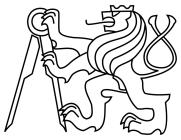
ČESKÉ VYSOKÉ UČENÍ TECHNICKÉ V PRAZE
FAKULTA INFORMAČNÍCH TECHNOLOGIÍ



ZADÁNÍ DIPLOMOVÉ PRÁCE

Název:	Integrace doporučovacího systému do portálu SSP
Student:	Bc. Josef Dvořák
Vedoucí:	Ing. Stanislav Kuznetsov
Studijní program:	Informatika
Studijní obor:	Webové a softwarové inženýrství
Katedra:	Katedra softwarového inženýrství
Platnost zadání:	Do konce letního semestru 2016/17

Pokyny pro vypracování

Cílem práce je integrace vybraného doporučovacího systému do portálu SSP (Spolupráce studentů s průmyslem). Doporučovací systém bude například doporučovat zadání studentům, studenty k zadání a vylepší našeptávání při hledání informací na portálu.

1. Seznamte se s problematikou doporučovacích systémů a proveďte rešerši doporučovacích algoritmů.
2. Proveďte rešerši metod pro porovnávání doporučovacích algoritmů a zvolte vypovídající metriky.
3. Proveďte analýzu současného stavu doporučování v SSP.
4. Navrhněte a implementujte případy užití doporučování v SSP. Tedy, proveďte integraci vybraných doporučovacích metod/systémů ve vhodných případech užití v SSP.
5. Celkové řešení otestujte pomocí vybraných testovacích metrik.

Seznam odborné literatury

Dodá vedoucí práce.

L.S.

Ing. Michal Valenta, Ph.D.
vedoucí katedry

prof. Ing. Pavel Tvrdík, CSc.
děkan

V Praze dne 15. října 2015

České vysoké učení technické v Praze
Fakulta informačních technologií
Katedra softwarového inženýrství

Diplomová práce

Integrace rekomendačního systému do portálu SSP

Bc. Josef Dvořák

Vedoucí práce: Ing. Stanislav Kuznetsov

1. července 2016

Poděkování

Chtěl bych poděkovat Ing. Stanislavu Kuznetsovovi za správné směrování, cenné rady a řešení otázek, se kterými jsem se na něj obracel. Dále bych chtěl podekovat rodině a všem blízkým za podporu při psaní mé práce a za podporu během celých mých studií.

Prohlášení

Prohlašuji, že jsem předloženou práci vypracoval(a) samostatně a že jsem uvedl(a) veškeré použité informační zdroje v souladu s Metodickým pokynem o etické přípravě vysokoškolských závěrečných prací.

Beru na vědomí, že se na moji práci vztahují práva a povinnosti vyplývající ze zákona č. 121/2000 Sb., autorského zákona, ve znění pozdějších předpisů. V souladu s ust. § 46 odst. 6 tohoto zákona tímto uděluji nevýhradní oprávnění (licenci) k užití této mojí práce, a to včetně všech počítačových programů, jež jsou její součástí či přílohou, a veškeré jejich dokumentace (dále souhrnně jen „Dílo“), a to všem osobám, které si přejí Dílo užít. Tyto osoby jsou oprávněny Dílo užít jakýmkoli způsobem, který nesnižuje hodnotu Díla, a za jakýmkoli účelem (včetně užití k výdělečným účelům). Toto oprávnění je časově, teritoriálně i množstevně neomezené. Každá osoba, která využije výše uvedenou licenci, se však zavazuje udělit ke každému dílu, které vznikne (byť jen zčásti) na základě Díla, úpravou Díla, spojením Díla s jiným dílem, zařazením Díla do díla souborného či zpracováním Díla (včetně překladu), licenci alespoň ve výše uvedeném rozsahu a zároveň zpřístupnit zdrojový kód takového díla alespoň srovnatelným způsobem a ve srovnatelném rozsahu, jako je zpřístupněn zdrojový kód Díla.

V Praze dne 1. července 2016 .....

Odkaz na tuto práci

Dvořák, Josef. *Integrace rekomendačního systému do portálu SSP*. Diplomová práce. Praha: České vysoké učení technické v Praze, Fakulta informačních technologií, 2016.

Abstrakt

Tato práce se věnuje možnostem využití rekomendačních systémů a problematice jejich technik a používaných algoritmů. Dále se v práci zabývám porovnáním jednotlivých rekomendačních systémů a jejich hodnotícími metrikami. V praktické části popisuji návrh a implementaci aplikace pro vyhodnocování a poměřování rekomendačních systémů. V práci jsou také zahrnuty výsledky porovnávání doporučovacích modelů vybraného rekomendačního systému.

Klíčová slova rekomendační systémy, portál SSP, vyhodnocovací metriky RS, Collaborative filtering, Cold start problém

Abstract

This thesis concern with problems and possibilities of usage recommendation system (RS) and with problems of recommendation techniques and algorithms. It also concern with comparing and evaluation metrics of RS. The theoretical part of a thesis presents the domain of RS and actual problems. Then it presents the metrics and evaluation techniques that used for comparing different RS. At the last part, it presents the summary of actual RS a choose the most appropriate. The practical part of work describe design and implementation of an application for comparing and evaluating RS. The test part of the thesis includes results of comparing of recommendation models of selected RS.

Keywords recommendation systems, portal SSP, evaluation metrics of RS, Collaborative filtering, Cold start problem

Obsah

Seznam obrázků

Seznam tabulek

Úvod

Portál spolupráce studentů s průmyslem (SSP) je informační systém, který umožňuje průmyslovým partnerům zadávat pro studenty projekty. Studenti si z nabízených projektů, vyberou ty projekty, co je zajímají, a nebo se jim hodí jako zadání semestrální práce v rámci některého předmětu. Studenti za splnění zadání projektu získají hlavně zkušenosti a kontakty z praxe, ale také kredity a finanční odměnu. Průmysloví partneři získají možnost, využít zkušeností univerzitních expertů, spolupracovat s mladými talenty a ušetřit lidské zdroje na vedení projektu.

S přibývajícím počtem zadaných projektů v systému SSP je pro studenty čím dál tím víc časově náročnější pročíst většinu nabízených projektů a také se začal objevovat problém s včasným nalezením řešitelů z řad studentů pro některé projekty. Proto se objevila myšlenka využít doporučovacího (rekomendačního) systému pro doporučování zadání projektů studentům a pro doporučení studentů pro projekty.

Kapitola 1

Cíl práce

Cílem práce seznámit se s problematikou rekomendačních systémů, navrhnout a implementovat co nejefektivnější případy užití pro využití RS v informačním systému SSP.

Část I

Teoretická část

Kapitola 2

Úvod do problematiky rekomendačních systémů

Rekomendační (doporučovací) systémy jsou softwarové nástroje a techniky poskytující uživateli návrhy, které položky z databáze by mohl využít[2]. V komerční sféře se RS nejčastěji využívají k navrhování položek uživateli v e-shopu, filmů ke zhlédnutí ve filmových databázích, článků nebo knih k přečtení. Rekomendační systémy dokáží uživateli pomoci s výběrem položek v nepřeberných databázích a tím uživateli šetří čas. Rekomendační systémy jsou založeny na různých algoritmech a vycházejí z různých dat. Z rostoucím počtem dat a počtem uživatelů se RS stávají více komplexnější a spojují další oblasti informatiky jako jsou data minning [3], machine learning [4], information retrieval [5] a dalších.

Rekomendační systémy můžeme dělit na 6 níže zmíněných typů RS dle techniky doporučování [2].

- **Content-based:**

  RS systémy založené na podobnosti doporučovaných položek. Podobnost položek se počítá z hodnot atributů (features) porovnávaných položek. Pokud bychom doporučovali automobily, na základně features spotřeba benzínu a objem kufru, tak RS uživateli, který projevil zájem o automobil s danou spotřebou benzínu a objemem doporučí automobily, které budou mít co nejpodobnější hodnoty těmto atributům[2].

- **Collaborative filtering:**

  Tento typ RS doporučuje položky, které si oblíbili uživatelé s podobným vkusem jako uživateli, kterému zrovna doporučujeme. Podobný vkus se vypočítává pomocí minulých interakcí uživatele s interakcemi ostatních uživatelů. Tato technika je považována za nejoblíbenější a často implementovanou[2].

- **Demografické:**

  Demografické RS člení uživatele do různých skupin podle jejich demografického profilu. Profily se vytváří například pomocí věku, jazyka, kterým uživatel mluví, země, v které žije, pohlaví výše příjmu [2].

- **Knowledge-based:**

  Tento typ RS doporučuje na základě specifické znalosti domény systému. Knowledge-based RS doporučuje na základě shody požadavků uživatele s tím, co daná položka nabízí. Například když budeme o uživateli hobby marketu vědět, že je lakýrník, bude mu RS nabízet položky z této kategorie produktů [2].

- **Community-based:**

  RS založené na komunitách, doporučují uživateli položky, které kladně ohodnotili jeho přátelé. Tato technika sází na to, že lidé zjevně dávají víc na doporučení od přítele, než na podobnost s anonymními uživateli [2].

- **Hybrid-based:**

  Hybridní RS systémy kombinují přístup dvou a více výše zmíněných přístupů. Toho se využívá k tomu, aby se eliminovaly nevýhody první techniky, technikou druhou. Například níže zmíněný Cold Start problém u Collaborative filterig technik, lze vyřešit tím, že pro uživatele s málo interakcemi v RS využijeme například Content-based techniku [2].

2.1 Rekomendační algoritmy

Pro implementaci RS je možné použít spoustu data miningových algoritmů jako kNN (k-nearest neighbors)[1], rozhodovací stromy, Bayesovské sítě, SVM (Support vector machines), ANN (artificial neural network)[2], Asociační pravidla[7] a další. Tyto algoritmy lze použít pro různé výše zmíněné techniky.

Content-based techniku a Collaborative filtering techniku, lze ještě rozdělit dle toho, jestli pro doporučování využijeme podobnosti uživatelů, to se jedná o user-based přístup, nebo využijeme podobnosti doporučovaných položek, pak se jedná o item-based.

Item-based Při doporučování položky I uživateli U nejprve vezmeme množinu položek, které uživatele U v systému zaujaly a těmto položkám najdeme podobné položky a ty uživateli U doporučíme.

[1] k nejbližších sousedů

[2] umělé neuronové sítě

User-based Při doporučování položky I uživateli U můžeme nejprve najít množinu nejpodobnějších uživatelů uživateli U a uživateli U doporučit položku, která zaujala co nejvíc uživatelů z nalezené množiny podobných uživatelů.

