PV211 – Zbierka príkladov

Bc. Dominik Szalai
359775@mail.muni.cz

Michal Krajčovič
422783@mail.muni.cz

19. mája 2015: zanesené opravy reportované M. Šmídem v 2.2 a 2.9

Abstrakt

Tento dokument slúži ako zbierka príkladov k predmetu PV211. Súčasťou zbierky sú aj vypracované riešenia s príslušnými definíciami a odkazmi na literatúru. Pretože vypracované príklady môžu obsahovať chyby, tak je doporučená účasť na prednáškach a cvičeniach. Primárne má slúžiť ako náhrada pre študentov, ktorý sa z nejakých dôvodov cvičení nemohli zúčastniť a chýbajú im zápisky. Všetky definície použité v dokumente sú prebraté z [2] ak nie je uvedené inakšie.

Obsah

1 Prvé cvičenie **3**
2. príklad . . . 4
3. príklad . . . 4
4. príklad . . . 5
5. príklad . . . 6
6. príklad . . . 7
7. príklad . . . 9
8. príklad . . . 9
9. príklad . . . 9
10. príklad . . . 10

2 Druhé cvičenie **11**
1. príklad . . . 12
2. príklad . . . 12
3. príklad . . . 13
4. príklad . . . 14
5. príklad . . . 14
6. príklad . . . 15
7. príklad . . . 16
8. príklad . . . 17

3 Tretie cvičenie **19**
1. príklad . . . 20
2. príklad . . . 20
3. príklad . . . 22
4. príklad . . . 24

5. príklad 25
6. príklad 25

4 Štvrté cvičenie **26**
1. príklad 27
2. príklad 27
3. príklad 28
4. príklad 29

5 Piate cvičenie **30**
1. príklad 31
2. príklad 33
3. príklad 33
4. príklad 34
5. príklad 34

1 Prvé cvičenie

Algoritmus 1 (Algoritmus pre výpočet Levenhsteinovej vzdialenosti)

```
1:  function LevenshteinDistance(s1, s2)
2:      for i = 0 to |s1| do
3:          m[i, 0] = i
4:      end for
5:      for j = 0 to |s2| do
6:          m[0, j] = j
7:      end for
8:      for i = 1 to |s1| do
9:          for i = 1 to |s1| do
10:             if s1[i] == s2[j] then
11:                 m[i, j] = min{m[i − 1, j] + 1, m[i, j − 1] + 1, m[i − 1, j − 1]}
12:             else
13:                 m[i, j] = min{m[i − 1, j] + 1, m[i, j − 1] + 1, m[i − 1, j − 1] + 1}
14:             end if
15:         end for
16:     end for
17:     return m[|s1|, |s2|]
18: end function
```

Algoritmus 2 (Výpočet Soundex kódu)

1. *prvý znak ostane*
2. *znaky z množiny* $\{A, E, I, O, U, H, W, Y\}$ *prepíšeme na 0*
3. *prepíšeme znaky*
   (a) $\{B, F, P, V\}$ *na 1*
   (b) $\{C, G, J, K, Q, S, X, Z\}$ *na 2*
   (c) $\{D, T\}$ *na 3*
   (d) $\{L\}$ *na 4*
   (e) $\{M, N\}$ *na 5*
   (f) $\{R\}$ *na 6*
4. *opakovane z každého páru odstránime jednu hodnotu.*
5. *odstránime všetky 0 z výsledku.*
6. *ak je dĺžka výsledku menšia ako 4, tak doplníme zvyšok nulami zprava.*

Algoritmus 3 (Dotazovanie v permuterm indexe)

Postup je prevzatý z [5, sl. 26]. Pre dotaz q nájdeme kľúče podľa následujúcej schémy:

- *pre* $q = X$ *hladaj kľúče v tvare* $X\$$
- *pre* $q = X^*$ *hladaj kľúče v tvare* $\$X^*$
- *pre* $q = {}^*X$ *hladaj kľúče v tvare* $X\$^*$
- *pre* $q = {}^*X^*$ *hladaj kľúče v tvare* X^*
- *pre* $q = X^*Y$ *hladaj kľúče v tvare* $Y\$X^*$

2. príklad

Vytvorte invertovaný index, zostavený pre následujúcu kolekciu dokumentov:

Doc 1: new home sales top forecasts

Doc 2: home sales rise in july

Doc 3: increase in home sales in july

Doc 4: july new home sales rise

Postup je pomerne jednoduchý. Vezmeme každý term (slovo) z dokumentu a vložíme ako kľúč do tabuľky spolu s ID dokumentom. Ak sa kľúč už v tabuľke nachádza vložíme pre tento kľúč iba ID dokumentu. Týmto postupom dostaneme invertovaný index reprezentovaný tabuľkou 1.

new	1	4		
home	1	2	3	4
sales	1	2	3	4
top	1			
forecasts	1			
rise	2	4		
in	2	3		
july	2	3	4	
increase	3			

Tabuľka 1: Invertovaný index

3. príklad

Nižšie je časť indexu s pozíciami v tvare term: doc1: $\langle pos1, pos2, pos3, \ldots \rangle$; doc2: $\langle pos1, pos2, \ldots \rangle$

- angels: $2 : \langle 36, 174, 252, 651 \rangle$; $4 : \langle 12, 22, 102, 432 \rangle$; $7 : \langle 17 \rangle$;
- fools: $2 : \langle 1, 17, 74, 222 \rangle$; $4 : \langle 8, 78, 108, 458 \rangle$; $7 : \langle 3, 13, 23, 193 \rangle$;
- fear: $2 : \langle 87, 704, 722, 901 \rangle$; $4 : \langle 13, 43, 113, 433 \rangle$; $7 : \langle 18, 328, 528 \rangle$;
- in: $2 : \langle 3, 37, 76, 444, 851 \rangle$; $4 : \langle 10, 20, 110, 470, 500 \rangle$; $7 : \langle 5, 15, 25, 195 \rangle$;
- rush: $2 : \langle 2, 66, 194, 321, 702 \rangle$; $4 : \langle 9, 69, 149, 429, 569 \rangle$; $7 : \langle 4, 14, 404 \rangle$;
- to: $2 : \langle 47, 86, 234, 999 \rangle$; $4 : \langle 14, 24, 774, 944 \rangle$; $7 : \langle 199, 319, 599, 709 \rangle$;
- tread: $2 : \langle 57, 94, 333 \rangle$; $4 : \langle 15, 35, 155 \rangle$; $7 : \langle 20, 320 \rangle$;
- where: $2 : \langle 67, 124, 393, 1001 \rangle$; $4 : \langle 11, 41, 101, 421, 431 \rangle$; $7 : \langle 15, 35, 735 \rangle$;

Ktoré dokumenty zodpovedajú následujúcim dotazom a na ktorých pozíciach, kde každý výraz v úvodzovkách je frázový dotaz (phrase query)?

a) „fools rush in“

b) „fools rush in“ AND „angels fear to tread“.

c) v uvedenom indexe je chyba, kde?

K tomu, aby bol dotaz nájdený je potrebné, aby boli slová za sebou. T.j. ak je slovo *angels* v dokumente 1 na tretej pozícii tak slovo *fear* musí byť v tom istom dokumente na štvrtej pozícii. Pre zadanie **a)** si vypočítame všetky možné pozície kde sa táto fráza môže nachádzať.

Slovo *fools* začína na v dokumente 1 na pozíciach $\langle 1, 17, 74, 222\rangle$. To znamená, že slovo *rush* sa musí nachádzať na pozíciach $\langle 2, 18, 75, 223\rangle$ a slovo in je potom na pozíciach $\langle 13, 19, 76, 224\rangle$. Rovnaký postup aplikujeme na dokument 4 a 7 čím dostaneme následovné výskyty.

doc2 $\langle 1, 2, 3\rangle$, $\langle 17, 18, 19\rangle$, $\langle 74, 75, 76\rangle$, $\langle 222, 223, 224\rangle$

doc4 $\langle 8, 9, 10\rangle$, $\langle 78, 79, 80\rangle$, $\langle 108, 109, 110\rangle$, $\langle 458, 459, 460\rangle$

doc7 $\langle 3, 4, 5\rangle$, $\langle 13, 14, 15\rangle$, $\langle 23, 24, 25\rangle$, $\langle 193, 194, 195\rangle$

Teraz sa pozrieme na pôvodný pozičný index a hľadáme či sa nájde zhoda medzi požadovanými pozíciami a reálnymi. Vezmime si dokument 2 a overíme, či sú slová *fools rush in* za sebou na pozíciach $\langle 1, 2, 3\rangle$. Pretože odpoveď je áno, tak systém vráti dokument 2 ako relevantný nášmu dotazu. Rovnakou analógiou overíme zvyšné pozície a dokumenty čím dostaneme výsledok **doc2**: $\{\langle 1, 2, 3\rangle\}$; **doc4**: $\{\langle 8, 9, 10\rangle\}$; a **doc7**: $\{\langle 3, 4, 5\rangle, \langle 13, 14, 15\rangle\}$. Pre zadanie **b)** urobíme totožným postupom možné pozície pre termy *angels fear to tread*. Druhú časť dotazu už máme z riešenia **a)**.

doc2 $\langle 36, 37, 38, 39\rangle$, $\langle 174, 175, 176, 177\rangle$, $\langle 252, 253, 254, 255\rangle$, $\langle 651, 652, 653, 654\rangle$

doc4 $\langle 12, 13, 14, 15\rangle$, $\langle 22, 23, 24, 25\rangle$, $\langle 102, 103, 104, 105\rangle$, $\langle 432, 433, 434, 435\rangle$

doc7 $\langle 17, 18, 19, 20\rangle$

Teraz opäť overíme, či na daných miestach sú v pozičnom indexe zhody. Výsledkom je iba **doc4**: $\{\langle 12, 13, 14, 15\rangle\}$. Takže ak sprienikujeme riešenie **a)** s riešením **b)** dostaneme $\{$**doc2**, **doc4**, **doc7**$\} \cap \{$**doc4**$\} =$ **doc4**. Pre zadanie **c)** sa treba pozrieť do dokumentu 7, ktorý ma na pozícii 15 termy *in* a *where*.

4. príklad

Odporučte stratégiu spracovania dotazu (*tangerine* OR *trees*) AND (*marmalade* OR *skies*) AND (*kaleidoscope* OR *eyes*) vzhľadom na nasledujúce veľkosti postings zoznamov:

eyes 213312

kaleidoscope 87009

marmalade 107913

skies 271658

tangerine 46653

trees 316812

V príklade využijeme rovnaký trik ako z databáz, kde sa snažíme vždy filtrovať výsledky klauzulou `where` najprv s najnižším výskytom. *or* budeme chápať ako sčítanie a *and* ako násobenie. Zostavíme si rovnice:

$$\begin{array}{rcccrcl} tangerine \vee trees & = & x & \Rightarrow & 46653 + 316812 & = & 363465 \\ marmalade \vee skies & = & y & \Rightarrow & 107913 + 271658 & = & 379571 \\ kaleidoscope \vee eyes & = & z & \Rightarrow & 87009 + 213312 & = & 300321 \end{array}$$

V poslednom kroku usporiadame x, y, z podľa veľkostí, čím dostaneme usporiadanie $(kaleidoscope < eyes) < (tangerine < trees) < (marmalade < skies)$.

