UNIVERZITET U BEOGRADU
MATEMATIČKI FAKULTET

Jovana D. Pejkić

METODA UZAJAMNOG FILTRIRANJA I PRIMENE U ELEKTRONSKOJ TRGOVINI

master rad

Beograd, 2023.

Mentor:

dr Aleksandar Kartelj, docent
Univerzitet u Beogradu, Matematički fakultet

Članovi komisije:

dr Nenad Mitić, redovan profesor
Univerzitet u Beogradu, Matematički fakultet

dr Jelena Graovac, docent
Univerzitet u Beogradu, Matematički fakultet

Datum odbrane: _____

Posvećeno porodici

Naslov master rada: Metoda uzajamnog filtriranja i primene u elektronskoj trgovini

Rezime:

Elektronska trgovina postaje sve zastupljeniji vid kupovine proizvoda i usluga. Preplavljenost informacijama predstavlja izazovan i veoma zastupljen problem u komercijalnom svetu. Rukovanje ogromnom količinom podataka uz korišćenje postojećih alata postalo je nemoguće, te se javlja potreba za naprednijim pristupima u pretraživanju i filtriranju informacija. Mnogi veb sajtovi za e-trgovinu već uveliko koriste alate kao što su sistemi preporuka, koji pomažu kupcima da u moru proizvoda pronađu one koje žele. Glavni ishod dobrog sistema preporuke je povećanje lojalnosti kupaca, a time i povećanje zarade.

U ovom radu opisano je više različitih implementacija ovih sistema koji rade nad skupom podataka koji sadrži informacije o proizvodima koji se prodaju na Amazonu. Najpre je izvršena priprema i analiza odabranog skupa podataka, a zatim odvajanje bitnih atributa i njihovo enkodiranje. Nakon toga, predstavljene su implementacije modela, njihovo testiranje i tumačenje rezultata. Cilj razvijanih pristupa je predviđanje ocene korisnika za određeni proizvod i generisanje preporuka. Korišćene tehnike su prethodno teorijski opisane. Podaci na osnovu kojih modeli vrše predviđanja preuzeti su iz javno dostupnih baza podataka.

Ključne reči: e-trgovina, sistem preporuke, uzajamno filtriranje, predviđanje

Sadržaj

Glava 1

O radu

1.1 Opis problema

Sve do pojave interneta, pretraživanje robe za kupovinu se zasnivalo na sopstvenom ukusu ili na preporuci osobe od poverenja. Uglavnom je proces kupovine tekao tako što se pregleda sva roba u radnji ili tržnom centru i klasifikuju stvari na: „ovo mi se sviđa", „ovo mi se ne sviđa". Na kraju se izlazi sa jednim ili više artikala, koji odnosno koje, vredi kupiti [6].

Ipak, u poslednjih par decenija, sa razvojem interneta, informacije rastu eksponencijalno, te razmatranje svega ponuđenog postaje nemoguće. Tokom ranih godina interneta, dok se kupovina onlajn kretala sporim tempom, bilo je manje veb lokacija sa onlajn prodavnicama i njihov izbor proizvoda nije bio veliki [6].

Stvari su se značajno promenile sa početkom Veb 2.0 oko 2004. godine. Tada su se i prodavnice i kupci podjednako „preselili" na internet, a radikalne promene grafičkog korisničkog interfejsa veb stranica, posebno onih koje su koristile AJAKS tehnologiju, omogućile su mnogo bogatije i impresivnije iskustvo pregledanja. Od tada pa nadalje, protok informacija je počeo značajno da se uvećava, da je čak i fizički sloj mreže u početku imao problema sa obezbeđivanjem tolike količine sadržaja [6].

Ubrzo, ograničenje brzine koje su nametnuli rani modemi ustupilo je mesto brzom ADSL-u, i od tog trenutka pa nadalje, veb stranice su se nadmetale jedna sa drugom u prikazivanju više grafičkog sadržaja, više slika i više teksta. Informaciono doba je rođeno, a preopterećenost informacijama postala je uobičajena pojava [6].

Iako e-trgovina nije nužno omogućila preduzećima da proizvode više proizvoda, omogućila im je da kupcima pruže veći izbor. Tako, umesto 10000 knjiga u prodavnici, kupci mogu da biraju među milionima knjiga u nekoj onlajn prodavnici.

Ograničenje u broju potencijalnih stavki koje se mogu pogledati na vebu praktično ne postoji. Uvećanjem izbora, povećana je količina informacija koju korisnici moraju da procesuiraju pre nego što budu u mogućnosti da odaberu koji proizvod zadovoljava njihove potrebe. Iako se mogućnost da se potrošačima pruži veći izbor smatra prednošću, to za sobom povlači nove probleme. Na primer, javljaju se sledeća pitanja: Kako bi onlajn kupac trebalo da filtrira ogromnu količinu opcija koje se sada nude? Zadatak je postao sličan pronalaženju igle u plastu sena. Korišcenje staromodne pre-internet metode verovanja sopstvenom ukusu ili ukusu bliskog prijatelja više ne pomaže. Vidljivost informacija je znatno smanjena. Kako otkriti nove skrivene stvari? Nove proizvode? Nove ideje? Kako izabrati između bukvalno milion artikala, onaj koji se traži? Pretražiti termin i za rezultat očekivati razumnu kolicinu proizvoda više nije realno očekivanje. Pretraživači su dobri u vraćanju podudaranja ključnih reči, ali rezultati nisu prilagođeni korisniku. Proces obrađivanja datih informacija i donošenje odluke šta kupiti postao je mukotrpan.

1.2 Motivacija

Onlajn prodavnice su se prilično rano suočile sa problemom ogromne količine informacija, shvatajući da ce ih vremenom biti sve više i da će kupcima biti potrebna pomoć da pronađu artikle koje žele. Da bi rešile ovaj problem, e-prodavnice primenjuju različite principe, tehnike i alate kako bi potrošačima pružile što bolje korisničko iskustvo i preporuku o tome šta bi mogli sledeće da pregledaju[1].

Osnovna ideja koja stoji iza ovih algoritama je prilično jednostavna: analiza korisnikovih podataka, njegovih kupovina, ocena i odnosa sa drugim korisnicima može poslužiti kao osnov prema kome se korisniku mogu ponuditi drugi artikli koji će mu se potencijalno dopasti. Takođe, korisniku se nude i informacije koje mogu da mu pomognu da odluči koje proizvode da kupi [1].

Porodica opisanih algoritama nazvana je **sistemi preporuka (eng. *Recommender Systems, RS*)**. Ovi sistemi su razvijani kako bi se izborili sa eksponencijalnim rastom podataka u doba interneta. Ubrzo, sistemi preporuka su prešli od istraživačkih projekata do potpunog uključivanja u komercijalne svrhe. Cilj sistema preporuka je da nauči preference ciljnog korisnika, sa namerom da mu preporuči relevantne, ne još viđene artikle[1].

Upotreba sistema preporuka je postala značajan i neizostavan deo e-industrije. Onlajn prodavnice koje inkorporiraju ove sisteme postaju sve posećenije u odnosu

na one tradicionalne. Sistemi preporuka na sajtovima za elektronsku trgovinu su jedan sve više prisutan biznis alat koji dovodi do povećanja zarade. Istraživači kao i vlasnici sajtova za e-trgovinu traže sve moguće načine kako da unaprede performanse preporučivanja. Dobar sistem preporuke zasigurno uvećava biznis tih sajtova, jer korisnici mogu da naprave izbor bez iscrpne pretrage, što doprinosi bržim novčanim transakcijama. Danas je teško pronaći onlajn prodavnicu ili onlajn zajednicu koja ne koristi ove sisteme u pozadini, učeći preference korisnika i povezivajući ih sa listom dostupnih proizvoda. To je razlog zašto sistemi preporuka imaju značajnu ulogu na sajtovima kao što su Amazon, Facebook, YouTube i mnogi drugi [1][4].

Glava 2

Uvod u sisteme preporuka

2.1 Uvod

Sistem preporuke predstavlja softverski alat koji koristi skup tehnika i algoritama za davanje preporuka koje će najverovatnije interesovati ciljnog korisnika [1]. Sistemi preporuka su sastavni deo raznih aplikacija, gde pokušavaju da korisnicima pruže tačnu i pouzdanu preporuku tako da zadovolje njihove potrebe i doprinesu razvoju biznisa kompanija. Sistemi preporuka imaju značajnu ulogu u procesima koji se tiču donošenja odluka (eng. *decision-making processes*), kao što su to: koje proizvode kupiti, koju muziku slušati, ili koju knjigu pročitati [1].

Stavka (eng. *Item*) je opšti termin koji se koristi da označi sta sistem preporučuje korisnicima. Jedan sistem preporuka se obično specijalizuje za određene stavke (na primer isključivo za vesti ili muziku), i koristi tehnike da generiše preporuke koje su prilagođene tako da pružaju korisne i efektivne sugestije samo za tu spečificnu vrstu stavki [1].

Sistemi preporuka su primarno namenjeni individuama kojima nedostaje adekvatno znanje ili iskustvo da bi procenili i ocenili potencijalno ogroman broj alternativa koje veb sajt može da nudi. Osim što uz sistem preporuka, kupci mogu lakše da pronađu artikle koji im se sviđaju, sistem kupcima predstavlja i artikle o kojima nisu ni razmišljali, ali koji zapravo odgovaraju njihovim potrebama. Zbog ove prednosti, sistemi preporuka mogu biti i značajniji od polja za pretragu, jer korisnici tako mogu naći proizvode koje ne bi našli ni na koji drugi način [1].

Preporučivanje proizvoda može biti zasnovano na sveukupno najprodavanijim proizvodima na sajtu, na demografiji potrošača ili na analizi kupovine potrošača u prošlosti kako bi se predvidelo njegovo ponašanje u budućnosti [1][4].

Na primer, na veb sajtu Amazon.com, sistem preporuke prilagođava onlajn prodavnicu svakom kupcu posebno, odnosno individualno. Naravno, u virtuelnom svetu, sve što se menja jeste odabir proizvoda koji se prikazuju kupcu, a ne i fizička radnja koja stoji iza toga. Ovakve preporuke zovu se personalizovane (eng. *personalized*)[1].

Kod njih, sistemi preporuka prikupljaju informacije od korisnika vezano za njegove preference i u skladu sa tim, oni nude personalizovane preporuke koje su najčešće predstavljene u vidu rangiranih listi (eng. *ranked lists*) stavki. Personalizovani sistemi preporuka se sastoje iz više delova koji međusobno interaguju, a to su: metode za procesiranje podataka, modeli korisnika, tehnike filtriranja i razne metrike. Osim ovih, postoje i ne-personalizovane preporuke (eng. *non-personalized*). One su mnogo lakše da se generisu. Tipični primeri uključuju „deset najboljih knjiga", „tri najgledanija filma" i slično [1].

Preference korisnika mogu biti izražene eksplicitno ili implicitno. Najmerodavnija je eksplicitna povratna informacija (eng. *feedback*), gde korisnici direktno iskazuju interesovanje za neki proizvod. Na primer, Netflix prikuplja ocene u vidu zvezdica (eng. *star ratings*) za filmove, dok TiVo korisnici iskazuju svoje preference za TV emisiju tako sto biraju *thumbs-up* ili *thumbs-down* dugme. S obzirom na to da eksplicitna povratna informacija nije uvek dostupna, neki sistemi preporuka donose zaključke o preferencama korisnika preko obilnije implicitne povratne informacije i to kroz posmatranje ponašanja korisnika. Tipovi implicitne povratne informacije uključuju: istoriju kupovine, istoriju pregledanja, šablone pretrazivanja (eng. *search patterns*), ili čak kretanje miša. Na primer, iz prethodnih kupovina korisnika, u kojima je on kupio mnogo knjiga od istog autora, može se zaključiti da verovatno voli tog autora i da bi kupio još njegovih knjiga ukoliko mu se preporuče [1].

U ovom radu fokus je na modelima prilagođenim za rad sa eksplicitnim povratnim informacijama. Ipak, time se ne umanjuje značaj implicitnih povratnih informacija, koje su od posebne važnosti onda kada korisnik ne pruža dovoljno eksplicitnih informacija [1].

Generalno, algoritmi razvijeni za sisteme preporuka se oslanjaju na kupovine i preglede stranica koje su se prethodno desile. Na primer, ukoliko je korisnik pogledao neku kameru na veb sajtu, sistem može da nauči (odnosno zaključi) da je korisnik zainteresovan za kamere i onda mu preporuči još sličnih proizvoda. Međutim, to ne mora da se poklapa sa korisnikovom pravom namerom. Moć sistema preporuka je veća ukoliko on ima više informacija od korisnika - informacije tokom jedne sesije i informacije tokom više sesija. Da bi se rešio ovaj prolem, koriste se *session-aware* si-

stemi preporuka. Ovakvi sistemi mogu da razumeju korisnikov kratkoročni cilj (eng. *short-term goal*) i korisnikove dugoročne preference (eng. *long-term preference*), u cilju da se prema tome preporuče odgovarajući artikli [1].

Skorašnja istraživanja primenjuju nove metode za kreiranje sistema preporuka zasnovane na dubokom učenju i time upotpunjuju ciljeve veštačke inteligencije. Osim mašinskog učenja i statistike, razvoj ovih sistema uključuje i tehnike za analiziranje sadržaja artikla (eng. *Natural Language Processing*)[5].

Uspeh sistema preporuka je danas evidentan. To se može videti i kroz jedno od Guglovih najbitnijih poboljsanja njihovog pregledača (eng. *search engine*) - praćenje istorije pregledanja (eng. *browsing history tracking*). Naime, naučene korisnikove navike pri pretraživanju su uspešno iskorišćene za preporučivanje reklama za potencijalnu posetu nekom veb sajtu [1].

Mnoga istraživanja, kako akademska tako i u industriji, postignuta su na polju razvoja sistema preporuka. I dok je oblast sada stara već 20 godina, razvijene tehnike ostaju jednostavne. Pored raznovrsnih algoritama koji su razvijeni, ističu se dve opšte kategorije. Jedna od njih se odnosi na preference korisnika i poznata je kao uzajamno filtriranje (eng. *Collaborative Filtering*). A druga se odnosi na specifikacije stavki i poznata je kao filtriranje zasnovano na sadržaju (eng. *Content Based*) [1] [6]. Obe kategorije su detaljno obrađene u nastavku rada.

2.2 Istorijski pregled

Jedan od prvih istrazivačkih projekata o sistemima preporuka bio je GroupLens iz 1994., za filtriranje vesti, koji pomaže korisnicima da pronađu željene artikle u gomili dostupnih artikala. Ovo je bio prvi sistem preporuke koji je koristio uzajamno filtriranje. Njihova platforma je pružala personalizovane predikcije ocena za UseNet vesti, koristeći susede ciljnog korisnika kao strategiju, kombinovano sa Pirsonovom korelacijom kao funkcijom sličnosti, kako bi dodali težine na ocene suseda i srednju ocenu suseda i time izračunali konačnu predikciju [6].

Nakon GroupLens, nastao je Ringo sistem preporuke za muziku iz 1995. i Bellcore sistem preporuke za video iz iste godine. Ringo je koristio sličnu strategiju kao GroupLens, ali je ograničio Pirsonovu korelaciju (funkciju sličnosti). Takođe su ograničili i inkluziju suseda, pa umesto razmatranja svih suseda kao što je to urađeno kod GroupLens, razmatrani su samo oni iznad određenog praga [6].

Ista istrazivačka grupa kao za GroupLens je proizvela MovieLens 1997., čiji je

cilj bio da korisnicima preporuči odgovarajuće filmove, na osnovu preferenci drugih korisnika (zajednice) ljubitelja filmova [6].

Prva komparativna analiza algoritama zasnovanih na susedstvu koji koriste uzajamno filtriranje izvršena je 1998. U tom radu, autori su suprotstavili rezultate dobijene korišćenjem Pirsonove korelacije i kosinusne sličnosti, pronalazeći da je prvi rezultat bolji od drugog [6].

Veliki pomak za sisteme preporuka desio se kada je ova tehnologija usvojena od strane Amazona 1990-ih. Njihova implementacija se zasnivala na sličnostima proizvoda pre nego na sličnostima korisnika. Ovo je omogućilo kompaniji da plasira tvrdnje kao što su: „Korisnici koji su kupili ovaj artikal takođe su kupili i ove artikle." [6].

