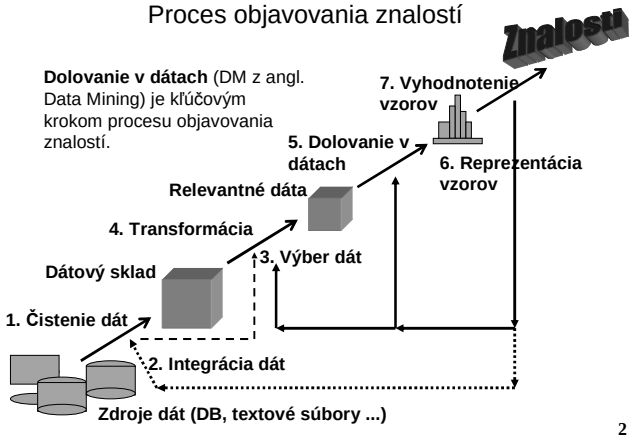
Prediktívne dolovanie v dátach 1. (Klasifikácia)

OBSAH PREDNÁŠKY

- Základné pojmy
- Formálny opis klasifikačnej úlohy
- Rozhodovacie stromy
- Bayesovská klasifikácia
- Klasifikátory na princípe k-najbližších susedov
- Vyhodnotenie kvality klasifikácie
- Zvyšovanie kvality klasifikátorov

Proces objavovania znalostí



Základné pojmy

- ***Prediktívne dolovanie v dátach*** v sebe zahŕňa dve pravdepodobne najčastejšie sa objavujúce úlohy DM
  - Klasifikácia (modeluje a predpovedá nominálne atribúty – triedy)
  - Predikcia (modeluje a predpovedá numerické hodnoty)
- Základný prístup je však u oboch typov DM rovnaký
  - *V prvej fáze* snaží vybudovať (naučiť sa) model správania dát na základe nejakej trénovacej množiny
  - Zostavený (naučený) model správania sa dát je potom *v druhej fáze* používaný na predpovedanie (predikciu) hodnoty cieľového atribútu u nových objektov (záznamov)
- Príklady
  - Klasifikácia žiadateľov o úver v banke
  - Predikcia spotreby pitnej vody

Klasifikácia vs. predikcia

- **Klasifikácia:**
  - Predikuje kategorické označenia tried (*predikovaný atribút je **nominálny***)
  - Klasifikuje dáta (konštruuje model) na základe trénovacej množiny a daných zaradení do tried v klasifikačnom atribúte
  - Skonštruovaný model potom využíva pre klasifikáciu nových príkladov
- **Predikcia:**
  - Modeluje funkcie spojitých premenných, t.j. predikuje neznáme alebo chýbajúce hodnoty spojitého atribútu (*predikovaný atribút je **numerický***)
- Zhlukovanie = **nekontrolované učenie** vs.
- Klasifikácia + predikcia = **kontrolované učenie**

Klasifikácia – dvojkrokový proces (1)

1. **Konštrukcia modelu**: popisujúceho množinu preddefinovaných tried
   - Predpokladá sa, že každý príklad patrí do jednej z preddefinovaných tried tak, ako to určuje hodnota klasifikačného atribútu
   - Množina príkladov použitá pre konštrukciu klasifikačného modelu: trénovacia množina
   - Model môže byť reprezentovaný vo forme:
     - Logické konjunkcie (VSS, EGS, HGS ...)
     - Rozhodovacie stromy (ID3, C4.5, ID5R, ...)
     - Rozhodovacie zoznamy (NEX, CN2, RISE, ...)
     - Pravdepodobnostný popis (Naivný Bayes, Bayesovské siete ...)

Klasifikácia – dvojkrokový proces (2)

2. **Použitie modelu**: pre klasifikáciu budúcich (neznámych) prípadov
   - Odhad presnosti modelu
     - Známe zatriedenie testovacích príkladov je porovnávané s klasifikáciou na základe vygenerovaného modelu
     - Presnosť je percentuálny podiel testovacích príkladov, ktoré boli modelom klasifikované správne
     - Testovacia množina musí byť nezávislá na trénovacej, ináč hrozí preučenie

Formálny popis klasifikácie

- Daná je množina O objektov $o = (o_1, ..., o_d)$
- c_j je trieda, $c_j \in C = \{ c_1, ..., c_n \}$
- D je *základný súbor objektov*, ktoré je potrebné klasifikovať
- Pre každý z objektov v D sú známe hodnoty atribútov A_i, $1 \le i \le d$
- Príslušnosť D k triede je známa len u objektov z tzv. *trénovacej množiny* $O \subset D$
- Hodnota triedy teda nie je známa u objektov z $D \setminus O$
- *Klasifikátor* je potom funkcia K, K: $D \rightarrow C$

