Ostvarenje sustava preporučivanja koristeći algoritam matrične faktorizacije

Vrankić, Fran

SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

DIPLOMSKI RAD br. 745

OSTVARENJE SUSTAVA PREPORUČIVANJA KORISTEĆI ALGORITAM MATRIČNE FAKTORIZACIJE

Fran Vrankić

Zagreb, veljača 2025.

SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

DIPLOMSKI RAD br. 745

OSTVARENJE SUSTAVA PREPORUČIVANJA KORISTEĆI ALGORITAM MATRIČNE FAKTORIZACIJE

Fran Vrankić

Zagreb, veljača 2025.

SVEUČILIŠTE U ZAGREBU
FAKULTET ELEKTROTEHNIKE I RAČUNARSTVA

Zagreb, 30. rujna 2024.

DIPLOMSKI ZADATAK br. 745

Pristupnik:	**Fran Vrankić (0036500702)**
Studij:	Računarstvo
Profil:	Računarska znanost
Mentor:	izv. prof. dr. sc. Klemo Vladimir

Zadatak: **Ostvarenje sustava preporučivanja koristeći algoritam matrične faktorizacije**

Opis zadatka:

Proučiti i opisati sustave preporučivanja s naglaskom na postupke suradničkog filtriranja. Posebno opisati postupak izgradnje sustava preporučivanja primjenom algoritma matrične faktorizacije te opisati njegovu matematičku podlogu, uporabu u sustavima preporučivanja i implementaciju u Surprise knjižnici programskog jezika Python. Implementirati vlastiti sustav preporučivanja koristeći algoritam matrične faktorizacije i osnovni postupak suradničkog filtriranja, kao i mjere uspješnosti RMSE i MAE. Opisati arhitekturu programskog rješenja, a rezultate izvođenja nad MovieLens skupom podataka detaljno prikazati i usporediti sa standardnom implementacijom algoritma i osnovnim postupkom suradničkog filtriranja.

Rok za predaju rada: 14. veljače 2025.

Sadržaj

1. Uvod

Ubrzanim rastom digitalizacije sve je više dostupnih opcija i sadržaja na dohvat ruke. Javlja se potreba približiti kupcima one sadržaje koji ih interesiraju. Takav postupak naziva se filtracijom, a cilj ovo rada je implementacija sustava preproučivanja temeljenog na metodama filtracije. Sustavi preporučivanja analiziraju korisničke preference i daju prikladne osobne preporuke. Platforme za reprodukciju muzike, filmova i serija te velike web-trgovine samo su jedni od primjera tvrtki koje koriste metode filtriranja u svrhu boljeg pružanja usluge. Jedna od prednosti sustava preporuka je njihova sposobnost iskorištavanja fenomena dugog repa (distribucije), predlažući specijalizirane stavke koje možda nisu široko popularne, ali su vrlo relevantne za određene korisnike specifičnog ukusa [1]. Veća je šansa da će se korisniku svidjeti neka stavka preko personalizirane preporuke, nego da mu je predložen samo skup najpopularnijih stavki. Uz cilj implementacije relevantnog sustava preporuke, drugi ciljevi su: objasniti teoriju i načela rada sustava preporučivanja, razmotriti izazove i mogućnosti metoda filtriranja te analizirati osnovne i naprednije algoritme preporučivanja, odnosno filtriranja.

2. Sustavi preporučivanja

2.1. Motivacija

Sustavi preporučivanja su algoritmi koji analiziraju korisničke preferencije i na temelju njih im daju personalizirane preporuke. S ekspanzijom internetske ponude i sve većom dostupnošću informacija, javila se potreba za učinkovitim filtriranjem sadržaja kako bi se relevantne ponude preciznije usmjerile prema krajnjim korisnicima. Budući da nisu svi proizvodi i usluge jednako relevantni za svakog pojedinca, veća je vjerojatnost da će osoba zainteresirana za, primjerice, fotografiju kupiti profesionalnu kameru ili objektiv, nego opremu za ribolov. Za razliku od fizičkih prodavaonica, koje su ograničene kapacitetom i izložbenim prostorom, internetske trgovine nemaju takva ograničenja te mogu ponuditi praktički neograničen asortiman proizvoda. Međutim, upravo ta obilnost može postati "dvosjekli mač". Prevelik izbor može preplaviti korisnike i otežati im donošenje odluke, što može rezultirati suprotnim učinkom od željenog. Ovaj prijelaz iz oskudice u obilje zahtijeva učinkovite sustave preporučivanja kako bi se korisnicima omogućilo brzo i precizno pronalaženje relevantnih proizvoda ili sadržaja [2]. Nema smisla korisniku preporučivati svaku dostupnu stavku, posebno ako uzmemo u obzir popularnu platformu *YouTube*, na koju se svake minute učitava čak petsto minuta novog sadržaja [3]. Kako bi korisnici dobili relevantne preporuke prilagođene njihovim interesima, nužno je primijeniti učinkovite algoritme filtriranja i personalizacije sadržaja. Sustavi preporučivanja zapravo pokušavaju "pogoditi" što bi korisnik mogao konzumirati, a to rade na temelju prikupljenih povratnih informacija od strane korisnika. Najveći problem je upravo prikupljanje tih informacija, jer korisnik rijetko ostavlja recenzije. Pritom je važno razlikovati dvije vrste povratnih informacija: eksplicitne i implicitne.

2.2. Povratne informacije korisnika

Eksplicitne povratne informacije predstavljaju ocjene koje korisnici izravno daju kako bi izrazili svoje zadovoljstvo nekom stavkom. Primjerice broj zvjezdica na skali od 1 do 5 dodijeljen nakon kupnje proizvoda, ocjene u obliku "sviđa mi se" i "ne sviđa mi se" nakon gledanja filmskog sadržaja i slično. Ovakve povratne informacije pružaju detaljan uvid u to koliko se korisniku određena stavka svidjela i najpoželjnije su za sustave preporučivanja, no njihovo prikupljanje predstavlja izazov jer, kao što je već spomenuto, većina korisnika se rijetko odluči za ovakvu recenziju. S druge strane, implicitne povratne informacije temelje se na pretpostavci da interakcije korisnika s određenim stavkama direktno odražavaju njihove preferencije. Ovakav tip povratnih informacija izuzetno je obilniji od eksplicitnih, no istovremeno manje precizan jer nema skalu po kojoj se da zaključiti koliko se točno korisniku nešto svidjelo i često sadrži šum. Primjerice, korisnik može kupiti proizvod kao poklon, a ne za osobnu upotrebu, što može dovesti do pogrešnih zaključaka o njegovim preferencijama [4]. Unatoč tome, s obzirom na ogromnu količinu dostupnih podataka ove vrste, šum u podacima može postati zanemariv [3].

2.3. Problemi

Od glavnih problema s kojima se sustavi preporučivanja susreću treba izdvojiti problem "hladnog početka" (eng. *cold start*). Teško je stvoriti preporuku za korisnika koji je tek otvorio račun i ne znaju se njegove preferencije, tako i za stavku koja je tek dodana u katalog, zato što nedostaje dovoljno podataka za analizu. Algoritmi preporučivanja moraju biti optimizirani i dobro prilagođeni skupu podataka kako bi se dobro nosili s problemom skalabilnosti. Broj korisnika i stavki se stalno povećava i time se usporava izračun sličnosti potrebnih za stvaranje predikcije. Postoji i problem pristranosti popularnih stvari (eng. *popularity bias*) čiji se učinak vidi na način da sustav preporučivanja favorizira popularnije stavke, dok manje poznate stavke ostaju zanemarene. To se događa jer popularnije stavke imaju više interakcija i ocjena (povećava se algoritamska pristranost takvim stavkama), a samim time drugim se korisnicima češće preporučuju. Tako dolazimo do "začaranog kruga", gdje manje poznate stavke ostaju u sjeni, a korisnici ostaju uskraćeni za manje poznate opcije koje bi im se zapravo svidjele. Sustav naprosto ne daje dovoljno raznolike preporuke. Drugi opće poznati problem je problem šupljikavosti (eng. *data*

sparsity), odnosno nedostatka ocjena.

