FAKULTA ELEKTROTECHNIKY A KOMUNIKAČNÍCH TECHNOLOGIÍ
VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ

UČENÍ ZALOŽENÉ NA INSTANCÍCH

Autor textu:
Ing. Petr Honzík, Ph.D.

Květen 2014

Komplexní inovace studijních programů a zvyšování kvality výuky na FEKT VUT v Brně
OP VK CZ.1.07/2.2.00/28.0193

INVESTICE DO ROZVOJE VZDĚLÁVÁNÍ

k-NN, *k* nejbližších sousedů

- Mějme instanci ***x***, chceme určit její výstupní hodnotou *g*
- Máme uložené instance, u kterých známe jejich výstupní hodnoty (učení s učitelem)
- Mezi těmito uloženými prvky nalezneme *k* prvků, které jsou novému prvku ***x*** nejpodobnější
- Výstupní hodnotu prvku ***x*** určíme např. jako nejčastěji zastoupenou hodnou nebo průměr z těchto *k* nejpodobnějších prvků – *k* nejbližších sousedů, k-NN

Některé metriky, normalizace

- instance $\langle a_1(\boldsymbol{x}), a_2(\boldsymbol{x}), \ldots, a_n(\boldsymbol{x}), g \rangle$
  - a_i i-tý atribut vstupní instance $\boldsymbol{x}$
  - g klasifikace instance $\boldsymbol{x}$
- normalizace kvantitativních atributů $x_{norm} = \frac{x - X_{\min}}{X_{\max} - X_{\min}}$
- vzdálenost $d(x_i, x_j)$
  - euklidovská $d(x_i, x_j) = \sqrt{\sum_{r=1}^{n} \left[a_r(x_i) - a_r(x_j)\right]^2}$
  - hammingova $d(x_i, x_j) = \sum_{r=1}^{n} \left|a_r(x_i) - a_r(x_j)\right|$
  - překrytí (kvantitativní) $d(x_i, x_j) = \sum_{r=1}^{n} \left[1 - \partial\left(a_r(x_i), a_r(x_j)\right)\right]$

Klasifikační algoritmus *k*-NN

- výsledný vztah pro výpočet klasifikace:

$$\hat{f}(\boldsymbol{x}_q) = \arg\max_{g \in G} \sum_{i=1}^{k} \partial(g, f(\boldsymbol{x}_i))$$

$$\partial(a,b) = 1, když\ a = b,\ jinak\ \partial(a,b) = 0$$

- jinými slovy, mezi *k* nejbližšími instancemi je určena nejpočetněji zastoupená třída (modus) a do té je také zařazena nová instance

Regresní algoritmus *k*-NN

- pro funkci *f* platí následující relace $f: \boldsymbol{R}^n \rightarrow \boldsymbol{R}$
- výpočet výstupní kvantitativní veličiny *y* je dán vztahem:

$$\hat{f}(\boldsymbol{x_q}) = \frac{\sum_{i=1}^{k} f(\boldsymbol{x_i})}{k}$$

Varianty *k*-NN váhované vzdáleností

- váha *w* rozlišuje přínos souseda podle jeho vzdálenosti vůči analyzovanému prvku

- klasifikace

$$\hat{f}(\boldsymbol{x_q}) = \arg\max \sum_{i=1}^{k} w_i \cdot \partial(g, f(\boldsymbol{x_i}))$$

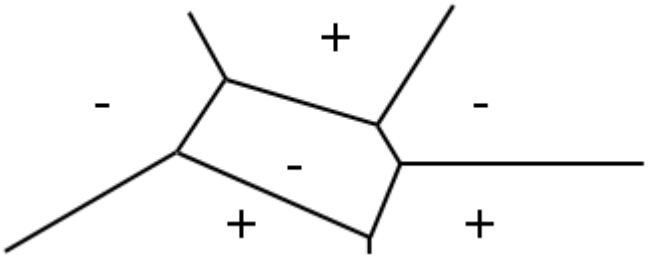
$$w_i = \frac{1}{d(x_q, x_i)^2}$$

- regresní model

$$\hat{f}(\boldsymbol{x_q}) = \frac{\sum_{i=1}^{k} w_i \cdot f(\boldsymbol{x_i})}{\sum_{i=1}^{k} w_i}$$

Velikost *k*

- Zvláštní případ *k=1*
  - Vorného diagram (Voronoid) – rozhodovací prostor indukovaný 1-NN algoritmem (konvexní polygon okolo každé trénovací instance indikuje oblast nejblíže tomuto bodu)



- Velikost *k*
  - malé (1-NN) – **náchylné k šumu** v datech
  - velké – **robustní**

