EVROPSKÁ UNIE
Evropské strukturální a investiční fondy
Operační program Výzkum, vývoj a vzdělávání

MINISTERSTVO ŠKOLSTVÍ, MLÁDEŽE A TĚLOVÝCHOVY

Název projektu	Rozvoj vzdělávání na Slezské univerzitě v Opavě
Registrační číslo projektu	CZ.02.2.69/0.0./0.0/16_015/0002400

Dolování dat

Asociační pravidla

Jan Górecki

SLEZSKÁ UNIVERZITA
OBCHODNĚ PODNIKATELSKÁ FAKULTA V KARVINÉ

Obsah přednášky

SLEZSKÁ UNIVERZITA
OBCHODNĚ PODNIKATELSKÁ FAKULTA V KARVINÉ

- Co jsou Asociační pravidla
- Základní charakteristiky pravidel
- Hledání asociačních pravidel
- Generování kombinací
- Algoritmus apriori

Asociační pravidla

SLEZSKÁ UNIVERZITA
OBCHODNĚ PODNIKATELSKÁ FAKULTA V KARVINÉ

- Úloha hledání souvislostí mezi hodnotami atributů ve tvaru **pravidel**.

Ant $\Rightarrow$ **Suc**,
kde **Ant** (antecedent) i **Suc** (sukcedent) jsou konjunkce hodnot KATEGORIÁLNÍCH atributů (kategorií)

párky & hořčice $\Rightarrow$ rohlíky

- Úloha typu **učení *bez* učitele**
- Cílem je nalézt **všechna pravidla** podpořená daty, která splňují předem stanovená **kritéria**
- **Kritéria** ovlivňují **množství** a **vlastnosti** nalezených pravidel
- Analýza nákupního košíku (Agrawal, 1993)

Základní charakteristiky pravidel

SLEZSKÁ UNIVERZITA
OBCHODNĚ PODNIKATELSKÁ FAKULTA V KARVINÉ

Ant ⇒ Suc

párky & hořčice ⇒ rohlíky

	Suc	¬Suc
Ant	a	b
¬Ant	c	d

podpora (support)

$$sup(Ant \Rightarrow Suc) = P(Ant \wedge Suc) = \frac{a}{a+b+c+d}.$$

spolehlivost (confidence)

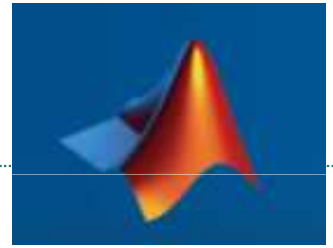
$$conf(Ant \Rightarrow Suc) = P(Suc|Ant) = \frac{P(Suc \wedge Ant)}{P(Ant)} = \frac{a}{a+b}$$

Párek	Hořčice	Rohlíky	Pivo
0	1	1	0
1	1	1	1
1	1	0	1
1	1	1	0

Jak najít všechna pravidla s danou podporou a spolehlivostí?

Kolik kombinací je možno generovat z těchto dat?

Hledání asociačních pravidel

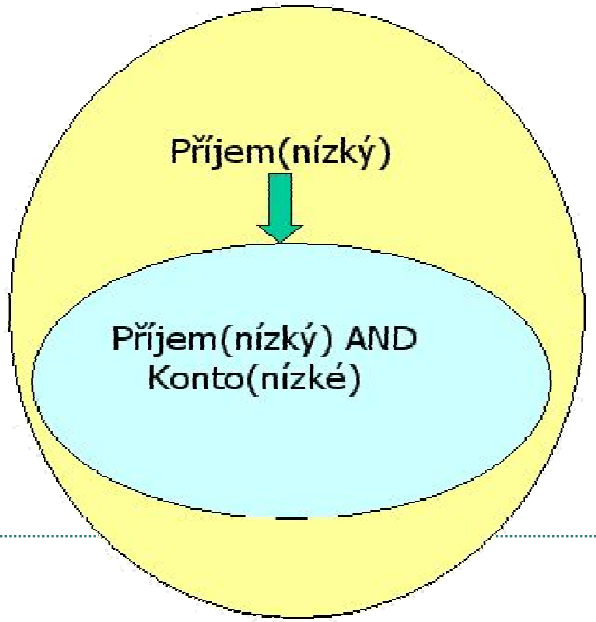


SLEZSKÁ UNIVERZITA
OBCHODNĚ PODNIKATELSKÁ FAKULTA V KARVINÉ

1) **Generování** syntakticky korektního pravidla = prohledávání prostoru pravidel, neboli generování všech přípustných konjunkcí atributů (atribut se nesmí opakovat!)
   a) Do šířky, do hloubky
   b) Heuristické