5. príklad

Máme dotaz zložený z dvoch výrazov. Postings zoznam jedného výrazu je zložený z nasledujúcich 16 položiek: $P = [4, 6, 10, 12, 14, 16, 18, 20, 22, 32, 47, 81, 120, 122, 157, 180]$ a druhý výraz ma postings zoznam len jednoprvkový: $R = [47]$. Zistite koľko porovnaní a prečo je potreba vykonať na prienik týchto dvoch postings zoznamov s nasledujúcimi stratégiami:

a) použitie štandardných postings zoznamov

b) použitie postings zoznamov uložených s preskakujúcimi odkazmi, s dĺžkou preskoku $\sqrt{|P|}$

Naivný algoritmus porovná každý prvok x z množiny A s prvkami y z množiny B.

$$\underbrace{4 \times 47, 6 \times 47, 10 \times 47, \cdots, 47 \times 47}_{11\times}$$

To znamená, že na nájdenie zhody **a)** je potrebných 11 porovnaní. Ak implementujeme preskoky o veľkosti $\sqrt{|P|}$ kde $|P| = 16$ čím dostaneme $\sqrt{|P|} = \sqrt{16} = 4$, znížime počet porovnaní. Miesto toho aby sme mali ukazateľ iba na ďalší prvok, tak máme ešte ukazateľ na $(i + \sqrt{|P|})$. prvok. Pre zadanie **b)** vykoná algoritmus následovné kroky, pričom začíname na pozícii 0:

1. porovná 47×4. Pretože $4 < 47$ tak porovná prvok na $i + 4$ pozícii ak je menší než číslo, s ktorým porovnávame tak urobíme skok. Pretože P[4]=14 < R[0]=47,
2. skočí na pozíciu $0 + 4$,
3. pretože číslo na našej pozícii $P[4] = 14$ nie je požadované, pokúsi sa urobiť skok na $i + 4$ pozíciu a keď je opäť menšie tak preskočí. Pretože P[8]=22 < R[0] = 47,
4. skočí na pozíciu $4 + 4$,
5. pretože číslo na našej pozícii P[8]=22 nie je požadované, pokúsi sa skočiť. Pretože P[12]=120 > R[0] = 47 nemôže urobiť skok a je nútený ísť po jednom takže,

6. porovná 32×47,

7. porovná 47×47 čo je zhoda a končí.

Celkovo teda došlo k

$$\underbrace{4 \times 47,\ 14 \times 47,\ 22 \times 47,\ 120 \times 47,\ 32 \times 47,\ 47 \times 47}_{6}$$

porovnaniam.

6. príklad

Pomocou matice vypočítajte Levenshteinovu vzdialenosť slov jablko a malina.

Vzdialenosť budeme počítať pomocou tabuľky, do ktorej vložíme slová, pred ktoré vložíme prázdne slovo ε.

	ε	m	a	l	i	n	a
ε							
j							
a							
b							
l							
k							
o							

Tabuľka 2: Inicializácia matice

V ďalšom kroku vyplníme prvý riadok a prvý stĺpec hodnotami $0, 1, 2, \ldots, 5, 6$ podľa riadkov 2-7 v algoritme 1.

	ε	m	a	l	i	n	a
ε	0	1	2	3	4	5	6
j	1						
a	2						
b	3						
l	4						
k	5						
o	6						

Tabuľka 3: Matica po aplikovaní riadkov 2-7 v algoritme.

Čísla na týchto pozíciach znamenajú v koľkých znakoch sa líši slovo vľavo od slova hore, pričom začíname vždy s určitým prefixom. Prefixy ε a ε sa líšia v 0 znakoch.[1] Potom porovnáme ε a m čo je rozdiel 1, potom ε a ma čo je 2. Takto pokračujeme až na koniec

[1] Viď definícia IA005, IB102

riadku / stĺpca. Preskočíme generovanie druhého riadku a ukážeme si výpočet na treťom pomocou tabuľky:

	ε	m	a	l	i	n	a
ε	0	1	2	3	4	5	6
j	1	1	2	3	4	5	6
a	2						
b	3						
l	4						
k	5						
o	6						

Tabuľka 4: Matica po iterácii pre $i = 1$.

Algoritmus funguje slovami následovne: Ak sa znaky na danej pozícii rovnajú tak zapíšeme číslo podľa podmienky v algoritme 1 na riadku 11. V opačnom prípade vyberieme minimum z $min(\{A[i-1][j]+1,\ A[i][j-1]+1,\ A[i-1][j-1]+1\})$. Pretože $a \neq m$ vyberieme $min(\{2+1, 1+1, 1+1\}) = min(\{3, 2, 2\})$ a dostaneme tabuľku

	ε	m	a	l	i	n	a
ε	0	1	2	3	4	5	6
j	1	1	2	3	4	5	6
a	2	2					
b	3						
l	4						
k	5						
o	6						

Tabuľka 5: Matica v stave $i = 2$, $j = 1$.

Pretože teraz porovnávame $a = a$ volím číslo podľa prvej podmienky a hodnotu do tabuľky zapíšeme

	ε	m	a	l	i	n	a
ε	0	1	2	3	4	5	6
j	1	1	2	3	4	5	6
a	2	2	1				
b	3						
l	4						
k	5						
o	6						

Tabuľka 6: Matica v stave $i = 2$, $j = 2$.

Porovnáme $a \neq l$ čo je druhé pravidlo a vyberáme $min(\{2+1, 3+1, 1+1\}) = 2$ a zapíšeme. Takto pokračujeme až do úplného vyplnenia tabuľky t.j.:

	ε	m	a	l	i	n	a
ε	0	1	2	3	4	5	6
j	1	1	2	3	4	5	6
a	2	2	1	2	3	4	5
b	3	3	2	2	3	4	5
l	4	4	3	2	3	4	5
k	5	5	4	3	3	4	5
o	6	6	5	4	4	4	**5**

Tabuľka 7: Hotový výpočet vzdialenosti.

7. príklad

a) Nájdite dve rozdielne napísané podstatné mená (anglicky), ktoré majú rovnaký soundex kód.

b) Nájdite dve foneticky podobné podstatné mená (anglicky), ktoré majú rozdielny soundex kód

---

Pre **a)** napríklad *Sword* a *Short*, kedže podľa algoritmu 2 dostaneme *Sword* = S063 a pre *Short* = S063. Pre **b)** to môže byť napr *Rubin* = R150 a *Rupert* = R163.

8. príklad

Vypíšte prvky slovníka permuterm indexu ktoré su generované slovom mama.

Term sa do permuterm indexu zapisuje následovným spôsobom. Term t zreťazíme s ukončovacím symbolom napr \$. Potom spravím rotáciu slova tak, že prvý znak zapíšeme za ukončovací znak. Iným spôsobom povedané, vypočítame permutácie termu. Vstupný term teda bude vyzerať ako *mama\$*.

0. rotácia mama\$

1. rotácia ama\$m

2. rotácia ma\$ma

3. rotácia a\$mam

4. rotácia \$mama

9. príklad

Aké kľúče sú použiteľné na nájdenie termu s*ng v permuterm wildcard indexe.

K riešeniu príkladu použijeme algoritmus 3. Pretože $q = s^*ng$ čo je tvar $q = X^*Y$, tak aplikujeme posledné pravidlo čím dostaneme $ng\$s^*$.

10. príklad

Pre $n = 2$ a $1 \leq T \leq 30$, vykonajte krok za krokom simuláciu algoritmu 4.7 [2, str. 79]. Vytvorte tabuľku, ktorá pre každý okamih v čase, v ktorom je spracovaných $T = 2k$ tokenov ($1 \leq k \leq 15$), ukazuje ktoré zo štyroch indexov $I_0, \ldots, I_3$ sú používané. Prvé tri riadky tabuľky sú uvedené nižšie:

	I_3	I_2	I_1	I_0
2	0	0	0	0
4	0	0	0	1
6	0	0	1	0

Tabuľka 8: Prvé 3 riadky riešenia.

Keďže každý index je 2-krát väčší ako ten predošlý, po preskúmaní algoritmu si môžeme uvedomiť, že sa budú postupne do seba „prelievať“. Celá tabuľka bude teda vyzerať takto:

	I_3	I_2	I_1	I_0
2	0	0	0	1[2]
4	0	0	1	0
6	0	0	1	1[3]
8	0	1	0	0
10	0	1	0	1
12	0	1	1	0
14	0	1	1	1
16	1	0	0	0
18	1	0	0	1
20	1	0	1	0
22	1	0	1	1
24	1	1	0	0
26	1	1	0	1
28	1	1	1	0
30	1	1	1	1

Tabuľka 9: Riešenie príkladu.

[2] Tu si myslíme, že je chyba v zadaní. Tabuľka v podstate reprezentuje binárne čísla.
[3] Viď vyššie.

2 Druhé cvičenie

Definícia 1 (Zipfov zákon)
Zipfov zákon hovorí, že i-te najfrekventovanejší term ma frekvenciu $\frac{1}{i}$. *V príklade použijeme vzťah Zipfovho zákonu* $cf_i \propto \frac{1}{i} = ci^k$, *kde* cf_i *je počet výskytov termu* t_i *v kolekcii pre* $k = -1$.

Definícia 2 (Unárny kód)
Unárny kód, zvaný taktiež α *kód je typ kódovania kde číslo n je reprezentované n jednotkami ukončených nulou.*

Definícia 3 (γ kód)
γ *kód je typ kódovania, ktorý sa skladá z dĺžky a offsetu. Offsetom sa rozumie binárna reprezentácia čísla n bez najvyššieho bitu (1). Dĺžku potom tvorí dĺžka offsetu zakódovaná pomocou* α *kódu.*

Definícia 4 (δ kód)
δ *kód sa kóduje následovným spôsobom. Pre číslo n vypočítame offset čísla n a dĺžku n zakódujeme* γ *kódom. Za* γ *kód zapíšeme ešte hodnotu offsetu čísla n. Celkový tvar je teda* $\delta(n) = \gamma(|\beta(n)|), off(n)$

Algoritmus 4 (Variabilný byte kód)
Číslo n prevedieme na variabilný byte kód následovne [1]:

1. *prevedieme číslo do binárnej podoby,*
2. *binárne číslo rozdelíme po 7 bitoch, ak je skupina menšia ako 7 bitov doplníme nuly zľava na 7 bitov,*
3. *sedmicu doplníme o 0, alebo 1 ak ide o prvú sedmicu z prava.*

Definícia 5 (Inverse document frequency)
Definujeme inverse document frequency *termu* t *ako*

$$idf_t = \log\left(\frac{N}{df_t}\right)$$

kde df_t *je* document frequency *termu* t. *Výhodou logaritmického váženia je fakt, že term, ktorý sa vyskytuje často bude naberať pre logaritmus malé hodnoty, zatiaľ čo term, ktorý sa často nevyskytuje bude naberať vysoké hodnoty a bude mať väčšiu váhu.*

Definícia 6
Váhu termu t v dokumente d vypočítame ako

$$wf_{t,d} = \begin{cases} 1 + \log\left(tf_{t,d}\right) & ak\ n > 0 \\ 0 & inakšie \end{cases}$$

kde $tf_{t,d}$ *je počet výskytov termu t v dokumente d.*

Definícia 7
tf-idf *schéma váženia priradí termu* t *váhu v dokumente* d *ako*

$$tf\text{-}idf_{t,d} = tf_{t,d} \times idf_t$$

Definícia 8 (Normalizácia vektoru)

$$v = \frac{v}{||v||} = \frac{v_j}{\sqrt{\sum_{k=1}^{|v|} {v_k}^2}}$$

kde v_j a v_k je položka vektoru na j resp. k-tom mieste.