Još jedan značajan projekat poznat kao The Music Genome Project pokrenut od strane Pandora.com 2000. godine, sastojao se od sistema preporuke koji je učio sličnosti između muzičkih žanrova umetnika i pesama i prema tome korisniku preporučivao melodije (muziku) koja se poklapa sa njegovim ukusom i preferencama. Iste godine nastao je Netflix.com koji je pružao iznajmljivanje filmova onlajn i nudio preporuke filmova sa svojim *in-house* sistemom preporuke zvanim Cinematch. Nekoliko godina kasnije, 2006., Netflix je obećao nagradu od milion dolara za unapređenje postojećeg sistema preporuka za 10% ili više, gde je za merenje korišćen koren srednje kvadratne greške. Nagrađen je bio tim programera tri godina kasnije. Važan doprinos takmičenja bilo je uvođenje pristupa zasnovanog na modelu (eng. *Model Based approaches*) kod korišćenja tehnika uzajamnog filtriranja [6].

Glava 3

Osnovni pojmovi sistema preporuka

3.1 Vrste sistema preporuka

Postoji više načina filtriranja ulaznih podataka, pa samim time i više vrsta sistema preporuka. Neke od najzastupljenijih tehnika koje ovi sistemi implementiraju su:

- Filtriranje zasnovano na sadržaju (eng. *Content-based filtering*): Preporuke su zasnovane na prethodnim odabirima korisnika. Na primer, korisniku bi bila preporučena nova knjiga iz programiranja ukoliko je on prošle godine kupio mnogo knjiga na ovu temu.

- Uzajamno filtriranje (eng. *Collaborative filtering*): Preporuke za svakog korisnika (ciljnog korisnika) se izračunavaju uzimajući u obzir preference drugih korisnika koji su ocenili proizvode slično kao ciljni korisnik.

- Filtriranje zasnovano na demografiji (eng. *Demographic filtering*): Preporuke se generišu na osnovu demografskih karakteristika korisnika. Demografski parametri uključuju godine, pol, nacionalnost i slično.

- Hibridno filtriranje (eng. *Hybrid filtering*): Preporuke se kreiraju kombinacijom prethodnih tipova filtriranja. Na primer, upotrebom filtriranja zasnovanog na sadržaju ili uzajamnog filtriranja i demografskog filtriranja.

U nastavku su detaljno opisane tehnike filtriranja zasnovane na sadržaju, uzajamnom i hibridnom filtriranju. U sekciji 3.1 je opisano filtriranje zasnovano na sadržaju, u sekciji 3.1 uzajamno filtriranje i u sekciji 3.1 hibridno filtriranje.

Preporuke zasnovane na sadržaju

Sistem preporuke koji koristi pristup zasnovan na sadržaju, korisniku preporučuje proizvode na osnovu njegovih pretohdnih ocena i karakteristika proizvoda. Proizvod koji ima karakteristike slične proizvodima kojima je korisnik davao visoke ocene su oni koji bivaju preporučeni. Sadržaj ovde referiše na karakteristike ili atribute proizvoda. Ideja kod ovog pristupa je da uz svaki proizvod stoji i njegov opis ili ključne reči. Opis doprinosi tome da se razume kakve proizvode ciljni korisnik voli, ali i da se pronađu drugačiji proizvodi sa sličnim karakteristikama [1] [6].

Pristup zasnovan na sadržaju zahteva određenu količinu informacija u vezi karakteristika proizvoda, jer je to ono na šta se oslanja prilikom generisanja preporuka (nije zasnovan na interakcijama korisnika). Na primer, za sistem preporuke koji preporučuje filmove, za jednu stavku (film) relevantne informacije (karakteristike) bi bile: žanr, godina, direktor, glumci i slično. Prema [5], ceo proces preporučivanja zasnovanog na sadržaju se može podeliti na tri osnovna koncepta:

1. **Pretprocesiranje (eng. *Preprocessing*) i ekstraktovanje karakteristika (eng. *Feature extraction*)**: Cilj je transformisati skup podataka koji predstavljaju opise stavki na vektorski prostor zasnovan na ključnim rečima (eng. *keyword-based vector space*) - terminima, a potom odrediti njihovu važnost. Ovo se postiže upotrebom modela „Vreća reči" (eng. *Bag of Words model*), gde je svaki *dokument*[1] predstavljen kao „vreća" koja sadrži termine bez ikakvog redosleda [17]. Pre ovog koraka potrebno je očistiti podatke - eliminisati stop-reči (eng. *stop-word removing*), primeniti stemovanje (eng. *stemming*) i lematizaciju (eng. *lemmatization*) - dva pristupa za smanjenje varijabiliteta termina izvučenih iz nekog teksta (kroz svođenje termina na njihov osnovni/koreni oblik), kao i ekstraktovanje fraza. Koncepti koji se ovde takođe provlače, a koriste se da odrede značaj, odnosno važnost kjucnih reči (termina) u dokumentu su:

    - Učestalost termina (eng. *Term Frequency, TF*) : Predstavlja broj pojavljivanja datog termina u razmatranom dokumentu. Ideja je da što se termin češće pojavljuje u dokumentu, to je značajniji za taj dokument. Formula za računanje TF data je u nastavku.

$$TF(\tau) = \frac{BrojPojavljivanjaTermina\tau UDokumentu}{UkupanBrojTerminaUDokumentu}$$

[1] Dokument je skup termina.

- Inverzna učestalost dokumenta (eng. *Inverse Document Frequency, IDF*): Koristi se da odredi relativnu važnost dokumenta. Ideja je dodeliti veće težine neuobičajenim terminima tj. onima koji nisu toliko prisutni u korpusu[2]. IDF se određuje na osnovu kompletnog koprusa i opisuje korpus kao celinu, a ne pojedinačne dokumente. U formuli, u imeniocu se dodaje $+1$, kako bi se sprečilo deljenje nulom, do čega dolazi kada je *Broj Dokumenata Koji Sadrže Termin* τ jednak 0. Formula za računanje IDF data je u nastavku.

$$IDF(\tau) = \log(\frac{UkupanBrojDokumenata}{BrojDokumenataKojiSadreTermin\tau + 1})$$

- $TF^{\smile}IDF$ skor: Svaka reč ili termin ima svoj TF i IDF skor. Proizvod TF i IDF skorova termina predstavlja $TF*IDF$ težinu termina. Ideja je vrednovati one termine koji nisu uobičajeni u korpusu (imaju visok IDF), a pri tome imaju nezanemarljiv broj pojavljivanja u datom dokumentu (imaju visok TF). Što je veći $TF*IDF$ skor, termin je ređi i značajniji, i obrnuto. Formula za računanje $TF-IDF$ skora data je u nastavku.

$$TF-IDF(\tau) = TF(\tau) * IDF(\tau)$$

2. **Učenje profila korisnika zasnovano na sadržaju**: U ovom koraku koriste se povratne informacije korisnika (eksplicitne ili implicitne) u kombinaciji sa informacijama o opisima stavki, da bi se konstruisao skup podataka za treniranje. Nad ovim trening podacima, konstruiše se profil korisnika koji predstavlja preference svakog korisnika.

3. **Filtriranje i preporučivanje**: Naučeni model iz prethodnog koraka prima ulazne podatke i generiše listu preporuka za svakog korisnika.

Prednosti i mane filtriranja zasnovanog na sadržaju

Filtriranje zasnovano na sadržaju, za razliku od uzajamnog filtriranja, nije efikasno za velike skupove podataka. Međutim, za razliku od uzajamnog filtriranja, ovaj metod prevazilazi problem „nove stavke" jer i nove stavke sadrže opis. Takođe, sistem zasnovan na sadržaju je nezavisan od korisnika (eng. *user independent*), jer ne koristi ocene drugih korisnika [1] [6].

[2]Korpus je skup dokumenata.

Preporuke zasnovane na uzajamnom filtriranju

Sistemi zasnovani na uzajamnom filtriranju prikupljaju informacije o interakcijama korisnik-stavka, kao što je to implicitno ili eksplicitno ocenjivanje, i to čuvaju u formi matrice ocena. Na osnovu tih ocena, ovi sistemi nastoje da predvide korisnost pojedinog proizvoda za određenog korisnika. To postižu na osnovu vrednovanja tog proizvoda od strane ostalih korisnika sistema [1] [6].

Mnogi veb sajtovi, a naročito sajtovi sa elektronsku trgovinu, koriste tehnike uzajamnog filtriranja u svojim sistemima preporuka da personalizuju korisničko iskustvo pregledanja. Tako je Amazon, koristeći ove tehnike povećao proizvodnju za 29%, Netflix povećao iznajmljivanje filmova za 60% i Google vesti su povećali broj klikova (eng. *click-through rates*) za 30.9% [10].

Kod sistema zasnovanih na uzajamnom filtriranju, ocene koje su korisnici dodelili proizvodima koriste se kao relativna prezentacija njihovih interesa i potreba. Za razliku od preporuka na osnovu sadržaja, ovi modeli ne sadrže podatke o proizvodima već se ocene dodeljene od strane ciljnog korisnika upoređuju sa ocenama koje su dodelili drugi korisnici sistema, pa se na osnovu toga određuje najbliži skup proizvoda od interesa. Glavna podela algoritama za uzajamno filtriranje je na [1] [6]:

- Algoritme zasnovane na memoriji (eng. *Memory-based algorithms*)
- Algoritme zasnovane na modelu (eng. *Model-based algorithms*)

Kod algoritama zasnovanih na memoriji ciljnom korisniku će među proizvodima koje još nije ocenio biti preporučeni oni koje su visoko ocenili njemu slični korisnici. Osim donošenja odluka razmatranjem sličnih korisnika, moguće je razmatrati i slične stavke. Za određivanje sličnosti između korisnika, odnosno stavki, mogu se koristiti neke od mera sličnosti iz sekcije 3.2. Za predviđanje ocena mogu se koristiti funkcije za predviđanje opisane u sekciji 3.3. Algoritmi zasnovani na memoriji opisani su u sekciji 3.1.

Kod algoritama zasnovanih na modelu koriste se razni algoritmi mašinskog učenja kako bi se predvidele ocene korisnika za neocenjene stavke. Algoritmi zasnovani na modelu opisani su u sekciji 3.1.

Algoritmi zasnovani na memoriji

Sistemi koji koriste algoritme zasnovane na memoriji primenjuju statističke tehnike kako bi pronašli skup korisnika, poznatih kao susedi, koji imaju slične obrasce ponašanja kao ciljni korisnik (na primer, različite proizvode ocenjuju slično ili kupuju uglavnom isti ili sličan skup proizvoda). Takođe, ove tehnike se mogu iskoristiti i za nalaženje skupa sličnih stavki (na primer, proizvoda koji su slični kao proizvodi koje je korisnik već ocenio). Stoga, pod susedima se mogu podrazumevati susedi zasnovani na korisnicima ili susedi zasnovani na stavkama [1] [6]. Prema ovome se dele i varijante algoritma uzajamnog filtriranja zasnovanog na memoriji. Oba tipa, uzajamno filtriranje zasnovano na korisnicima i uzajamno filtriranje zasnovano na stavkama opisani su u nastavku ove sekcije.

Algoritmi zasnovani na memoriji rade sa $m \times n$ korisnik-stavka matricama ocena, gde je m broj korisnika i n je broj stavki. Ova matrica ocena će nadalje biti označena sa R. Svaka ćelija r_{ui} matrice R predstavlja ocenu za stavku i datu od strane korisnika u. Skup stavki ocenjen od strane korisnika u označava se kao I_u, a skup stavki ocenjen od strane oba korisnika u i v kao $I_u \cap I_v$ (ili kraće I_{uv}), skup korisnika koji su ocenili stavku i označava se kao U_i, a skup korisnika koji su ocenili stavke i i j označava se kao $U_i \cap U_j$ (ili kraće U_{ij}). U praksi, matrica ocena R je veoma retka, sa mnogo nepoznatih ulaza, s obzirom na to da je prosečan korisnik ocenio samo mali deo svih postojećih stavki [1] [6].

Najčešća aplikacija pristupa zasnovanog na memoriji je algoritam k najbližih suseda (eng. *Nearest Neighbor algorithm, KNN*), koji može biti zasnovan na korisnicima ili stavkama. Ovaj algoritam se može ugrubo opisati u dva koraka:

1. Pronalaženje k najsličnijih korisnika/stavki. Esencijalni deo ovog koraka je izbor odgovarajuće funkcije sličnosti, koja meri rastojanje između svaka dva korisnika ili svake dve stavke. Mere sličnosti su diskutovane u sekciji 3.2.

2. Predviđanje ocena koje bi ciljni korisnik dao svim neocenjenim stavkama koristeći ocene k korisnika pronađenih u prvom koraku. Varijante funkcija za predviđanje diskutovane su u sekciji 3.3.

Uzajamno filtriranje zasnovano na korisnicima Osnovna ideja ovog metoda je predviđanje ocena na osnovu preferenci suseda ciljnog korisnika. Na primer, kod sistema koji preporučuje knjige, da bi knjiga bila preporučena nekom korisniku, sistem koji koristi uzajamno filtriranje zasnovano na korisnicima, najpre pokušava

da pronađe druge korisnike koji imaju slične preference (ocenili su istu knjigu slično). Zatim, samo knjige koje se sviđaju k najsličnijim korisnicima bivaju preporučene [1] [6].

Funkcija sličnosti Da bi slični korisnici bili pronađeni koriste se funkcije (mere) sličnosti koje su oblika $s(u, v)$ i koje računaju sličnost između vektora ocena korisnika u i v, odnosno između redova matrice ocena R. U praksi se koriste različite funkcije sličnosti, a odabir odgovarajuće je veoma bitan za sistem preporuke, s obzirom na to da različite funkcije sličnosti daju različite rezultate. Neke od funkcija sličnosti koje se najčešće koriste kod metoda uzajamnog filtriranja date su u sekciji 3.2.

Funkcija za predviđanje Onda kada su izračunate sličnosti između korisnika i kada je definisana grupa k najsličnijih, mogu se predvideti ocene za bilo koji par korisnik-stavka. Ovo se postiže koristeći neku od varijanti funkcije za predviđanje (zasnovane na korisnicima).

Uzajamno filtriranje zasnovano na stavkama Pristup zasnovan na stavkama pruža predikcije zasnovane na ocenama koje je dao ciljni korisnik stavkama sličnim ciljnim stavkama. Na primeru sistema koji preporučuje knjige, da bi se predvidela ocena ciljnog korisnika A za neku knjigu B, detektuje se najsličnija knjiga (na primer S) knjizi B iz skupa već ocenjenih knjiga od strane korisnika A. Zatim, prosečna ocena (sa težinama) za knjigu S se koristi da se izračuna, odnosno predvidi ocena za knjigu B [1] [6].

Funkcija sličnosti U ovom slučaju, funkcije sličnosti se računaju između kolona matrice ocena R, kako bi se pronašle slične stavke. S obzirom na to da su algoritmi zasnovani na stavkama slični algoritmima zasnovanim na korisnicima, slične varijante funkcije sličnosti se mogu razmatrati u ovom pristupu. Jedina razlika je ta što se funkcija sličnosti računa za dve stavke umesto za dva korisnika i kalkulacije se vrše nad skupom korisnika koji su ocenili stavke i i j (označeno kao $Ui \cap Uj$, ili U_{ij}). Na primer, ukoliko bi se primenila prilagođena kosinusna mera sličnosti nad stavkama i i j, ona bi se izračunala ovako:

$$AC(i,j) = \frac{\sum_{u \in U_{ij}} (r_{iu} - \bar{r_u})(r_{ju} - \bar{r_u})}{\sqrt{\sum_{u \in U_i} (r_{iu} - \bar{r_u})^2 \sum_{u \in U_j} (r_{ju} - \bar{r_u})^2}}$$

gde $AC(i,j)$ predstavlja prilagođenu kosinusnu meru sličnosti stavki i i j, za sve korisnike koji su ocenili te stavke, r_{iu} i r_{ju} predstavljaju ocene koje je korisnik u dodelio stavkama i i j, respektivno, $\bar{r_u}$ predstavljaju srednju ocenu korisnika u u odnosu na sve ocenjene stavke, U_{ij} predstavlja skup zajedničkih korisnika koji su ocenili stavke i i j, U_i predstavlja skup zajedničkih korisnika koji su ocenili stavku i, U_j predstavlja skup zajedničkih korisnika koji su ocenili stavku j [2] [1] [6].

Funkcija za predviđanje Jednom kada je sličnost među stavkama izračunata, može se preći na predviđanje ocena koristeći neku od funkcija za predviđanje (zasnovanu na stavkama). U nastavku je prikazano predviđanje ocene koristeći sumu sa težinama. Ova funkcija za predviđanje kod pristupa zasnovanog na stavkama računa predikciju stavke i za korisnika u tako što računa sumu ocena koje kao težinu koriste odgovarajuće sličnosti između stavke i i njenih k sličnih stavki.

$$\hat{r}_{ui} = \frac{\sum_{j \in K_i(u)} AC(i,j) * r_{uj}}{\sum_{j \in K_i(u)} |AC(i,j)|}$$

gde $K_i(u)$ predstavlja k najslicnijih stavki ciljnoj stavci i koja je ocenjena od strane korisnika u. Drugim rečima, ovaj pristup pokušava da zaključi kako ciljni korisnik ocenjuje slične stavke. Suma sa težinama je podeljena sumom sličnih stavki, kako bi se osiguralo da je predviđena ocena između predefinisanog opsega.

