VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
BRNO UNIVERSITY OF TECHNOLOGY

FAKULTA INFORMAČNÍCH TECHNOLOGIÍ
FACULTY OF INFORMATION TECHNOLOGY

ÚSTAV INTELIGENTNÍCH SYSTÉMŮ
DEPARTMENT OF INTELLIGENT SYSTEMS

SYSTÉM PRO DOPORUČOVÁNÍ FILMŮ

MOVIE RECOMMENDATION SYSTEM

BAKALÁŘSKÁ PRÁCE
BACHELOR'S THESIS

AUTOR PRÁCE **VÍTEK HNATOVSKYJ**
AUTHOR

VEDOUCÍ PRÁCE **doc. Ing. FRANTIŠEK ZBOŘIL, Ph.D.**
SUPERVISOR

BRNO 2023

VYSOKÉ UČENÍ TECHNICKÉ V BRNĚ
FAKULTA INFORMAČNÍCH TECHNOLOGIÍ

Zadání bakalářské práce

144995

Ústav:	Ústav inteligentních systémů (UITS)
Student:	**Hnatovskyj Vítek**
Program:	Informační technologie
Specializace:	Informační technologie
Název:	**Systém pro doporučování filmů**
Kategorie:	Umělá inteligence
Akademický rok:	2022/23

Zadání:

1. Nastudujte současné přístupy k tvorbě doporučovacích systémů pro filmy na základě předpokládaných preferencí uživatele.
2. Identifikujte slabá místa a možná rozšíření a zkvalitnění těchto systémů s použitím současných metod umělé inteligence. Dále opatřete datové sady dostatečné pro vytvoření systému tohoto typu.
3. Navrhněte systém, který současná řešení, pokud možno, překonává, nebo aspoň dosahuje srovnatelných výsledků.
4. Řešení implementujte a otestujte na vzorku uživatelů.

Literatura:

- Daneshmandmehrabani , M.: "Towards a New Algorithm for Event Recommendation System", MSc Thesis, Brock University, Ontario, 2017
- Russel, S., Norvig, P.: "Artificial Intelligence, A Modern Approach", Pearson, 2009

Podrobné závazné pokyny pro vypracování práce viz https://www.fit.vut.cz/study/theses/

Vedoucí práce:	**Zbořil František, doc. Ing., Ph.D.**
Vedoucí ústavu:	Hanáček Petr, doc. Dr. Ing.
Datum zadání:	1.11.2022
Termín pro odevzdání:	10.5.2023
Datum schválení:	21.3.2023

Abstrakt

Tato práce se zaměřuje na systém pro doporučování filmů. Nejdříve je popsána problematika doporučovacích systémů obecně a jsou popsány jednotlivé typy těchto systémů. Hlavním cílem je implementovat systém, který uživateli doporučí relevantní filmy na základě jeho preferencí. Tento systém je hybridní a skládá se ze systému založeném na obsahu a systému kolaborativního filtrování. Pro otestování systému je implementována jednoduchá aplikace, která je v této práci také popsána. Následně je systém evaluován za pomoci offline metrik a také pomocí testování na uživatelích.

Abstract

This work focuses on a movie recommendation system. First, the issue of recommendation systems in general is described, and the various types of these systems are outlined. The main goal is to implement a system that recommends relevant movies to the user based on their preferences. This system is hybrid, consisting of a content-based system and a collaborative filtering system. To test the system, a simple application is implemented, which is also described in this work. Subsequently, the system is evaluated using offline metrics and user testing.

Klíčová slova

doporučovací systém, kolaborativní filtrování, filtrování založené na obsahu, faktorizace matic, doporučování filmů

Keywords

recommendation system, collaborative filtering, content-based filtering, matrix factorization, movie recommendation

Citace

HNATOVSKYJ, Vítek. *Systém pro doporučování filmů*. Brno, 2023. Bakalářská práce. Vysoké učení technické v Brně, Fakulta informačních technologií. Vedoucí práce doc. Ing. František Zbořil, Ph.D.

Systém pro doporučování filmů

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně pod vedením pana doc. Ing. František Zbořil, Ph.D. Uvedl jsem všechny literární prameny, publikace a další zdroje, ze kterých jsem čerpal.

.....
Vítek Hnatovskyj
8. května 2023

Poděkování

Chci poděkovat panu doc. Ing. Františku Zbořilovi, Ph.D, za rady a kvalitní vedení při zpracování této práce.

Obsah

1 Úvod **4**

2 Úvod do problematiky **5**
2.1 Motivace pro tvorbu kvalitních doporučovacích systémů 5
2.2 Základní modely 7
2.2.1 Systémy kolaborativního filtrování 7
2.2.2 Doporučovací systémy založené na obsahu 8
2.2.3 Doporučovací systémy založené na znalostech 9
2.2.4 Demografické doporučovací systémy 9
2.2.5 Hybridní doporučovací systémy 9
2.2.6 Modely specifické pro určité domény 10
2.3 Problémy doporučovacích systémů 12
2.4 Evaluace doporučovacích systémů 12
2.4.1 Orientace na doporučení 13
2.4.2 Orientace na uživatele 13
2.4.3 Orientace na systém 14
2.4.4 Orientace na doručení 15
2.5 Existující systémy pro doporučování 15
2.5.1 Netlix 16
2.5.2 Amazon Prime Video 16
2.5.3 Youtube 17

3 Implementace **18**
3.1 Technologie využité k implementaci 18
3.1.1 Doporučovací systém 18
3.1.2 Webová aplikace 18
3.2 Datová sada a její zpracování 19
3.2.1 Popis datové sady 20
3.2.2 Předzpracování dat 20
3.3 Filtrování založené na obsahu 21
3.3.1 TD-IDF 21
3.3.2 Kosinová podobnost 21
3.4 Kolaborativní filtrování 22
3.5 Hybridní doporučovací systém 23
3.6 Aplikace 24
3.6.1 Návrh databáze 25
3.6.2 Koncové body (API) 25
3.6.3 Modul Recommendation 27

3.6.4 Uživatelské rozhraní 28

4 Vyhodnocení a testování **32**
4.1 Vyhodnocení 32
4.1.1 Přesnost (Precision) 32
4.1.2 Úplnost (Recall) 32
4.1.3 Míra zásahu (Hit Rate) 32
4.2 Výsledky vyhodnocení 33
4.3 Uživatelské testování 34
4.4 Výsledky uživatelského testování 34

5 Závěr **36**

Literatura **37**

A Obsah přiloženého paměťového média **42**

Seznam obrázků

2.1 Vztahy v sociálním doporučovacím systému 11
2.2 Výběr personalizovaných faktorů, které ovlivňují spokojenost uživatelů s doporučením, převzatý z [6] 16
2.3 Abstraktní přehled doporučovacího systému využívaného v Youtube, převzatý z [11] 17

3.1 Řídkost dat v tabulce `movie_metadata.csv`, tmavé čáry znamenají, že data z vybraného sloupce jsou obsažena v daném záznamu 20
3.2 ER diagram aplikace 25
3.3 Seznam koncových bodů 26
3.4 Diagram užití 28
3.5 Stránka sloužící pro přihlášení a registraci 29
3.6 Stránka obsahující populární filmy 30
3.7 Výsledné obrazovky filmů 31

4.1 Počty hodnocení pro jednotlivé stupně hodnocení 33

Kapitola 1

Úvod

S prudce rostoucím množstvím dostupných informací na internetu může být pro uživatele velmi náročné získat relevantní informace. To může vést ke zbytečně dlouhému hledání produktů či služeb, které konkrétního uživatele zajímají. Každý uživatel internetu je v něčem jiný, má jiné zájmy, záliby a koníčky. Kvůli tomu je filtrování a doporučováním obsahu na internetu náročné. Zároveň uživatelé nechtějí vyplňovat zdlouhavý formulář k vyfiltrování obsahu u daného poskytovale služeb nebo produktů. K řešení tohoto problému se využívají doporučovací systémy, které patří do skupiny filtrovacích systémů.

Doporučovací systémy se využívají například k doporučování filmů, knih, článků, služeb nebo společenských událostí. Tyto systémy filtrují položky na základě dostupných informacích o uživateli a následně mu doporučí seznam nejvíce relevantních položek.

Cílem této práce je vytvořit systém pro doporučování filmů, který je přesný, ale zároveň není výpočetně náročný. Pro výsledný systém bude následně vytvořeno jednoduché webové uživatelské rozhraní, ve kterém je možné systém otestovat.

V kapitole 2 je čtenář uveden do problematiky doporučovacích systémů, jejich rozdělení a také je seznámen s výzvami, které je nutné překonat při tvorbě robustních systémů.

V kapitole 3 je popsaná datová sada, využité technologie a implementace jak systému, tak výsledné webové aplikace.

Kapitola 4 popisuje testování doporučovacího systému. Jsou zde vysvětleny evaluační metriky, které byly použity. Také jsou v tabulkách vyobrazeny nejlepší konfigurace a jejich výsledky. Součástí testování je také uživatelské testování. V této kapitole je souhrn z testování, které proběhlo na 24 uživatelích.

Motivací této práce byla chuť poznat jak fungují doporučovací systémy v oblasti filmů a snaha o hlubší pochopení celé problematiky spojené s doporučovacími systémy.

Kapitola 2

Úvod do problematiky

Doporučovací systémy hrají velkou roli v životě mnoha lidí. Zpočátku se jednalo o oblast převážně z oboru IT, v posledních letech jsou ale využívány také poznatky například z fyziky či psychologie [30]. Entita, pro kterou se jednotlivá doporučení generují, se nazývá **uživatel**. Jednotlivé produkty, které jsou uživatelům doporučovány, se nazývají **položky**. Důležitou části těchto systémů je získání zpětné vazby od uživatele. Ta může být buď **explicitní**, ve formě ohodnocení produktu na číselné škále, nebo **implicitní**, například ve formě informace o zhlédnutí produktu. Bez zpětné vazby by tyto systémy v podstatě nemohly fungovat, jelikož by bylo náročné získat uživatelské preference, na základě kterých se generují doporučení.

Problém doporučovacích systémů může být formulován dvěma způsoby [3]:

- **Verze problému s ohodnocením** – Jedná se o předpovězení hodnocení pro kombinaci uživatele a položky. Tedy pro každou položku z datové sady přiřadíme ohodnocení relevantnosti k danému uživateli. Tento problém se také označuje jako problém **doplnění matice**, protože je dostupná neúplná matice obsahující uživatele a jejich hodnocení jednotlivých položek, následně jsou hodnocení pro doposud nehodnocené položky předpovězeny pomocí různých metod učení algoritmu.
- **Verze problému s pořadím** – Není potřeba předpovědět konkrétní hodnocení pro kombinaci uživatele a položek, ale stačí pouze seznam N položek, které jsou pro uživatele nejvíce relevantní. Lze ale využít první způsob a z něho následně vytvořit tento seznam. Určení nejrelevantnějších N položek pro uživatel je častější než určení nejrelevatnějších K uživatelů pro položku, ačkoli obě metody jsou zcela analogické.

2.1 Motivace pro tvorbu kvalitních doporučovacích systémů

Jak je popsáno v [44], **doporučovací systémy** jsou softwarové nástroje a techniky, které uživatelům poskytují návrhy položek, které by pro ně mohly být užitečné. Doporučovací systémy hrají různé role pro poskytovatele služeb a uživatele.

Poskytovatelé služeb mohou zavést doporučovací systém k dosažení následujících cílů [44]:

- **Zvýšení počtu prodaných položek** – Komerční doporučovací systémy podporují prodej tím, že poskytují personalizovaná doporučení přizpůsobená preferencím uživatelů. Navrhováním položek, které se úzce shodují se zájmy uživatelů, zvyšují doporučovací systémy pravděpodobnost nákupu, a tím zvyšují celkový objem prodeje.

- **Prodej více různých položek** – Doporučovací systémy mohou uživatelům představit méně známé nebo specializované položky, které by jinak neobjevili. Propagací širšího sortimentu položek pomáhají doporučovací systémy poskytovatelům služeb udržovat rozmanité zásoby a zabraňují přílišnému zaměření pouze na nejoblíbenější produkty.
- **Zvýšení spokojenosti uživatelů** – Dobře navržený doporučovací systém zvyšuje uživatelský komfort tím, že poskytuje relevantní a zajímavá doporučení. Díky intuitivnímu a uživatelsky přívětivému rozhraní mohou doporučovací systémy zlepšit celkové vnímání platformy uživateli, což povede k jejímu většímu využívání a vyšší pravděpodobnosti, že budou spokojeni s doporučeními.
- **Zvýšení loajality uživatelů** – Doporučovací systémy budují loajalitu tím, že se učí z minulých interakcí uživatelů a v průběhu času vylepšují doporučení. Jak uživatelé pokračují ve spolupráci s platformou, doporučovací systém zlepšuje své chápání jejich preferencí, což vede k přesnějším a relevantnějším doporučením, která následně povzbuzují uživatele, aby službu nadále využívali.
- **Lepší porozumění preferencím uživatelů** – Doporučovací systémy shromažďují a předpovídají preference uživatelů, čímž pomáhají poskytovatelům služeb získat přehled o jejich zákaznické základně. Díky pochopení preferencí uživatelů mohou poskytovatelé služeb přijímat rozhodnutí založená na datech o řízení zásob, výrobě, marketingových strategiích a dalších oblastech a optimalizovat tak svou nabídku.

Doporučovací systémy nabízí uživatelům následující služby [44]:

- **Doporučování některých relevantních položek** – Doporučovací systémy doporučují seřazený seznam položek, které se uživatelům pravděpodobně budou líbit na základě jejich preferencí. Tato doporučení mohou uživatelům pomoci objevit nové produkty, služby nebo obsah, které odpovídají jejich zájmům.
- **Doporučování všech relevantních položek** – V kritických aplikacích, jako jsou lékařské nebo finanční oblasti, je cílem doporučovacích systémů doporučit všechny položky, které by mohly uspokojit potřeby uživatele. V těchto případech mohou mít uživatelé prospěch z komplexního seznamu možností seřazených podle relevance nebo kvality spolu s dalšími vysvětleními poskytnutými doporučovacím systémem.
- **Anotace v kontextu** – Doporučovací systémy zvýrazňují položky, které jsou v daném kontextu zajímavé, například zdůrazňují doporučené televizní pořady v elektronickém programovém průvodci. Tato funkce umožňuje uživatelům při procházení širšího seznamu možností rychle identifikovat položky, které nejvíce odpovídají jejich preferencím.
- **Doporučování balíčku** – Doporučovací systémy mohou doporučit skupiny položek, které se vzájemně doplňují, například cestovní plán složený z atrakcí, destinací a ubytování. Uživatelé mohou tyto balíčky zvážit a vybrat jako jeden balíček, což jim zjednoduší rozhodovací proces.

Souhrnně lze říci, že doporučovací systémy plní v informačních systémech různé role a řeší potřeby poskytovatelů i uživatelů služeb. Využívají různé zdroje dat a techniky k vytváření personalizovaných doporučení, což v konečném důsledku zvyšuje uživatelský žážitek a spokojenost.

2.2 Základní modely

Základní modely pro doporučovací systémy pracují převážně se dvěma druhy dat.

