VŠB – Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra informatiky

Doporučovací systém

Recommender system

2016 Bc. Matúš Máčik

VŠB - Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra informatiky

Zadání diplomové práce

Student: **Bc. Matúš Máčik**

Studijní program: N2647 Informační a komunikační technologie

Studijní obor: 2612T025 Informatika a výpočetní technika

Téma: Doporučovací systém
Recommender System

Jazyk vypracování: čeština

Zásady pro vypracování:

Cílem této práce je prostudovat metody používané v oblasti doporučovacích systémů a vytvořit vlastní doporučovací systém buď pro zpravodajský portál nebo pro portál pracovních příležitostí.

1. Prostudujte problematiku doporučovacích systémů.
2. Popište data a vybrané metody.
3. Implementujte vybrané metody.
4. Vyhodnoťte získaná doporučení.

Seznam doporučené odborné literatury:

[1] Leskovec, Jure, Anand Rajaraman, and Jeffrey David Ullman. Mining of massive datasets. Cambridge University Press, 2014.
[2] Jannach, D., & Friedrich, G. (2011, July). Tutorial: Recommender Systems. In Proceedings of the International Joint Conference on Artificial Intelligence, Barcelona.
[3] Cleger-Tamayo, S., Fernández-Luna, J. M., & Huete, J. F. (2012). Top-N news recommendations in digital newspapers. Knowledge-Based Systems, 27, 180-189.

Formální náležitosti a rozsah diplomové práce stanoví pokyny pro vypracování zveřejněné na webových stránkách fakulty.

Vedoucí diplomové práce: **Mgr. Pavla Dráždilová, Ph.D.**

Datum zadání: 01.09.2015
Datum odevzdání: 29.04.2016

doc. Dr. Ing. Eduard Sojka
vedoucí katedry

prof. RNDr. Václav Snášel, CSc.
děkan fakulty

Prehlasujem, že som túto diplomovú prácu vypracoval samostatne. Uviedol som všetky literárne pramene a publikácie, z ktorých som čerpal.

V Ostrave 29. apríla 2016 .....

Súhlasím so zverejnením tejto diplomovej práce podľa požiadavkov čl. 26, odst. 9 Studijního a zkušebního řádu pro studium v magisterských programech VŠB-TU Ostrava.

V Ostrave 29. apríla 2016 .....

Vďaka patrí v prvom rade mojim rodičom Mgr. Eve Máčikovej a Ing. Ľubošovi Máčikovi, ktorí ma vždy podporovali v mojich cieľoch a snoch. Práve oni mi umožnili štúdium na vysokej škole podľa mojich predstáv. Za podporu ďakujem aj celej svojej rodine a priateľom. Ďakujem Mgr. Pavle Dráždilovej za odbornú pomoc a vedenie pri vypracovávaní nielen tejto diplomovej práce, ale aj predošlej bakalárskej práce.

Abstrakt

V súčasnosti, v dobe internetu, je bežný používateľ denne zaplavovaný množstvom informácií z rôznych zdrojov. Nielen objem, ale aj obsah a ich povaha vytvára efekt informačného zahltenia. Táto diplomová práca sa zaoberá problematikou vytvárania odporúčaní novinových článkov pre používateľov spravodajského portálu. Práca obsahuje prehľad základných metód pre odporúčacie systémy, popis spracovania neštruktúrovaných textových dát, získavania implicitného hodnotenia používateľov a odporúčacích algoritmov. Súčasťou diplomovej práce je návrh a implementácia hybridného odporúčacieho systému a jeho test v reálnom prostredí na portáli www.info.sk. Systém pozostáva z troch celkov a to zo sledovacieho kódu, ktorý zhromažďuje záznamy o aktivite používateľov, serverovej aplikácie s databázou a serveru pre výpočet odporúčaní.

Kľúčové slova: odporúčací systém, personalizácia, odporúčanie obsahu, správanie používateľa, web, novinové články

Abstract

Nowadays, in a time of internet, users are daily overloaded by amouts of an information from various sources. Not just the amount but also their content makes effect called information overload. This thesis deals with articles recommendations for users of news portal. The work includes an overview of basic methods for recommender systems, description of processing unstructured text data, collection of users implicit ratings and recommender algorithms. A part of this work is design and implementation of hybrid recommender system and its testing on real users of news portal www.info.sk. The system consists of three main parts: user activity tracking code, server application with database storage and computational server.

Key Words: recommender system, personalization, content recommendation, user behavior, web, news articles

Obsah

Zoznam použitých skratiek a symbolov

NRSys	–	News Recommender System
PHP	–	Hypertext Preprocessor
HTML	–	HyperText Markup Language
MySQL	–	Databázový relačný systém
API	–	Application Programming Interface
HTTP	–	Hypertext Transfer Protocol
URL	–	Uniform Resource Locator
SQL	–	Structured Query Language
SSL	–	Secure Sockets Layer
FTP	–	File Transfer Protocol

Zoznam obrázkov

Zoznam tabuliek

Zoznam výpisov zdrojového kódu

1 Úvod

Od dôb priemyselnej revolúcie sa informácia stala najhodnotnejším prvkom takmer pre všetky ľudské aktivity. V posledných dekádach umožnil rozvoj technológií a špeciálne rozšírenie internetu exponenciálny nárast prístupných informácií. Tým pádom sa hľadanie relevantných a uspokojivých informácií stalo náročnejším z časového aj rozhodovacieho hľadiska. Tento fakt je označovaný ako informačné zahltenie a opisuje stav, kedy máme príliš veľa informácií na robenie relevantných rozhodnutí alebo udržanie si obrazu o určitej téme. Úlohou hľadania relevantných informácií sa zaoberá výskum v oblasti získavania informácií (Information Retrieval). Táto oblasť vedy vyvinula niekoľko riešení pre problém informačného zahltenia ako sú inteligentní agenti, hodnotiace algoritmy, zhlukovacie algoritmy, hĺbková analýza dát (data mining) a odporúčacie systémy (Recommender Systems) [1].

Obsahom tejto diplomovej práce sú informácie a znalosti o odporúčacích systémoch a ich použitie pri návrhu a implemenácií vlastného odporúčacieho systému založeného na spojení prístupov content-based a collaborative filtering. Úlohou vytvoreného systému bude generovať odporúčania novinových článkov pre používateľov spravodajského portálu.

Cieľom práce je priblíženie základných techník v oblasti odporúčacích systémov a návrh a implementácia systému pre potreby spravodajského portálu. Vyvinutý systém bude schopný zbierať a analyzovať informácie o novinových článkoch a používateľoch ktorí ich čítajú. Tieto dáta bude systém následne využívať pri generovaní odporúčania teoreticky zaujímavého obsahu pre používateľov. Ďalším z cieľov je systém implementovať s ohľadom na výkon, teda rýchlosť generovania odporúčaní a nasadiť ho do testovania v ostrej prevádzke na používateľoch spravodajského portálu.

V prvej časti práce sú predstavené dva základné prístupy k tvorbe odporúčacích systémov a to collaborative filtering, ktorý vytvára odporúčania na princípe podobnosti používateľov a content-based filtering zameraný na hľadanie obsahu podľa preferencií používateľa. Ich výhody a slabé stránky vyúsťujú do spojenia v podobe hybridného systému. Ďalšia kapitola popisuje teóriu spracovania neštruktúrovaných textových dát a získavanie kľúčových slov z textu a návrh budúceho hybridného systému z pohľadu dát a použitých algoritmov pre výpočet odporúčaní. Následne je popísaný návrh a implementácia vyvíjaného systému z pohľadu softwarového inžinierstva. V predposlednej kapitole 6 je obsiahnutý spôsob nasadenia a testovania vyvinutého odporúčacieho systému v ostrej prevádzke na spravodajskom portáli. V závere je zhrnutý prínos, využitie a možnosti rozšírenia tejto diplomovej práce v budúcom vývoji.

Literatúra použitá pri tvorbe tejto práce je prevažne v anglickom jazyku, a preto neboli niektoré názvy z dôvodu zachovania terminológie prekladané do slovenčiny.

2 Odporúčacie systémy

V posledných rokoch exponenciálne narastá objem dostupných digitálnych informácií, elektronických zdrojov a on-line služieb. Informačné zahltenie vytvára potenciálny problém ako triediť a efektívne zobrazovať používateľovi relevantné informácie. Z tohto problému vyplýva potreba informácie filtrovať a predpovedať záujem používateľa o nové alebo doposiaľ nezobrazené informácie. Systémy tohto druhu sa nazývajú *odporúčacie systémy* [2]. Odporúčacie systémy sa používajú nielen na predchádzanie informačnému zahlteniu, ale aj pre marketingové aktivity, zdokonaľovanie vyhľadávania, zvyšovanie konverzie, udržanie pozornosti a podobne [3]. Tieto systémy typicky pracujú s dátami o *používateľoch* a *položkách*. Pod pojmom používateľ chápeme osobu používajúcu webové rozhranie nešpecifikovanej webovej stránky ktorá v sebe implementuje odporúčací systém. Používateľ vykonávaním rôznych aktivít na stránkach poskytuje o sebe informácie, ktoré je možné použiť pre zisťovanie jeho záujmov či potrieb. Používateľa definujeme ako prvok u z m prvkovej množiny používateľov $U = \{u_1, u_2, ..., u_m\}$. Položka je všeobecný pojem použitý pre označenie niečoho, čo systém odporúča používateľovi. Položku definujeme teda ako prvok i z n prvkovej množiny položiek $I = \{i_1, i_2, ..., i_n\}$.
Keď používateľ u_j ohodnotí položku i_k, vzniká *hodnotenie* $r_{j,k}$ pre vzťah používateľ-položka. Hodnotením používateľ vyjadruje svoju mieru záujmu, súhlasu alebo obľúbenosť položky. Samozrejme, povaha a význam hodnotenia sa môže meniť naprieč rôznymi typmi položiek. Toto vyjadrenie je možné reprezentovať pomocou úžitkovej (utility) funkcie f (vzorec 1) [2]:

$$f : U \times I \to R, \tag{1}$$

kde R je úžitková matica a $r_{j,k}$ je typicky reprezentované kladným celým alebo reálnym číslom v určitom rozmedzí. Pri veľkom počte používateľov a položiek sa dá predpokladať, že táto funkcia nie je známa pre celý priestor $U \times I$, ale je špecifikovaná iba na jeho podmnožinu ohodnotených položiek. Tým pádom je snahou odporúčacieho systému pre každého používateľa $u_j \in U$ odhadnúť úžitkovú funkciu $f(u_j, i_k)$ pre položku $i_k \in I$, kde $f(u_j, i_k)$ nie je známa [10].

	i_1	i_2	i_3	$\cdots$	i_n
u_1	1	1	0	$\cdots$	1
u_2			1	$\cdots$	
u_3	1	1	0	$\cdots$	0
$\vdots$	$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$
u_m	0		1	$\cdots$	1

Tabuľka 2: Matice funkcie $f : U \times I \to R$, kde $r \in \{0, 1\}$ a prázdne miesta predstavujú neohodnotené položky.

Odporúčacie systémy sa podľa prístupu delia do dvoch hlavných kategórií na *content-based filtering* (podkapitola 2.3.1) a *collaborative filtering* (podkapitola 2.3.2) [9]. Okrem spomenutých prístupov literatúra uvádza aj odporúčacie systémy na základe vedomostí (knowledge-based)

alebo demografických údajov (demographic recommender), ktoré sú pre nami vyvíjaný systém nevhodné a z toho dôvodu sa v texte nespomínajú. Každá metóda pristupuje k tvorbe odporúčaní iným spôsobom, a preto sa líšia aj v slabých a silných stránkach. Spojením výhod rôznych metód je možné niektoré z problémov obmedziť alebo eliminovať v podobe *hybridného* odporúčacieho systému.

2.1 Odporúčacie systémy v praxi

Po krátkom úvode do problematiky si pre lepšiu predstavu o odporúčacích systémov ukážeme príklady ich použitia v praxi, ktoré sú jasným dôkazom o ich využiteľnosti a prínose. V dnešnej dobe tieto systémy používajú najmä najväčší svetový hráči na poli internetových služieb ako Facebook, YouTube, Google, Netflix, Amazon, Spotify a mnoho ďalších. Tieto spoločnosti vďaka obrovskému využívaniu ich služieb získavajú dáta o používateľoch a obsahu, ktoré dokážu úspešne analyzovať a využívať nielen v oblasti odporúčaní obsahu, ale aj vo zvyšovaní kvality svojich služieb a ziskov.

2.1.1 YouTube

Služba YouTube sa od jej vzniku v roku 2005 stala svetovo najväčšou a najpopulárnejšou online video komunitou na svete. Používatelia denne prichádzajú na YouTube hľadať, sledovať a zdieľať originálne videá. Každý deň je zaznamenaných viac ako 4 bilióny pozretí, čo predstavuje stovky miliónov minút strávených sledovaním videí. Každú hodinu taktiež pribudne nový obsah rádovo v stovkách hodín [37]. Obrovské množstvo videí teda vytvára potrebu pomôcť používateľom pri nachádzaní kvalitného, zaujímavého a relevantného obsahu. Odporúčací systém v YouTube sa predovšetkým zameriava na prihlásených a dlhodobých používateľov. Jeho prioritou je používateľa zabávať, udržať jeho záujem a reflektovať jeho aktivitu na stránkach. Odporúčania sú generované na základe správania používateľa (vzhliadnutie videa, pridanie k obľúbeným, odoberanie autora, hodnotenie videa), metadát a obsahu videa [18].

2.1.2 Netflix

Netflix je služba pre streamovanie seriálov a filmov cez internet. V oblasti odporúčacích systémov sa dostala značne do povedomia vyhlásením súťaže o 1 milión dolárov. Úlohou bolo vytvoriť odporúčací systém s presnosťou odporúčaní o 10% vyššou oproti ich dovtedajšiemu systému Cinematch a znížiť RMSE (root-mean-square error) z pôvodnej hodnoty 0.9514 na 0.8572. RMSE je v skratke stredná kvadratická chyba, ktorá sa používa ako ukazovateľ presnosti odporúčaní. Cena bola odovzdaná v roku 2009 tímu BellKor's Pragmatic Chaos [38]. Netflix tvrdí, že až 75% filmov a seriálov v ich službe sledujú používatelia na základe odporúčaní [8]. Správne odporúčanie obsahu je pri 65 miliónoch platiacich používateľoch a 100 miliónov hodín prehraných filmov a seriálov denne kľúčom k úspechu a udržaniu si zákazníkov. Výskum v Netflixe ukázal, že používateľ stráca záujem už po 60 - 90 sekundách vyhľadávania, pričom si prezrie 10 - 20 titulov,

z čoho priemerne 3 detailnejšie. Z toho dôvodu je ich snahou čo najlepšie analyzovať správanie používateľa a nespoliehať sa iba na explicitné hodnotenia, vďaka čomu môžu maximalizovať mieru personalizácie a efektivity vyhľadávania. Netflix považuje odporúčací systém za jadro svojho biznisu a pripisuje mu veľkú časť úspechu [19].