Ilustračný príklad (1)

ID (A_1)	Vek (A_2)	Typ auta (A_3)	Riziko (C)
1	23	rodinné	vysoké
2	17	športové	vysoké
3	43	športové	vysoké
4	68	rodinné	nízke
5	32	nákladné	nízke
6	35	rodinné	
7	58	rodinné	

c_1 = vysoké
c_2 = nízke
O
D

Ilustračný príklad (2)

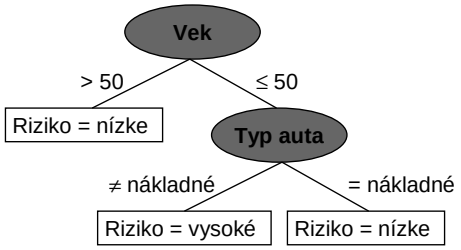
Príklad možného klasifikátora:

```
if Vek > 50 then Riziko = nízke;
if Vek ≤ 50 and Typ auta ≠ nákladné
  then Riziko = vysoké;
if Vek ≤ 50 and Typ auta = nákladné
  then Riziko = nízke;
```


Rozhodovacie stromy

- Rozhodovacie stromy sú najpopulárnejšou formou reprezentácie klasifikátorov najmä pre svoju ľahko pochopiteľnú reprezentáciu získaných znalostí
- **Rozhodovací strom** je strom s nasledujúcimi vlastnosťami:
  - ***Medziľahlý uzol*** reprezentuje vybraný atribút (prípadne skupinu atribútov)
  - ***Listový uzol*** reprezentuje niektorú z tried
  - ***Hrana*** reprezentuje test na atribút (skupinu atribútov) z nadradeného uzla

Príklad rozhodovacieho stromu



Algoritmus

KonštruujRozhodovacíStrom (TrénovaciaMnožina T, Float *min_conf*)
  if minimálne *min_conf* objektov z T patrí do triedy C
    then vytvor listový uzol zaradzujúci do C;
    return;
  else
    for each atribút A **do**
      for each možné rozdelenie hodnôt A **do**
        ohodnoť kvalitu rozdelenia, ktoré by takýmto spôsobom vzniklo;
    vykonaj najlepšie zo všetkých možných rozdelení;
    nech $T_1, T_2, ..., T_m$ sú množiny ktoré vzniknú týmto rozdelením;
    KonštruujRozhodovacíStrom(T_1, *min_conf*);
    ...
    KonštruujRozhodovacíStrom(T_m, *min_conf*);
    return;

Výber testovacieho atribútu (1)

- Teória informácií vyžíva pre meranie množstva informácie entrópiu.
  - Ak jednotlivé správy $x_1, x_2, \ldots, x_n$ sú možné s pravdepodobnosťami $p(x_1), p(x_2), \ldots, p(x_n)$
  - pričom pravdepodobnosti vytvárajú úplný súbor pravdepodobností $\sum_{j=1}^{n} p(x_j) = 1$
  - potom entrópiu (neurčitosť) súboru správ $x_1, x_2, \ldots, x_n$ možno vyjadriť ako $H = -\sum_{j=1}^{n} p(x_j)\log_2(p(x_j))\ [bit]$
- V rozhodovacom strome:
  - v koreňovom uzle je entrópia maximálna
  - v listových uzloch minimálna, prípadne nulová

Výber testovacieho atribútu (2)

- Ak klasifikačné triedy príkladov sú $c_1, c_2, \ldots, c_n$, potom *entrópia v uzle S* je *H(S)*

$$H(S) = -\sum_{j=1}^{n} p(c_j)\log_2(p(c_j))$$

- Ak použitím atribútu A_i sa uzol rozvetví na *m* vetiev $s_1, s_2, \ldots, s_m$, tak *celková entrópia v uzle S použitím atribútu* A_i na jeho rozdelenie bude

$$H(S, A_i) = \sum_{j=1}^{m} p(s_j)H(S_j)$$

- Algoritmus ID3 používa ako kritérium pre výber testovacieho atribútu tzv. informačný zisk $I(S, A_i)$

$$I(S, A_i) = H(S) - H(S, A_i)$$

Výber testovacieho atribútu (3)

- ***Kritérium informačný zisk*** má závažný nedostatok v uprednostňovaní výberu testovacích podmienok s mnohými výstupmi
- Preto napr. algoritmus C4.5 používa normalizovaný informačný zisk, tzv. ***pomerový informačný zisk*** $I_p(S,A_i)$

$$I_p(S, A_i) = \frac{I(S, A_i)}{H_p(S, A_i)}$$

- kde $H_p(S,A_i)$ je tzv. pomerová entrópia

$$H_p(S, A_i) = -\sum_{j=1}^{m} p(s_j)\log_2(p(s_j))$$

Výber testovacieho atribútu (4)