Vlastnosti IBL metod - VÝHODY

- není nutná apriorní znalost o modelu, neparametrická metoda
- tzv. lazy learning (líné učení)
- odolné vůči šumu ve veličinách
- řeší složité problémy s relativně vysokou přesností
- predikce kvantitativních i kvalitativních veličin
- inkremetnální algoritmus

? uveď hlavní výhody a nevýhody metod IBL, vysvětlete princip metod IBL

Vlastnosti IBL metod - NEVÝHODY

- z řešení nevyplývají žádné interpretovatelné souvislosti
- v IB1 nedochází ke generalizaci
- citlivé vůči irelevantním atributům, **nutné předzpracování**
- při velké bázi instancí výpočetně náročné
- minimální schopnost extrapolace
- je třeba mít dostatečně velký počet dat

? uveď hlavní výhody a nevýhody metod IBL, vysvětlete princip metod IBL

Parametry metod typu IBL (typicky *k*-NN)

- **metrika** (uvnitř atributu, mezi prvky)
- **typ okolí** bodu (sférické, elipsa, …)
- **velikosti okolí** neznámého bodu (velikost, počet, procenta?)
- **jaké instance si zapamatovat** (vytvoření modelu)
- **predikce ze sousedů** (z výstupů sousedů, lokální model)