2.1.3 Amazon

Amazon patrí medzi najväčšie internetové obchody na svete a ako jeden z prvých použil odporúčací systém v oblasti online nakupovania. Amazon ponúka sortiment s miliónmi produktov širokej skupine zákazníkov, z čoho vychádza aj potreba personalizovaných odporúčaní pre každého používateľa. Odporúčania sa na Amazone rapídne menia s každým kliknutím používateľa na ďalší produkt. Amazon využíva odporúčania ako nástroj cielenej reklamy nielen na svojich stránkach, ale aj v mnohých emailových kampaniach. Keďže existujúce odporúčacie algoritmy neboli pre potreby Amazonu dostatočne škálovateľné vzhľadom na veľký počet položiek a desiatky miliónov zákazníkov, museli vyvinúť vlastnú techniku. Amazon používa techniku *item-to-item* collaborative filtering, ktorá zvláda rozsiahle objemy dát, pričom produkuje vysoko kvalitné odporúčania. Princíp odporúčania spočíva v náchádzaní podobných produktov k produktom ktoré používateľ kúpil, ohodnotil, alebo zobrazil. Z nájdenej skupiny podobných produktov systém kombinuje zoznam výsledných odporúčaní. Podobnosť je založená na kombináciách produktov pri už uskutočnených nákupoch ostatných používateľov, pričom sa vychádza z vopred vypočítanej tabuľky podobných produktov [30].

2.2 Dáta potrebné pre odporúčacie systémy

Ako bolo v úvode tejto kapitoly naznačené, odporúčacie systémy pracujú s troma základnými typmi dát. Sú nimi informácie o položkách, používateľoch a ich hodnoteniach pre jednotlivé položky systému. Táto podkapitola v skratke popisuje tieto dáta, aby sme objasnili pojmy používané v ďalšom texte.

2.2.1 Položky

Položka je všeobecný pojem pre niečo, čo systém odporúča používateľom. Položkami teda možu byť produkty, videá, filmy, knihy, hudba, ostatní používatelia systému a iné. V prípade nami vyvíjaného systému sú položky reprezentované novinovými článkami. To, aké informácie musí systém o položke evidovať určujú vstupné parametre jednotlivých metód, ktoré sú posísané v kapitole 2.3.

2.2.2 Hodnotenie používateľov

Odporúčacie systémy pre vytváranie odporúčaní potrebujú vedieť, čo používateľ považuje za zaujímavé, čo sa mu páči alebo nepáči. Tieto údaje je možné zbierať pomocou dvoch typov hodnotení. *Explicitné* hodnotenie zadáva používateľ priamo. V tomto prípade teda používateľ musí

položku preskúmať, posúdiť a priradiť jej hodnotu zo stupnice. Tento proces však kladie nárok na kognitívne schopnosti používateľa a vyžaduje jeho aktivitu. Typickým príkladom je hodnotenie pomocou hviezdičiek, stupnice od 1 do 5, výberom emotikony, palec nahor alebo nadol, či textová recenzia. Použitím *implicitného* hodnotenia sa odstránia spomenuté nároky na používateľa a tieto úkony sa snaží nahradiť systém pomocou zaznamenávania jeho aktivít a interakcií so systémom. Na druhej strane však vznikajú nároky na výpočtový a úložný priestor, keďže objem implicitných dát môže nepomerne narastať. V konečnom dôsledku je táto situácia výhodnejšia oproti riedkym explicitným hodnoteniam aj v prípade, že výpovedná hodnota implicitných dát môže byť čiastočne skreslená. Implicitné hodnotenie predstavuje napríklad história objednávok, história zobrazených položiek, pridanie k obľúbeným alebo čas strávený prezeraním položky [7].

2.2.3 Profil používateľa

V texte sa bude často vyskytovať pojem profil používateľa. Ak chceme používateľovi odporúčať položky, ktoré odpovedajú jeho záujmom, potrebujeme tieto záujmy nejakým spôsobom rozumne interpretovať. Pod pojmom profil používateľa je teda zvyčajne chápaná štruktúrovaná reprezentácia záujmov a potrieb používateľa [16]. O tvorbe profilu hovoríme najmä pri metóde content-based filtering, pretože collaborative filtering ako profil používateľa berie jeho vektor hodnotení pre jednotlivé položky. Na modelovanie profilu používateľa pre content-based filtering je možné použiť napríklad algoritmy ako:

- rozhodovacie stromy a pravidlá,
- spätná väzba relevancie a Rocchio algoritmus,
- lineárne klasifikátory,
- pravdepodobnostné metódy a Naivná Bayesova metóda.

Vytvorený profil je spolu s vektormi položiek vstupom pre ďalšie algoritmy, ktorých výsledkom je odhad pravdepodobnosti, s akou sa bude položka používateľovi páčiť, alebo numerická hodnota vyjadrujúca mieru záujmu používateľa o položku[27]. Kvalita profilu má zásadný dopad na úspešnosť výsledných odporúčaní. Ako problematické sa ukazuje vygenerovanie počiatočného profilu pre nového používateľa, teda problém studeného štartu a správna aktualizácia profilu v čase [17]. V nami vyvíjanom systéme bude pre zostavovanie profilu používateľa použitý Rocchio algoritmus popísaný v podkapitole 2.3.1.1. Podrobnejší popis vymenovaných algoritmov je obsiahnutý v Pazzaniho práci [27].

2.2.4 Problém studeného štartu

Studený štart (cold-start problem) je problémom nedostatku vstupných dát. Ten sa môže týkať nielen nových používateľoch ale aj nových položiek v systéme. Keď používateľ príde prvýkrát do

odporúčacieho systému, systém o ňom nič nevie a tým pádom nie je schopný pre takéhoto používateľa generovať odporúčania. Studený štart nových položiek sa prejavuje u systémov, ktoré odporúčajú položky podľa toho, aké hodnotenia dostávajú od používateľov. V tomto prípade nové alebo málo hodnotené položky nemajú šancu byť odporúčané [20]. Systémy pracujúce s implicitnými dátami efekt studeného štartu odbúravajú rýchlejšie ako systémy postavené len na explicitných hodnoteniach. Riešením pre tento problém sa ukazuje spojenie rôznych odporúčacích metód a teda vytvorenie hybridného odporúčacieho systému, ktorý popisujeme v podkapitole 2.4.

2.3 Metódy používané v odporúčacích systémoch

2.3.1 Content-based filtering

Content-based filtering vytvára odporúčania na základe porovnávania položiek s *profilom používateľa* [4]. Aby bolo porovnávanie možné, je potrebné položky analyzovať a popísať ich skupinou atribútov. V množstve prípadov sa dajú položky jednoznačne označiť atribútmi, ktoré vyjadrujú význam položky ako napr. veľkosť operačnej pamäte, počet jadier či výrobca v prípade počítačov. Problém analýzy a získania atribútov nastáva pri textových dokumentoch, hudbe či iných neštruktúrovaných položkách, kde atribúty typu autor, žáner alebo jazyk nie sú dostatočne popisné. Obsah takýchto položiek je teda potrebné vhodným spôsobom strojovo spracovať a pripraviť pre následné porovnávanie. Okrem získania atribútov sa zvyčajne určuje aj ich významnosť pre položku. Nájdené atribúty a ich hodnoty sú zväčša reprezentované ako *vektory* v *m-rozmernom priestore*, kde m je počet všetkých jedinečných atribútov pre množinu položiek. Profil používateľa je tiež vektor vytvorený z atribútov ním ohodnotených položiek v minulosti [11]. Odporúčané sú teda doposiaľ používateľom neohodnotené položky *podobné* s tými, ktoré ohodnotil pozitívne [5]. Dôležitou úlohou systému je potom správne analyzovať obsah každej položky a získať z nej vektor najviac popisných atribútov a ich váhu pre danú položku. V prípade vyvíjaného systému budú atribúty reprezentované *termami*, čo sú v podstate kľúčové slová, a ich *tf-idf váhou*, ktorá vyjadruje dôležitosť termu pre daný novinový článok v rámci celého korpusu článkov. Spracovanie textu, výber a váženie termov popisuje kapitola 3. Podobnosť profilu s položkou, podobnosť dvoch položiek alebo dvoch profilov sa teda určuje ako podobnosť vektorov v priestore. Najrozšírenejšou funkciou [11] pre určenie podobnosti vektorov je kosínusová podobnosť (vzorec 2):

$$w_{x,y} = \frac{\sum_{i=1}^{m} x_i y_i}{\sqrt{\sum_{i=1}^{m} x_i^2}\sqrt{\sum_{i=1}^{m} y_i^2}}, \tag{2}$$

kde x_i a y_i sú prvky porovnávaných vektorov x a y, a m predstavuje dimenziu vektorov. S približovaním kosínusu uhla vektorov k 0 narastá miera ich podobnosti. Nájdenie odporúčaní pre používateľa prebieha porovnaním všetkých vektorov položiek s vektorom profilu. Výsledkom je

zoznam n najviac podobných položiek k profilu používateľa.

Čisté content-based systémy majú určité výhody, ktoré často chýbajú práve iným druhom odporúčacích metód [9]:

- **Nezávislosť používateľa** - odporúčania sú odvodené len na základe profilu aktívneho používateľa. Systém sa teda nespoliehať na vzťahy medzi používateľmi s podobným vzorcom hodnotenia.
- **Transparentnosť** - vysvetlenie ako systém pracuje môže byť poskytnuté priamym vymenovaním atribútov alebo položiek, na základe ktorých prebehol výber odporúčaných položiek. Táto výhoda môže prispieť v dôvere používateľa k odporúčaciemu systému.
- **Studený štart nových položiek** - systém je schopný odporúčať aj nové alebo málo hodnotené položky. Preto je táto metóda vhodná do prostredia, kde neustále pribúdajú nové položky [6], čo je aj prípad vyvíjaného systému.

Content-based filtering má samozrejme aj svoje slabiny, avšak na niektoré z nich existujú pomerne jednoduché riešenia. Medzi nevýhody patria napríklad:

- **Limitovaná analýza položiek** - content-based filtering má prirodzene limitovaný počet a typ atribútov automaticky alebo manuálne priradených k položkám. Výber a správna reprezentácia atribútov položiek sa líši naprieč rôznymi doménami [4, 5]. Žiaden systém tohto typu nedokáže ponúkať vhodné odporúčania bez dostatočných informácií o položkách.
- **Prílišná špecializácia** - ak systém odporúča položky s najvyšším skóre k profilu používateľa, môže nastať situácia keď sú používateľovi predkladané iba veľmi podobné položky tým, ktoré už ohodnotil. Často sa tento problém rieši určitou formou náhodnosti [5].
- **Studený štart nového používateľa** - o novom používateľovi systém nič nevie. Pre pochopenie záujmov používateľa a spoľahlivé odporúčanie však musí systém najskôr nazbierať dostatočné množstvo hodnotení od používateľa.

2.3.1.1 Profil používateľa pre content-based filtering

Spätná väzba relevancie a Rocchio algoritmus je široko používaný algoritmus, ktorý pomáha vylepšovať profil používateľa na základe jeho spätnej väzby pre potreby metódy content-based filtering. Ak používateľ označil niektoré položky za relevantné a iné naopak nerelevantné, tak sa algoritmus postará o úpravu pôvodného profilu. Tento algoritmus bol vyvinutý práve pre potreby a prácu vo vektorovom priestore a je označovaný za jeden z najlepších algoritmov pre spätnú väzbu a tvorbu profilu používateľa [28]:

$$P_m = \alpha \cdot P_o + \beta \cdot \sum_{I_i \in I_r} \frac{I_i}{|I_r|} - \gamma \cdot \sum_{I_j \in I_n} \frac{I_j}{|I_n|}, \tag{3}$$

kde $I_r \in I$ je množina relevantných a $I_n \in I$ množina nerelevantných položiek z množiny všetkých položiek I. P_m je modifikovaný profil a P_o je originálny profil používateľa pred aktualizáciou. Parametre α, β, γ ovplyvňujú dopad pôvodného vektoru a dvoch vypočítaných vektorov pre relevantné a nerelevantné dokumenty. Použitím vzorca sa modifikovaný vektor profilu približuje k centroidu relevantných dokumentov a zároveň vzďaľuje od centroidu tých nerelevantných [29]. Ak neexistuje originálny profil P_o, môže byť namiesto vzorca (3) použitý vzorec (4):

$$P_m = \sum_{I_i \in I_r} \frac{I_i}{|I_r|} - \sum_{I_j \in I_n} \frac{I_j}{|I_n|}, \tag{4}$$

ktorý maximalizuje rozdiel medzi priemerným skóre relevantných a nerelevantných položiek [28]. Použitie vytvoreného profilu spočíva v jeho porovnaní pomocou kosínusovej podobnosti (vzorec 2) s položkami, ktoré používateľ doposiaľ nevidel alebo neohodnotil. Odporúčania sú vygenerované ako zoznam *top-N* položiek s najväčšou mierou kosínusovej podobnosti.

2.3.2 Collaborative filtering

Základným predpokladom metódy collaborative filtering je tvrdenie, že ak používateľia u a v ohodnotia n položiek podobne, alebo majú podobné správanie (napr. nakupovanie, počúvanie hudby, pozeranie videí), budú podobne hodnotiť alebo sa správať aj pri iných položkách. Metóda collaborative filtering pre výpočty používa databázu hodnotení používateľov. Prevedením databázy hodnotení do matice používateľ-položka vznikne v matici množstvo prázdnych miest, čiže neohodnotených položiek, pre ktoré sa algoritmus snaží zistiť možné hodnotenie alebo pravdepodobnosť, s akou by sa daná položka mohla používateľovi páčiť [4]. Táto metóda nevyžaduje analýzu položiek, pretože žiaden z ich atribútov nie je na vstupe [21]. Aj keď je metóda collaborative filtering považovaná za pomerne výkonnú a odporúčania generuje s vysokou presnosťou, má svoje slabé miesta:

- **Riedkosť dát** - ak odporúčací systém pracuje s množstvom položiek a používateľov, matica vzťahu používateľ-položka je extrémne riedka, čo má zásadný dopad na kvalitu odporúčaní.
- **Škálovateľnosť** - s rýchlym nárastom používateľskej základne a počtu položiek vzrastá aj výpočtová a priestorová náročnosť.
- **Studený štart** - collaborative filtering trpí nielen problémom nového používateľa, ale aj problémom novej alebo málo hodnotenej položky.
- **Zvýhodňovanie** (Shilling attacks) - používateľ zámerne pozitívne hodnotí svoje položky, pričom negatívne hodnotí položky konkurencie s účelom zvýhodniť pozíciu a dosiahnuť častejšie odporúčanie svojich položiek. Toto správanie je možné pozorovať hlavne vo sfére online obchodov, kde môžu svoje produkty či služby ponúkať samotní používatelia [23].

Na rozdiel od metódy content-based filtering, ktorá odporúča podobné položky, táto metóda môže odporúčať aj položky s úplne nesúvisiacim obsahom. Collaborative filtering združuje v sebe niekoľko rôznych prístupov a algoritmov pre tvorbu odporúčaní, ktoré sú stručne zhrnuté v nasledujúcich podkapitolách.