Sustavi preporučivanja se generalno baziraju na matricu ocjena gdje su redovi korisnici, a stupci stavke. Kako bismo mogli dobiti što preciznije rezultate, potrebno je prikupiti što više ocjena u toj matrici. Povratne informacije korisnika daju nam referentne vrijednosti za učenje i evaluiranje modela. U slučaju eksplicitnih ocjena (tablica 2.1.) matrica sadrži brojeve, primjerice brojeve od 1 do 5 kao preslika broja zvjezdica koje je korisnik dao određenoj stavci. Kod implicitnih ocjena (tablica 2.2.) ta matrica sadrži indikatore, primjerice 0 i 1, koje predstavljaju manjak ili postojanje interakcije korisnika i stavke. Valja primijetiti kako će ta matrica gotovo uvijek bit šupljikava, odnosno neće imati puno poznatih vrijednosti. To je zato što će korisnik imati interakciju s malim brojem stavci u odnosu na cijeli skup stavci, a za još manji broj njih će ostaviti eksplicitnu recenziju. Sustavi preporučivanja dijele se na dvije skupine: sustavi temeljeni na sadržaju (eng. *content-based*) i sustavi temeljeni na metodama suradničkog filtriranja (eng. *collaborative filtering*).

Tablica 2.1. Tablica matrice eksplicitnih ocjena

	Stavka 1	**Stavka 2**	**Stavka 3**	**Stavka 4**	**Stavka 5**
Ana	-	4	-	-	5
Ivan	-	-	3	-	2
Petra	5	-	-	2	4
Marko	-	3	-	4	-
Luka	1	-	2	-	-

Tablica 2.2. Tablica matrice implicitnih ocjena

	Stavka 1	**Stavka 2**	**Stavka 3**	**Stavka 4**	**Stavka 5**
Ana	0	1	0	0	1
Ivan	0	0	1	0	1
Petra	1	0	0	1	1
Marko	0	1	0	1	0
Luka	1	0	1	0	0

2.4. Sustavi preporučivanja temeljeni na sadržaju

Sustavi temeljeni na sadržaju opisuju interakciju korisnika i stavci preko metapodataka. Svaka stavka je predstavljena vektorom atributa, a svaki korisnik s korisničkim profilom. Metapodaci za stavke mogu biti atributi poput žanra, glavnih glumačkih uloga i trajanja, ako su stavke filmovi. Korisnički profil može sadržavati podatke poput dobi korisnika, spolu, državi i slično. Sustav zatim traži slične stavke onoj koja se svidjela korisniku na način da uspoređuje vektore atributa različitih stavki. Predikcija za korisnika može se temeljiti na prosjeku vektora atributa od stavki pozitivno recenziranih od strane tog korisnika, a može se i heuristički odrediti s pomoću sličnosti kosinusa vektora atributa i korisničkog profila. Konačna preporuka će biti primjerice skup od pet najsličnijih stavki onoj za koju je korisnik izrazio najveće zanimanje. Prednosti ovakvog pristupa su što ne zahtijeva podatke o drugim korisnicima, već se oslanja isključivo na profil korisnika, čime isključuje probleme "hladnog početka" (eng. *cold-start*) i šupljikavosti matrice (eng. *sparsity*). Primjerice u trenutku dodavanja novog filma u ponudu, može se dogoditi da se preporuči korisniku na temelju žanra i redatelja. Takve preporuke mogu biti iznimno cijenjene od korisnika specifičnih interesa, pogotovo ako tako otkrije i neki manje popularan film koji odgovara njegovim preferencijama. Problem ovog pristupa je pronalaženje i izvlačenje prikladnih atributa jer zahtjeva stručno znanje. Problem je i što ne može stvoriti preporuku za novog korisnika koji nema podataka u svom korisničkom profilu. Takvi sustavi preporučuju samo stavke koje su bazirane na korisničkom profilu, odnosno dolazi do problema kad korisnik ima više različitih interesa. Ne mogu iskoristiti korisne podatke o ocjenama drugih korisnika, za razliku od sustava preporučivanja temeljenih na metodama suradničkog filtriranja.

2.5. Sustavi preporučivanja temeljeni na suradničkom filtriranju

Suradničko filtriranje temelji se isključivo na znanim ocjenama, bez ikakvih metapodataka o korisnicima i stavkama. Primjerice, radeći preporuku za korisnika, sustav traži njemu najsličnije korisnike na temelju zajedničkih ocjena i iz njih računa konačnu predikciju. [2]

Za razliku od sustava temeljenih na sadržaju, suradničko filtriranje može prepoznati skrivene obrasce u ponašanju korisnika i na temelju njih generirati neočekivane, ali relevantne preporuke. Primjerice, ako korisnik visoko ocijeni triler utemeljen na stvarnim događajima, sustav može preporučiti dokumentarni film o stvarnom zločinu koji je poslužio kao inspiracija za taj triler, prepoznajući povezanost kroz slične preferencije drugih korisnika. Više o metodama i realizaciji suradničkog filtriranja u narednim poglavljima.

3. Postupci suradničkog filtriranja i matrična faktorizacija

3.1. Suradničko filtriranje temeljeno na metodama susjedstva

Kao što je već spomenuto u prethodnom poglavlju (poglavlje 2.3.), algoritmi suradničkog filtriranja baziraju se na eksplicitnim ocjenama korisnika stavkama. Postoje dva pristupa osnovnog algoritma suradničkog filtriranja, pristup iz perspektive korisnika i onaj iz perspektive stavki, odnosno korisnik-korisnik (eng. *user-user*) i stavka-stavka (eng. *item-item*). Pristup korisnik-korisnik traži za određenog korisnika njemu najsličnijih N-korisnika, gdje je N broj dobiven algoritmom optimizacije ili nekom heuristikom. Drugi pristup, stavka-stavka, radi identičnu stvar, samo bira N-stavci najsličnijih određenoj stavki. Sličnost se određuje metrikom poput Jaccardove udaljenosti, sličnosti kosinusa ili Pearsonovog koeficijenta korelacije (3.1). Nadalje će biti naveden postupak za varijantu stavka-stavka jer se upravo taj pristup češće koristi u praksi [2]. Naravno varijanta korisnik-korisnik je identična, samo se gledaju ocjene korisnika umjesto stavki. Uzima se vektor svih ocjena koje je određena stavka dobila od strane svih korisnika i računa se sličnost sa svim drugim vektorima ocjena stavci. Jaccardova udaljenost je mjera sličnosti dvije stavke definirana kao omjer presjeka i unije dvaju skupova, gdje su skupovi spomenuti vektori ocjena. Koliko će takva mjera biti visoka ovisi o tome koliko je istih korisnika ocijenilo dvije promatrane stavke. Takva metrika nema dovoljnu ekspresivnost u kontekstu sustava preporučivanja jer uzima u obzir samo prisutnost ili odsutnost zajedničkih elemenata u skupovima, bez uvažavanja intenziteta ili razlika u ocjenama stavci. Ova ograničenost dovodi do gubitka informacija o stvarnoj sličnosti između stavci, osobito u sustavima gdje su ocjene numeričke i variraju po intenzitetu. Sličnost kosinusa je mjera kuta između dva vektora ocjena u višedimenzijskom prostoru. Dobivena vrijednost je

unutar intervala od -1 do 1, gdje "1" označava savršeno poklapanje vektora i potpunu sličnost, "0" označava nepostojanje korelacije, a "-1" potpuno suprotnu orijentaciju vektora i time maksimalnu razliku među tim vektorima. Pearsonov koeficijent (formula 3.1) je mjera linearne korelacije između dva vektora. Temelji se na sličnosti kosinusa, samo što prvo normalizira ocjene tako da uklanja prosječnu ocjenu svakog vektora. S tim se eliminira problem različitih skala ocjenjivanja među korisnicima, odnosno njihove pristranosti. Zbog toga sustavi koji koriste ovu metriku u pravilu bolje generaliziraju. Vrijede ista pravila za izlaz metrike kao i kod sličnosti kosinusa. Generalno se ne može reći da postoji najbolja metrika, najbolja je ona koja daje najbolje rezultate za određeni problem.