2.2 Cold start problém

Obecně se dá říct, že čím víc máme informací o uživateli a jeho interakcích v systému, tím lepší doporučení od RS dostáváme. Problém nastává, když máme velmi málo těchto dat nebo žádná. Pak je třeba zvážit, které atributy by nám mohly přinést požadované doporučování. Když se nám nedaří najít vhodné řešení, může se stát paradox , že když se nám nedaří nabídnout kvalitní doporučování, uživatelé přestanou používat RS, což povede k tomu, že se nám nemusí podařit zvýšit kvalitu doporučování se snižující se aktivitou uživatelů. Tento netriviální problém se nazývá cold start problem. Ukážeme si tři typy tohoto problému[6].

Cold start problém nového systému nastává, když nemáme žádná počáteční hodnocení od uživatelů z důvodu chybějících uživatelů v systému. V této situaci většina doporučovacích systémů nemá žádný základ, podle kterého by mohl doporučovat a stěží může fungovat.

Cold start problém nového uživatele je tehdy, když je RS už nějakou dobu zaběhnutý a když existuje množina profilů uživatelů a hodnocení položek, ale v systému neexistuje informace o novém uživateli. Většina RS v této situaci vykazuje velmi špatné výsledky.

Doporučovací systémy založené na CF se nemůžou vypořádat s tímto problémem, protože nemohou najít skupinu podobný uživatelů se stejným chováním, když u uživatel zatím v systému ještě neevidujeme žádné jeho interakce s položkami. Content-based a hybrid-based RS jsou na tom o něco lépe, neboť potřebují menší množství informací k tomu, aby našly podobnosti.

Cold start problém nové položky nastává, když do systému přibude nová položka. Zde nastává problém u RS založených na CF, když žádný uživatel ještě s touto položkou neinteragoval a systém se k doporučení této položky nemá jak dostat. Content-based RS tento problém nemají.

Kapitola 3

Testování rekomendačních systémů

Existuje spousta různých RS založených na různých algoritmech a technikách, proto se objevila potřeba tyto systémy porovnávat. Pro porovnávání a testování RS existují tři hlavní přístupy: online testování, offline testování a user studies.

Online testování RS spočívá v testování RS v živém provozu. Pro online testování RS se nejvíce používá technika zvaná A/B testování, převzatá z marketingu a Business intelligence. Při této technice se náhodně rozdělí množina uživatelů na dvě poloviny, tak aby různé podskupiny uživatelů byly v každé polovině zastoupeny rovnoměrně, a nad každou polovinou uživatelů se pustí různé rekomendační systémy a ty se poté porovnají[8].

User studies se provádí, tak že se osloví skupina lidí, které se připraví scénář s jednotlivými postupy, které se tito uživatelé snaží provést. Před, během a po provádění je možné uživatelům pokládat různé otázky ohledně testovaného systému. Hlavní výhodou tohoto přístupu testování je získání odpovědí na otázky kvalitativního charakteru, které není možné pomocí online a offline testování získat. Nevýhodou tohoto přístupu je časová náročnost na testování uživateli[9].

Offline testování je založeno na měření převážně výkonnostních metrik nad již nasbíranými daty o interakcích uživatelů s položkami v systému. Pomocí nasbíraných dat můžeme simulovat chování uživatelů v systému a hlavní výhodou tohoto přístupu je, že není potřeba spolupráce reálných uživatelů[9].

Křížová validace K naměření metrik slouží různé statistické metody, jednou z nich je křížová validace[10]. Obecná k-fold křížová validace rozdělí data

na k disjunktních podmnožin o velikosti *N/k* kde *N* je počet nasbíraných dat. Poté se pro každou z *k* podmnožin provede validace, tak že se tato množina vezme jako testovací data a zbylá data ze vstupní množiny poslouží jako trénovací data pro validaci. Těchto validací se provede *N/k* a výsledek se vypočítá jako průměr validací. Existuje více variant křížové validace podle velikosti *k*. Varianta ,kdy *k* se rovná *N*, se nazývá leave-one-out křížová validace. Tato validace může být oproti *k*-fold, kdy $k < N$, výrazně větší výpočetní složitost, neboť při leave-one-out křížové validaci se musí rekomendační model přepočítat *N - 1* krát namísto *k* krát.

3.1 Metriky

Při offline testování se může využívat množství různých metrik, podle kterých se RS porovnávat. Některé tu vysvětlím.

3.1.1 Metriky přesnosti predikce ratingu

Pokud máme data o ohodnocení položek jednotlivými uživateli můžeme k porovnání RS použít metriky přesnosti predikce ratingu. Mezi tyto metriky patří například oblíbená RMSE (střední kvadratická chyba) a MAE (střední absolutní chyba). Systém předpovídá rating $\hat{r}_{u,i}$ pro testovací množinu $\mathcal{T}$ user-item párů (u,i), u nichž známe pravou hodnotu ratingu $r_{u,i}$ uživatele u pro položku i.

$$\text{RMSE} = \sqrt{\frac{1}{|\mathcal{T}|} \sum_{(u,i)\in\mathcal{T}} (\hat{r}_{u,i} - r_{u,i})^2} \tag{3.1}$$

$$\text{MAE} = \frac{1}{|\mathcal{T}|} \sum_{(u,i)\in\mathcal{T}} |\hat{r}_{u,i} - r_{u,i}| \tag{3.2}$$

RMSE na rozdíl od MAE bude upřednostňovat RS, který bude generovat víc menších chyb při odhadování ratingů pro jednotlivé user-item páry, než systém, který dělá větší chyby na pár odhadnutých ratinzích.

3.1.2 Metriky predikce využití doporučení

Spousta RS nemusí sbírat ratingy uživatelů k položkám, ale jen doporučují položky, které by je mohly zajímat. Tyto RS lze poměřovat podle následujících metrik tak, že pro každý user-item pár z testovací množiny ze systému odstraní interakce tohoto páru a poté se nechá systémem doporučit *N* položek. V závislosti zda uživatel o danou položku skutečně projevil zájem a jestli se položka objevila mezi *N* doporučeními rekomendačního systému, rozlišujeme tyto 4 různé výsledky testu.

Metriky predikce využití doporučení jsou vhodnější pro poměřování RS, které řeší takzvanou *top N doporučovací úlohu* (top N recommendation task). To je taková úloha, kde nám stačí doporučit uživateli seznam položek o velikosti N, v kterém se vyskytne jen pár položek, které uživatele nejvíce osloví. Běžně používané metriky jako RMSE a MAE se pro tuto úlohu moc nehodí, při top N doporučovaní úloze nás chyby v predikci ratingu uživatelů přímo nezajímají[11].

Tabulka 3.1: Možnosti výsledků jednotlivých testů

	Recommended	Not recommended
Used	True-Positive (TP)	False-Negative (FN)
Not Used	False-Positive (FP)	True-Negative (TN)

- TP položky doporučené systémem a uživatel o ně projevil zájem.
- FN položky nedoporučené systémem a uživatel o ně projevil zájem.
- FP položky doporučené systémem a uživatel o ně neprojevil zájem.
- TN položky nedoporučené systémem a uživatel o ně neprojevil zájem.

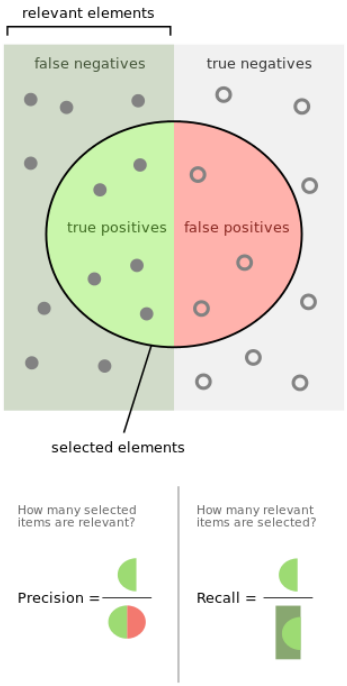
Precision je poměr relevantně doporučených položek, k celkovému počtu doporučených položek. Tato metrika se spíše hodí pro online testování nebo user studies, kde se uživatele zeptáme, jestli je pro něj doporučená položka relevantní. Při offline testování máme většinou informace o tom, zda o danou položku uživatel projevil zájem, ale nevíme přesně, které položky ho nezajímají. Považovat položky, o které uživatel neprojevil zájem, za nerelevantní není úplně správné, protože uživatel nemusel o položku neprojevil zájem z důvodu dozatimní nevědomosti o její existenci v systému[1].

Recall je poměr relevantně doporučených položek, k celkovému počtu relevantních položek, tedy těch o kterých víme, že o ně uživatel někdy projevil zájem[1][12].

$$\text{Precision} = \frac{\#\text{TP}}{\#\text{TP} + \#\text{FP}} \qquad \text{Recall} = \frac{\#\text{TP}}{\#\text{TP} + \#\text{FN}} \tag{3.3}$$

F1 score je metrika, která bere v úvahu jak precision, tak i recall a vypočítá se dle následujícího vzorce.

$$\text{F} = 2 \cdot \frac{\text{precision} \cdot \text{recall}}{\text{precision} + \text{recall}} \tag{3.4}$$



Obrázek 3.1: Precision s Recall[1]

3.1.3 Ostatní metriky

Coverage je termín, který popisuje více vlastností RS. Obecně popisuje kolik různých položek je RS schopný doporučit z celkové nabídky dostupných položek. Pokud se coverage dále nespecifikuje, většinou je na mysli takzvaná *catalog coverage*. Catalog coverage se vypočítá pomocí následujícího vzorce, kde $\mathcal{I}$ označuje množinu všech dostupných položek v RS, $\mathcal{RI}$ je seznam doporučených položek jednoho doporučení, z množiny všech doporučení o velikosti N během testování[13].

$$\text{Cataloge Coverage} = \frac{|\cup_{j=1...N} \mathcal{RI}_j|}{|\mathcal{I}|} \tag{3.5}$$

U RS založených na CF může nastávat problém, že systém doporučuje z pohledu metrik jako preccision a recall velmi přesně, ale doporučuje jen malou část položek z nabídky. Pokud máme zájem na tom najít uživatele pro co nejširší nabídku položek vyplatí se obětovat trochu přesnosti doporučování pro zvýšení coverage.[12].

Diverzita je další vlastnost, která stojí u RS za pozornost. Většinou pomocí RS chceme uživateli nabídnout co nejrozmanitější nabídku položek, která by ho zajímala. Diverzitu doporučeného seznamu položek můžeme poměřovat

pomocí sumy podobností dvojic z doporučeného seznamu položek. Zvyšování diverzity stejně jako zvyšování coverage nejspíše povede ke snížení přesnosti doporučování.