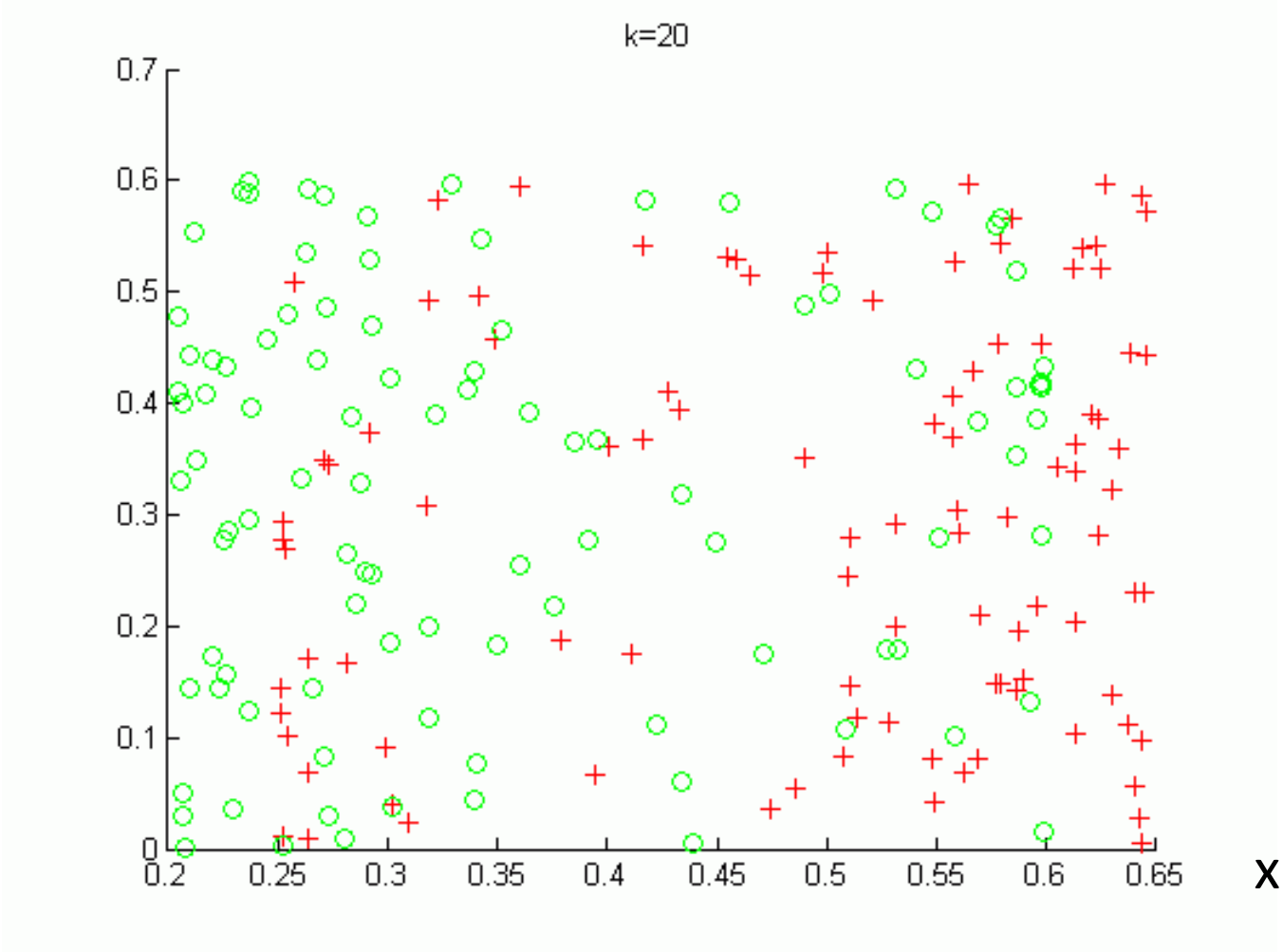
? uveďte základní parametry metod IBL

Co *k*-NN VŽDYCKY VYŽADUJE

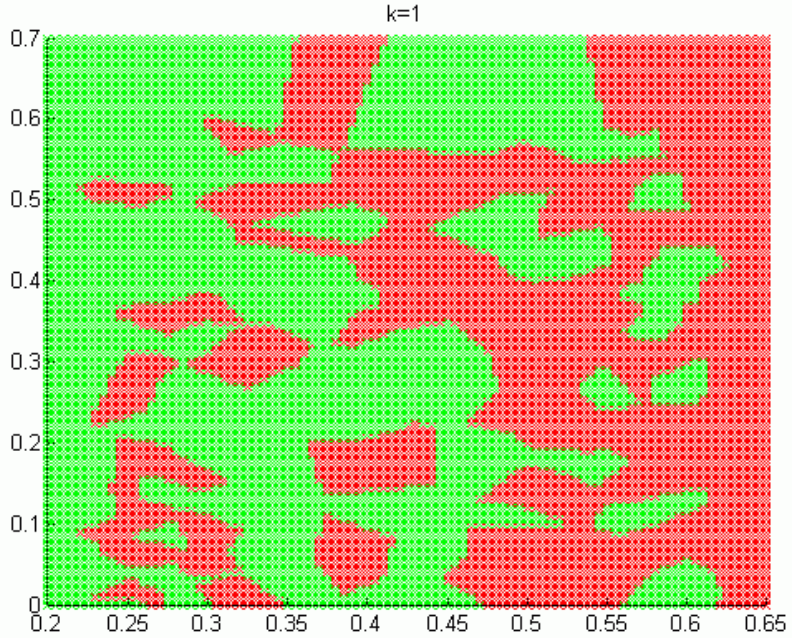
- **dostatečný počet** trénovacích dat
- **normalizaci** vstupních veličin
- **odstranění** irelevantních a redundantních veličin