Prednosti i mane tehnika zasnovanih na memoriji Glavna prednost tehnika zasnovanih na memoriji je to što su lake za implementaciju, dok pružaju relativno visok kvalitet predviđanja. Takođe, sadržajno su nezavisne (eng. *context independent*). Međutim, ovi sistemi nailaze i na neke probleme, kao što je to mali broj ocena na mnogo stavki, odnosno retkost podataka i problemi sporog početka (eng. *cold start problem*) koji se odnose na uvođenje novog proizvoda i uvođenje novog korisnika. U oba slučaja nedostaju podaci, odnosno interakcije korisnik-stavka, a to je nešto na šta su sistemi na uzajamnom filtriranju zasnovani [1] [6].

Algoritmi zasnovani na modelu

Ovaj pristup pre svega koristi tehnike istraživanja podataka i mašinskog učenja da nauči prediktivan model iz skupa podataka za trening. Taj model treba da karakteriše ocenjivanje krajnjih korisnika. Metode zasonave na modelu koriste istrenirani model da naprave predikcije, umesto da direktno koristi celu korisnik-stavka matricu za računanje predikcija [1] [8].

Tipični primeri metoda za filtriranje zasnovanih na modelu uključuju drveća odlučivanja (eng. *decision trees*), pravila pridruživanja (eng. *association rules*), Bajesove mreže (eng. *Bayesian networks*), latentni semantički modeli (eng. *latent semantic models*) i model klasterovanja (eng. *clustering model*). U nastavku ove sekcije će, bez umanjenja značaja ostalih modela, biti opisana SVD (eng. *Singular Value Decomposition*) tehnika, koja pripada latentnim semantičkim modelima. O ostalim nabrojanim modelima se može pročitati u radu [8].

Prednosti i mane algoritama zasnovanih na modelu Algoritmi zasnovani na modelu su brži nego algoritmi zasnovani na memoriji, jer vreme potrebno da se od modela zatraži da nešto predvidi je uglavnom mnogo manje od vremena potrebnog da se pretraži ceo skup podataka. Ipak, njihova mana je ta što su mnogi modeli kompleksni i nekada iako u teoriji deluju dobro, u stvarnosti se ne mogu dobro prilagoditi stvarnim podacima. Takođe, mnogo je vremena potrebno za implementaciju i kreiranje ovakvih modela kao i za njihovo ažuriranje, što ih čini nefleksibilnim [1] [8].

SVD Dekompozicija singularnih vrednosti (SVD) je metod linearne algrebre koji se u mašinskom učenju koristi kao tehnika za smanjenje dimenzionalnosti. SVD je tehnika faktorizacije matrice koja uzima matricu A dimenzija $m \times n$ i dekomponuje je na sledeći način [2] [11]:

$$SVD(A) = USV^T$$

U je leva orthogonalna singularna matrica dimenzija $m \times r$, koja predstavlja vezu između korisnika i latentnih faktora (karakteristika), S je $r \times r$ diagonalna matrica koja opisuje jačinu (značaj) latentnih faktora (vrednosti na dijagonali su ne-negativni realni brojevi) i V je $r \times n$ dijagonalna desno singularna matrica, koja predstavlja sličnost između stavki i latentnih faktora [2] [11].

Latentni faktori su zapravo karakteristike stavki, na primer muzički žanr. SVD redukuje dimenziju matrice A tako što ekstraktuje njene značajne informacije (latentne faktore) i preslikava svakog korisnika i svaku stavku u r-dimenzionalni latentni prostor. Ovime je postignuta jasna reprezentacija veze između korisnika i stavki [2] [11].

U kontekstu sistema preporuka, SVD se koristi kod uzajamnog filtriranja i primenjuje se nad matricom ocena (gde svaki red predstavlja korisnika, svaka kolona

predstavlja stavku, a elemnti ove matrice su ocene koje su date stavkama od strane tih korisnika). Najpre se pronalaze latentni faktori matrica iz procesa faktorizacije početne matrice korisnik-stavka-ocena. Zatim se data matrica dekomponuje odnosno razlaže na manje matrice. Na primer, matrica koja ima 2000 korisika i 1000 artikala imaće čak 2 miliona ulaza (ocena). Umesto da se čuva ovakva matrica, ona se može razložiti na dve matrice - jednu od 2000 korisnika i 100 atributa i drugu od 1000 stavki i 100 atributa. Ove matrice sada imaju 200 000 i 100 000 ulaza.

Ukoliko je potrebno pristupiti određenoj oceni, odnosno polju velike matrice, dovoljno je izračunati skalarni proizvod odgovarajućih vektora (vektora koji predstavlja tog korinika i vektora koji predstavlja tu stavku) i dobiti odgovarajuću ocenu [2] [11].

Jedna od najčešćih primena SVD-a jeste redukcija dimenzionalnosti skupa podataka. Prednosti modela zasnovanih na faktorizaciji matrice su takođe efikasnot, tačnost i skalabilnost. Međutim, ako su podaci strogo ne-linearni, ispostavlja se da ponašanje nije uvek dobro [2] [11].

Ograničenja sistema koji koriste uzajamno filtriranje

Budući da je broj korisnika i stavki konstantno u porastu, prikupljanje ulaznih podataka i kreiranje sistema preporuka koji koristi uzajamno filtriranje, a koji tačno predviđa je veliki izazov [1] [6] [8]. Dva glavna problema (izazova) kod uzajamnog filtriranja opisana su u nastavku.

Oskudnost podataka (eng. *Data Sparsity*) Uzajamno filtriranje zasnovano na korisnicima zavisi od eksplicitne povratne informacije, kao što su to ocene date artiklu od strane korisnika. Ulazna matrica podataka korisnik-stavka može imati samo nekoliko ocena u odnosu na ukupan broj stavki, iako su korisnici veoma aktivni.

Dodatno, s obzirom na to da korisnici uglavnom ne ocenjuju stavke aktivno, računanje sličnosti nad ovakvim skupom stavki može biti izazovno. Ovi problemi dovode do lošijih preformansi, odnosno nedovoljno tačnih predviđanja sistema preporuka. Čak je i problem sporog početka uzrokovan sa nedostatkom podataka. Uzajamno filtriranje predviđa stavke prema korisnikovom prethodnom ponašanju. To znači da ne može da predvidi stavke novim korisnicima sve dok novi korisnik ne oceni određeni broj stavki.

Mnoge metode za prevazilaženje problema oskudnosti podataka uključuju pristupima zasnovan na sadržaju budući da se oni ne oslanjaju u potpunosti samo na

ocene korisnika. Ovi pristupi koriste eksterne informacije o sadržaju da bi izvršili predviđanje za nove korisnike i stavke [1] [6] [8].

Skalabilnost podataka (eng. *Data Scalability*) S obzirom na to da su sistemi preporuke dizajnirani da navigiraju korisnike kroz ogromne kolekcije stavki, jedan od najvažnijih ciljeva takvog sistema je da dobro skalira kada su u pitanju pravi skupovi podataka. Kada broj postojećih korisnika i stavki dovoljno poraste, tradicionalni algoritmi zasnovani na uzajamnom filtriranju „pate" od problema skalabilnosti. Ovi algoritmi računaju sličnost između svaka dva korisnika/svake dve stavke, što dovodi do uskog grla (eng. *bottleneck*) u takvim sistemima. Sistemi preporuke koji koriste algoritme uzajamnog filtriranja zasnovane na memoriji donkle rešavaju ovaj problem upotrebom KNN algoritma i predviđanjem uzimajući u obzir samo susede. Algoritmi uzajamnog filtriranja zasnovani na modelu, kao što su to algoritmi klasterovanja, pristupaju problemu skalabilnosti tako što particionisu korisnike na male, a slične klastere, koristeći susedstvo, odnosno susedne particije za predikciju. Tehnike za smanjenje dimenzionalnosti, kao što je to SVD, mogu se izboriti sa problemom skalabilnosti i brzo proizvesti kvalitetne preporuke. Međutim, ove tehnike podrazumevaju skupe operacije faktorizacije matrice [1] [6] [8].

Hibridni sistemi

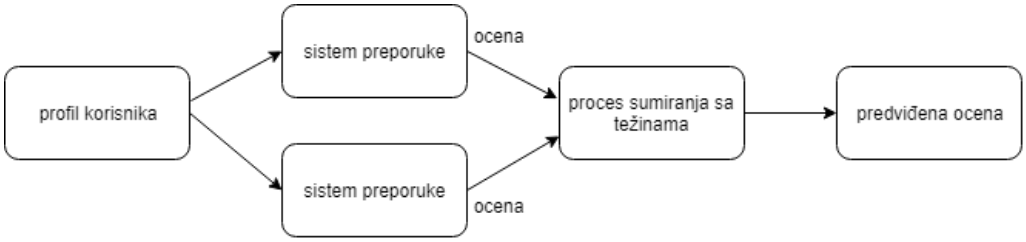
Svi do sada pomenuti algoritmi imaju različite prednosti i mane i svaki od tih algoritama je efikasan u različitim slučajevima. Na primer, sistemi koji koriste uzajamno filtriranje zavise od ocena korisnika, algoritmi zasnovani na sadržaju se oslanjaju na tekstualni opis stavki, a metode zasnovane na znanju (eng. *knowledge-based methods*) se oslanjaju na interakcije korisnika. Hibridni sistemi kombinuju različite tehnike, pokušavajući da iskoriste prednosti jedne i time poprave, odnosno zaobiđu, nedostatke druge tehnike. Na primer, metode za uzajamno filtriranje „pate" od problema „nova stavka" (eng. *„new item" problem*), odnosno ne mogu da preporuče stavku koja nema ni jednu ocenu. Međutim, ovo nije nikakvo ograničenje za pristup zasnovan na sadržaju, budući da se predviđanje ocene za novu stavku bazira na njenom opisu (karakteristikama) koje su uglavnom lako dostupne [8] [7].

Postoje različite strategije hibridizacije koje se mogu primeniti, zavisno od načina kombinovanja različitih tehnika za generisanje preporuka. Najčešće korišćen hibridni sistem je onaj koji kombinuje uzajamno filtriranje i metode zasnovane na sadržaju, gde se u obzir uzimaju karakteristike sadržaja i vrednovanje sadržaja od strane

korisnika [8]. Neki od tipova hibridnih sistema koji se najčešće koriste opisani su u nastavku.

Hibridni pristup sa težinama

Hibridni pristup sa težinama (eng. *Weighted hybrids)* podrazumeva zajedničko prezentovanje preporuka dobijenih na osnovu različitih tehnika. Izlaz, odnosno ocene dobijene od strane individualnih sistema preporuka su kombinovane upotrebom težina. Ovakav hibridni pristup karakteriše dizajn ensambla, jer podrazumeva kombinovanje zasebnih metoda za preporučivanje. Na primer, inicijalno se mogu dodeliti jednake težine preporukama generisanim koristeći uzajamno filtriranje i onim zasnovanim na sadržaju. A onda se težine posebno prilagođavaju u zavisnosti od toga koje predikcije korisnikovih ocena su tačne, odnosno netačne. Prednost ove vrste hibridnog sistema je što je u njemu integrisano nekoliko različitih modela koji učestvuju u procesu preporučivanja, dok je ceo proces prilično linearan [2] [8] [7]. Arhitektura sistema koji implementira hibridni pristup sa težinama je prikazana na slici 3.1.



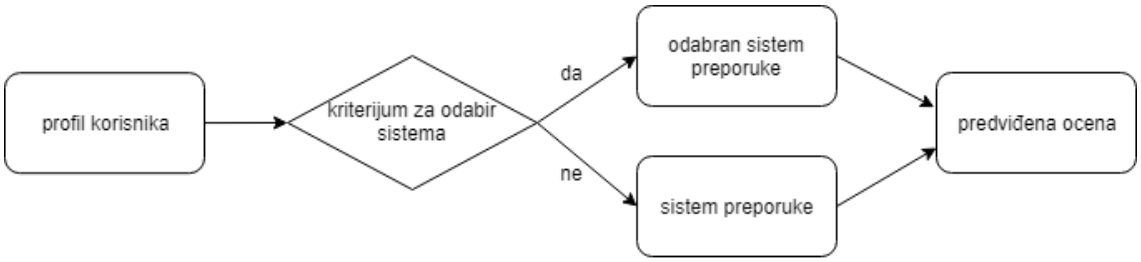
Slika 3.1: Hibridni sistem sa težinama

Smenjivanje modela

Smenjivanje modela (eng. *Switching model)* podrazumeva da se na osnovu određenog kriterijuma vrši izmena korišćenih tehnika u nekoliko faza i da se primenom jedne tehnike dobija model koji služi kao ulaz za narednu tehniku. Hibridni sistem sa smenjivanjem bira jedan jedini sistem preporuke u određenoj situaciji. Uglavnom se postavlja kriterijum na osnovu kojeg se bira sistem preporuka prema profilu korisnika ili na osnovu nekih drugih karakteristika. U svakom slučaju, izabran je onaj sistem preporuke koji je u datom trenutku „bolji" od drugog na osnovu neke metrike za ocenjivanje kvaliteta preporuke. Na primer, ukoliko se kombinuju sistemi preporuka zasnovani na sadržaju i oni koji koriste uzajamno filtriranje, prvo

može biti odabran sistem preporuke zasnovan na sadržaju a onda u nekom trenutku da se promeni i odabere sistem zasnovan na uzajamnom filtriranju. Kod ovog konkretnog hibridnog modela i dalje ostaje problem dodavanja novog korisnika, jer oba sistema, i sistem zasnovan na filtriranju i sistem zasnovan na sadržaju ne rešavaju ovaj problem [2] [7].

Benefit hibridnog sistema sa smenjivanjem je to što se mogu iskoristiti prednosti svakog od sistema. Međutim, ovaj sistem sa sobom donosi i nivo kompleksnosti u proces preporučivanja, budući da postoji dodatni kriterijum koji se mora odrediti, što uključuje još jedan nivo parametrizacije [2] [7]. Dizajn hibridnog sistem sa smenjivanjem prikazan je na slici 3.2.



Slika 3.2: Hibridni sistem sa smenjivanjem

Mešoviti hibrid

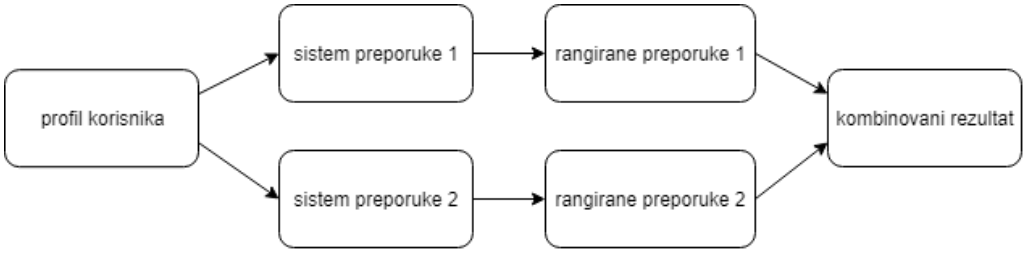
Kada je poželjno imati veći broj preporuka istovremeno, moguće je iskoristiti **mešoviti hibridni sistem (eng. *Mixed hybrid)***. Ovaj sistem podrazumeva da se preporuke generisane primenom više različitih tehnika prikazuju zajedno.

PTV sistem[3] koristi ovaj pristup da sastavi preporučeni program za gledanje televizije. Korišćene su tehnike zasnovane na sadržaju koje koriste tekstualne opise TV emisija i tehnike zasnovane na uzajamnom filriranju koje koriste informacije o preferencama korisnika. Preporuke dobijene ovim tehnikama su kombinovane zajedno u konačni program koji se predlaže. Ovakav hibridni sistem uspešno prelazi preko problema „nova stavka" jer se može osloniti na sistem zasnovan na sadržaju, koji će preporučiti nove emisije prema njihovim opisima, iako one još nisu ocenjene. Međutim, ne zaobilazi se problem „novi korisnik" (eng. "new user" problem), budući da oba sistema, i sistem zasnovan na sadržaju i sistem zasnovan na uzajamnom

[3]Cotter Smyth's PTV (2000) je primer hibridnog sistema preporuke koji koristi rezonovanje zasnovano na slučaju (eng. *case-based reasoning*) i uzajamno filtriranje za učenje preferenci korisnika [9].

filtriranju zahtevaju neke podatke o preferencama korisnika kako bi otpočeli proces preporučivanja [2] [7] [9].