Kapitola 4

Rešerše

Současně se v portálu SSP doporučuje pomocí Knowledge-based techniky na základě podobnosti úrovně dovedností studentů s požadovanou úrovní dovedností řešitele zadání v SSP. Studentova úroveň dovedností je buď vypočítána ze známek z předmětů, které student absolvoval nebo se využije studentovo samo-ohodnocení dovedností. Viz kód níže.

```
public double calculateEuklid(Student student, List<SkillDto>
    assignmentSkills) {

   double computeValue = 0;
      for (SkillDto skillDto : assignmentSkills) {
      final Integer skillStars = skillDto.getStars();
      final Integer studentStars = studen.getStars(skillDto);
      if (studentStars >= skillStars) {
         continue;
      }
      computeValue += (skillStars - studentStars) * (skillStars -
          studentStars);
   }
   return sqrt(computeValue);
}
```

Tento přístup se zdá na první pohled dost přínosný, ale přináší řadu úskalí. Například studentům, kteří mají absolvovaných jen pár předmětů systém portálu SSP nemůže vypočítat úrovně dovedností a studentovo samohodnocení je pravděpodobné, že bude dost subjektivní a dovednosti jednotlivých studentů nejde objektivně porovnávat. To že student má vysokou úroveň dovednosti v jisté oblasti, že ho zrovna bude zajímat zadání projektu, kde se tato dovednost vyžaduje.

4.1 Dostupné rekomendační systémy

Při hledání použitelných rekomendačních systémů pro integraci do portálu jsem objevil tyto komerční nástroje:

Microsoft Azure Machine Learning Studio `https://studio.azureml.net/` se zdá být velmi propracované drag-and-drop rozhraní pro nejrůznější data miningové úlohy. Toto rozhraní je možné rozšířit o skriptovací Python modul Jupyter Notebooks. Rozhraní umožňuje i testovat a porovnávat rekomendační modely.

Nosto `http://www.nosto.com/` a BrainSINS `http://www.brainsins.com/en/` nabízí personalizované doporučovaní produktů pro eshopy s Javascriptovým API, bohužel se mi nepodařilo zjistil více o tom, jaké rekomendační modely využívají.

Dato `https://dato.com/` také nabízí personalizované doporučování s podporou vlastní Pythoní knihovnu GraphLab Create, která umožňuje porovnávat a vyhodnocovat rekomendační modely. Nepodařilo se mi zjistit, pomocí kterých metrik jde modely porovnávat.

Pro integraci RS do portálu SSP jsem rozhodl pro rekomendační systém *Recombee* `https://www.recombee.com/`, protože na jeho vývoji se podílí kolegové z FIT, kteří mi poskytli přístup k HTTP API, pomocí kterého se dají nastavovat různé parametry jednotlivých rekomendačních modelů a dají se kombinovat.

4.2 Recombee

Recombee nabízí 5 níže zmíněných rekomendačních modelů.

UserkNN model je v Recombee zjednodušený název pro User-based, CF, Non-normalized Cosine Neighborhood k-NN Algoritmus[11].

Ranking Item kNN model je obdobný jako *UserkNN model* jen místo User-based přístupu je použit Item-base přístup, tedy algoritmus najde pro pro všechny uživatelem ohodnocené položky k nejpodobnějších položek a z těch vybere N nejčastěji zmiňované položky mezi podobnými položkami položkám, které už uživatel ohodnotil.

Token Item kNN model je stejně jako *Ranking Item kNN model* Item-base a je to Non-normalized Cosine Neighborhood k-NN Algoritmus, jen místo CF pro měření podobnosti využívá Contet-Based přístup a tedy k výpočtu podobnosti nevyužívá interakce uživatelů s položkami, ale hodnoty atributů položek.

Associační pravidla jsou analytická metoda, při které se hledají asociace mezi různými položkami sortimentu, které poté použijí k doporučení dalších položek. Touto problematikou se začal zabývat Rakesh Agrawal[14], který hledal asociace mezi položkami nákupů v supermarketech. V Recombee se hledají pravidla pro uživatele

Pro danou trojici $(\mathcal{I}, \mathcal{U}, s_{min})$, kde $\mathcal{I}$ je množina položek, $\mathcal{U}$ množina uživatelů RS a $s_{min} \in (0.0, 1.0]$ je hodnota minimální podpory(support) pro uplatnění pravidla pro doporučování, je asociační pravidlo taková implikace $X \Rightarrow Y$ kde $X \cup Y \in U$, $X \neq \emptyset$, $Y \neq \emptyset$, $X \cap Y \neq \emptyset$. Pravidlo podporované, když platí:

$$\frac{|T \in \mathcal{U} | X \cup Y \in T|}{|\mathcal{U}|} \geq s_{min} \tag{4.1}$$

Popularity-biasing parametr β Výše zmíněné rekomendační modely dost upřednostňují bestsellery a tím se snižuje coverage doporučování. Proto tvůrci Recombee implementovali, takzvaný Popularity-biasing parametr, který nejoblíbenější položky penalizuje a tím dá šanci položkám, které nejsou v tak často hodnoceny, abychom dosáhli větší coverage[12]. Popularita se vypočítá pomocí následujícího vzorce.

$$\text{popularity}(i) = \sum_{u \in \mathcal{U}} (r_{u,i} - \overline{ru}) \tag{4.2}$$

Takto vypočítaná popularita se následovně využije pro snížení ohodnocení(ranku) získaného pomocí některého modelu, tímto způsobem.

$$\text{rank}^{\beta}_{PS}(u,i) = \frac{\text{rank}(u,i)}{\text{popularity}(i)^{\beta}} \tag{4.3}$$

Je doporučeno, aby $\beta \in [0.0, 1.0]$.

Recombee umožňuje přímo ovlivnit diverzitu nastavením parametru diversity v intervalu $[0.0, 1.0]$, kde s nastavením *diversity = 0.0* bude systém doporučovat i naprosto podobné položky a se zvyšující se hodnotou k *diversity = 1.0* bude systém doporučovat co nejméně podobné položky za cenu doporučování položek s menším ratingem.

4.3 Návrh experimentů

Za zhruba dva roky fungování portálu SSP se nashromáždila data o 920 studentech, kteří se do portálu přihlásili a hledali v něm zadání prací, které tam byly za tu dobu vloženy. Za tuto dobu bylo evidováno zhruba 9000 zobrazení detailů zadání studenty (dále views).

To vychází odhadem průměrně za týden okolo 90-100 views, což je pro online testování nedostatečný počet. Testování pomocí přístupu user studies je obecně velmi časově náročné, nemluvě o potřebě sehnání testovacích uživatelů. Proto jsem se rozhodl testovat doporučování pomocí offline přístupu.

V portálu SSP uživatelé nehodnotí zadání, tak není možné použít pro poměřování rekomendačních modelů nad daty portálu SSP metriky pro měření chybovosti odhadování ratingu položek jako *RMSE* a *MAE*. Protože portál SSP je "párovací"systém, pro jedno zadání se snaží najít jednoho řešitele, moc nás nezajímá kolik doporučených položek je relevantních pro uživatele, ale zajímá nás, jestli se najde mezi doporučenými položkami, aspoň jedna relevantní. Tedy v doméně SSP nám jde o to, zda studentovi nabídneme zadání, na které se přihlásí nebo jestli pro zadání najdeme studenta, který o něj bude mít zájem. Proto jsem se rozhodl při experimentování měřit *recall* a *precission* jsem vyhodnotil jako nezajímavou metriku pro doporučování v portálu SSP. V portálu SSP by bylo dobré, kdyby pro každé zadání našel RS studenta, který bude dané zadání řešit, proto v experimentech měřím a také dost přihlížím ke *coverage*. Průměrný počet interakcí studentů Pro naměření recall a coverage jsem se rozhodl využít leave-one-out křížovou validaci, neboť implementace rozdělení interakcí uživatelů do k foldů je kvůli velkému množství studentů s cold start problémem problematická. A také data portálu SSP nejsou tolik objemná, aby leave-one-out křížová validace přinášela problém s výpočetní náročností oproti k-fold, kdy $k < N$.

4.4 Aplikace pro porovnávání a vyhodnocování rekomendačních systémů

Během seznamování se rekomendačnícm systémem Recombee jsem zjistil, že pro poměřování RS nad relativně malou databází uživatelů neexistuje, jako je databáze interakcí portálu SSP, neexistuje žádný offine nástroj. V komerční sféře se RS využívají hlavně v systémech, kde jsou tisíce uživatelů a pro porovnávání RS se využívá především online testování, konkrétně A/B testování[8].

Po konzultaci s vedoucím práce a týmem vývojářů portálu SSP jsme se dohodli, že implementace integrace doporučování bude spočívat hlavně v nastavení komunikace s API systému Recombee a nalezení optimálního rekomendačního modelu a hodnot jeho parametrů. Dále jsme po konzultaci s týmem vývojářů týmu SSP dospěli k názoru, že doporučování pomocí RS pro zlepšení našeptávání v portálu SSP je těžko využitelné a nemá cenu ho provádět. Proto jsem se v praktické části soustředil na implementaci aplikace, která by poměřování a banchmarking RS umožňovala provádět a dala se využít pro nalezení optimálního rekomendačního modelu a hodnot parametrů vybraného rekomendačního modelu.

Část II

Praktická část

Kapitola 5

Analýza

Z výše uvedených důvodů a potřeby nalezení optimálního rekomendačního modele s nastavením parametrů se cílem praktické části této práce stala implementace aplikace, která tento problém bude řešit. Cílem praktické části práce je implementace aplikace, která bude umožňovat připravit sadu testů s různými nastaveními parametrů, tyto testy spustit, vypočítat výsledky metrik pomocí nichž půjde nalézt nejoptimálnější nastavení parametrů rekomendačního modelu. Aplikace bude umožňovat vykreslení výsledků do přehledného grafu, pro nalezení optimálních parametrů rekomendačního modelu.

5.1 Funkční požadavky

- **F1 Vytvoření testu**

  Uživatel bude mít možnost vytvářet testy pro rekomendační modely.

- **F2 Nastavení parametrů testu**

  Uživatel bude mít možnost si vybrat, jaké parametry rekomendačního modelu bude chtít v testu iterovat a po jakých hodnotách parametru.

- **F3 Zobrazení stavu testu**

  V aplikaci bude vidět stav postupu testu v aplikaci.

- **F4 Spuštění testu**

  Uživatel bude moct na požádání zahájit testování připraveného testu.