- **Data obsahující vztahy mezi položkami a uživateli** – Vztah mezi položkou a uživatelem většinou představuje explicitní hodnocení položky, které udělil uživatel. Může mít také formát implicitního hodnocení, například zda si uživatel zobrazil položku, nebo jak dlouho si položku prohlížel.
- **Data obsahující informace o uživateli a položkách** – Jedná se o metadata pro položky a uživatele, tedy například název nebo popis položek, případně věk nebo pohlaví uživatele.

2.2.1 Systémy kolaborativního filtrování

Systémy kolaborativního filtrování využívají tzv. **řídké matice** obsahující ohodnocení položek jednotlivými uživateli [45]. Obecně tyto modely vychází z předpokladu, že pokud mají uživatel A a B stejné či podobné preference na N položek, je více pravděpodobné, že uživatel A bude mít podobnější preference u položky α jako uživatel B, než náhodně vybraný uživatel. V 2.1 lze vidět příklad předpovídání hodnocení pro uživatele B. Uživatelé A a B mají podobné(kladné) hodnocení pro položky $Movie1$ a $Moive2$, zde by tedy model předpokládal, že uživatel B ohodnotí položku $Movie3$ také kladně.

	Movie 1	**Movie 2**	**Movie 3**	**Movie 4**
User A	4.0	3.0	5.0	-
User B	3.5	4.0	-	3.5
User C	-	3.5	4.0	4.5

Tabulka 2.1: Tabulka obsahující příklad collaborativního filtrování se 3 uživateli a 4 položkami

Tyto modely se dále dělí na:

Metody založené na paměti

Jedná se o jedny z prvních metod. Jsou jednoduché na implementaci, ale trpí **problémem škálovatelnosti**. Tyto modely uchovávají v paměti celé matice obsahující uživatele, položky a jejich hodnocení, což může být při velkém množství dat problém. Na tyto modely se dá nahlížet následovně [48] [52]:

- **Kolaborativní filtrování založené na uživateli** – Výsledné hodnocení uživatele A pro položku X bude podobné jako hodnocení uživatelů podobných uživateli A.
- **Kolaborativní filtrování založené na položce** – Výsledné hodnocení uživatele A pro položku X se dá předpovídat na základě hodnocení uživatele A pro položky podobné k X.

Metody založené na modelu

U těchto metod není nutné mít v paměti matici ohodnocení uživatelů a položek. Místo toho se k předpovědi a výpočtu toho, jak uživatel hodnotí jednotlivé položky, využívají modely

strojového učení, které předpovídají hodnocení položek doposud neviděných uživatelem. Tyto metody se dále dělí na [52]:

- **Faktorizace matic** – Tato technika rozkládá matici interakce mezi uživatelem a položkou na matice nižších rozměrů, které zachycují **latentní faktory** (latentní faktory jsou popsány v sekci 3.4) vysvětlující preference uživatelů a charakteristiky položek [25]. Mezi populární techniky faktorizace matic patří **Alternating Least Squares(ALS)** a **Singular Value Decomposition (SVD)**. Technika SVD je více popsána v sekci 3.4
- **Shlukování** – Shlukovací algoritmy, jako je **K-means** [42] nebo hierarchické shlukování, seskupují podobné uživatele nebo položky. Doporučení pak lze generovat tak, že se navrhují položky ze stejného shluku jako položky, se kterými uživatel interagoval.
- **Hluboké učení** – Modely hlubokého učení, jako jsou **neuronové sítě** nebo **rekurentní neuronové sítě**, lze použít k zachycení složitých vzorců v datech a generování doporučení. Tyto modely si poradí s rozsáhlými soubory dat a v některých případech poskytují přesnější doporučení.

2.2.2 Doporučovací systémy založené na obsahu

Doporučovací systémy založené na obsahu vytvářejí personalizovaná doporučení analýzou vnitřních vlastností nebo atributů položek a identifikací podobností mezi nimi. Hlavní myšlenkou filtrování založeného na obsahu je doporučovat položky, které jsou podobné těm, s nimiž uživatel dříve interagoval nebo o něž projevil zájem. Tento přístup je obzvláště účinný, pokud mají uživatelé specifické preference nebo pokud jsou k dispozici jen omezené údaje o interakcích.

Při implementanci takového systému se obvykle postupuje dle následujících kroků [37]:

- **Extrakce příznaků** – Extrakce relevantních příznaků nebo atributů z položek. Například v systému doporučování filmů mohou vlastnosti zahrnovat žánr, režiséra, herce nebo klíčová slova z popisu děje.
- **Vytvoření profilu uživatele** – Vytvoření profilu pro každého uživatele na základě jeho historie interakcí, například položek, které se mu líbily, které hodnotil nebo které si prohlížel. Profil uživatele shrnuje preference uživatele z hlediska vlastností položek.
- **Filtrace položek** – Vypočítá se podobnost mezi položkami a profily uživatelů pomocí vhodných metrik podobnosti, jako je **Kosinová podobnost**, **Jaccardova podobnost** nebo **Euklidova vzdálenost**. Položky jsou seřazeny podle podobnosti a poté je vybrán určitý počet nejlepších položek, které jsou uživateli doporučeny.

Doporučovací systémy založené na obsahu mají několik výhod, například jsou schopny poskytovat doporučení pro nové položky bez interakce s uživatelem (částečně řeší problém **studeného startu**, který bude popsán v sekci 2.3) a jsou odolnější vůči problémům, jako je zkreslení popularity položky. Mohou však trpět přílišnou specializací, to znamená doporučovat položky, které jsou příliš podobné předchozím preferencím uživatele a potenciálně omezovat objevování nového obsahu. Kromě toho může být vytvoření přesných uživatelských profilů a extrakce smysluplných rysů náročné, zejména u složitých položek nebo při práci s nestrukturovanými daty, jako je text nebo obrázky.

2.2.3 Doporučovací systémy založené na znalostech

Fungují na principu **interaktivního dotazování** uživatele a na základě zodpovězených otázek vytvářejí profil uživatele. Tyto systémy se snaží získat co nejvíce informací o uživateli a jeho preferencích, a na základě těchto informací generují doporučení. Využívá se k řešení problému studeného startu. Je nutná znalost domény, to znamená schopnost systému porozumět specifickým aspektům daného oboru, například typům produktů, kategoriím, specifickým charakteristikám atd [8].

2.2.4 Demografické doporučovací systémy

Fungují tak, že sbírají informace o uživateli, jako je jeho věk, pohlaví, geografická poloha a další demografické údaje. Tyto informace se používají k vytvoření profilu uživatele a k poskytnutí doporučení, která jsou založena na trendech a preferencích dané demografické skupiny. Tyto systémy se mohou opírat o velké množství anonymních údajů o uživatelích a jejich preferencích, které jsou shromažďovány a analyzovány, aby se stanovily obecné trendy v dané demografické skupině [43].

2.2.5 Hybridní doporučovací systémy

Hybridní doporučovací systémy jsou systémy, které kombinují různé zdroje informací a různé přístupy, aby dosáhly lepšího výsledku než jednotlivé systémy pouze založené na obsahu, demografii, nebo historii uživatele. Spojují silné stránky více doporučovacích technik, čímž zmírňují omezení spojená s každou jednotlivou metodou. Tyto systémy kombinují výhody jednotlivých systémů a využívají jejich silné stránky, aby poskytly uživateli co nejrelevantnější doporučení.

Existuje několik přístupů [9] ke tvorbě hybridních doporučovacích systému, zde je seznam nejvíce využívaných:

Váhované

Tyto systémy obvykle využívají kolaborativní filtrování společně s alternativními technikami. Výstupu každého systému přidělí váhu a následně z nich vypočítá výsledné předpovídané hodnocení. Navzdory přímočaré integraci různých technik a následně snadnému přidělování vah jsou vážené hybridy založeny na předpokladu, že relativní hodnoty technik zůstávají konzistentní u všech položek - což se ukázalo jako nepřesné ve scénářích, kdy kolaborativní doporučovací systémy selhávají kvůli nedostatečnému počtu hodnotitelů [9].

Přepínací

Používají kritérium pro střídání doporučovacích technik. Přepínací hybridy, jako je například ten, který navrhli autoří této práce [54], umožňuje doporučení napříč žánry tím, že využíva silné stránky jednotlivých doporučovacích metod, a poskytuje tak relevantní návrhy, i když nejsou sémanticky podobné dříve hodnoceným položkám. Navzdory dodatečné složitosti a parametrizaci, které jsou nutné kvůli kritériím přepínání, poskytují tyto hybridy výhodu větší citlivosti na silné a slabé stránky jednotlivých doporučovacích technik.

Smíšené

Smíšené hybridy prezentují doporučení z více technik současně. Tato metoda účinně řeší problém nové položky, protože komponenty založené na obsahu mohou doporučovat nové pořady na základě jejich popisu i bez předchozího hodnocení. Neřeší však zcela problém nového uživatele, protože jak obsahové, tak kolaborativní metody vyžadují k efektivnímu fungování určité údaje o preferencích uživatele [9].

Kombinující příznaky

Integrují kolaborativní informace jako další příznaková data do stávajícího souboru dat a využívají techniky založené na obsahu nad rozšířenými daty. Tento typ systémů dokáže zohlednit kolaborativní data bez úplné závislosti, čímž se snižuje jeho citlivost na počet uživatelů, kteří danou položku hodnotili. Kromě toho tento přístup umožňuje systému posoudit vnitřní podobnost položek, která by mohla být čistě kolaborativnímu systému nedostupná [55].

Kaskádové

Využívají postupný proces, kdy jedna doporučovací technika nejprve vytvoří hrubé pořadí kandidátů, po němž následuje druhá technika zpřesňující doporučení ze souboru kandidátů. Kaskádové hybridy nabízejí zvýšenou efektivitu tím, že druhou, méně prioritní techniku použijí pouze tehdy, když je požadováno další rozlišování, čímž se vyhnou nutnosti zpracovávat všechny položky současně, jako je tomu u vážených hybridů [26].

2.2.6 Modely specifické pro určité domény

Mimo základní modely existují také modely specifické pro určité domény, kterým se také říká kontextuální modely:

Doporučovací systémy citlivé na čas

V mnoha případech má na doporučování velký vliv čas. Je pravděpodobné, že se preference uživatele budou časem měnit. Z toho důvodu je tedy vhodné sledovat jak jeho krátkodobé preference, tak ty dlouhodobé. Oblíbenost může ovlivnit také aktuální roční období, nebo přímo hodina, kdy byla položka doporučena. Například je více pravděpodobné, že na podzim bude uživatel vyhledávat zimní bundy, než plavky. Čas lze zakomponovat do systému jako explicitní proměnnou, případně se může jednat pouze o speciální typ kontextuálního modelu, který bere čas v potaz. Navrhnout takové systémy je ale náročně, jelikož data v praxi jsou velice řídká a je tedy nutná velká datová sada.

Doporučovací systémy založené na poloze

Stále více zařízení má dostupnou GPS a má aktivní zjišťování polohy. Doporučování míst musí mít nějakým způsobem zakomponovanou informaci o poloze. Systém sice může doporučit například událost, kterou by uživatel za normálních podmínek navštívil, pokud ale bude na druhé straně planety, s velikou pravděpodobností ji uživatel nenavštíví. Tyto systémy lze rozdělit na dva druhy [57]:

- **Systémy pro doporučování samostaných lokací** – Slouží k doporučení například restaurace nebo obchodu.

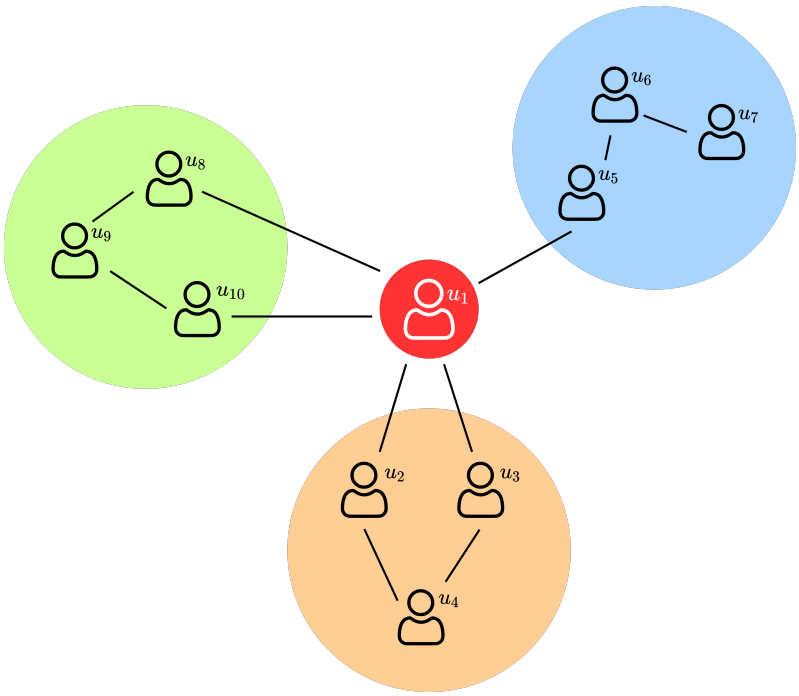
- **Systémy pro doporučování sekvenčních lokací** – Doporučuje například různé turistické stezky.

Sociální doporučovací systémy

Jedná se o systémy, které mají na svém vstupu informace o sociálních vztazích mezi uživateli. Zkoumají se tedy vazby mezi jednotlivými uživateli a na základě nich se ovlivní doporučení položek. Vztahy v systému mohou mít 2 formy:

- **Explicitni** – Dva uživatelé si například potvrdili přátelství, případně sledování.
- **Implicitní** – Emailové sítě, sítě spolupracovníků.

Velká většina systémů se zaměřuje na explicitní vztahy. Uživatelé na internetu jsou ze své podstaty korelovaní [32]. Je přirozené, že se uživatel zeptá někoho ze svých přátel, než si koupí některou z položek. Dá se předpokládat, že uživatelé, kteří mají mezi sebou nějaký sociální vztah, budou mít více podobné preference, než dva náhodně vybraní uživatelé. Právě tuto skutečnost využívají tyto systémy. Je ale nutné modelovat také vzájemnou důvěru dvou přátel. Fakt, že jsou přátelé neimplikuje, že mají stejné zájmy. Pomocí důvěry lze přidat váhu jednotlivým doporučením a tím zvýšit úspěšnost výsledného systému. Jednoduchý příklad vztahů v sociálním doporučovacím systému lze vidět na obrázku 2.1. Každý kruh přátel značí uživatele s určitými filmovými preferencemi. Zelený kruh označuje uživatele, kteří preferují komedii, oranžový označuje preferenci dokumentárních filmů a modrý preferenci akčních filmů. Pokud u_1 uživatel zrovna vyhledává v filmy v sekci akčních filmů, je vhodné zahrnout do doporučení také filmy, které patří do tohoto žánru a kladně jej ohodnotil uživatel u_5, případně lze vzít v potaz hodnocení všech uživatelů v modrém kruhu.