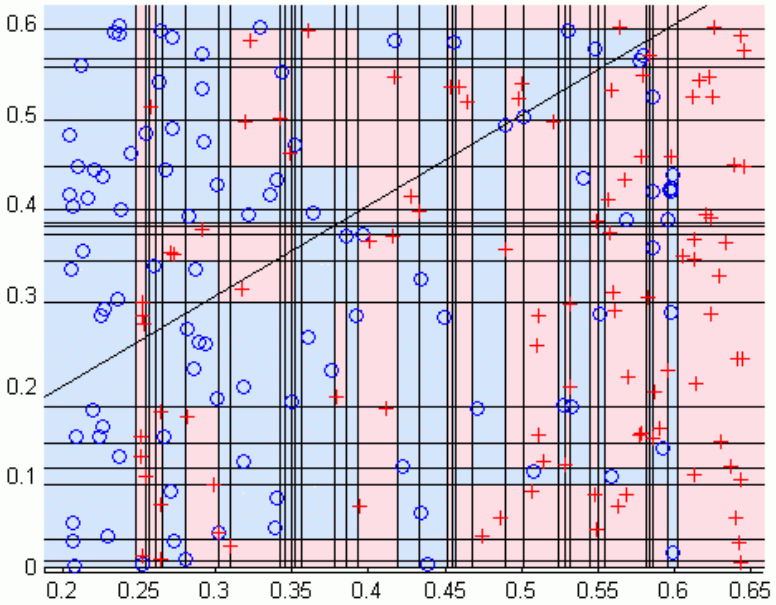
? uveďte základní parametry metod IBL

k-NN - příklad

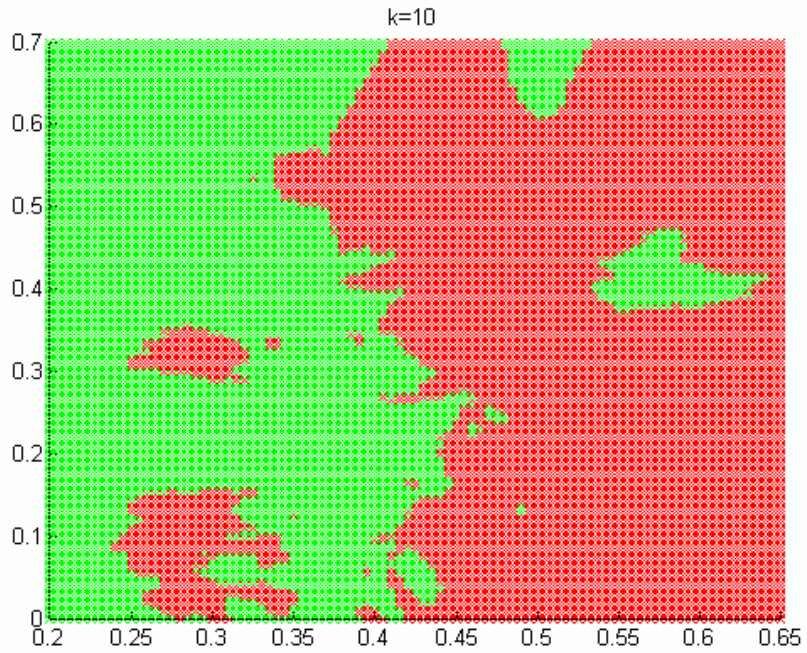


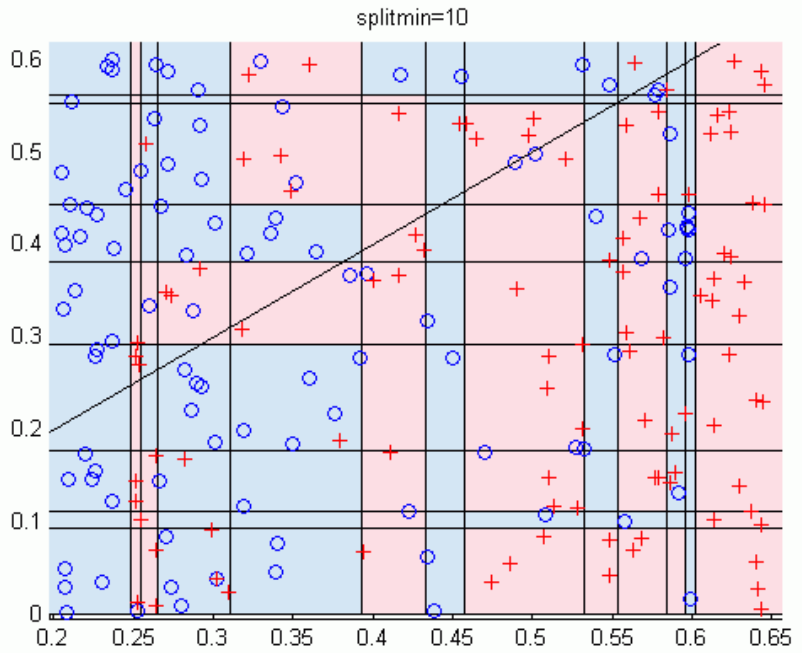
Příklad: *k*-NN vs. rozhodovací strom CART – nejspecifičtější případ (k=1, dělení=1)