- **F5 Výpočet výsledků testu a vykreslení výsledků**

  Uživatel bude mít možnost z nasbíraných dat testu vypočítat výsledky a vykreslit pro ně graf.

- **F6 Naplánování běhu testu**

  Uživatel bude moct naplánovat dobu spuštění testu.

- **F7 Smazání testu**

  Uživatel bude moct smazat již proběhlé testy.

- **F8 Reversní doporučování**

  Aplikace bude schopná doporučovat i ve směru doporučování uživatelů k položkám (dále pudu pro tento směr doporučování používat označení *reversní doporučování*).

Nefunkční požadavky

- **N1 Perzistentní uložení testů**
- **N2 Logování**
- **N3 Využití rozhraní Recombee API**
- **N4 Programovací jazyk - Java, verze JDK 1.8 nebo vyšší.**
- **N5 Uživatelské prostředí - knihovna JavaFX, poskytující prvky grafického rozhraní**

Případy užití

Nefunkční požadavky

- **UC1 Vytvoření testu**

  V systému bude možné vytvářet testy rekomendačních modelů.

- **UC2 Výběr testovaného modelu doporučování**

  Při vytváření testu si uživatel vybere testovaný model.

- **UC3 Nastavení parametrů testu**

  Při vytváření testu uživatel nastaví parametry testu.

- **UC4 Výběr existujícího testu**

  Uživatel si bude moct vybrat test ze seznamu testů v různém stavu.

- **UC5 Spuštění testu**

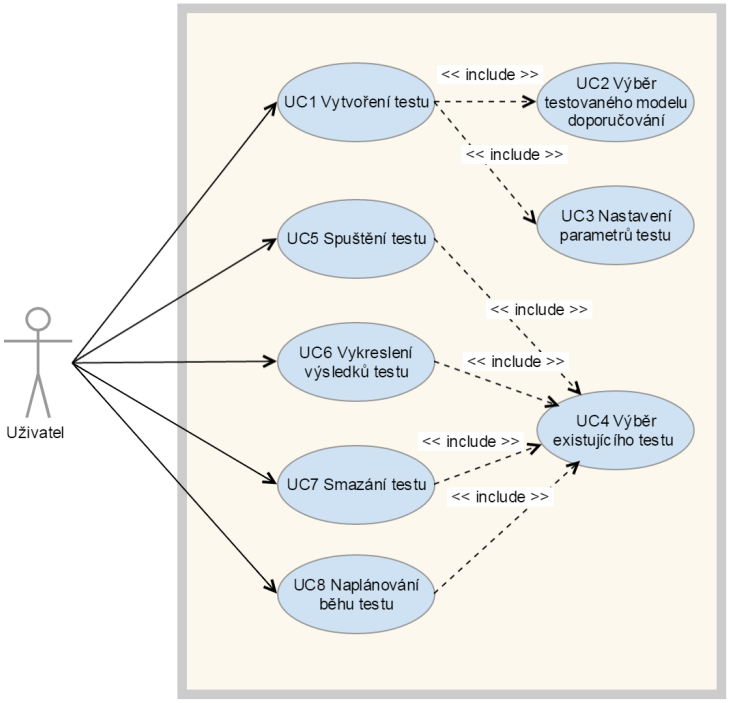
  Uživatel bude moct spustit připravené testy.

- **UC6 Vykreslení výsledků testu**

  Uživatel si bude moct nechat vykreslit graf výsledků proběhlého testu.

- **UC7 Smazání testu**

  Uživatel bude moc mazat testy ze seznamu testů.



Obrázek 5.1: Use Case Diagram

- **UC8 Naplánování běhu testu**

  Uživatel bude moct naplánovat běh připraveného testu.

- **UC9 Výběr směru doporučování**

  Uživatel bude mít možnost zvolit směr doporučování, jestli se budou doporučovat položky uživatelů, nebo reverzně položkám uživatele.

Tabulka 5.1: Tabulka pokrytí funkčních požadavků případy užití

	F1	F2	F3	F4	F5	F6	F7	F8
UC1	X							
UC2		X						
UC3		X						
UC4			X	X	X	X	X	
UC5				X				
UC6					X			
UC7							X	
UC8						X		
UC9								X

Kapitola 6

Návrh

6.1 Recombee API

Rekomendační doména Recombee se skládá ze tří komponent:

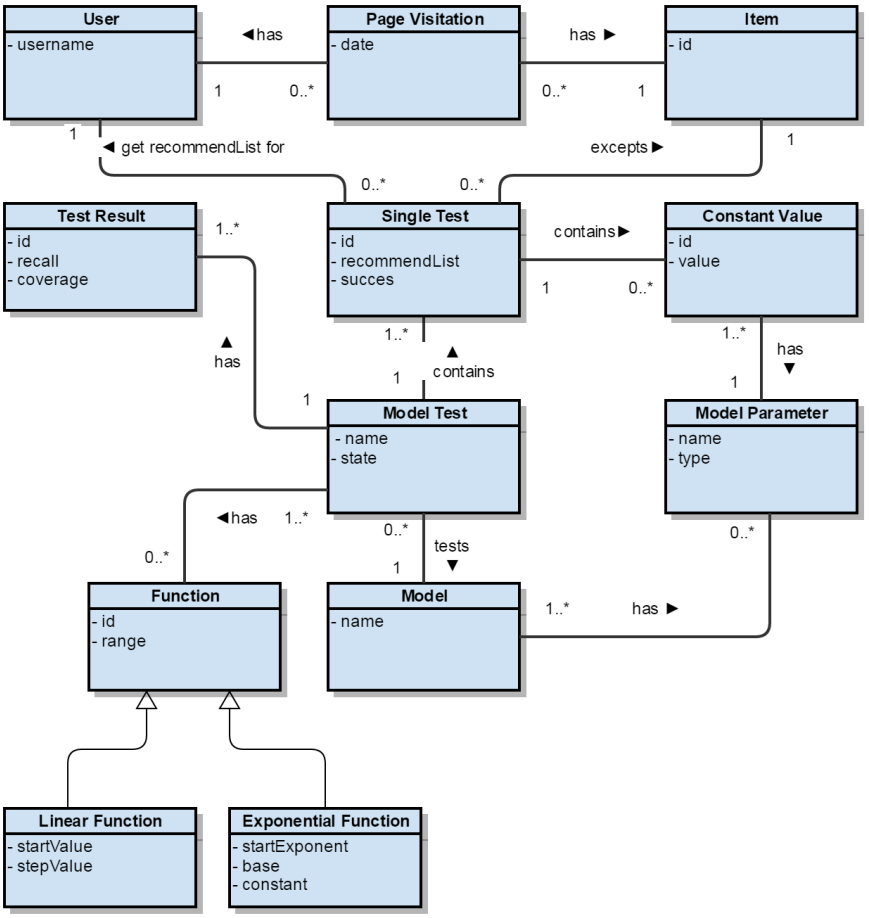
- Items - položky nabízené uživatelům
- Users - uživatelé, kteří si mají hledat položky
- User-Item Interactions - interakce mezi uživatelů s položkami

V Recombee lze od sebe rozeznat tyto typy User-Item interakcí.

- Purchase - Doméně e-Shopu se jedná o nákup zboží. Z pohledu portálu SSP se jedná o přihlášení se jako řešitel úlohy.
- Rate - Ohodnocení položky. Pro portál SSP nevidím využití, v něm studenti nehodnotí zadání projektů.
- Bookmark - Záložky.
- View details - Rozkliknutí detailu položky. V SSP nejčastější interakce.

6.2 Návrh testovací aplikace

Aplikaci jsem rozdělil tři oddělené části podle architektonického vzoru MVC(Model, View, Controller). View obsahuje hlavně třídy z knihovny JavaFX pro tvorbu GUI. Model drží data a logiku aplikace a Controller umožňuje View volat změny nad Modelem a tyto dvě zbývající části se snaží udržet co nejvíce nezávislé.



Obrázek 6.1: Doménový model

6.2.1 Doménový model

ModelTest je ústřední třída aplikace, jejíž objekty representují vytvořené sady testů rekomendačních modelů pomocí této aplikace. Sada testů je určená podle počtu různých nastavení, které chceme otestovat. Například rekomendační model User-kNN ze systému Recombee umožnuje nastavit tři parametry *alpha*, *beta* a *k-knn*. Hodnoty parametrů *alpha beta* je možné nastavovat v rozmezí -1.0 až 1.0 a hodnoty paramateru *k-knn* mohou být kladná čísla do počtu položek v databázi. Aktuálně v portálu SSP 920 studentů, takže v reverzním doporučování má smysl testovat hodnoty parametru *k-knn* do hodnoty 920. Například můžeme vytvořit sadu testů, kde budeme hodnoty parametrů *alpha beta* iterovat po hodnotě 0,1 od -1.0 (tedy -1.0, -0.9, ..., 1.0) a hodnotu *k-knn*

nastavíme konstantě třeba na hodnotu 10 , pak tedy sada bude obsahovat celkem $21 \cdot 21$ testů.

Model je třída, jejíž objekty representují rekomendační modely systému Recombee. Tato třída drží informaci o tom, jaké parametry lze u daného rekomendačního modelu nastavovat.

Single Test je třída, jejíž objekty representují jednotlivé user-view páry testované během leave-one-out křížové validace s nastavením hodnot parametrů rekomendačního modelu.

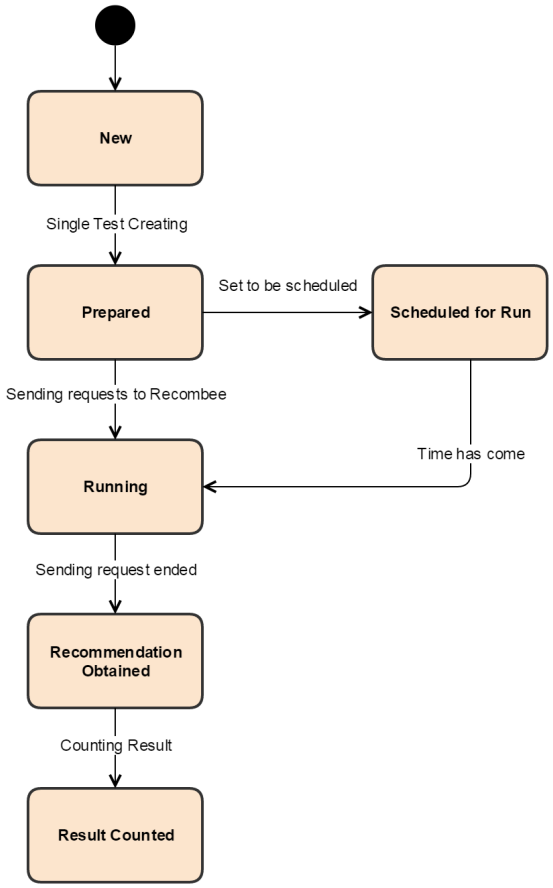
Test Result je třída, jejíž objekty representují výsledné hodnoty měřených metrik racall a coverage pro různé nastavení parametrů, které jsou representovány seznamem objektů třídy Constant Value.