Obrázek 2.1: Vztahy v sociálním doporučovacím systému

2.3 Problémy doporučovacích systémů

Implementace doporučovacích systémů přináší několik velkých výzev, na které je nutno myslet, aby byl implementovaný systém robustní a přesný:

- **Problém studeného startu (Cold-start problem)** – Tento problém je spjatý s řídkostí dat. Dochází k němu v případě, kdy systém nemá dostatek informací o uživateli nebo položce. Nastává většinou při jejich vstupu do systému, od toho je název `Cold-start`. Nastává ve třech situacích [30]:
  - **Nedostatek údajů o uživateli**
  - **Nedostatek údajů o položce**
  - **Kombinace předchozích dvou**

  Jedním z možných řešení je využít Hybridní doporučovací systém, nebo požádat uživatele o zodpovězení pár otázek, díky kterým dá systému o sobě dost informací.

- **Škálovatelnost (Scalability)** – V praxi mohou datové sady obsahující uživatele a položky růst velmi rychle. V případě zvolení méně efektivních algoritmů nebo při neefektivní práci s daty může být výsledný systém v praxi nepoužitelný. Tento problém lze řešit pomocí využití efektivních algoritmů, případě lze také využít metody pro redukci dat, například metody redukce dimenzí dat. K redukci dimenzí lze využít metodu **Singular Value Decomposition (SVD)** [16].

- **Rozmanitost vs přesnost (Diversity vs. accuracy)** – Při doporučování lze zvolit metodu doporučování populárních položek nebo zvolit popularitu jako velký faktor při doporučování. Tato metoda však není pro uživatele moc přínosná, jelikož populární položky si může uživatel sám jednoduše najít bez doporučovacího systému. Je tedy vhodné do doporučovaných položek přidávat také méně populární položky, aby byly doporučované položky více různorodé [35]. K řešení problému lze využít **Bounded greedy algorithm** [50].

- **Zranitelnost vůči útokům (Vulnerability to attacks)** – Utočníci mohou napadnout systém za účelem zvýšení pravděpodobnosti doporučení jimi vybraných položek [36]. Tento problém lze řešit různými způsoby, například zablokování vstupu informací od útočníků do systému, nebo pomocí sofistikovaných doporučovacích technik. Jedná se o složitý úkol, jelikož útočníci se stále vyvíjí a je tedy nutné se přizpůsobit a vyvíjet stále nové techniky.

- **Synonymy** [23] – Jedná se o tendenci, kdy dvě velmi podobné věci mají jiné jméno. Většina doporučovacích systémů obtížně rozlišuje mezi úzce souvisejícími položkami. Tento problém se vyskytuje u systému kolaborativního filtrování. Lze ho řešit pomocí již zmíněné metody SVD nebo také pomocí **Latent Semantic Indexing (LSI)** [46].

2.4 Evaluace doporučovacích systémů

V [4] autoři představují komplexní rámec pro hodnocení různých aspektů doporučovacích systémů. Hodnotící rámec se skládá z několika dimenzí, které jsou rozděleny do čtyř skupin: **orientace na doporučení**, **orientace na uživatele**, **orientace na systém** a **orientace**

na doručení. Autoři se zabývají vztahy mezi těmito dimenzemi a poskytují praktické pokyny pro využití hodnotícího rámce při ověřování jednotlivých doporučovacích systémů. Využitím tohoto vícerozměrného přístupu mohou výzkumní pracovníci a odborníci z praxe získat lepší pochopení výkonnosti doporučovacího systému v konkrétní doméně. Tato komplexní metoda hodnocení umožňuje vývojářům identifikovat silné a slabé stránky různých doporučovacích systémů, což jim v konečném důsledku umožňuje činit informovaná rozhodnutí o nejvhodnějším systému pro jejich potřeby.

Popis jednotlivých dimenzí a využívaných metrik je následující:

2.4.1 Orientace na doporučení

Správnost doporučení

Pro vyhodnocení předpovídaného hodnocení uživatelů se běžně používají metriky **root-mean-squared-error** (RMSE) [25] a **mean absolute error** (MAE) [33]. Při vyhodnocování uspořádaného seznamu doporučení uživatelům se používají míry pořadí, jako je normalized distance-based performance measure (NDPM) [56], Spearmanova korelace pořadí a Kendallova korelace pořadí. Když systém doporučuje zajímavé položky, často se používají klasifikační metriky, jako je **přesnost**, **úplnost** a **míra zásahu**. Uvádí se také **F-measure**, která je harmonickým průměrem přesnosti a úplnosti.

Pokrytí

V doporučovacích systémech je pokrytí kritickým aspektem, který se týká podílu dostupných informací, včetně položek a uživatelů, pro které lze generovat doporučení. Zjednodušeně se dá říct, že pokrytí označuje schopnost systému doporučovat všechny dostupné položky. Dva základní typy pokrytí jsou **pokrytí katalogu**, které označuje podíl dostupných položek doporučených uživatelům, a **pokrytí predikcí**, které označuje podíl uživatelů nebo interakcí s uživateli, pro které může systém vytvářet predikce [21].

Rozmanitost

Zásadní roli hraje také rozmanitost, která zvyšuje uživatelský komfort tím, že nabízí komplexnější nabídku možností. Doporučení, která jsou si příliš podobná, nemusí být pro uživatele cenná, protože se mohou jevit jako nadbytečná a prodlužovat čas, který uživatelé stráví zkoumáním prostoru položek. Rozmanitost lze považovat za opak podobnosti a lze ji definovat jako průměrnou nepodobnost mezi všemi dvojicemi položek v souboru. Protože zlepšení rozmanitosti může vyžadovat obětování podobnosti, optimální strategií je vyvážit kompromis mezi podobností a rozmanitostí. Kvalitní metrika kombinuje diverzitu i podobnost, aby se dosáhlo této rovnováhy. Byly zavedeny různé metody, například **Q-statistics** a **difficulty measures**, které mají personalizovat doporučení a zvýšit rozmanitost vrácených položek [10].

2.4.2 Orientace na uživatele

Důvěryhodnost

Uživatelé očekávají, že dostanou spolehlivé a užitečné návrhy. Vnímaná užitečnost silně koreluje s důvěryhodnými doporučeními. Systém, který soustavně poskytuje nesprávná doporučení, ztratí důvěru uživatelů, což povede k ignorování doporučení a snížení celkové

užitečnosti systému. V některých případech si uživatelé budují důvěru tím, že vidí doporučované známé položky. Poskytování vysvětlení k doporučením může uživatelům rovněž pomoci vybudovat si důvěru v systém [7].

Měření důvěry lze provádět prostřednictvím uživatelských studií, kdy jsou uživatelů dotazováni, zda považují doporučení za rozumná. Dalším přístupem k měření důvěry je analýza četnosti, s jakou uživatelé doporučení využívají. Toho lze dosáhnout sledováním interakcí uživatelů se systémem.

Novost

Novost v doporučovacích systémech se týká doporučení, o kterých uživatelé dříve nevěděli. Úzce souvisí s emocionální reakcí uživatelů na doporučení, což z ní činí náročnou dimenzi pro měření. Jeden z přístupů k vytvoření systému, který doporučuje novinky spočívá v odstranění položek, které uživatelé již hodnotili nebo používali, ze seznamu doporučení. Pokud jsou tyto informace k dispozici, lze novost doporučování měřit porovnáním doporučení s předchozími volbami nebo hodnoceními uživatelů. To vyžaduje udržování profilů uživatelů pro sledování interakcí s položkami [21].

Jiný přístup k měření novosti zahrnuje počítání počtu doporučených populárních položek. Tato metrika předpokládá, že vysoce hodnocené a oblíbené položky jsou uživatelům pravděpodobně známé, a proto nejsou nové.

Serendipita

Serendipita v doporučovacích systémech označuje neočekávaná a náhodná doporučení, která přinášejí užitek uživatelům. Ačkoli souvisí s novostí, serendipita se liší v tom, že u doporučovaných položek vyžaduje prvek správnosti, který brání tomu, aby náhodná doporučení byla serendipitní. Nová doporučení mohou, ale nemusí být serendipitní. Pokud budou nová doporučení postrádat užitečnost, nebudou považovány za serendipitní, ale spíše za chybné a rušivé. Vyvážení správnosti a serendipity je nezbytné.

Pro dosažení serendipity je třeba se vyhnout podobným doporučením, protože jejich očekávaný výskyt není pro uživatele zpravidla přínosný. Profily uživatelů nebo označení podobných položek mohou pomoci taková doporučení odfiltrovat [35]. Definice podobnosti by však měla záviset na kontextu, v němž je doporučující nástroj používán.

2.4.3 Orientace na systém

Robustnost

Robustnost v doporučovacích systémech se týká schopnosti tolerovat nepravdivé nebo chybné informace poskytnuté uživateli, ať už záměrně nebo náhodně. Takové chyby mohou zahrnovat nesprávné hodnocení položek, chyby ve specifikaci profilu uživatele a použití doporučujícího systému v nesprávném kontextu nebo pro nesprávné úkoly.

Aby bylo možné vyhodnotit odolnost systému vůči útokům, porovnávali výzkumníci hodnocení predikce před a po poskytnutí nepravdivých informací a analyzovali posun predikce, který odráží, jak se predikce následně změnila. Posun predikce položky a její průměr lze vypočítat pomocí různých vzorců, některé z nich jsou zmíněny v tomto článku [39].

Velký posun však nemusí mít vždy vliv na výkonnost systému, pokud falešná informace nezmění položky doporučené uživatelům. V takových případech může být skutečné hod-

nocení konkrétních položek natolik nízké, že je chyby stejně nedokážou posunout na první místo doporučených položek.

Škálovatelnost

Pokud systém nedokáže zpracovat velké množství dat, mohou být ohroženy další dimenze, jako je pokrytí a správnost. Některé doporučovací systémy dobře pracují s malými soubory dat, ale mají problémy s velkými soubory položek nebo počtem uživatelů.

Problém škálovatelnosti lze rozdělit na dvě části:

- **Doba trénování doporučovacího algoritmu**
- **Propustnost systému při práci s velkým množstvím položek**

Pokud není system škálovatelný, může to negativně ovlivnit použitelnost doporučovacího systému, protože doba odezvy může být příliš velká na to, aby byla pro uživatele efektivní [4].

Stabilita

Stabilita se týká konzistence předpovědí doporučovacího systému v průběhu času za předpokladu, že nová hodnocení nebo položky přidané během tohoto období odpovídají stávajícím hodnocením v systému. Stabilní doporučující systém může zvýšit důvěru uživatelů tím, že poskytuje konzistentní předpovědi. Časté změny a výkyvy v předpovědích mohou u uživatelů vyvolat zmatek a následně vést k nedůvěře v systém.

Stabilitu lze měřit porovnáním předpovědi v určitém časovém okamžiku s okamžikem, kdy jsou přidána nová hodnocení. Autoři [1] hodnotili stabilitu tak, že trénovali doporučovací algoritmus s existujícími hodnoceními a provedli počáteční předpověď. Po přidání nových hodnocení v následujícím období je algoritmus znovu natrénován s novým souborem dat a provede druhou předpověď.

2.4.4 Orientace na doručení

Použitelnost

Pro účinnost doporučovacích systémů je zásadní jejich použitelnost, protože se musí snadno používat a dodržovat obecné zásady použitelnosti. Měly by být efektivní, účinné a poskytovat určitou míru spokojenosti koncovým uživatelům [40].

Doporučovací systémy mají obvykle uživatelské rozhraní, které hraje významnou roli při přijímání doporučení. Tímto rozhraním může být jednoduše návrh na určitém místě v rámci aplikace, nebo častěji seznam doporučení, často seřazených, poskytovaných uživateli na vyžádání. Mnoho doporučovacích systémů navíc vyžaduje zadání konfiguračních parametrů, uživatelských preferencí a určité formy uživatelského profilu. Všechna tato rozhraní výrazně ovlivňují použitelnost doporučovacího systému jako celku [40].

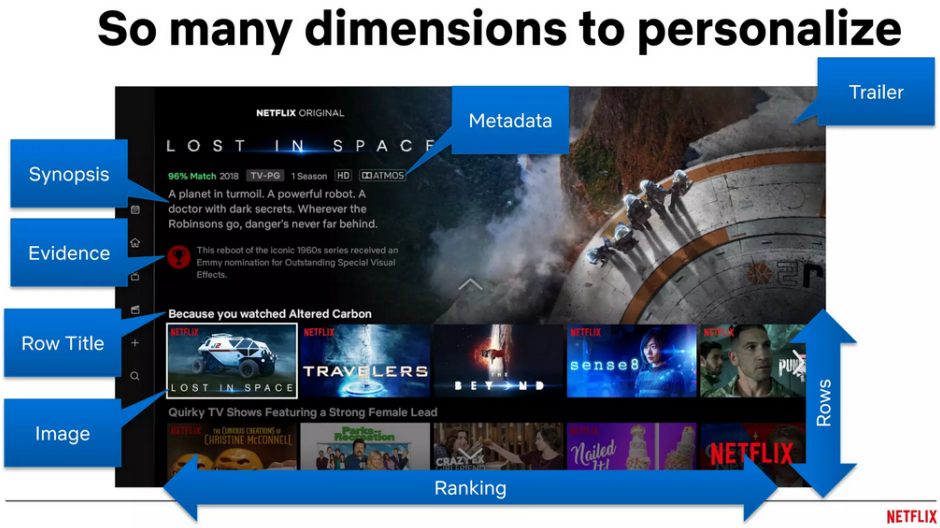
2.5 Existující systémy pro doporučování

Tato sekce je zaměřená na seznámení s existujícími komerčními doporučovacími systémy, nejen pro doporučování filmů.

2.5.1 Netlix

Ve článku [17] autoři Gomez-Uribe a Neil Hunt podávají přehled doporučovacího systému, který v té době používala společnost Netflix. Autoři zdůrazňují význam doporučení pro udržení a zapojení zákazníků, přičemž významné procento spotřeby obsahu na Netflixu je určeno personalizovanými návrhy. Systém využívá různé algoritmické přístupy, jako je kolaborativní filtrování (na základě uživatelů a položek), maticová faktorizace a filtrování na základě obsahu, k identifikaci a řazení filmů a televizních pořadů, které odpovídají preferencím uživatelů. Článek zdůrazňuje význam personalizovaného řazení a integrace více algoritmů pro generování účinných doporučení. Kromě toho zdůrazňuje potřebu vyvážit průzkum nového obsahu s využíváním populárního obsahu a zajistit, aby uživatelé objevili rozmanitou škálu filmů a televizních pořadů, které odpovídají jejich zájmům.

Netflix však nepersonalizuje pouze samotné doporučování filmů pro každého uživatele. Na obrázku 2.3 lze vidět, že personalizují také další faktory, jako jsou metadata či vybraný trailer u filmu.



Obrázek 2.2: Výběr personalizovaných faktorů, které ovlivňují spokojenost uživatelů s doporučením, převzatý z [6]

2.5.2 Amazon Prime Video

Doporučovací algoritmus společnosti Amazon se od svého počátečního vývoje výrazně vyvinul. V roce 2003 vyšel článek [28] od Grega Lindena, Brenta Smithe a Jeremyho Yorka. V něm byl představen koncept kolaborativního filtrování mezi položkami, který se zaměřuje spíše na vztahy mezi položkami než na podobnosti mezi uživateli. Tento přístup, který analyzuje historii nákupů na úrovni položek, zlepšil kvalitu doporučení a nabídl výpočetní výhody ve srovnání s kolaborativním filtrováním založeným na uživatelích.