1. príklad

Kolekcia dokumentov obsahuje 4 slová: a, b, c, d. Vzájomná frekvencia slov je a > b > c > d. Celkový počet tokenov v kolekcii je 5000. Predpokladajte, že pre túto kolekciu presne platí Zipfov zákon. Aké sú frekvencie vyššie uvedených štyroch slov?

K riešeniu príkladu je potrebný Zipfov zákon popísaný v definícii 1. Najmenej frekventovanejší term je d, potom c, b a najfrekventovanejší je a. Dosadením do vzťahu pre Zipfov zákon dostávame

$$\begin{aligned} cf_1 + cf_2 + cf_3 + cf_4 &= 5000 \\ c.1^{-1} + c2^{-1} + c3^{-1} + c4^{-1} &= 500 \\ c + \tfrac{1}{2}c + \tfrac{1}{3}c + \tfrac{1}{4}c &= 5000 \\ 12c + 6c + 4c + 3c &= 60000 \\ 25c &= 60000 \\ c &= 2400 \end{aligned}$$

Dosadením do vzťahu $cf_i = ci^k$ dostaneme hodnoty:

$$\begin{aligned} cf_1 &= 2400\tfrac{1}{1} = 2400 \\ cf_2 &= 2400\tfrac{1}{2} = 1200 \\ cf_3 &= 2400\tfrac{1}{3} = 800 \\ cf_4 &= 2400\tfrac{1}{4} = 600 \end{aligned}$$

2. príklad

Vypočítajte γ a δ kódy pre 1, 2, 3, 4, 31, 63, 127, 1023.

Demonštratívne vypočítame hodnoty pre čísla 63 a 1023. Podľa definície 3 je potrebné vypočítať offset čísla ako jeho binárnu reprezentáciu bez najvyššieho bitu. $63_{10} = 111111_2$ bez najvyššieho bitu dostaneme $off(63) = 11111$. Dĺžku offsetu podľa definície vypočítame ako jeho veľkosť zakódovaná α kódom. $|11111| = 5 \rightsquigarrow \alpha(5) = 111110$. Keďže poznáme offset, aj jeho dĺžku tak $\gamma(63) = 111110, 11111$. To isté urobíme pre číslo 1023. $1023_{10} = 1111111111_2$, z čoho plynie offset 111111111, ktorého veľkosť je $|111111111| = 9 \rightsquigarrow \alpha(9) = 1111111110$. Potom $\delta(1023) = 1111111110, 111111111$. Výpočet δ kódu je komplikovanejší. V prvom kroku vypočítame offset$(n+1)$, čo je offset$(64) = 000000$. Dĺžka offsetu je potom $\alpha(|000000|) = 6$. Hodnotu 6 teraz zakódujeme γ kódom takže $\gamma(6) = 11010$. Z definície 4 potom $\delta(63) = 11010, 000000$. Pre 1023 vypočítame offset$(1024) = 0000000000$, ktorého

dĺžka je 10. Vypočítame $\gamma(10) = 110010$ a spojíme s offsetom čím dostaneme $\delta(1023) = 1110010, 0000000000$.

číslo	binárne	offset	|offset|	α	γ	δ
2	10	0	1	10	10,0	0,1,0
3	11	1	1	10	10,1	0,1,1
4	100	00	2	110	110,00	01,0,00
19	10011	0011	4	11110	11110,0011	001,00,0011
31	11111	1111	4	11110	11110,1111	001,00,1111
63	111111	11111	5	111110	111110,11111	001,01,11111
127	1111111	111111	6	1111110	1111110,111111	001,10,111111
1023	1111111111	111111111	9	1111111110	1111111110,111111111	0001,001,111111111

Tabuľka 10: γ kódy

3. príklad

Vypočítejte variabilní byte a γ kód pro postings seznam $\langle 777, 17\,743, 294\,068, 31\,251\,336\rangle$. Používejte medzery místo docID tam, kde je to možné. Binární kódy napište v 8bitových blocích.

Zadání od nás požaduje použití mezer místo docID. Mezeru vypočítáme podle [6, sl. 39] jako

$$\forall j > k \;\; docID_j = docID_j - docID_k$$

kde k je docID prvního dokumentu. To znamená, že pro docID_{17743} použijeme mezeru $17743 - 777 = 16966$, pro docID_{294068} dostaneme $294068 - 17743 = 276325$, pro $\text{docID}_{31251336}$ dostaneme $31251336 - 294068 = 30957268$.

- VB(777) = 00000011 00001001 = **0**000 0110 **1**000 1001
- VB(16 966) = 01000010 01000110 = **0**000 0001 **0**000 0100 **1**100 0110
- VB(276 325)= 00000100 00110111 01100101 = **0**001 0000 **0**110 1110 **1**110 0101
- VB(30 957 268)=00000001 11011000 01011110 11010100 = **0**000 1110 **0**110 0001 **0**011 1101 **1**101 0100
- $\gamma(777 = 2^9 + 265)$= 11 1111 1110,1 0000 1001
- $\gamma(16\,966 = 2^{14} + 582)$ = 111 1111 1111 1110,00 0010 0100 0110
- $\gamma(276\,325 = 2^{18} + 14181)$ = 111 1111 1111 1111 1110,00 0011 0111 0110 0101
- $\gamma(30\,957\,268 = 2^{24} + 14180052)$ = 1 1111 1111 1111 1111 1111 1110,00 0011 0111 0110 0101

Pro γ-kód byl využit postup popsaný na Wikipedii `http://en.wikipedia.org/wiki/Elias_gamma_coding`.

4. príklad

Posúďte tabuľku s frekvenciami slov troch dokumentov Doc1, Doc2, Doc3 nižšie. Vypočítajte tf-idf váhy termov *car, auto, insurance, best* pre každý dokument. Idf hodnoty termov sú uvedené v tabuľke.

	Doc1	**Doc2**	**Doc3**	**idf**
car	27	4	24	1.65
auto	3	33	0	2.08
insurance	0	33	29	1.62
best	14	0	17	1.5

Tabuľka 11: Zadanie príkladu.

Po vypočítaní tf-idf váh podľa definície 7 pre každý term zvlášť dostaneme nasledujúcu tabulku

	tf-idf		
	Doc1	**Doc2**	**Doc3**
car	44.55	6.6	39.6
auto	6.24	68.64	0
insurance	0	53.46	46.98
best	21	0	25.5

Tabuľka 12: Riešenie príkladu.

5. príklad

Vypočítajte normalizované Euklidovské vektory pre každý dokument z predchádzajúceho príkladu, kde každý vektor má štyri komponenty, jednu pre každý zo štyroch termov.

Normalizované Euklidovské vektory vypočítame podľa definície 8. Menovatele pre jednotlivé dokumenty budú vyzerať následovne:

$$m_{d1} = \sqrt{44.55^2 + 6.24^2 + 21^2} = 49.6451$$
$$m_{d2} = \sqrt{6.6^2 + 68.64^2 + 53.46^2} = 87.2524$$
$$m_{d3} = \sqrt{39.6^2 + 46.98^2 + 25.5^2} = 66.5247$$

Po dosadení dostávame:

$$Doc_1 = \left(\frac{44.55}{49.6451}; \frac{6.24}{49.6451}; \frac{0}{49.6451}; \frac{21}{49.6451}\right) = (0.8974; 0.1257; 0; 0.423)$$

$$Doc_2 = \left(\frac{6.6}{87.2524}; \frac{68.64}{87.2524}; \frac{53.46}{87.2524}; \frac{0}{87.2524}\right) = (0.0756; 0.7876; 0.6127; 0)$$

$$Doc_3 = \left(\frac{39.6}{66.5247}; \frac{0}{66.5247}; \frac{46.98}{66.5247}; \frac{25.5}{66.5247}\right) = (0.5953; 0; 0.7062; 0.3833)$$

6. príklad

S váhami slov ako boli vypočítané v predchádzajúcom príklade, oznámkujte tri dokumenty podľa vypočítaného skóre pre dotaz „car insurance“, pre každý z nasledujúcich prípadov váženia slov:

a) váha termu je 1 ak sa v dotaze nachádza, inak 0

b) Euclidovské normalizované idf

$$Q = car\ insurance = (1, 0, 1, 0)$$

	Query	**idf**	**tf-idf**
car	1	1.65	1.65
auto	0	2.08	0
insurance	1	1.62	1.62
best	0	1.5	0

Tabuľka 13: Dotaz s váhami.

Bod **a)** vypočítame ako skalárny súčin vektorov $Q \times Doc_n$, kde Doc_n je normalizovaný vektor pre dokument z predošlého príkladu. Tým dostávame:

$$Q \times Doc_1 = 1 \times 0.8974 + 0 \times 0.1257 + 1 \times 0 + 0 \times 0.423 = 0.8974$$
$$Q \times Doc_2 = 1 \times 0.0756 + 0 \times 0.7876 + 1 \times 0.6127 + 0 \times 0 = 0.6883$$
$$Q \times Doc_3 = 1 \times 0.5953 + 0 \times 0 + 1 \times 0.7062 + 0 \times 0.3833 = 1.3015$$

Pre bod **b)** potrebujeme najprv normalizovaný vektor Q_w, ktorý získame vydelením každej zložky z Q dĺžkou vektora idf.

$$Q_w = \left(\frac{1.65}{\sqrt{1.65^2 + 2.08^2 + 1.62^2 + 1.5^2}}; 0; \frac{1.62}{\sqrt{1.65^2 + 2.08^2 + 1.62^2 + 1.5^2}}; 0 \right) =$$
$$= \left(\frac{1.65}{3.45301}; 0; \frac{1.62}{3.45301}; 0 \right) = (0.4778; 0; 0.4692; 0)$$

Následne stačí vektor Q_w vynásobiť s vektormi dokumentov.

$$Q_w \times Doc_1 = 0.4778 \times 0.8974 + 0 \times 0.1257 + 0.4692 \times 0 + 0 \times 0.423 = 0.4288$$
$$Q_w \times Doc_2 = 0.4778 \times 0.0756 + 0 \times 0.7876 + 0.4692 \times 0.6127 + 0 \times 0 = 0.3236$$
$$Q_w \times Doc_3 = 0.4778 \times 0.5953 + 0 \times 0 + 0.4692 \times 0.7062 + 0 \times 0.3833 = 0.6158$$

7. príklad

Vypočítajte vektor-space podobnosť medzi dotazom „digital cameras“ a dokumentom „digital cameras and video cameras“ doplnením prázdných stĺpcov v tabuľke nižšie. Predpokladajte $N = 10000000$, logaritmické váženie termov (stĺpce wf) pre dotaz aj dokumenty, idf váženie len pre dotaz a kosínovú normalizáciu len pre dokument. „And“ považujte za STOP slovo. Napíšte počty termov do tf stĺpca. Aké je konečné skóre podobnosti?

		Query				Document			product
	df	tf	wf	idf	$q_i = wf\text{-}idf$	tf	wf	d_i=normalized wf	$q_i \times d_i$
digital	10 000								
video	100 000								
cameras	50 000								

Tabuľka 14: Zadanie príkladu.