$$\text{Pearson}(x, y) = \frac{\sum_{i=1}^{n}(x_i - \overline{x})(y_i - \overline{y})}{\sqrt{\sum_{i=1}^{n}(x_i - \overline{x})^2 \sum_{i=1}^{n}(y_i - \overline{y})^2}} \tag{3.1}$$

Nakon izračuna sličnosti stavke sa svim drugim stavkama odabire se njih N s najvećom mjerom sličnosti. Na temelju njih se računa prosjek, odnosno konačna predikcija. Prosjek se može računati tako da je svaka ocjena ravnopravna, odnosno tako da se sve ocjene zbroje i uprosječe (3.2) ili tako da i veličina sličnosti ulazi u obzir (3.3). Što je stavka sličnija promatranoj, to više doprinosi konačnoj predikciji sa svojom ocjenom. Uobičajena praksa je koristiti formulu za težinski prosjek (3.4) s uračunatom pristranošću korisnika (3.6) i stavci (3.7) [2].

$$r_{xi} = \frac{1}{k} \sum_{y \in N_r} y_i \tag{3.2}$$

$$r_{xi} = \frac{\sum_{y \in N_{sim}} x_y \cdot r_{yi}}{\sum_{y \in N_{sim}} x_y} \tag{3.3}$$

$$r_{xi} = b_{xi} + \frac{\sum_{j \in N} \text{sličnost}_{ij} \cdot (r_{xi} - b_{xj})}{\sum_{j \in N} \text{sličnost}_{ij}} \tag{3.4}$$

$$b_{xi} = \mu + b_x + b_i \tag{3.5}$$

$$b_x = \overline{b_x} - \mu \tag{3.6}$$

$$b_i = \overline{b_i} - \mu \tag{3.7}$$

3.2. Suradničko filtriranje temeljeno na latentnim faktorima

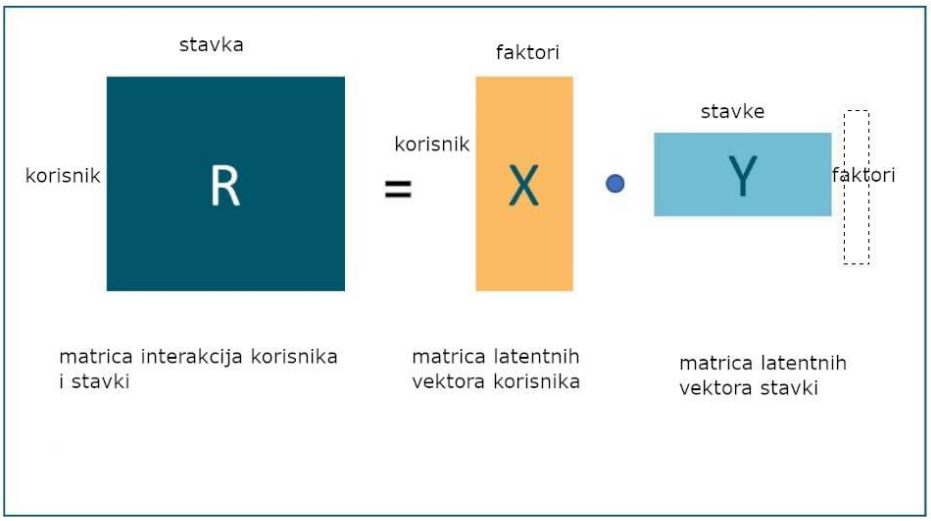
Suradničko filtriranje temelji se isključivo na povijesnom ponašanju korisnika. Dva primarna modela stvaranja preporuka su: modeli bazirani na metodama susjedstva (već spomenuti korisnik-korisnik, odnosno stavka-stavka) i modeli latentnih faktora. Metode susjedstva baziraju se na računanju odnosa između korisnika, odnosno stvari. Identificiranje korisnika sličnih preferencija omogućuje poboljšanje preporuka kroz međusobno dopunjavanje njihovih ocjena. Slično tome, pronalaženje sličnih stavki omogućuje korištenje postojećih ocjena za preciznije predviđanje preferencija korisnika. Ideja modela baziranim na latentnim faktorima je pridjeljivanje faktora svakom korisniku i stavci na temelju uzoraka njihovog ocjenjivanja, odnosno ocjena. Latentni faktori uklanjaju potrebu za ručno generiranim atributima, kao što se radi u sustavima temeljenim na sadržaju. Primjerice, faktor za stavku može biti mjera humorističkog tona u filmu, dok faktor za korisnika može biti njegov interes za navedeni faktor humora u filmu. Matrična faktorizacija temelji se na modeliranju korisnika i stavki u zajedničkom latentnom prostoru. Svaki korisnik i svaka stavka predstavljeni su kao K-dimenzionalni vektori, gdje je K broj latentnih faktora određen nekakvom metodom optimizacije ili heuristikom. Vektor korisnika (oznaka P) opisuje njegove preferencije, dok vektor stavke (oznaka Q) opisuje njezine karakteristike. [5]

$$R_{m\times n} \approx P_{m\times k} Q^T_{n\times k} \tag{3.8}$$

Ocjena korisnika za stavku procjenjuje se s pomoću skalarnog produkta njihovih vektora (3.8). Zadatak je naučiti vektore korisnika i stavki na temelju poznatih ocjena, odnosno da njihov skalarni produkt bude što bliži njihovoj stvarnoj ocjeni. Tu se vidi da je matrična faktorizacija zapravo algoritam strojnog učenja. Cilj modela je minimizacija razlike predviđenih i stvarnih ocjena. Model je zadan s formulom 3.8 , a funkcija gubitka (eng. *loss function*) s formulom 3.9 [5]. Optimizacijski postupak provodi se stohastičkim gradijentnim spustom ili alternirajućim najmanjim kvadratima. Detaljan opis navedenih postupaka je u poglavlju 4.5. . U modelu matrične faktorizacije se također u konačnu predikciju dodaju pristranosti korisnika i stavki radi preciznije predikcije i bolje

generalizacije.

$$L(\mathbf{P}, \mathbf{Q}) = \sum_{(u,i)\in\omega} (R_{u,i} - \mathbf{p}_u^T \mathbf{q}_i)^2 + \lambda(||\mathbf{p}_u||^2 + ||\mathbf{q}_i||^2) \tag{3.9}$$



Slika 3.1. Prikaz matrične faktorizacije [4]

Ako imamo na raspolaganju vremenske oznake, bilo bi ih korisno upotrijebiti da ne bi imali samo statične modele. Problem sa statičnim modelima je što se preferencije korisnika mijenjaju s vremenom, kao i popularnost stavki. Rješenje je da se doda više parametara u model, odnosno da dosadašnji parametri postanu ovisni o vremenu [6]. To je samo jedan od mogućih primjera nadogradnje modela latentnih faktora.