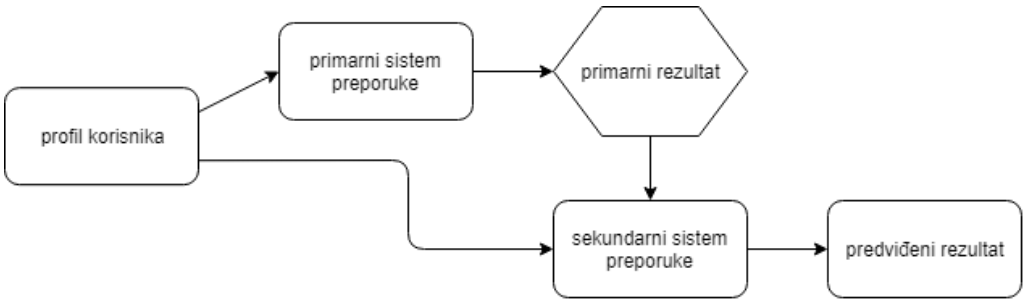
Nekada se kod ovih sistema više preporuka prezentuje jedna do druge. Međutim, nekada je potrebno rangirati preporuke odnosno preporučene stavke ili čak odabrati i prikazati samo jednu, najbolju preporuku, u kom slučaju se primenjuje neka tehnika za kombinaciju [2] [7]. Sistem preporuke koji implementira arhitekturu mešovitog hibrida, prikazan je na slici 3.3



Slika 3.3: Mešoviti hibrid

Kaskadni hibridni sistem

Kaskadni hibridni sistem (eng. *Cascade hybrid system)* definiše strogu hijerarhijsku strukturu sistema preporuka, tako da glavni sistem preporuke daje primarni rezultat, nakon kojeg se koristi sekundarni model kako bi rešio neke manje probleme primarnog rezultata, kao što je to „razbijanje" izjednačavanja u ocenjivanju. U praksi, većina skupova podataka je oskudna, pa sekundarni model sistema preporuke može biti efikasan po pitanju problema jednakog ocenjivanja (eng. *equal scoring issue*) ili problema sa nedostajućim podacima (eng. *missing data issue*) [2] [7]. Arhitektura Kaskadnog hibridnog sistema je prikazana na slici 3.4.



Slika 3.4: Kaskadni hibridni sistem

3.2 Mere sličnosti

U srcu algoritma uzajamnog filtriranja leži stroga pretpostavka da se "sličnim korisnicima sviđaju slične stavke". Iako se čini da je ova pretpostavka u velikoj meri istinita, ključno pitanje postaje: kako definisati i identifikovati slične korisnike da bi se zatim, na osnovu toga, pronašle slične stavke koje mogu biti preporučene? U obzir treba uzeti podatke koji su na raspolaganju - matricu korisnik-stavka, koja je uglavnom retka, jer je većina korisnika ocenila samo mali deo mnogobrojnih stavki (na primer, artikala u prodavnici).

U GroupLens sistemu preporuka, za odgovarajuću meru za računanje sličnosti korisnika predložena je Pirsonova korelacija (eng. *Pearson Correlation, PC*). Osim ove, postoje i druge mere sličnosti. Neke od njih su Euklidsko rastojanje (eng. *Euclidean Distance, ED*) i Kosinusno rastojanje (eng. *Cosine Distance*). O tome koja je najbolja vodi se otvorena rasprava, gde neki autori ističu da je Pirsonova korelacija najpogodnija za pristup zasnovan na sličnosti korisnika, dok je Kosinusno rastojanje najbolji izbor za pristup zasnovan na sličnosti stavki [2] [1] [6]. U nastavku su detaljno opisane funkcije sličnosti i odgovarajuće formule. Korisnici i stavke su predstavljeni kao vektori atributa.

Euklidsko rastojanje

Euklidsko rastojanje meri sličnost dva vektora tako što računa rastojanje između njih. Euklidsko rastojanje između dva korisnika se računa po formuli:

$$ED(u,v) = \sqrt{\frac{\sum_{i \in I_{uv}} (r_{ui} - r_{vi})^2}{|I_{uv}|}}$$

gde $ED(u,v)$ predstavlja sličnost korisnika u i v, za sve stavke koje su zajednički ocenjene, r_{ui} i r_{vi} predstavljaju ocene dodeljene stavki i od strane korisnika u i v, respektivno, a I_{uv} predstavlja skup stavki koje su zajednički ocenjene od strane korisnika u i v. Naredna formula predstavlja Euklidsko rastojanje između dve stavke:

$$ED(i,j) = \sqrt{\frac{\sum_{u \in U_{ij}} (r_{iu} - r_{ju})^2}{|U_{ij}|}}$$

gde $ED(i,j)$ predstavlja sličnost stavki i i j, za sve korisnike koji su ocenili te stavke, r_{iu} i r_{ju} predstavljaju ocene koje je korisnik u dodelio stavkama i i j, respektivno, U_{ij} predstavlja skup korisnika koji su zajednički ocenili stavke i i j. U nekim implementacijama sistema preporuka, Euklidska sličnost je definisana kao:

$$\hat{ED}(a,b) = \frac{1}{1 + ED(a,b)}$$

gde $\hat{ED}(a,b)$ predstavlja meru Euklidske sličnosti, a $ED(a,b)$ predstavlja euklidsko rastojanje. Efekat ove formule je ograničenje ocene na interval (0..1] [2] [1] [6].

Pirsonova korelacija

Ovu meru je predložio Karl Pirson (Karl Pearson, 1895) za merenje linearne korelacije između dve (neprekidne) varijable i ona je postala veoma široko korišćena u domenu statistike. Što se tiče sistema preporuka, mera je prvi put uvedena od strane projekta GroupLens 1994, i od toda se svuda koristi kao osnov za poređenje sa drugim metrikama [6].

Pirsonov koeficijent korelacije predstavlja linearnu korelaciju između korisnika/stavki, a izračunava se kao količnik kovarijanse dva korisnika i standardne devijacije između njih. Vrednosti koeficijenta korelacije kreću se u rasponu od -1 do +1. Predznak pokazuje da li je korelacija pozitivna (obe promenljive zajedno i opadaju i rastu) ili negativna (jedna promenljiva opada kada druga raste i obrnuto). Apsolutna vrednost tog koeficijenta pokazuje jačinu veze. Korelacija koja iznosi +1 ili -1, pokazuje da se vrednost jedne promenljive može tačno utvrditi ukoliko se zna vrednost druge (+1 predstavlja strogo pozitivnu korelaciju, -1 predstavlja strogo negativnu korelaciju). S druge strane, korelacija jednaka nuli pokazuje da između te dve promenljive ne postoji nikakva veza, te poznavanje vrednosti jedne promenljive nimalo ne pomaže u predviđanju vrednosti druge. Jednačina Pirsonove korelacije za računanje sličnosti između dva korisnika definiše se kao:

$$PC(u,v) = \frac{\sum_{i \in I_{uv}} (r_{ui} - \bar{r_u})(r_{vi} - \bar{r_v})}{\sqrt{\sum_{i \in I_{uv}} (r_{ui} - \bar{r_u})^2} \sqrt{\sum_{i \in I_{uv}} (r_{vi} - \bar{r_v})^2}}$$

gde $PC(u,v)$ predstavlja sličnost korisnika u i v, I_{uv} predstavlja skup stavki ocenjenih od strane oba korisnika u i v, r_{ui} i r_{vi} predstavljaju ocene korisnika u odnosno v za stavku i, i $\bar{r_u}$ i $\bar{r_v}$ predstavljaju srednju ocenu korisnika u odnosno v, u odnosu na sve ocenjene stavke. U slučaju sistema preporuke zasnovanih na stavkama, formulacija je sledeća:

$$PC(i,j) = \frac{\sum_{u \in U_{ij}} (r_{ui} - \bar{r_i})(r_{uj} - \bar{r_j})}{\sqrt{\sum_{u \in U_{ij}} (r_{ui} - \bar{r_i})^2} \sqrt{\sum_{u \in U_{ij}} (r_{uj} - \bar{r_j})^2}}$$

gde $PC(i,j)$ predstavlja sličnost stavke i i j, U_{ij} predstavlja skup zajedničkih korisnika koji su ocenili stavke i i j, r_{ui} i r_{uj} predstavljaju ocene koje je korisnik u dodelio stavkama i i j, respektivno i $\bar{r_i}$ i $\bar{r_j}$ koji predstavljaju srednju ocenu stavke i odnosno j, u odnosu na sve korisnike koji su ocenili te stavke [2] [1] [6].

Kosinusna sličnost

Upotrebom Kosinusne sličnosti (rastojanja) određuje se sličnost između dva vektora (dve stavke/dva korisnika), i to tako što se računa kosinus ugla između njih. Kosinusna sličnost se često koristi da izmeri sličnost dokumenata prilikom analize teksta. Osim toga, ima primenu i kod sistema preporuka. Skala vrednosti koja se koristi kod Kosinusne sličnosti ista je kao ona koja se koristi kod Pirsonove korelacije, gde +1 i -1 predstavljaju visoku (direktnu i inverznu) korelaciju. Naime, +1 označava da su vektori 100% slični, dok 0 predstavlja nekorelisanost. U formi vektora, kosinusna sličnost je definisana kao:

$$\cos(a,b) = \frac{\vec{a} \cdot \vec{b}}{|\vec{a}|^2 |\vec{b}|^2}$$

gde $cos(a,b)$ predstavlja sličnost vektora a i b, a a i b su dati vektori. Jednačina Kosinusne sličnosti koja računa sličnost između dva korisnika definiše se kao:

$$CD(u,v) = \frac{\sum_{i \in I_{uv}} r_{ui} r_{vi}}{\sqrt{\sum_{i \in I_u} r_{ui}^2 \sum_{i \in I_v} r_{vi}^2}}$$

gde $CD(u,v)$ predstavlja sličnost korisnika u i v, preko svih zajednički ocenjenih stavki, r_{ui} i r_{vi} predstavljaju ocene korisnika u odnosno v za stavku i, I_{uv} predstavlja skup stavki ocenjenih od strane oba korisnika u i v, I_u predstavlja skup stavki ocenjen od strane korisnika u, I_v predstavlja skup stavki ocenjen od strane korisnika v. Kosinusna sličnost između dve stavke računa se po formuli:

$$CD(i,j) = \frac{\sum_{u \in U_{ij}} r_{iu} r_{ju}}{\sqrt{\sum_{u \in U_i} r_{iu}^2 \sum_{u \in U_j} r_{ju}^2}}$$

gde $CD(i,j)$ predstavlja sličnost stavki i i j, preko svih korisnika koji su ocenili takve stavke, r_{iu} i r_{ju} predstavljaju ocene koje je korisnik u dodelio stavkama i i j, respektivno, U_{ij} predstavlja skup zajedničkih korisnika koji su ocenili stavke i i j, U_i predstavlja skup zajedničkih korisnika koji su ocenili stavku i, U_j predstavlja skup zajedničkih korisnika koji su ocenili stavku j [2] [1] [6].

Prilagođena kosinusna mera sličnosti

Prilagođena kosinusna mera sličnosti (eng. *Adjusted Cosine similarity measure, AC*) je modifikovani oblik vektorske (kosinusne) sličnosti gde se u obzir uzima i činjenica da različiti korisnici imaju različite šeme ocenjivanja. Drugim rečima, neki korisnici mogu generalno visoko ocenjivati stavke, a drugi mogu dati niže ocene iako im se artikal sviđa. Da bi se uklonio ovaj nedostatak iz sličnosti zasnovane na vektorima, oduzima se prosečna ocena od svake individualne ocene, odnosno od svake ocene korisnika za svaku stavku. U ovom slučaju, formulacija kod pristupa zasnovanog na korisnicima postaje:

$$AC(u,v) = \frac{\sum_{i \in I_{uv}} (r_{ui} - \bar{r_i})(r_{vi} - \bar{r_i})}{\sqrt{\sum_{i \in I_u} (r_{ui} - \bar{r_i})^2 \sum_{i \in I_v} (r_{vi} - \bar{r_i})^2}}$$

gde $AC(u,v)$ predstavlja sličnost korisnika u i v za sve stavke koje su zajednički ocenjene, r_{ui} i r_{vi} predstavljaju ocene korisnika u odnosno v za stavku i, $\bar{r_i}$ predstavlja srednju ocenu stavke i u odnosu na sve korisnike koji su ocenili tu stavku, I_{uv} predstavlja skup stavki ocenjenih od strane oba korisnika u i v, I_u predstavlja skup stavki ocenjen od strane korisnika u, I_v predstavlja skup stavki ocenjen od strane korisnika v. U slučaju sistema preporuka zasnovanih na stavkama, formulacija postaje:

$$AC(i,j) = \frac{\sum_{u \in U_{ij}} (r_{iu} - \bar{r_u})(r_{ju} - \bar{r_u})}{\sqrt{\sum_{u \in U_i} (r_{iu} - \bar{r_u})^2 \sum_{u \in U_j} (r_{ju} - \bar{r_u})^2}}$$

gde $AC(i,j)$ predstavlja sličnost stavki i i j za sve korisnike koji su ocenili te stavke, r_{iu} i r_{ju} predstavljaju ocene koje je korisnik u dodelio stavkama i i j, respektivno, $\bar{r_u}$ predstavljaju srednju ocenu korisnika u u odnosu na sve ocenjene stavke, U_{ij} predstavlja skup zajedničkih korisnika koji su ocenili stavke i i j, U_i predstavlja skup zajedničkih korisnika koji su ocenili stavku i, U_j predstavlja skup zajedničkih korisnika koji su ocenili stavku j [2] [1] [6].

Žakardova sličnost

Žakardov indeks (eng. *Jaccard index, J*) određuje sličnost, odnosno različitost dva skupa. Žakardov indeks između dva konačna skupa je definisan kao količnik kardinalnosti preseka i kardinalnosti unije. Odnosno, meri udeo broja elemenata koji su zajednički za dva skupa u odnosu na ukupan broj elemenata u oba skupa. Vrednost Žakardovog indeksa je u opsegu između 0 i 1. Što je vrednost bliža 1, data dva vektora su sličnija. Razultat je 0 ukoliko dva vekora nemaju nikakvih sličnosti.

Žakardov indeks ignoriše vrednosti ocena, proveravajući samo da li je korisnik izrazio preferencu ili ne. Ova metrika meri sličnost između dva korisnika tako što računa ukupan broj jedinstvenih stavki za koje je bar jedan od dva korisnika zainteresovan i to deli sa brojem stavki koje su zajednički ocenjene. Dakle, dva korisnika će biti sličnija kada imaju više stavki koje su zajedno ocenili. Jednačina koja računa Žakardov indeks (koeficijent) sličnosti između korisnika (vektora) u i v je data sa:

$$J(u,v) = \frac{|I_u \cap I_v|}{|I_u \cup I_v|}$$

gde I_u predstavlja skup stavki ocenjenih od strane korisnika u, a I_v predstavlja skup stavki ocenjenih od strane korisnika v [2] [1] [6].

Srednje kvadratno rastojanje

Sličnost preko **srednje kvadratnog rastojanja (eng. *Mean Square Distance, MSD*)** se definiše kao:

$$MSD(u,v) = \frac{\sum_{i \in I_{uv}} (r_{ui} - r_{vi})^2}{|I_{uv}|}$$

gde $MSD(u,v)$ predstavlja sličnost korisnika u i v za sve stavke koje su zajednički ocenjene, r_{ui} i r_{vi} predstavljaju ocene korisnika u odnosno v za stavku i, a I_{uv} predstavlja skup stavki ocenjenih od strane oba korisnika u i v. U slučaju sistema preporuka zasnovanog na stavkama, formulacija postaje:

$$MSD(i,j) = \frac{\sum_{u \in U_{ij}} (r_{iu} - r_{ju})^2}{|U_{ij}|}$$

gde $MSD(i,j)$ predstavlja sličnost stavki i i j za sve korisnike koji su ocenili ove stavke, r_{iu} i r_{ju} predstavljaju ocene koje je korisnik u dodelio stavkama i i j, respektivno, a U_{ij} predstavlja skup zajedničkih korisnika koji su ocenili stavke i i j. U nekim radovima [1] se preporučuje upotreba inverznog srednje kvadratnog rastojanja:

$$\hat{MSD}(a,b) = \frac{1}{MSD(a,b)}$$

gde $\hat{MSD}(a,b)$ predstavlja srednje kvadratnu sličnost, a $MSD(a,b)$ predstavlja srednje kvadratno rastojanje [2] [1] [6].