Jednou z výzev, které výzkumníci společnosti Amazon čelili, bylo přesné měření příbuznosti. Aby tento problém vyřešili, zavedli metriku příbuznosti založenou na rozdílových pravděpodobnostech, která zohledňuje zvýšenou pravděpodobnost nákupu položky B vzhledem k nákupu položky A.

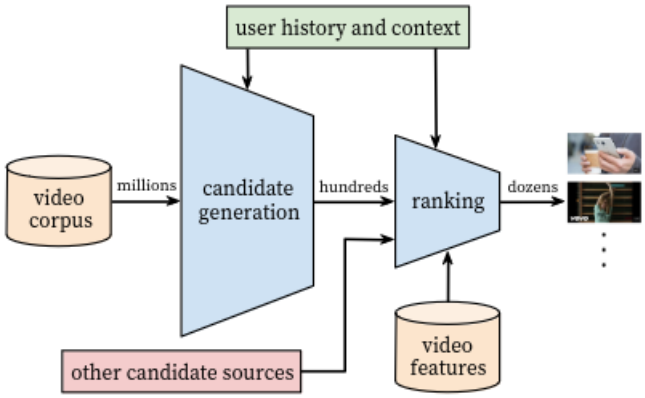
V roce 2014 dostal tým vedený Vijai Mohanem za úkol navrhnout nový doporučovací algoritmus pro službu Prime Video. K řešení problému doplňování matic použili hluboké neuronové sítě, konkrétně autoenkodéry. Trénováním autoenkodéru na chronologicky seřazených datech o zhlédnutých filmech dosáhl tým výrazného zlepšení výkonu algoritmu. Při měření úspěšnosti doporučení alespoň jednoho filmu, který by uživatel zhlédl během dvou týdnů, překonal tento přístup algoritmus kolaborativního filtrování podle položek dvojnásobně [19].

2.5.3 Youtube

Článek [13] pojednává o systému doporučování videí na **YouTube**, který přihlášeným uživatelům poskytuje personalizované sady videí na základě jejich předchozích aktivit na webu.

Cílem doporučovacího systému je pomoci uživatelům objevit kvalitní videa odpovídající jejich zájmům a udržet jejich pozornost pravidelnou aktualizací doporučení. Systém se potýká s problémy, jako jsou nedostatečná metadata, rozsáhlé množství uživatelů a krátké délky videí. Pro generování personalizovaných doporučení systém využívá osobní aktivity uživatele (sledovaná, oblíbená, lajkovaná videa) jako tzv. seeds a rozšiřuje množinu videí procházením grafu videí založeného na společném navštěvování. Doporučení jsou pak seřazena pomocí různých signálů relevance a rozmanitosti. Systém využívá dávkově orientovaný přístup k předběžným výpočtům, který umožňuje přístup k velkému množství dat a dostatečným prostředkům procesoru při zachování nízké latence při obsluze.

Článek se rovněž zabývá cíli systému, výzvami, návrhem, uživatelským rozhraním a podrobnostmi implementace.



Obrázek 2.3: Abstraktní přehled doporučovacího systému využívaného v Youtube, převzatý z [11]

Kapitola 3

Implementace

Výstup bakalářské práce se skládá z doporučovacího systému a uživatelské aplikace. V této sekci jsou popsány implementace obou částí.

3.1 Technologie využité k implementaci

3.1.1 Doporučovací systém

- **Python** [47] – Jazyk Python byl využit k implementaci doporučovacího systému. Byl zvolen z důvodu, že obsahuje velké množstvi knihoven, které lze využít jak při zpracování dat, tak při vytváření samotného doporučovacího systému.
- **Pandas** [34] – Tato knihovna pomáhá při práci s daty. Lze pomocí ní jednoduše načíst data do paměti, modifikovat je a následně je uložit do souboru.
- **Numpy** [38] – Knihovna Numpy byla využita k matematickým operacím. Nabízí například různé vektorové matematické operace, které jsou navíc optimalizované. Je tedy o mnoho více efektivní využít tyto operace, než využívat například cykly.
- **Surprise** [22] – Jedná se o `scikit`[1] jazyka Python. Slouží k tvorbě a analýze doporučovacích systémů. Nabízí mnoho algoritmů využívaných v doporučovacích systémech, funkcí pro analýzu dat nebo také funkce k evaluaci těchto systémů.
- **scikit-learn** [41] – Z této knihovny byly využity funkce pro tvoru systému v sekci 3.3, jako jsou `TfidfVectorizer` nebo funkce pro výpočet kosinové podobnosti.

3.1.2 Webová aplikace

- **Django** [2] – Vysokoúrovňový webový framework napsaný v jazyce Python, který podporuje rychlý vývoj webových aplikací. Django používá architekturu Model-View-Template (MVT), která usnadňuje oddělení logiky aplikace, prezentace a datových modelů. Framework se zaměřuje na automatizaci běžných úkolů, jako je správa databází, formulářů a autentizace uživatelů.
- **React** [5] – Populární JavaScriptová knihovna pro vývoj uživatelských rozhraní webových aplikací, kterou vyvinul Facebook. Jedná se o deklarativní a komponentově ori-

[1] https://projects.scipy.org/scikits.html

entovaný framework, který se zaměřuje na vytváření jednostránkových aplikací (Single Page Applications, SPA).

- **TypeScript** [20] – Nadstavba jazyka JavaScript, která přidává statické typování, což umožňuje lepší kontrolu nad kódem a detekci chyb v raných fázích vývoje. Vyvinul jej Microsoft a jeho první verze byla zveřejněna v roce 2012. TypeScript rozšiřuje možnosti JavaScriptu tím, že poskytuje typovou kontrolu, třídy, rozhraní, generické typy a další pokročilé funkce. Podporuje lepší čitelnost kódu, usnadňuje údržbu a zvyšuje spolehlivost aplikací.

- **Material-UI** [53] – Významná knihovna komponent React, která poskytuje ucelenou sadu komponent uživatelského rozhraní v souladu s pokyny Material Design společnosti Google. Tato knihovna usnadňuje rychlý a konzistentní vývoj vizuálně atraktivních, responzivních a přístupných uživatelských rozhraní ve webových aplikacích. Díky integraci Material-UI s Reactem mohou vývojáři využívat širokou škálu předpřipravených komponent, jako jsou tlačítka, formuláře, navigační prvky a dialogy, a zároveň se zaměřit na budování bezproblémového uživatelského prostředí a zkrácení doby vývoje.

- **Lodash** [12] – Všestranná a výkonná obslužná knihovna jazyka JavaScript, která zjednodušuje práci s poli, objekty, řetězci a dalšími datovými strukturami při vývoji webových stránek. Poskytuje rozsáhlou škálu modulárních a výkonných funkcí určených k provádění běžných operací, jako je iterace, manipulace, filtrování a transformace, a zároveň řeší mnohá omezení a nekonzistence nativních metod jazyka JavaScript.

- **React-Query** [29] – Robustní knihovna pro načítání dat a správu stavu aplikací React, která je navržena tak, aby zjednodušila proces načítání, ukládání do mezipaměti, synchronizace a aktualizace dat na straně serveru. React Query nabízí intuitivní a deklarativní rozhraní API, které vývojářům umožňuje bez námahy zpracovávat asynchronní úlohy načítání dat, jako je například provádění požadavků HTTP, bez nutnosti složitých řešení správy stavu. Využitím funkcí, jako je načítání dat na pozadí, automatické ukládání do mezipaměti a aktualizace v reálném čase, zvyšuje React Query celkový uživatelský komfort a výkon webových aplikací.

- **DRF_YASG** [24] – Flexibilní a přizpůsobitelná knihovna, která usnadňuje vytváření specifikací **OpenAPI 2.0 (Swagger)** a **OpenAPI 3.0.0 (ReDoc)** pro rozhraní API založená na frameworku Django REST. Generuje schémata pomocí serializátorů, sad pohledů a směrovačů DRF a extrahuje příslušná metadata, jako jsou parametry dotazu a schémata odpovědí, pomocí dokumentových řetězců Pythonu a vlastních inspektorů. Pomocí DRF_YASG mohou vývojáři snadno vytvářet přehlednou a interaktivní dokumentaci API, což zlepšuje použitelnost a údržbu jejich aplikací založených na API.

3.2 Datová sada a její zpracování

K implementaci této práce byla využita datová sada[2] poskytovaná `MovieLense`[3]. Jedná se o nekomerční web, zprostředkovávající personalizovaná doporučení filmů.

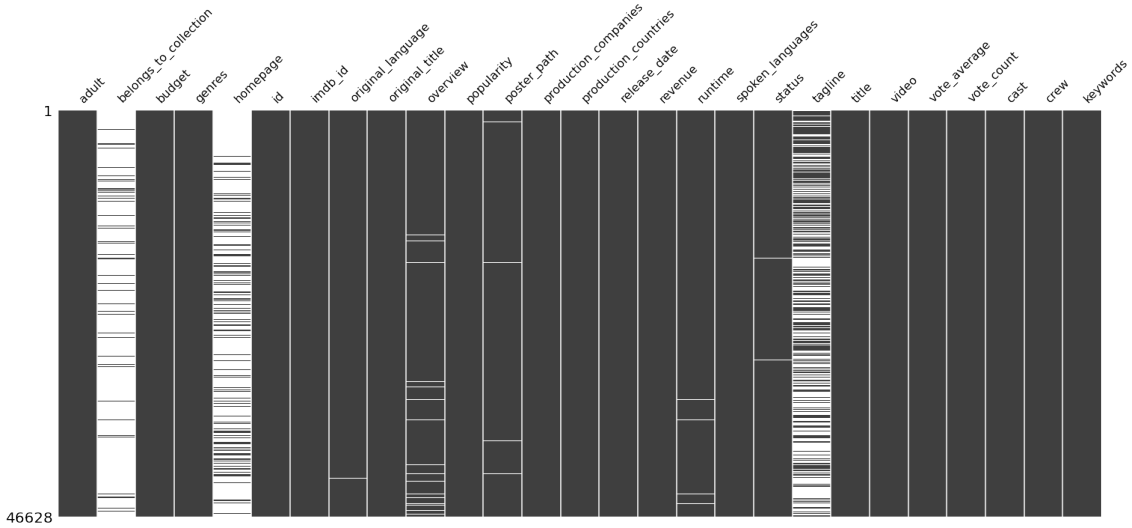
[2] https://www.kaggle.com/datasets/rounakbanik/the-movies-dataset
[3] https://movielens.org/

3.2.1 Popis datové sady

Datová sada obsahuje několik tabulek, z nichž byly využity následující:

- `ratings.csv` – Obsahuje 4 sloupce: `movieId`, `userId`, `rating`, `timestamp`. Rating označuje hodnocení filmu daným uživatelem, je v intervalu $\langle 0, 5; 5 \rangle$. `Timestamp` znázorňuje, kdy bylo hodnocení uděleno. Tato tabulka je řídká, jelikož není zvykem, že by uživatel ohodnotil všechny filmy v systému.
- `credits.csv` – Zde jsou informace o lidech, kteří se podíleli na tvorbě daného filmu. Jsou zde jak herci, tak členové filmového štábu.
- `keywords.csv` – Klíčová slova uvedená ke každému filmu.
- `movies_metadata.csv` – V této tabulce jsou filmy a informace o nich. Je zde například popis filmu, název, do jaké patří kolekce, průměrné hodnocení na IMDB[4], rozpočet, země původu a další. K této tabulce byly pomocí `movieId` připojeny také tabulky `credits.csv` a `keywords.csv`.

Jak lze vidět v obrázku 3.1, tabulka `movies_metadata.csv` není řídka a většina filmů má všechny důležité položky vyplněné, tento fakt byl využit při návrhu systému. Pokud by byla tabulka více řídká, bylo by možno například přikládat rozdílné váhy jednotlivým položkám v případě, že by ostatní u konkrétního filmu chyběli.



Obrázek 3.1: Řídkost dat v tabulce `movie_metadata.csv`, tmavé čáry znamenají, že data z vybraného sloupce jsou obsažena v daném záznamu

3.2.2 Předzpracování dat

Tabulka `movies_metadata.csv` obsahuje 46 628 záznamů filmů. Tabulka `ratings.csv` má 26 024 289 záznamů, z nichž je 45 115 unikátních záznamů filmů a 270 896 unikátních záznamů uživatelů.

[4] https://www.imdb.com/

Nejdříve jsou z tabulky `movies_metadata.csv` odebrány filmy, které byly natočeny po roce 1970. Dále jsou odebrány z tabulky `ratings.csv` filmy, které mají méně než 50 ohodnocení v této tabulce. Následně jsou odebráni uživatelé, kteří mají méně než 50 udělených hodnocení. Hodnoty těchto konstant byly zvoleny na základě experimentování s velikostí datové sady. Cílem experimentů bylo najít takové hodnoty, které by vyprodukovaly datovou sadu s optimální velikostí.

Výsledná datová sada obsahuje 2 916 filmů a 49 354 uživatelů. Celkový počet záznamů s hodnocením filmů od uživatelů je 6 102 966.

3.3 Filtrování založené na obsahu

Jako první systém byl zvolen systém **založený na obsahu** (dále pouze **CB**) Hlavní funkcionalitou této časti je doporučení filmů, které jsou podobné vybranému filmu. Nebere se zde v potaz popularita filmů, či uživatelská preference. Pro výpočet podobnosti byla využita kombinace algoritmů TD-IDF a kosínova podobnost [49][51].

3.3.1 TD-IDF

Implementaci algoritmu TD-IDF nabízí knihovna `scikit-learn` a vzorec zní následovně:

$$\text{TF-IDF}(t, d, D) = \text{TF}(t, d) \cdot \text{IDF}(t, D) \tag{3.1}$$

kde t je termín obsažený v dokumentu, d je dokument a D je korpus, tedy soubor všech dokumentů.

Vzorce pro $\text{TF}(t, d)$ a $\text{IDF}(t, D)$ jsou následující:

$$\text{TF}(t, d) = \frac{\text{Počet výskytů termínu } t \text{ v dokumentu } d}{\text{Celkový počet termínu v dokumentu } d} \tag{3.2}$$

$$\text{IDF}(t, D) = \log \frac{\text{Celkový počet dokumentů v korpusu } D}{\text{Počet dokumentů obsahujících termín } t} \tag{3.3}$$

Pro výpočet podobnosti se využívají následující sloupce z tabulky filmů: název, popis, klíčová slova. Také se využívá jméno režiserá, kterému je přidána trojnásobná váha. V potaz se také berou první tři hlavní herci ve filmu. U jmen je potřeba si dávat pozor a odebrat z nich mezery. Je totiž nutné rozlišovat ruzné herce a řežiséry. Například herečky Jennifer Lawrence a Jennifer Aniston by mohly způsobovat částečnou podobnost mezi filmy, přitom jsou to ale dvě rozdílné osoby. Nakonec je využit i žánr filmu s čtyřnásobnou váhou. Zmíněné váhy byly zvoleny po uživatelském testování.