Hodnotu tf vyplníme podľa výskytu termov v dotaze a dokumente.

$$\begin{aligned} tf_q &= digital\ cameras & &= (1,0,1) \\ tf_d &= digital\ cameras\ and\ video\ cameras & &= (1,1,2) \end{aligned}$$

Pre logaritmické váženie použijeme definíciu 6. Pre dotaz budu tieto hodnoty vyzerať následovne:

$$\begin{aligned} wf_{digital} &= 1 + \log(1) &= 1 + 0 &= 1 \\ wf_{video} &= 0 \\ wf_{cameras} &= 1 + \log(1) &= 1 + 0 &= 1 \end{aligned}$$

Pre dokument:

$$\begin{aligned} wf_{digital} &= 1 + \log(1) &= 1 + 0 &= 1 \\ wf_{video} &= 1 + \log(1) &= 1 + 0 &= 1 \\ wf_{cameras} &= 1 + \log(2) &= 1 + 0.301 &= 1.301 \end{aligned}$$

Ďalej potrebujeme pre dotaz vypočítať idf váhy, ktoré vypočítame pomocou definície 5.

$$\begin{aligned} idf_{digital} &= \log\left(\tfrac{10^7}{10^4}\right) &= \log\left(10^3\right) &= 3 \\ idf_{video} &= \log\left(\tfrac{10^7}{10^5}\right) &= \log\left(10^2\right) &= 2 \\ idf_{cameras} &= \log\left(\tfrac{10^7}{5\times 10^4}\right) &= \log(200) &= 2.301 \end{aligned}$$

Kosínovú normalizáciu pre dokument vypočítame rovnako ako v predošlých príkladoch pomocou 8 s použitím wf.

$$\begin{aligned} d_{digital} &= \tfrac{1}{\sqrt{1^2+1^2+1.301^2}} &= 0.5204 \\ d_{video} &= \tfrac{1}{\sqrt{1^2+1^2+1.301^2}} &= 0.5204 \\ d_{cameras} &= \tfrac{1.301}{\sqrt{1^2+1^2+1.301^2}} &= 0.677 \end{aligned}$$

Po dosadení a dopočítaní $q_i = wf \times idf$ a skóre do tabuľky dostávame:

		Query				Document			product
	df	tf	wf	idf	$q_i = wf\text{-}idf$	tf	wf	d_i=normalized wf	$q_i \times d_i$
digital	10 000	1	1	3	3	1	1	0.5204	1.5612
video	100 000	0	0	2	0	1	1	0.5204	0
cameras	50 000	1	1	2.301	2.301	2	1.301	0.677	1.5578

Tabuľka 15: Riešenie príkladu.

8. príklad

Ukážte, že pre dotaz Q_1 = „affection“ je radenie skóre troch dokumentov z tabuľky nižšie v opačnom poradí ako pre dotaz Q_2 = „jealous gossip“. Dotaz je vážený normalizáciou *tf*.

	SaS	PaP	WH
affection	0.996	0.993	0.847
jealous	0.087	0.120	0.466
gossip	0.017	0	0.254

Tabuľka 16: Zadanie príkladu.

K pôvodnej tabuľke pripojíme dotazy čím dostaneme

	SaS	PaP	WH	Q_1	Q_2
affection	0.996	0.993	0.847	1	0
jealous	0.087	0.120	0.466	0	1
gossip	0.017	0	0.254	0	1

Tabuľka 17: Zadanie s dotazmi.

Teraz vektory Q_i normalizujeme podľa definície 8 a dostaneme

	SaS	PaP	WH	Q_1	Q_2	Q_{1n}	Q_{2n}
affection	0.996	0.993	0.847	1	0	1	0
jealous	0.087	0.120	0.466	0	1	0	0.7071
gossip	0.017	0	0.254	0	1	0	0.7071

Tabuľka 18: Zadanie s dotazmi po normalizácii.

V poslednom kroku vypočítame skóre medzi dotazom a dokumentom ako

$$score(d, q) = \sum_{i=1}^{|d|} (d_i \cdot q_i)$$

Dosadením do vzorca uvedeného vyššie dostaneme

$$\begin{aligned}
score(SaS, Q_1) &= 0.9961 \cdot 1 + 0.087 \cdot 0 + 0.017 \cdot 0 && = 0.9961 \\
score(PaP, Q_1) &= 0.993 \cdot 1 + 0.120 \cdot 0 + 0 \cdot 0 && = 0.993 \\
score(WH, Q_1) &= 0.847 \cdot 1 + 0.466 \cdot 0 + 0.254 \cdot 0 && = 0.847 \\
\\
score(SaS, Q_2) &= 0.9961 \cdot 0 + 0.087 \cdot 0.7071 + 0.017 \cdot 0.7071 && = 0.0735 \\
score(PaP, Q_2) &= 0.993 \cdot 0 + 0.120 \cdot 0.7071 + 0 \cdot 0.7071 && = 0.0849 \\
score(WH, Q_2) &= 0.847 \cdot 0 + 0.466 \cdot 0.7071 + 0.254 \cdot 0.7071 && = 0.5091
\end{aligned}$$

Z čoho plynie usporiadanie pre $Q_1 = \text{SAS} > \text{PaP} > \text{WH}$ a pre $Q_2 = \text{WH} > \text{PaP} > \text{SAS}$, čo je poradie opačné.

3 Tretie cvičenie

Definícia 9 (Precision – presnosť)
Presnosť určuje koľko relevantných dokumentov vráti systém voči dotazu.

$$Precision = P = \frac{\#relevant\ retrieved}{\#retrieved}$$

Definícia 10 (Recall – pokrytie)
Pokrytie určuje koľko relevantných dokumentov je vrátených z celkového počtu relevantných dokumentov v systéme.

$$Recall = R = \frac{\#relevant\ retrieved}{\#relevant}$$

Definícia 11 (F hodnota)
Definujeme F hodnotu ako vzťah medzi Recall a Precision reprezentovanú váženým harmonickým priemerom následovne:

$$F = \frac{1}{\alpha\frac{1}{P} + (1-\alpha)\frac{1}{R}} = \frac{(\beta^2+1)PR}{\beta^2 P + R} \text{ kde } \beta^2 = \frac{1-\alpha}{\alpha}$$

kde $\alpha \in \langle 0; 1\rangle$. Vyrovnaná F hodnota váži rovnako R a P práve vtedy keď $\alpha = 0.5 \Longrightarrow \beta = 1$.

Definícia 12 (MAP – Mean Average Precision)
MAP vyjadruje presnosť v každom bode, v ktorom je nový relevantný dokument zahrnutý do výsledku. Počíta sa ako

$$MAP(Q) = \frac{1}{N} \cdot \left(\sum_{j=1}^{N} \frac{1}{m_j} \cdot \left(\sum_{k=1}^{m_j} P(doc_i) \right) \right)$$

kde $N = |Q|$ je počet dotazov, m_j je počet relevantných dokumentov pre dotaz j a $P(doc_i)$ je Precision i-teho dokumentu.

Definícia 13 (κ zhoda)
Nech $P(A) = \frac{\#zhodnych}{N}$ je počet dokumentov na ktorom sa zhodnú sudcovia. Ďalej definujeme $P(E)$ ako $P(E) = P(NR)^2 + P(R)^2$ ako odhadovaný počet nezhody medzi sudcami, kde $P(NR)$ je počet nerelevantných dokumentov, definovaných ako

$$P(NR) = \frac{NR_1 + NR_2}{N + N}$$

kde N je celkový počet dokumentov a NR_i je počet nerelevantných dokumentov podľa sudcu i. Vo vzťahu pre $P(E)$ je potrebné definovať ešte $P(R)$ ako počet relevantných dokumentov následovne

$$P(R) = \frac{R_1 + R_2}{N + N}$$

kde R_i je počet relevantých dokumentov podľa sudcu i. Potom definujeme κ štatistiku ako metriku zhody medzi sudcami vzťahom

$$\kappa = \frac{P(A) - P(E)}{1 - P(E)}$$

Definícia 14 (Rocchio relevance feedback)
Rocchio relevance feedback má následovný tvar[4]

[4]Publikácia [2] miesto d_r a d_{nr} používa jednotnú notáciu d_j. Platí teda $d_r = d_{nr} = d_j$.

$$q_m = \alpha q_0 + \beta \frac{1}{|D_r|} \sum_{\vec{d_r} \in D_r} \vec{d_r} - \gamma \frac{1}{|D_{nr}|} \sum_{\vec{d_{nr}} \in D_{nr}} \vec{d_{nr}}$$

kde

- q_0 *je pôvodný vektor dotazu*
- D_r *je množina relevantných dokumentov*
- D_{nr} *je množina nerelevantných dokumentov*
- *a hodnoty* $\alpha,\ \beta,\ \gamma$ *závisia od nastavenia systému.*

1. príklad

IR systém vráti 8 relevantných dokumentov a 10 nerelevantných dokumentov. Dohromady je v kolekcii 20 relevantných dokumentov. Aká je presnosť a úplnosť (precision, recall) systému pri tomto vyhľadávaní?

Precision vypočítame podľa definície 9 ako:

$$P = \frac{8}{10+8} = \frac{8}{18} = \frac{4}{9}$$

a Recall podľa definície 10 ako:

$$R = \frac{8}{20} = \frac{2}{5}.$$

2. príklad

Nasledujúci zoznam písmen R a písmen N reprezentuje relevantné (R) a nerelevantné (N) dokumenty vrátené v usporiadanom zozname 20 výsledkov ako odpoveď na dotaz z kolekcie 10 000 dokumentov. Prvý (najrelevantnejší) výsledok zoznamu je naľavo. Tento zoznam obsahuje 6 relevantných dokumentov. Predpokladajte, že kolekcia obsahuje dohromady 8 relevantných dokumentov ku dotazu. R R N N N N N N R N R N N N R N N N N R.

a) Aká je presnosť systému na prvých 20 výsledkoch?

b) Aká je F1 na prvých 20 výsledkoch?

c) Aká je neinterpolovaná presnosť systému pri 25% pokrytí?

d) Aká je interpolovaná presnosť systému pri 33% pokrytí?

e) Predpokladajte, že týchto 20 dokumentov je úplný zoznam výsledkov systému. Aký je MAP systému pre tento dotaz?

Teraz predpokladajte, že systém vrátil všetkých 10 000 dokumentov v zoradenom zozname a hore je uvedených prvých 20 vrátených výsledkov.

f) Aký najvyšší možný MAP môže tento systém dosiahnuť?

g) Aký najnižší možný MAP môže tento systém dosiahnuť?

h) Pri sade experimentov bolo vyhodnotených len prvých 20 výsledkov. Výsledok (e) bol použitý na na aproximovanie rozsahu (f)-(g). Aká veľká môže byť chyba pre výpočet MAP pri počítaní (e) namiesto (f) a (g) pre tento dotaz?

Vo výsledku sa nachádza 6 relevantných dokumentov. Aplikáciou definície 9 na výpočet Precision dostaneme pre **a)** $P = \frac{6}{20} = \frac{3}{10}$. Rovnako za pomoci definície 10 vypočítame Recall. Zo zadania vieme, že celkovo je je v kolekcii 8 relevantných dokumentov $R = \frac{6}{8} = \frac{3}{4}$. Pre zadanie **b)** je potrebné vypočítať F hodnotu podľa definície 11 pričom zvolíme $\alpha = 0.5$.

$$\frac{(\beta^2+1)PR}{\beta^2 P + R} = \frac{(1^2+1)\cdot\frac{3}{10}\cdot\frac{3}{4}}{\frac{3}{10}+\frac{3}{4}} = \frac{\frac{9}{20}}{\frac{21}{20}} = \frac{3}{7}$$

Pre výpočet neinterpolovanej presnosti systému pri 25% pokrytí (**c)**) potrebujeme vypočítať presnosti pre dokumenty s Recall rovným 25 %.