- Testovacia podmienka pre spojité atribúty pozostáva z prahovej hodnoty *h*, ktorá rozdeľuje usporiadanú množinu čísel na dve podmnožiny
- Algoritmus C4.5 vyberá prahovú hodnotu nasledovne:
  - Trénovacie príklady sú najprv usporiadané vzostupne podľa daného spojitého atribútu $\{v_1, v_2, \ldots, v_k\}$
  - Prahová hodnota ležiaca medzi dvoma hodnotami v_i a v_{i+1}, rozdelí množinu príkladov na množiny $\{v_1, v_2, \ldots, v_i\}$ a $\{v_{i+1}, v_{i+2}, \ldots, v_k\}$. Takýchto možných rozdelení je *k-1*.
  - Pre všetky rozdelenia sa vypočíta hodnotiaca funkcia a vyberie sa rozdelenie s maximálnym ohodnotením. Hodnotiacou funkciou môže byť informačný zisk alebo pomerový informačný zisk.
  - Prahová hodnota medzi dvoma hodnotami spojitého atribútu v_i a v_{i+1} sa určí ako ich priemerná hodnota, t.j $h = \frac{v_i + v_{i+1}}{2}$

Výber testovacieho atribútu (5)

- Iným kritériom pre výber testovacej podmienky je tzv. ***Gini-index*** (používaný napr. v systéme IBM Intelligent Miner)
- Gini-index pre množinu trénovacích príkladov *T* označovaný *gini(T)* možno vypočítať nasledovne:

$$gini(T) = 1 - \sum_{i=1}^{n} p_i^2$$

  - kde p_i je relatívna početnosť triedy c_i
- Gini-index pre rozdelenie množiny *T* na podmnožiny $T_1, T_2, \ldots, T_k$ označovaný $gini(T_1, T_2, \ldots, T_k)$ sa vypočíta nasledovne $gini(T_1, T_2, \ldots, T_k) = \sum_{i=1}^{k} \frac{|T_i|}{|T|} \cdot gini(T_i)$

Generovanie rozhodovacieho stromu – ilustračný príklad

Počasie	Teplota [℃]	Vlhkosť [%]	Vietor?	Trieda
Slnečno	24	70	Áno	Hrá sa
Slnečno	27	90	Áno	Nehrá sa
Slnečno	29	85	Nie	Nehrá sa
Slnečno	22	95	Nie	Nehrá sa
Slnečno	21	70	Nie	Hrá sa
Zamračené	22	90	Áno	Hrá sa
Zamračené	28	78	Nie	Hrá sa
Zamračené	18	65	Áno	Hrá sa
Zamračené	27	75	Nie	Hrá sa
Dážď	22	80	Áno	Nehrá sa
Dážď	18	70	Áno	Nehrá sa
Dážď	24	80	Nie	Hrá sa
Dážď	20	80	Nie	Hrá sa
Dážď	21	96	Nie	Hrá sa

Koreňový uzol (1)

- Po inicializácii sa v koreňovom uzle nachádzajú všetky príklady
- Entrópia v koreňovom uzle *je*

  $H(S_0) = -p(\text{Hrá sa})*\log_2(p(\text{Hrá sa})) - p(\text{Nehrá sa})*\log_2(p(\text{Nehrá sa}))$
  $= -9/14*\log_2(9/14) - 5/14*\log_2(5/14) =$ ***0.940***

1. V prípade výberu atribútu ***A_1* = „*Počasie*“** by bolo výsledné delenie jednoznačné na 3 vetvy:

   $H(\text{Slnečno}) = -2/5*\log_2(2/5) - 3/5*\log_2(3/5) = 0.971$

   $H(\text{Zamračené}) = -4/4*\log_2(4/4) = 0$

   $H(\text{Dážď}) = -3/5*\log_2(3/5) - 2/5*\log_2(2/5) = 0.971$

- Celková entrópia v koreňovom uzle za predpokladu, ak sa na jeho rozdelenie použije atribút *Počasie,* bude:

  $H(S_0, \text{Počasie}) = p(\text{Slnečno})*H(\text{Slnečno}) + p(\text{Zamračené})*H(\text{Zamračené}) + p(\text{Dážď})*H(\text{Dážď})$
  $= 5/14*0.971 + 4/14*0 + 5/14*0.971 =$ ***0.694***

  Teda zodpovedajúci informačný zisk bude
  $I(S_0, \text{Počasie}) = H(S_0) - H(S_0, \text{Počasie}) =$ ***0.246***