3.3 Funkcije predviđanja

Nakon odabira suseda, poslednji korak u procesu preporučivanja je predviđanje ocena. Funkcije za predviđanje podrazumevaju različite načine za procenu nepoznatih ocena za stavke koje su ocenjene od strane suseda ciljnog korisnika, ali nisu ocenjene od samog ciljnog korisnika. Sa listom procenjenih ocena za neocenjene stavke, sistem preporuke jednostavno sortira listu opadajuće i preporuči prvih n stavki ciljnom korisniku. Tri najčešće korišćene varijante funkcije za predviđanje su:

- Suma sa težinama (eng. *Weighted sum*)
- Pristup zasnovan na centriranju proseka (eng. *Mean Centering approach*)
- Pristup zasnovan na Z-skoru (eng. *Z-score approach*)

U nastavku su opisana sva tri pristupa, i to, suma sa težinama u sekciji 3.3, pristup zasnovan na centriranju proseka u sekciji 3.3, i pristup zasnovan na Z-skoru u sekciji 3.3.

Suma sa težinama

Kod ovog pristupa, predikcija se računa kao prosek svih dostupnih ocena, sa težinama. Za težinu se uzima vrednost korelacije (sličnosti) koja se izračunava pomoću funkcije sličnosti. Formula za računanje procene ocene u sistemu zasnovanom na korisnicima data je sa:

$$\hat{r}_{ui} = \frac{\sum_{v \in K_i(u)} s(u,v) r_{vi}}{\sum_{v \in K_i(u)} |s(u,v)|}$$

gde $K_i(u)$ predstavlja broj suseda koji imaju stavku i zajedničku sa korisnikom u, od kojih je v jedan takav sused, a $s(u,v)$ predstavlja sličnost između korisnika u i jednog od njegovih suseda v. Za sisteme zasnovane na stavkama, formula postaje:

$$\hat{r}_{ui} = \frac{\sum_{j \in K_u(i)} s(i,j) r_{ju}}{\sum_{j \in K_u(i)} |s(i,j)|}$$

gde $K_u(i)$ predstavlja broj susednih stavki za korisnika u, od kojih je j jedan takav sused, a $s(i,j)$ predstavlja sličnost stavke i i jedne od njenih suseda - stavke j.

Pristup zasnovan na centriranju proseka

Upotrebom pristupa zasnovanog na centriranju proseka, formula za sisteme koji su zasnovani na korisnicima postaje:

$$\hat{r}_{ui} = \bar{r}_u + \frac{\sum_{v \in K_i(u)} s(u,v)(r_{vi} - \bar{r}_v)}{\sum_{v \in K_i(u)} |s(u,v)|}$$

gde $\hat{r}_{ui}$ predstavlja ocenu korisnika u za stavku i, $K_i(u)$ predstavlja k najsličnijih korisnika ciljnom korisniku u, koji su ocenili stavku i, $s(u,v)$ je funkcija sličnosti za korisnike u i v, r_{vi} je ocena korisnika v za stavku i, a $\hat{r}_v$ je prosečna ocena korisnika v. Za sisteme zasnovane na stavkama, ova fromula postaje:

$$\hat{r}_{ui} = \bar{r}_i + \frac{\sum_{j \in K_u(i)} s(i,j)(r_{ju} - \bar{r}_j)}{\sum_{j \in K_u(i)} |s(i,j)|}$$

gde $\hat{r}_{ui}$ predstavlja ocenu korisnika u za stavku i, $K_u(i)$ predstavlja k najsličnijih stavki sa stavkom i ocenjenih od strane korisnika u, $s(i,j)$ je funkcija sličnosti za stavke i i j, r_{ju} je ocena korisnika u za stavku j, a $\hat{r}_j$ je prosečna ocena za stavku j.

Pristup zasnovan na Z-skoru

Formula koja predstavlja varijaciju prethodne funkije, u kojoj se koristi $Z-skor$, za sisteme zasnovane na korisnicima, definiše se na sledeći način:

$$\hat{r}_{ui} = \bar{r}_u + \sigma u \frac{\sum_{v \in K_i(u)} s(u,v)(r_{vi} - \bar{r}_v)/\sigma u}{\sum_{v \in K_i(u)} |s(u,v)|}$$

gde $\hat{r}_{ui}$ predstavlja ocenu korisnika u za stavku i, $K_i(u)$ predstavlja k najsličnijih korisnika ciljnom korisniku u, koji su ocenili stavku i, $s(u,v)$ je funkcija sličnosti za korisnike u i v, r_{vi} je ocena korisnika v za stavku i, $\hat{r}_v$ je prosečna ocena korisnika v, a σu je varijansa ocena korisnika u. Za sisteme zasnovane na stavkama, ova fromula postaje:

$$\hat{r}_{ui} = \bar{r}_i + \sigma i \frac{\sum_{j \in K_u(i)} s(i,j)(r_{ju} - \bar{r}_j)/\sigma i}{\sum_{j \in K_u(i)} |s(i,j)|}$$

gde $\hat{r}_{ui}$ predstavlja ocenu korisnika u za stavku i, $K_u(i)$ predstavlja k najsličnijih stavki sa stavkom i ocenjenih od strane korisnika u, $s(i,j)$ je funkcija sličnosti za stavke i i j, r_{ju} je ocenea korisnika u za stavku j, $\hat{r}_j$ je prosečna ocena za stavku j, a σu je varijansa ocena stavke i.

3.4 Evaluacija sistema preporuka

Ukoliko je dato više različitih implementacija sistema preporuka, potrebno je nekako izmeriti koji sistem je bolji u smislu efikasnosti, tačnosti predviđanja i drugih performansi. Evaluacija (eng. *evaluation*) sistema preporuka podrazumeva upotrebu različitih metrika za merenje performansi sistema u različitim aspektima. Zbog raznovrsnosti sistema preporuka, ne postoji jedinstvena metrika kojom se mogu oceniti svi aspekti sistema preporuke. Naime, različite metrike mogu biti kombinovane kako bi se izvršila celokupna evaluacija sistema [2][3].

Sistemi preporuka dele nekoliko konceptualnih sličnosti sa problemom klasifikacije i problemom regresije. Kod modela za klasifikaciju i regresiju treba predvideti klasu odnosno ciljnu promenljivu, koristeći preostale vrednosti atributa. U sistemima preporuka, data je matrica ocena u kojoj mnoga polja (ocene) nedostaju i treba ih predvideti, uzimajući u obzir ostatak polja (ocena) iz matrice. U ovom smislu, sistemi preporuka mogu biti posmatrani kao generalizacija problema klasifikacije. Stoga, mnoge metrike koje se koriste za evaluaciju modela za klasifikaciju, uz određene modifikacije, mogu biti iskorišćene i kod sistema preporuka. U ovoj sekciji, fokus je na merenju tačnosti, koja se smatra za jedan od najvažnijih kriterijuma. Ipak, najpre je objašnjeno prema čemu se sve može vršiti evaluacija [2].

Sistemi preporuke mogu biti evaluirani u pogledu tačnosti predviđenih ocena ili u pogledu tačnosti rangiranja stavki. Što se tiče tačnosti predviđenih ocena, koriste se metrike kao što su srednje apsolutna greška (eng. *mean absolute error, MAE*) i koren srednje kvadratne greške (eng. *root mean squared error, RMSE*). Ove metrike su detaljno opisane u sekciji 3.4. Evaluacija rangiranja se može izvršiti korišćenjem različitih metoda, kao što su to proračuni zasnovani na korisnosti (eng. *utility-based computations*), koeficijenti korelacije ranga (eng. *rank-correlation coefficients*) i ROC krive (eng. *receiver operating characteristic curve*). Više o ovim metrikama može se pročitati u [2], u poglavljima 7.5.2 - 7.5.4.

Merenje tačnosti predviđenih ocena

U ovom radu je fokus na merenju kvaliteta odnosno tačnosti predviđenih ocena i to onih koje su dobijene oslanjajući se na eksplicitne povratne informacije, gde korisnik direktno izražava svoje preference koristeći ocene. Za ovu evaluaciju najčešće se koriste već pomenute metrike: srednje apsolutna greška i koren srednje kvadratne greške. Ove metrike su zasnovane na razlici između ocene koja je predviđena za neku

stavku i ocene koju je korisnik zaista dao toj stavci [2][3]. Obe metrike su detaljno opisane u nastavku, u podsekcijama 3.4 i 3.4.

Takođe, za obe metrike važe sledeća tvrđenja. Tačnost se meri na skupu za testiranje. Neka je S skup svih razmatranih podataka, a $E \subset S$ skup podataka korišćenih za testiranje. Svaki podatak skupa E je uređeni par korisnik-stavka oblika (u, j) što odgovara poziciji u matrici ocena. Neka je r_{uj} vrednost (skrivene) ocene na poziciji $(u, j) \in E$ i neka je $\hat{r}_{uj}$ predviđena ocena na poziciji (u, j), korišćenjem modela koji je prethodno treniran na skupu za treniranje. Razlika ove dve ocene označena je sa $e_{uj} = \hat{r}_{uj} - r_{uj}$[2].

MAE

Srednje apsolutna greška predstavlja prosečnu apsolutnu grešku između predviđene i prave ocene date od strane korisnika. Prednosti ove metrike su to što je jednostavna za razumevanje i laka za implementaciju. Formula za računanje MAE data je u nastavku [2].

$$MAE = \frac{\sum_{(u,j)\in E} |e_{uj}|}{|E|}$$

RMSE

Koren srednje kvadratne greške predstavlja koren prosečne kvadratne greške između predviđene i prave ocene date od strane korisnika. Jedna od karakteristika ove metrike, a ujedno i razlika u odnosu na srednje apsolutnu grešku, jeste to što RMSE ima tendenciju da nesrazmerno kažnjava velike greške zbog kvadratnog člana unutar sume. Formula za računanje RMSE data je u nastavku [2].

$$RMSE = \sqrt{\frac{\sum_{(u,j)\in E} e_{uj}^2}{|E|}}$$

Glava 4

Eksperimentalni rezultati rada

U sklopu ovog rada razvijeno je nekoliko različitih modela sistema preporuka. U ovoj sekciji je najpre opisan skup podataka koji se koristi u implementaciji ovih rešenja, a zatim je prikazan postupak pretprocesiranja podataka, formiranja modela, tumačenje njihovog rezultata i evaluacija. Razvijeni modeli su nabrojani u nastavku, kao i brojevi sekcija u kojima su oni opisani:

- Sistem preporuke koji koristi uzajamno filtriranje i k najbližih suseda (zasnovan na memoriji) - pristup zasnovan na korisnicima i pristup zasnovan na stavkama (sekcija 4.5)
- Sistem preporuke koji koristi uzajamno filtriranje i k najbližih suseda, zasnovan na memoriji - upotrebom biblioteke Surprise (sekcija 4.2)
- Sistem preporuke koji koristi uzajamno filtriranje, zasnovan na modelu - upotrebom biblioteke Surprise (sekcija)
- Sistem preporuke zasnovan na dubokom učenju - upotrebom biblioteke Keras (sekcija)
- Hibridni sistem preporuke sa težinama i sa smanjivanjem (sekcija)

Određivanje hiperparametara pomenutih modela je izvršeno validacijom. Skup podataka je podeljen na skup za treniranje i skup za testiranje (i validaciju). Ocenjivanje modela se vrši u odnosu na test skup.

4.1 O skupu podataka

Korišćeni su podaci o proizvodima koji se prodaju na Amazonu, a preuzeti su iz javno dostupne baze podataka. Skup podataka se sastoji iz dve tabele. Jedna tabela sadrži samo informacije o proizvodima, kao što su: naziv proizvoda, brend, dimenzije, težina i ostale propratne informacije vezane za proizvode. Druga tabela sadrži informacije o interakcijama korisnik-stavka, odnosno sadrži atribute o korisniku, oceni, proizvodu, vremenu pregleda proizvoda, utisku i ostale prateće informacije. S obzirom na to da skup podataka sadrži dosta atributa koji za sprovođenje pomenutih algoritama nisu od značaja, mnogi atributi su eliminisani u procesu pretprocesiranja. Vrednosti atributa koji predstavljaju ocenu su iz intervala [1, 5] i mogu biti samo celi brojevi. Primer skupa podataka (koji se sastoji iz dve tabele) je prikazan na slikama 4.1 i 4.2.

	title	brand	feature/0	feature/1	rank	date	asin	imageURL/0	imageURLHighRes/0	imageURL/1
0	Slime Time Fall Fest [With CDROM and Collector...	Group Publishing (CO)	Product Dimensions:\n \n8.7...	Shipping Weight:\n \n2.4 po...	13,052,976inClothing,Shoesamp;Jewelry(	8.70 inches	0764443682	https://images-na.ssl-images-amazon.com /images...	https://images-na.ssl-images-amazon.com /images...	NaN
1	XCC Qi promise new spider snake preparing men'...	NaN	NaN	NaN	11,654,581inClothing,Shoesamp;Jewelry(	5 star	1291691480	https://images-na.ssl-images-amazon.com /images...	https://images-na.ssl-images-amazon.com /images...	https://images-na.ssl-images-amazon.com /images...
2	Magical Things I Really Do Do Too!	Christopher Manos	Package Dimensions:\n \n8.5...	Shipping Weight:\n \n6.1 ou...	19,308,073inClothing,ShoesJewelry(	5 star	1940280001	https://images-na.ssl-images-amazon.com /images...	https://images-na.ssl-images-amazon.com /images...	https://images-na.ssl-images-amazon.com /images...

Slika 4.1: Skup podataka - tabela sa informacijama o proizvodima

	overall	verified	reviewTime	reviewerID	asin	reviewerName	reviewText	summary	unixReviewTime	vote	...	image/6	image/7	image
0	5	True	10 20, 2014	A1D4G1SNUZWQOT	7106116521	Tracy	Exactly what I needed.	perfect replacements!!	1413763200	NaN	...	NaN	NaN	Na
1	2	True	09 28, 2014	A3DDWDH9PX2YX2	7106116521	Sonja Lau	I agree with the other review, the opening is ...	I agree with the other review, the opening is ...	1411862400	3.0	...	NaN	NaN	Na
2	4	False	08 25, 2014	A2MWC41EW7XL15	7106116521	Kathleen	Love these... I am going to order another pack...	My New 'Friends' !!	1408924800	NaN	...	NaN	NaN	Na

Slika 4.2: Skup podataka - tabela sa informacijama o interakcijama

4.2 Implementacija pristupa KNN i SVD korišćenjem biblioteke Surprise

Biblioteka Surprise ima nekoliko ugrađenih algoritama za kreiranje sistema preporuka zasnovanih na ocenama. Ovde su, uz korišćenje pomenute biblioteke, prezen-

tovana dva različita pristupa: pristup zasnovan na memoriji, gde se koriste različite varijacije algoritma KNN i pristup zasnovan na modelu, u kome se koristi SVD tehnika. U nastavku su, u podsekcijama 4.2 i 4.2, najpre ukratko opisana oba pristupa. Nakon toga, u sekciji 4.2 je opisano pretprocesiranje podataka. Performanse modela su diskutovane u sekciji 4.2. Modeli sa najboljim performansama iskoričćeni su za generisanje preporuka.

KNN - pristup zasnovan na memoriji

Ovaj algoritam za kreiranje preporuka uzima u obzir do k najbližih korisnika (kod uzajamnog filtriranja zasnovanog na korisnicima) ili do k najbližih stavki (kod uzajamnog filtriranja zasnovanog na stavkama). Podrazumevano, algoritam je zasnovan na korisniku i k ima vrednost 40 ($kmin$ je 1). To znači da se razmatraju 40 najbližih korisnika da bi neka stavka bila preporučena korisniku. Varijante osnovnog KNNBasic algoritma (opisanog u podsekciji 4.2) uključuju WithMeans, WithZScore i Baseline (opisane u podsekcijama 4.2, 4.2 i 4.2, respektivno), gde se dodatno, razmatraju i prosečna ocena korisnika, normalizovani Z-skorovi ocena ili početna ocena, radi generisanja preporuka. Svakom od ovih algoritama se prosleđuje broj suseda koji ulazi u razmatranje. Takođe, uključeni su samo oni susedi za koje mere sličnosti daju pozitivne rezultate, s obzirom na to da nema smisla prikupljati ocene od korisnika (za stavke) koji su negativno korelisani. Za dato predviđanje, stvarni broj suseda se može pročitati iz polja $actual\_k$ iz rečnika detalja predviđanja.

KNNBasic

KNNBasic predstavlja osnovni algoritam zasnovan na susedstvu koji koristi uzajamno filtriranje za predviđanje ocena. Argumenti koje prima metod KNNBasic su:

- k (int): Maksimalan broj suseda koji se uzima u razmatranje. Podrazumevana vrednost ovog argumenta je 40.
- min_k (int): Minimalan broj suseda koji se uzima u razmatranje. Ukoliko nema dovoljno suseda, za predikciju se uzima globalna prosečna vrednost svih ocena. Podrazumevana vrednost ovog argumenta je 1.
- $sim\_options$ (dict): Rečnik opcija vezano za meru sličnosti koja se koristi.