1. $P = \frac{1}{1}$ $R = \frac{1}{8}$ $= 12.5\%$
2. $P = \frac{2}{2}$ $R = \frac{2}{8}$ $= 25\%$
3. $P = \frac{2}{3}$ $R = \frac{2}{8}$ $= 25\%$

$\cdots$

8. $P = \frac{2}{8}$ $R = \frac{2}{8}$ $= 25\%$
9. $P = \frac{3}{9}$ $R = \frac{3}{8}$ $= 37.5\%$

Vidíme, že prvý dokument ma $R = 12.5\%$ a táto hodnota nás nezaujíma. Pre dokumenty 2 až 8 dostávame želané pokrytie 25 %. 9 dokument už presahuje želanú hodnotu a preto ho do výsledku nezaradíme. Neinterpolovaná presnosť tým pádom bude množina prvých presností prvých ôsmych dokumentov.

$$P = \left\{\frac{2}{2}, \frac{2}{3}, \frac{2}{4}, \frac{2}{5}, \frac{2}{6}, \frac{2}{7}, \frac{2}{8}\right\}$$

Pre interpolovanú presnosť (**d)**) hľadáme najväčšiu presnosť pre relevantné dokumenty s Recall väčším ako 33 %. Recall sa zmení iba ak výsledok bude obsahovať relevantný dokument. preto hodnoty vypočítame pre dokumenty 11, 15 a 20.

11. $P = \frac{4}{11}$ $R = \frac{4}{8}$ $= 50\%$
15. $P = \frac{5}{15}$ $R = \frac{5}{8}$ $= 62.5\%$
20. $P = \frac{6}{20}$ $R = \frac{6}{8}$ $= 75\%$

Požadovaný Recall prekročíme vrátením dokumentu 9 (37.5 %). Teraz už iba ostáva určiť $max(P_9, P_{11}, P_{15}, P_{20}) = P_{11} = \frac{4}{11} = 0.\overline{36}$.

Pre vypočítanie MAP systému **e)** použijeme definíciu 12. Kedže dotaz máme len jeden $N = |Q| = 1$ a v prvých 20 dokumentov máme $m_j = 6$ relevantných dokumentov tak

$$MAP(Q) = \frac{1}{1}\left(\sum_{j=1}^{1}\frac{1}{6}\left(\sum_{i=1}^{6}P(doc_i)\right)\right) = \frac{1}{1}\cdot\frac{1}{6}\left(\underbrace{\frac{1}{1}}_{P(1.)} + \underbrace{\frac{2}{2}}_{P(2.)} + \underbrace{\frac{3}{9}}_{P(9.)} + \underbrace{\frac{4}{11}}_{P(11.)} + \underbrace{\frac{5}{15}}_{P(15.)} + \underbrace{\frac{6}{20}}_{P(20.)}\right)$$

$$= \frac{1}{1}\cdot\frac{1}{6}\cdot\frac{1099}{330} = \frac{1099}{1980} = 0.55\overline{50}$$

Z definície výpočtu presnosti 9 a MAP 12 vyplýva, že keby boli zvyšné 2 relevantné dokumenty na miestach 21 a 22 tak MAP bude najvyšší možný (**f)**).

$$MAP(Q) = \frac{1}{8}\left(\frac{1}{1} + \frac{2}{2} + \frac{3}{9} + \frac{4}{11} + \frac{5}{15} + \frac{6}{20} + \frac{7}{21} + \frac{8}{22}\right) = 0.503\overline{409}$$

Naopak keby boli na posledných miestach bol by MAP najmenší(**g)**).

$$MAP(Q) = \frac{1}{8}\left(\frac{1}{1} + \frac{2}{2} + \frac{3}{9} + \frac{4}{11} + \frac{5}{15} + \frac{6}{20} + \frac{7}{9999} + \frac{8}{10000}\right) = 0.4164\overline{7538}$$

Pre **h)** je podľa **f)** a **g)** priemerná predpokladaná presnosť všetkých 10 000 dokumentov rovná $\frac{0.503+0.416}{2} = 0.46$. Ak tento výsledok využijeme na výpočet chyby pre MAP z úlohy **e)** dostávame $0.555 - 0.46 = 0.095$.

3. príklad

Nižšie je tabuľka ukazujúca ako dvaja znalci ohodnotili relevanciu množiny 12 dokumentov k nejakej informačnej potrebe (0=nerelevantné, 1=relevantné). Predpokladajme, že ste vyvinuli IR systém, ktorý pre tento dotaz vráti dokumenty $\{4, 5, 6, 7, 8\}$.

Doc ID	Judge 1	Judge 2
1	0	0
2	0	0
3	1	1
4	1	1
5	1	0
6	1	0
7	1	0
8	1	0
9	0	1
10	0	1
11	0	1
12	0	1

Tabuľka 19: Názory sudcov na dokumenty.

a) Vypočítajte Kappa mieru zhody medzi týmito znalcami.

b) Vypočítajte presnosť, pokrytie a F1 vášho systému, ak je dokument relevantný len ak sa na ňom zhodli obaja znalci.

c) Vypočítajte presnosť, pokrytie a F1 vášho systému, ak je dokument relevantný ak si to myslí aspoň jeden zo znalcov.

Pre zadanie **a)** je nutné použiť definíciu 13. V prvom kroku vypočítame $P(A)$ ako počet dokumentov, na ktorom sa sudcovia zhodli. Keďže ide o dokumenty $\{1,2,3,4\}$ a celkový počet je $N = 12$ potom $P(A) = \frac{|\{1,2,3,4\}|}{12} = \frac{4}{12} = \frac{1}{3}$. Ďalej potrebujeme počty nezhôd medzi jednotlivými sudcami. Sudca 1 považuje dokumenty $NR_1 = \{1,2,9,10,11,12\}$ a sudca 2, $NR_2 = \{1,2,5,6,7,8\}$ za nerelevantné. Dosadením do vzorca pre $P(NR)$ dostaneme $P(NR) = \frac{|NR_1|+|NR_2|}{12+12} = \frac{6+6}{24} = \frac{1}{2}$. Teraz výpočet zopakujeme pre $P(R)$. Pretože počet relevantných je rovnaký nerelevantným tak potom $P(R) = P(NR) = \frac{1}{2}$. Teraz môžeme vypočítať $P(E)$ ako

$$P(E) = P(NR)^2 + P(R)^2 = \left(\frac{1}{2}\right)^2 + \left(\frac{1}{2}\right)^2 = \frac{1}{4} + \frac{1}{4} = \frac{1}{2}$$

vďaka čomu máme všetky premenné pre výpočet κ.

$$\kappa = \frac{P(A) - P(E)}{1 - P(E)} = \frac{\frac{1}{3} - \frac{1}{2}}{1 - \frac{1}{2}} = -\frac{1}{3}$$

Podľa [2, str. 166] ak je hodnota $\kappa < 0$ tak zhoda medzi sudcami je viac než náhodná. Na vyhodnotenie aspoň približnej zhody požadujeme hodnoty z intervalu $(0.67-0.8)$. Pre zadanie **b)** je potrebné vypočítať Precision a Recall. Náš systém vráti ako relevantné dokumenty $\{4,5,6,7,8\}$. Zhoda sudcov pre relevantné dokumenty je $\{3,4\}$. Prienikom týchto množín je tým pádom dokument 4. Podľa definície 9 a dosadení hodnôt dostaneme

$$P = \frac{|\{4\}|}{|\{4,5,6,7,8\}|} = \frac{1}{5}$$

Pretože počet relevantných dokumentov je $|\{3,4\}| = 2$ potom Recall je $R = \frac{1}{2}$. Potom podľa definície 11 je F hodnota rovná

$$F = \frac{(\beta^2+1)PR}{\beta^2 P + R} = \frac{(1+1)\frac{1}{5}\frac{1}{2}}{\frac{1}{5}+\frac{1}{2}} = \frac{2}{7}$$

Rovnakým spôsobom vypočítame zadanie **c)**, ktoré hovorí, že dokument je relevantný ak sa na ňom zhodol aspoň jeden sudca. Znamená to, že ide o dokumenty $\{3,4,5,6,7,8,9,10,11,12\}$. Ich prienik s naším výsledkom $\{4,5,6,7,8\}$ dáva množinu $\{4,5,6,7,8\}$, ktorej veľkosť je rovná 5. Precision je potom rovné $P = \frac{5}{5} = 1$ a Recall $R = \frac{|\{4,5,6,7,8\}|}{|\{3,4,5,6,7,8,9,10,11,12\}|} = \frac{5}{10} = \frac{1}{2}$. F je potom vypočítaná ako

$$F = \frac{(1+1)\frac{1}{2}}{1+\frac{1}{2}} = \frac{2}{3}$$

4. príklad

Užívateľov prvotný dotaz je „cheap CDs cheap DVDs extremely cheap CDs“. Užívateľ preskúma dva dokumenty doc1 a doc2. Ohodnotí dokument doc1 „CDs cheap software cheap CDs“ ako relevantný a doc2 „cheap thrills DVDs“ ako nerelevantný. Predpokladajme, že používame jednoduchú tf bez dĺžkovej normalizácie vektorov. Použitím Rocchio relevance feedbacku aký by bol prepracovaný vektor dotazu po zvážení relevance feedbacku? Použite hodnoty $\alpha = 1,\ \beta = 0.75,\ \gamma = 0.25$.

Zadanie si prevedieme do tabuľky pre pohodlnejšie spracovanie.

	relevantný	nerelevantný	
termy	doc1	doc2	dotaz
Cds	2	0	2
cheap	2	1	3
software	1	0	0
thrills	0	1	0
DVDs	0	1	1
extremely	0	0	1

Tabuľka 20:

Teraz si označíme vstup algoritmu podľa definície 14.

$$d_r \in D_r = \begin{pmatrix} 2 \\ 2 \\ 1 \\ 0 \\ 0 \\ 0 \end{pmatrix},\ d_{nr} \in D_{nr} = \begin{pmatrix} 0 \\ 1 \\ 0 \\ 1 \\ 1 \\ 0 \end{pmatrix},\ q = \begin{pmatrix} 2 \\ 3 \\ 0 \\ 0 \\ 1 \\ 1 \end{pmatrix}$$

Dosadením hodnôt do vzorca pre q_m dostaneme

$$
\begin{aligned}
q_m &= 1 \cdot \begin{pmatrix} 2 \\ 3 \\ 0 \\ 0 \\ 1 \\ 1 \end{pmatrix} + 0.75 \cdot \tfrac{1}{1} \begin{pmatrix} 2 \\ 2 \\ 1 \\ 0 \\ 0 \\ 0 \end{pmatrix} - 0.25 \cdot \tfrac{1}{1} \begin{pmatrix} 0 \\ 1 \\ 0 \\ 1 \\ 1 \\ 0 \end{pmatrix} \\
&= \begin{pmatrix} 2 \\ 3 \\ 0 \\ 0 \\ 1 \\ 1 \end{pmatrix} + \begin{pmatrix} 1.5 \\ 1.5 \\ 0.75 \\ 0 \\ 0 \\ 0 \end{pmatrix} - \begin{pmatrix} 0 \\ 0.25 \\ 0 \\ 0.25 \\ 0.25 \\ 0 \end{pmatrix} \\
&= \begin{pmatrix} 3.5 \\ 4.25 \\ 0.5 \\ -0.25 \\ 0.75 \\ 1 \end{pmatrix}
\end{aligned}
$$

5. príklad

Prečo je pozitívny feedback pravdepodobne lepší ako negatívny feedback pre IR systém? Prečo je možno lepšie použiť na feedback len jeden nerelevantný dokument ako ich použiť viac?