Koreňový uzol (2)

- Pre výpočet pomerového informačného zisku musíme ešte najprv určiť pomerovú entrópiu

  $H_p(S_0, \text{Počasie}) = -p(\text{Slnečno})*\log_2(p(\text{Slnečno})) - p(\text{Zamračené})*\log_2(p(\text{Zamračené})) - p(\text{Dážď})\log_2(p(\text{Dážď}))$
  $= -5/14*\log_2(5/14) - 4/14*\log_2(4/14) - 5/14*\log_2(5/14) =$ ***1.577***

- Takže nakoniec pomerový informačný zisk v koreňovom uzle pre prípad výberu testovacieho atribútu ***Počasie*** na rozdelenie príkladov bude

  $I_p(S_0, \text{Počasie}) = I(S_0, \text{Počasie}) / H_p(S_0, \text{Počasie}) =$ ***0.157***

2. V prípade výberu atribútu ***A_2* = „*Teplota*“** je možných niekoľko rozdelení, vždy však na dve vetvy podľa zvolenej hranice h, pričom jednou vetvou pôjdu príklady s hodnotou $A_2 \leq h$ a druhou potom ostatné, t.j. u ktorých $A_2 > h$

Koreňový uzol (3)

- Najprv je potrebné zoradiť vyskytujúce sa hodnoty atribútu *Teplota* a určiť možné alternatívne prahové hodnoty.
  - Usporiadané hodnoty: {18, 20, 21, 22, 24, 27, 28, 29}
  - Prahové hodnoty: {19, 20.5, 21.5, 23, 25.5, 27.5, 28.5}
- Pre prvú prahovú hodnotu (19) sa vypočíta pomerový informačný zisk podobne ako v predchádzajúcich výpočtoch

  $H(\text{Teplota} \leq 19) = -1/2*\log_2(1/2) - 1/2*\log_2(1/2) = 1$

  $H(\text{Teplota} > 19) = -4/12*\log_2(4/12) - 8/12*\log_2(8/12) = 0.918$

  $H(S_0, \text{Teplota}(19)) = 2/14*1 + 12/14*0.918 = 0.930$

  $I(S_0, \text{Teplota}(19)) = H(S_0) - H(S_0, \text{Teplota}(19)) = 0.010$

  $H_p(S_0, \text{Teplota}(19)) = -2/14*\log_2(2/14) - 12/14*\log_2(12/14) = 0.592$

  $I_p(S_0, \text{Teplota}(19)) = I(S_0, \text{Teplota}(19)) / H_p(S_0, \text{Teplota}(19)) =$ ***0.017***

Koreňový uzol (4)

- Takýmto postupom sa vypočítajú pomerové informačné zisky pre všetky prahové hodnoty:

	19	20.5	21.5	23	25.5	27.5	28.5
I_p	0.027	0	0.048	0.001	0.029	0.017	**0.304**

3. Pomerový informačný zisk pre rozdelenia koreňového uzla pomocou atribútu ***A_3* = „*Vlhkosť*“** možno určiť rovnakým postupom ako pre atribút A_2 (*Teplota*)

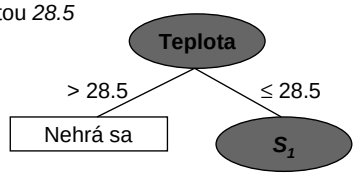
   $I_p(S_0, \text{Vlhkosť}(95.5)) =$ ***0.129***

4. Pomerový informačný zisk pre rozdelenia koreňového uzla pomocou atribútu ***A_4* = „*Vietor*“** možno určiť rovnakým postupom ako pre atribút A_1 (*Počasie*)

   $I_p(S_0, \text{Vietor}) =$ ***0.049***

Koreňový uzol (5)

- Maximálny pomerový informačný zisk v koreňovom uzle sa teda dosiahne použitím atribútu *Teplota* s prahovou hodnotou *28.5*

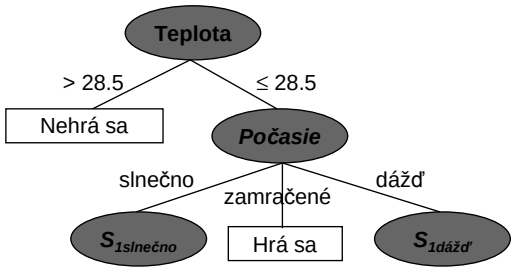


- Ľavá vetva je ukončená listovým uzlom s triedou *Nehrá sa*, ktorý obsahuje jeden trénovací príklad
- V pravej vetve nie je splnená koncová podmienka (teda vzniká medziľahlý uzol S_1)