Function je interface umožnující iterovat hodnoty parametrů testovaných rekomendačních modelů.

6.2.2 Stavy testů

V aplikaci se rozeznávají tyto stavy ModelTestů

- **New** Nově vytvořený test s nastavenými parametry testovaného rekomendačního systému.
- **Prepared** Stav kdy se do DB připraví dávka Single Testů, které se budou posílat do systému Recombee.
- **Running** Běžící test, jednotlivé Single Testy se posílají do systému Recombee. Tento stav může trvat i několik hodin v závislosti na počtu Single Testů.
- **RecommendationObtained** Test je proběhlý a u záznamů Single Testů je uložen seznam doporučených položek.
- **ResultCounted** Ze Single Testů jsou vypočítány výsledné hodnoty měřených metrik pro jednotlivá nastavení hodnot parametrů testovaného rekomendačního modelu.
- **PreparedToPrint** Výsledky jsou zaspány do vstupního textového souboru s daty výsledků v JSON formátu pro vykreslení grafu.
- **ScheduledForRun** Test připravený pro posílání Single Testů do systému Recombee v určenou dobu.



Obrázek 6.2: Stavový diagram entity Model Test

6.3 Návrh integrace doporučování do portálu SSP

6.3.1 Případy užití doporučování v portálu SSP

UC1 - Doporučení zadání studentovi

Aktéři: Student
Portál SSP
RS Recombee

Prerekvizity: Optimálně vybraný rekomendační model

Kroky scénáře:

1. Student se přihlásí do portálu SSP

2. Portál SSP pošle požadavek na doporučení zadání studentovi s optimálními hodnotami parametrů rekomendačního modelu.
3. Recombee vrátí portálu SSP seznam doporučených zadání.
4. Portál SSP nabídne uživateli doporučená zadání k prohlédnutí.
5. Student si zobrazí detail zadání, která ho zaujala.

UC2 - Doporučení studentů k zadání

Aktéři: Zadavatel zadání
Portál SSP
RS Recombee

Prerekvizity: Optimálně vybraný rekomendační model

Kroky scénáře:

1. Zadavatel zadání se přihlásí do portálu SSP.
2. Zadavatel si zobrazí vložené zadání, ve stavu hledání řešitele.
3. Portál SSP nabídne zadavateli zadání doporučit studenty pro zadání.
4. Zadavatel si nechá doporučit studenty jako řešitele zadání.
5. Portál SSP pošle požadavek na doporučení studentů k zadání s optimálními hodnotami parametrů rekomendačního modelu.
6. Recombee vrátí portálu SSP seznam doporučených studentů.
7. Portál SSP nabídne zadavateli odeslat doporučeným studentům email o existenci zadání projektu.
8. Zadavatel potvrdí rozeslání e-mail.
9. Portál SSP rozešle e-maily.

Získávání optimálního nastavení rekomendačního modelu Pro používání optimálního doporučování navrhuji jednou za 2-4 týdny spustit testování rekomendačních modelů nad aktuálními daty portálu SSP a případně upravovat nastavení hodnot parametrů pro optimální doporučování podle aktuálních výsledků testů, neboť s přibývajícími interakcemi studentů se zadáními, budou měnit optimální nastavení parametrů a může dojít k tomu, že jeden rekomendační model může nabývat lepších hodnot doporučování.

6.3.2 Ukázka requestů o doporučení pomocí Recombee API

Směr doporučování - zadání studentům

```
GET /<datase>/users/<username>/recomms/?count=10&settings=<settings>    HTTP/1.0
Host: rapi.recombee.com

<datase> je název databáze interakcí nahrané do Recombee

<username> username uživatele v databázi Recombee, v tomto směru doporučování studen

<settings> je JSON v následujícím formátu

"model": {
  "name": "user-knn",
  "settings": {
    "parameters": {
      "k": 80,
      "beta": 0.55
    }
  }
}
```

Z výsledků měření v příloze B pro směr doporučování zadání studentů považuji za aktuálně optimální model User-kNN model s nastavením $k = 80$ a $beta = 0.6$, kde tento model má $recall = 25\%$ a $coverage = 66\%$.

Směr doporučování - studenti k zadání

```
GET /<datase>/users/<username>/recomms/?count=10&settings=<settings>    HTTP/1.0
Host: rapi.recombee.com

<datase> je název databáze interakcí nahrané do Recombee

<username> username uživatele v databázi Recombee, v tomto směru doporučování ID zad

<settings> je JSON v následujícím formátu

"model": {
  "name": "user-knn",
  "settings": {
    "parameters": {
      "k": 3,
      "beta": -0.2
```

```
    }
  }
}
```

Z výsledků měření v příloze B pro směr doporučování studentů k zadání považuji za aktuálně optimální model User-kNN model s nastavením $k = 3$ a $beta = -0.2$, kde tento model má $recall = 17\%$ a $coverage = 38\%$.

Kapitola 7

Implementace

Aplikaci pro poměřování a evaluaci, kterou jsem nazval "Remendation Tester", je implementovaná především Javě 1.8, jen vykreslování grafů je implementováno pomocí Python skriptu s využitím knihovny Matplotlib.

7.1 Rozdělení implementace do balíčků

Database tento balíček obsahuje třídy pro komunikaci aplikace s databází Postgresql pomocí JDBC knihovny.

DataModel tento balíček obsahuje třídy entit Datamodelu z doménového modelu.

RecombeeRequest tento balíček obsahuje třídy pro sestavování http requestů do RS Recombee a parsování odpovědí na requesty ve formátu JSON. Pro sestavování requestů a parsování odpovědí ve formátu JSON, používám externí knihovnu org.json.

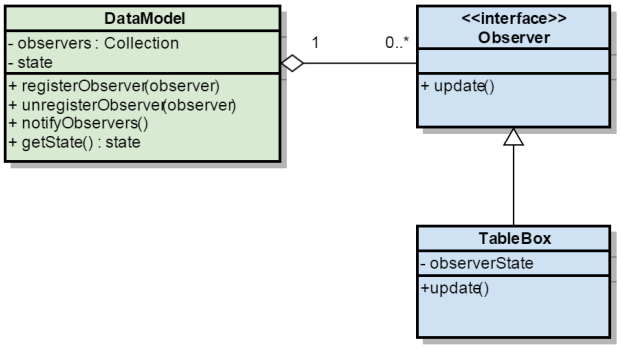
View tento balíček obsahuje třídy implementaci GUI pomocí knihovny JavaFX.

7.2 Popis tříd hlavních entit logiky aplikace

ModelTest Je ústřední třída, jejíchž instance drží data o sadě nastaveních parametrů testovaného vybraného rekomendačního modelu, vybranou testovací množinou nashromážděných interakcí z portál SSP, nad kterou se testy provádějí, směr doporučování a svůj stav.

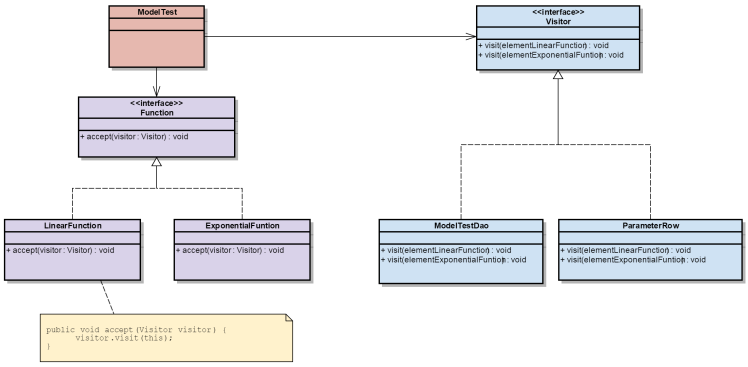
7.3 Použité návrhové vzory

Observer Při implentaci jsem použil návrhový vzor Observer pro aktualizaci pro aktualizaci View při změně v dat Modelu. View obsahuje tableView, v kterém jsou zobrazeny instance třídy ModelTest. Table box implementuje rozhraní Observer a je registrován jako Observer třídy DataModel, která spravuje instance třídy ModelTest, při změně těchto instancí DataModel notifikuje své Observery, aby si mohli aktualizovat kolekci instancí ModelTestů.



Obrázek 7.1: Observer diagram

Visitor Třída ModelTest pracuje s objekty tříd LinearFunctiona a ExponetialFunction, tyto třídy mají ve většině případů stejné chování a je možné, že v budoucnosti přibudou další podobné třídy, proto implementují interface Function a třída ModelTest s nimi pracuje pomocí tohoto interface. Jen ve dvou případech potřebuji, aby se chovali rozdílě, při jejich ukládání do databáze pomocí třídy ModelTestDAO a při vykreslování v třídě ParameterRow. Proto jsem využil návrhový vzor Visitor, abych se vyhnul if konstrukcím a volání metody instanceOf().