2.3.2.1 Metóda najbližších susedov

Teoretický základ metódy najbližších susedov vychádza z predpokladu, že ak existuje používateľ alebo skupina používateľov s podobnými záujmami ako aktívny používateľ u, môžeme predpokladať, že položky, ktoré sa páčili podobným používateľom, sa s nejakou pravdepodobnosťou môžu páčiť aj používateľovi u [17]. Metóda založená na porovnávaní používateľov vníma profil používateľa ako vektor hodnotení ktoré priradil jednotlivým položkám. Namiesto odporúčania položiek podobných tým, ktoré používateľ ohodnotil v minulosti, sú odporúčané tie, ktoré sa páčili iným, jemu podobným používateľom. Zvyčajne sa pre každého používateľa nájde takzvaných *k-najbližších susedov* (k-Nearest Neighbors), s ktorými má najviac podobný vzorec hodnotenia rovnakých položiek. Odporúčania pre používateľom neohodnotené položky sú vypočítané na základe kombinácie hodnotení jeho najbližších susedov, ktorí tieto položky už ohodnotili [13]. Pre výpočet podobnosti používateľov je možné použiť viacero metrík, no často používanou je práve Pearsonova korelácia (vzorec 5):

$$w_{u,v} = \frac{\sum\limits_{i \in I}(r_{u,i} - \bar{r}_u)(r_{v,i} - \bar{r}_v)}{\sqrt{\sum\limits_{i \in I}(r_{u,i} - \bar{r}_u)^2}\sqrt{\sum\limits_{i \in I}(r_{v,i} - \bar{r}_v)^2}}, \tag{5}$$

kde $i \in I$ sú indexy položiek ohodnotených oboma používateľmi u a v, $\bar{r}_u$ a $\bar{r}_v$ je priemerné hodnotenie spoločne ohodnotených položiek pre daného používateľa [9]. Výsledkom Pearsonovej korelácie je číslo v rozmedzí -1 až 1, ktoré vyjadruje tendenciu spoločného vývoju dvoch číselných množín spárovaných jedna k jednej. Ak je tendencia bližšie k 1, potom prvok jednej množiny rastie podobne ako rovnaký prvok druhej množiny a opačne [21]. Pre výpočet možného ohodnotenia položky i aktívnym používateľom u stačí použiť vážený priemer (vzorec 6) nad všetkými hodnoteniami pre i od podobných používateľov $v \in U$ [9]:

$$P_{u,i} = \bar{r}_u + \frac{\sum\limits_{v \in U}(r_{v,i} - \bar{r}_v) \cdot w_{u,v}}{\sum\limits_{v \in U}|w_{u,v}|}. \tag{6}$$

Popísaný postup tvorby odporúčaní je význačný pre tzv. odporúčanie založené na používateľoch, označované tiež ako *user-based* collaborative filtering. Druhou možnosťou pri tvorbe odporúčaní na princípe najbližších susedov je odporúčanie podľa podobných položiek, *item-based* collaborative filtering. Táto metóda nehľadá podobných používateľov, ale sa naopak zameriava na nájdenie podobných položiek. Každá položka je popísaná vektorom hodnotení, ktoré obdržala od rôznych používateľov. Dimenzia tohto vektoru závisí od celkového počtu používateľov. Úlohou algoritmu

je nájsť pre množinu položiek $I_u \in I$ ohodnotených používateľom u, množinu podobných položiek $S_u \in I$. Množina S_u je nájdená porovnávaním vektorov položiek z I_u a zvyšných položiek v databáze pomocou kosínusovej podobnosti, Pearsonovej korelácie alebo inej podobnostnej funkcie. Pravdepodobné hodnotenie položky $s_i \in S_u$ pre používateľa u sa vypočíta napr. ako vážený priemer hodnotení ostatných používateľov pre túto položku [22]. Metóda podobných položiek bola vyvinutá z dôvodu obmedzenej škálovateľnosti u metódy podobných používateľov.

2.3.2.2 Metódy založené na modeloch

Ďalšiu skupinu algoritmov pre collaborative filtering tvoria algoritmy vytvárajúce modely (model-based collaborative filtering). Ich dizajn a vývoj dáva možnosť systému naučiť sa z trénovacích dát rozpoznávať komplexné vzorce a na základe naučeného modelu vytvárať inteligentné odporúčania pre reálne dáta. Pre túto úlohu zvyčajne slúžia klasifikačné algoritmy, regresívne modely alebo rozklady matíc. Metódy založené na tvorbe modelov dosahujú výrazne lepšiu škálovateľnosť a presnosť odporúčaní. Problematika týchto metód je obsiahla, a keďže v našej práci takéto metódy nepoužívame, nebudeme ich ani popisovať. Záujemca nájde podrobnejší náhľad do tejto komplexnej problematiky v práci Xiaoyuana [9], ktorá obsahuje aj celkový prehľad o metóde collaborative filtering.

2.4 Hybridný odporúčací systém

Hybridný odporúčací systém vhodným spôsobom kombinuje viacero možných metód pre dosiahnutie lepšieho výsledku. Vytvorenie hybridného systému skladajúceho sa z content-based a collaborative filtering môže viditeľne zvýšiť kvalitu odporúčacieho systému. Ako ukazujú výsledky práce [12], hybridný systém dokáže generovať výrazne kvalitnejšie výsledky ako každý z prístupov samostatne. Content-based metóda zvláda vytvárať odporúčania pre studený štart používateľov aj položiek [14]. Collaborative filtering zvyčajne poskytuje presnejšie hodnotenia, ale zlyháva na studenom štarte položiek.
V práci Burkeho z roku 2002 [15] je popísaných sedem spôsobov ako spojiť viaceré techniky do hybridného systému:

- **váženie** - skóre viacerých techník je číselne kombinované,
- **prepínanie** - systém vyberá aktuálne najvhodnejšiu odporúčaciu techniku,
- **mixovanie** - odporúčania z viacerých techník sú prezentované naraz,
- **rozšírenie vlastností** - výstup z jednej techniky je vstupom pre ďalšiu techniku,
- **kombinovanie vlastností** - vlastnosti získané z rôznych znalostných zdrojov sú kombinované a použité v jednom algoritme,

- **kaskádovanie** - techniky s nižšie stanovenou prioritou vylepšujú výsledky techník s vyššou prioritou,
- **meta-level** - jedna technika sa naučí model, ktorý je vstupom pre ďalšiu techniku.

3 Spracovanie textových dát

Spracovaním veľkého množstva dát reprezentovaných v podobe kníh, novinových článkov či iných textových *dokumentov* sa zaoberá oblasť počítačovej vedy nazývanej Information Retrieval. Textové dokumenty predstavujú neštruktúrované dáta. Je teda vhodné pre efektívnejšie a rýchlejšie spracovanie, či rôzne analýzy takýchto dát vytvoriť index ich obsahu. Ako vhodným riešením sa ukazuje reprezentácia indexu v podobe *vektorového modelu*. Samotné spracovanie dokumentov a vyťaženie správnych slov pre indexáciu, tzv. *termov* má za úlohu *lexikálna analýza* [31].

3.1 Vektorový model

Hlavnou myšlienkou vektorového modelu je reprezentácia každého dokumentu v kolekcií ako bodu vo viacrozmernom priestore (vektor vo vektorovom priestore), kde termy predstavujú dimenzie a hodnoty vektoru napr. *váhu termov* pre daný dokument. Body nachádzajúce sa blízko seba v tomto priestore sú sémanticky podobné a naopak. Vektory zapísané do matice vytvárajú tzv. maticu *termov v dokumentoch*. Pre vytvorenie vektorového modelu je potrebné dokumenty spracovať a vhodným spôsobom z nich vybrať termy, ktoré budú tieto dokumenty popisovať [34]. Vytvorením matice termov v dokumentoch vo väčšine prípadov vzniká riedka matica. Problém veľkej dimenzionality je možné riešiť indexáciou latentnej sémantiky (Latent Semantic Indexing), ktorá využíva matematickú metódu singulárny rozklad (Singular Value Decomposition) [33].

3.2 Term

Všeobecne je možné za term považovať akúkoľvek nedeliteľnú jazykovú jednotku, ktorá nemusí byť slovom z abecedy jazyka. Term môže byť teda označený aj ako kľúčové slovo, pojem, výraz či termín. Pre indexáciu nie je vhodné používať morfologické tvary slov, ale ich základný tvar, respektíve koreň. Ak chceme indexovať napríklad slová počítače, počítačový a pod., za term zvolíme slovo počítač.

3.3 Lexikálna analýza

Automatické spracovanie textových dokumentov sa nazýva lexikálna analýza. Lexikálna analýza prevádza vstupnú postupnosť znakov na slová alebo termy. Je to proces, kde sa na vstupe analýzy nachádza dokument s textom v jeho úplnej podobe a výstupom je multimnožina slov, ktoré môžeme považovať za indexačné jednotky, teda termy. Tento proces spracovania podlieha niekoľkým krokom, ktorých implementácia je rôzna a závisí od potrieb nadradeného systému. Počas nich môžu byť z textu odstránené nežiadúce znaky (čísla, interpunkcia a pod.), často frekventované slová (stopwords) a nevýznamové slová ako spojky, citoslovcia a iné. Z takto upraveného textu sa pomocou *lematizácie* vyfiltrujú výsledné termy.

3.4 Lematizácia

Lemma je označenie pre základný tvar slova [32]. Lematizácia je teda postup, ktorý sa zaoberá hľadaním koreňa slova z jeho morfologických tvarov. Niektoré štúdie ukazujú, že celkový počet indexovaných termov sa lematizáciou zredukuje na 25-30% celkového objemu [31]. Existujú dva základné prístupy k lematizácií. *Algoritmická* hľadá koreň slova pomocou algoritmu na základe gramatických pravidiel jazyka. Gramatika niektorých jazykov je však natoľko zložitá, že algoritmická lematizácia môže poskytovať nepresné výsledky. *Slovníková* lematizácia je založená na databáze koreňov a k nim priradených morfologických tvaroch. Tento prístup je náročnejší z hľadiska vytvorenia databázy, no v prípadoch zložitých jazykov takmer nevyhnutný.

3.4.1 Slovenský jazyk

Slovenčina patrí medzi jazyky s bohatou morfológiou, kde sa tvar slova mení podľa významu. Táto zložitosť jazyka vyúsťuje k obrovskému množstvu výnimiek. Ukázalo sa, že nie je možné realizovať zachytenie gramatiky slovenského jazyka do exaktných pravidiel. Vhodným spôsobom sa teda ukazuje použitie lematizácie pomocou databázy vytvorenej na základe Slovníku slovenského jazyka [32].

3.5 Frekventované slová

Prakticky takmer všetky dokumenty obsahujú niektoré totožné slová a ich rozlišovacia hodnota je malá. V bežných textoch tak 20-30% všetkých slov tvoria frekventované slová nevhodné pre indexovanie. Ich eliminácia pri indexovaní urýchli spracovanie, zmenší rozsah indexovej databázy a pritom neovplyvní výsledok algoritmov pre spracovanie indexovej databázy. Typickou ukážkou takýchto slov sú spojky (a, alebo, atď.), ukazovacie zámená (on, ona, atď.) a iné.

3.6 Váženie termov

Ako bolo spomenuté, pomocou termov je možné popísať význam dokumentu. Každý term však môže mať pre rôzne dokumenty v korpuse inú váhu. Pre určovanie významu termu sa používajú hodnotiace funkcie. Za najjednoduchšiu a pomerne rozšírenú je považovaná funkcia *tf-idf* (term frequency × inverted document frequency) [35]:

$$w_{i,j} = tf_{i,j} \times log\left(\frac{N}{df_i}\right), \tag{7}$$

kde $w_{i,j}$ je váha pre term i v dokumente j, N je počet všetkých dokumentov v korpuse, $tf_{i,j}$ je počet výskytov termu i v dokumente j normalizovaný počtom všetkých termov v dokumente j a df_i je počet dokumentov, v ktorých sa vyskytuje term i [36].

4 Návrh odporúčacích algoritmov pre spravodajský portál

V tejto kapitole sú popísané návrhy jednotlivých komponentov a ich využitie pri vytvorení hybridného odporúčacieho systému pre spravodajský portál. Novinové články v mnohých prípadoch rýchlo strácajú svoju aktuálnosť a z toho dôvodu sme sa rozhodli odporúčať iba články v rozmedzí 7 dní. Spôsob využitia tohto časového údaju je popísaný pre každý z komponentov osobitne, pretože v rôznych implementáciách má inú úlohu a dopad na generovanie odporúčaní.

4.1 Hodnotenie používateľov

Ak položky v odporúčacom systéme rýchlo pribúdajú a záujem používateľov o položky klesá s ich strácajúcou sa aktuálnosťou, spoľahnutie sa na explicitné hodnotenie neprichádza v úvahu. Pre hodnotenie položiek sme teda zvolili implicitné hodnotenie, ktoré nevyžaduje aktivitu používateľa. Vychádzame z práce [26], ktorá sa zaoberá koreláciou medzi dĺžkou času strávenou čítaním článku a explicitným hodnotením, ktoré následne zadávali čitatelia. Dĺžka čítania sa teda ukazuje ako vhodný indikátor *miery záujmu* používateľa o tému obsiahnutú v novinovom článku. Ako uvádza štúdia [25] zaoberajúca sa rýchlosťou čítania v 17 svetových jazykoch, priemerná rýchlosť čítania je 184 slov za minútu. Na základe tejto hodnoty môžeme vyjadriť $r_{u,i}$, teda záujem používateľa u o položku i na stupnici 0 - 5 ako (vzorec 8):

$$r_{u,i} = \frac{t_{u,i}}{c_i \cdot \alpha} \cdot 5, \tag{8}$$

kde $t_{u,i}$ je dĺžka čítania článku i používateľom u v sekundách, c_i je počet slov v článku i a $\alpha = 0,326$ je priemerná rýchlosť čítania jedného slova v sekundách. Ak je hodnota $r_{u,i} > 5$, potom $r_{u,i} = 5$. Mieru záujmu budeme v texte aj naďalej označovať ako hodnotenie používateľa.

4.2 Content-based filtering

Odporúčanie založené na analýze obsahu položiek a tvorbe používateľského profilu bude v našom systéme použité až v 3 odporúčacích schémach:

Odporúčanie podobných položiek bude určené primárne novým používateľom. Odporúčaných bude n najviac podobných článkov pre aktuálne zobrazený článok i. Používateľ tak bude môcť zostať v rovnakej alebo príbuznej tematike. Princíp tohto typu odporúčaní spočíva vo vopred vypočítanej databáze podobných článkov za $\pm$ 7 dní od dátumu pridania článku i.

Odporúčanie podľa histórie zobrazených článkov používateľom vychádza z predošlej schémy. Rozdiel je v tom, že algoritmus nezobrazuje odporúčania pre aktuálne zobrazený článok, ale pre každý článok z posledných m zobrazených článkov používateľom u nájde jeden podobný článok a tak vyskladá zoznam n odporúčaní. Tento spôsob odporúčania neberie

v úvahu hodnotenie predošlých článkov. Výhodou je taktiež vopred vypočítaná databáza podobných článkov, z ktorej vyplýva okamžitá reakcia na aktivitu používateľa.

Odporúčanie podľa profilu je určené pre tých používateľov, ktorí dosiahli minimálne 10 a viac zobrazených článkov. Zostavovanie profilu používateľa prebieha použitím Rocchio algoritmu (vzorec 3). Profil nie je potrebné udržiavať trvalo v databáze, pretože sa dá vyskladať z histórie jeho hodnotení a vektorov jednotlivých článkov, ktoré prečítal. Keďže Rocchio algoritmus je založený na aktualizácií profilu podľa relevancie článkov pre používateľa, za relevantné články sme označili tie, ktoré systém podľa dĺžky čítania používateľa ohodnotil na 1,5 - 5 bodov. Využitím kosínusovej podobnosti (vzorec 2) porovnáme články pridané za posledných 7 dní s profilmi používateľov a pre každý profil zostavíme zoznam 10 najviac odpovedajúcich článkov, ktorý je udržiavaný v databáze. Profil používateľa a zoznam jeho odporúčaní je vypočítavaný offline, preto odporúčania nemusia byť vždy aktuálne.