Tyto informace jsou z filmů extrahovány, transformovány do textové podoby a vloženy všechny do jednoho sloupce. Tyto jednotlivé informace jsou odděleny pouze mezerou. Poté je využit tento mix sloupců společně s identifikátorem filmu jako vstup do algoritmu TD-IDF. Výstupem TD-IDF je matice obsahující **příznaky** filmů, kde každá hodnota udává skóre udělené algoritmem TD-IDF pro jednotlivé termíny obsažené ve filmech.

3.3.2 Kosinová podobnost

Výstup algoritmu TD-IDF je následně předán do algoritmu `cosine_similarity` z knihovny `scikit-learn`. Ten pouze vypočítá kosinovou podobnost a vrátí matici, která má velikost $M \times M$, kde M je počet filmů v systému. Tato matice obsahuje párovou podobnost filmů.

Vzorec kosinové podobnosti je následující:

$$\cos(\theta) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\|\|\mathbf{B}\|} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2}\sqrt{\sum_{i=1}^{n} B_i^2}} \tag{3.4}$$

kde **A** a **B** jsou porovnávané vektory. Výstupem kosinové podobnosti je číslo z rozsahu $(-1; 1)$, kdy 1 udává maximální možnou podobnost a -1 je prakticky pravý opak, tedy naprosto jiný film. Kosinová podobnost byla zvolena z důvodu, že některé filmy mohou mít například delší popis. To by znamenalo, že některá slova, která by měl daný film stejné s jiný filmem, který má kratší popis, by byla v textu častěji. To by způsobilo relativně velkou vzdálenost mezi těmito filmy, což není žádané. Kosinová podobnost neměří vzdálenost mezi položkami, ale měří úhel, který svírají od počátku souřadnicového systému. Kosinová podobnost dosahovala nejlepších výsledků narozdíl od Euklidovy vzdálenosti [51], nebo Pearsonova korelačního koeficientu [51].

Výslednému systému jsou následně předány dva parametry, prvním je jedinečný identifikátor filmu a druhým je `k`, který značí, kolik filmů s největší podobností k filmu v prvním parametru má být navráceno.

3.4 Kolaborativní filtrování

Jako další systém byl zvolen systém **kolaborativního filtrování** (dále pouze **CF**). Z metod zmíněných v sekci 2.2.1 byla vybrána metoda **faktorizace matic**, konkrétně algoritmus **SVD** [31] a to hned z několika důvodu. Hlavním byl ten, že má jedny z nejlepších výsledku [18][27] na datových sadách, které mají podobnou velikost jako datová sada v této práci. Je také lépe škálotavelný, jelikož není nutné mít v paměti párové podobnosti položek nebo uživatelů. V neposlední řadě, je v této práci využita datová sada, která obsahuje tabulku hodnocení, která je řídká. Metody založené na modelu mohou účinně zpracovávat řídká data tím, že odhalí základní vzory a latentní faktory v matici interakcí mezi uživatelem a položkou. Má také své nevýhody, jedna z nich je delší doba pro trénování.

Byla využita implementace nabízená knihovnou `surprise`. SVD rozkládá matici hodnocení $\boldsymbol{R}$ (uživatelů a položek) na tři matice: $\boldsymbol{U}$, $\boldsymbol{\Sigma}$ a $\boldsymbol{V^T}$.

- U obsahuje informace o uživatelích a jejich vztahu k latentním faktorům.
- Σ je diagonální maticí singulárních hodnot, která obsahuje informace o síle a důležitosti latentních faktorů.
- V^T obsahuje informace o položkách a jejich vztahu k latentním faktorům.

Rozklad SVD můžeme zapsat takto:

$$R_{m \times n} = U_{m \times k} \Sigma_{k \times k} V^T_{n \times k} \tag{3.5}$$

R je matice hodnocení o rozměrech $m \times n$, kde m je počet uživatelů a n je počet položek, k je počet latentních faktorů.

Latentní faktory jsou skryté proměnné, které ovlivňují pozorování a chování v určitém systému nebo datasetu. V kontextu doporučovacích systémů jsou latentní faktory vlastnosti, které nejsou vidět na první pohled. Latentní faktory pomáhají vysvětlit preference a chování uživatelů a interakce mezi uživateli a položkami.

Při použití SVD v kolaborativním doporučovacím systému se snižuje dimenzionalita matice hodnocení R tím, že se zvolí menší počet latentních faktorů. Tímto způsobem lze získat kompaktnější reprezentaci dat, která zahrnuje důležité vlastnosti a zároveň redukuje šum.

Při trénování tohoto modelu se minimalizuje následující chyba:

$$\sum_{r_{ui} \in R_{train}} (r_{ui} - \hat{r}_{ui})^2 + \lambda \left(b_i^2 + b_u^2 + ||q_i||^2 + ||p_u||^2 \right) \tag{3.6}$$

kde r_{ui} je hodnocení uživatele u udělené položce i, $\hat{r}_{ui}$ značí předpovídané hodnocení uživatele u pro položku i, které je vypočítáno následujícím vzorcem:

$$\hat{r}_{ui} = \mu + b_u + b_i + q_i^T p_u \tag{3.7}$$

kde μ značí průměrné hodnocení napříč všemi hodnoceními, b_u značí tzv. zkreslení (bias) pro uživatele u, b_i je zkreslení pro položku i, q_i je faktor položky i a p_u je faktor uživatele u. Pokud jsou uživatel nebo položka neznámí, předpokládá se, že zkreslení a faktory jsou nulové. Výpočet zkreslení a faktorů se provádí velmi jednoduchým **stochastickým gradientním sestupem**:

$$\begin{aligned} b_u &\leftarrow b_u + \gamma(e_{ui} - \lambda b_u) \\ b_i &\leftarrow b_i + \gamma(e_{ui} - \lambda b_i) \\ p_u &\leftarrow p_u + \gamma(e_{ui} \cdot q_i - \lambda p_u) \\ q_i &\leftarrow q_i + \gamma(e_{ui} \cdot p_u - \lambda q_i) \end{aligned} \tag{3.8}$$

kde γ udává rychlost učení a je nastaven na hodnotu 0.005, λ je regularizační výraz nastavený na hodnotu 0.02. e_{ui} značí chybu u předpovídaného hodnocení a je vyjádřena jako $e_{ui} = r_{ui} - \hat{r}_{ui}$. Základní hodnoty jsou inicializovány na 0. Uživatelské a položkové faktory jsou náhodně inicializovány podle normálního rozdělení. Tyto kroky se provedou nad všemi hodnoceními v trénovací datové sadě 20x.

Souhrnem, SVD v kontextu kolaborativních doporučovacích systémů nám umožňuje identifikovat latentní faktory, které ovlivňují preference uživatelů a interakce mezi uživateli a položkami, a vytvářet tak přesnější a efektivnější doporučení.

Vstupní data byla náhodně rozdělena na trénovací a testovací v poměru 80/20. Poté byl model natrénován na trénovacích datech. Trénování na stroji, na kterém byl systém vyvíjen, trvalo zhruba 40 sekund.

Vstupem tohoto systému je unikátní identifikátor uživatele a filmu, který uživatel zatím neohodnotil. Systém se následně snaží hodnocení předpovídat a vrací číslo v intervalu $\langle 0.5; 5 \rangle$.

3.5 Hybridní doporučovací systém

Hybridní doporučovací systém vrací doporučení pro konkrétního uživatele a kombinuje výstupy předchozích dvou zmíněných systémů. Z existujících metod zmíněných v sekci 2.2.5 byla zvolena pro implementaci tohoto systému **kaskádová** metoda. Na vstupu systému je identifikátor uživatele a seznam filmů, které ohodnotil. Tento seznam je ještě před vstupem do tohoto systému seřazen podle hodnocení a je z něj vybráno maximálně 25 položek. Tyto filmy jsou na vstupu systémů CB, následně je vybráno 25 nejvíce podobných filmů

pro každý z filmů ve vstupním parametru. Z tohoto seznamu jsou vyjmuty duplicitní filmy a je pro ně následně vygenerované skóre CF systémem. Výsledné skóre je tedy kombinací předchozích dvou systémů.

Hlavní výhoda kombinace CF a CB systémů je v tomto případě škálovatelnost a rychlost doporučování. Není tedy nutné generovat CF doporučení pro každou kombinaci filmu a uživatele, ale pouze pro filmy podobné těm, které se mu líbili. Výpočet kosinové podobnosti v CB systémů není výpočetně náročný, velké množství nových filmů zde tedy nezpůsobí tolik problémů. Tyto problémy nastávají až u CF systému, kdy pro kombinaci uživatele a všech filmů je tato operace výpočetně náročná a zdlouhavá. Rostoucí množství uživatelů a hodnocení nemá vliv na CB systém. Ovlivní to pouze trénování CF systémů, které ale není možné provádět v realném čase, právě z důvodu výpočetní náročnosti. Díky svojí rychlosti dokáže systém generovat doporučování v reálném čase a není nutné si tyto doporučení vypočítat předem a následně ukládat buď do paměti, nebo do uložiště. Navíc by se tyto předvypočítané doporučení musely přepočítavat pro konkrétního uživatele vždy, kdy by přidal nové hodnocení. Náročnost této operace by rostla s každým novým uživatelem nebo filmem v systému.

Pro vygenerování doporučení musí uživatel ohodnotit alespoň 5 filmů, tímto je alespoň částečně vyřešen problém studeného startu. Takto nízké číslo bylo zvoleno z důvodu, že noví uživatelé nechtějí strávit příliš hodně času výběrem filmů, které se jim líbí. Také bylo nutno vyřešit problém, kdy je do systému přidán nový uživatel, který nebyl viděn při trénování systému kolaborativního filtrování. Trénování tohoto systému je poměrně zdlouhavý proces a proto to není možné provést při každém vyžádání nových doporučení. Byl tedy zvolen přístup, kdy je vytvořen vektor o rozměru odpovídajícím počtu filmů v systému, který obsahuje hodnocení uživatele pro jednotlivé filmy. Poté je vypočítana podobnost mezi vybraným uživatelem a ostatními uživateli v systému, kteří jsou rezeprezentovaní maticí, která je řídká. Jelikož je počítána podobnost pouze jednoho uživatele vůči všem uživatelům, je tento proces přijatelně dlouhý. Na stroji, kde byl systém vyvíjen, daná operace trvá zhruba 4,5 sekundy. Výstupem je identifikátor uživatele, který byl v trénovací sadě a je pro něj tedy možno generovat doporučení. Dokud nedojde k novému trénování modelu, jsou doporučení pro přihlášeného uživatele generována na základě nejpodobnějšího uživatele v systému.

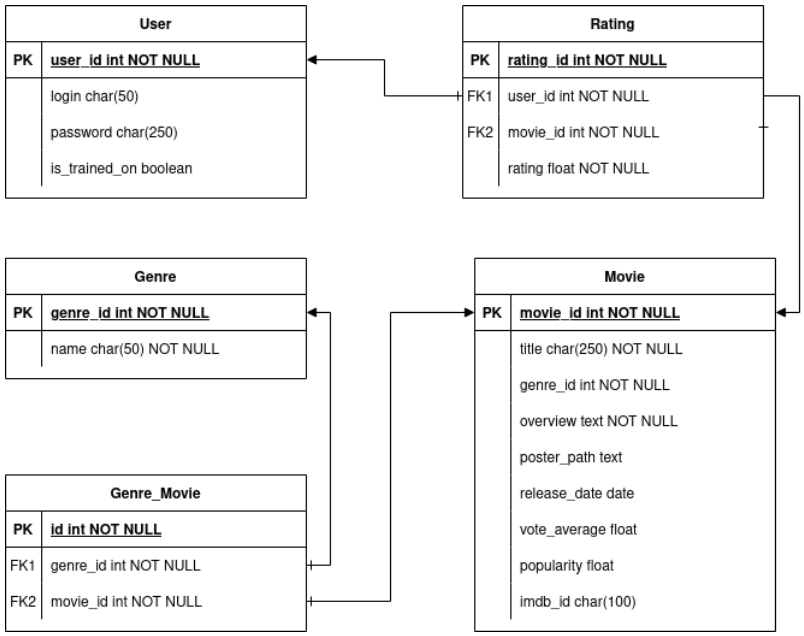
3.6 Aplikace

Cílem aplikace bylo demonstrovat chování systému a také byla využita k testování na uživatelích. Zpracování požadavků a správa databáze je implementovaná pomoci již zmíněného frameworku Django, v jazyce Python(dále pouze **backend**). Uživatelské rozhraní, které je dostupné z webového prohlížeče, je implementováno pomocí React a TypeScript(dále pouze **frontend**). Komunikaci mezi frontendem a backendem zprostředkovává architektura REST[15].

Pro inicializaci doporučovacího systému je nutné nahrát do paměti model pro kolaborativní filtrování a matici zobrazující podobnost jednotlivých filmů pro filtrování založeném na obsahu. Tyto operace mohou být zdlouhavé a bylo by velice neefektivní je provádět při zpracování každého dotazu. Z toho důvodu byl vytvořen Modul **Recommendation** 3.6.3, který tuto inicializaci provede při startu samotného systému a následně data uchováva v paměti. Data spolu s modelem, jsou poté v paměti po celou dobu běhu serveru. Při příchozím požadavku na generování doporučení pro uživatele se následně pouze zavolá funkce tohoto modulu a doporučování je vygenerováno.

3.6.1 Návrh databáze

Data z datové sady, využité při vytváření doporučovacího systému, lze pomocí scriptu nahrát do databáze, jejíž schéma je popsáno pomocí ER diagramu na obrázku 3.2.



Obrázek 3.2: ER diagram aplikace

Skripty pro inicializaci databáze je nutné spustit v rámci první inicializace systému. Tyto skript nahrají všechna data, která byla použita při tvorbě systému. V tabulce uživatelů si lze všimnout, že pole `login` a `password` jsou nepovinná. To je z důvodu, že jsou do databáze importovaní uživatelé ze souboru `ratings.csv`, o kterých jsou známé pouze jejich identifikátory. Při vytváření nového uživatele je však přihlašovací jméno a heslo nutné, jelikož bez nich se uživatel nedokáže do systému přihlásit.

3.6.2 Koncové body (API)

Jedná se koncové body sloužící ke komunkaci mezi backendem a frontendem. Klient na tyto koncové body odesílá dotazy a server mu odpovídá.

Na obrázku 3.3 lze vidět všechny koncové body rozdělené do skupin. V této kapitole je jejich stručný popis.

Koncové body filmů

- `movies` – Vrací seznam filmů na základě předaných identifikátorů.
- `popular` – Filmy jsou seřazeny podle popularity a poté jsou odeslány na klienta, tyto filmy lze stránkovat. V případě nespecifikované stránky se na klienta odešle první stránka filmů.
- `recommendations` – Tento koncový bod volá funkce modulu **Recommendation** 3.6.3. Vyžádá si z hybridního systému identifikátory doporučených filmů pro uživatele a následně je získá z databáze a pošle je klientovi. V aktuální implementaci vrací maximálně 25 filmů.

Obrázek 3.3: Seznam koncových bodů

- `search_movies` – Slouží pro vyhledávání mezi filmy a vrací všechny filmy odpovídající vyhledávanému výrazu. U filmů vyhledávaných pomocí textu lze také přepínat stránky a na jedné stránce je maximálně 24 filmů.