   (někdy nutná *binarizace*)
2) **Testování** vygenerovaného pravidla = zjišťování (na datech), zda pravidlo splňuje zadané požadavky na hodnoty numerických charakteristik

Konto	Konto(vysoké)	Konto(střední)	Konto(nízké)
Vysoké	1	0	0
Střední	0	1	0
Nízké	0	0	1
Střední	0	1	0



Generování kombinací

SLEZSKÁ UNIVERZITA
OBCHODNĚ PODNIKATELSKÁ FAKULTA V KARVINÉ

- do šířky
- do hloubky
- heuristicky

počet kombinací $= \prod_{j=1}^{m} \left(1 + K_{A_j}\right) - 1,$
kde K_{A_j} je počet hodnot j–tého atributu a
m je maximální délka kombinace

příjem	konto	pohlaví	nezaměstnaný	úvěr
vysoký	vysoké	žena	ne	ano
vysoký	vysoké	muž	ne	ano
nízký	nízké	muž	ne	ne
nízký	vysoké	žena	ano	ano
nízký	vysoké	muž	ano	ano
nízký	nízké	žena	ano	ne
vysoký	nízké	muž	ne	ano
vysoký	nízké	žena	ano	ano
nízký	střední	muž	ano	ne
vysoký	střední	žena	ne	ano
nízký	střední	žena	ano	ne
nízký	střední	muž	ne	ano

kombinace
1n
1v
2n
2s
2v
3m
3z
4a
4n
5a
5n
1n 2n
1n 2s
1n 2v
1n 3m
1n 3z
1n 4a
1n 4n
1n 5a
1n 5n
1v 2n
1v 2s
1v 2v
1v 3m
1v 3z
1v 2v 3z 4n 5a

Do šířky

kombinace
1n
1n 2n
1n 2n 3m
1n 2n 3m 4a
1n 2n 3m 4a 5a
1n 2n 3m 4a 5n
1n 2n 3m 4n
1n 2n 3m 4n 5a
1n 2n 3m 4n 5n
1n 2n 3m 5a
1n 2n 3m 5n
1n 2n 3z
1n 2n 3z 4a
1n 2n 3z 4a 5a
1n 2n 3z 4a 5n
1n 2n 3z 4n
1n 2n 3z 4n 5a
1n 2n 3z 4n 5n
1n 2n 3z 5a
1n 2n 3z 5n
1n 2n 4a
1n 2n 4a 5a
1n 2n 4a 5n
1n 2n 4n
2n
5n

Do hloubky

Frq	kombinace
8	5a
7	1n
6	3m
6	3z
6	4a
6	4n
5	1v
5	1n 4a
5	4n 5a
5	1v 5a
4	2v
4	2s
4	2n
4	5n
4	3m 5a
4	1n 3m
4	3z 5a
4	3z 4a
4	3m 4n
4	1v 4n
4	2v 5a
4	1n 5n
4	1v 4n 5a
3	1n 5a
3	1n 3z
1	1v 2s 3z 4n 5a

Heuristicky

Lidl.cz: „Stálý sortiment zahrnuje přibližně 2 400 druhů výrobků.“ Tesco: 14 000 online

Algoritmus apriori – 1. krok

Geniální myšlenka:
Mám-li kombinaci *Comb* délky *k*, tak pokud její **jakákoli podkombinace** délky *k*-1 nesplňuje minimální podporu, tak ani *Comb* **nemůže** splňovat minimální podporu

1. *do L_1 přiřaď všechny hodnoty atributů, které dosahují alespoň požadované četnosti*
2. polož k=2
3. dokud L_{k-1} je neprázdná:
   a) pomocí **apriori-gen(L_{k-1})** vygeneruj množinu kandidátů C_k
   b) do L_k zařaď ty kombinace z C_k, které dosáhly alespoň požadovanou četnost
   c) proveď $k := k + 1$

Funkce apriori-gen(L_{k-1})

1) pro všechny dvojice kombinací *p*, *q* z L_{k-1}:
   Pokud *p* a *q* se shodují v k-2 kategoriích přidej sjednocení p ∧ q do C_k
2) pro každou kombinaci *c* z C_k:
   Pokud některá z jejich podkombinací délky k-1 není obsažena v L_{k-1} **odstraň** *c* z C_k