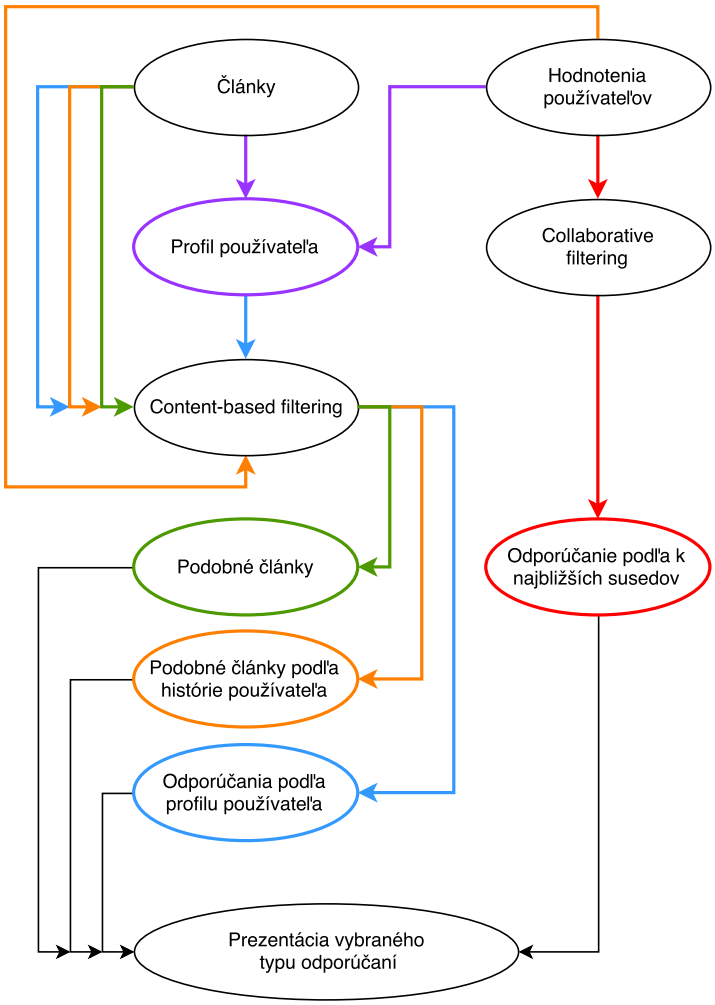
Výhodou metódy content-based filtering je fakt, že dokáže odporúčať aj nové články, ktoré ešte neboli zobrazené žiadnym používateľom, teda pokrýva studený štart položiek. Prvé dve spomenuté schémy je možné priradiť novým používateľom a tak pokryť aj studený štart používateľa.

4.3 Collaborative filtering

Z metód collaborative filteringu bolo vybrané odporúčanie na základe podobných používateľov. Algoritmus nájde k najbližších susedov používateľa u a z ich hodnotení zostaví zoznam 10 odporučených článkov pre u, ktoré čítali jemu podobní používatelia. Hodnotu parameteru k v návrhu nešpecifikujeme, pretože ho budeme testovať v samotnej implementácií a hľadať jeho vhodnú hodnotu. Algoritmus pre najbližších susedov bude pre porovnávanie používateľov používať Pearsonovu koreláciu (vzorec 5). Táto metóda bude použitá iba pre používateľov, ktorí majú vo svojej histórií 10 a viac prečítaných položiek.

4.4 Hybridné odporúčanie

Z podkapitoly 2.4 sme ako techniku hybridizácie zvolili *prepínanie*, ktorá bude vyberať podľa jednoduchých pravidiel vhodný typ odporúčaní pre daného používateľa. Ako bolo popísané v predošlých podkapitolách, je v tomto prípade potrebné rozlišovať medzi novým a stálym používateľom. Podmienky, na základe ktorých budeme vyberať aký typ odporúčaní a akým používateľom sa bude zobrazovať, popisuje až kapitola 6. Obrázok 1 ponúka štruktúrovaný náhľad na tvorbu odporúčaní v prostredí vyvíjaného hybridného systému.



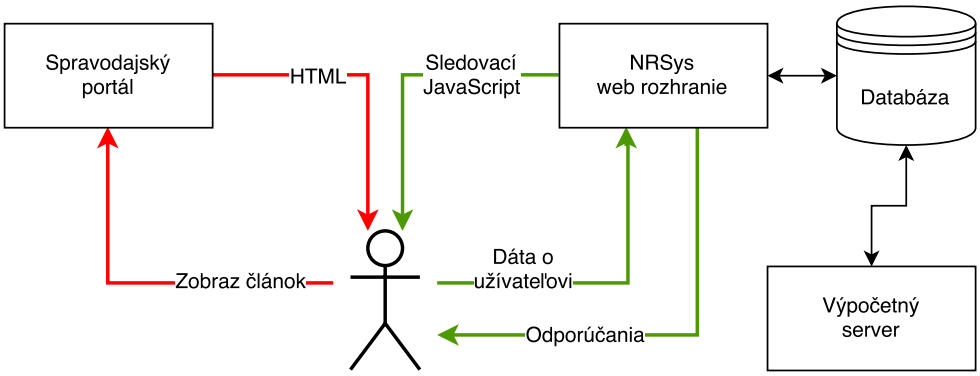
Obr. 1: Štruktúrovaný náhľad na návrh hybridného systému.

Vhodným spôsobom pre generovanie výsledných odporúčaní by mohla byť aj technika hybridizácie mixovaním, pretože systém obsahuje 4 rôzne druhy odporúčaní. Mixovanie by mohlo prebiehať na základe náhodného výberu odporúčaní z výsledkov jednotlivých komponentov, nájdením ich prieniku alebo určením poradia a počtu odporúčaní vyberaných z rôznych výsledkov.

5 Návrh a implementácia systému

V tejto kapitole je popísaný postup návrhu a implementácie prototypu systému pre generovanie odporúčaní novinových článkov pre koncových používateľov spravodajského portálu. Vyvíjaný systému dostal pracovné pomenovanie *NRSys*. Kapitola tiež oboznamuje s výberom a odôvodnením použitia jednotlivých technológií, nástrojov a programovacích jazykov.

Celý systém je navrhnutý ako samostatná služba bežiaca v prostredí internetu so vzdialeným prístupom. Tento spôsob by z pohľadu biznis modelu v budúcom vývoji a zdokonaľovaní poskytol priestor pre nasadenie na viacerých webových lokalitách súčasne. Obrázok 5 ukazuje zjednodušený náhľad na architektúru systému.



Obr. 2: Architektúra systému.

Náplňou systému je zaznamenávať a spracovávať údaje o používateľoch, teda čitateľoch článkov na internetovom spravodajskom portáli pomocou *sledovacieho kódu*. Okrem aktivity používateľov systém zhromažďuje aj obsah jednotlivých článkov z portálu, ktoré sú následne analyzované a prevedené do vektorového modelu. Z nazbieraných a spracovaných údajov bude systém schopný generovať štyri rôzne druhy odporúčaní navrhnutých v kapitole 4. Každému používateľovi bude technikou pripínania priradený najvhodnejší druh odporúčaní. Tieto odporúčania budú používateľovi zobrazované prostredníctvom sledovacieho kódu z dôvodu zabezpečenia jednoduchej implementácie pre prevádzkovateľa spravodajského portálu. Je dôležité podotknúť, že cieľom návrhu a implementácie je vytvorenie *funkčného prototypu* odporúčacieho systému. Prototyp poslúži ako nástroj pre otestovanie teoretických východísk tejto práce, umožní testovanie na vlastných dátach a v prípade dobrých výsledkov aj testovanie v reálnom prostredí za ostrej prevádzky. Keďže sa jedná o prototyp, návrh sa nezaoberá úrovňou zabezpečenia a hardwarovej infraštruktúry, nakoľko nie je sú dopredu známe nároky na výpočtovú a priestorovú náročnosť systému.

5.1 Funkčné požiadavky na systém

V tejto podkapitole sú stručne popísané hlavné požiadavky, ktoré musí systém spĺňať z funkčného hľadiska. NRSys bude plne automatizovaný systém pre zber a spracovanie dát, na základe ktorých bude generovať odporúčania pre používateľov. Systém nespracováva žiadne vstupy priamo zadávané používateľmi. Všetky potrebné dáta o používateľovi sú zbierané na pozadí jeho aktivity počas prehliadania webových stránok spravodajského portálu.

5.1.1 Zaznamenávanie implicitných hodnotení

Systém musí byť schopný zbierať, spracovávať a uchovávať dáta o aktivite používateľov na webových stránkach spravodajského portálu. Systém bude zaznamenávať dĺžku čítania jednotlivých článkov pre každého používateľa.

5.1.2 Spracovanie článkov

Systém musí byť schopný automaticky detekovať a zbierať obsah nových novinových článkov. Články bude schopný spracovávať a uchovávať ich vektorovú reprezentáciu pre zefektívnenie ďalšej práce s článkami.

5.1.3 Podobné články

Systém musí byť schopný pre každú položku i nájsť *top-N* položiek pojednávajúcich o podobnej tematike, ktoré boli pridané maximálne o 7 dní skôr alebo neskôr ako položka i. Bude tolerovaná určitá miera nepresnosti, nakoľko nie všetky témy sa môžu dostatočne vyskytovať vo vymedzenom časovom úseku.

5.1.4 Generovanie odporúčaní

Systém bude schopný použiť metódy content-based a collaborative filtering na generovanie odporúčaní pre používateľov spravodajského portálu. Technika hybridizácie prepínaním bude použitá pre zobrazovanie výsledných odporúčaní používateľovi.

5.1.5 Studený štart

Systém musí byť schopný odporúčať články pre nových používateľov a tiež odporúčať aj nové články.

5.1.6 Pravidelný prepočet odporúčaní

Výpočtový server bude v krátkych pravidelných intervaloch prepočítavať odporúčania pre používateľov. Z toho vyplýva, že výpočet odporúčaní bude prebiehať offline a nie v momente, keď majú byť zobrazované používateľovi. Výsledky výpočtov budú udržiavané v aktualizovanej databáze odporúčaní.

5.1.7 Rýchlosť odpovede

Systém musí byť schopný vydávať používateľom odporúčania na požiadanie rádovo v desiatkach milisekúnd, aby nebola narušená dĺžka načítania stránky v prehliadači. Maximálny časový strop generovania odpovede po prijatí požiadavky je určený na 500 milisekúnd na strane serveru.

5.2 Sledovací kód

Ako už bolo niekoľkokrát spomenuté, neoddeliteľnou súčasťou systému je zaznamenávanie aktivity používateľa z dôvodu určovania jeho záujmov. Ako popisuje kapitola 4, pri realizácií odporúčacieho systému použijeme ako implicitné hodnotenie čas, ktorý používateľ strávil čítaním článku. Sledovací kód bude zbierať nasledujúce dáta o aktivite používateľa:

- URL a ID aktuálne zobrazeného článku
- ID a zaznamenanú dĺžku čítania predošlého článku

Pre umožnenie sledovania prevádzkovateľ portálu vloží do HTML článkov krátky JavaScript kód (výpis 1). Princíp funkcie tohto kódu je veľmi jednoduchý. Používateľ zo svojho webového prehliadača odošle požiadaviek na zobrazenie článku. Po jeho načítaní a zobrazení sa aktivuje vložený JavaScript a ten následne stiahne zo serveru NRSys súbor obsahujúci samotný sledovací JavaScript kód, ktorý okamžite začne vykonávať svoju sledovaciu funkciu. Popísaný proces je zobrazený v sekvenčnom diagrame (obrázok 3).

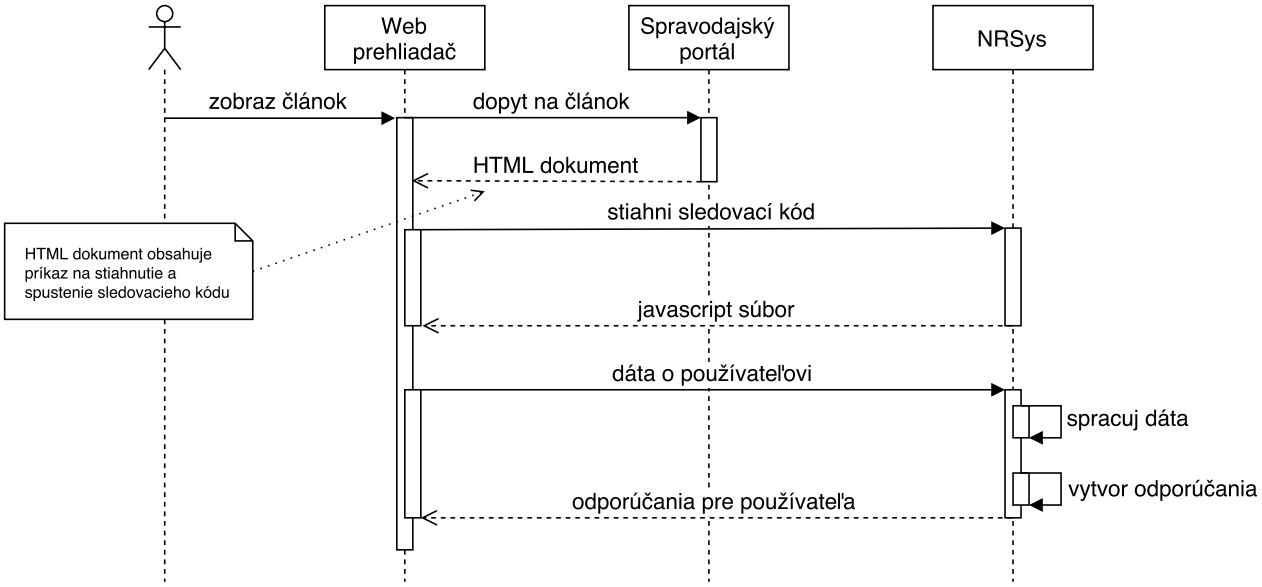
```
<script type="text/javascript">
    var nrsContent = 0; //int - id clanku
    (function(d, t) {
        var s = d.createElement(t);
        s.src = "//app.nrsys.tech/public/js/nrs.min.js";
        s.async = 1;
        d.body.appendChild(s);
    })(document, "script");
</script>
```

Výpis 1: JavaScript kód pre asynchrónne načítanie sledovacieho kódu.

Táto technika umožňuje udržiavanie sledovacieho kódu vždy aktualizovaného bez potreby zásahu zo strany prevádzkovateľa, keďže sa kód nachádza na serveri nášho systému. Po stiahnutí a spustení sa kód pokúsi identifikovať používateľa na základe existencie cookies súborov z predošlej návštevy. Ak súbory neexistujú, predpokladá sa, že ide o nového používateľa a kód odošle do systému požiadavku na vygenerovanie nového používateľského identifikátora. Spolu s touto požiadavkou sa odosiela aj URL adresa a číselný identifikátor zobrazeného článku. Odpoveďou je vygenerovaný identifikátor a zoznam odporúčaní pre nového používateľa. V prípade, že kód identifikoval vracajúceho sa používateľa, je na server odosielaný záznam o predošlej návšteve, ktorý bol uložený v súboroch cookies vygenerovaných pri poslednej návšteve. Záznam obsahuje

identifikátory používateľa, článku a dĺžku čítania predošlého článku a zároveň údaje o článku aktuálnom. Odpoveďou sú v tomto prípade takisto odporúčania pre používateľa. Po tomto procese zobrazenom v diagrame aktivít (obrázok 4) nasleduje aktualizácia cookies, do ktorých sú zapísané údaje o aktuálnej návšteve.