4. Ostvarenje sustava preporučivanja

4.1. Korišteni alati

Za programski jezik ostvarenja zadatka odabran je *Python*. Ponajviše je odabran zbog svoje jednostavnosti i fleksibilnosti. *Python* ima ugrađene strukture podataka, dinamičko tipiziranje i podršku za objektno orijentirano programiranje. Velika zajednica i dostupnost otvorenog koda uvelike pomažu s implementacijom rješenja i njegovom usavršavanju. *Python* ima pristup velikom broju vanjskih knjižnica koje olakšavaju i ubrzavaju implementaciju. Jedna od najpoznatijih, odnosno najkorištenijih knjižnica je *NumPy* koja služi za učinkovitu obradu vektora i matrica, što će se pokazati korisno kod matrične faktorizacije. *NumPy* omogućava vrlo brze linearne operacije, primjerice množenje matrica, faktorizaciju matrica i singularnu dekompoziciju vrijednosti (eng. *Singular Value Decomposition*, SVD). Također *NumPy* efikasnije pohranjuje podatke, memorijski je učinkovitiji od osnovnih struktura podataka u programskog jeziku *Python*. Vrijedi spomenuti i knjižnicu *Surprise* koja nije korištena kao primarna implementacija, već kao referentna točka za validaciju rezultata. Na temelju dobivenih rezultata, implementirano rješenje uspoređeno je s referentnim modelima iz *Surprise* knjižnice, koja je specijalizirana za sustave preporuka. *Surprise* (4.6.) pruža optimizirane implementacije različitih algoritama korištenih u sustavima preporuke, omogućujući evaluaciju preciznosti i performansi razvijenog sustava u odnosu na postojeće standarde. Upravo iz gore navedenih razloga, kombinacija *Python* i *NumPy* imaju široku primjenu u računarstvu i strojnom učenju. Korištene su verzije *Python* 3.9.20 i *NumPy* 1.26.4., te *Surprise* 1.1.4.

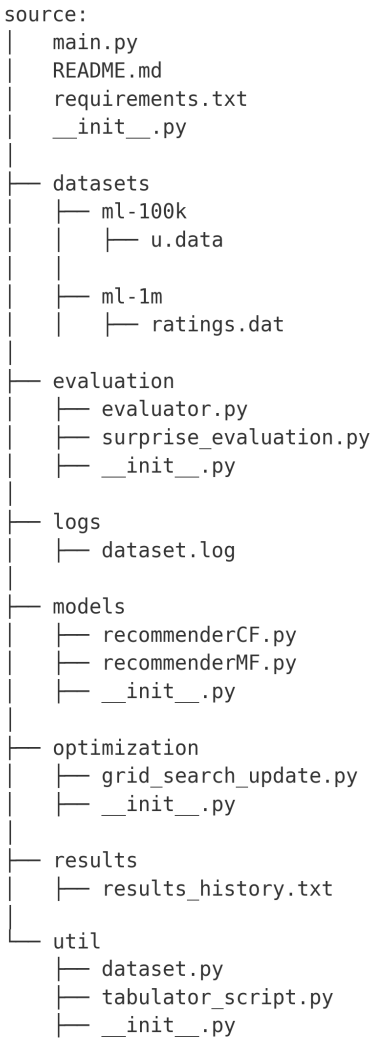
4.2. Korišteni podatci

Skup podataka nad kojim su provedeni spomenuti algoritmi suradničkog filtriranja je *MovieLens*, točnije njegove dvije verzije: *MovieLens 100k* i *MovieLens 1M*. Brojčane oznake uz ime skupa podataka označavaju okviran broj ocjena filmova iz baze *MovieLens*. Programsko rješenje prilagođeno je rukovanju navedenim skupovima podataka i svi su hiperparametri postavljeni sukladno s tim. *MovieLens 100k* sadrži 100.000 ocjena dano od strane 943 korisnika za 1.682 filma, dok *MovieLens 1M* sadrži nešto više od milijun ocjena, 6.040 korisnika i 3.900 filmova. Skupovi također sadrže i vremenske oznake ocjena, što omogućava analizu promjena u preferencijama korisnika tijekom vremena, te sadrže i demografske podatke o korisnicima (starosna dob, spol, zanimanje). Sadržane ocjene imaju raspon "1-5", gdje je "1" oznaka najmanjeg zadovoljstva korisnika, a "5" najvećeg. Prosjek ocjena po korisniku je otprilike 106, a po filmu 60 za *MovieLens 100k*, odnosno 166 po korisniku i 256 po filmu za *MovieLens 1M*. *MovieLens* je jedan od najčešće korištenih skupova podataka u istraživanjima vezanim za sustave preporučivanja, što omogućuje laku usporedbu implementacije i rezultata s postojećim istraživanjima. Popularan je jer sadrži bogatstvo eksplicitnih ocjena korisnika što je pogodno algoritmima suradničkog filtriranja. Omogućuje duboku analizu podataka i preporuka s pomoću metapodataka (spomenute vremenske i demografske oznake) te time omogućava potencijalnu nadogradnju sustava preporučivanja. Valja spomenuti kako i usprkos velikog prosjeka ocjena po korisniku, a i ocjena po filmu, skup podataka *MovieLens* ima problema s rijetkosti podataka (eng. *data sparcity*) kao i drugi skupovi sličnih namjena. Više detalja u radu [7].

4.3. Struktura rješenja

Rješenje je strukturirano kao modularni sustav koji omogućuje jednostavnu nadogradnju i prilagodbu različitim skupovima podataka i algoritmima suradničkog filtriranja. U svrhu organizacije i skalabilnosti, projekt je podijeljen u nekoliko ključnih direktorija, čija svrha je jasna iz njihovog nazivlja (4.1.). Dva spomenuta skupa podataka (poglavlje 4.2.) nalaze se u direktoriju "datasets". Implementacije osnovnih modela suradničkog filtriranja i onih naprednijih koji koriste matričnu faktorizaciju sadržani su u direktoriju "models". Evaluacija performansi modela sustava preporuka provodi se s pomoću

skripti iz direktorija "evaluation". Sve pomoćne funkcije nalaze se u skriptama u direktoriju "util", od kojih većina služi za učitavanje i pretprocesiranje podataka. Glavna skripta "main.py" inicijalizira svaku kombinaciju implementiranih sustava preporuka, njihove varijante i skupa podataka. Potom kroz zadani broj iteracija provodi evaluaciju istih i na izlaz ispisuje tablicu rezultata. Dokumentacija, upute za korištenje i popis potrebnih knjižnica nalaze se u datotekama "README.md" i "requirements.txt". U dodatku na kraju rada nalazi se poglavlje s detaljnim uputama za pokretanje i korištenje sustava (6.). Skripta "dataset.py" sadrži podršku za klasu "MovieLensDataset" čije je uloga učitati podatke iz direktorija "datasets" ili, po potrebi, nekog drugog proizvoljnog direktorija. Po uobičajenom načinu rada ti se podaci učitavaju iz datoteka "u.data" za *MovieLens 100k* i "ratings.dat" za *MovieLens 1M*. Svi korisnici i filmovi (predmeti ocijenjivanja) su predstavljeni rednim brojevima od jedan nadalje, no u svrhu pojednostavljenja, skripta prilagođava identifikatore korisnika i filmova da počinju od nule jer programski jezik *Python* koristi 0-baziranu indeksaciju (eng. *zero-based indexing*). Skripta "grid_search.py" u direktoriju "optimization" služi za rešetkastu pretragu hiperparametara modela kako bi dobili optimalna rješenja koja minimiziraju greške. Skripta "evaluator.py" mjeri točnost modela s pomoću metrika *RMSE* (eng. *Root Mean Square Error* 5.1) i MAE (eng. *Mean Absolute Error* 5.2). Isto tako generira sažetak rezultata evaluacije i sprema podatke u tekstualnu datoteku "results_history.txt" u direktoriju "results". Skripta u svome "main" dijelu ima mogućnost testiranja evaluacijske funkcije nad svakom pojedinom kombinacijom sustava preporuke i skupom podataka. U istom direktoriju "evaluation" nalazi se i skripta "surprise_evaluation.py" koja radi istu stvar kao i navedena "evaluation.py" skripta, samo za sustave preporuke iz knjižnice *Surprise* koji služe isključivo za usporedbu rezultata.