Ďalší výpočet pre uzol S_1

Atribút	**Počasie**	**Teplota** (prah)	**Vlhkosť** (prah)	**Vietor**
$I_p(S_1, A_i)$	**0.133**	0.11 (27.5)	0.11 (95.5)	0.11
Atribút *Počasie* rozvetví uzol S_1 na $S_{1Slnečno}$, $S_{1Zamračené}$ (listový uzol) a $S_{1Dážď}$				

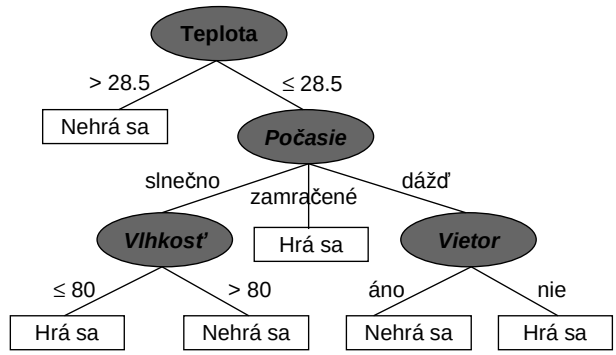
Strom po rozvetvení uzla S_1



Ďalší výpočet pre uzly $S_{1slnečno}$ a $S_{1dážď}$

Atribút	**Počasie**	**Teplota** (prah)	**Vlhkosť** (prah)	**Vietor**
$I_p(S_{1Slnečno}, A_i)$	-	0.383 (25.5)	**1 (80)**	0
$I_p(S_{1Dážď}, A_i)$	-	0.446 (19)	0.446 (75)	**1**

Výsledný rozhodovací strom



Orezávanie rozhodovacích stromov

- **Pre-prunning** – počas generovania rozhodovacieho stromu. Ak hodnota zvolenej štatistickej mierky významnosti (χ^2, informačný zisk a pod.) pre vybraný testovací atribút nepresiahne stanovený prah, ďalšie vetvenie sa zastaví.
- **Post-prunning** – po vygenerovaní rozhodovacieho stromu. Toto orezávanie odstraňuje vetvy z už vygenerovaného stromu takým spôsobom, že porovnáva veľkosť očakávanej chyby klasifikácie pre daný podstrom a jeho náhradu listovým uzlom. Ak sa chyba po náhrade listovým uzlom zmenší, daný podstrom je možné odrezať.

Bayesovská klasifikácia

- Bayesovské klasifikátory predikujú pravdepodobnosti, s ktorými daný príklad patrí do tej – ktorej triedy
- Vychádzajú pritom z určenia podmienených pravdepodobností jednotlivých hodnôt atribútov pre rôzne triedy
- *Bayesovská teoréma* hovorí ako možno vypočítať podmienenú pravdepodobnosť *P(H|X)*

$$P(H \mid X) = \frac{P(X \mid H) \cdot P(H)}{P(X)}$$

  - *X* je objekt (povedzme červený a guľatý)
  - *C* je trieda (napr. typ ovocia)
  - *H* je hypotéza (napr. že *X* patrí do triedy c_1, t.j. jablká)
  - *P(H)* a *P(X)* sú apriórne pravdepodobnosti
  - *P(X|H)* je posteriórna pravdepodobnosť, ktorú je možné určiť na základe danej databázy

Naivný Bayesovský klasifikátor (1)

- Vychádza z predpokladu, že efekt, ktorý má hodnota (každého) atribútu na danú triedu, nie je ovplyvnený hodnotami ostatných atribútov
  - O je množina objektov $o = (o_1, ..., o_d)$
  - Pre každý objekt o sú známe jeho hodnoty atribútov A_i, $1 \le i \le d$, ako aj trieda c_j, $c_j \in C = \{c_1, ..., c_n\}$
  - Neznámy príklad $X = (x_1, ..., x_d)$ bude klasifikovaný do triedy c_i s najväčšou posteriórnou pravdepodobnosťou $P(c_i|X) > P(c_j|X)$, $i \neq j$

$$P(c_i \mid X) = \frac{P(X \mid c_i) \cdot P(c_i)}{P(X)}$$

  - Keďže $P(X)$ je konštantná pre všetky triedy c_i, stačí nájsť minimálnu hodnotu výrazu $P(X| c_i) \cdot P(c_i)$.
  - Pravdepodobnosť zaradenia ľubovoľného objektu do triedy c_i je

$$P(c_i) = \frac{N_O^i}{N_O}$$

N_o je počet všetkých príkladov z trénovacej množiny O
N_o^i je počet tých príkladov z O, ktoré patria do triedy c_i