- *verbose* (bool): Vrednost ovog argumenta govori o tome da li će se štampati prateće poruke vezano za procenu bias-a, sličnosti, itd. Podrazumevana vrednost je *True*.

Predviđena ocena $\hat{r}_{ui}$ se računa na sledeći način:

$$\hat{r}_{ui} = \frac{\sum_{v \in N_i^k(u)} s(u,v) * r_{vi}}{\sum_{v \in N_i^k(u)} s(u,v)}$$

ili

$$\hat{r}_{ui} = \frac{\sum_{j \in N_u^k(i)} s(i,j) * r_{uj}}{\sum_{j \in N_u^k(i)} s(i,j)}$$

zavisno od vrednosti polja *user_based* unutar parametra *sim_options* [12].

KNNWithMeans

Osnovni algoritam zasnovan na susedima koji koristi uzajamno filtriranje i koji uzima u obzir i prosek ocena za svakog korisnika. Argumenti koje prima metod KNNWithMeans su:

- k (int): Maksimalan broj suseda koji se uzima u razmatranje. Podrazumevana vrednost ovog argumenta je 40.
- *min_k* (int): Minimalan broj suseda koji se uzima u razmatranje. Ukoliko nema dovoljno suseda, predikcija postaje jednaka prosečnoj vrednosti μu ili μi. Podrazumevana vrednost ovog argumenta je 1.
- *sim_options* (dict): Rečnik opcija vezano za meru sličnosti koja se koristi.
- *verbose* (bool): Vrednost ovog argumenta govori o tome da li će se štampati prateće poruke vezano za procenu bias-a, sličnosti, itd. Podrazumevana vrednost je *True*.

Predviđena ocena $\hat{r}_{ui}$ se računa na sledeći način:

$$\hat{r}_{ui} = \mu u + \frac{\sum_{v \in N_i^k(u)} s(u,v) * (r_{vi} - \mu v)}{\sum_{v \in N_i^k(u)} s(u,v)}$$

ili

$$\hat{r}_{ui} = \mu i + \frac{\sum_{j \in N_u^k(i)} s(i,j) * (r_{uj} - \mu j)}{\sum_{j \in N_u^k(i)} s(i,j)}$$

zavisno od vrednosti polja *user_based* unutar parametra *sim_options* [12].

KNNWithZScore

Osnovni algoritam zasnovan na susedstvu koji koristi uzajamno filtrieanje i koji uzima u obzir Z-skor normalizaciju (eng. *Z-score normalisation*) svakog korsinika. Argumenti koje prima metod KNNWithZScore su:

- k (int): Maksimalan broj suseda koji se uzima u razmatranje. Podrazumevana vrednost ovog argumenta je 40.
- *min_k* (int): Minimalan broj suseda koji se uzima u razmatranje. Ukoliko nema dovoljno suseda, predikcija postaje jednaka prosečnoj vrednosti μu ili μi. Podrazumevana vrednost ovog argumenta je 1.
- *sim_options* (dict): Rečnik opcija vezano za meru sličnosti koja se koristi.
- *verbose* (bool): Vrednost ovog argumenta govori o tome da li će se štampati prateće poruke vezano za procenu bias-a, sličnosti, itd. Podrazumevana vrednost je *True*.

Predviđena ocena $\hat{r}_{ui}$ se računa na sledeći način:

$$\hat{r}_{ui} = \mu u + \sigma u * \frac{\sum_{v \in N_i^k(u)} s(u,v) * (r_{vi} - \mu v)/\sigma v}{\sum_{v \in N_i^k(u)} s(u,v)}$$

ili

$$\hat{r}_{ui} = \mu i + \sigma i * \frac{\sum_{j \in N_u^k(i)} s(i,j) * (r_{uj} - \mu j)/\sigma j}{\sum_{j \in N_u^k(i)} s(i,j)}$$

zavisno od vrednosti polja *user_based* unutar parametra *sim_options* [12].

KNNBaseline

Osnovni algoritam zasnovan na susedstvu koji koristi uzajamno filtrieanje i koji uzima u obzir početnu ocenu (eng. *baseline rating*). Argumenti koje prima metod KNNBaseline su:

- k (int): Maksimalan broj suseda koji se uzima u razmatranje. Podrazumevana vrednost ovog argumenta je 40.
- *min_k* (int): Minimalan broj suseda koji se uzima u razmatranje. Ukoliko nema dovoljno suseda, predikcija postaje jednaka početnoj oceni (eng. *baseline rating*). Podrazumevana vrednost ovog argumenta je 1.

- *sim_ options* (dict): Rečnik opcija vezano za meru sličnosti koja se koristi. Preporuka je da se koristi *pearson_ baseline* mera sličnosti.
- *bsl_ options* (dict): Rečnik opcija na osnovu kojih se računa odnosno procenjuje početna ocena (eng. *baseline*).
- *verbose* (bool): Vrednost ovog argumenta govori o tome da li će se štampati prateće poruke vezano za procenu bias-a, sličnosti, itd. Podrazumevana vrednost je *True*.

Predviđena ocena $\hat{r}_{ui}$ se računa na sledeći način:

$$\hat{r}_{ui} = bui + \frac{\sum_{v \in N_i^k(u)} s(u,v) * (r_{vi} - bvi)}{\sum_{v \in N_i^k(u)} s(u,v)}$$

ili

$$\hat{r}_{ui} = bui + \frac{\sum_{j \in N_u^k(i)} s(i,j) * (r_{uj} - buj)}{\sum_{j \in N_u^k(i)} s(i,j)}$$

zavisno od vrednosti polja *user_ based* unutar parametra *sim_ options* [12].

SVD - pristup zasnovan na modelu

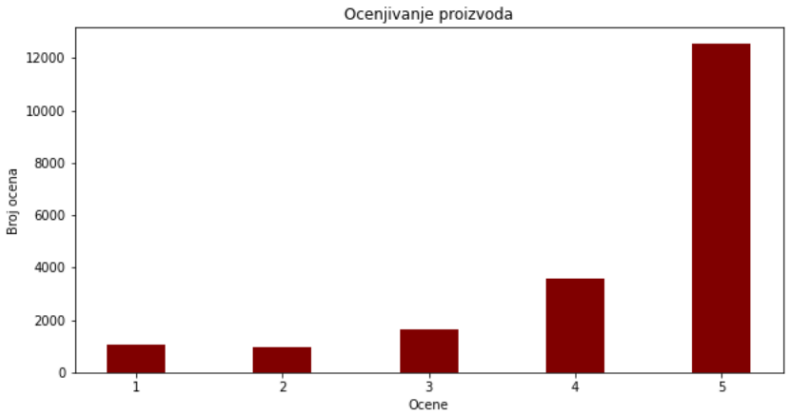
Ovaj algoritam koristi pristup faktorizacije matrica. Matrica ocena korisnik-stavka je faktorisana na matrice korisnika i stavki, manjih dimenzija, koje se sastoje od latentnih faktora (skrivenih karakteristika). Podrazumevano, broj skrivenih karakteristika je 100. Ovi latentni faktori mogu da obuhvate poznate ocene i učestvuju u procesu predviđanja ocene za sve parove korinsik-stavka gde korisnik još nije ocenio stavku [13].

Pretprocesiranje podataka

Nakon učitavanja skupa podataka, eliminisani su atributi koji nisu značajni za dalju implementaciju. Dve tabele, iz kojih se sastoji skup podataka, a koje su opisane u sekciji 4.1, su spojene, kako bi se vrednosti atributa koji predstavlja id proizvoda zamenile nazivom proizvoda (radi čitljivosti). Takođe, iz datog skupa podataka eliminisani su duplikati. Nakon ovoga, izgled skupa podataka prikazan je na slici 4.3. Grafik koji predstavlja koliko je kojih ocena dato prikazan je na slici 4.4.

	reviewerID	title	overall
0	AVH1YOFM8WG9W	Calvin Klein Men's 3-Pack Classic V-Neck T-Shirt	5
1	A39ZXN06R8UBQ0	Calvin Klein Men's 3-Pack Classic V-Neck T-Shirt	4
2	A174NYS7K8Q24K	Calvin Klein Men's 3-Pack Classic V-Neck T-Shirt	5
3	A1I32A4YCT5ITX	Calvin Klein Men's 3-Pack Classic V-Neck T-Shirt	4
4	A1I32A4YCT5ITX	Calvin Klein Men's 3-Pack Classic V-Neck T-Shirt	5
...	...	...	...
2932	AVH1YOFM8WG9W	Ray-Ban Junior Kid's RJ9506S Aviator Sunglasses	5
2933	AFGGH4YX5U2R1	i play. Girls' Baby Brim Sun Protection Hat, A...	4
2934	A2ZN3Z5U9TV4C	i play. Girls' Baby Brim Sun Protection Hat, A...	4
2935	A35KNTHJO9Q3YE	i play. Girls' Baby Brim Sun Protection Hat, A...	3
2936	A1KPFFU7NOVNCY	i play. Girls' Baby Brim Sun Protection Hat, A...	4

Slika 4.3: Skup podataka



Slika 4.4: Ocenjivanje proizvoda

Svi opisani modeli (sa podrazumevanim vrednostima parametara) su trenirani korišćenjem 5-slojne unakrsne validacije[1] (eng. *5 fold cross validation*). Za evaluaciju su korišćene metrike MAE i RMSE. Performanse modela su prikazane na slici 4.5.

Među isprobanim algoritmima, najmanju grešku RMSE ima SVD. Što se tiče varijacija KNN algoritama, najmanju grešku RMSE ima KNNBasic. Maksimalan *fit_time* je za model KNNWithZScore, ali *test_time* je najmanji.

Skup podataka je podeljen na skup za treniranje (80%) i skup za testiranje

[1]k-slojna unakrsna validacija podrazumeva podelu podataka D na k približno jednakih podskupova, takozvanih slojeva (eng. *folds*) S1, ..., Sk. Zatim se za $i = 1, ..., k$ obučava model nad podacima D. Potom se izvrše predviđanja dobijenim modelom na sloju Si. Na kraju se izračuna ocena kvaliteta na osnovu svih predviđanja na celom skupu D.

	MAE	RMSE	fit_time	test_time
matrix_factorization.SVD	0.876430	1.092983	0.146007	0.004494
knns.KNNBasic	0.873115	1.094342	0.067622	0.004021
knns.KNNBaseline	0.873433	1.098013	0.070540	0.003561
knns.KNNWithMeans	0.890062	1.125666	0.087415	0.003575
knns.KNNWithZScore	0.891067	1.126460	0.160040	0.003326

Slika 4.5: Performanse modela

(20%). Za modele KNNBasic i SVD izvršeno je podešavanje parametara pri čemu je opet korišćena 5-slojna unakrsna validacija.

Ispostavilo se da se najbolje performanse modela KNNBasic postižu sa sledećim vrednostima parametara: MSD kao mera sličnosti, vrednost 3 za *min_ support* i vrednost *False* za *user_ based*. Tada MAE iznosi 0.86647, a RMSE 1.08989.

Za model SVD najbolje performanse se postižu za ove vrednosti parametara: 30 za parametar *n_ factors*, 5 za *n_ epochs*, 0.002 za *lr_ all* i 0.5 za *reg_ all*. MAE iznosi 0.84535, a RMSE 1.07110.

Nakon što su pronađene odgovarajuće vrednosti parametara i prosleđene ovim modelima koji su trenirani na skupu podataka za trening, izvršeno je testiranje na testnom skupu podataka i evaluacija korišćenjem MAE i RMSE. Dobijene vrednosti ovih metrika za model KNNBasic iznose: 0.04426 za MAE i 0.21232 za RMSE, što je uporedivo sa vrednostima dobijenim koristeći unakrsnu validaciju na skupu za treniranje, pa se može zaključiti da model dobro generalizuje. Vrednosti metrika za model SVD, pri testiranju su: MAE iznosi 0.91220, a RMSE 1.14101, što je uporedivo sa vrednostima dobijenim koristeći unakrsnu validaciju na skupu za treniranje, što znači da i ovaj model dobro generalizuje.

Ovi modeli, čije su performanse zadovoljavajuće, su iskorišćeni za generisanje preporuka u metodama generate_recommendationsKNN i generate_recommendationsSVD. Kao mera sličnosti u algoritmu generate_recommendationsKNN korišćena je srednje kvadratna distanca *msd*.

Implementacija modela i evaluacija

Metod generate_recommendationsKNN

Ovaj metod generiše preporuke koristeći KNNBasic i filtriranje zasnovano na stavkama. Ovaj metod prima tri argumenta: id korisnika (*userID*) za koga se preporuke generišu, broj ocena (ocenjenih proizvoda od strane tog korisnika) koje se razmatraju prilikom kreiranja preporuka (*like_recommend*) i broj preporuka koje trebaju biti generisane (*get_recommend*).

Metod generate_recommendationsSVD

Ovaj metod generiše preporuke koristeći dekompoziciju singularnim vrednostima. Funkcija prima dva argumenta: id korisnika (*userID*) za koga se generišu preporuke i željeni broj preporuka (*get_recommend*) koji treba biti generisan.

Upoređivanje preporuka

Preporuke generisane koristeći KNNBasic i SVD su iste, samo je redosled, odnosno prioritet, drugačiji. Ovo je i očekivano s obzirom na to da su u pitanju različiti algoritmi.

Zaključak i sledeći koraci

Uspešno je implementiran pristup zasnovan na memoriji kao i pristup zasnovan na modelu. Međutim, s obzirom na to da modeli uzajamnog filtriranja nisu dobra opcija u slučaju novog korisnika ili nove stavke, kada je malo toga poznato u vezi preferenci i ocena, ovaj model se može unaprediti implementiranjem hibridnog pristupa, gde bi se kod predviđanja ocena u slučaju novog korisnika ili stavke iskoristio model zasnovan na sadržaju, koji je pogodniji u tom slučaju.

4.3 Implementacija modela zasnovanog na dubokom učenju korišćenjem biblioteke Keras

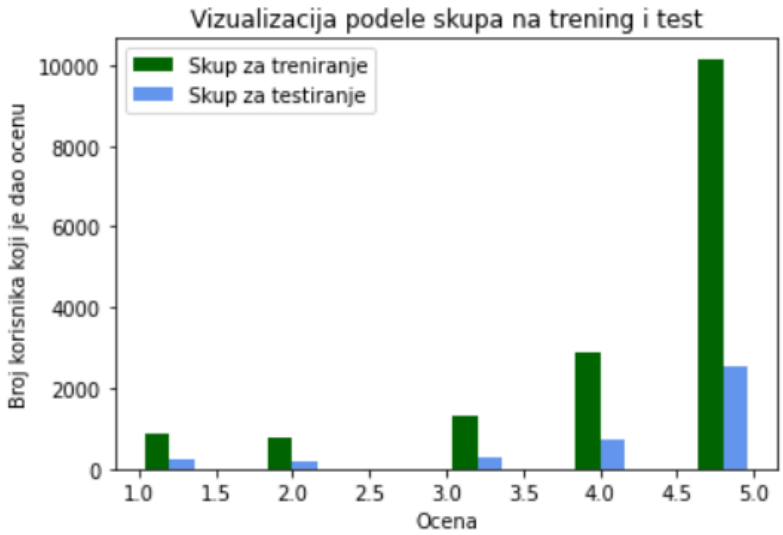
U ovoj sekciji predstavljen je model zasnovan na dubokom učenju koji koristi latentne faktore, nalik na SVD tehniku. Za implementaciju je korišćena biblioteka Keras.

Nakon učitavanja skupa podataka, uklonjeni su atributi koji nisu značajni za implementaciju, a potom je izvršeno enkodiranje vrednosti atributa koji identifikuju korisnika i proizvod. Nakon toga, podaci su podeljeni na skup za treniranje i skup za testiranje u razmeri 80:20. Ovo je opisano u sekciji 4.3. Kreirana je arhitektura modela, koji je treniran na skupu podataka za treniranje. Korišćen je različit broj epoha i različite vrednosti parametra *batch_ size*. Za evaluaciju modela korišćena je srednje kvadratna greška kao funkcija gubitka, prilikom treniranja i prilikom testiranja modela. Ovo je opisano u sekciji 4.3.

Pretprocesiranje podataka

Atributi koji predstavljaju korisnika i proizvod iz učitanog skupa podataka su enkodirani pomoću metoda **encode_user_item(df, user_col, item_col)** koja modifikuje prosleđeni skup podataka tako što dodaje još dva atributa - **User**, koji predstavlja indeks korisnika i **Item** koji predstavlja indeks proizvoda. Vrednosti novododatih atributa su izračunate pomoću metoda **LabelEncoder()** biblioteke **sklearn**, koji enkodira ne-numeričke labele (id korisnika i id proizvoda) u numeričke labele.