Slika 4.1. Prikaz strukture programskog rješenja

4.4. Osnovni algoritam suradničkog filtriranja

Skripta "recommenderCF.py" u direktoriju "modules" implementira podršku za sustav preporučivanja temeljen na algoritmu suradničkog filtriranja. Algoritam suradničkog filtriranja se bazira na povijesne ocjene korisnika za preporuku novih stvari (filmova), bez potrebe za eksplicitnim atributima filmova ili korisnika. Podržane su dvije osnovne metode suradničkog filtriranja: korisnik-korisnik (eng. *user-user*) i stavka-stavka (eng. *item-item*). Prva metoda se temelji na izračunu sličnosti korisnika, a druga na sličnosti stvari, u ovom slučaju filmova. Prilikom inicijalizacije "recommenderCF" klase sustav pohranjuje podatke o ocjenama filmova, određuje metodu suradničkog filtriranja i

sprema dva hiperparametra, vidljivo u 1. liniji u pseudokodu algoritma (dalje će se podrazumijevati da se linije odnose na pseudokod 1). Na temelju njih postavlja minimalan broj zajedničkih ocjena potrebnih za izračun sličnosti i definira broj najsličnijih objekta koje uzima u obzir prilikom predikcije ocjena. Priprema podataka se odvija na način da se podaci premiješaju kako bi se slučajnim odabirom dobio skup podataka za učenje i skup podataka za validaciju modela (linija 4). Dijele se na temelju parametra r koji označava odnos veličine skupa za učenje i validaciju. Kreira se rječnik ocjena korisnika i filmova radi bržeg dohvaćanja podataka. U svrhu uobičajene prakse kod računanja predikcija rezultata, računa se globalni prosjek ocjena, prosječna ocjena svakog korisnika i prosječna ocjena svakog filma kako bi se dobile pristranosti skupa podataka, korisnika i filmova (linija 5). Sličnost objekata računa se Pearsonovom korelacijom (3.1). Traži se presjek filmova koji su ocijenila dva korisnika (ili u drugoj varijanti presjek korisnika koji su ocijenili isti film). Ako je taj presjek veći od ugovorenog zahtjeva (po osnovi postavljen na dva) radi se normalizacija ocjena oduzimanjem srednje ocjene i računa se sličnost kosinusa. Slijedi zahtjevan proces predračunanja sličnosti između svaka dva objekta i sličnost se sprema u matricu sličnosti (linije 6-14). Ovaj korak je ključan jer ubrzava kasniji postupak predikcije ocjena. Predikcija ocjene temelji se na dohvaćanju najsličnijih k korisnika (linija 16), odnosno filmova u drugoj varijanti. Računa se otežani prosjek ocjena (linija 17), gdje su težine temeljene na sličnosti objekata (najsličniji objekt najviše doprinosi prosječnoj ocjeni). Ako nema dovoljno sličnih objekata, vraća se globalni prosjek ocjena skupa podataka. Evaluacija se vrši nad podskupom podataka za validaciju (linije 15-19). Za svaku ocjenu u tom skupu računa se predikcija ocjene (linija 18) gore navedenim postupkom. Zatim se računa apsolutna razlika stvarne ocjene i predikcije ocjene (linija 20) i iz toga se dobivaju konačne mjere $RMSE$ i MAE.

Algorithm 1 Suradničko filtriranje

Ulaz: skup korisničkih ocjena, parametri *varijanta*, c, k
Izlaz: predviđene ocjene i evaluacija modela
Učitaj skup podataka
Podijeli podatke na skup za učenje i validacijski skup (r)
Izračunaj prosječne ocjene korisnika i filmova, izračunaj globalni prosjek
for svaki objekt A u skupu **do**
    for svaki objekt B u skupu **do**
        Pronađi zajedničke ocjene
        if broj zajedničkih ocjena $\geq c$ **then**
            Izračunaj Pearsonovu korelaciju
            Pohrani sličnost u memoriju
        end if
    end for
end for
for svaki korisnik-film par u validacijskom skupu **do**
    Dohvati k najsličnijih korisnika/filmova
    Izračunaj otežani prosjek ocjena susjeda
    Izračunaj predviđenu ocjenu
end for
Izračunaj RMSE i MAE
Prikaži konačne rezultate evaluacije

4.5. Algoritmi matrične faktorizacije u suradničkom filtriranju

Skripta "recommenderMF.py" u direktoriju "modules" implementira sustav preporučivanja temeljen na suradničkom filtriranju koristeći algoritam matrične faktorizacije. Ova metoda rješava problem predikcije ocjena korisnika dekompozicijom matrice ocjena u dvije manje matrice latentnih faktora, jednu za korisnike (oznaka P) i jednu za filmove (oznaka Q). U usporedbi s osnovnim suradničkim filtriranjem, matrična faktorizacija u pravilu omogućuje bolju generalizaciju jer ekstrakcijom latentnih faktora iz matrice

ocjena može identificirati složenije obrasce i skrivene relacije među korisnicima i stavkama. Za razliku od klasičnih algoritama koji se oslanjaju na eksplicitne sličnosti između korisnika ili stavki, matrična faktorizacija pronalazi apstraktne dimenzije preferencija koje nisu izravno vidljive ili intuitivne, već proizlaze iz statističkih odnosa u podacima. Tako model ne interpretira podatke na temelju unaprijed definiranih kriterija sličnosti, već automatski uči značajke koje najbolje objašnjavaju korisničke ocjene, omogućujući preciznije i prilagodljivije preporuke. Podržane su dvije metode suradničkog filtriranja koristeći algoritam matrične faktorizacije: *SGD* (eng. *Stochastic Gradient Descent*) i *ALS* (eng. *Alternating Least Squares*). Prva metoda se temelji na iterativnom ažuriranju matrica latentnih faktora primjenom gradijentnog spusta, pri čemu se minimizira kvadratna pogreška između stvarnih i predviđenih ocjena. Ova metoda koristi mali broj unaprijed definiranih faktora k i ažurira ih korak po korak na temelju pogreške predikcije. Druga metoda, *ALS*, zasniva se na alternativnoj optimizaciji, gdje se u svakoj iteraciji jedna od dviju matrica (korisničkih ili predmetnih faktora) drži fiksnom, dok se druga ažurira rješavanjem sustava linearnih jednadžbi. Ovaj pristup omogućuje stabilniju konvergenciju i može biti učinkovitiji za veće skupove podataka, no zahtijeva veću računalnu složenost. Algoritam *ALS* koristi dinamičko ažuriranje pristranosti korisnika i stvari. Glavna razlika između ovih metoda leži u načinu optimizacije: *SGD* koristi iterativni pristup zasnovan na prilagođavanju faktora uz pomoć gradijenta, dok *ALS* koristi matricu ocjena i rješava sustav jednadžbi kako bi pronašao optimalne vrijednosti latentnih faktora. Prilikom inicijalizacije objekta klase "RecommenderMF", prima se nekoliko hiperparametara, uključujući broj latentnih faktora k, broj iteracija i, regularizacijski faktor l (lambda - λ) i stopu učenja e (etha - η) (linija 2 u oba pseudoalgoritma: 2 i 3). Priprema podataka se odvija na identičan način kao što je već opisano u prethodnog poglavlju kod modela "recommenderCF". Podaci se premiješaju kako bi se slučajnim odabirom dobio skup podataka za učenje i skup podataka za validaciju modela. Dijele se na temelju parametra r koji označava odnos veličine skupa za učenje i validaciju. Na isti se princip računaju prosjeci i na kraju evaluacija. Kod predikcije se koristi uobičajena formula za uračunavanje pristranosti ocjenjivanja korisnika i filmova (inicijalizacija u liniji 6), a završni izlaz funkcije predviđanja se reže kako bi bio u normalnom intervalu "1-5". Pseudokod algoritma *SGD* (opisan ispod 2) latentni faktori se inicijaliziraju nasumičnim vrijednostima i onda skaliraju malim brojem kako ne bi došlo do eksplozije gradijenta i divergencije (linija 5).