Meranie času sa aktivuje okamžite po spustení sledovacieho kódu a každých 100 milisekúnd je nameraná hodnota aktualizovaná a uložená v určenom cookies súbore. Meranie sa zastaví v prípade, že používateľ nevykonáva žiadnu aktivitu dlhšie ako 10 sekúnd (pohyb myši alebo scrolovanie), prepne sa na inú záložku prehliadača, prehliadač minimalizuje alebo zatvorí. Jediným nedostatkom tohto prístupu je, že systém nepozná dĺžku čítania aktuálneho článku a nevie s ňou rátať pri výpočte najnovších odporúčaní. Aby systém mohol s touto informáciou disponovať, sledovací kód by musel v krátkych pravidelných intervaloch posielať priebežne nameraný čas na server, čo nie je práve efektívne riešenie. Dĺžka čítania je teda získavaná spätne a do tej doby systém pracuje z predpokladom, že článok bol prečítaný celý.



Obr. 3: Priebeh udalostí pri zobrazení článku.

Sledovací kód odosiela správy cez protokol HTTP pomocou URL adresy, do ktorej sú pridávané vyššie spomenuté dáta vo forme GET parametrov (výpis 2).

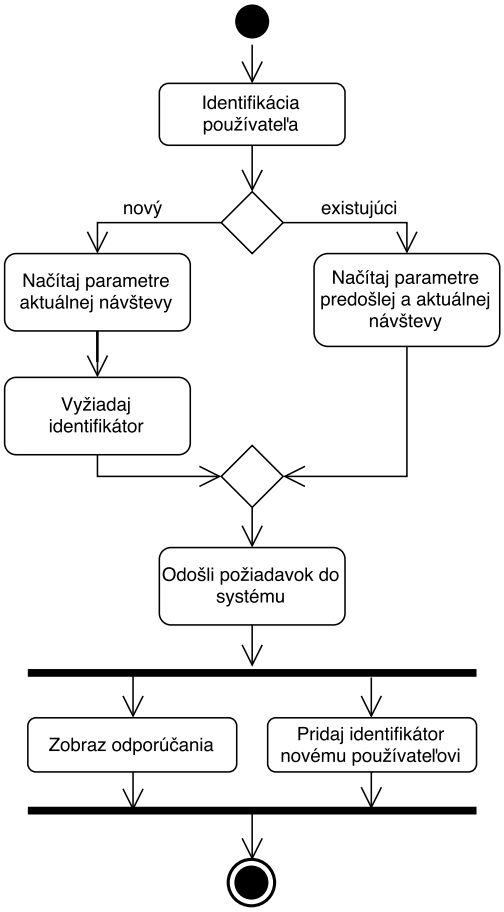
```
http://app.nrsys.tech/event?u=d251004f6bdfd513491d584147a64b35&c=94997&v=11.52&Ac=94997&Al=http
    ://www.info.sk/sprava/94997/chysta-sa-najvacsia-letecka-preprava-levov-v-dejinach/
```

Výpis 2: Odoslanie dát pomocou URL adresy.

V súčasnosti majú prehliadače implementovanú bezpečnostnú zásadu *same-origin policy*, ktorá neumožňuje JavaScript kódom odosielať HTTP požiadavky na iné URL adresy ako je adresa zdroju HTML dokumentu. To je však možné vyriešiť dvoma veľmi jednoduchými spôsobmi:

1. V prípade, že potrebujeme dáta na nejaký server len poslať a nezaujíma nás odpoveď, môžeme pomocou JavaScriptu vložiť do HTML kódu skrytý obrázok (tag *<img>*). URL adresa, kde sa má obrázok nachádzať bude obsahovať URL adresu cieľového serveru a dáta vo forme GET parametrov (výpis 2). Prehliadač sa v tom momente pokúsi adresu obrázku načítať a náš server prijme potrebné dáta.

2. Ak dáta potrebujeme zo serveru aj prijať, stačí podobným spôsobom do HTML vložiť vnorený rám (tag *<iframe>*), v ktorom je možné zobraziť inú stránku. Tomuto rámu nastavíme rovnakú URL adresu zdroju ako v predošlom prípade. Rozdiel nastáva v tom, že prehliadač do tohto rámu načíta z adresy obsah s dátami, ktoré môžeme používateľovi okamžite zobrazovať.



Obr. 4: Diagram aktivít vykonaných po spustení sledovacieho kódu.

Keďže JavaScript vo vnorenom ráme nemá priamy prístup pre prácu v rodičovskej záložke, je možné použiť jednoduchý mechanizmus pre bezpečnú komunikáciu medzi záložkami a oknami prehliadača. Výmena dát prebieha pomocou funkcie *window.postMessage*, ktorá rozošle správu všetkým záložkam a vyvolá udalosť *message*. V záložkách, kde je JavaScript, ktorý implementuje odchytenie tejto udalosti, môže dáta prijať a reagovať na základe ich obsahu. Náš sledovací

kód používa druhú spomenutú techniku, pretože bude vždy prijímať dáta v podobe odporúčaní alebo nového používateľského identifikátora. Pomocou spomenutej výmeny správ medzi záložkami bude používateľský identifikátor posielaný do rodičovskej záložky, kde ho príjme sledovací kód a uloží do cookies súboru.

5.3 Serverová aplikácia

Systém prijíma a vydáva dáta na základe HTTP požiadaviek z internetu. Požiadavky prichádzajú do systému z dvoch zdrojov a to zo sledovacích kódov a z výpočetového servera. Ďalšou z úloh aplikácie je zbieranie a spracovanie obsahu novinových článkov do vektorového modelu. Pre spomenuté potreby bol vybraný skriptovací jazyk PHP, ktorý dostatočne pokrýva nároky pre spomenuté úlohy. Serverová aplikácia bude bežať na webhostingu.

5.3.1 PHP

PHP (Hypertext Preprocessor) je open source skriptovací jazyk, ktorý sa používa najmä na programovanie klient-server aplikácií a pre vývoj dynamických webových stránok bežiacich na strane servera. PHP kód nie je kompilovaný, ale interpretovaný webovým serverom vo chvíli jeho volania a výstupom je napríklad HTML. PHP jazyk bol v tejto práci použitý aj z dôvodu jeho dobrej znalosti, jednoduchosti a predošlých skúsenostiach.

5.3.2 Slim framework

Pre zostavenie webovej aplikácie sme použili *Slim framework*[1]. Je to pomerne jednoduchý mikro framework napísaný v jazyku PHP. Slúži na vytváranie webových aplikácií a API rozhraní. Keďže je to mikro framework, jeho funkcionalita je značne okresaná a sústredená do prehľadného a spoľahlivého routeru. V doméne webových aplikácií je router časť frameworku alebo aplikácie zodpovedná za prijímanie HTTP požiadaviek a vydávanie správnych odpovedí na základe požiadavky. Router teda podľa nastavených pravidiel rozpozná, na čo sa daná URL adresa požiadavky dopytuje a vykoná proces určený pre dané pravidlo routeru. Okrem popísaného routovania požiadaviek Slim umožňuje jednoduchú manipuláciu s HTTP hlavičkami a telom odpovede. Slim je teda možné považovať za základnú kostru pre vytvorenie webovej aplikácie, ktorú je možné podľa potrieb rozširovať doplnením PHP knižníc pre prácu s databázou, šablónovacie či rôzne iné knižnice.

Kód vo výpise 3 spracuje požiadavku typu GET (napr. kliknutie na odkaz) s URL adresou http://www.example.com/event a ako odpoveď odošle HTML dokument. Podobným spôsobom sa dajú oddeľovať aj požiadavky typu POST, PUT, DELETE, HEAD, PATCH, OPTIONS a tak framework použiť ako REST aplikáciu.

[1] http://www.slimframework.com/

```php
$app = new \Slim\App();
$app->get("/event", function ($request, $response, $args) {
   /* ... prikazy pre dany router ...  */

   return $response->withHeader("Content-type", "text/html")
      ->write(/* ... telo HTML odpovede ... */);
});
$app->run();
```

Výpis 3: Routeru v Slim frameworku.

5.3.3 Dibi

Podstatou vyvíjaného systému je práca s dátami a ich uchovávanie v relačnej databáze MySQL. Pre zjednodušenie práce s databázou v jazyku PHP bola použitá knižnica *Dibi*[2] od českých autorov a vývojárov populárneho frameworku Nette. Knižnica Dibi vytvára vrstvu medzi kódom programátora a databázou. Obsahuje množstvo funkcií pre rutinnú prácu s databázou, zjednodušuje zápis SQL príkazov, podporuje konvencie a umožňuje písať prehľadný a efektívny kód.

```php
$options = array(
   "driver"   => DB_DRIVER,
   "host"     => DB_HOST,
   "username" => DB_USERNAME,
   "password" => DB_PASSWORD,
   "database" => DB_DATABASE
);

try {
   dibi::connect($options);
} catch (DibiException $e) {
   echo $e->getMessage();
}

$result = dibi::query("SELECT * FROM nrs_event WHERE id_user = %i", $id);

foreach ($result as $row) {
   /* spracovanie ziskanych riadkov */
}
```

Výpis 4: Pripojenie a dopyt na dáta z databázy pomocou knižnice Dibi.

Jednou z predností Dibi je statický register, ktorý udržuje objekt pripojenia k databáze v globálnom úložisku. Preto po vytvorení pripojenia *(dibi::connect($options))* sa programátor nemusí

[2] https://dibiphp.com/cs/quick-start

starať o predávanie inštancie pripojenia k databáze naprieč rôznymi triedami, ale statický objekt je vždy k dispozícií použitím volania *dibi::* (výpis 4).

5.3.4 Autentizácia

Serverová aplikácia umožňuje aj vzdialený prístup k dátam, ku ktorým by sa nemal bežný používateľ dostať. Ide o dáta slúžiace pre výpočet odporúčaní a teda by mali byť prístupné len pre výpočtový server. Ako bolo spomenuté, celý systém je vyvíjaný ako prototyp a na zabezpečenie týchto prístupov bola použitá metóda *HTTP Basic Auth*. Je to zabezpečenie na nižšej úrovni, nakoľko každá požiadavka od výpočtového serveru musí obsahovať prihlasovacie meno a heslo. Zvýšenie bezpečia pri komunikácií by mohla poskytnúť implementácia protokolu SSL alebo iný spôsob autentizácie ako napr. OAuth 2.0. Pre potreby prototypu je však tento variant postačujúci, keďže vo fáze testovania nepredpokladáme na systém útok typu man-in-the-middle pri komunikácií server-server.

5.4 Dátový model

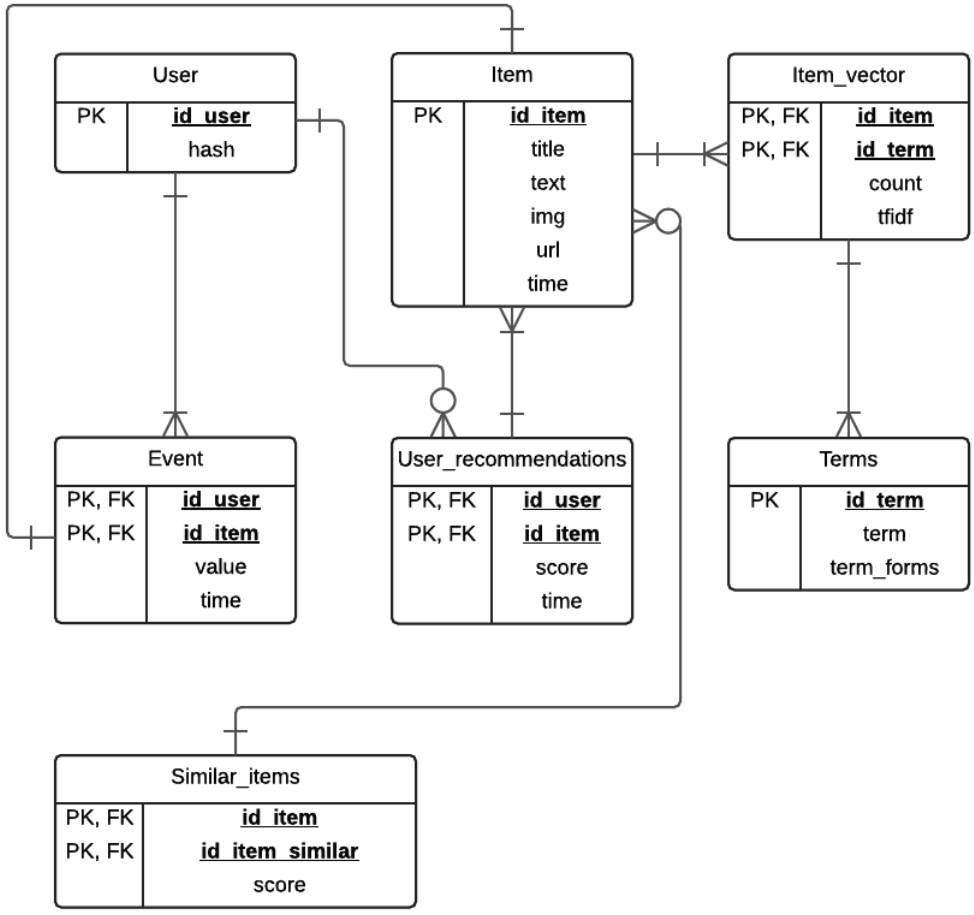
Dátový model systému bol navrhnutý s ohľadom na offline spracovanie dát. Výpočet odporúčaní, ani žiadne ďalšie spracovanie dát neprebieha priamo nad databázou ale v oddelenom systéme a databáza je následne len aktualizovaná. Odporúčania sú teda pre používateľov pravidelne vypočítavané na oddelenom serveri. Tento prístup síce zníži aktuálnosť používateľových odporúčaní, no systém bude schopný vydávať odporúčania okamžite, čo je základná požiadavka pre nasadenie do reálnej prevádzky.

5.4.1 Modelované entity

Dátový model vychádza len z dvoch základných entít, ktorými sú používateľ a položka. Ostatné tabuľky databázez týchto entít priamo vychádzajú.

Používateľ je konkrétny návštevník webovej stránky, ktorá v sebe implementuje funkcie systému NRSys. Používateľ je jednoznačne *identifikovaný* pomocou alfanumerického identifikačného reťazca, ktorý je vygenerovaný pri jeho prvej návšteve webovej stránky. O používateľovi nie sú známe žiadne ďalšie údaje, ktoré by bolo možné použiť pre identifikáciu konkrétnej osoby. Systém zaznamenáva ktoré články, kedy a ako dlho používateľ čítal. Tieto záznamy sú pomenované ako *udalosti* a poskytujú dáta, z ktorých sa následne určujú záujmy používateľa.

Položka je v našom systéme prezentovaná novinovým článkom. Každá položka obsahuje titulku, text článku, URL adresy obrázku aj článku a čas pridania do systému. Analýza obsahu článku je základom pre content-based metódu. Aby bolo možné medzi článkami hľadať podobnosti, je potrebné ich previesť do vektorového modelu, ktorý je v databáze uložený v podobe invertovaného zoznamu.