Výsledek kombinace *p* a *q* má pak délku *k*

Kurzívou jsou vyznačeny kroky, kde je potřeba sahat na původní data (pomalé)

Algoritmus apriori – příklad

SLEZSKÁ UNIVERZITA
OBCHODNĚ PODNIKATELSKÁ FAKULTA V KARVINÉ

Pro data o klientech banky: n = 12
Minimální podpora: minsup = 1/3 (**4** z 12)
Minimální spolehlivost: minconf = **0.8**

1. krok

L_1 : 5a(8), 1n(7), 3m(6), 3z(6), 4a(6), 4n(6), 1v(5), 2v(4), 2s(4), 2n(4), 5n(4)

C_2 : 5a1n, 5a3m, 5a3z, 5a4a, 5a4n, 5a1v, 5a2v, 5a2s, 5a2n, 1n3m, 1n3z, 1n4a, 1n4n, 1n2v, 1n2s, 1n2n, 1n5n, 3m4a, 3m4n, 3m1v, 3m2v, 3m2s, 3m2n, 3m5n, 3z4a, 3z4n, 3z1v, 3z2v, 3z2s, 3z2n, 3z5n, 4a1v, 4a2v, 4a2s, 4a2n, 4a5n, 4n1v, 4n2v, 4n2s, 4n2n, 4n5n, 1v2v, 1v2s, 1v2n, 1v5n, 2v5n, 2s5n, 2n5n

L_2 : 5a3m(4), 5a4n(5), 5a1v(5), 5a3z(4), 5a2v(4), 1n3m(4), 1n4a(5), 3m4n(4), 3z4a(4), 1n3m(4), 1n5n(4), 1v4n(4)

C_3 : 5a4n1v, 3m4n5a

L_3 : 5a4n1v(4)

1	2	3	4	5
příjem	konto	pohlaví	nezaměstnaný	úvěr
vysoký	vysoké	žena	ne	ano
vysoký	vysoké	muž	ne	ano
nízký	nízké	muž	ne	ne
nízký	vysoké	žena	ano	ano
nízký	vysoké	muž	ano	ano
nízký	nízké	žena	ano	ne
vysoký	nízké	muž	ne	ano
vysoký	nízké	žena	ano	ano
nízký	střední	muž	ano	ne
vysoký	střední	žena	ne	ano
nízký	střední	žena	ano	ne
nízký	střední	muž	ne	ano

Algoritmus apriori – 2. krok (do hry přichází *spolehlivost*)

SLEZSKÁ UNIVERZITA
OBCHODNĚ PODNIKATELSKÁ FAKULTA V KARVINÉ

$$\text{conf}(\text{Ant} \Rightarrow \text{Suc}) = \text{P}(\text{Suc}|\text{Ant}) = \frac{\text{P}(\text{Suc} \wedge \text{Ant})}{\mathbf{P(Ant)}} = \frac{a}{a+b}$$

- Každá kombinace Comb se rozdělí na všechny možné dvojice podkombinací Ant a Suc takové, že Suc = Comb – Ant.
- Hledají se pravidla Ant $\Rightarrow$ Suc tak, že se postupně přesouvají kategorie z Ant do Suc

Je-li Ant' podkombinací Ant (tedy Ant' je kratší než Ant), potom

$$\text{conf}(\text{Ant}' \Rightarrow \text{Comb-Ant}') \leq \text{conf}(\text{Ant} \Rightarrow \text{Comb-Ant})$$

tedy, odebíráním kategorií z předpokladu pravidla snižujeme (přesněji nemůžeme zvýšit) jeho spolehlivost!