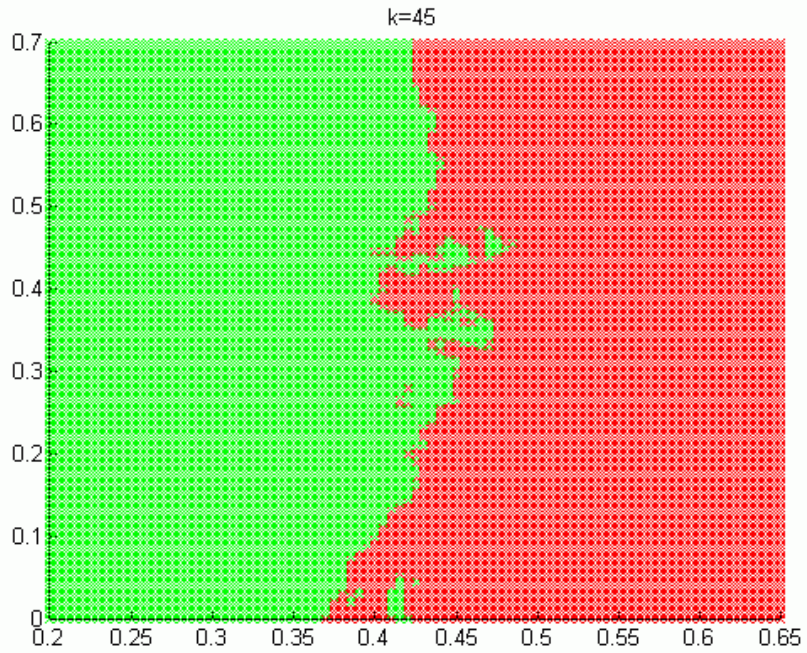


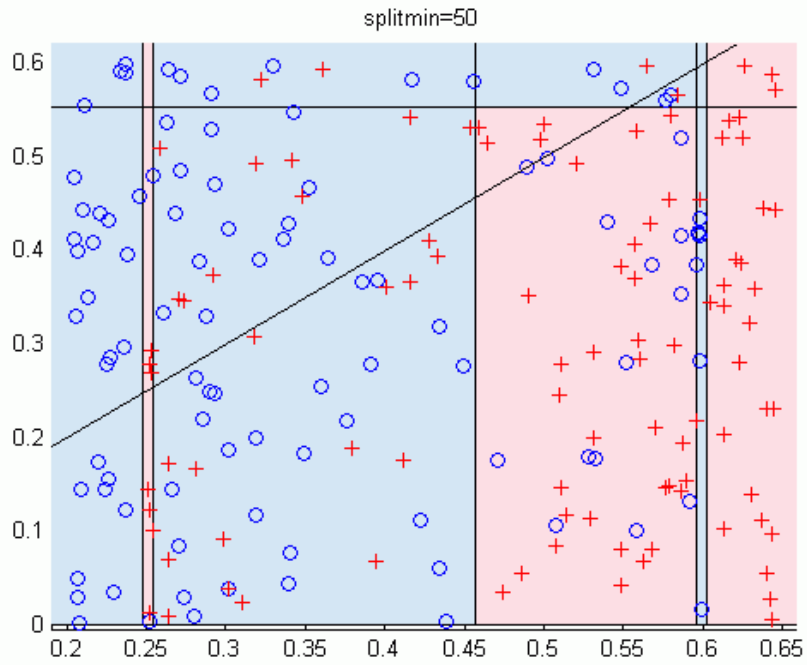
Příklad: *k*-NN vs. rozhodovací strom CART – (k=10, dělení=10)





Příklad: *k*-NN vs. rozhodovací strom CART – významná generalizace (k=45, dělení=45)





Lokálně platné modely

- máme nový prvek (x_q, *?*) a jeho ***k* sousedů** (x_i, y_i)
- ze sousedů vytvořím **lokálně platný model** (typicky lineární, logitová funkce, lze cokoliv…)
- Používá se upravená chybová funkce (s použitím tzv. kernelové funkce)

$$E(\boldsymbol{x_q}) = \frac{1}{2} \cdot \sum_{i=1}^{k} \left(f(\boldsymbol{x}) - \hat{f}(\boldsymbol{x}) \right)^2 \cdot K\left(d(\boldsymbol{x_q}, \boldsymbol{x_k})\right)$$

- Hlavní výhoda
  - schopnost extrapolace
  - řeší velice složité, nelineární a nespojité systémy
- Hlavní nevýhoda
  - časově náročné (zejména při modelu typu neuronová síť…)
  - model platný pouze pro daný klasifikovaný prvek

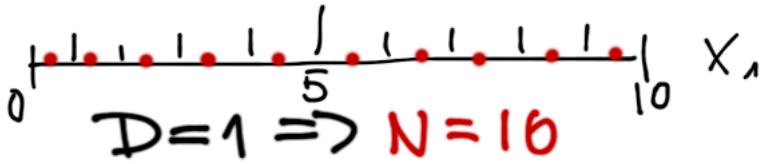
Kde je problém?

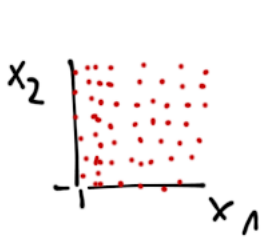
- Curse of dimensionality (prokletí dimenzionality)
- k nalezení nejbližšího souseda je v případě *N* prvků třeba spočítat *N-1* vzdáleností v *M*-dimenzionálním prostoru; je-li $N=10^2$ a $M=5$, proč ne; je-li $N=10^8$, a $M=10^4$ časově náročné (a *M* bude třeba redukovat – viz. přednáška Předzpracování dat).
  - Možná řešení:
  1. **efektivní uložení instancí** do stromové struktury, která umožní rychlé prohledávání: použití k-d stromů
  2. **omezení počtu uložených instancí**: metody IB2, IB3, IB4

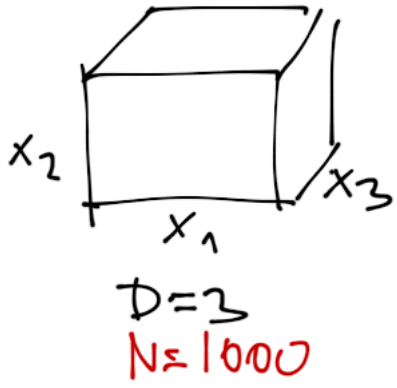
? jak lze řešit problém při velkém počtu instancí v případě metody k-NN

Prokletí dimenzionality (Curse of Dimensionality)

- Vstupní veličina x_1 je definována na intervalu <0;10>.
- Chci mít uloženo *N* prvků, které budou vstupní prostor dostatečně charakterizovat – na jejich základě budu klasifikovat nové prvky.
- Rozhodl jsem se, že mi bude stačit 10 prvků rovnoměrně rozložených na veličině x_i (v hodnotách 0,5; 1,5, ...; 9,5).
- Kolik prvků potřebuju při D=1 (D je počet vstupních veličin), D=2, D=10?