Koncové body uživatelů

- `login` – V případě korektních údajů odešle klientovi udáje o uživateli a ten je následně přihlášen.
- `register` – Slouží k registraci uživatele. Při úspěšné registraci vrací zprávu informující o této skutečnosti klienta.

Koncové body hodnocení

- `rate_movie` – Na základě předaného identifikátoru uživatele, filmu a hodnocení tyto hodnoty uloží do databáze.
- `users_rating` – Vrací všechny filmy, které uživatel ohodnotil.

Koncové body modelu

Tyto koncové body slouží pouze k práci s modulem **Recommendation** 3.6.3.

- `retrain_model` – Spustí trénování modelu pro kolaborativní filtrování s předchozími trénovacími daty, ke kterým jsou přidány všechny nové hodnocení.
- `is_model_being_trained` – Vrací pouze informaci o tom, zda probíhá trénování nebo ne.

3.6.3 Modul Recommendation

V této části je popsán modul, který je součastí backend aplikace. Obsahuje hybridní doporučovací systém, jednotlivé podsystémy a další podpůrné funkce. Nejdříve je tento modul inicializován při startu backend aplikace.

Části, které je nutno inicializovat jsou následující:

- **Podobnosti jednotlivých filmů** – Ke generování doporučení podsystémem CB, je nutné znát podobnosti jednotlivých filmů. Do paměti jsou načtena data, která jsou upravena tak, jak je popsáno v 3.3. Na nich je poté vypočítáná vysledná matice podobnosti a je uložena v paměti.
- **Model systémů CF** – Do paměti je také načten model, který byl vytvořen a natrénován v části 3.4.
- **Hodnocení uživatelů** – Také je potřeba do paměti nahrát všechna hodnocení, která jsou v systému. Je tak učiněno z důvodu, že je nutno vyřešit problém přidání nového uživatele a získávání všech hodnocení z databáze je příliš zdlouhavý proces (i několik minut). V tomto případě budou sice hodnocení uložena duplicitně, a to v csv souboru a v samotné databázi, je to ale kompromis, díky kterému může být doporučení generováno v poměrně rychlém čase. Při přidání nového hodnocení, je toto hodnocení uloženo do databáze, do paměti a také do souboru, ze kterého se načítají hodnocení při inicializaci systému.

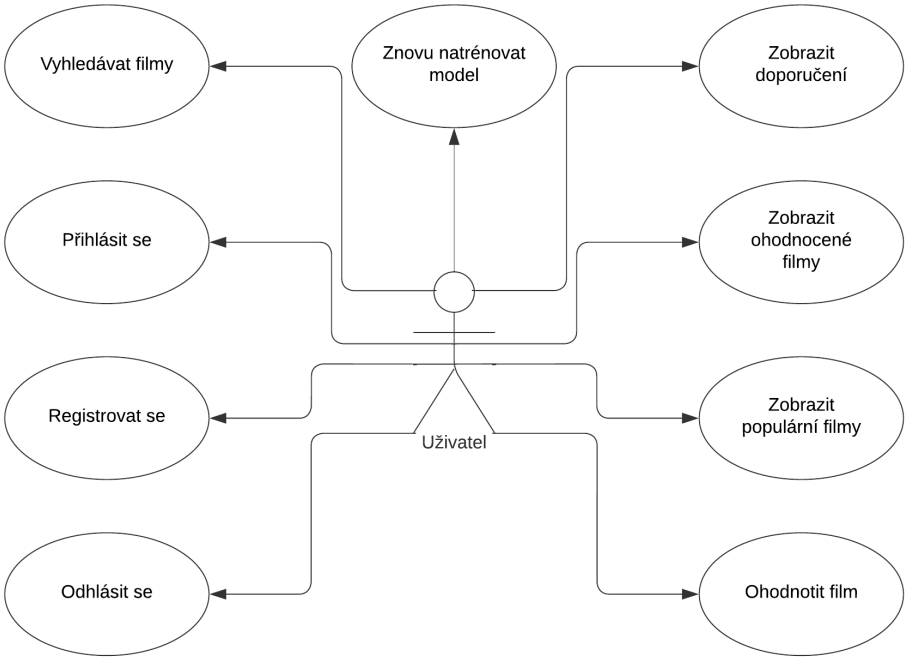
Dále tento modul obsahuje tři podpůrné funkce:

- `new_user` – Tato funkce je volaná v případě, že se generují doporučení pro nového uživatele, na kterém ještě nebyl model trénovaný. Byl zvolen přístup, kdy se při vyžádání doporučení pro uživatele, na kterém není systém ještě trénovaný, vypočítá kosinová podobnost pro jednoho uživatele vůči všem uživatelům. Tento výpočet trvá zhruba 0,5 vteřiny pro právě jednoho uživatele. Jelikož je v systému přes 49 tisíc uživatelů, tento výpočet by trval zhruba 6 hodin a 45 minut pro všechny uživatele, což není přijatelné. Pravděpodobně by bylo náročné udržovat v paměti v jednu dobu matici o rozměrech $49\,000 \times 49\,000$, která by neustále rostla s každým novým uživatelem. Navíc by to bylo poměrně zbytečné, protože by se hodnoty mohly hodně změnit s každým novým ohodnocením, které do systému vstoupí. Tento přístup je komporimis mezi kvalitou doporučovaných položek a dobou potřebnou k vygenerování doporučení.
- `retrain` – Jedná se o funkci, která znovu natrénuje SVD model využívaný v CF systému 3.4. Je možné ji využívat různými způsoby, například v pravidelných intervalech několikrát denně, aby měly doporučení co největší kvalitu. V této práci je toto využití poměrně zbytečné, jelikož aplikace nemá reálné uživatele, ale slouží pouze k testování. Uživatel může vyzkoušet doporučování před tím, než jsou jeho hodnocení zahrnuta v modelu SVD a znovu poté, když jsou již v natrénovaném modelu a nedochází tedy k hledání nejpodobnějšího uživatele. Nové natrénování probíhá na stejných datech jako první trénování, jsou k nim však navíc přidány všechny nové hodnocení, která v dobu prvního trénování nebyla v systému.
- `add_new_user` – Díky této funkci se přidá hodnocení uživatele také do paměti, do souboru `ratings.csv`, který obsahuje všechna hodnocení v systému a následně do souboru `trainset.csv`. Soubor `trainset.csv` obsahuje hodnocení, na kterých je systém

trénovaný. Hodnocení do souboru `ratings.csv` jsou ukládany pouze z důvodu konzistence dat. Pokud by došlo například ke změně rozložení dat pro trénovací a testovací, mohlo by být nepříjemné zpětně upravovat soubor `ratings.csv`.

3.6.4 Uživatelské rozhraní

Všechny povolené uživatelské intekrace s aplikací lze vidět na diagramu užití 3.4

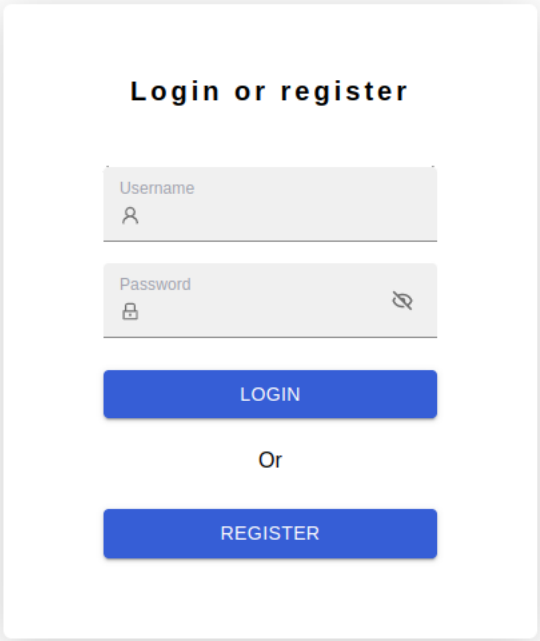


Obrázek 3.4: Diagram užití

Při prvním zapnutí uživatel vidí formulář, kde se může buď zaregistrovat nebo přihlásit viz 3.5. Při registraci musí zadat jméno, které v systému doposud neexistuje. Pokud zadá již existující jméno, bude na tuto skutečnost upozorněn chybovou hláškou. V případě úspěšné registrace je ve vrchní části stránky zobrazeno upozornění zeléne barvy. Uživatel není okamžitě přihlášen, kdyby náhodou chtěl registrovat další účty. Pokud by se ale chtěl uživatel také rovnou přihlásit, stačí kliknout pouze na tlačítko `Login`, jelikož vyplněné údaje ze stránky nezmizí.

Po registraci a následném přihlášení se uživatel uloží do do tzv. **LocalStorage**, což způsobí, že se nemusí přihlašovat při každém obnovení stránky. LocalStorage je uložiště v prohlížeči, do kterého si webové stránky mohou ukládat data, která poté mohou číst a upravovat pouze ony. LocalStorage nemá expirační dobu a data tedy zůstanou v prohlížeči, dokud nejsou někým vymazána.

Po přihlášení uživatel vidí úvodní stránku aplikace, na které jsou zobrazeny filmy, které jsou populární viz 3.6. Filmy jsou napříč celou aplikací zobrázovány v řádcích, kdy na každém řádku jsou maximálně 4 filmy. V případě, že jich je na jednom řádku méně, jsou zarovnány horizontálně do středu. Modré záhlaví stránky je zobrazováno všude, kromě přihlašovací/registrační stránky. Uživatel se pomocí tohoto záhlaví může přepínat mezi



Obrázek 3.5: Stránka sloužící pro přihlášení a registraci

různými stránkami, vidí zde jméno, pod kterým je přihlášen a také se může odhlásit pomocí tlačítka `Logout`

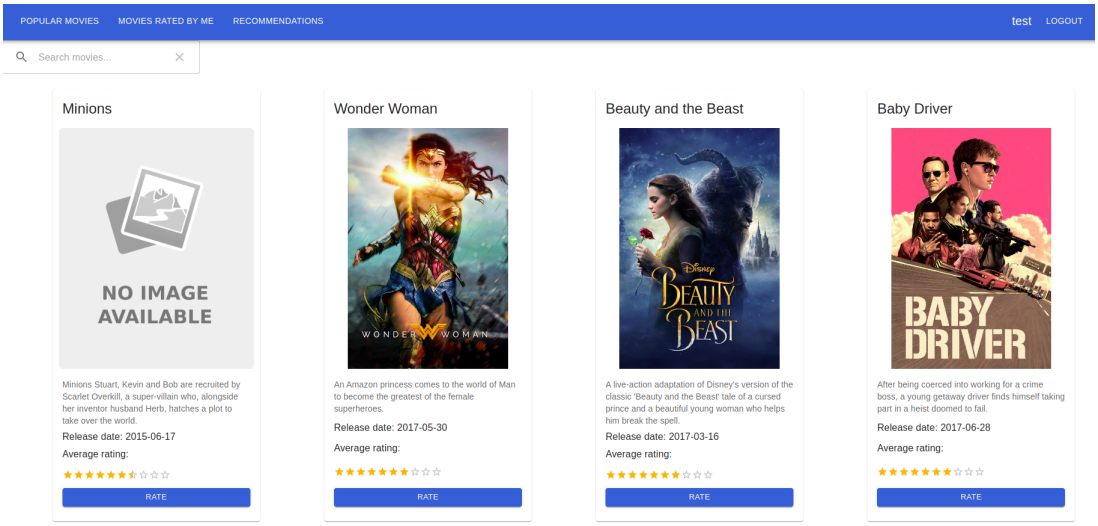
Uživatel může také vyhledávat mezi populárními filmy. Vyhledávání je aplikováno pouze na názvy filmů a proběhne po jedné vteřině od zadání poslední znaku. Vyhledávání může také uživatel vymazat kliknutím na křížek v pravo od vyhledávaného textu. Ve spodní části stránky obsahující populární filmy se nachází stránkování, které může uživatel využit pro hledání filmů, které chce ohodnotit.

U každého filmu uživatel uvidí jeho název, rok vydání, krátký popis a také jeho obrázek, pokud je odkaz na něj v datové sadě stále validní 3.7b. Obrázky jsou získávány pomocí sloupce `imdb_id` z API nabízené stránkou **TMDB**[5]. Jedná se o veřejnou databází filmů, seriálů a televizních pořadů. Pomocí této API lze získat různe informace o filmech a také je hodnotit. Pokud není obrázek dostupný, je zobrazený tzv. placeholder, tedy výchozí obrázek viz 3.7a. Filmy lze hodnotit tlačítkem `Rate`. Po kliknutí na počet hvězdiček je hodnocení uloženo a není potřeba ho nijak potvrzovat.

Na záložce `Movies rated by me` uvidí uživatel všechny filmy, které ohodnotil spolu s hodnocením, které jim dal. Pokud chce své hodnocení změnit, stačí pouze kliknout na tlačítko `Rate` a uvést nové hodnocení.

Dále lze u každého filmu vidět průměrné hodnocení a také hodnocení samotného uživatele, pokud nějaké udělil. Po kliknutí na tlačítko `Rate` může uživatel ohodnotit film. Hodnotí na škále 1-10, pomocí hvězd, viz obrázek 3.7c. Takováto forma hodnocení byla hlavně z dů-

[5] https://www.themoviedb.org

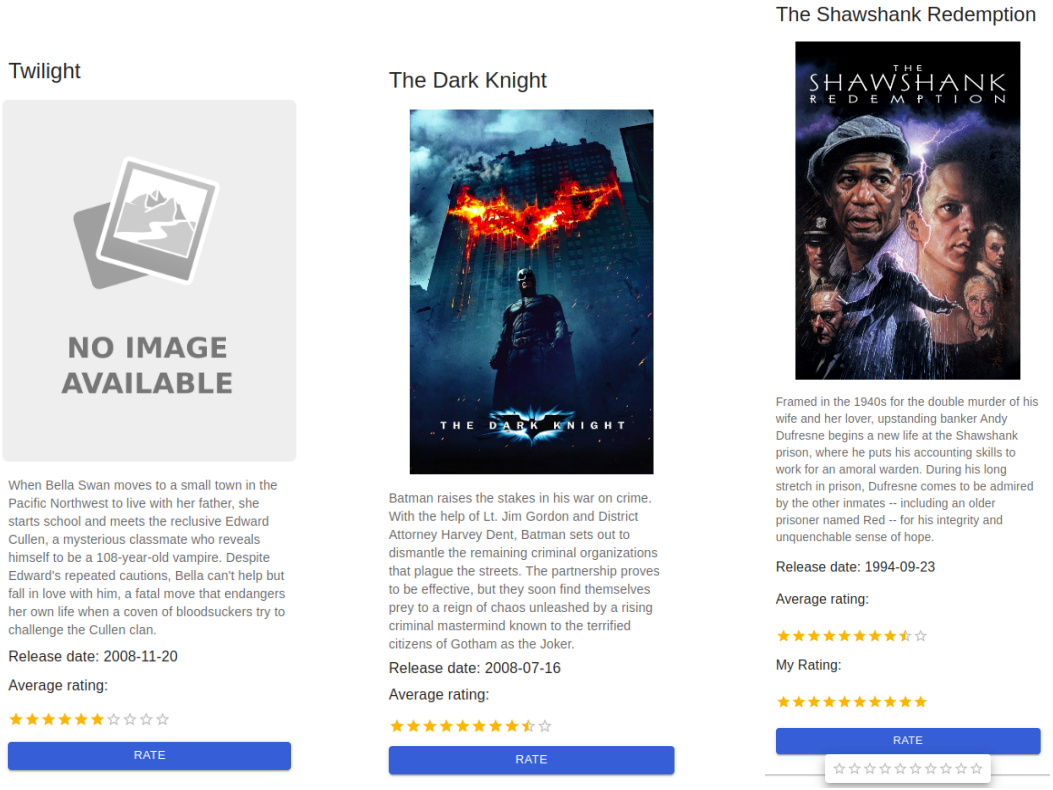


Obrázek 3.6: Stránka obsahující populární filmy

vodu kompatibility s již existujícím hodnocením. Za zmínku ale také stojí výsledky tohoto výzkumu [40], kdy uživatele preferují zmíněnou formu hodnocení položek. Hodnocení je následně standardizováno do intervalu $\langle 0.5; 5 \rangle$.

