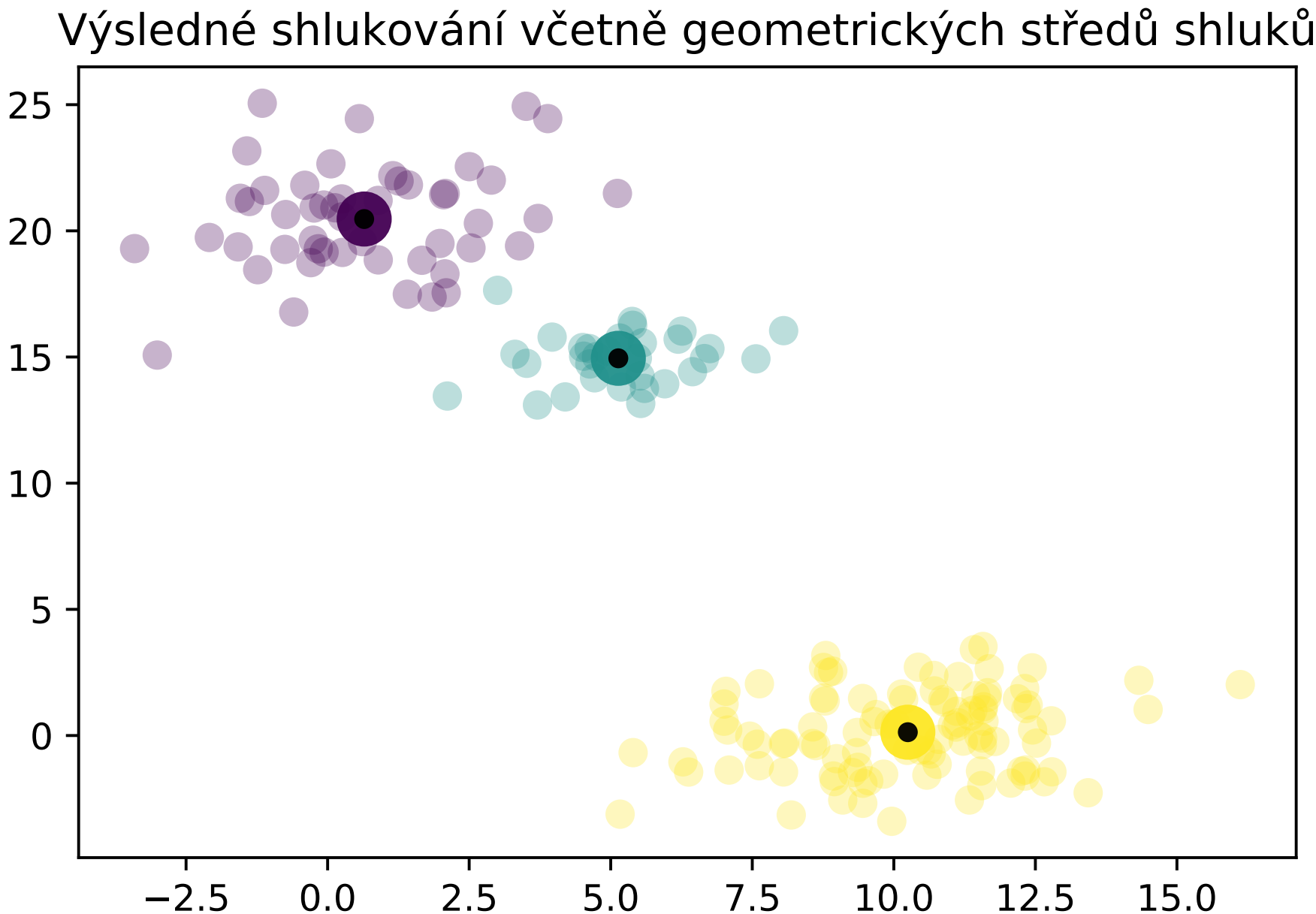
Inicializace: počáteční rozmístění k středových bodů $\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_k$.
Iterativní část:

i. Roztřídíme body do shluků: $C_i = \left\{\boldsymbol{x} \in \mathcal{D} \mid i = \arg\min_j \|\boldsymbol{x} - \boldsymbol{\mu}_j\|\right\}$.

ii. Přepočítáme body $\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_k$ jako geometrické středy těchto shluků: $\boldsymbol{\mu}_i \leftarrow \frac{1}{|C_i|} \sum_{\boldsymbol{x} \in C_i} \boldsymbol{x}$.

- Z předhozích úvah plyne, že v každé iteraci má účelová funkce stejnou nebo nižší hodnotu.
- Běh algoritmu zastavíme, jakmile je změna hodnoty účelové funkce mezi jednotlivými iteracemi dostatečně malá.
- Výsledek algoritmu významně závisí na incializační části.
- Obvykle jsou počáteční středové body generovány náhodně (např. náhodným výběrem z dat), existují ale i „chytřejší metody“ jako např. k-means++ (viz [David, Vassilvitskii, (2007)]).
- Algoritmus následně opakovaně spouštěn. Jako finální pak bereme výsledek běhu s nejnižší hodnotou účelové funkce výsledku.