Naivný Bayesovský klasifikátor (2)

- Pravdepodobnosti $P(X|c_i)$. Tieto pri predpoklade nezávislosti jednotlivých atribútov A_i možno vypočítať nasledovne:
  - Pre kategorické atribúty:

$$P(X \mid c_i) = \prod_{k=1}^{d} P(x_k \mid c_i)$$

$$P(x_k \mid c_i) = \frac{N_O^{i,k}}{N_O^i}$$

$N_o^{i,k}$ je počet tých príkladov z O, ktoré patria do triedy c_i a pre ktorých hodnota atribútu $A_k = x_k$

  - Pre *spojité atribúty* A_k sa obvykle predpokladá Gaussovo normálne rozdelenie hodnôt, a potom:

$$P(X \mid c_i) = \frac{1}{\sqrt{2\pi}\sigma_{c_i}} \cdot e^{-\frac{x_k - \mu_{c_i}}{2\sigma^2_{c_i}}}$$

μ_{c_i} je stredná hodnota a
σ_{c_i} je rozptyl hodnôt atribútu A_k z tých príkladov trénovacej množiny, ktoré patria do triedy c_i

Naivný Bayesovský klasifikátor – ilustračný príklad

ID	vek	príjem	študent	kreditné hodnotenie	trieda (kúpi si počítač)
1	≤ 30	vysoký	nie	priemerné	nie
2	≤ 30	vysoký	nie	výborné	nie
3	31 .. 40	vysoký	nie	priemerné	áno
4	> 40	stredný	nie	priemerné	áno
5	> 40	nízky	áno	priemerné	áno
6	> 40	nízky	áno	výborné	nie
7	31 .. 40	nízky	áno	výborné	áno
8	≤ 30	stredný	nie	priemerné	nie
9	≤ 30	nízky	áno	priemerné	áno
10	> 40	stredný	áno	priemerné	áno
11	≤ 30	stredný	áno	výborné	áno
12	31 .. 40	stredný	nie	výborné	áno
13	31 .. 40	vysoký	áno	priemerné	áno
14	> 40	stredný	nie	výborné	nie

Úloha

- Je potrebné navrhnúť naivný Bayesovský klasifikátor a klasifikovať ním nového zákazníka
- Nový zákazník X má tieto hodnoty atribútov:
  - vek = „≤ 30“
  - príjem = „stredný“
  - študent = „áno“
  - kreditné ohodnotenie = „priemerné“
- P(**X si kúpi počítač** | vek X = „≤ 30“ ∧ príjem X = „stredný“ ∧ X je študent ∧ kreditné ohodnotenie X = „priemerné“) = ?
- P(**X si nekúpi počítač** | vek X = „≤ 30“ ∧ príjem X = „stredný“ ∧ X je študent ∧ kreditné ohodnotenie X = „priemerné“) = ?

Postup výpočtu (1)

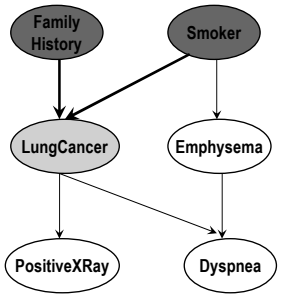
- Apriórne pravdepodobnosti jednotlivých tried $\boldsymbol{P(c_i)}$ možno jednoducho určiť z trénovacej množiny
  P(kúpi si počítač) = 9/14 = **0.643**
  P(nekúpi si počítač) = 5/14 = **0.357**
- Pre výpočet posteriórnych pravdepodobností $P(X|c_i)$ je potrebné najskôr vypočítať nasledovné podmienené pravdepodobnosti $\boldsymbol{P(x_k|c_i)}$ pre jednotlivé hodnoty atribútov A_k zadaného nového zákazníka X:
  P(vek = „≤ 30“ | kúpi si počítač) = 2/9 = 0.222
  P(vek = „≤ 30“ | nekúpi si počítač) = 3/5 = 0.600
  P(príjem = „stredný“ | kúpi si počítač) = 4/9 = 0.444
  P(príjem = „stredný“ | nekúpi si počítač) = 2/5 = 0.400
  P(študent = „áno“ | kúpi si počítač) = 6/9 = 0.667
  P(študent = „áno“ | nekúpi si počítač) = 1/5 = 0.200

Postup výpočtu (2)

P(kreditné ohodnotenie = „priemerné“ | kúpi si počítač) = 6/9 = 0.667
P(kreditné ohodnotenie = „priemerné“ | nekúpi si počítač) = 2/5 = 0.400