Pro vygenerování a zobrazení doporučení musí uživatel ohodnotit alespoň 5 filmů. Ty najde v záložce `Recommendations`. Doporučení se vygenerují při kliknutí na záložku. Po vygenerování může uživatel tyto filmy také ohodnotit.

Tlačítko `Retrain` způsobí, že dojde k znovu natrénování modelu. Nyní však budou do trénování zahrnuty nové hodnocení od uživatele. Předpokladem je, že doporučení od systému budou po před trénováním s novými hodnoceními horší, než po něm. To z důvodu, že uživatelovi skutečné preference jsou předtím pouze aproximovány a je hledán jemu nejpodobnější uživatel. Jelikož není předem známo, jak dlouho bude trénování trvat, není možne zobrazit načítací obrazovku na konkrétní dobu. Jelikož není nutné v rámci této práce snižovat odezvu co největším možným způsobem, byl zvolen přístup, kdy se aplikace klienta periodicky dotazuje každou sekundu. Alternativně je možné tuto komunikaci implementovat například pomocí **websocketu** [14]. Díky nim by mohl server klienta upozornit, že trénováni již skončilo a nebylo by nutné zatěžovat server dotazy v průběhu trénování. Jakmile skončí trénování, uživateli jsou zobrazeny doporučení vygenerovaná nově natrénovaným systémem.



(a) Film bez validního obrázku (b) Film s validním obrázkem (c) Film s hodnocením

Obrázek 3.7: Výsledné obrazovky filmů

Kapitola 4

Vyhodnocení a testování

4.1 Vyhodnocení

K evaluaci doporučovacích systému se využívá mnoho technik. Je tedy nutné zvolit takové techniky, které jsou vhodné pro konkrétní doporučovací systém.

Jelikož je systém v této práci implementován tak, že vrací 25 filmů, které jsou pro uživatele podle systému nejvíc relevantní, byly zvoleny metriky, které tuto skutečnost zachycují nejlépe.

4.1.1 Přesnost (Precision)

Přesnost představuje podíl relevantních položek mezi doporučenými položkami. Je to počet skutečně pozitivních doporučení (relevantní položky, které byly doporučeny) dělený celkovým počtem doporučených položek (skutečně pozitivní + falešně pozitivní). Vyšší přesnost naznačuje, že doporučovací systém je dobrý v navrhování položek, které by uživatel považoval za relevantní nebo zajímavé.

$$\text{Přesnost} = \frac{\text{Skutečně pozitivní}}{\text{Skutečně pozitivní} + \text{Falešně pozitivní}} \tag{4.1}$$

4.1.2 Úplnost (Recall)

Úplnost udává podíl relevantních položek, které byly doporučeny ze všech relevantních položek. Je to počet skutečně pozitivních doporučení (relevantní položky, které byly doporučeny) dělený celkovým počtem relevantních položek (skutečné pozitivní + falešně negativní). Vyšší úplnost naznačuje, že doporučovací systém je dobrý v zachycení všech položek, které by uživatel považoval za relevantní nebo zajímavé.

$$\text{Úplnost} = \frac{\text{Skutečně pozitivní}}{\text{Skutečně pozitivní} + \text{Falešně negativní}} \tag{4.2}$$

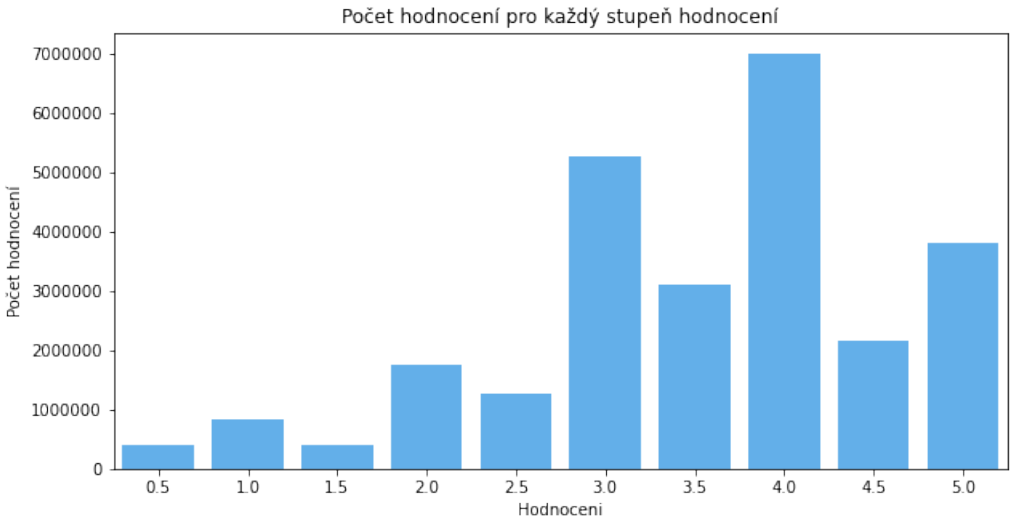
4.1.3 Míra zásahu (Hit Rate)

Míra zásahu je další metrika používaná k hodnocení výkonu doporučovacího systému. Je to počet uživatelů, pro které byla doporučena alespoň jedna relevantní položka, dělený celkovým počtem uživatelů. Vyšší míra zásahu naznačuje, že doporučovací systém je dobrý v poskytování alespoň jedné relevantní položky většině uživatelů.

$$\text{Míra zásahu} = \frac{\text{Počet uživatelů s alespoň jedním relevantním doporučením}}{\text{Celkový počet uživatelů}} \tag{4.3}$$

4.2 Výsledky vyhodnocení

K evaluaci bylo nutné určit, které položky jsou pro uživatele relevantní. V obrázku 4.1 lze vidět, že pokud uživatel ohodnotí nějaký film, je to většinou buď průměrné hodnocení (3.0) nebo výše. Pokud je položka lehce nad průměrem, dá se o ní hovořit jako o relevantní položce. Hranice při evaluaci byla tedy stanovena na hodnocení 3.5. Jakákoliv položka s větším nebo rovnym hodnocením tomuto hodnocení je považováno za relevantní položku. Lze zvolit jiný způsob výběru této hranice. Například průměrné hodnocení uživatele pro všechny filmy, nebo průměrné hodnocení daného filmu všemi uživateli.



Obrázek 4.1: Počty hodnocení pro jednotlivé stupně hodnocení

Při evaluacích tohoto systému byly upravovány tyto 3 hodnoty:

- **N** = Počet filmů, které uživatel hodnotil a jsou brány v potaz při generování doporučovaní
- **K** = Množství filmů, které je uživateli doporučeno.
- **H** = Počet nejpodobnějších filmů k zadenému filmu v systému CB

Jelikož bylo testováno velké množství možností, v následujících tabulkách je pouze 5 nejlepších konfigurací.

Tabulka 4.1 zobrazuje názvy konfigurací a jejich hodnoty pro N, K, H.

Tabulka 4.2 zobrazuje výsledky jednotlivých konfigurací pro zvolené metriky. Výsledky jsou v rozmezí $\langle 0, 1 \rangle$, kde 0 odpovídá 0 % a 1 odpovídá 100 %.

Konfigurace **Hybrid5** má nejlepší výsledky pro všechny metriky, ale hlavně má nejvyšší míru zásahu, 79,42 %. Konkrétně při doporučování filmů je důležité doporučit uživateli alespoň jeden relevantní film, na který se následně podívá a ohodnotí ho. Pokud by se chtěl

Název	N	H	K
Hybrid1	15	50	25
Hybrid2	20	25	25
Hybrid3	20	50	25
Hybrid4	25	25	25
Hybrid5	25	50	25

Tabulka 4.1: Názvy konfigurací a zvolené parametry

Název	Přesnost	Úplnost	Míra zásahu
Hybrid1	0.0698	0.0775	0.7678
Hybrid2	0.0644	0.0718	0.7489
Hybrid3	0.0737	0.0821	0.7843
Hybrid4	0.0683	0.0763	0.7663
Hybrid5	0.0767	0.0856	0.7942

Tabulka 4.2: Porovnání jednotlivých konfigurací pro ruzné počty doporučených filmů

podívat následně na další film, budou doporučení v případě kladného hodnocení zhlédnutého filmu změněny. V případě dostupnosti implicitních hodnoceních by nebylo nutné film ohodnotit explicitně, ale počítalo by se právě s implicitním hodnocením, což by mohlo být například příznak, zda film zhlédnul celý a tím pádem je pravděpodobnější, že se mu líbil.

4.3 Uživatelské testování

Testování se zůčastnilo **24** lidí. Většinou se jednalo o spolužáky nebo přátele, tudíž průměrný věk byl zhruba 24 let. Testování probíhalo v osobní formě, jelikož systém nebyl nasazený na žádném serveru, ze kterého by aplikace byla veřejně dostupná. V osobní formě jsem mohl lépe dohlédnout na testování a případně se uživatelů ptát na doplňující otázky nebo odpovídat na jejich otázky v reálném čase, což bylo v některých případech přínosné.

Testování probíhalo téměř vždy podle následující scénáře:

1. Uživatel se zaregistroval a přihlásil.
2. Ohodnotil minimálně 5 filmů, mohl však také více.
3. Nechal si vygenerovat prvotní doporučení. Poté odpověděl na otázku, kolik z doporučených filmů se mu líbí a považuje je za relevantní doporučení.
4. Následně kliknul na tlačítko `Retrain` a nechal si vygenerovat výsledná doporučení.
5. Poté mu byla položena stejné otázka jako v bodě číslo 3.

4.4 Výsledky uživatelského testování

Jelikož při prvotním doporučení filmů dochází k hledájí nejpodobnějšího uživatele, na kterém by systém již trénován, bylo předpokládáno, že výsledky budou o něco horší, než při generování doporučení po trénování. Čtyři uživatelé řekli, že je žádný z doporučených filmů nezaujal a pravděpodobně by se na žádný ani nepodívali. To tedy odpovídá míře zásahu

83.4%. Při porovnání s výsledky při evaluaci systému v sekci 4.1 jsou výsledky poměrně podobné a očekávané. Nejvíce doporučených relevantních filmů pro jednoho uživatele bylo 5. Průměrný počet byl 1-2 filmy pro uživatele.

Následně byl model znovu natrénován i na aktuálním uživateli a výsledky byly znatelně lepší. Pouze jeden uživatel neměl mezi novými doporučeními žádný film, který ho zaujal. Průměrný počet relevantní filmů byl nyní 3-4. Z tohoto pozorování vyplývá, že mírá zásahu byla až 95,74%. Přesnost byla při výpočtu s průměrným počtem relevantních filmů 12,5%, což je lepší výsledek než při evaluaci systému.

Takovéto zlepšení je poměrně překvapující, jelikož počet uživatelů v systému je poměrně velký a nemělo by tedy být náročné najít uživatele, který je hodně podobný aktuálnímu. Nicméně v opravdové aplikaci by se při rostoucím množství uživatelů tento rozdíl postupně zmenšoval, jelikož by bylo pravděpodobnější, že je v systému uživatel, který je podobný vybranému uživateli natolik, že by to na doporučení pravděpodobně nemělo skoro žádný vliv.

Kapitola 5

Závěr

V této práci byla nejdříve popsána motivace ke tvorbě doporučovacích systémů, základní modely jako je kolaborativní filtrování nebo filtrování založené na obsahu. Také byly popsány více specifické systémy, jako jsou například sociální doporučovací systémy nebo systémy citlivé na čas. Následně byly popsány problémy spojené s implementací těchto systémů. V rámci práce se podařilo úspěšně implementovat systém pro doporučování filmů. Také byla implementována aplikace, která umožňuje jednoduché otestování uživatelem. V rámci evaluace systému bylo dosaženo dostačujících výsledků, které také potvrdilo uživatelské testování. Další motivací bylo také navrhnout systém tak, aby byl škálovatelný. Pokud by byl výsledný systém na výkonějším serveru a měl by k dispozici alespoň přiměřené množství paměti, neměl by nastat problém se škálovatelností.

Je velmi náročné vytvořit vysoce kvalitní doporučovací systém, který je zároveň škálovatelný, jelikož je potřeba vyřešit mnoho problémů, jak je zmíněno v sekci 2.3. Je nutno dbát na nejen potřeby uživatelů, ale také na požadavky poskytovatelů služeb, které jsou zmíněny v sekci 2.1.

V rámci navazující práce je možné zahrnout do doporučování mnoho dalších informací, jako je například informace o tom, zda je film součástí kolekce, zemi původu nebo rozpočet pro vytvoření filmu. Jako vhodné by bylo také zakomponovat informaci o čase, kdy bylo hodnocení uděleno. Tento přístup by mohl značně vylepšit výsledná doporučení, jak tomu bylo také v případě systému od společnosti Amazon, zmíněném v sekci 2.5.2. Také je možné experimentovat s hybridním systémem a způsobem, jakým využívá jednotlivé systémy. Například generovat doporučení na základě váhovaného průměru jednotlivých systémů. Další možné vylepšení může být zakomponování dalších druhů systémů nebo se pokusit získat další informace na základě již známých informací, například získat přepis filmu do textové podoby a z tohoto přepisu extrahovat příznaky, které by dokázali lépe zařadit film do určitých kategorií.

Literatura

[1] Adomavicius, G. a Zhang, J. Iterative smoothing technique for improving stability of recommender systems. *CEUR Workshop Proceedings*. CEUR-WS. prosinec 2012, sv. 910, s. 3–8. ISSN 1613-0073. Workshop on Recommendation Utility Evaluation: Beyond RMSE, RUE 2012 - Workshop at the 6th ACM International Conference on Recommender Systems, RecSys 2012 ; Conference date: 09-09-2012 Through 09-09-2012.

[2] Adrian Holovaty, J. K.-M. a Foundation, D. S. *Django* [https://www.djangoproject.com/]. 2005. Available at https://www.djangoproject.com/.

[3] Aggarwal, C. C. et al. *Recommender systems*. Springer, 2016.

[4] Avazpour, I., Pitakrat, T., Grunske, L. a Grundy, J. Dimensions and Metrics for Evaluating Recommendation Systems. In: Robillard, M. P., Maalej, W., Walker, R. J. a Zimmermann, T., ed. *Recommendation Systems in Software Engineering*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2014, s. 245–273. DOI: 10.1007/978-3-642-45135-5_10. ISBN 978-3-642-45135-5. Dostupné z: https://doi.org/10.1007/978-3-642-45135-5_10.