Skup podataka je podeljen na skupove za trening i test u razmeri 80:20. Prikaz broja korisnika koji su dali određenu ocenu, nakon podele na trening i test skup dat je na slici 4.6.



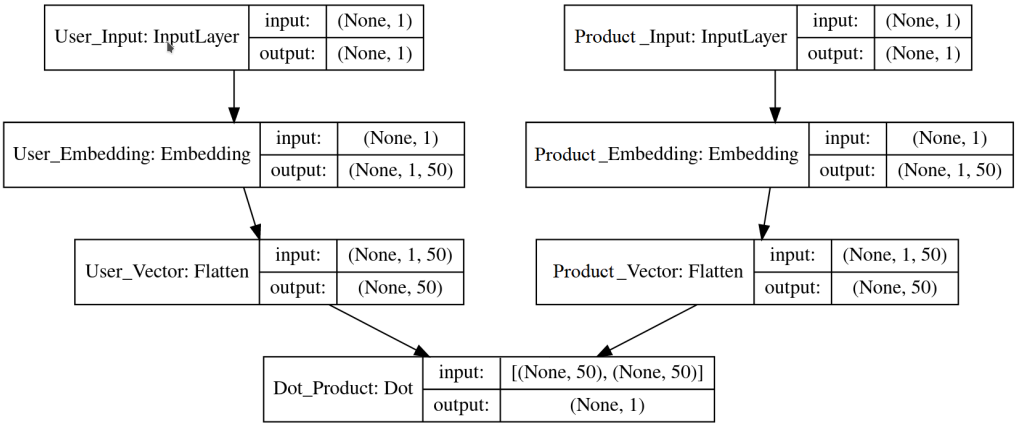
Slika 4.6: Vizuelni prikaz podele skupa podataka

Arhitektura modela i evaluacija

Arhitektura se sastoji iz tri sloja:

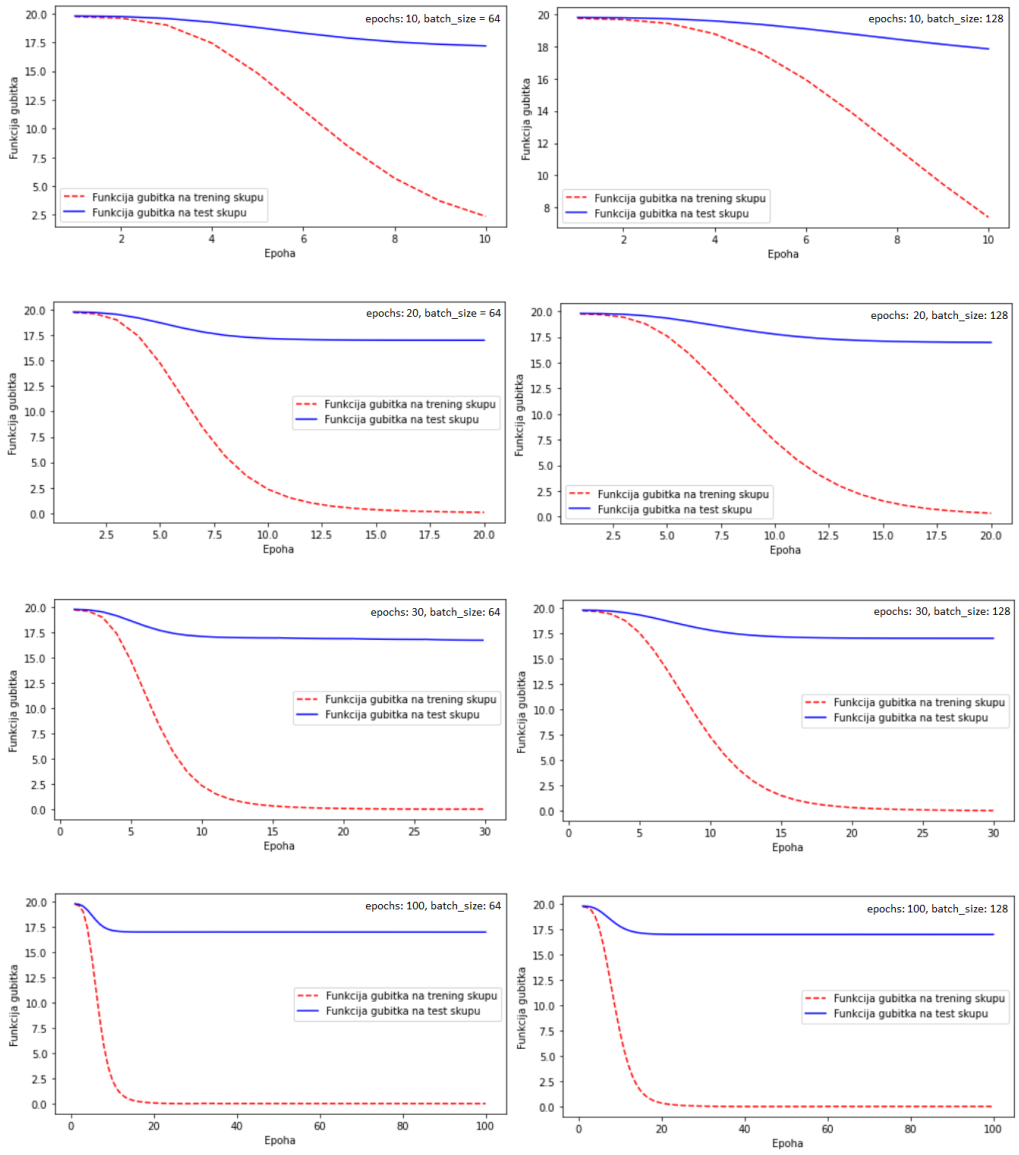
- Ulazni sloj (eng. *Input Layer*) za korisnika i za proizvod - ovaj sloj binarizuje vektore za identifikaciju korisnika i proizvoda, gde za Item(i) vrednost 1 označava da je korisnik u interagovao sa stavkom Item(i), dok User(u) identifikuje korisnika.
- Ugnježdeni sloj (eng. *Embedding layer*) je potpuno povezani sloj koji preslikava retku reprezentaciju u gust vektor. Generisani gusti vektori za korisnika i stavku se mogu tumačiti kao vektori sa latentnim karakteristikama.
- Ravnajuci sloj (eng. *Flatten Layer*) koji konvertuje multi-dimenzionalni niz u jedno-dimenzioni niz.

Na kraju je izračunat skalarni proizvod i time izračunata predviđena ocena. Arhitektura modela je prikazana na slici: 4.7.



Slika 4.7: Arhitektura mreže

Treniranje modela je izvršeno za različite vrednosti parametara *epochs*, *batch_ size*, *n_ latent_ factor* (broj latentnih faktora - karakteristika). Korišćen je *Adam* optimizator sa stopom učenja 0.0005. Za funkciju greške (eng. *Loss function*) korišćena je srednje kvadratna greška. U nastavku su prikazani grafici funkcije gubitka za različite vrednosti nabrojanih parametara, na slici 4.8.



Slika 4.8: Vizuelni prikaz funkcije gubitka za različite vrednosti parametara *epochs* i *batch_size*

4.4 Implementacija hibridnih sistema preporuka

U ovoj sekciji predložene su dve implementacije hibridnog sistema: hibrid zasnovan na težinama i hibrid sa smenjivanjem (u okviru podsekcije). Oba sistema koriste prethodno implementirane metode zasnovane na sadržaju i uzajamnom filtriranju. Fokus ovog dela rada je na razumevanju kako dva metoda mogu biti kombinovana i iskorišćena u predviđanju ocena. Takođe, ova rešenja, s obzirom na to da koriste filtriranje zasnovano na sadržaju, mogu (uz male modifikacije) biti iskorišćena

za rešavanje problema sporog početka, u situacijama kada se dodaje novi korisnik. Naime, ovaj metod radi sa ostalim atributima vezanim za proizvod, a interakcije korisnika ovde nisu od presudnog značaja. Skup podataka koji je korišćen u ostatku rada, korišćen je i ovde, u malo drugačijem formatu. Detaljniji opis podataka dat je u sekciji za pretprocesiranje (4.4). Implementacija modela i evaluacija opisana je u sekciji 4.4.

Pretprocesiranje podataka

Nakon učitavanja inicijalnog skupa podataka (dve tabele), izdvajanja atributa od značaja, uklanjanja redova koji sadrže NaN vrednosti i uklanjanja duplikata, podaci su grupisani u tri tabele: users_all, items_all i ratings_all. Atributi ovih tabela prikazani su na slikama 4.9, 4.10 i 4.11 respektivno.

	reviewerID	reviewerName	asin	reviewText	summary	overall	userID	id
0	A1D4G1SNUZWQOT	Tracy	7106116521	Exactly what I needed.	perfect replacements!!	5	1840	1
1	A3DDWDH9PX2YX2	Sonja Lau	7106116521	I agree with the other review, the opening is ...	I agree with the other review, the opening is ...	2	11443	2

Slika 4.9: Tabela users_all

	asin	title	brand	dimensions	weight	itemID	id
0	0764443682	Slime Time Fall Fest [With CDROM and Collector...	Group Publishing (CO)	Product Dimensions:\n \n8.7...	Shipping Weight:\n \n2.4 po...	0	1
2	1940280001	Magical Things I Really Do Do Too!	Christopher Manos	Package Dimensions:\n \n8.5...	Shipping Weight:\n \n6.1 ou...	1	2

Slika 4.10: Tabela items_all

	reviewerID	userID	asin	itemID	overall
0	A1BB77SEBQT8VX	1591	B00007GDFV	11	3
1	AHWOW7D1ABO9C	15861	B00007GDFV	11	3

Slika 4.11: Tabela ratings_all

Implementacija modela i evaluacija

Implementacija metode koja koristi uzajamno filtriranje je prikazana na slici 4.12. Ovaj metod za predviđanje ocena koristi KNNBasic algoritam biblioteke Surprise. Metod prima tri parametara: tabelu ocena, id korisnika i id stavke. Povratna vrednost metoda je predviđena ocena, razlika između predviđene i stvarne ocene i stvarna ocena.

Implementacija metode koja koristi filtriranje zasnovano na sadržaju prikazano je na slici 4.13. Ovaj metod za predviđanje ocena koristi klasifikator KNeighborsClassifier biblioteke sklearn. Metod prima četiri parametara: tabelu ocena, tabelu stavki, id korisnika i id stavke. Povratna vrednost metoda je predviđena ocena, razlika između predviđene i stvarne ocene i stvarna ocena.

```
def cf_predict(ratings, user_id, item_id):
    is_target = (ratings['userID'] == user_id) & (ratings['itemID'] == item_id)
    target = ratings[is_target].iloc[0]

    train_set = sp.Dataset.load_from_df(
        ratings[~is_target][['userID', 'itemID', 'overall']],
        sp.Reader(rating_scale=(0, 100))
    ).build_full_trainset()

    algo = sp.KNNBasic(verbose=False)
    algo.fit(train_set)
    prediction = algo.predict(target['userID'], target['itemID'], verbose=False)

    return prediction.est, prediction.est - target['overall'], target['overall']
```

Slika 4.12: Metod za predviđanje ocene zasnovan na uzajamnom filtriranju

```
def cb_predict(ratings, items, user_id, item_id):

    user_ratings = ratings[ratings['userID'] == user_id].join(items.set_index('id'),
                                                              on='itemID', how='left',
                                                              lsuffix='_left', rsuffix='_right')
    is_target = (user_ratings['itemID_left'] == item_id)

    features = pd.get_dummies(user_ratings.drop(columns=['overall']))
    train_features = features[~is_target]
    target_features = features[is_target]

    encoder = LabelEncoder()
    train_labels = encoder.fit_transform(user_ratings[~is_target]['overall'])
    target_label = user_ratings[is_target]['overall'].iloc[0]

    # kreiranje i treniranje modela
    clf = KNeighborsClassifier(n_neighbors=1)
    clf.fit(train_features, train_labels)

    # predvidjanje ocene
    prediction = encoder.inverse_transform(clf.predict(target_features))[0]

    return prediction, prediction - target_label, target_label
```

Slika 4.13: Metod za predviđanje ocene zasnovan na sadržaju

Implementacija hibridnog sistema koji koristi težine, prikazana je na slici 4.14 a rezultati korišćenja ovog sistema na slici 4.15. Implementacija hibridnog sistema sa smenjivanjem prikazana je na slici 4.16, a rezultati na slici 4.17 i 4.18. Pristup zasnovan na težinama dodaje vrednosti (značaj/težine) predviđenim ocenama od strane modela zasnovanog na uzajamnom filtriranju, odnosno modela zasnovanog na sadržaju. Pristup sa smenjivanjem koristi model zasnovan na uzajamnom filtriranju samo ukoliko posoji dovoljan broj ocena (više od 3) za datu stavku (čiji se id prosleđuje metodu). Inače, koristi model zasnovan na sadržaju. Modeli su evaluirani tako što su izračunate razlike predviđene i stvarne ocene. Što je razlika manja, predikcija je bolja. S obzirom na to da je skup podataka relativno mali, nije bilo

prostora za eksperimentisanje sa različitim vrednostima parametra koji predstavlja granicu koja određuje kada se koristi model zasnovan na uzajamnom filtriranju, a kada model zasnovan na sadržaju.

```
def predict_weighted(ratings, items, user_id, item_id):
    prediction_cf, _, true_rating = cf_predict(ratings, user_id, item_id)
    prediction_cb, _, true_rating = cb_predict(ratings, items, user_id, item_id)

    # Težine mogu biti odabrane drugačije,
    # prema tome koji model daje bolje preporuke.

    prediction = 0.5 * prediction_cf + 0.5 * prediction_cb
    error = prediction - true_rating

    return prediction, error, true_rating
```

Slika 4.14: Hibridni sistem zasnovan na težinama

```
Hibrid sa težinama:      predviđena ocena: 4.56250       greška: -0.43750
```

Slika 4.15: Hibridni sistem zasnovan na težinama - rezultati

```
def predict_switching(ratings, items, user_id, item_id):

    # Odabir modela (sistema preporuka) je izvršen na osnovu
    # broja ocena koje su dostupne za datu stavku.

    num_ratings = len(ratings[ratings['itemID'] == item_id])
    if num_ratings > 3:
        print('Korišćenjem sistema zasnovanog na uzajamnom filtriranju')
        return predict_cf(ratings, user_id, item_id)
    else:
        print('Korišćenjem sistema zasnovanog na sadržaju')
        return predict_cn(ratings, items, user_id, item_id)
```

Slika 4.16: Hibridni sistem sa smenjivanjem

```
Korišćenjem sistema zasnovanog na uzajamnom filtriranju
Hibrid sa smenjivanjem:          predviđena ocena: 4.12500       greška: -0.87500
```

Slika 4.17: Hibridni sistem sa smenjivanjem - rezultati

```
Korišćenjem sistema zasnovanog na sadržaju
Hibrid sa smenjivanjem:          predviđena ocena: 2.00000       greška: -3.00000
```

Slika 4.18: Hibridni sistem sa smenjivanjem - rezultati

4.5 Implementacija sistema preporuka zasnovanih na korisnicima i stavkama

U ovoj sekciji predložene su dve implementacije sistema preporuka: sistem zasnovan na korisnicima i sistem zasnovan na stavkama. Fokus ovog dela rada je na razumevanju ova dva pristupa i implementaciji korak po korak, bez korišćenja postojećih metoda neke od biblioteka. Skup podataka koji je korišćen za potrebe ove implementacije sastoji se iz tabele sa ocenama proizvoda koje su im korisnici dodelili. Razlog korišćenja drugačijeg skupa podataka u odnosu na ostatak rada je što se za ovaj skup podataka dobijaju reprezentativniji rezultati. Skup podataka i matrica ocena kreirana na osnovu ovog skupa, prikazani su na slikama 4.19 i 4.20 u okviru sekcije za pretprocesiranje podataka (4.5). Implementacija modela i evaluacija opisana je u sekciji 4.5.

Pretprocesiranje podataka

Nakon učitavanja skupa podataka (tabela na slici 4.19), kreirana je matrica ocena i sve NaN vrednosti su zamenjene nulom. S obizirom na to da je dobijena retka matrica jer većina polja ima vrednost nula, ona je transformisana u retku matricu biblioteke scipy. Korišćenje retkih matrica za čuvanje podataka koji sadrže veliki broj nula elemenata štedi memoriju i ubrzava procesiranje tih podataka. Skup podataka i matrica ocena prikazani su na slikama 4.19 i 4.20, respektivno.

Iz matrice ocena u obzir su uzeti samo redovi sa više od 700 ocena, kako bi modelu bilo pruženo što više ocena. Zbog velikog vremena izvršavanja, urađena je i redukcija dimenzija ove matrice. Nakon toga izvršena je podela skupa podataka na skupove za trening (i validaciju) i test. Metodi za filtriranje i podelu skupa podataka prikazani su na slikama 4.21 i 4.22, respektivno.