Obrázek 7.2: Visitor diagram

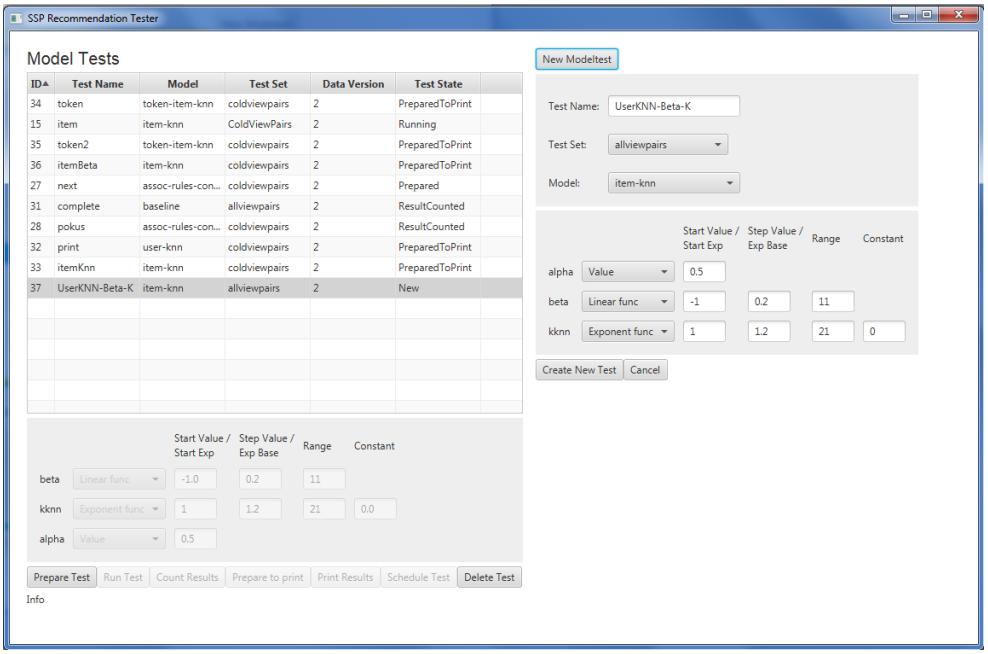
Kapitola 8

Testování

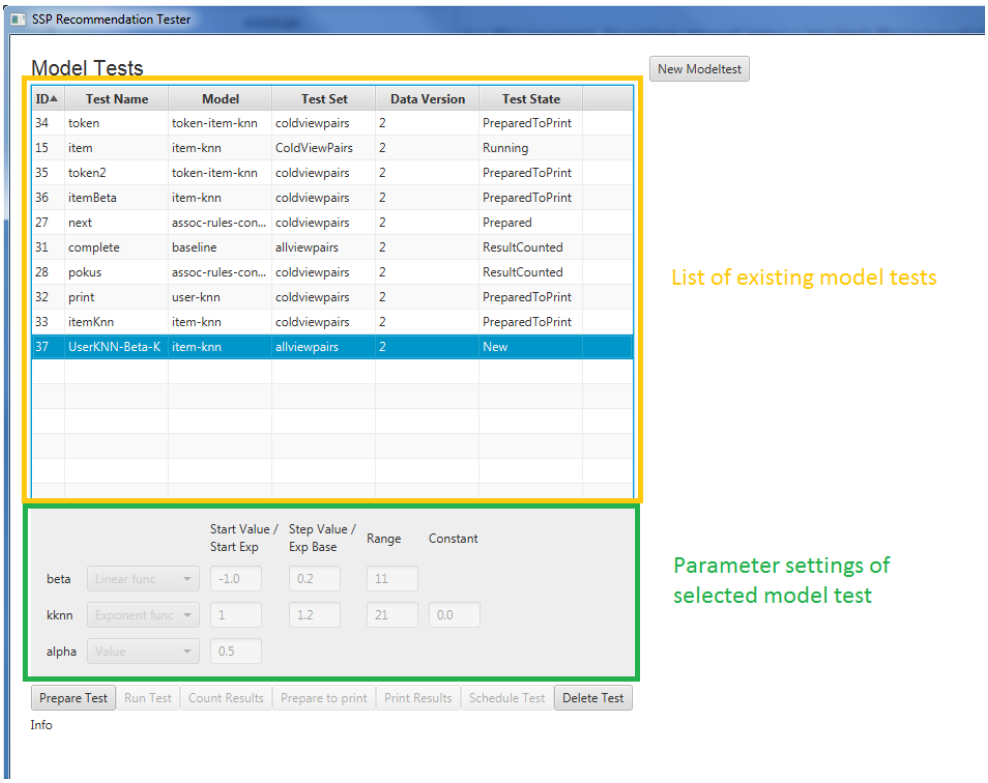
8.1 Testování aplikace

Aplikaci jsem testoval manuálně podle případů užití. Testování proběhlo a nenarazil jsem na žádné závažné chyby, které by znamenaly nesplnění požadavků na aplikaci.

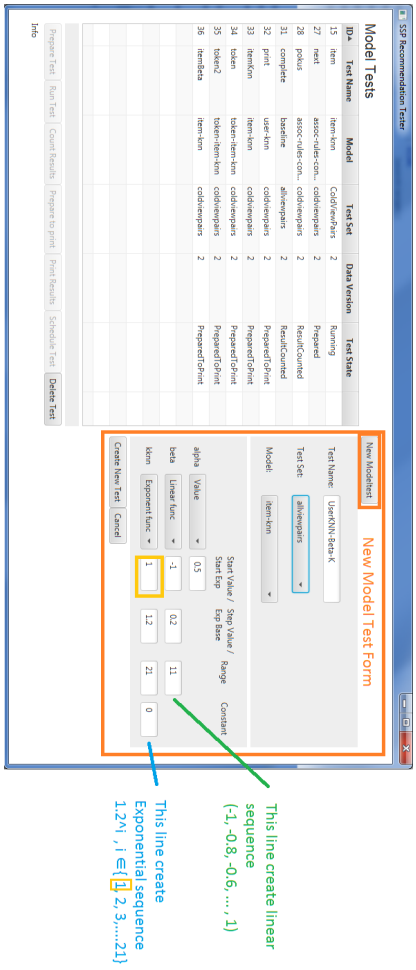
Zde jsou screenshoty zobrazující GUI aplikace, pomocí kterého byla aplikace také testována.



Obrázek 8.1: GUI aplikace

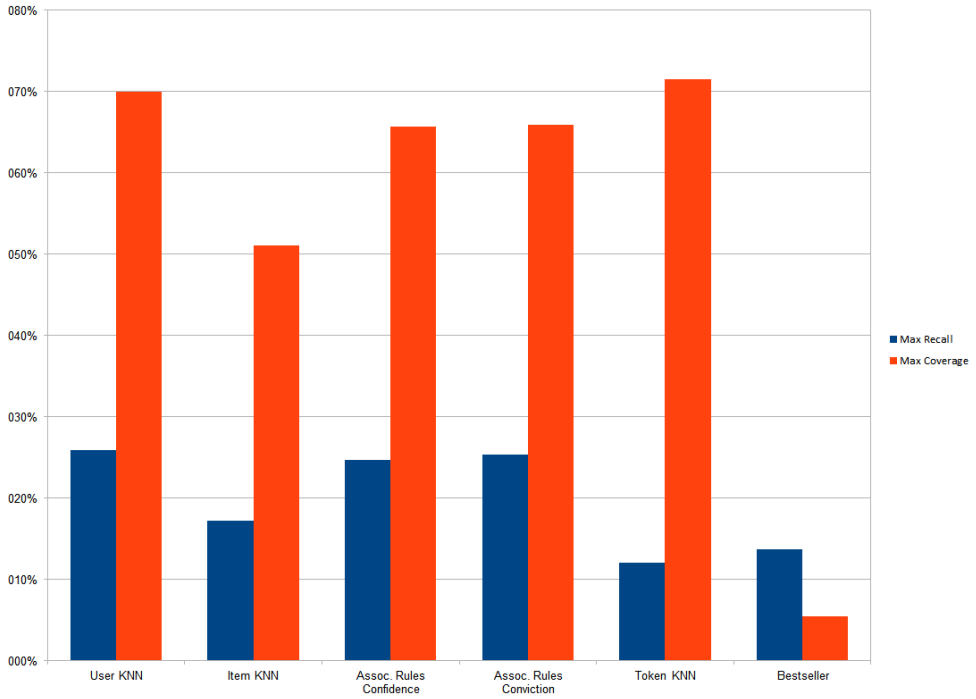


Obrázek 8.2: GUI aplikace - tabulka Model Testů

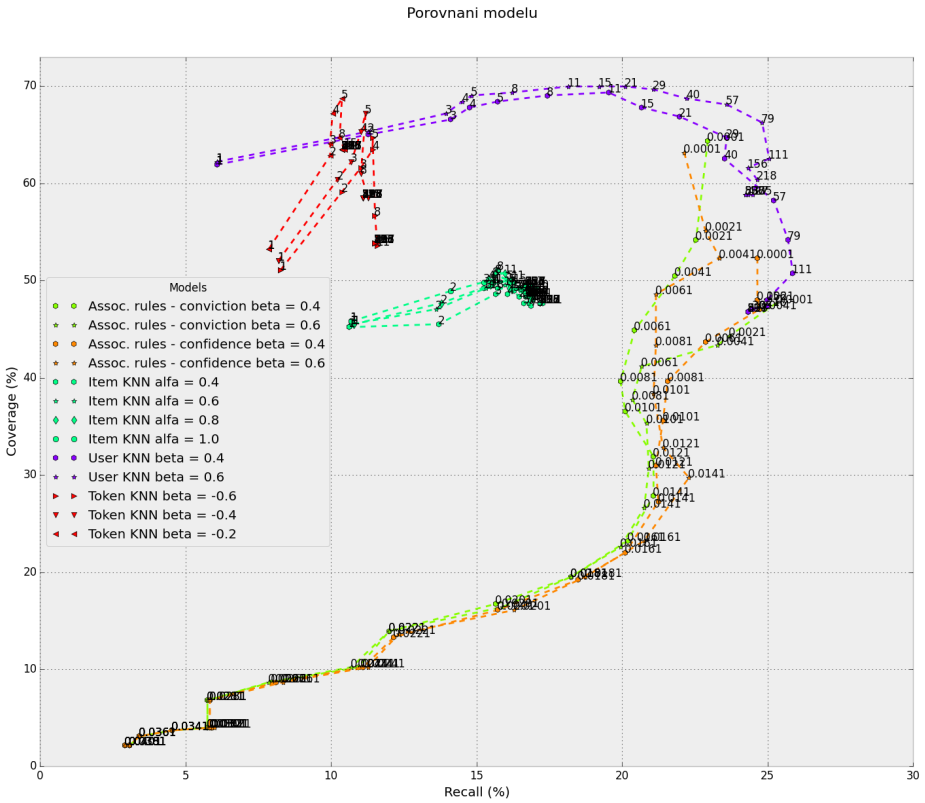


Obrázek 8.3: GUI aplikace - vytváření nového testu

8.2 Výsledky porovnávání rekomendačních modelů



Obrázek 8.4: Graf maximálních naměřených hodnot recall a coverage u měřených rekomendačních modelech



Obrázek 8.5: Graf naměřených hodnot recall a coverage u měřených rekomendačních modelech

Kapitola 9

Nasazení

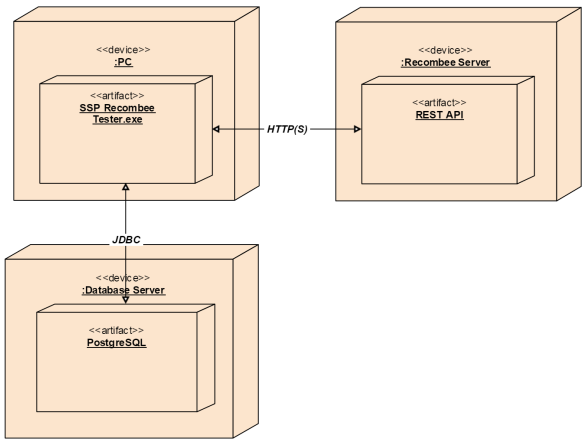
9.1 Potřebné aplikace

Java 8

Python 2.7.12

9.2 Databáze

Aplikace vyžaduje PosgreSQL databázi, tu lze naimportovat pomocí zálohovacího souboru *db-deploy.backup* uloženého na CD. Je také potřeby vytvořit přihlašovací roli, jejíž přihlašovací údaje je nutno zadat do connection stringu v konfiguračním souboru aplikace.