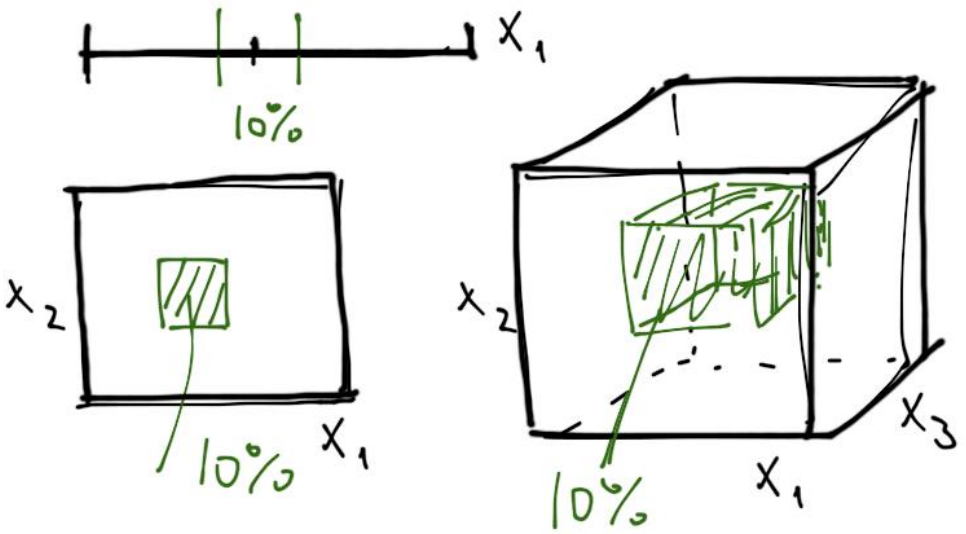


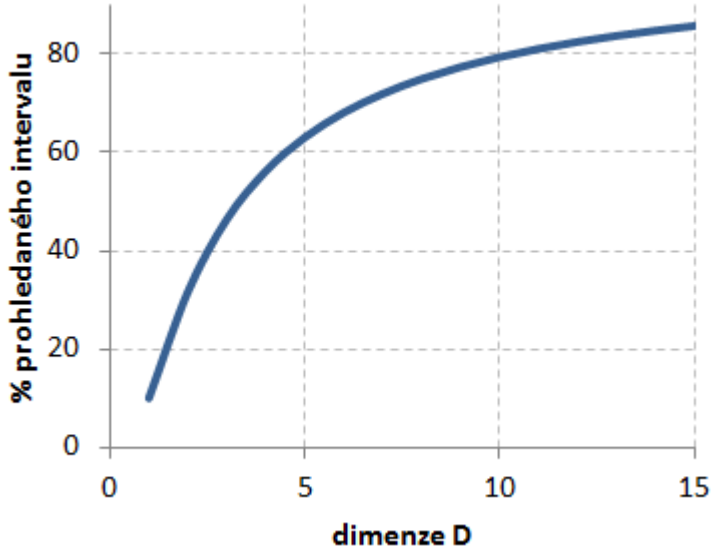


$N = 10^D$

Prokletí dimenzionality (Curse of Dimensionality)

- Chci klasifikovat nový prvek tak, že **prohledám 10%** definičního oboru definovaného *M*-rozměrnou krychlí
- Kolik procent z intervalu na každé vstupní veličině x_i budu prohledávat při D=1, D=2, D=3, D=10?





Při D=1 je třeba prohledat **10%** intervalu každé veličiny.

Při D=2 je třeba prohledat **32%** intervalu každé veličiny.

Při D=3 je třeba prohledat **46%** intervalu každé veličiny.

k-d stromy

- instance lze ukládat do binárního stromu
- k nalezení nejbližšího souseda je tak zapotřebí místo n pouze $log_2 n$ kroků
- každý test ve stromu je charakterizován **souřadnicí** (dělicí atribut), **prahem** (kritická hodnota na atributu) a **neutrální zónou** (velikost okolí prahu, ve které se nenachází žádná instance)
- *k-d* strom je binární, dělí instance na 2 poloviny (podle četnosti)

? čím je charakterizován každý test (uzel) v k-d stromu?