Od linije 8 do linije 21 zadan je proces učenja algoritma. Za svaki redak ulaznog skupa podataka algoritam računa predikciju skalarnim umnoškom vektora latentnih faktora (linija 11). Zatim u liniji 12 računa grešku zadanu preko funkcije pogreške. U linijama 13 i 14 primjenjuje se korak gradijentnog spusta u kojem parametri dobivaju pomak u smjeru suprotnom od smjera gradijenta funkcije pogreške. Ažuriraju se na temelju parcijalnih derivacija funkcije pogreške i pomiču se za zadanu stopu učenja. U liniji 15 je vidljivo kako sustav pamti ukupni kvadratni gubitak, koji će u liniji 17 uprosječiti. Linije 18-20 služe za prijevremeni prekid algoritma ako je već došlo do konvergencije (nema potrebe trošiti vrijeme tražiti neznatno bolje rješenje). Na kraju (linija 23) algoritam isporučuje optimalne vrijednosti P i Q koji se koriste u predikciji ocjena neviđenih podataka. Pseudokod algoritma *ALS* (opisan ispod 3) latentni faktori se inicijaliziraju nasumičnim vrijednostima i onda skaliraju malim brojem kako ne bi došlo do eksplozije gradijenta i divergencije (linija 5). Od linije 8 do linije 25 zadan je proces učenja algoritma. *ALS* fiksira jednu od matrica latentnih faktora dok optimizira drugu, konstantno zamjenjujući pozicije matrica pri svakoj iteraciji. Prvo se ažuriraju korisnički faktori P koristeći ocjene koje su korisnici dali stavkama (linije 9-14). Računa se podskup značajki vezanih uz ocijenjene stavke te se rješava sustav linearnih jednadžbi kako bi se ažurirali faktori svakog korisnika. Nakon toga, isti postupak se provodi za faktore stavki Q, ovaj put koristeći informacije o korisnicima koji su ocijenili određenu stavku (linije 17-21). Linije 15 i 23 služe za ažuriranje pristranosti. Na kraju (linija 27) algoritam isporučuje optimalne vrijednosti P i Q koji se koriste u predikciji ocjena neviđenih podataka i, naravno, za validaciju modela.

Algorithm 2 Stohastički gradijentni spust (SGD) za matričnu faktorizaciju

Ulaz: skup korisničkih ocjena, parametri k, i, l, e

Izlaz: optimizirani latentni faktori P i Q

Inicijaliziraj latentne faktore P i Q s malim slučajnim vrijednostima

Inicijaliziraj pristranosti korisnika i stvari b_u i b_i

for svaki epoch $t = 1$ to i **do**

    $loss \leftarrow 0$

    for svaki korisnik, stavka, ocjena (u, i, r) u skupu za učenje **do**

        $pred =$ predikcija(u, i) ▷ Računanje predviđene ocjene

        $error \leftarrow r - pred$

        Ažuriraj $P_u \leftarrow P_u + e \cdot (error \cdot Q_i - l \cdot P_u)$

        Ažuriraj $Q_i \leftarrow Q_i + e \cdot (error \cdot P_u - l \cdot Q_i)$

        $loss \leftarrow loss + error^2$

    end for

    $loss \leftarrow loss /$ broj uzoraka

    if $|prethodni\_loss - loss| < 10^{-8}$ **then**

        break ▷ Ako je promjena u gubitku zanemariva, prekini učenje

    end if

end for

Izlaz: P, Q s optimiziranim latentnim faktorima

Algorithm 3 Alternirajući najmanji kvadrati (ALS) za matričnu faktorizaciju

Ulaz: skup korisničkih ocjena, parametri k, i, l
Izlaz: optimizirani latentni faktori P i Q

Inicijaliziraj latentne faktore P i Q s malim slučajnim vrijednostima
Inicijaliziraj pristranosti korisnika i stvari b_u i b_i

for svaki epoch $t = 1$ to i **do**
    for svaki korisnik u **do**
        $R_u \leftarrow$ ocjene korisnika u
        $non\_zero\_indices \leftarrow$ indeksi koji imaju ocjenu
        $Q\_tilde \leftarrow Q[non\_zero\_indices]$
        $R\_tilde \leftarrow R_u[non\_zero\_indices] -$ prosjek_ocjena
        Ažuriraj $P_u \leftarrow$ rješenje$(Q_tilde^T Q_tilde + l \cdot I_k, Q_tilde^T R_tilde)$
        Ažuriraj $b_u[u] \leftarrow$ nova vrijednost pristranosti korisnika
    end for
    for svaki stvar j **do**
        $R_j \leftarrow$ ocjene za stavku j
        $non\_zero\_indices \leftarrow$ indeksi koji imaju ocjenu
        $P\_tilde \leftarrow P[non\_zero\_indices]$
        $R\_tilde \leftarrow R_j[non\_zero\_indices] -$ prosjek_ocjena
        Ažuriraj $Q_j \leftarrow$ rješenje$(P_tilde^T P_tilde + l \cdot I_k, P_tilde^T R_tilde)$
        Ažuriraj $b_i[i] \leftarrow$ nova vrijednost pristranosti stvari
    end for
end for

Izlaz: P, Q s optimiziranim latentnim faktorima

4.6. Knjižnica *Surprise*

Surprise je *Python* knjižnica koja omogućava jednostavnu implementaciju i evaluaciju različitih algoritama korištenih u sustavima preporučivanja [8]. U svrhu rada, korišteni su algoritmi temeljeni na suradničkom filtriranju kako bi dobili referentne rezultate za usporedbu s prethodno implementiranim modelima koji koriste samo osnovne *Python* knjižnice i *NumPy*. Knjižnica podržava različite algoritme preporuke, od kojih su u interesu algoritmi *kNN* (eng. *k-Nearest Neighbours*) i *SVD* (eng. *Singular Value Decomposition*). Knjižnica sadrži i funkcije za izračun preciznosti modela koristeći metrike poput *RMSE* (eng. *Root Mean Squared Error*) i *MAE* (eng. *Mean Square Error*), koje su korištene pri procjeni preciznosti prijašnjih modela u radu. Princip implementacije je sličan prijašnjim modelima. *Surprise* nudi opciju ugrađenih skupova podataka među kojima su i do sad korišteni *Movie Lens* skupovi. Postavljaju se opcije za željene sustave preporučivanja koje se predaju željenoj klasi (*kNN* ili *SVD*) prilikom inicijalizacije. Zatim se željeni podatkovni skup dijeli na dva dijela, skup za učenje i skup za validaciju. Preko klase "accuracy" se računaju potrebne metrike i dobivaju traženi rezultati. *Surprise* knjižnica korištena je u skripti "surprise_evaluation.py" u direktoriju "evaluation".