Obrázek 9.1: Deployment diagram

Závěr

Cílem této práce bylo seznámit se s problematikou rekomendačních systémů, provedení rešerše pro porovnávání rekomendačních algoritmů. Dále byl cílem práce návrh a implementace případů užití doporučování v portálu SSP a otestování celkového řešení. Během práce jsem se seznámil s problematikou rekomendačních systémů a provedl rešerši rekomendačních algoritmů a metod pro jejich porovnávání. Analyzoval jsem současný stav doporučování v portálu SSP. Navrhl jsem případy užití doporučování v portálu SSP pomocí rekomendačního systému. Dále jsem po dohodě s vedoucím práce implementovat aplikaci pro porovnávání rekomendačních algoritmů pomocí zvolených metrik. Aplikaci jsem otestoval a pomocí ní jsem nalezl optimální algoritmus s optimálním nastavením parametrů pro navržené případy užití doporučování v portálu SSP. Aplikaci zbývá nasadit do produkce, kde se může začít periodicky testovat na aktuálních datech portálu SSP pro udržování si optimálních hodnot parametrů rekomendačního systému.

Literatura

[1] Wikipedia: Precision and recall — Wikipedia, The Free Encyclopedia. 2016, [Online; accessed 10-June-2016]. Dostupné z: https://en.wikipedia.org/wiki/Precision_and_recall

[2] Francesco Ricci, B. S.-P. B. K., Lior Rokach (editor): *Recommender Systems Handbook*. Springer, 2010.

[3] Robert Nisbet, G. M., John Elder: *Handbook of Statistical Analysis and Data Mining Applications*. Academic Press, 2009.

[4] Mitchell, T. M.: *Machine Learning*. McGraw-Hill, 1997.

[5] Ricardo A. Baeza-Yates, B. R.-N.: *Modern Information Retrieval*. Addison-Wesley Longman Publishing Co, 1999.

[6] Shaghayegh Sahebi, W., Cohen: Community-Based Recommendations: a Solution to the Cold Start Problem. *Workshop on Recommender Systems and the Social Web (RSWEB), held in conjunction with ACM RecSys'11*, 2011. Dostupné z: http://www.dcs.warwick.ac.uk/~ssanand/RSWeb11/7Sahebi.pdf

[7] Wikipedia: Association rule learning — Wikipedia, The Free Encyclopedia. 2016, [Online; accessed 10-June-2016]. Dostupné z: https://en.wikipedia.org/wiki/Association_rule_learning

[8] Wikipedia: A/B testing — Wikipedia, The Free Encyclopedia. 2016, [Online; accessed 10-June-2016]. Dostupné z: https://en.wikipedia.org/wiki/A/B_testing

[9] Guy Shani, A. G.: Evaluating Recommendation Systems. Technická zpráva, Microsoft Research, 2009. Dostupné z: http://research.microsoft.com/pubs/115396/EvaluationMetrics.TR.pdf

[10] Wikipedia: Cross validation — Wikipedia, The Free Encyclopedia. 2016, [Online; accessed 10-June-2016]. Dostupné z: https://en.wikipedia.org/wiki/Cross-validation_(statistics)

[11] Paolo Cremonesi, R. T., Yehuda Koren: Performance of recommender algorithms on top-n recommendation tasks. *Fourth ACM Conference on Recommender Systems, ser. RecSys '10. New York, NY, USA: ACM, 2010, pp. 39–46*, 2010. Dostupné z: http://dl.acm.org/citation.cfm?id=1864721

[12] Tomas Rehorek, P. K.: Multi-Objective Evaluation in Recommender Systems. 2015.

[13] Mouzhi Ge, D. J., Carla Battenfeld: Beyond Accuracy: Evaluating Recommender Systems by Coverage and Serendipity. *RecSys '10 Proceedings of the fourth ACM conference on Recommender systems Pages 257-260*, 2010. Dostupné z: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.464.8494&rep=rep1&type=pdf

[14] Rakesh Agrawal, A. S., Tomasz Imielinski: Mining association rules between sets of items in large databases. *SIGMOD Rec., vol. 22, no. 2, pp. 207–216*, 1993. Dostupné z: http://doi.acm.org/10.1145/170036.170072