Algoritmus vytváření k-d stromu

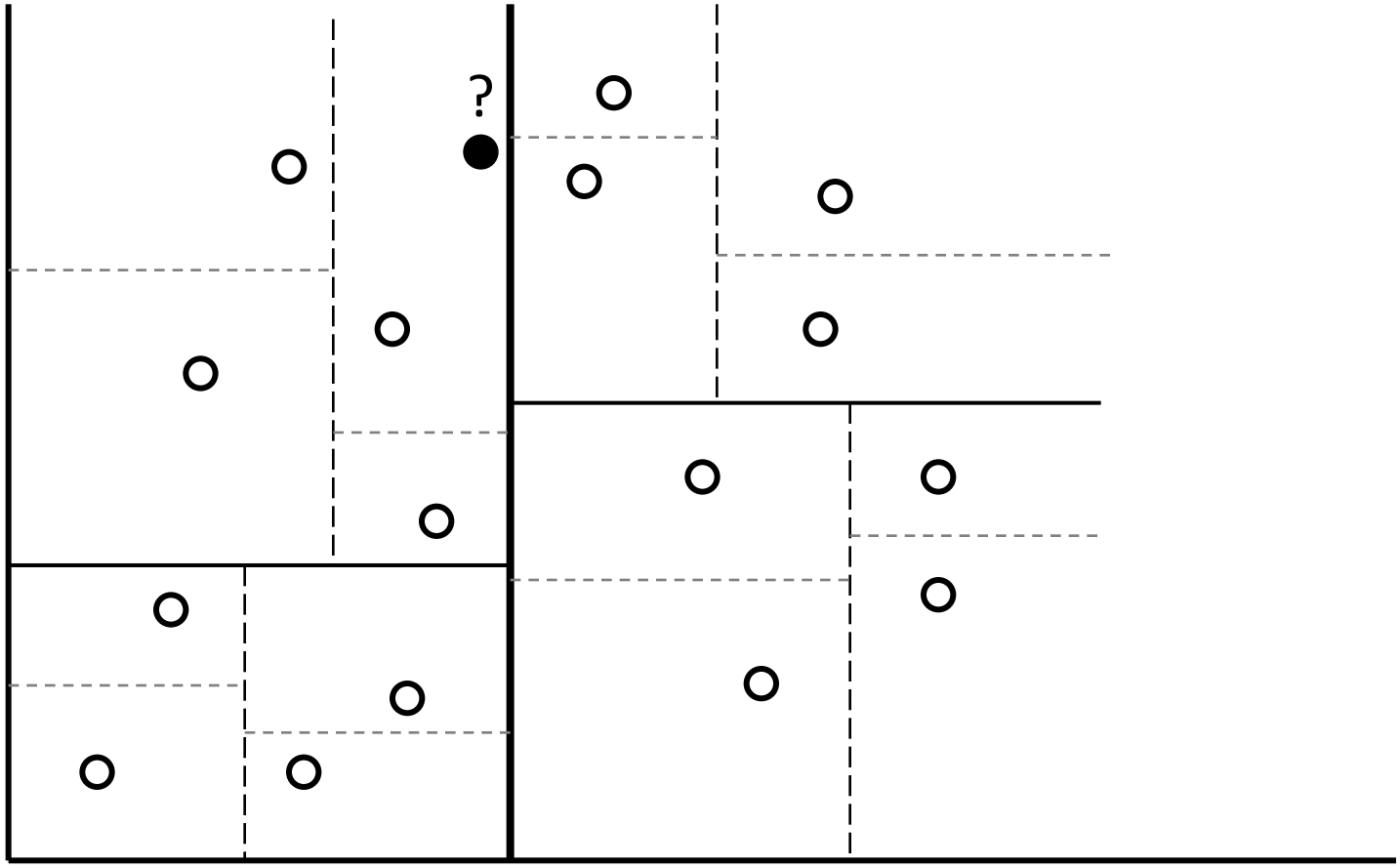
1. je-li jen 1 prvek, KONEC
2. zvol dělicí atribut rozdílný od dříve použitého
3. urči **práh** (hodnotu dělicího atributu), který rozdělí data na 2 stejné skupiny; práh je v polovině vzdálenosti mezi dvěma nejbližšími prvky z rozdílných skupin
4. ulož velikost **neutrální zóny**
5. pokračuj, dokud není každý prvek ve svém listu nebo nelze dál dělit podle žádného atributu

Algoritmus vyhledávání v k-d stromu

1. porovnej klasifikovaný objekt s prahem na ose porovnání a urči novou množinu podobných objektů
2. použitím této procedury nalezni nejbližšího souseda

POZOR! metoda negarantuje nalezení skutečně nejbližšího souseda (podrobněji viz. skripta)

Příklad (k-d strom)