Obr. 5: ER diagram pre databázu.

5.4.2 Databázový systém

Pre ukladanie a správu dát bol vybraný relačný databázový systém *MySQL*. Tento systém bol zvolený prevažne na základe predošlých skúseností so systémom a SQL jazykom. Do budúcna by bolo vhodné urobiť prieskum a experimenty s rôznymi druhmi databázových systémov, či už z dôvodu rýchlosti spracovania dotazov alebo škálovateľnosti. Ako zaujímavou oblasťou pre odporúčacie systémy sa ukazujú grafové databázy, pomocou ktorých je možné hľadať vzťahy medzi entitami v reálnom čase.

5.4.3 Implementácia databázy

Po vytvorení štruktúry databázy a naplnení testovacími dátami bol zistený menší nedostatok v návrhu. Problematickým sa ukázal nedostatočný rozbor použitých dátových typov pre jednotlivé stĺpce tabuliek. Pre niektoré stĺpce v tabuľkách boli použité údajové typy so zbytočne veľkým rozsahom, ktoré následne zaberali viac miesta na disku. V prípade reálneho nasadenia

by tento problém spôsobil zbytočné vynakladanie financií na úložný priestor. Napríklad použitie dátového typu SMALLINT (2 Byte) z pôvodného INT (4 Byte) pre udržiavanie početnosti termu v dokumente znížilo veľkosť tabuľky o 9,5% pôvodnej veľkosti. Neoptimálnym riešením sa ukázalo tiež ukladanie 32 znakového identifikátora používateľa ku každému záznamu o jeho aktivite. Tento problém bol vyriešený pridaním tabuľky používateľov, kde každému identifikátoru typu VARCHAR(32) (33 Byte) bol priradený nový číselný identifikátor typu INT. Ušetrených tak bolo 65,4% pôvodnej veľkosti tabuľky s tým, že bol upravený aj primárny index tabuľky. V ďalšom prípade sa pomocou úpravy indexov podarilo znížiť veľkosť tabuľky o 28,7%. Celková úspora po úprave dátových typov a indexov predstavovala zhruba 65,5% pôvodnej veľkosti všetkých upravených tabuliek. SQL príkazy pre vytvorenie štruktúry databázy sú obsiahnuté v prílohe.

5.5 Spracovanie obsahu článkov

Content-based systémy sú založené na práci a porovnávaní položiek na základe atribútov, ktoré ich popisujú. Povaha a význam novinových článkov sa zvyčajne neudáva v podobe atribútov (pokiaľ neberieme do úvahy kategóriu, autora a podobné atribúty) ale vyjadruje ich samotný textový obsah článkov. Z pohľadu strojového spracovania sú články neštruktúrované dáta a ako bolo spomenuté v kapitole 3, takéto dáta je vhodné vyjadriť pomocou vektorového modelu. Pre získanie vektorovej reprezentácie postupne prejde článok 6 bodmi procesu spracovania:

1. Odstránenie HTML tagov, diakritiky, interpunkčných znamienok a čísel
2. Rozdelenie textu podľa medzier na jednotlivé slová, tzv. tokeny
3. Odstránenie stop slov
4. Slovníková lematizácia a zistenie početnosti jednotlivých termov
5. Výpočet tf-idf váh termov
6. Uloženie vektoru do databázy

Popísané spracovanie textu je implementované v jazyku PHP v triede pod názvom *LexicalAnalyzer*. Načítanie obsahu a spracovanie nových článkov prebieha v PHP skripte, ktorý je spúšťaný v pravidelných intervaloch pomocou spúšťaču pravidelných úloh Cron. Tento skript najskôr zistí, či v databáze pribudli URL adresy nových článkov, ktoré neboli doposiaľ zaindexované. Ak sú nájdené nové články, skript stiahne ich HTML obsah a vykoná vyššie popísaný proces spracovania.

Kritickou časťou v procese spracovania textu je lematizácia a získanie termov z článkov. Pre túto úlohu sme sa na základe kapitoly 3 rozhodli pre lematizáciu použiť slovník termov s ich morfologickými tvarmi. Voľne prístupné a kvalitné slovníky pre lematizáciu slovenského jazyka nie sú dostupné a preto sme museli vytvoriť vlastný slovník. Prevádzkovateľ portálu poskytol vlastný slovník slovenských slov s ich morfologickými tvarmi. Tento slovník neobsahoval označenie slovného druhu, iba základ slova a morfologické tvary, ktoré v mnohých prípadoch neboli úplné. Tento slovník obsahoval 41 800 termov a preto sme sa rozhodli pre tvorbu vektorového modelu

použiť iba podstatné mená. Pre ich selekciu bol použitý dostupný slovník Wordnet[3], ktorý okrem iného obsahuje aj najviac frekventované slovenské podstatné mená. Na základe Wordnetu bolo z predošlého slovníku vyfiltrovaných 5044 podstatných mien. Presnosť vzniknutého slovníku je spochybniteľná, no pre účely tejto práce postačujúca. Výsledný slovník je udržiavaný v databáze a taktiež je súčasťou prílohy diplomovej práce.

5.6 Výpočet odporúčaní

V kapitole 4 boli navrhnuté štyri spôsoby generovania odporúčaní novinových článkov. V tejto kapitole bude popísaná ich implementácia, spôsob použitia a prostredie v ktorom bude prebiehať výpočet odporúčaní.

5.6.1 Výpočtový server

Výpočet odporúčaní je časovo náročný a drahý proces z pohľadu výpočtového výkonu. Náročnosť stúpa úmerne s narastajúcim počtom položiek, používateľov a ich hodnotení. Nasadenie udržateľného a spoľahlivého odporúčacieho systému si vyžaduje premyslenie infraštruktúry a platformy, na ktorej budú výpočty realizované. V našom prípade sme pre výpočty použili knižnicu *Apache Mahout*, ktorá obsahuje množstvo algoritmov pre zhlukovanie, redukciu dimenzionality a tiež algoritmy pre collaborative filtering. Táto knižnica je navrhnutá pre distribuované výpočty, no neobsahuje samotnú platformu pre distribuované prostredie. Dá sa však použiť aj bez nej. Takouto platoformou je napríklad škálovateľný framework pre distribuované výpočty a úložisko v počítačových clusteroch *Apache Hadoop*. Pre simuláciu výpočtového serveru sme použili počítač so špecifikáciou:

- 4 jadrový procesor AMD Phenom II X4 965, 3400MHz
- operačná pamäť 8GB
- operačný systém Ubuntu 14.04.4 LTS 64-bit

Na tomto výpočtovom serveri je nastavený Cron, ktorý automaticky spúšťa výpočet odporúčaní. Výstupom sú textové súbory s odporúčaniami, ktoré sú cez protokol FTP nahrané do serverovej aplikácie, ktorá na základe ich obsahu aktualizuje databázu. Nejedná sa o optimálnu architektúru, pretože výpočtový server nemá priamy prístup do databázy bežiacej na hostingovej službe spolu so serverovou aplikáciou. Pre účely testovania a ladenia algoritmov tento prístup ponúka dostatočný náhľad do problematiky. Pre budúcu reálnu prevádzku by bolo vhodné systém zjednotiť, algoritmy prerobiť na distribuovateľné a celý systém nasadiť napríklad na cloudovú platformu spoločnosti Google.

[3] http://korpus.juls.savba.sk/WordNet.html

5.6.2 Apache Mahout

Apache Mahout je open-source knižnica napísaná v jazyku Java obsahujúca implementáciu konkrétnych distribuovateľných a škálovateľných algoritmov pre oblasť strojového učenia (machine learning). Je vhodnou nadstavbou pre Apache Hadoop a teda aj pre vytvorenie škálovateľného odporúčacieho systému. V našom prípade táto knižnica poslúži pre odporúčanie metódou najbližších susedov. Okrem tejto funkcie využijeme aj jej implementácie rôznych tried pre prácu s vektormi a výpočet podobností vektorov. Platforma Apache Hadoop v našom riešení zatiaľ použitá nebude, nakoľko nami navrhnuté algoritmy pre prototyp systému nedosahujú takú výpočtovú náročnosť a nepracujú s tak veľkým objemom dát aby bolo potrebné ich výpočty distribuovať medzi viac počítačov.

5.6.3 Nájdenie podobných článkov

Princíp nájdenia podobných článkov je veľmi priamočiary. Algoritmus z databázy vyberie vektory všetkých článkov s tf-idf váhami pridaných za posledných 14 dní. Pre každý z týchto článkov, ktorý nie je starší viac ako 7 dní algoritmus podľa kosínusovej podobnosti (vzorec 2) nájde zoznam 10 najviac podobných článkov v rozmedzí $\pm$ 7 dní od dátumu pridania pozorovaného článku. Prepočet podobností prebieha na výpočtovom serveri, ktorý výsledky uloží do databázy webovej aplikácie. Tento algoritmus je napísaný v jazyku Java. Z knižnice Mahout používa triedy pre prácu s vektormi a tiež implementáciu kosínusovej podobnosti. Kód tejto funkcie je zbalený v JAR balíku a spustiť sa dá príkazom (výpis 5):

```
java -jar nrsys_recommender_package.jar item_item
```

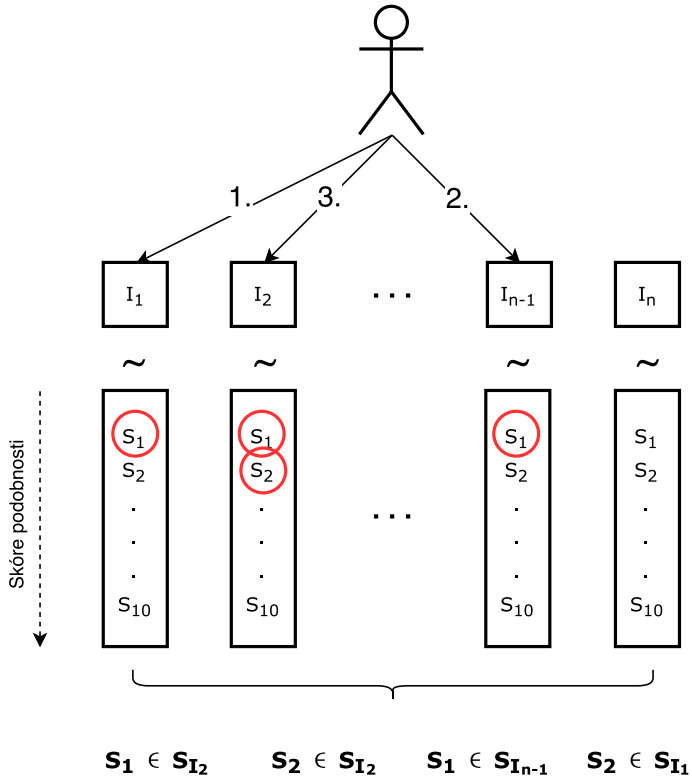
Výpis 5: Spustenie programu pre nájdenie podobných článkov.

Pri frekvencií 130 - 160 pridaných článkov denne bolo na vstupe do programu priemerne 2 300 vektorov článkov. Načítanie dát, výpočet podobností a aktualizácia databázy trvala priemerne 30 sekúnd.

5.6.4 Odporúčanie článkov podľa histórie používateľa

Pri tomto type odporúčaní nie je potrebný žiaden samostatný výpočet. Odporúčania sú nájdené vďaka dopredu vypočítanej databáze podobných článkov. Predpokladajme, že používateľovi chceme zobrazovať m odporúčaní. Princíp spočíva v tom, že algoritmus vyberie z histórie používateľa n článkov, kde $1 \leq n \leq m$, zoradených zostupne podľa času, kedy ich používateľ videl. V prípade, že $m > n$, teda požadujeme viac odporúčaní ako používateľ videl článkov, je pre každý článok z histórie vybraný jeden najviac podobný článok a pre posledný článok je vybraných $m - n$ článkov, aby bola pokrytá požiadavka na m odporúčaní (obrázok 6). Ak je $m \leq n$, každému z n článkov histórie je priradený práve jeden najviac podobný článok. Nájdené články tvoria výsledné odporúčania pre používateľa. S kritériami pre výber podobných článkov

je možné akokoľvek manipulovať a hľadať optimálnu konfiguráciu.

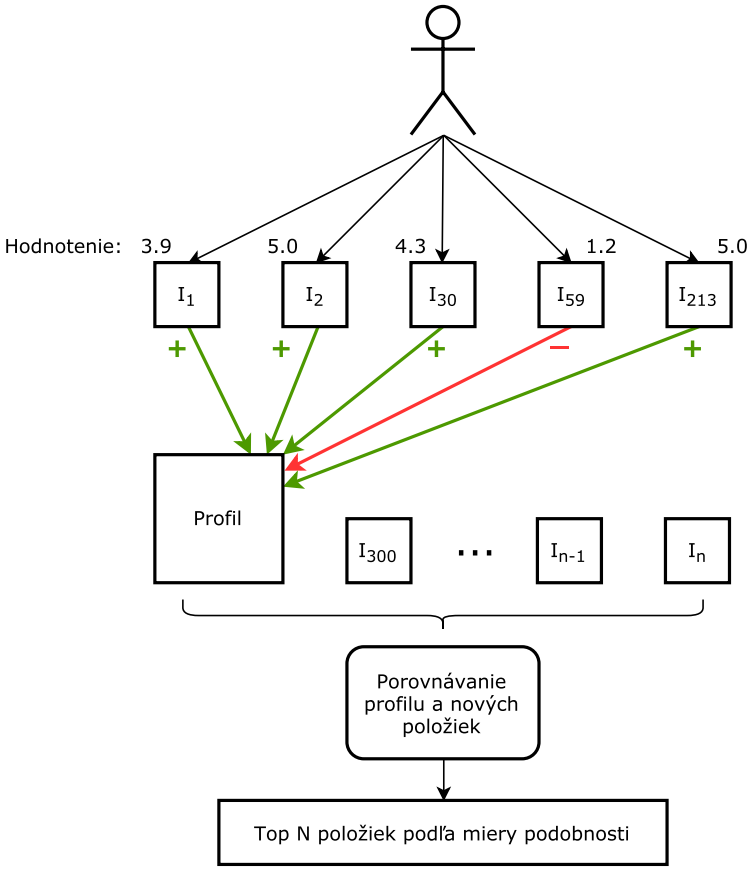


Obr. 6: Nájdenie odporúčaní podľa histórie používateľa pre $m = 4$ a $n = 3$.

Tento prístup môže používateľa rýchlo uzavrieť v jednej téme, čo niektorí používatelia môžu vnímať negatívne, no pre iných to môže byť naopak výhodou. Tento fakt je vyvážený tým, že používateľ dostáva vždy aktuálne výsledky podľa jeho poslednej aktivity.

5.6.5 Odporúčanie článkov podľa profilu používateľa

Tento typ odporúčaní je určený pre pravidelných používateľov s 10 a viac videnými článkami. Odporúčanie podľa profilu používateľa sa nemusí obmedzovať na minimálny počet videných článkov, no z dôvodu obmedzeného úložného priestoru pre databázu na webhostingu nemôžeme udržiavať odporúčania pre každého používateľa. Ak by systém mal generovať odporúčania pre všetkých používateľov, znamenalo by to milióny riadkov záznamov a tiež zvýšenie výpočtovej náročnosti. Pre vytváranie profilu používateľa sme použili Rocchio algoritmus (vzorec 3), ktorý zachytáva vývoj profilu používateľa v čase. Výsledkom algoritmu je vektor s rovnakou reprezentáciou ako sú vektory článkov. Nájdenie odporúčaní prebieha porovnávaním profilu používateľa a vektorov článkov pridanými za posledných 7 dní kosínusovou podobnosťou. Výhodou tohto prístupu je penalizácia položiek, ktoré sa používateľovi nepáčili. Celý proces tvorby profilu a nájdenia odporúčaní je zjednodušene zobrazený na obrázku 7.