5. Vrednovanje i rezultati

5.1. Korištene mjere

Mjere korištene za validaciju rješenja su spomenute *RMSE* (5.1) i *MAE* (5.2) u poglavlju 4.3. Obje formule se baziraju na udaljenosti predviđenih ocjena od stvarnih. *MAE* je prosjek svih razlika u ocjenama i tretira greške jednako. *RMSE*, s druge strane, više kažnjava veće greške (kvadrira razliku u ocjenama). *RMSE* je u većini slučajeva glavna metrika sustava preporuke, dok je *MAE* korisna druga opcija. Velik *RMSE* može značiti da model radi velike greške. Velik *MAE*, a mali *RMSE* može značiti da model radi puno malih grešaka, ali ne i velike. Idealan slučaj je kada su obje mjere što bliže nuli, jer to znači da model daje najtočnije preporuke.

$$\mathrm{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(r_{ui} - \hat{r}_{ui})^2} \tag{5.1}$$

$$\mathrm{MAE} = \frac{1}{n}\sum_{i=1}^{n}|r_{ui} - \hat{r}_{ui}| \tag{5.2}$$

5.2. Vrednovanje modela

Obje varijante modela osnovnog algoritma suradničkog filtriranja vrednovane su u 20 navrata. Modeli matrične faktorizacije, odnosno latentnih faktora vrednovane su u preko 100 navrata. Skup podataka podijeljen je na dva dijela: skup za učenje i skup za validaciju (tablica 5.1.). Sustav pamti uklonjene ocjene i proizvodi njihovu predikciju na temelju drugih, još poznatih ocjena.

Tablica 5.1. Primjer uklonjenih ocjena u MovieLens skupu podataka u svrhu validacije

User	Item	Rating
1	101	4
2	102	5
12	103	4
4	104	2
51	105	5
1	106	—
7	101	3
82	108	1
9	101	—
12	102	3

5.3. Usporedba rezultata

Rezultati programskog rješenje i rezultati knjižnice *Surprise* su u okviru očekivanih za skup *MovieLens*. Metode susjedstva imaju veću grešku od metodi matrične faktorizacije kod obje implementacije. Nažalost, knjižnica *Surprise* nema implementiran model matrične faktorizacije bazirane na alternirajućim najmanjim kvadratima tako da se taj model neće koristiti u usporedbi rezultata. Rezultati osnovnih modela suradničkog filtriranja razlikuju se unutar 10% u korist knjižnice *Surprise*, no valja primijetiti kako se vremena izvođenja bitno razlikuju. Obje implementacije računaju matricu sličnosti prije same primjene metoda sličnosti, ali knjižnica *Surprise* koristi dodatne metode izračuna te matrice kako bi ubrzala postupak (koristi dodatak *Cython*). Rezultati modela matrične faktorizacije bazirane na stohastičkom gradijentnom spustu su gotovo identični u obje implementacije.

Tablica 5.2. Rezultati implementiranih sustava dobiveni iz skripte "main.py"

vrsta sustava	verzija	skup	avg RMSE	avg MAE	vrijeme(s)
RecommenderCF	item-item	100k	0.9671	0.7577	299.506 s
RecommenderCF	item-item	1m	0.9338	0.7330	2709.537 s
RecommenderCF	user-user	100k	1.1416	0.9078	207.589 s
RecommenderCF	user-user	1m	1.1330	0.8974	4653.646 s
RecommenderMF	sgd	100k	0.9300	0.7241	87.633 s
RecommenderMF	sgd	1m	0.8727	0.6752	916.937 s
RecommenderMF	als	100k	1.1163	0.8374	14.185 s
RecommenderMF	als	1m	0.8986	0.6904	134.302 s

Tablica 5.3. Rezultati sustava knjižnice "Surprise" dobiveni iz skripte "surprise_evaluation.py"

vrsta sustava	verzija	skup	avg RMSE	avg MAE	vrijeme(s)
RecommenderCF	item-item	100k	1.0299	0.8045	4.45
RecommenderCF	user-user	100k	1.0238	0.8034	3.87
RecommenderMF	sgd	100k	0.9393	0.7409	1.33
RecommenderCF	item-item	1m	0.9678	0.7519	87.69
RecommenderCF	user-user	1m	1.0088	0.7954	223.94
RecommenderMF	sgd	1m	0.8711	0.6841	13.19

5.4. Potencijalne nadogradnje rješenja

Jedna od potencijalnih nadogradnji je korištenje vremenske dimenzije, spomenuto u poglavlju 3.2. Četvrti stupac skupa podataka *MovieLens* predstavlja vremensku oznaku (eng. *timestamp*), što nam omogućuje pretvaranje svih varijabli u funkcije ovisne o vremenu. Drugi pristup bi se mogao bazirati na ne davanje iste važnosti svakoj ocjeni u skupu podataka. Kratkoročna pristranost uzrokovana oglašavanjem ili neiskreni korisnici mogu negativno utjecati na kvalitetu preporuka. Ovaj problem posebno dolazi do izražaja kod implicitnih podataka, gdje razina pouzdanosti može ovisiti o numeričkim vrijednostima poput trajanja ili učestalosti korisničkih radnji. Kako bi se poboljšala točnost modela, matrična faktorizacija može uključiti razine povjerenja pri obradi podataka. Za više detalja pogledaj rad [9]. Suradničko filtriranje temeljeno na neuronskim mrežama uvodi težinske faktore u produkt latentnih vektora korisnika i stavci. U radu [10] iz 2017. godine primijećeno je kako produkt latentnih faktora može biti ograničavajući. Generalizirana matrična faktorizacija uvodi težinske faktore u unutarnji produkt i primjenjuje nelinearnu aktivacijsku funkciju kako bih dobila predikcije ocjena. Primjerice, osnovni model matrične faktorizacije odgovara generaliziranoj matričnoj faktorizaciji s uniformnim težinskim faktorima i funkcijom identiteta za aktivacijsku funkciju. Sličan pristup koristi višeslojni perceptron (eng. *Multy Layer Perceptron - MLP*) skupa s generaliziranom matričnom faktorizacijom. Takav model baziran na dubokim neuronskim mrežama omogućuje dublje učenje korisničkih preferencija. Mogu se koristi i autokoderi (eng. *autoencoder*) za učenje niskodimenzionalne reprezentacije korisničkih ocjena (pogledaj rad [11]). Autokoderi predstavljaju nadogradnju matrične faktorizacije jer omogućuju nelinearno učenje latentnih korisničkih preferencija, za razliku od klasične matrične faktorizacije koja se oslanja isključivo na linearne kombinacije faktora. Autokoderi efikasno obrađuju rijetke podatke, uče se na komprimiranoj reprezentaciji korisnič-

kim ocjena, a dekodiranjem daju gusti izlazni vektor s preporučenim vrijednostima za sve stavke. Sve ove spomenute tehnike i još njih opisane su u prezentaciji [5].