Pozitívny feedback je lepší ako negatívny, pretože väčšinou v kolekcii je oveľa viac nerelevantných dokumentov ako relevantných. Preto keď pri relevance feedbacku označíme nejaký dokument nerelevantný, tak tým iba vylúčime určitú množinu nerelevantných dokumentov, ktorých ale v systéme ostáva ešte veľa.

6. príklad

Prečo je prírastková relevancia viacej realistické merítko užívateľskej spokojnosti? Udajte príklad kde neprírastková metrika ako napríklad presnosť alebo úplnosť je zavádzajúce merítko užívateľskej spokojnosti a naopak prírastková je lepšia?

Presnosť a úplnosť je zavádzajúce merítko, pretože keby na každý dotaz vrátime úplne všetky dokumenty, ktoré sa nachádzajú v kolekcií, úplnosť by bola 100 % a keby sme na každý dotaz vracali 0 výsledkov tak presnosť systému je 99.99 %.

4 Štvrté cvičenie

Definícia 15 (Pomer indexov)
Predpokladajme, že vyberieme náhodnú stránku z indexu E_1, overíme či je v indexe E_2 a symetricky overíme, či náhodná stránka z E_2 je v E_1. Tieto testy nám dajú zlomky x a y také, že náš odhad je, že zlomok x stránok z E_1 sú v E_2 zatiaľ čo zlomok y stránok z E_2 sú v E_1. Potom označíme $|E_i|$ ako veľkosť indexu vyhľadávača E_i dostaneme

$$x|E_1| \approx y|E_2|,$$

úpravou čoho dostaneme

$$\frac{|E_1|}{|E_2|} \approx \frac{y}{x}.$$

Definícia 16 (Markovova matica prechodu)
Je daný graf $G = (V, E)$. Nech A je matica susedností o veľkosti $N \times N$, kde prvok $a_{uv} = 1 \iff (u, v) \in E \wedge u, v \in V$. Maticu prechodu P potom vypočítame následovne

1. *ak riadok i matice A nemá žiadnu 1, potom nahradíme každý prvok v riadku i hodnotou $\frac{1}{N}$. V inom prípade pokračujeme v bode 2*
2. *vydelíme každú hodnotu 1 v matici A počtom jednotiek v danom riadku*
3. *vynásobíme maticou číslom $1 - \alpha$*
4. *pripočítame hodnotu $\frac{\alpha}{N}$ ku každej hodnote v matici*

kde α je pravdepodobnosť teleportu.

Algoritmus 5 (Výpočet PageRank)
1: ***function*** PageRank*(P)*
2: $i \leftarrow 0$
3: $\overrightarrow{x}_i = (1, 0, 0)$
4: $\overrightarrow{x}_{i+1} = (0, 0, 0)$
5: ***repeat***
6: $\overrightarrow{x}_{i+1} = \overrightarrow{x}_i \cdot P$
7: $i = i + 1$
8: ***until*** $x_i \neq x_{i-1}$
9: ***end function***

Definícia 17 (Huby a autority)
Označme $h(v)$ ako hub skóre a $a(v)$ ako skóre autority. V prvom kroku nastavíme položky vektorov $h(v)$ a $a(v)$ na 1 pre všetky uzly $v \in V$.

$$\begin{aligned} h(v) &= A \cdot a(v) \\ a(v) &= A^T \cdot h(v) \end{aligned}$$

čo je ekvivalentné zápisu

$$\begin{aligned} h(v) &= A \cdot A^T \cdot h(v) \\ a(v) &= A^T \cdot A \cdot a(v) \end{aligned}$$

1. príklad

Každý z dvoch webových vyhľadávacích systémov A a B zo svojich indexov generujú veľké množstvo stránok rovnomerne náhodne. 30 % stránok z A sa nachádza v indexe B a 50 % stránok z B sa nachádza v indexe A. Aký je pomer stránok medzi systémami A a B?

Jednoduchým dosadením do definície 15 dostaneme vzťahy

- zlomok $\frac{3}{10}$ z indexu A je v B
- zlomok $\frac{5}{10}$ z indexu B je v A

vďaka čomu dostaneme vzťah

$$\begin{aligned} 0.3|A| &\approx 0.5|B| \\ \frac{|A|}{|B|} &\approx \frac{0.5}{0.3} \\ \frac{|A|}{|B|} &\approx \frac{5}{3} \end{aligned}$$

2. príklad

Každý z dvoch webových vyhľadávacích systémov A a B zbierajú (crawl) náhodnú, ale rovnako veľkú podmnožinu Webu. Niektoré zozbierané stránky sú duplikáty - presné textové kópie na rôznych URL. Predpokladajte, že sú duplikáty distribuované rovnomerne medzi stránkami zozbierané systémom A aj B. Ďalej predpokladajte, že duplikát je stránka, ktorá má presne dve kópie - žiadne stránky nemajú viac ako dve kópie. A indexuje stránky bez eliminácie duplikátov, kdežto B indexuje len jednu kópiu duplikovaných stránok. Tieto dve náhodné podmnožiny majú rovnakú veľkosť pred odstránením duplikátov. Ak sa 45 % stránok z A nachádza v indexe B, a 50 % stránok z B v indexe A, aká veľká časť Webu sa skladá zo stránok, ktoré nemajú duplikáty?

Nech D je počet duplikátov. Vieme, že duplikáty sú distribuované rovnomerne medzi A a B. To znamená, že $|A| \approx |B|$. Ďalej vieme, že B indexuje iba jednu kópiu duplikovaných stránok teda počet stránok indexovaných systémom B je $|B| - \frac{D}{2}$[5]. Použitím podobného postupu ako v predchádzajúcom príklade dostaneme

$$0.45|A| \approx 0.5\left(|B| - \frac{D}{2}\right) \tag{1}$$

$$0.45|A| \approx 0.5|B| - 0.25D \tag{2}$$

$$0.45|A| \approx 0.5|A| - 0.25D \tag{3}$$

$$\frac{0.05}{0.25}|A| \approx D \tag{4}$$

$$\frac{1}{5}|A| \approx D \tag{5}$$

Na riadku 3 sme využili faktu, že $|A| \approx |B|$. Teraz vieme, že web obsahuje $\frac{1}{5}$ duplikátov čo znamená, že $\frac{4}{5}$ webu netvoria duplikáty.

[5] zo zadania duplikát = 2 stránky, takže tento systém ich musí mať o polovicu menej.

3. príklad

Daný je nasledujúci web graf $G = (V = \{a, b, c\},\ E = \{a \to b, a \to c, b \to c, c \to b\})$. Vypočítajte PageRank, hub skóre a autoritatívne skóre pre každú z troch stránok. Zoraďte stránky podľa jednotlivých skóre a pozorujte prípadné väzby. Pre výpočet PageRank môžte predpokladať, že sa v každom kroku náhodnej prechádzky teleportujeme na náhodnú stránku s pravdepodobnosťou 0.1 a s rovnomernou distribúciou stránok, na ktoré sa teleportujeme. Pre huby a autority normalizujte skóre tak, aby maximum bolo 1.

Graf si podľa 16 zapíšeme ako maticu susedností, čím dostaneme

$$\begin{pmatrix} 0 & 1 & 1 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{pmatrix}$$

pokúsime sa aplikovať krok 1 v algoritme, ale naša matica neobsahuje riadok so samými jednotkami a preto prejdeme na krok 2. Prvý riadok obsahuje dve jednotky a preto všetky hodnoty predelíme 2. Druhý a tretí majú iba jednu jednotu a tieto hodnoty ostanú nezmenené

$$\begin{pmatrix} 0 & 1 & 1 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{pmatrix} \begin{matrix} :2 \\ :1 \\ :1 \end{matrix} = \begin{pmatrix} 0 & \frac{1}{2} & \frac{1}{2} \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{pmatrix}$$

Teraz aplikujeme krok 3. Keďže $\alpha = 0.1$ tak vynásobíme maticu hodnotou 0.9

$$\begin{pmatrix} 0 & \frac{1}{2} & \frac{1}{2} \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{pmatrix} \cdot \ 0.9 \ = \begin{pmatrix} 0 & \frac{9}{20} & \frac{9}{20} \\ 0 & 0 & \frac{9}{10} \\ 0 & \frac{9}{10} & 0 \end{pmatrix}$$

a podľa kroku 4 pripočítame k matici hodnotu $\frac{\alpha}{N} = \frac{0.1}{3} = \frac{1}{30}$

$$\begin{pmatrix} 0 & \frac{9}{20} & \frac{9}{20} \\ 0 & 0 & \frac{9}{10} \\ 0 & \frac{9}{10} & 0 \end{pmatrix} + \ \frac{1}{30} \ = \begin{pmatrix} \frac{1}{30} & \frac{29}{60} & \frac{29}{60} \\ \frac{1}{30} & \frac{1}{30} & \frac{14}{15} \\ \frac{1}{30} & \frac{14}{15} & \frac{1}{30} \end{pmatrix} = \ P$$

Teraz použijeme algoritmus 5 na výpočet PageRank. Zvolíme $\overrightarrow{x}_0 = (1, 0, 0)$ a maticu P z predchádzajúceho výpočtu.

$$\overrightarrow{x}_1 = \overrightarrow{x}_0 \cdot P \tag{6}$$

$$\overrightarrow{x}_1 = (1, 0, 0) \cdot \begin{pmatrix} \frac{1}{30} & \frac{29}{60} & \frac{29}{60} \\ \frac{1}{30} & \frac{1}{30} & \frac{14}{15} \\ \frac{1}{30} & \frac{14}{15} & \frac{1}{30} \end{pmatrix} \tag{7}$$

$$\overrightarrow{x}_1 = \left(\frac{1}{30}, \frac{29}{60}, \frac{29}{60}\right) \tag{8}$$

$$\overrightarrow{x}_2 = \overrightarrow{x}_1 \cdot P \tag{9}$$

$$\overrightarrow{x}_2 = \left(\frac{1}{30}, \frac{29}{60}, \frac{29}{60}\right) \cdot \begin{pmatrix} \frac{1}{30} & \frac{29}{60} & \frac{29}{60} \\ \frac{1}{30} & \frac{1}{30} & \frac{14}{15} \\ \frac{1}{30} & \frac{14}{15} & \frac{1}{30} \end{pmatrix} \tag{10}$$

$$\overrightarrow{x}_2 = \left(\frac{1}{30}, \frac{29}{60}, \frac{29}{60}\right) \tag{11}$$

Pretože $x_i = x_{i-1}$ prehlásime položky vektoru x_2 za PageRanky jednotlivých stránok. Podľa definície 17 nastavíme vektory $h(v)$ a $a(v)$ na 1.