Obr. 7: Tvorba profilu používateľa a hľadanie odporúčaní na jeho základe.

Popísaný algoritmus je implementovaný v jazyku Java s použitím tried pre prácu s vektormi z knižnice Mahout. Vstupom pre program je matica hodnotení používateľov za posledné 2 týždne, vektorový model článkov a zoznam článkov pridaných za posledných 7 dní.

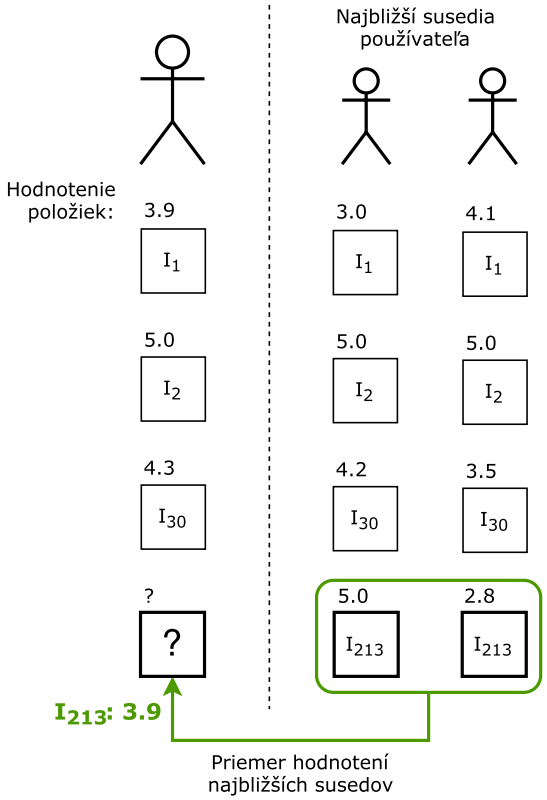
```
java -jar nrsys_recommender_package.jar user_item
```

Výpis 6: Spustenie programu pre výpočet odporúčaní podľa profilu.

Počas testovania trval beh tohto algoritmu priemerne 4,5 minúty v rámci stiahnutia dát, výpočtu a aktualizácie databázy. Na vstupe bolo priemerne zhruba 30 000 používateľov.

5.6.6 Odporúčanie článkov podľa podobných používateľov

Pre metódu collaborative filtering bol vybraný spôsob nájdenia odporúčaní pomocou najbližších susedov, ktorý je v knižnici Mahout priamo implementovaný pre jednoduché použitie. Tak ako v predošlom prípade, aj táto metóda je aplikovaná len pre pravidelných používateľov. Výpočet podobnosti používateľov zabezpečuje Pearsonova korelácia (vzorec 5). Výsledkom programu je nájdenie 10 odporúčaní pre všetkých pravidelných používateľov. Pre predstavu je na obrázku 8 veľmi zjednodušene ilustrovaný výpočet odporúčaní z hodnotení podobných používateľov.



Obr. 8: Nájdenie odporúčaní podľa najbližších susedov používateľa.

Na vstupe do algoritmu sú len hodnotenia používateľov. V dobe testovania bol ich priemerný počet zhruba 30 000. Priemerná doba stiahnutia dát, výpočtu odporúčaní a aktualizácie databázy dosahovala 5 minút. Pri tejto metóde sa môže stať, že používateľovi nie sú nájdení žiadni najbližší susedia a teda ani žiadne odporúčania.

```
java -jar nrsys_recommender_package.jar user_user
```

Výpis 7: Spustenie programu pre výpočet odporúčaní podľa podobných používateľov.

5.7 Súkromie používateľov

Súkromie používateľov je vždy citlivá téma. V našom prípade systém nezbiera osobné či citlivé dáta automaticky, ani nie sú do systému vkladané prevádzkovateľom spravodajského portálu. Systém v žiadnom prípade pre svoje fungovanie nevyžaduje konkrétnu identifikáciu používateľa, jeho osobné alebo iné demografické údaje. Rozpoznávanie používateľa na strane systému prebieha len na základe náhodne vygenerovaného identifikačného reťazca priradeného používateľovi pri jeho prvej návšteve spravodajského portálu. Tento reťazec a ďalšie zaznamenávané informácie sú na strane používateľa uchovávané v cookies súboroch a tým pádom má používateľ možnosť tieto súbory kedykoľvek odstrániť či úplne zakázať.

6 Testovanie systému v reálnom prostredí

Pre potreby nasadenia a testovania systému bola vytvorená dohoda s prevádzkovateľom spravodajského portálu *www.info.sk*. Ide o webové stránky, na ktorých používatelia nájdu aktuálne spravodajstvo v rôznych kategóriách. Mesačná návštevnosť týchto stránok sa pohybuje okolo 400 000 unikátnych používateľov.

6.1 Pôvodný stav

Pred nasadením systému NRSys portál využíval JavaScript modul pre zobrazovanie odporúčaní *Matched content*[4] od spoločnosti Google. Presný princíp tvorby odporúčaní zobrazovaných v tomto module nie je zverejnený, a ani subjektívnym pozorovaním sme ho nevedeli zaradiť k žiadnej z metód. Zásadným nedostatkom tohto modulu je odporúčanie neaktuálnych a starých článkov, čo dokazuje aj ukážka (obrázok 9) s odporúčaniami pre úvodnú stránku portálu zo dňa 24.4.2016.

Obr. 9: Modul Matched content od spoločnosti Google.

[4]https://support.google.com/adsense/answer/6111336?hl=en&ref_topic=1307438

6.2 Nasadenie v ostrej prevádzke

NRSys bol do prevádzky v reálnom prostredí nasadený v dvoch fázach. V prvej fáze bol do HTML kódu portálu vložený JavaScript kód (výpis 1) pre sledovanie aktivity používateľov a odosielanie zaznamenaných dát do systému NRSys. Výsledky nazbieraných dát sú zhrnuté v podkapitole 6.3. V tejto fáze bol kód a celkový systém niekoľkokrát upravený, čoho dôsledkom boli aj výpadky v zaznamenávaní dát.

Druhá fáza nasadenia spočívala v povolení od prevádzkovateľa použiť naše rozhranie pre zobrazovanie odporúčaní používateľom. S otvoreným prístupom prevádzkovateľa k novým technológiám nebol takmer žiadny problém. Po osobnej konzultácií, prezentácií funkcionality a názornej ukážke systému nám prevádzkovateľ portálu dal povolenie na testovanie v ostrej prevádzke. Odporúčania sme začali zobrazovať na mieste predošlého odporúčacieho modulu od spoločnosti Google. Pri testovaní jednotlivých odporúčacích metód nám prevádzkovateľ portálu nechal voľné ruky a nijak výrazne nás neobmedzoval.

6.3 Zaznamenané dáta

V tabuľke 3 sú stručne uvedené súhrnné informácie o nazbieraných dátach, ktoré boli sledovacím kódom zhromaždené v období od 4.2.2016 do 4.4.2016 na webových stránkach portálu. Je potrebné podotknúť, že systém NRSys počas tohto obdobia zaznamenal niekoľko dlhších výpadkov z dôvodu chýb pri aktualizáciách a úpravách systému, a tiež počas migrácie na nový webhostingový server. Počet zobrazení v tabuľke 3 nie je možné porovnávať s výsledkom z Google Analytics, nakoľko v našom prípade sledovací kód nie je umiestnený na hlavnej stránke a podstránkach kategórií, ale iba na podstránkach článkov. Štatistika nových používateľov by sa mohla priblížiť tej z Google Analytics počas budúceho obdobia, kedy bude systém bežať v stabilnej produkčnej verzií.

	NRSys	Google Analytics
Unikátni používatelia	725 878	784 063
Počet zobrazení	1 874 871	2 580 530

Tabuľka 3: Porovnanie dát návštevnosti.

Z údajov tabuľky 4 je vidieť, akým výrazným problémom je studený štart používateľov. Preto sme ako nových používateľov označili tých, ktorí videli menej ako 10 článkov. Podiel pravidelných používateľov bol len 4.24% z celkového počtu unikátnych používateľov. Ako už bolo spomenuté, výpočet odporúčaní podľa profilu a najbližších susedov prebiehalo len pre pravidelných používateľov. Odporúčanie pre všetkých používateľov by bolo mimo rozsah priestorového aj výpočtového výkonu aktuálnej infraštruktúry prototypu.

	počet
Články	79 300
Zobrazené články	15 960
Používatelia s 1 až 4 zobrazeniami	657 330
Používatelia s 5 až 9 zobrazeniami	37 726
Používatelia s 10 a viac zobrazeniami	30 822

Tabuľka 4: Súhrn zaznamenaných dát.

6.4 Testovanie

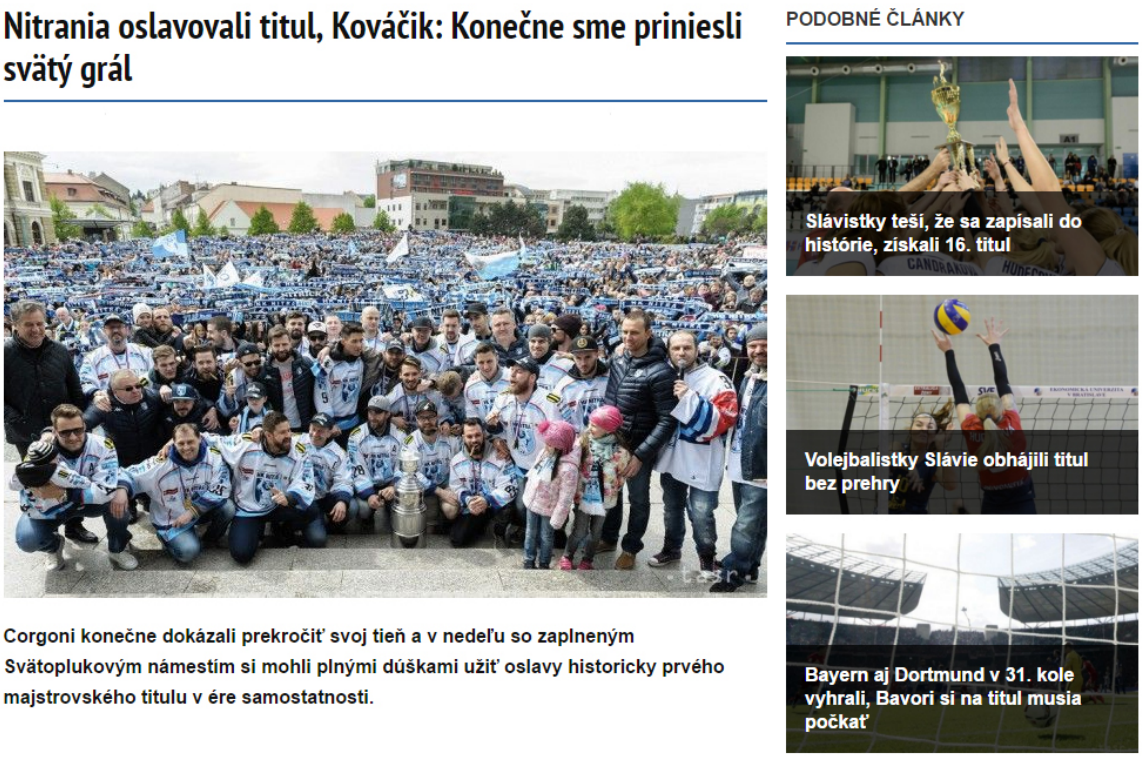
Hybridizácia systému bola dosiahnutá formou prepínania medzi jednotlivými druhmi odporúčaní na základe jednoduchých kritérií odvíjajúcich sa od množstva článkov, ktoré používateľ prečítal. Ako testovaciu metódu sme zvolili A/B test, respektíve v našom prípade A/B/C/D test. Testy A a B sú vždy náhodne priraďované novým používateľom z dôvodu studeného štartu. Tieto testy boli súčasne priradené aj existujúcim používateľom s históriou pod 10 článkov v pomere 1:1 a v rovnakom pomere aj polovici pravidelných používateľov. Zvyšné testy C a D boli priradené druhej polovici pravidelných používateľov takisto v pomere 1:1. Teoretický predpoklad tohto testu by mal priniesť štatistiku, akým spôsobom sú využívané jednotlivé typy odporúčaní.

Výsledky testu nie sú v tejto práci prezentované, pretože z hľadiska časovej náročnosti testu nebolo nazbieraných dostatok dát pre vyvodenie konkrétnych záverov. S určitosťou však môžeme povedať, že najmenej používanou bola metóda collaborative filtering. Empirickým pozorovaním z doteraz nazbieraných dát sme dospeli k názoru, že používateľov viac zaujímajú tematicky alebo obsahovo podobné články. Toto správanie používateľov však podstata metódy collaborative filtering narušuje. Takéto tvrdenie by sa teoreticky dalo otestovať mixovaním a zobrazovaním odporúčaní z oboch metód súčasne.

Konfigurácie všetkých typov testov spomenutých v nasledujúcich podkapitolách budú v pravidelných mesačných intervaloch upravované, aby sme zistili dopad zmien parametrov jednotlivých algoritmov. Keď sa nový používateľ stane pravidelným, náhodným spôsobom systém rozhodne, či bude takémuto používateľovi priradený iný, a ak áno, aký typ odporúčaní. Po určitej dobe je možné testy medzi používateľmi aj rotovať a pozorovať zmenu v ich správaní. Výsledky dlhodobého testovania s postupným ladením konfigurácií odporúčacích komponentov by mali ukázať, ktoré metódy má zmysel ďalej rozvíjať a akým spôsobom vplývajú zmeny ich parametrov na správanie používateľov.