Implementacija modela i evaluacija

Metodi korišćeni za implementaciju pristupa zasnovanog na korisnicima i stavkama prikazani su na slikama u nastavku. Metrika za merenje tačnosti predviđenih ocena je RMSE. Na osnovu grafičkog prikaza rezultata (4.27) može se zaključiti da pristup zasnovan na korisnicima predviđa bolje.

	userId	itemId	rating
0	1	1	4.0
1	1	3	4.0
2	1	6	4.0
3	1	47	5.0
4	1	50	5.0
...	...	...	...
100831	610	166534	4.0
100832	610	168248	5.0
100833	610	168250	5.0
100834	610	168252	5.0
100835	610	170875	3.0

100836 rows × 3 columns

Slika 4.19: Skup podataka - tabela ocena

itemId	1	2	3	4	5	6	7	8	9	10	...	193565	193567	193571	193573	193579	193581	193583	193585	193587	193609
userId																					
1	4.0	0.0	4.0	0.0	0.0	4.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
2	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
3	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
4	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0
5	4.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	...	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0	0.0

5 rows × 9724 columns

Slika 4.20: Matrica ocena

```python
def filter_users(train_and_validation, m):
    """
    Funkcija koja filtrira matricu ocena tako da ostaju redovi koji imaju više od m ne-nula elemenata.

    Parametri:
    train_and_validation: skup podataka
    m: broj ne-nula elemenata
    """
    xy = train_and_validation.toarray()
    xy_filtered_matrix = []
    for y in xy:
        sum = 0
        nz = np.count_nonzero(y)
        if (nz > m):
            xy_filtered_matrix.append(y)

    arr_shape = np.vstack(xy_filtered_matrix).shape
    filtered_arr = np.vstack(xy_filtered_matrix)

    return sps.csr_matrix(filtered_arr)
```

Slika 4.21: Filtriranje korisnika

```python
def split_train_test(ratings_csr_matrix, percentage):
    """
    Ova funkcija vrši podelu datog skupa podataka na skup za trening i skup za test
    tako što se iz datog skupa "uzima" procenat ocena koje se upisuju u test skup,
    a uklanjaju iz trening skupa.

    Parametri:
    ratings_csr_matrix: retka matrica ocena - skup podataka koji treba podeliti na trening i test skup
    percentage: broj koji predstavlja procenat ocena koje ulaze u test skup
    """
    test_ratings_number = percentage / 100
    print("Odnos ocena u skupu za testiranje: ", percentage, "%")
    print("Odnos ocena u skupu za treniranje: ", 100-percentage, "%")

    total_ratings = ratings_csr_matrix.toarray()

    dimensions_of_total_ratings = total_ratings.shape
    print("Ukupan broj korisnika: ", dimensions_of_total_ratings[0])
    print("Ukupan broj stavki: ", dimensions_of_total_ratings[1])

    test = np.zeros(dimensions_of_total_ratings)
    train = total_ratings.copy()

    nonzero_ratings_per_row = (total_ratings != 0).sum(1)
    print("Ukupan broj ne-nula ocena u svim redovima: \n", nonzero_ratings_per_row)

    for user in range(dimensions_of_total_ratings[0]):

        # indeksi ne-nula elemenata
        nonzero_test_ratings_per_user = int(np.ceil(test_ratings_number*nonzero_ratings_per_row[user]))

        test_ratings = np.random.choice(total_ratings[user, :].nonzero()[0], size = nonzero_test_ratings_per_user, replace = False)

        # ocene se izbacuju iz trening skupa (upisuju se nule na odabranim pozicijama)
        train[user, test_ratings] = 0

        # ubacuju se u test skup (upisuju se ocene iz polaznog skupa, koje su uklonjene iz trening skupa)
        test[user, test_ratings] = total_ratings[user, test_ratings]

    if (not(np.all((train * test) == 0))):
        print("Greška!")
    else:
        return sps.csr_matrix(train),sps.csr_matrix(test)
```

Slika 4.22: Podela skupa podataka

```
def selection(train_and_validation, k_values, cf_type):
    """
    Funkcija vrši predviđanje ocena za sve korisnike i sve stavke
    za različite vrednosti parametra k (različit broj suseda).

    Parametri:
    train_and_validation: skup podataka
    k_values: lista sa brojevima suseda
    cf_type: tip - zasnovano na korisnicima ili stavkama
    """
    print('k_values = ', k_values)
    train, validation = split_train_test(train_and_validation, 20)
    errors = np.array([])
    predictions_arr = np.array([])

    if cf_type == 'user':

        similarity = cosine_similarity(train) + EPS

        for k in k_values:
            print('k = ', k)
            predictions = get_prediction(train, similarity, 'user', k)
            error = np.sqrt(mean_squared_error(validation.toarray()[validation.nonzero()],predictions[validation.nonzero()]))

            errors = np.append(errors, error)
            predictions_arr = np.append(predictions_arr, predictions)
    else:

        similarity = cosine_similarity(train.T) + EPS

        for k in k_values:
            print('k = ', k)
            predictions = get_prediction(train, similarity, 'item', k)
            error = np.sqrt(mean_squared_error(validation.toarray()[validation.nonzero()],predictions[validation.nonzero()]))

            errors = np.append(errors, error)
            predictions_arr = np.append(predictions_arr, predictions)

    k_optimal = k_values[np.argmin(errors)]

    if cf_type == 'user':
        similarity = cosine_similarity(train_and_validation) + EPS
        predictions = get_prediction(train_and_validation, similarity, 'user', k_optimal)
        return errors, predictions
    else:
        similarity = cosine_similarity(train_and_validation.T) + EPS
        predictions = get_prediction(train_and_validation, similarity, 'item', k_optimal)
        return errors, predictions
```

Slika 4.23: Metod *selection*

```python
def get_prediction(ratings, similarity_matrix, cf_type, k = 5):
    """
    Funkcija za predviđanje ocena svih korisnika za sve stavke
    koja primenjuje user-based ili item-based pristup,
    zavisno od prosleđenog parametra.

    Parametri:
    ratings: matrica ocena
    similarity_matrix: matrica sličnosti
    cf_type: tip - zasnovano na korisnicima ili stavkama
    k: broj suseda
    """
    predictions = np.zeros(ratings.shape)

    if cf_type == 'user':

        for u in range(ratings.shape[0]):
            # print('Predvidjanje ocene za korisnika u = ', u)
            for i in range(ratings.shape[1]):
                # print('Predvidjanje ocene za korisnika u = {} i stavku i = {}.'.format(u, i))
                predictions[u,i] = user_based_ratings_prediction(u, i, similarity_matrix , ratings, k)

    elif cf_type == 'item':

        for u in range(ratings.shape[0]):
            # print('Predvidjanje ocene za korisnika u = ', u)
            for i in range(ratings.shape[1]):
                # print('Predvidjanje ocene za korisnika u = {} i stavku i = {}.'.format(u, i))
                predictions[u,i] = item_based_ratings_prediction(u, i, similarity_matrix, ratings, k)

    else:
        print("Greška! Tip mora biti user ili item.")
        return

    return predictions
```

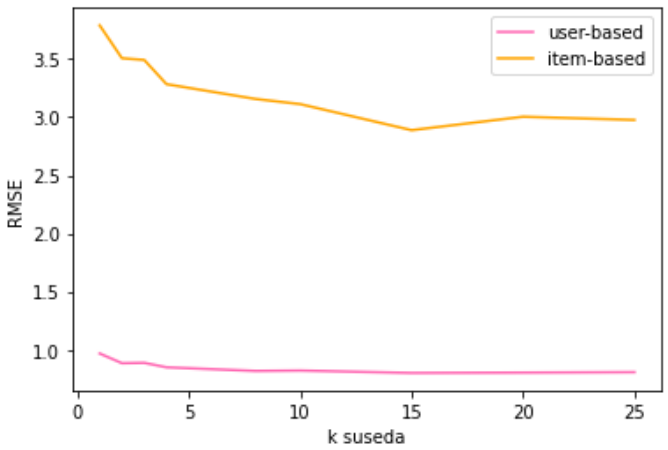
Slika 4.24: Metod *get_prediction*

```python
def user_based_ratings_prediction(u, i, users_similarity, ratings, k = 5):
    """
    Funkcija koja računa ocenu korisnika u za stavku i na osnovu matrice sličnosti korisnika.

    Parametri:
    u: id korisnika
    i: id stavke
    users_similarity: matrica sličnosti korisnika
    ratings: matrica ocena
    k: broj suseda
    """
    neighbors = []
    similarities = list(zip(users_similarity[u][:], range(users_similarity.shape[0])))

    similarities_sorted = sort_descending(similarities)

    # najslicnijih k
    for i in range(1, k + 1):
        neighbors.append(similarities_sorted[i][1])
    # print ("neighbors: ", neighbors)

    rated_by_u = ratings[u].nonzero()[1]
    # print("Ocenio korisnik u: ", rated_by_u)

    user_u_mean = 0
    arr = ratings[u, :].toarray()[0]
    user_u_mean = np.sum(arr)

    if len(rated_by_u) != 0:
        user_u_mean = user_u_mean / len(rated_by_u)

    numerator, denominator = 0.0, 0.0

    for v in neighbors:
        rated_by_v = ratings[v].nonzero()[1]
        user_v_mean = 0
        # user_v_mean = sum(ratings[v, :].toarray()[0])

        if len(rated_by_v) != 0:
            for i in rated_by_v:
                user_v_mean += ratings[v, i]

            user_v_mean = user_v_mean / len(rated_by_v)


        r_vi = ratings[v,i]
        numerator += users_similarity[u][v]*(r_vi - user_v_mean)
        denominator += users_similarity[u][v]

    return user_u_mean + numerator/denominator
```

Slika 4.25: Predviđanje zasnovano na korisnicima

```
def item_based_ratings_prediction(u, i, items_similarity, ratings, k = 5):
    """
    Funkcija računa ocenu korisnika u za stavku i na osnovu matrice sličnosti stavki.

    Parametri:
    u: id korisnika
    i: id stavke
    items_similarity: matrica sličnosti stavki
    ratings: matrica ocena
    k: broj suseda
    """
    neighbors = []
    similarities = list(zip(items_similarity[i][:],range(items_similarity.shape[0])))
    similarities_sorted = sort_descending(similarities)

    for i in range(1,k+1):
        neighbors.append(similarities_sorted[i][1])

    rated_i = ratings[:, i].nonzero()[0]

    item_i_mean = 0
    item_i_mean = np.sum(ratings[:, i].toarray()[0])

    if len(rated_i) != 0:
        item_i_mean = item_i_mean / len(rated_i)

    numerator, denominator = 0.0, 0.0

    for j in neighbors:
        rated_j = ratings[:, j].nonzero()[0]
        item_j_mean = 0

        if len(rated_j) != 0:
            arr2 = ratings[:, j].toarray()[0]
            item_j_mean = np.sum(arr2)
            item_j_mean = item_j_mean / len(rated_j)

        r_uj = ratings[u,j]
        numerator += items_similarity[i][j]*(r_uj - item_j_mean)
        denominator += items_similarity[i][j]

    return item_i_mean + numerator/denominator
```

Slika 4.26: Predviđanje zasnovano na stavkama



Slika 4.27: Grafički prikaz greške RMSE za pristup zasnovan na stavkama i pristup zasnovan na korisnicima

Glava 5

Zaključak

Kao što je pomenuto na početku rada, sistemi preporuka su široko zastupljeni u elektronskoj trgovini. Precizno predviđanje i generisanje preporuka može da dovede do poboljšanja biznisa, a samim time i do povećanja profita. Prilikom implementacija ovih sistema nailazi se na različite probleme koji uključuju oskudnost podataka, nove korisnike, preporučivanje novih proizvoda i drugo. Pored toga što sistem preporuke treba da prevaziđe sve ove probleme, on treba da radi brzo, efikasno i tačno nad realnim skupovima podataka ogromne veličine.

U ovom radu su opisani različiti pristupi za implementaciju sistema preporuka koji koriste uzajamno filtriranje, ali i filtriranje zasnovano na sadržaju i duboko učenje. Ovi sistemi rade sa skupom proizvoda koji se prodaju na Amazonu. S obzirom na to da sistemi preporuka generalno donose bolje zaključke što im je više ulaznih podataka prosleđeno, a da je u ovom radu korišćeno relativno malo podataka, preptpostavka je da bi implementirani pristupi postizali još bolje rezultate nad većim skupom podataka. Implementacija pomenutih pristupa je izvršena u Python programskom jeziku uz odgovarajuću vizualizaciju podataka. Neki pristupi daju odlične rezultate, dok su rezultati dobijeni drugim pristupima nezadovoljavajući. Za implementaciju su korišćene biblioteke Surprise, scikit learn, Keras i druge biblioteke koje se standardno koriste za učitavanje, pretprocesiranje i vizualizaciju podataka.

Postoji prostor za usavršavanje razvijenih sistema preporuka. Rad sa većim skupom podataka, odnosno obogaćivanje baze podataka može doprineti dobijanju boljih rezultata. Dodavanjem novih atributa ili korišćenjem drugačijih atributa, posebno za učenje modela zasnovanog na sadržaju, može dovesti do tačnijih predviđanja ocena. Takođe nešto što nije razmatrano, jeste efekat promene preferenci korisnika. U „živom" sistemu preporuka, korisnici konstantno dodaju nove i menjaju date

ocene. Onda kada se određeni broj ovih podataka promeni, treba izvršiti preračunavanje. Sam prag na osnovu kojeg se određuje kada se vrši preračunavanje, treba biti pažljivo odabran. Moguće je isprobati i druge varijacije predloženih modela i eksperimentisati sa različitim vrednostima njihovih parametara. Kombinovanjem više različitih modela u okviru hibridnog sistema, moguće je doći do unapređenog sistema za generisanje preporuka.

Bibliografija

[1] Ricci, F., Rokach, L., Shapira, B. (2011). Introduction to recommender systems handbook. In Recommender systems handbook (pp. 1-35). Springer, Boston, MA

[2] Charu, C. A. (2016). Recommender Systems: The Textbook.

[3] A Survey of Collaborative Filtering Techniques, Xiaoyuan Su i Taghi M. Khoshgoftaar, Department of Computer Science and Engineering, Florida Atlantic University, 777 Glades Road, Boca Raton, FL 33431, USA, 2009

[4] Recommender Systems in E-Commerce, J. Ben Schafer, Joseph Konstan i John Riedl, Department of Computer Science and Engineering, University of Minnesota, Minneapolis, MN 55455, 1-612-625-4002, 1999

[5] Semantics-aware Content-based Recommender Systems, Marco de Gemmis, Pasquale Lops, Cataldo Musto, Fedelucio Narducci i Giovanni Semeraro, Department of Computer Science, University of Bari "Aldo Moro", Italy, 2015

[6] An Analysis of Memory Based Collaborative Filtering Recommender Systems with Improvement Proposals, Claudio Adrian Levinas, UPC, URV, UB, 2014

[7] Hybrid Recommender Systems: Survey and Experiments†, Robin Burke, California State University, Fullerton, Department of Information Systems and Decision Sciences, 2002)

[8] Algorithms for collaborative filtering in Point-of-Interest Recommendation Systems, Bc. Guzel Samigullina, Faculty of Information Technology, CTU in Prague, Department of Software Engineering, 2019

[9] TV Personalization System, John Zimmerman, Kaushal Kauapati, Anna L. Buczak i Dave Schaffer, Carnegie Mellon University, IBM Corporation, Lockheed Martin Corporation, Philips Research, MIT, 2004

[10] Recommendation system using collaborative filtering, Yunkyoung Lee, The Faculty of the Department of Computer Science, San Jose State University, 2015

[11] A Scalable Recommender System Using Latent Topics and Alternating Least Squares Techniques, András Péter Kolbert, NOVA Information Management School, Instituto Superior de Estatística e Gestão de Informação, Universidade Nova de Lisboa, 2017

[12] Dokumentacija biblioteke Surprise (KNN), onlajn na: https://surprise.readthedocs.io/en/stable/knn_inspired.html

[13] Dokumentacija biblioteke Surprise (SVD), onlajn na: https://surprise.readthedocs.io/en/stable/matrix_factorization.html

Biografija autora

Jovana Pejkić *(Negotin, 6. decembar 1996.)* je diplomirani informatičar. Pohađala je Osnovnu školu "Vuk Karadžić" u Negotinu od 2003. do 2011. godine koju je završila sa odličnim uspehom. Negotinsku gimnaziju je upisala 2015. godine i završila 2019. godine, takođe sa odličnim uspehom. Godine 2020. završila je Matematički fakultet, Univerziteta u Beogradu, na smeru Informatika.