$$a(v) = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}, \quad h(v) = \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix}$$

Dosadením do druhého vzťahu v definícii dostaneme

$$\begin{aligned} h(v) &= A \cdot A^T \cdot h(v) \\ h(v) &= \begin{pmatrix} 0 & 1 & 1 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{pmatrix} \cdot \begin{pmatrix} 0 & 0 & 0 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \end{pmatrix} \cdot \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} \\ h(v) &= \begin{pmatrix} 2 & 1 & 1 \\ 1 & 1 & 0 \\ 1 & 0 & 1 \end{pmatrix} \cdot \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} \\ h(v) &= \begin{pmatrix} 4 \\ 2 \\ 2 \end{pmatrix} \end{aligned}$$

Skóre autorít potom vypočítame ako

$$\begin{aligned} a(v) &= A^T \cdot A \cdot a(v) \\ a(v) &= \begin{pmatrix} 0 & 0 & 0 \\ 1 & 0 & 1 \\ 1 & 1 & 0 \end{pmatrix} \cdot \begin{pmatrix} 0 & 1 & 1 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{pmatrix} \cdot \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} \\ a(v) &= \begin{pmatrix} 0 & 0 & 0 \\ 0 & 2 & 1 \\ 0 & 1 & 2 \end{pmatrix} \cdot \begin{pmatrix} 1 \\ 1 \\ 1 \end{pmatrix} \\ a(v) &= \begin{pmatrix} 0 \\ 3 \\ 3 \end{pmatrix} \end{aligned}$$

V zadaní je od nás ešte požadovaná normalizácia vektorov a teda $h(v) = \begin{pmatrix} 4 \\ 2 \\ 2 \end{pmatrix} \rightsquigarrow \begin{pmatrix} 1 \\ 0.5 \\ 0.5 \end{pmatrix}$ a $a(v) = \begin{pmatrix} 0 \\ 3 \\ 3 \end{pmatrix} \rightsquigarrow \begin{pmatrix} 0 \\ 1 \\ 1 \end{pmatrix}$.

4. príklad

Priemerný vstupný stupeň všetkých uzlov vybraného grafu webu je 9. Čo môžeme povedať o priemernom výstupnom stupni všetkých uzlov tohto grafu?

Podľa [2, str. 426] je stupeň výstupných uzlov približne rovnaký ako stupeň vstupných uzlov. Keďže vstupný stupeň je 9 rovnaký bude aj výstupný.

5 Piate cvičenie

Definícia 18
Podobnosť medzi dotazom XPath a cestou v dokumente sa počíta ako

$$CR(c_q, c_d) = \begin{cases} \frac{1+|c_q|}{1+|c_d|} & \text{ak } c_q \text{ zodpovedá } c_d \\ 0 & \text{inakšie} \end{cases}$$

Pojem zodpovedá znamená následovnému: c_q zodpovedá c_d práve vtedy keď c_q sa dá rozšíriť na c_d pridaním uzlov do cesty.

Definícia 19
Structural term je definovaný ako dvojica existujúcej cesty k hodnote a hodnota samotná, pričom hodnota sama o sebe je je taktiež uzol v XML dokumente. To znamená, že XML dokument obsahujúci iba root element s textom ako je napríklad

```
<root>
   test
</root>
```

Obsahuje 2 structural termy <c,t> a to `</root/,test>` *a potom hodnotu samotnú t.j.* `</,test>`.

Definícia 20
Bernoulliho model reprezentuje dokument ako binárny vektor, v ktorom jednotlivé položky tvoria dané termy zo slovníka V. Prítomnosť termu $t_i \in V$ označíme vo vektore v hodnotou 1, pričom jeho neprítomnosť hodnotou 0.

Definícia 21
Multinomický model je rovnaký ako Bernoulliho s tým rozdielom, že vektor miesto prítomnosti termu v dokumente udáva počet výskytov.

Definícia 22
Nech $P(c)$ je pravdepodobnosť, že sa dokument nachádza v triede c a platí, že

$$P(c) = \frac{N_C}{N},$$

kde N je celkový počet dokumentov a N_c je počet dokumentov v triede c.

Definícia 23
Definujeme $P(t \mid c)$ ako pravdepodobnosť, výskytu terminu t v dokumente, ktorý patrí do triedy c pre Naive Bayes model.

$$P(t \mid c) = \frac{T_{ct} + 1}{(\sum_{t' \in V} T_{ct'}) + B},$$

4 kde T_{ct} je výskyt termu t v trénovacej množine dokumentov patriacich do triedy c, vrátane viacnásobnych výskytov a $B = |V|$, kde V je množina slov.

Definícia 24
Pre Bernoulliho Naive Bayes model je vzťah pre $P(t \mid c)$ trochu odlišný. Pretože Bernoulliho model podľa definície 20 rieši iba či je term prítomný (t.j. hodnota vo vektore je rovná 1) upravíme $P(t \mid c)$ ako

$$P(t \mid c) = \frac{T_{ct} + 1}{N_c + 2},$$

kde T_{ct} je súčet výskytov.

Definícia 25
Aplikácia testovacieho dokumentu d_t na testovaciu množinu T sa pomocou Multinomického Naive Bayes modelu počíta ako

$$P(c \mid d_t) = P(c) \cdot \prod_{t \in d_t} P(t \mid c),$$

kde $P(c)$ je z definície 22 a $P(t \mid c)$ z definície 23.

Definícia 26
Aplikácia testovacieho dokumentu d_t na testovaciu množinu T sa pomocou Bernoulliho Naive Bayes modelu počíta ako

$$P(c \mid d_t) = P(c) \cdot \prod_{t \in d_t} P(t \mid c) \cdot \prod_{t' \in M_d} (1 - P(t' \mid c)),$$

kde $P(c)$ je z definíce 22 a $P(t \mid c)$ z definície 23 a M_d je množina termov, ktorá vnikne ako

$$M_d = \left(\bigcup_{doc_i \in c} doc_i \right) \setminus d_t$$

t.j. všetky termy z trénovacej množiny patriacich do c a bez tých, ktoré sa nachádzajú v testovacom dokumente.

1. príklad

Pre XML dokument uvedený nižšie napíšte XPath výrazy.

a) Vráťte všetky názvy (title elementy), ako kurzu, tak oddelenia.

b) Vráťte názvy kurzov, ktoré majú v názve výraz „programming“.

c) Vráťte priezviská inštruktorov učiacich aspoň jeden kurz, ktorý má vo svojom popise slovo „software“.

d) Vráťte priezviská profesorov učiacich aspoň jeden kurz, ktorý má vo svojom popise slovo „software“.

```
<Course_Catalog>
  <Department Code="CS">
    <Title>Computer Science</Title>
    <Chair>
      <Professor>
```

```
        <First_Name>Jennifer</First_Name>
        <Last_Name>Widom</Last_Name>
      </Professor>
    </Chair>
    <Course Number="CS106A" Enrollment="1070">
      <Title>Programming Methodology</Title>
      <Description>Introduction to the engineering of computer applications
      emphasizing modern software engineering principles.
      </Description>
      <Instructors>
        <Lecturer>
          <First_Name>Jerry</First_Name>
          <Middle_Initial>R.</Middle_Initial>
          <Last_Name>Cain</Last_Name>
        </Lecturer>
        <Professor>
          <First_Name>Eric</First_Name>
          <Last_Name>Roberts</Last_Name>
        </Professor>
        <Professor>
          <First_Name>Mehran</First_Name>
          <Last_Name>Sahami</Last_Name>
        </Professor>
      </Instructors>
    </Course>
    <Course Number="CS106B" Enrollment="620">
      <Title>Programming Abstractions</Title>
      <Description>Abstraction and its relation to programming.</Description>
      <Instructors>
        <Professor>
          <First_Name>Eric</First_Name>
          <Last_Name>Roberts</Last_Name>
        </Professor>
        <Lecturer>
          <First_Name>Jerry</First_Name>
          <Middle_Initial>R.</Middle_Initial>
          <Last_Name>Cain</Last_Name>
        </Lecturer>
      </Instructors>
      <Prerequisites>
        <Prereq>CS106A</Prereq>
      </Prerequisites>
    </Course>
    </Department>
</Course_Catalog>
```

K riešeniu príkladu, resp naučeniu sa, a k evaluácii XPath sa odporúča [4].

a) `//Title`

b) `//Course//Title[contains(current(),'programming')]`

c) `//Course[contains(Description,'software')]/Instructors//Last_Name`

d) `//Course[contains(Description,'software')]/Instructors/Professor/Last_Name`

2. príklad

Vypočítajte podobnosť medzi dotazmi a im zodpovedajúcimi cestami v dokumente z Príkladu 1.

1. `//Instructors//Last_Name#Cain`
2. `//Course/Instructors/Lecturer/Last_Name#Cain`

Podľa definície 18 dotaz c_q zodpovedá dokumentu c_d práve vtedy keď sa dá rozšíriť. Pôvodný dotaz pre zadanie **a)** rozšírime na `Course_Catalog/Department/Course/Instructors/Professor/Last_Name`. Keďže c_q je možné rozšíriť na c_d môžeme použiť vzťah z definície. Dosadením dĺžky dotazu $c_q = 2$ a dĺžkou dokumentu $c_d = 6$ do vzorca dostaneme

$$CR(c_q, c_d) = \frac{1 + |c_q|}{1 + |c_d|} = \frac{1+2}{1+6} = \frac{3}{7}.$$

Pre **b)** zmeníme iba dĺžku dotazu, čím dostaneme

$$CR(c_q, c_d) = \frac{1 + |c_q|}{1 + |c_d|} = \frac{1+4}{1+6} = \frac{5}{7}.$$

3. príklad

Spočítajte, koľko štruktúrnych termov (structural terms, dvojíc kontext/term <c,t>) je v XML strome na nižšie.

```
<Course>
  <Title>Programming Abstractions</Title>
  <Description>Abstraction and its relation to programming</Description>
  <Instructors>
<Professor>
  <First_Name>Eric</First_Name>
  <Last_Name>Roberts</Last_Name>
</Professor>
  </Instructors>
</Course>
```

Označíme si každý element iba začiatočným písmenom takže Description bude ako D a Professor ako P. Použitím definície 19 spočítame všetky kombinácie a zapíšeme do tabuľky. Pre prehľad je možné menšie dokumenty zakresliť ako stromy a na nich si jednotlivé cesty ukázať.

C T Programming Abstraction	C I P F Eric	C I P L Roberts
T Programming Abstractions	I P F Eric	I P L Roberts
Programming Abstractions	P F Eric	P L Roberts
C D Abstractions...	F Eric	L Roberts
D Abstractions...	Eric	Roberts
Abstractions...		

Tabuľka 21:

Celkovo teda máme 16 štruktúrnych termov.

4. príklad

Ktorý z dokumentov uvedených nižšie má rovnaké alebo rozdielne bag of words reprezentácie pre Bernouliho a multinomický model? Aké sú rozdiely

Doc1: He moved from London, Ontario, to London, England.

Doc2: He moved from London, England, to London, Ontario.

Doc3: He moved from England to London, Ontario.