6.4.1 Test A

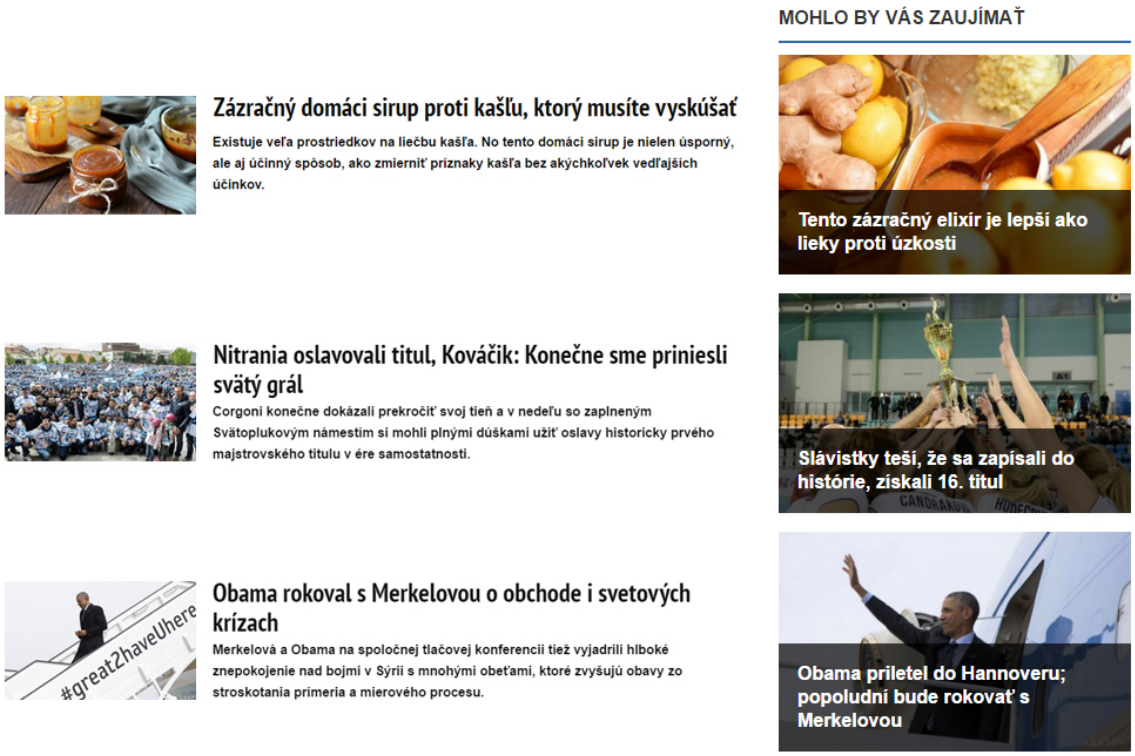
Test A je primárne určený pre nových používateľov, o ktorých systém nemá žiadne údaje. Odporúčania pre tento test sú generované ako 3 najviac podobné články pre aktuálne zobrazený článok (obrázok 13). Používateľ má tak možnosť pomocou odporúčaní zostať v podobnej alebo rovnakej tematike. Tieto odporúčania sú používateľom zobrazované pod titulkom *Podobné články*. Pre porovnanie výsledkov sme tento test priradili aj 25% pravidelných používateľov.



Obr. 10: Odporúčanie podobných článkov.

6.4.2 Test B

Test B je taktiež variantou pre nových používateľov, ktorá pri generovaní odporúčaní používa 3 naposledy zobrazené články zoradené podľa času zostupne (obrázok 11 vľavo). Pre tento druh odporúčaní bol zvolený titulok *Mohlo by Vás zaujímať*. Aj tento test bol súčasne priradený 25% pravidelných používateľov.



Obr. 11: Odporúčanie podobných článkov podľa histórie používateľa.

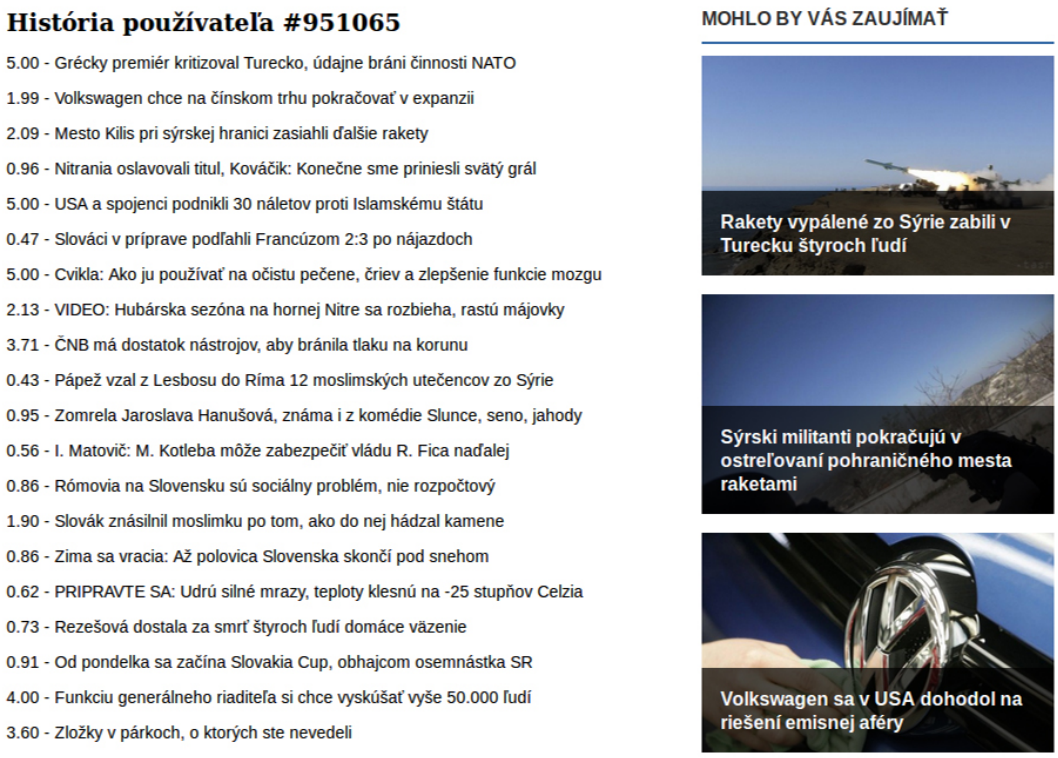
6.4.3 Test C

Test C zobrazuje pravidelným používateľom 3 odporúčania podľa metódy najbližších susedov s parametrom $k = 40$. Test bol priradený 25% pravidelných používateľov pod titulkom *Mohlo by Vás zaujímať*. Tento typ odporúčaní používatelia už od začiatku využívali najmenej, čo vytvára potrebu zistiť príčinu slabého záujmu. Ako problémovým sa môže ukazovať parameter $k = 40$, vstupné dáta výpočtu alebo samotná implementácia algoritmu pre collaborative filtering.

Obr. 12: Odporúčanie článkov metódou najbližších susedov.

6.4.4 Test D

Test D by mal byť odpoveďou na záujmy používateľa, keďže generuje odporúčania na základe jeho profilu vytvoreného z histórie prečítaných článkov. Pri výpočte profilu berie náš algoritmus v úvahu len tzv. krátkodobé záujmy používateľa, čo v našom prípade znamená, že z histórie používateľa vyberáme iba 15 naposledy zobrazených článkov. Vybrané sú 3 odporúčania pod titulkom *Mohlo by Vás zaujímať*. Test bol taktiež priradený 25% pravidelných používateľov.



Obr. 13: Odporúčania na základe profilu používateľa.

7 Záver

Táto diplomová práca popísala základné metódy pre tvorbu odporúčaní, metodiku spracovania neštruktúrovaných textových dokumentov, zaznamenávanie implicitných hodnotení používateľov a odporúčacie algoritmy. Na základe získaných znalostí bol navrhnutý a implementovaný prototyp hybridného odporúčacieho systému s použitím metód content-based a collaborative filtering. Systém je schopný vytvárať návrhy na potenciálne zaujímavé novinové články pre používateľa a podporiť ho v procese rozhodovania, ktoré články by ho mohli zaujať. Spomenuté ciele by mali teoreticky viesť k udržaniu pozornosti a zvýšeniu aktivity používateľa na stránkach spravodajského portálu. Zvýšenie počtu preklikov na stránkach by mohlo priamo súvisieť aj so zvýšením zisku generovaného zo zobrazovaných reklám.

Výhodou tejto práce bol vývoj postavený na vlastných reálnych dátach. Vyvinutý prototyp systému bol nasadený a prakticky otestovaný na portále www.info.sk, kde sa jeho odporúčania denne zobrazovali zhruba 25 000 používateľom (štatistika z Google Analytics).

Ďalšia práca a vývoj by mohli smerovať k zdokonaleniu infraštruktúry systému, čím je myslené zjednotenie webovej aplikácie a výpočtového serveru na jedno fyzické miesto, alebo tieto dva prvky dostať do čo najtesnejšej vzdialenosti z pohľadu prenosu dát. Bolo by potrebné zdokonaliť softwarovú architektúru systému a zabezpečiť a zefektívniť samotné výpočty pre prostredie distribuovaného systému. Najdôležitejšou časťou budúcej práce je vyhodnotenie dlhodobých výsledkov testovania na reálnych používateľoch a ich použitie pri dolaďovaní súčasných, alebo návrhu nových algoritmov pre výpočet odporúčaní. Systém bol navrhnutý ako samostatná webová služba, ktorá môže byť po určitých úpravách súčasne nasadená na viacerých webových portáloch s textovým obsahom, čo podľa nás prináša možnosti komerčného využitia.

Literatúra

[1] Brozovský, Lukáš. *Recommender system for a dating service.* Diss. Master's thesis, KSI, MFF UK, Prague, Czech Republic, 2006.

[2] Ghazanfar, Mustansar Ali, and Adam Prugel-Bennett. A scalable, accurate hybrid recommender system. *Knowledge Discovery and Data Mining, 2010. WKDD'10. Third International Conference on.* IEEE, 2010.

[3] Personalized hybrid recommendation for group of users: Top-N multimedia recommender Ondrej Kaššák, Michal Kompan, Mária Bieliková

[4] Pazzani, Michael J. A framework for collaborative, content-based and demographic filtering. *Artificial Intelligence Review* 13.5-6 (1999): 393-408.

[5] Balabanović, Marko, and Yoav Shoham. Fab: content-based, collaborative recommendation. *Communications of the ACM* 40.3 (1997): 66-72.

[6] Shani, Guy, et al. Establishing user profiles in the mediascout recommender system. *Computational Intelligence and Data Mining*, 2007. CIDM 2007. IEEE Symposium on. IEEE, 2007.

[7] David, Nichols. Implicit Rating and Filtering. *5th DELOS Workshop on Filtering and Collaborative Filtering (ERCIM), Budapest, Hungary.* 1997.

[8] Sarwat, Mohamed, James Avery, and Mohamed F. Mokbel. RecDB in action: recommendation made easy in relational databases. *Proceedings of the VLDB Endowment* 6.12 (2013): 1242-1245.

[9] Su, Xiaoyuan, and Taghi M. Khoshgoftaar. A survey of collaborative filtering techniques. *Advances in artificial intelligence* 2009 (2009): 4.

[10] Adomavicius, Gediminas, Nikos Manouselis, and YoungOk Kwon. Multi-criteria recommender systems. *Recommender systems handbook.* Springer US, 2011. 769-803.

[11] Lops, Pasquale, Marco De Gemmis, and Giovanni Semeraro. Content-based recommender systems: State of the art and trends. *Recommender systems handbook.* Springer US, 2011. 73-105.

[12] Burke, Robin. Hybrid recommender systems: Survey and experiments. *User modeling and user-adapted interaction* 12.4 (2002): 331-370.

[13] Park, Youngki, et al. Reversed CF: A fast collaborative filtering algorithm using a k-nearest neighbor graph. *Expert Systems with Applications* 42.8 (2015): 4022-4028.

[14] Gunawardana, Asela, and Christopher Meek. A unified approach to building hybrid recommender systems. *Proceedings of the third ACM conference on Recommender systems.* ACM, 2009.

[15] Burke, Robin. Hybrid web recommender systems. *The adaptive web. Springer Berlin Heidelberg,* 2007. 377-408.

[16] Amato, Giuseppe, and Umberto Straccia. User profile modeling and applications to digital libraries. *Research and Advanced Technology for Digital Libraries.* Springer Berlin Heidelberg, 1999. 184-197.

[17] Kuflik, Tsvi, and Peretz Shoval. Generation of user profiles for information filtering—research agenda (poster session). *Proceedings of the 23rd annual international ACM SIGIR conference on Research and development in information retrieval.* ACM, 2000.

[18] Davidson, James, et al. The YouTube video recommendation system. *Proceedings of the fourth ACM conference on Recommender systems.* ACM, 2010.

[19] Gomez-Uribe, Carlos A., and Neil Hunt. The Netflix Recommender System: Algorithms, Business Value, and Innovation. *ACM Transactions on Management Information Systems (TMIS)* 6.4 (2015): 13.

[20] Rashid, Al Mamunur, George Karypis, and John Riedl. Learning preferences of new users in recommender systems: an information theoretic approach. *ACM SIGKDD Explorations Newsletter* 10.2 (2008): 90-100.

[21] Anil, Robin, Ted Dunning, and Ellen Friedman. *Mahout in action.* Shelter Island: Manning, 2011.

[22] Sarwar, Badrul, et al. Item-based collaborative filtering recommendation algorithms. *Proceedings of the 10th international conference on World Wide Web.* ACM, 2001.

[23] Gunes, Ihsan, et al. Shilling attacks against recommender systems: a comprehensive survey. *Artificial Intelligence Review* 42.4 (2014): 767-799.

[24] Musto, Cataldo. Enhanced vector space models for content-based recommender systems. *Proceedings of the fourth ACM conference on Recommender systems.* ACM, 2010.

[25] Trauzettel-Klosinski, Susanne, and Klaus Dietz. Standardized Assessment of Reading Performance: The New International Reading Speed Texts IReSTStandardized Assessment of Reading Performance. *Investigative ophthalmology visual science* 53.9 (2012): 5452-5461.

[26] Kellar, Melanie, et al. Effect of task on time spent reading as an implicit measure of interest. *Proceedings of the American Society for Information Science and Technology* 41.1 (2004): 168-175.

[27] Pazzani, Michael J., and Daniel Billsus. Content-based recommendation systems. *The adaptive web*. Springer Berlin Heidelberg, 2007. 325-341.

[28] Schapire, Robert E., Yoram Singer, and Amit Singhal. Boosting and Rocchio applied to text filtering. *Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, 1998.

[29] Wang, Chong, et al. Improving Rocchio Algorithm for Updating User Profile in Recommender Systems. *Web Information Systems Engineering–WISE 2013*. Springer Berlin Heidelberg, 2013. 162-174.

[30] Linden, Greg, Brent Smith, and Jeremy York. Amazon. com recommendations: Item-to-item collaborative filtering. *Internet Computing, IEEE* 7.1 (2003): 76-80.

[31] Pokorný, Jaroslav, Václav Snášel, and Michal Kopecký. *Dokumentografické informační systémy*. Karolinum, 2005.

[32] Laclavík, Michal, et al. Dostupné zdroje a výzvy pre počítačové spracovanie informačných zdrojov v slovenskom jazyku. *1 st Workshop on Intelligent and Knowledge oriented Technologies*. 2006.

[33] Krátký, Michal. Využití SVD pro indexování latentní sémantiky. *Ostrava: VŠB Technická univerzita Ostrava* (2002).

[34] Turney, Peter D., and Patrick Pantel. From frequency to meaning: Vector space models of semantics. *Journal of artificial intelligence research* 37.1 (2010): 141-188.

[35] Ramos, Juan. Using tf-idf to determine word relevance in document queries. *Proceedings of the first instructional conference on machine learning*. 2003.

[36] Zhang, Wen, Taketoshi Yoshida, and Xijin Tang. A comparative study of TF*IDF, LSI and multi-words for text classification. *Expert Systems with Applications* 38.3 (2011): 2758-2765.

[37] YouTube Official Blog [online]. 23.01.2012, [cit. 2016-03-08]. *YouTube Official Blog*, Dostupné z : <https://youtube.googleblog.com/2012/01/holy-nyans-60-hours-per-minute-and-4.html>

[38] Netflix Prize [online]. [cit. 2016-03-21]. *Netflix Prize*, Dostupné z: <https://en.wikipedia.org/wiki/Netflix_Prize>

A Obsah CD

Súčasťou tejto diplomovej práce je priložené CD, ktoré obsahuje všetky zdrojové kódy, inštalačnú príručku a konfiguračnú príručku pre použitie systému.