- Použitím týchto pravdepodobností možno vypočítať hodnoty $P(X|c_i)$ pre jednotlivé triedy:
  P(X | kúpi si počítač) = 0.222 x 0.444 x 0.667 x 0.667 = ***0.044***
  P(X | nekúpi si počítač) = 0.600 x 0.400 x 0.200 x 0.400 = ***0.019***
- A následne aj hodnoty súčinov $P(X|c_i) \cdot P(c_i)$: pre jednotlivé triedy:
  P(X | kúpi si počítač) x P(kúpi si počítač) = 0.044 x 0.643 = ***0.028***
  P(X | nekúpi si počítač) x P(nekúpi si počítač) = 0.019 x 0.357 = ***0.007***
- To znamená, že naivný Bayesovský klasifikátor bude klasifikovať daný objekt do triedy **c_1 = „kúpi si počítač“**

Bayesovské siete



	(FH, S)	(FH, ~S)	(~FH, S)	(~FH, ~S)
LC	0.8	0.5	0.7	0.1
~LC	0.2	0.5	0.3	0.9

Tabuľka podmienených pravdepodobností pre premennú LC (LungCancer)

Klasifikátory na princípe k-NN (1)

- Založené na princípe učenia sa na základe analógie
- Spravidla sa predpokladá, že trénovacie príklady (množina *O*) sú popísané *n* numerickými atribútmi, a teda predstavujú vlastne body v *n*-dimenzionálnom priestore príkladov
- Ak príde nový príklad *q* s ešte neznámou hodnotou cieľového atribútu (triedy), klasifikátor na princípe *k*-najbližších susedov hľadá v priestore príkladov takých *k* trénovacích príkladov, ktoré sú k novému príkladu najbližšie (napr. v zmysle euklidovskej vzdialenosti)
- Neznámy príklad je potom klasifikovaný do tej triedy, ktorá sa najčastejšie vyskytuje medzi *k* jemu najbližšími trénovacími príkladmi

Klasifikátory na princípe k-NN (2)

- Tento typ klasifikátorov nekonštruuje žiaden klasifikačný model, t.j. odpadá vlastne prvá fáza klasifikácie, konštrukcia modelu
- Ku klasifikácii tu dochádza až v momente, keď je potrebné klasifikovať nejaký nový, dovtedy neznámy príklad
- Takéto klasifikátory sa zvyknú v literatúre nazývať aj
  - klasifikátory založené na inštanciách (**instance-based**), resp.
  - hovorí sa o „lenivom učení“ (**lazy learners**)
- Nevýhodou takýchto prístupov je skutočnosť, že potrebujú dlhší čas na klasifikáciu
  - Tu sa zvyknú používať rôzne efektívne indexovacie techniky

Klasifikátory na princípe k-NN (3)

- Jednotlivé varianty klasifikátorov na princípe k-najbližších susedov sa môžu líšiť:
  - Akým spôsobom vyberajú najbližších susedov
  - Koľko ich vyberú pre klasifikáciu nového príkladu
  - Ako vplývajú jednotliví susedia na rozhodnutie o výslednej triede nového príkladu
- Niektoré z používaných klasifikátorov na princípe *k*-najbližších susedov pracujú nasledovne:
  - Ako trénovacie príklady sa používajú len vektory stredných hodnôt pre jednotlivé triedy
  - Pre rozhodnutie o zaradení nového príkladu do triedy sa berie do úvahy len jeho najbližší sused (t.j. $k = 1$)
  - Vplyv jednotlivých susedov na rozhodnutie o klasifikačnej triede závisí od ich vzdialenosti

Príklad algoritmu k-NN

Klasifikátor_k-NajbližšíchSusedov (TrénovaciePríklady *O*, Objekt *q*, Integer *k*)
Vyber ako rozhodovaciu množinu E *k*-najbližších susedov objektu *q* z množiny *O*;

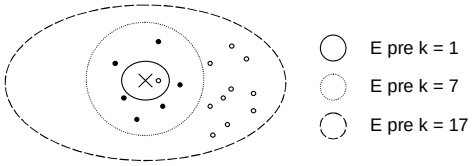
$$Trieda = \arg\max_{c_j \in C} \sum_{o \in E} w(d(o,q)) \cdot \delta(c_j, Trieda(o))$$

return *Trieda*;

- *Trieda(o)* označuje triedu trénovacieho príkladu *o*
- *w(x)* je váhová funkcia, napr. $w(x) = \frac{1}{x^2}$
- *d(x,y)* je funkcia vzdialenosti, napr. euklidovská

$$\delta(x,y) = \begin{cases} 1 \Leftrightarrow x = y \\ 0 \Leftrightarrow x \neq y \end{cases}$$