Příklad k-d stromu

- Je dáno 123 prvků, budou uloženy ve stromu vytvořeném pomocí k-d algoritmu. Jaká bude hloubka tohoto binárního stromu? A jaká by byla hloubka k-d stromu, pokud by se v uzlech dělilo do 3 větví?
  - $\log_2 64 < \log_2 123 < \log_2 128$
  - hloubka binárního k-d stromu bude =7
  - $\log_3 81 < \log_3 123 < \log_3 243$
  - hloubka k-d stromu s dělením do 3 větví = 5

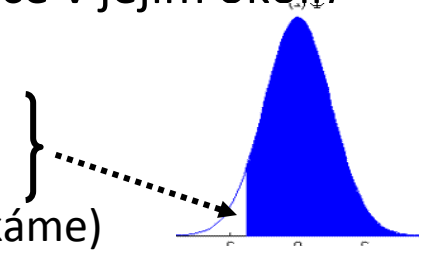
Metody IBL

- metody IB1,…popisují způsob **výběru vhodných prvků**
- 4 základní skupiny algoritmů IB1, IB2, IB3 a IB4
- inkrementální algoritmus s učitelem
- IB1
  - ukládá všechny trénovací prvky
- IB2
  - ukládá pouze chybně klasifikované prvky (uloží první, každý další trénovací uloží pouze je-li chybně zařazen)
  - snížené nároky na paměť
  - velice citlivé na zašuměná data (prakticky vždy uloží)
  - uložené prvky se kumulují kolem hranic (mezi oblastmi s různou klasifikací) a kolem zašuměných záznamů

IBL

- IB3
  - rozšířen o statistické testy, které rozlišují zašuměné instance (vychází z porovnání zařazení instance a klasifikace v jejím okolí)
  - každý nový prvek je buď
    - akceptován (významně dobře klasifikuje, α=0,05)
    - zamítnut (významně špatně klasifikuje, α=0,125)
    - sledován (nelze ani akceptovat ani zamítnout, čekáme)

    hladiny významnosti jsou jednostranné



  - na skutečné klasifikaci se podílí pouze **akceptované** prvky
  - odolné vůči šumu, neschopné akceptovat výjimky
- IB4
  - rozšíření významové interpretace jednotlivých atributů, vhodné v případě jejich velkého počtu - **relevance**
  - relevance atributu (či váha) se mění pro jednotlivé atributy během učení na základě úspěšnosti klasifikace

Jak funguje metoda IB3

Příklad 1 (IB3)

Rozhodni, zda při dané apriorní pravděpodobnosti p_A=0,5 klasifikace do třídy *A* v lze instanci $\boldsymbol{x}_1$ akceptovat, bylo-li jí provedeno N=5 pokusů klasifikovat jiný prvek, z čehož byly N_A=4 pozitivní (úspěšná klasifikace). (Tedy lze na hladině významnosti α=0,05 jednostranně zamítnout H_0: p_A=0,5 a přijmou H_A: p_A> 0,5?)

Přijmeme H_0 a zkusíme zjistit, s jakou pravděpodobností můžeme daný jev (4 z 5) nebo extrémnější (5 z 5) při platnosti H_0 pozorovat.

Pascalův trojúhelník: 1 5 10 10 5 1

Pravděpodobnosti vždy stejné: $0{,}5^5 = 1/32$

Potřebuji, aby 6/32<0,05; zřejmě není splněno.

H_0 nelze zamítnout, při platnosti H_0 mohou výběry (4 z 5) nebo (5 z 5) nastat téměř ve 20% případů. V případě použití této hypotézy v IB3 je prvek dále sledován, není však akceptován (tzn. prvek dále eviduji, na predikci se však nepodílí).

Pozor – výpočet jen z pascalova trojúhelníku, protože p_A=0,5

Příklad 2 (IB3)

Rozhodni, zda při dané apriorní pravděpodobnosti $p_A=0,3$ jevu A lze na hladině významnosti $\alpha=0,25$ zamítnout H_0: $p_A=0,3$. Bylo provedeno $N=500$ pokusů, z čehož byly $N_A=133$ pozitivní (A nastalo).

V Matlabu:

```
[phyp pint]=binofit(150,500,0.25)
phyp =  0.3000
pint =    0.2760    0.3251
```

Výpočet udává interval spolehlivosti (pint) pro p_A=0,3 (150/500) při hladině významnosti $\alpha=0,25$. Protože 133/500 = 0,2660 , ocitá se empirický výsledek mimo tento interval spolehlivosti a vybraný prvek je z databáze záznamů vyloučen.

Lze vypočíst též příkazem binocdf(133,500,0.3), který vrátí hodnotu 0,0525 , tedy pravděpodobnost, že 133 a méně pozitivních výsledků lze vysvětlit jako náhodu při apriorní pravděpodobnosti p_A=0,3.

Literatura

Aha, D. W. & Kibler, D.: Instance-based learning algorithms. *Machine Learning*, 1991, 37-66.