6. Zaključak

Sustavi preporučivanja su bitna stavka digitalnih platformi zaduženi za filtraciju i približavanje sadržaja krajnjem korisniku. Takvi sustavi koriste metode poput suradničkog filtriranja kako bi na temelju povijesnog ponašanja korisnika zaključili njegove iduće korake, naravno u cilju poboljšavanja zadovoljstva i iskustva korisnika. Algoritmi matrične faktorizacije poboljšavaju osnovne metode suradničkog filtriranja smanjenjem dimenzionalnosti podataka i otkrivajući skrivene uzorke ponašanja korisnika u latentnom prostoru. Rješenje sadrži sustav preporučivanja temeljen na korisnik-korisnik suradničkom filtriranju i sustav temeljen na stavka-stavka verziji suradničkog filtriranja. Rezultati potvrđuju bolje performanse pristupa stavka-stavka u praksi. Oba su algoritma korištena nad skupom od stotinu tisuća ocjena i nad skupom od milijun ocjena. Razlike u vremenu izvođenja nisu skalabilne zbog izračuna matrice sličnosti koja se radi po principu "svak sa svakim", umjesto da se koristi neka optimalnija metoda. U implementaciji rješenja korištena su i dva algoritma matrične faktorizacije: *SVD* i *ALS*. Na skupu *MovieLens* daju otprilike jednake rezultate, makar je *ALS* nešto brži i precizniji, odnosno robusniji je na većem skupu podataka. Postoje razni algoritmi nadogradnji na spomenute koji dodatno poboljšavaju točnost preporuka. Algoritmi matrične faktorizacije i dalje ostaju posljednje dostignuće (eng. *state of the art*) u svijetu preporuka zato što detektiraju obrasce ponašanja korisnika nevidljive ljudskom oku, imaju mogućnost skaliranja s porastom broja korisnika i stavki te nadogradnja za određene specifične probleme nije komplicirana.

Literatura

[1] J. Leskovec, A. Rajaraman, i J. D. Ullman, *Recommendation Systems*. Cambridge: Cambridge University Press, 2014., str. 292–324.

[2] M. Šilić, "Recommender systems", Lecture slides, University of Zagreb, 2024., accessed: Feb. 8, 2025. Available at: http://www.fer.hr/predmet/avsp.

[3] F. Casalegno, "Recommender systems: A complete guide to machine learning models", 2022., accessed: 2025-02-08. [Mrežno]. Adresa: https://medium.com/towards-data-science/recommender-systems-a-complete-guide-to-machine-learning-models-96d3f94ea748

[4] S. Yoo i J. Kim, "Merchant recommender system using credit card payment data", *Electronics*, sv. 12, br. 4, 2023. https://doi.org/10.3390/electronics12040811

[5] A. S. Kurdija, "Matrix factorization in recommender systems", Lecture slides, University of Zagreb, 2024., accessed: Feb. 8, 2025. Available at: http://www.fer.hr/predmet/avsp.

[6] Y. Koren, "Collaborative filtering with temporal dynamics", *Commun. ACM*, sv. 53, br. 4, str. 89––97, travanj 2010. https://doi.org/10.1145/1721654.1721677

[7] F. M. Harper i J. A. Konstan, "The movielens datasets: History and context", *ACM Transactions on Interactive Intelligent Systems (TiiS)*, sv. 5, br. 4, str. 19:1–19:19, 2015.

[8] N. Hug, "Surprise: A python library for recommender systems", *Journal of Open Source Software*, sv. 5, br. 52, str. 2174, 2020. https://doi.org/10.21105/joss.02174

[9] Y. Hu, Y. Koren, i C. Volinsky, “Collaborative filtering for implicit feedback datasets”, u *2008 Eighth IEEE International Conference on Data Mining*, 2008., str. 263–272. https://doi.org/10.1109/ICDM.2008.22

[10] X. He, L. Liao, H. Zhang, L. Nie, X. Hu, i T.-S. Chua, “Neural collaborative filtering”, u *Proceedings of the 26th International Conference on World Wide Web*, ser. WWW ’17. Republic and Canton of Geneva, CHE: International World Wide Web Conferences Steering Committee, 2017., str. 173––182. https://doi.org/10.1145/3038912.3052569

[11] O. Kuchaiev i B. Ginsburg, “Training deep autoencoders for collaborative filtering”, *ArXiv*, sv. abs/1708.01715, 2017. [Mrežno]. Adresa: https://api.semanticscholar.org/CorpusID:12570825

Popis tablica

Popis slika

Popis pseudo algoritama

Dodatak A: Upute za uporabu programskog rješenja

Struktura programskog rješenja zadana je slikom 4.1. Svaka skripta ima svoj glavni dio ("main") u kojemu se mogu postaviti parametri i inicijalizirati klasa te skripte u svrhu lokalnog testiranja. Dosta informacija o upotrebi se nalazi u "README.md" datoteci. Prije pokretanja sustava potrebno je instalirati Python (preporučena verzija 3.9.20 ili novija). Python se može preuzeti sa službene stranice (https://www.python.org/downloads) i instalirati prema uputama za vaš operacijski sustav. Provjera uspješne instalacije vrši se pokretanjem naredbe iz terminala "python –version" za operacijski sustav Windows, odnosno "python3 –version" za macOS i Linux. Potrebni paketi za podršku rješenja nalaze se u tekstualnoj datoteci "requirements.txt". Preporuka je koristiti virtualno okruženje naredbom instalacije "python -m venv venv", potom prikladnom naredbom za aktivaciju (u ovisnosti o sustavu). Najjednostavniji način instalacije potrebnih paketa je preko naredbe "pip install -r requirements.txt". Do rezultata se dolazi pokretanjem naredbe "python main.py" iz terminala u direktoriju gdje se nalazi skripta. Skripta u terminalu prikazuje proces izvođenja i izračuna različitih modela i na kraju tablicu rezultata. Detaljniji rezultati se spremaju u datoteku "results_history.txt". U glavnom dijelu skripte "main.py" postoji mogućnost odabira modela (uključeni su svi u listi "recommenders", no daju se lako isključiti jednostavnim procesom komentiranja linije koristeći prefiks "%"). Poštimavanje hiperparametara i testiranje istih vrši se u specifičnim skriptama za svaki tip modela.

Sažetak

Ostvarenje sustava preporučivanja koristeći algoritam matrične faktorizacije

Fran Vrankić

Ovaj rad proučava sustave preporučivanja, od njihovih varijanti i metoda do primjene i izazova s kojima se suočavaju. Objašnjeno je filtriranje sadržaja preko implicitnih povratnih informacija korisnika i suradničko filtriranje s pomoću eksplicitnih ocjena korisnika. Poseban je naglasak stavljen na algoritme matrične faktorizacije kao napredniju metodu koja daje kvalitetnije preporuke s većom pouzdanošću. U praktičnom dijelu rada implementiran je sustav preporučivanja koristeći osnovne metode suradničkog filtriranja i sustav koji koristi matričnu faktorizaciju. Sustavi su testirani nad skupom podataka *MovieLens*, a rezultati uspoređeni s referentnom *Python* knjižnicom *Surprise*. Rezultati su zadovoljavajući i pokazuju superiornost matrične faktorizacije nad običnim metodama filtracije preporuka. Rad potvrđuje važnost matrične faktorizacije u suvremenim sustavima preporučivanja te ukazuje na moguća poboljšanja (naprednije metode matrične faktorizacije).

Ključne riječi: Sustav preporučivanja; suradničko filtriranje; matrična faktorizacija; MovieLens; Surprise

Abstract

Recommendation Systems Implementation Using Matrix Factorization Algorithm

Fran Vrankić

This paper examines recommender systems, covering their variations, methods, applications, and the challenges they face. Content filtering based on users' implicit feedback and collaborative filtering using explicit user ratings are explained. Special emphasis is placed on matrix factorization algorithms as a more advanced method that provides higher-quality recommendations with greater reliability.

In the practical part of the paper, a recommender system was implemented using basic collaborative filtering methods, along with a system utilizing matrix factorization. The systems were tested on the *MovieLens* dataset, and the results were compared with the reference *Python* library *Surprise*. The results are satisfactory and demonstrate the superiority of matrix factorization over standard recommendation filtering methods. The paper confirms the importance of matrix factorization in modern recommender systems and highlights potential improvements (more advanced matrix factorization methods).

Keywords: Recommender System; Collaborative filtering; Matrix Factorization; MovieLens; Surprise