Vplyv hodnoty *k* na výsledok



- Príliš malé *k* vedie k prílišnej citlivosti na príklady, ktoré predstavujú šumy
- Príliš veľké *k* zase zvyšuje riziko prekročenia hraníc zhluku príkladov reprezentujúcich určitú triedu a zahrnutie mnohých príkladov z inej triedy
- Stredná hodnota *k* preto vo všeobecnosti poskytuje najlepšie výsledky klasifikácie
- Často platí pre hodnotu najlepšieho *k*: *1 << k < 10*

Vyhodnotenie kvality klasifikátorov (1)

- Je potrebné vedieť navzájom porovnať rôzne klasifikátory a zistiť, ktorý z nich je najlepší
- Ako najdôležitejšie kritérium porovnania sa používa ***chyba klasifikácie***, t.j. podiel chybne klasifikovaných objektov.
- Vzniká ale otázka, ktoré dáta sa majú použiť pre odhad chyby klasifikácie (ak použijeme trénovacie dáta – dochádza k preučeniu)
- Metóda ***trénovanie a testovanie*** rozdeľuje množinu príkladov na:
  - ***Trénovaciu množinu***, ktorá sa použije v procese budovania klasifikátora (t.j. pre učenie). Cca. 2/3 príkladov z *O*
  - ***Testovaciu množinu***, ktorá sa používa len na odhad chyby klasifikácie pre získaný klasifikátor. Cca. 1/3 príkladov z *O*

Vyhodnotenie kvality klasifikátorov (2)

- Pri ***m-násobnej krížovej validácii*** sa množina *O* rozdelí na *m* rovnako veľkých podmnožín,
  - Zakaždým sa použije *m–1* podmnožín na trénovanie klasifikátora
  - Zvyšná podmnožina potom následne na jeho testovanie
  - Takto sa získa *m* rôznych chýb klasifikátora, ktoré sa nakoniec skombinujú pre získanie výsledného odhadu chyby klasifikácie
- Ak rozdelenie množiny *O* na podmnožiny nie je náhodné, ale také, aby jednotlivé podmnožiny zachovávali distribúciu jednotlivých tried v každej z podmnožín, ide o tzv. ***rozvrstvenú násobnú krížovú validáciu*** (stratified cross validation)
- Vo všeobecnosti sa najčastejšie doporučuje 10-násobná krížová validácia na odhad kvality klasifikátorov

Algoritmus krížovej validácie

KrížováValidácia (Databáza *O*, Integer *m*, Procedúra *klasifikácia*)
Rozdeľ *O* na *m* podmnožín O_1, ..., O_m pokiaľ možno rovnakej veľkosti;
chyba = 0;
for *i* **from** 1 **to** *m*
klasifikátor$_i$:= klasifikácia($O_1 \cup .. \cup O_{i-1} \cup O_{i+1} \cup O_m$) ;
Nech *chyba*$_i$ je chyba klasifikácie *klasifikátor*$_i$ na O_i;
chyba = *chyba* + *chyba*$_i$;
return *chyba/m*;

Zvyšovanie presnosti klasifikátorov (1)

- Techniky ***bagging*** a ***boosting*** sa snažia využiť množinu *T* klasifikátorov $C_1, C_2, ..., C_T$ v snahe vytvoriť lepší, zložený klasifikátor C_*
- ***Bagging***
  - V každej iterácii *t* (*t* = 1, 2, ..., *T*) sa najprv z množiny *O* vytvorí vzorka O_t (existujú rôzne stratégie výberu)
  - Následne sa množina O_t použije ako trénovacia množina pre získanie klasifikátora C_t
  - Pre klasifikáciu neznámeho príkladu *X* zložený klasifikátor C_* najprv zistí klasifikácie všetkých čiastkových klasifikátorov C_t (*t* = 1, 2, ..., *T*), spočíta hlasy pre jednotlivé triedy
  - Príklad *X* nakoniec klasifikuje do tej triedy, ktorá získala najväčší počet hlasov

Zvyšovanie presnosti klasifikátorov (2)

- ***Boosting***
  - Každému príkladu v trénovacej množine je priradená určitá váha
  - Potom sa trénuje séria klasifikátorov C_t (*t* = 1, 2, ..., *T*) takým spôsobom, že po vygenerovaní každého klasifikátora C_t sa upravia váhy príkladov v trénovacej množine tak, aby sa pri učení nasledujúceho klasifikátora C_{t+1} venovala zvýšená pozornosť predtým chybne klasifikovaným príkladom
  - Výsledný zložený klasifikátor C_* kombinuje hlasy jednotlivých klasifikátorov tak, že váha hlasu každého z klasifikátorov je priamo úmerná jeho presnosti