[5] Banks, A. a Porcello, E. *Learning React: functional web development with React and Redux*. "O'Reilly Media, Inc.", 2017.

[6] Basilico, J. *Recent Trends in Personalization at Netflix*. Sep 2020. Presentation at the Netflix Expo session at RecSys 2020 Virtual Conference. Dostupné z: https://research.netflix.com/publication/Recent%20Trends%20in%20Personalization%20at%20Netflix.

[7] Bavota, G., De Lucia, A., Marcus, A. a Oliveto, R. Recommending Refactoring Operations in Large Software Systems. In: Robillard, M. P., Maalej, W., Walker, R. J. a Zimmermann, T., ed. *Recommendation Systems in Software Engineering*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2014, s. 387–419. DOI: 10.1007/978-3-642-45135-5_15. ISBN 978-3-642-45135-5. Dostupné z: https://doi.org/10.1007/978-3-642-45135-5_15.

[8] Burke, R. Knowledge-based recommender systems. *Encyclopedia of library and information systems*. Citeseer. 2000, sv. 69, Supplement 32, s. 175–186.

[9] Burke, R. Hybrid Recommender Systems: Survey and Experiments. *User Modeling and User-Adapted Interaction*. Nov 2002, sv. 12, č. 4, s. 331–370. DOI: 10.1023/A:1021240730564. ISSN 1573-1391. Dostupné z: https://doi.org/10.1023/A:1021240730564.

[10] Candillier, L., Chevalier, M., Dudognon, D. a Mothe, J. Diversity in Recommender Systems: Bridging the gap between users and systems (regular paper). In:. Leden 2011, s. 48–58.

[11] Covington, P., Adams, J. a Sargin, E. Deep neural networks for youtube recommendations. In: *Proceedings of the 10th ACM conference on recommender systems*. 2016, s. 191–198.

[12] Dalton, J.-D. a Contributors. *Lodash* [https://lodash.com/]. 2012. Available at https://lodash.com/.

[13] Davidson, J., Liebald, B., Liu, J., Nandy, P., Van Vleet, T. et al. The YouTube Video Recommendation System. In: *Proceedings of the Fourth ACM Conference on Recommender Systems*. New York, NY, USA: Association for Computing Machinery, 2010, s. 293–296. RecSys '10. DOI: 10.1145/1864708.1864770. ISBN 9781605589060. Dostupné z: https://doi.org/10.1145/1864708.1864770.

[14] Fette, I. a Melnikov, A. *The websocket protocol*. 2011.

[15] Fielding, R. T. *REST: Architectural Styles and the Design of Network-based Software Architectures*. 2000. Doctoral dissertation. University of California, Irvine. Dostupné z: http://www.ics.uci.edu/~fielding/pubs/dissertation/top.htm.

[16] Ford, W. Chapter 15 - The Singular Value Decomposition. In: Ford, W., ed. *Numerical Linear Algebra with Applications*. Boston: Academic Press, 2015, s. 299–320. DOI: https://doi.org/10.1016/B978-0-12-394435-1.00015-6. ISBN 978-0-12-394435-1. Dostupné z: https://www.sciencedirect.com/science/article/pii/B9780123944351000156.

[17] Gomez Uribe, C. A. a Hunt, N. The Netflix Recommender System: Algorithms, Business Value, and Innovation. *ACM Trans. Manage. Inf. Syst.* New York, NY, USA: Association for Computing Machinery. dec 2016, sv. 6, č. 4. DOI: 10.1145/2843948. ISSN 2158-656X. Dostupné z: https://doi.org/10.1145/2843948.

[18] Hansjons Vegeborn, V. a Rahmani, H. *Comparison and Improvement Of Collaborative Filtering Algorithms*. 2017.

[19] Hardesty, L. The history of Amazon's recommendation algorithm. *Amazon Science*. 2019, sv. 22.

[20] Hejlsberg, A. a Microsoft. *TypeScript* [https://www.typescriptlang.org/]. 2012. Available at https://www.typescriptlang.org/.

[21] Herlocker, J. L., Konstan, J. A., Terveen, L. G. a Riedl, J. T. Evaluating Collaborative Filtering Recommender Systems. *ACM Trans. Inf. Syst.* New York, NY, USA: Association for Computing Machinery. jan 2004, sv. 22, č. 1, s. 5–53. DOI: 10.1145/963770.963772. ISSN 1046-8188. Dostupné z: https://doi.org/10.1145/963770.963772.

[22] Hug, N. Surprise: A Python library for recommender systems. *Journal of Open Source Software*. The Open Journal. 2020, sv. 5, č. 52, s. 2174. DOI: 10.21105/joss.02174. Dostupné z: https://doi.org/10.21105/joss.02174.

[23] Isinkaye, F., Folajimi, Y. a Ojokoh, B. Recommendation systems: Principles, methods and evaluation. *Egyptian Informatics Journal*. 2015, sv. 16, č. 3, s. 261–273. DOI: https://doi.org/10.1016/j.eij.2015.06.005. ISSN 1110-8665. Dostupné z: https://www.sciencedirect.com/science/article/pii/S1110866515000341.

[24] Joita, R. a Contributors. *DRF-YASG - Yet Another Swagger Generator* [https://github.com/axnsan12/drf-yasg]. GitHub, 2017.

[25] Koren, Y., Bell, R. a Volinsky, C. Matrix Factorization Techniques for Recommender Systems. *Computer*. 2009, sv. 42, č. 8, s. 30–37. DOI: 10.1109/MC.2009.263.

[26] Lampropoulos, A. S., Lampropoulou, P. S. a Tsihrintzis, G. A. A Cascade-Hybrid Music Recommender System for mobile services based on musical genre classification and personality diagnosis. *Multimedia Tools and Applications*. Jul 2012, sv. 59, č. 1, s. 241–258. DOI: 10.1007/s11042-011-0742-0. ISSN 1573-7721. Dostupné z: https://doi.org/10.1007/s11042-011-0742-0.

[27] Lee, J., Sun, M. a Lebanon, G. A Comparative Study of Collaborative Filtering Algorithms. *CoRR*. 2012, abs/1205.3193. Dostupné z: http://arxiv.org/abs/1205.3193.

[28] Linden, G., Smith, B. a York, J. Amazon.com recommendations: item-to-item collaborative filtering. *IEEE Internet Computing*. 2003, sv. 7, č. 1, s. 76–80. DOI: 10.1109/MIC.2003.1167344.

[29] Linsley, T. a Contributors. *React Query* [https://react-query.tanstack.com/]. 2019. Available at https://react-query.tanstack.com/.

[30] Lü, L., Medo, M., Yeung, C. H., Zhang, Y.-C., Zhang, Z.-K. et al. Recommender systems. *Physics Reports*. 2012, sv. 519, č. 1, s. 1–49. DOI: https://doi.org/10.1016/j.physrep.2012.02.006. ISSN 0370-1573. Recommender Systems. Dostupné z: https://www.sciencedirect.com/science/article/pii/S0370157312000828.

[31] Ma, C.-C. A guide to singular value decomposition for collaborative filtering. *Computer (Long Beach, CA)*. Citeseer. 2008, sv. 2008, s. 1–14.

[32] Ma, H., Zhou, T. C., Lyu, M. R. a King, I. Improving Recommender Systems by Incorporating Social Contextual Information. *ACM Trans. Inf. Syst.* New York, NY, USA: Association for Computing Machinery. apr 2011, sv. 29, č. 2. DOI: 10.1145/1961209.1961212. ISSN 1046-8188. Dostupné z: https://doi.org/10.1145/1961209.1961212.

[33] Massa, P. a Avesani, P. Trust-Aware Recommender Systems. In: *Proceedings of the 2007 ACM Conference on Recommender Systems*. New York, NY, USA: Association for Computing Machinery, 2007, s. 17–24. RecSys '07. DOI: 10.1145/1297231.1297235. ISBN 9781595937308. Dostupné z: https://doi.org/10.1145/1297231.1297235.

[34] McKinney, W. a Development Team pandas. *Pandas* [https://pandas.pydata.org/]. 2008. Available at https://pandas.pydata.org/.

[35] McNee, S. M., Riedl, J. a Konstan, J. A. Being Accurate is Not Enough: How Accuracy Metrics Have Hurt Recommender Systems. In: *CHI '06 Extended Abstracts on Human Factors in Computing Systems*. New York, NY, USA: Association for Computing Machinery, 2006, s. 1097–1101. CHI EA '06. DOI: 10.1145/1125451.1125659. ISBN 1595932984. Dostupné z: https://doi.org/10.1145/1125451.1125659.

[36] Mobasher, B., Burke, R., Bhaumik, R. a Williams, C. Toward Trustworthy Recommender Systems: An Analysis of Attack Models and Algorithm Robustness. *ACM Trans. Internet Technol.* New York, NY, USA: Association for Computing Machinery. oct 2007, sv. 7, č. 4, s. 23–es. DOI: 10.1145/1278366.1278372. ISSN 1533-5399. Dostupné z: https://doi.org/10.1145/1278366.1278372.

[37] Musto, C., Gemmis, M. d., Lops, P., Narducci, F. a Semeraro, G. Semantics and Content-Based Recommendations. In: Ricci, F., Rokach, L. a Shapira, B., ed. *Recommender Systems Handbook*. New York, NY: Springer US, 2022, s. 251–298. DOI: 10.1007/978-1-0716-2197-4_7. ISBN 978-1-0716-2197-4. Dostupné z: https://doi.org/10.1007/978-1-0716-2197-4_7.

[38] Oliphant, T. E. a Developers, N. *NumPy* [https://numpy.org/]. 2006. Available at https://numpy.org/.

[39] O'Mahony, M., Hurley, N., Kushmerick, N. a Silvestre, G. Collaborative Recommendation: A Robustness Analysis. *ACM Trans. Internet Technol.* New York, NY, USA: Association for Computing Machinery. nov 2004, sv. 4, č. 4, s. 344–377. DOI: 10.1145/1031114.1031116. ISSN 1533-5399. Dostupné z: https://doi.org/10.1145/1031114.1031116.

[40] Ozok, A., Fan, Q. a Norcio, A. Design guidelines for effective recommender system interfaces based on a usability criteria conceptual model: Results from a college student population. *Behaviour & IT*. Leden 2010, sv. 29, s. 57–83. DOI: 10.1080/01449290903004012.

[41] Pedregosa, F. a Developers, S. learn. *Scikit-learn* [https://scikit-learn.org/]. 2007. Available at https://scikit-learn.org/.

[42] Phorasim, P. a Yu, L. Movies recommendation system using collaborative filtering and k-means. *International Journal of Advanced Computer Research*. Accent Social and Welfare Society. 2017, sv. 7, č. 29, s. 52.

[43] Prasad, R. a Kumari, V. V. A categorical review of recommender systems. *International Journal of Distributed and Parallel Systems*. Academy & Industry Research Collaboration Center (AIRCC). 2012, sv. 3, č. 5, s. 73.

[44] Ricci, F., Rokach, L. a Shapira, B. Introduction to Recommender Systems Handbook. In: Ricci, F., Rokach, L., Shapira, B. a Kantor, P. B., ed. *Recommender Systems Handbook*. Boston, MA: Springer US, 2011, s. 1–35. DOI: 10.1007/978-0-387-85820-3_1. ISBN 978-0-387-85820-3. Dostupné z: https://doi.org/10.1007/978-0-387-85820-3_1.

[45] Ricci, F., Rokach, L. a Shapira, B. Recommender Systems: Techniques, Applications, and Challenges. In: Ricci, F., Rokach, L. a Shapira, B., ed. *Recommender Systems Handbook*. New York, NY: Springer US, 2022, s. 1–35. DOI: 10.1007/978-1-0716-2197-4_1. ISBN 978-1-0716-2197-4. Dostupné z: https://doi.org/10.1007/978-1-0716-2197-4_1.

[46] Rosario, B. Latent semantic indexing: An overview. *Techn. rep. INFOSYS*. 2000, sv. 240, s. 1–16.

[47] Rossum, G. van a Foundation, P. S. *Python* [https://www.python.org/]. 1991. Available at https://www.python.org/.

[48] Sarwar, B., Karypis, G., Konstan, J. a Riedl, J. Item-based collaborative filtering recommendation algorithms. In: *Proceedings of the 10th International Conference on World Wide Web, WWW 2001*. Association for Computing Machinery, Inc, Duben 2001, s. 285–295. WWW '01. DOI: 10.1145/371920.372071. ISBN 1581133480. 10th International Conference on World Wide Web, WWW 2001 ; Conference date: 01-05-2001 Through 05-05-2001.

[49] Singh, R. H., Maurya, S., Tripathi, T., Narula, T. a Srivastav, G. Movie recommendation system using cosine similarity and KNN. *International Journal of Engineering and Advanced Technology*. 2020, sv. 9, č. 5, s. 556–559.

[50] Smyth, B. a McClave, P. Similarity vs. Diversity. In: Aha, D. W. a Watson, I., ed. *Case-Based Reasoning Research and Development*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2001, s. 347–361. ISBN 978-3-540-44593-7.

[51] Sondur, M. S. D., Chigadani, M. A. P. a Nayak, S. Similarity measures for recommender systems: a comparative study. *Journal for Research*. 2016, sv. 2, č. 3.

[52] Su, X. a Khoshgoftaar, T. M. A survey of collaborative filtering techniques. *Advances in artificial intelligence*. Hindawi. 2009, sv. 2009.

[53] Team, M.-U. *Material-UI* [https://material-ui.com/]. 2014. Available at https://material-ui.com/.

[54] Tran, T. a Cohen, R. Hybrid recommender systems for electronic commerce. In: *Proc. Knowledge-Based Electronic Markets, Papers from the AAAI Workshop, Technical Report WS-00-04, AAAI Press*. 2000, sv. 40.

[55] Vall, A., Dorfer, M., Eghbal zadeh, H., Schedl, M., Burjorjee, K. et al. Feature-combination hybrid recommender systems for automated music playlist continuation. *User Modeling and User-Adapted Interaction*. Apr 2019, sv. 29, č. 2, s. 527–572. DOI: 10.1007/s11257-018-9215-8. ISSN 1573-1391. Dostupné z: https://doi.org/10.1007/s11257-018-9215-8.

[56] Yao, Y. Measuring Retrieval Effectiveness Based on User Preference of Documents. *J. Am. Soc. Inf. Sci.* 1995, sv. 46, s. 133–145.

[57] Yochum, P., Chang, L., Gu, T. a Zhu, M. Linked Open Data in Location-Based Recommendation System on Tourism Domain: A Survey. *IEEE Access*. 2020, sv. 8, s. 16409–16439. DOI: 10.1109/ACCESS.2020.2967120.

Přílohy / Appendices

Příloha A

Obsah přiloženého paměťového média

- `archive/` – Datová sada.
- `src/app/` – Aplikace pro uživatelské testování.
- `src/rs/` – Python nooteboky obsahující doporučovací systém, jeho testování a také vizualizaci dat. Jsou zde také upravená data a natrénovaný model pro CF systém.
- `README.md` – Obsahuje pokyny ke spuštění aplikace.
- `doc/` – Zdrojové soubory pro vygenerování technické zprávy a také samotná technická zpráva.