Algoritmus k-means - ukázka běhu

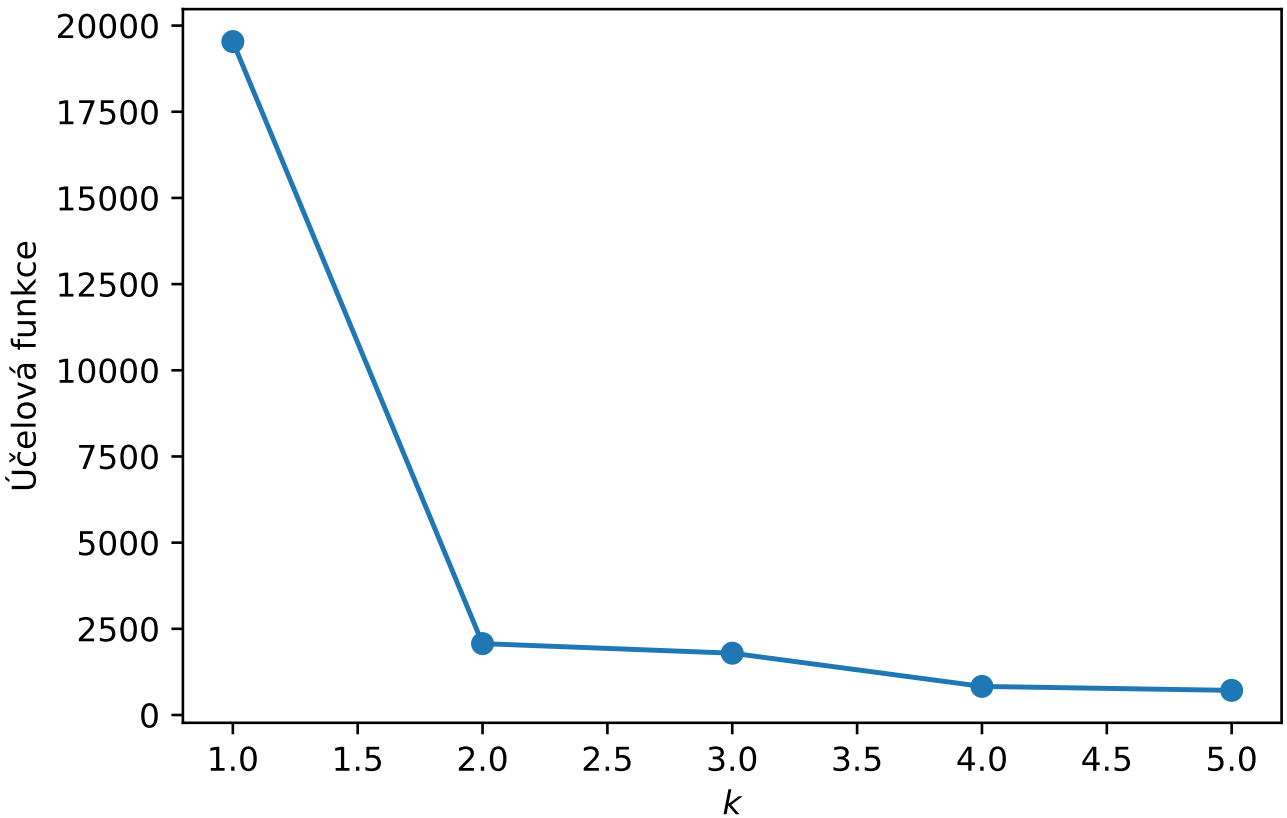


Algoritmus k-means - problematika volby k

- Na rozdíl od hierarchického shlukování, kde můžeme počet shluků určovat až na základě dendrogramu, je u algoritmu k-means nezbytné stanovit k dopředu.
- Bohužel neexistuje žádný univerzální způsob, jak počet shluků určit automaticky.
- Podívejme se tedy alespoň na jeden z používaných heuristických přístupů.
- Je-li k^* optimální počet shluků, lze očekávat, že pro $k < k^*$ bude účelová funkce s měnícím se k klesat hodně. Pro $k \geq k^*$ lze naopak očekávat zmenšení poklesu účelové funkce.
- Optimální k tedy můžeme detekovat jako hodnotu, pro kterou se mění pokles účelové funkce z hodně prudkého na méně prudký, hledat tzv. **loket** (angl. **elbow**). Je to ale hodně subjektivní a pro některá data v podstatě nepoužitelné.
- Existují i jiné metody, viz např. silhouette.

Algoritmus k-means - problematika volby k

Metoda lokte často nedává optimální výsledky, jak je pro předchozí data vidět na následujícím obrázku:



Zde bychom nejspíš určili jako optimální $k = 2$. Přitom ve skutečnosti byla data vygenerována jako směs tří různých dvourozměrných normálních rozdělení.