Příloha **A**

Seznam použitých zkratek

SSP Portál spolupráce studentů s průmyslem

IS Informační systém

RS Rekomendační (Doporučovací) systém

CF Collaborative filtering

kNN k-nearest neighbors

SVM Support vector machines

ANN Artificial neural network

RMSE Root-mean-square deviation - Střední kvadratická chyba

MAE Mean Absolute Error - Střední absolutní chyba

HTTP Hypertext Transfer Protocol

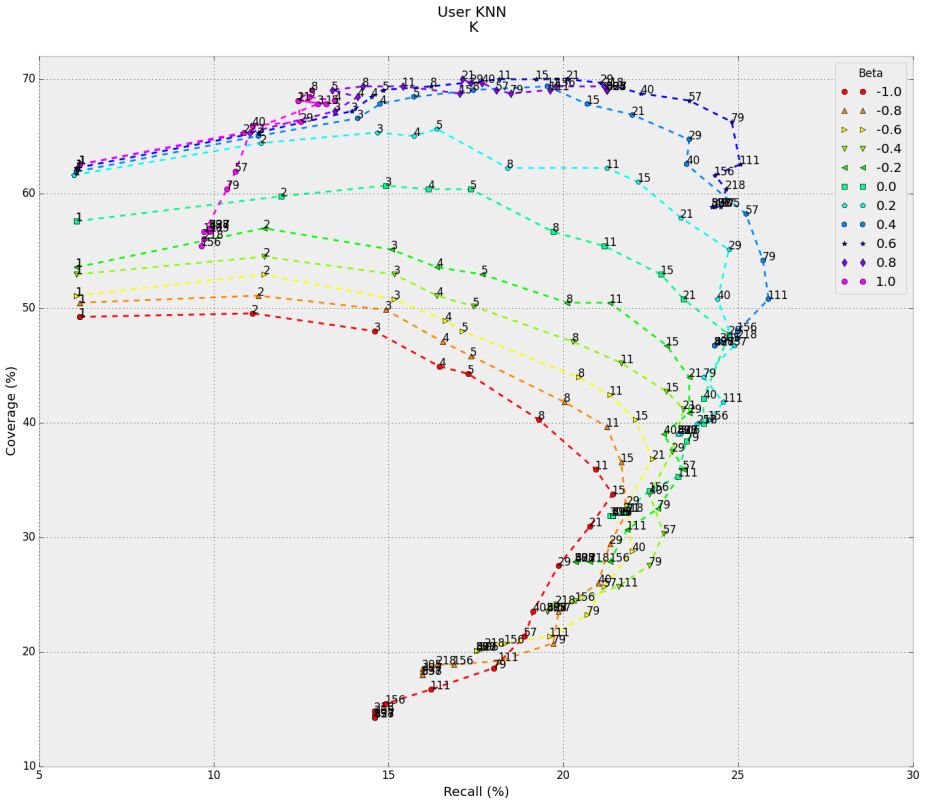
API Application Programming Interface

GUI Graphic User Interface

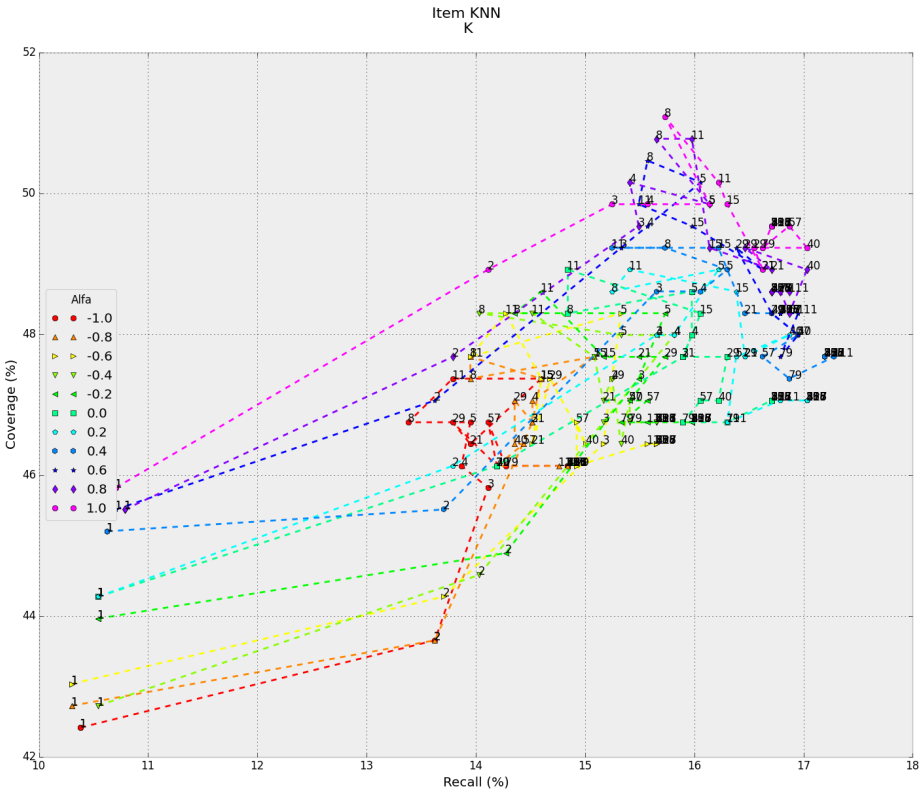
Příloha B

Naměřené testy rekomendačních modelů

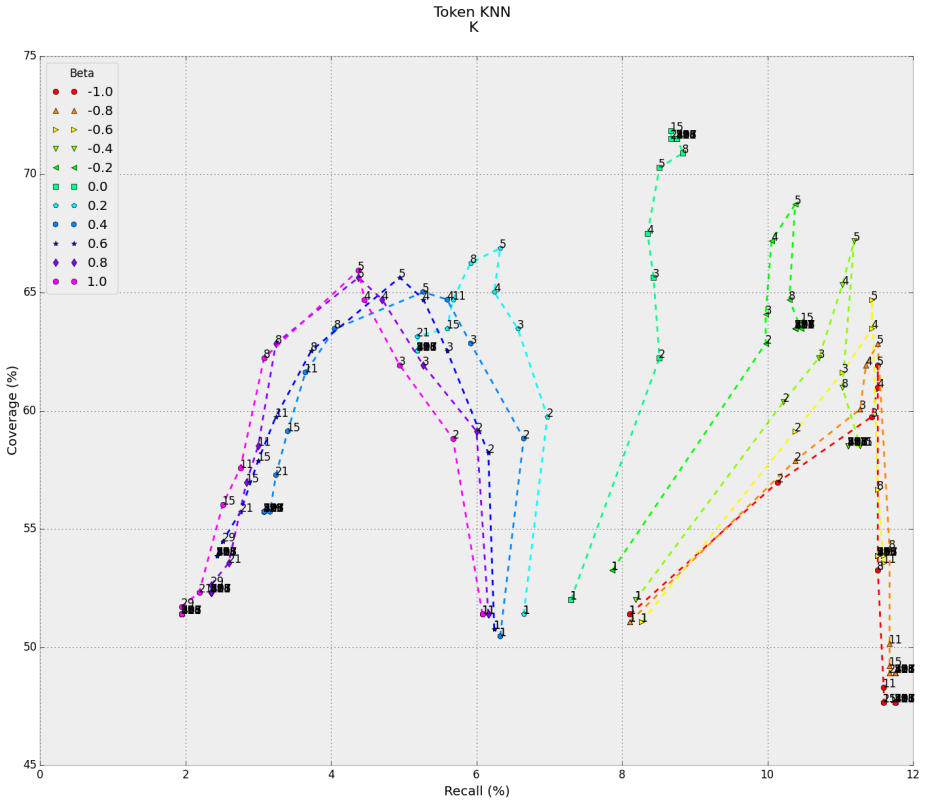
B.1 Doporučování zadání studentům



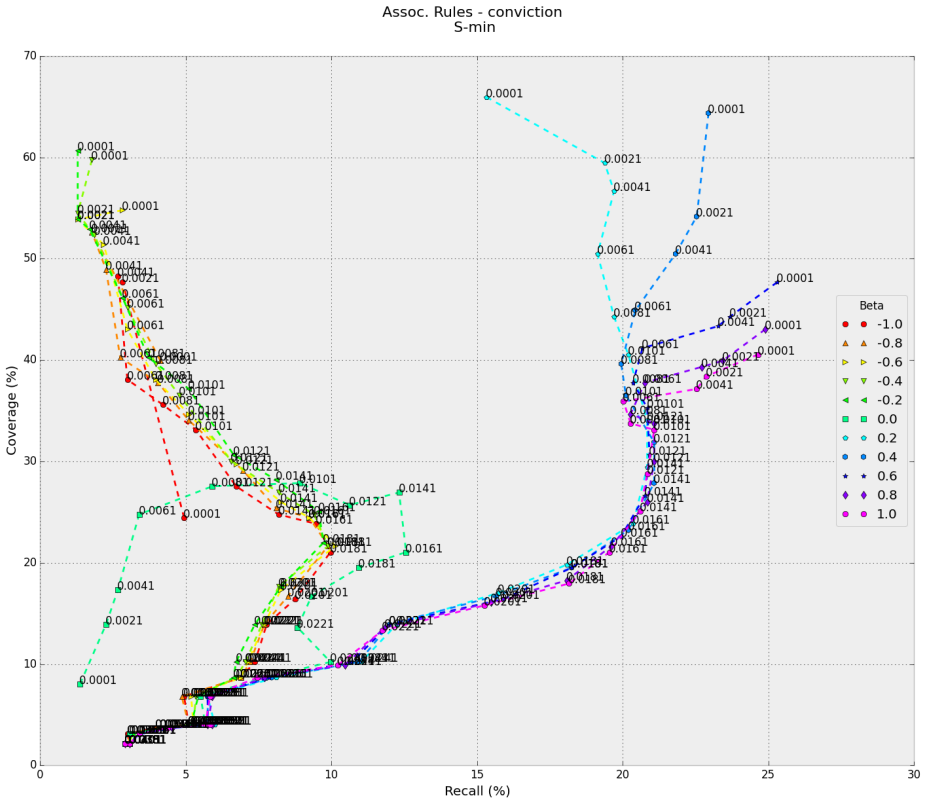
Obrázek B.1: Graf naměřených hodnot recall a coverage modelu User-kNN



Obrázek B.2: Graf naměřených hodnot recall a coverage modelu Item-kNN

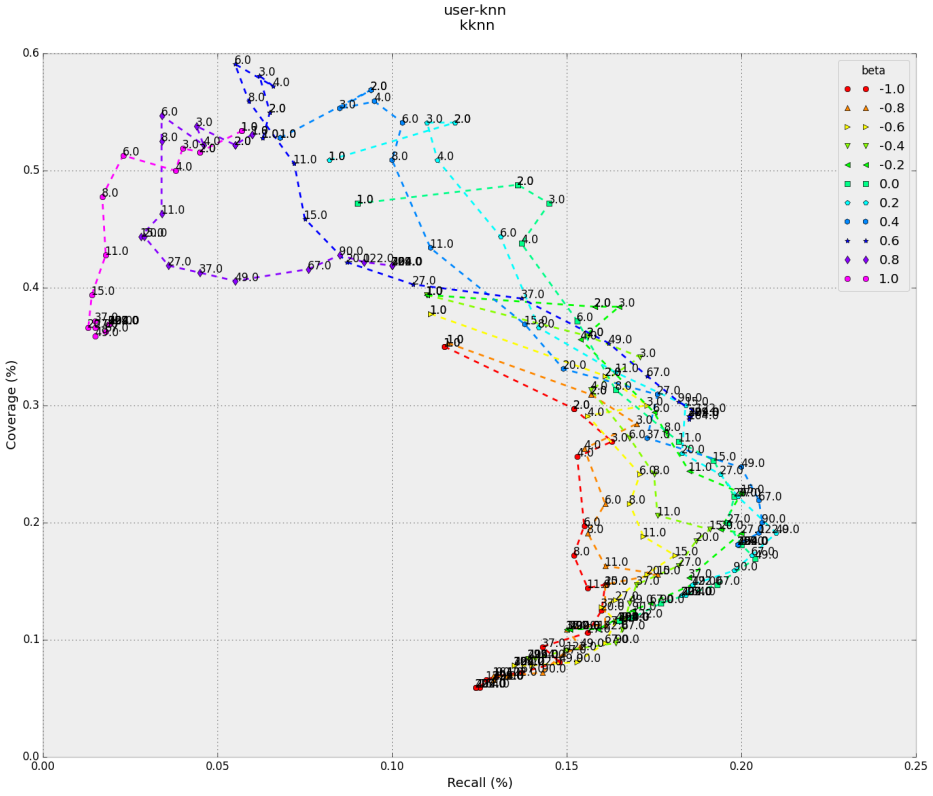


Obrázek B.3: Graf naměřených hodnot recall a coverage modelu Token-kNN

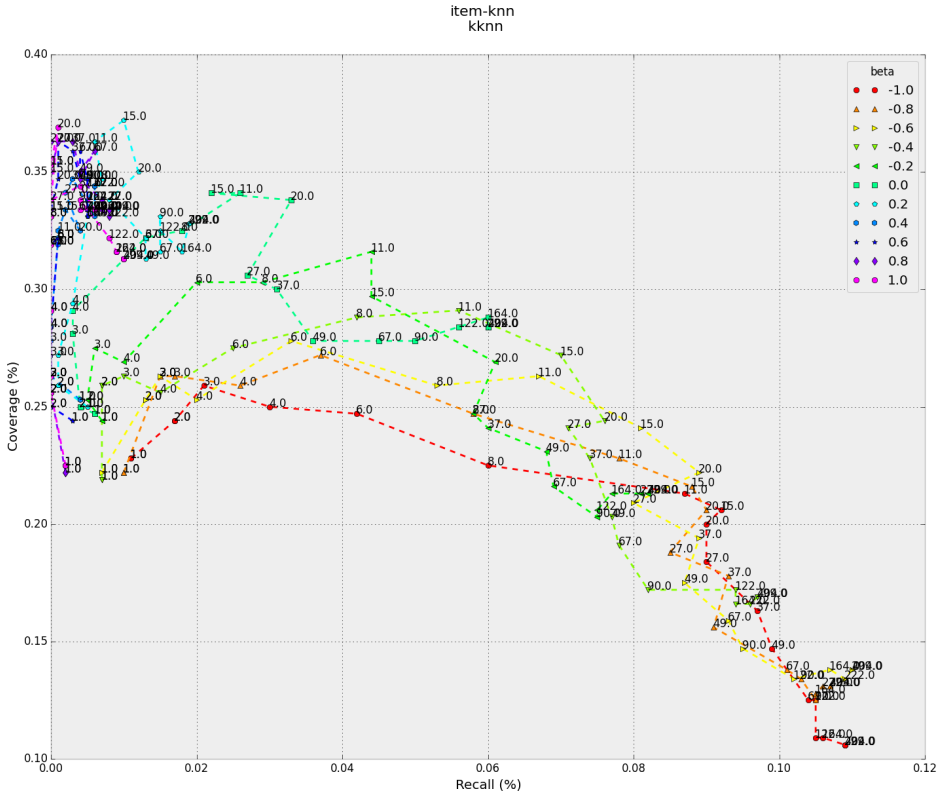


Obrázek B.4: Graf naměřených hodnot recall a coverage modelu Asociačních pravidel

B.2 Reverzní doporučování studentů k zadání



Obrázek B.5: Graf naměřených hodnot recall a coverage modelu User-kNN



Obrázek B.6: Graf naměřených hodnot recall a coverage modelu Item-kNN

Příloha C

Obsah přiloženého CD

- `readme.txt` ..... stručný popis obsahu CD
- `exe` ..... adresář se spustitelnou formou implementace
- `src`
  - `impl` ..... zdrojové kódy implementace
  - `thesis` ..... zdrojová forma práce ve formátu LaTeX
- `text` ..... text práce
  - `thesis.pdf` ..... text práce ve formátu PDF
  - `thesis.ps` ..... text práce ve formátu PS