Podľa definície 20 je dokument reprezentovaný vektorom v, ktorého položky tvoria slova zo slovníka. Náš slovník má tvar $V = \{\text{He, moved, from, London, Ontario, to, England}\}$. Položka vektoru v_i zodpovedá slovu na pozícii v množine V. Vektor pre dokument1 bude tým pádom vyzerať ako $v_1 = (1, 1, 1, 1, 1, 1, 1)$, pre dokument 2 ako $v_2 = (1, 1, 1, 1, 1, 1, 1)$ a dokument 3 ako $v_3 = (1, 1, 1, 1, 1, 1, 1)$. Je zrejmé, že ak neberieme ohľad na počty, tak podľa Bernoulliho modelu sú dokumenty rovnaké. Podľa definície 21 berie multinomický model ohľad na výskyty termov. Pre dokument 1 teda dostaneme $v_1 = (1, 1, 1, 2, 1, 1, 1)$, dokument 2 $v_2 = (1, 1, 1, 2, 1, 1, 1)$ a dokument 3 $v_3 = (1, 1, 1, 1, 1, 1, 1)$. Podľa multinomického modelu sú dokumenty 1 a 2 rovnaké a dokument 3 je odlišný pretože výskyt termu London nie je rovný 2. O týchto metódach je vhodné si pozrieť kapitolu 13 v [2], alebo [3].

5. príklad

Na základe dát z tabuľky nižšie

a) odhadnite multinomické Naive Bayes klasifikátory,

b) aplikujte ich na testovací dokument,

c) odhadnite Bernoulli Naive Bayes klasifikátor,

d) aplikujte ho na testovací dokument.

Nemusíte odhadovať parametre, ktore na klasifikáciu dokumentu nie sú potrebné.

	docID	obsah dokumentu	je v c = China?
trénovacia množ.	1	Taipei Taiwan	áno
	2	Macao Taiwan Shanghai	áno
	3	Japan Sapporo	nie
	4	Sapporo Osaka Taiwan	nie
testovacia množ.	5	Taiwan Taiwan Sapporo	?

Tabuľka 22: Zadanie príkladu.

Pre zadanie **a)** vypočítame hodnoty $P(t \mid c)$ podľa definície 23 spolu s opačnými javmi. Opačný jav sa počíta rovnako akurát sa aplikuje na dokumenty, ktoré nepatria do triedy c.

$$P(Taiwan \mid c) = \tfrac{2+1}{5+7} = \tfrac{1}{4}$$

$$P(Sapporo \mid c) = \tfrac{0+1}{5+7} = \tfrac{1}{12}$$

$$P(Taiwan \mid \bar{c}) = \tfrac{1+1}{5+7} = \tfrac{1}{6}$$

$$P(Sapporo \mid \bar{c}) = \tfrac{2+1}{5+7} = \tfrac{1}{4}$$

Rozoberieme prvý a posledný výpočet. Podľa definície 23 je $P(t \mid c)$ rovná

$$P(t \mid c) = \frac{T_{ct} + 1}{(\sum_{t' \in V} T_{ct'}) + B},$$

Term *Taiwan* sa v trénovacej množine v dokumentoch patriacich nachádza 2 krát. Raz v dokumente 1 a raz v dokumente 2. $T_{ct} = 1+1$. Ďalej je nutné spočítať $\sum_{t' \in V} T_{ct'}$ ako počet všetkých slov (aj s viacnásobnými výskytmi) v trénovacej množine dokumentov patriacich do c. Taipei + Taiwan + Macao + Taiwan + Shanghain = 5. Tým pádom $\sum_{t' \in V} T_{ct'} = 5$. Dosadením do vzorca dostaneme požadovaný výpočet a výsledok $P(Taiwan \mid c) = \frac{1}{4}$. Teraz popíšeme výpočet $P(Sapporo \mid \bar{c})$. Vzorec ostáva rovnaký, akurát okrem c použijeme $\bar{c}$. Term *Sapporo* sa v dokumentoch v trénovacej množine **nepatriacich** do c nachádza opäť 2 krát (doc3, doc4). $T_{\bar{c}t} = 2$. Opäť vypočítame počet slov, ktoré sa v c nenachádzajú a sú z trénovacej množiny. Japan + Sapporo + Sapporo + Osaka + Taiwan = 5 a teda $\sum_{t' \in V} T_{\bar{c}t'} = 5$. Hodnoty z **a)** použijeme teraz k výpočtu **b)**. K výpočtu je potrebné vypočítať podľa definíce 22 hodnotu $P(c)$ ako $P(c) = \frac{2}{4} = \frac{1}{2}$. Potom $P(\bar{c})$ je jav opačný teda $P(\bar{c}) = 1 - P(c) = \frac{1}{2}$. Ďalej z definície 25 plynie:

$$\begin{aligned} P(c \mid doc5) &= P(c) \cdot \textstyle\prod_{t \in d_t} P(t \mid c) = \\ &= \tfrac{2}{4} \cdot P(Taiwan \mid c) \cdot P(Taiwan \mid c) \cdot P(Sapporo \mid c) \\ &= \tfrac{2}{4} \cdot \tfrac{1}{4} \cdot \tfrac{1}{4} \cdot \tfrac{1}{12} \\ &= \tfrac{1}{384} = 0.0026 \end{aligned}$$

a jav opačný ako:

$$\begin{aligned} P(\bar{c} \mid doc5) &= P(\bar{c}) \cdot \prod_{t \in d_t} P(t \mid \bar{c}) = \\ &= \tfrac{2}{4} \cdot P(Taiwan \mid \bar{c}) \cdot P(Taiwan \mid \bar{c}) \cdot P(Sapporo \mid \bar{c}) \\ &= \tfrac{2}{4} \cdot \tfrac{1}{6} \cdot \tfrac{1}{6} \cdot \tfrac{1}{4} \\ &= \tfrac{1}{288} = 0.0035 \end{aligned}$$

Zadanie **c)** od nás vyžaduje použitie Bernoulli Naive Bayes klasifikátorov. Tie sa počítajú podľa definície 24. Zároveň pre zadanie **d)** potrebujeme vypočítať nielen hodnoty $p(t \mid c)$ z testovacieho dokumentu, ale aj $p(t' \mid c)$ hodnoty, ktoré sa v testovacom dokumente nenachádzajú, ale sú v trénovacej množine. Tento postup je oproti Multinomickým klasifikátorom odlišný. Najprv určíme množinu M_d ako množinu termov, ktoré sú z trénovacej množiny, zároveň patria do c, ale nie sú v testovacom dokumente. Ide o termy $M_d = \{Taipei, Macao, Shanghai\}$

$$\begin{aligned} P(Taiwan \mid c) &= \tfrac{2+1}{2+2} = \tfrac{3}{4} \\ P(Sapporo \mid c) &= \tfrac{0+1}{2+2} = \tfrac{1}{4} \\ P(Taipei \mid c) &= \tfrac{1+1}{2+2} = \tfrac{1}{2} \\ P(Macao \mid c) &= \tfrac{1+1}{2+2} = \tfrac{1}{2} \\ P(Shanghai \mid c) &= \tfrac{1+1}{2+2} = \tfrac{1}{2} \end{aligned}$$

Ďalej vypočítame javy opačné ako

$$\begin{aligned} P(Taiwan \mid \bar{c}) &= \tfrac{1+1}{2+2} = \tfrac{1}{2} \\ P(Sapporo \mid \bar{c}) &= \tfrac{2+1}{2+2} = \tfrac{3}{4} \\ P(Taipei \mid \bar{c}) &= \tfrac{0+1}{2+2} = \tfrac{1}{4} \\ P(Macao \mid \bar{c}) &= \tfrac{0+1}{2+2} = \tfrac{1}{4} \\ P(Shanghai \mid \bar{c}) &= \tfrac{0+1}{2+2} = \tfrac{1}{4} \end{aligned}$$

čím sme splnili zadanie **c)**. Teraz vypočítame **d)** podľa definície 26 ako

$$\begin{aligned} P(c \mid d_t) &= P(c) \cdot \prod_{t \in d_t} P(t \mid c) \cdot \prod_{t' \in M_d} (1 - P(t' \mid c)) \\ &= \tfrac{1}{1} \cdot ((P(Taiwan \mid c) \cdot P(Sapporo \mid c))) \cdot \\ &\quad ((1 - P(Taipei \mid c)) \cdot (1 - P(Macao \mid c)) \cdot (1 - P(Shanghai \mid c))) \\ &= \tfrac{2}{4} \cdot (\tfrac{3}{4} \cdot \tfrac{1}{4}) \cdot ((1 - \tfrac{1}{2}) \cdot (1 - \tfrac{1}{2}) \cdot (1 - \tfrac{1}{2})) \\ &= \tfrac{3}{256} = 0.0117 \end{aligned}$$

$$
\begin{aligned}
P(\bar{c} \mid d_t) &= P(\bar{c}) \cdot \textstyle\prod_{t \in d_t} P(t \mid \bar{c}) \cdot \prod_{t' \in M_d} (1 - P(t' \mid \bar{c})) \\
&= \tfrac{2}{4} \cdot ((P(Taiwan \mid \bar{c}) \cdot P(Sapporo \mid \bar{c}))) \cdot \\
&\quad ((1 - P(Taipei \mid \bar{c})) \cdot (1 - P(Macao \mid \bar{c})) \cdot (1 - P(Shanghai \mid \bar{c}))) \\
&= \tfrac{1}{1} \cdot (\tfrac{1}{2} \cdot \tfrac{3}{4}) \cdot ((1 - \tfrac{1}{4}) \cdot (1 - \tfrac{1}{4}) \cdot (1 - \tfrac{1}{4})) \\
&= \tfrac{81}{1024} = 0.0791
\end{aligned}
$$

Teória, definície a výpočty sú popísane v kapitole 13 v [2].

Literatúra

[1] GHETAU, Vladimir. Variable byte code - how to?. In: *The Web Systems Engineering Blog: Stuff you can do on their servers* [online]. 2012-012-11 [cit. 2014-04-28]. Dostupné z: `http://websystemsengineering.blogspot.cz/2012/12/variable-byte-code-how-to.html`

[2] MANNING, Christopher D, Prabhakar RAGHAVAN a Hinrich SCHÜTZE. *Introduction to information retrieval* [online]. 1st pub. Cambridge: Cambridge University Press, 2008, xxi, 482 s. [cit. 2014-05-15]. ISBN 978-052-1865-715. Dostupné z: `http://nlp.stanford.edu/IR-book/pdf/irbookonlinereading.pdf`

[3] HIROSHI, Shimodaira. The UNIVERSITY OF EDINBURGH. *Text Classification using Naive Bayes*. 2014, 9 s. Dostupné z: `http://www.inf.ed.ac.uk/teaching/courses/inf2b/learnnotes/inf2b-learn-note07-2up.pdf`

[4] Simple online XPath tester. [online]. [cit. 2014-05-16]. Dostupné z `http://www.xpathtester.com/xpath`

[5] SOJKA, Petr, Hinrich SCHÜTZE, et al., *PV211: Úvod do získávání informací: IIR 3: Dictionaries and tolerant retrieval* [online]. 2014, 114 s. [cit. 2014-05-21]. Dostupné z: `http://www.fi.muni.cz/~sojka/PV211/p03dict.pdf`

[6] SOJKA, Petr, Hinrich SCHÜTZE, et al., *PV211: Úvod do získávání informací: IIR 5: Index compression* [online]. 2014, 114 s. [cit. 2014-05-31]. Dostupné z: `http://www.fi.muni.cz/~sojka/PV211/p05comp.pdf`