Např. když Comb = A1A2A3 a Ant´ = A1, Ant = A1A2, pak:
je-li conf(A1A2 => A3) < *minconf*, pak conf(A1 => A2A3) < *minconf* , tedy pro A1 => A2A3 **není třeba** ověřovat minimální spolehlivost, protože víme, že ji splňovat **nemůže**

Využívají se četnosti kombinací (a a $a+b$) spočtené v kroku 1! (na data se již nesahá)

Algoritmus apriori – příklad

SLEZSKÁ UNIVERZITA
OBCHODNĚ PODNIKATELSKÁ FAKULTA V KARVINÉ

Pro data o klientech banky: n = 12
Minimální podpora: minsup = 1/3 (**4** z 12)
Minimální spolehlivost: minconf = **0.8**

1. krok

L_1 : 5a(8), 1n(7), 3m(6), 3z(6), 4a(6), 4n(6), 1v(5), 2v(4), 2s(4), 2n(4), 5n(4)

C_2 : 5a1n, 5a3m, 5a3z, 5a4a, 5a4n, 5a1v, 5a2v, 5a2s, 5a2n, 1n3m, 1n3z, 1n4a, 1n4n, 1n2v, 1n2s, 1n2n, 1n5n, 3m4a, 3m4n, 3m1v, 3m2v, 3m2s, 3m2n, 3m5n, 3z4a, 3z4n, 3z1v, 3z2v, 3z2s, 3z2n, 3z5n, 4a1v, 4a2v, 4a2s, 4a2n, 4a5n, 4n1v, 4n2v, 4n2s, 4n2n, 4n5n, 1v2v, 1v2s, 1v2n, 1v5n, 2v5n, 2s5n, 2n5n

L_2 : 5a3m(4), 5a4n(5), 5a1v(5), 5a3z(4), 5a2v(4), 1n3m(4), 1n4a(5), 3m4n(4), 3z4a(4), 1n3m(4), 1n5n(4), 1v4n(4)

C_3 : 5a4n1v, 3m4n5a

L_3 : 5a4n1v(4)

2. krok:

1v ⇒ 5a (1)
5n ⇒ 1n (1)
2v ⇒ 5a (1)
1v4n ⇒ 5a (1)
4n ⇒ 5a (0.83)
4a ⇒ 1n (0.83)
1v ⇒ 4n (0.8)
4n5a ⇒1v (0.8)
1v5a ⇒ 4n (0.8)
1v ⇒ 4n5a (0.8)

Algoritmus apriori – příklad (2. krok bez zkratek)

SLEZSKÁ UNIVERZITA
OBCHODNĚ PODNIKATELSKÁ FAKULTA V KARVINÉ

Rule (Support, Confidence)

uver_ne -> prijem_nizky (33.3333%, 100%)

prijem_vysoky -> uver_ano (41.6667%, 100%)

konto_vysoke -> uver_ano (33.3333%, 100%)

prijem_vysoky & nezamestnany_ne -> uver_ano (33.3333%, 100%)

nezamestnany_ano -> prijem_nizky (41.6667%, 83.3333%)

nezamestnany_ne -> uver_ano (41.6667%, 83.3333%)

prijem_vysoky -> nezamestnany_ne (33.3333%, 80%)

prijem_vysoky -> nezamestnany_ne & uver_ano (33.3333%, 80%)

prijem_vysoky & uver_ano -> nezamestnany_ne (33.3333%, 80%)

nezamestnany_ne & uver_ano -> prijem_vysoky (33.3333%, 80%)

Interpretace výsledků

SLEZSKÁ UNIVERZITA
OBCHODNĚ PODNIKATELSKÁ FAKULTA V KARVINÉ

- Je potřeba spolupracovat s experty, jinak hrozí **mylná interpretace** získaných pravidel
- Např. pleny & mléko => pivo (spolehlivost = 80%)

Shrnutí

SLEZSKÁ UNIVERZITA
OBCHODNĚ PODNIKATELSKÁ FAKULTA V KARVINÉ

- Hledání asociačních pravidel je metoda **učení bez učitele** – nevolí se žádný cílový atribut

- První krok algoritmu apriori je založen na faktu, že:
  mám-li kombinaci *Comb* délky *k*, tak pokud její **jakákoli podkombinace** délky *k*-1 nesplňuje minimální podporu, tak ani *Comb* **nemůže** splňovat minimální podporu => výrazné zrychlení prohledávání prostoru kombinací

- Druhý krok algoritmu apriori je založen na faktu, že:
  je-li *Ant'* podkombinací *Ant*, potom
  conf(*Ant'* $\Rightarrow$ *Comb*-*Ant'*) $\leq$ conf(*Ant* $\Rightarrow$ *Comb*-*Ant*) => výrazné zrychlení generování pravidel splňujících minimální podporu

- Aplikace metody zahrnuje např. analýzu nákupního košíku (supermarkety, e-shopy, atd.).

Děkuji za pozornost

Některé snímky převzaty od:

prof. Ing. Petr Berka, CSc. berka@vse.cz