Univerzitet u Nišu
Elektronski fakultet
Katedra za računarstvo

Sistem za preporuku filmova zasnovan na kolaborativnom filtriranju

Mentor:

doc. dr Miloš Bogdanović

Student:

Draginja Anđelković, 1028

Niš, jun 2021.

Sadržaj

1.Uvod

Kolaborativno filtriranje (engl. *collaborative filtering*) je najčešća tehnika koja se koristi kada se razvijaju inteligentni sistemi za preporuku (engl. *intelligent recommendation system*) koji mogu naučiti da daju bolje preporuke kako se prikuplja više informacija o korisnicima.

Većina web sajtova, kao što su *Amazon*, *YouTube* i *Netflix* koriste kolaborativno filtriranje kao deo svojih sofisticiranih sistema preporuka. Ovu tehniku možete koristiti za pravljenje sistema za preporuku koji daju predloge korisniku na osnovu sviđanja (engl. *likes*) i nesviđanja (engl. *dislikes*) sličnih korisnika. [1]

Prvo poglavlje ovog rada predstavlja uvod u obrađenu temu.

U drugom poglavlju biće reči kolaborativnom filtriranju kao tehnici koja se najčešće koristi prilikom razvoja sistema za preporuke. Objašnjene su dve najznačajnije metode kolaborativnog filtriranja, metod najbližeg susedstva i faktorizacija matrice. Takođe, izloženo je kako i kada je poželjno koristi tehniku kolaborativnog filtriranja.

Treće poglavlje predstavlja opis praktičnog dela ovog rada, sistema za preporuku filmova zasnovanom na kolaboratibnom filtriranju, koji je razvijen u programskom jeziku Pajton (engl. *Python*). Opisan je skup podataka koji je korišćen prilikom razvoja sistema za preporuku filmova i način na koji su podaci iz skupa podataka obrađeni. U okviru ovog poglavlja prikazan je jednostavan sistem za preporuku filmova zasnovan na popularnosti, kao i jedan nešto složeniji sistem za preporuku filmova zasnovan na kolaborativnom filtriranju. Na kraju poglavlja izvršena je evaluacija razvijenog sistema.

Poslednje poglavlje, poglavlje četiri, sadrži zaključak, kao i smernice za eventualno dalje unapređivanje modela razvijenog u sklopu praktičnog dela ovog rada.

2.Kolaborativno filtriranje

Kao i mnoge tehnike mašinskog učenja, sistem za preporučivanje vrši predviđanje na osnovu istorijskog ponašanja korisnika. Konkretno, to je predviđanje korisničkih preferenci za skup stavki na osnovu prethodnog iskustva. Da bi se napravio sistem za preporuke, najpopularnija su dva pristupa, filtriranje zasnovano na sadržaju i kolaborativno filtriranje.

Pristup zasnovan na sadržaju (engl. *content-based*) zahteva veliku količinu informacija o karakteristikama predmeta, umesto da koristi interakcije korisnika i povratne informacije. Na primer, to mogu biti atributi filma poput žanra, godine publikacije, režisera, glumace itd, ili pak tekstualni sadržaj članaka koji se mogu analizirati pomoću mehanizama za obradu prirodnog jezika.

S druge strane, **kolaborativnom filtriranju** nije potrebno ništa drugo osim preferencija korisnika na skup stavki kroz vreme. Budući da se zasniva na istorijskim podacima, ovde je suštinska pretpostavka da će se korisnici koji su se složili u prošlosti takođe slagati i u budućnosti. [2]

Kolaborativno filtriraje radi tako što pretražuje veliku grupu ljudi i pronalazi manji broj korisnika sa ukusom sličnim određenom korisniku. Razmatra predmete koji se sličnim korisnicima sviđaju i kombinuje ih da bi stvorio rangiranu listu predloga.

Postoji mnogo načina da se odluči koji su korisnici slični i da se kombinuju njihovi izbori da bi se kreirala lista preporuka. [1]

U pogledu korisničkih preferencija, reakcije korisnika na stavke se obično dele u dve kategorije. **Eksplicitna ocena** je ocena koju korisnik dodeli stavci na kliznoj skali, poput 5 zvezdica na određenu stavku. Ovo je najdirektnija povratna informacija korisnika koja pokazuje koliko se neki predmet korisniku sviđa. **Implicitna ocena** podrazumeva indirektne preference korisnika, kao što su prikazi stranice, klikovi, zapisi o kupovini, da li su korisnici slušali muzičku numeru ili ne, itd. [2]

U ovom radu će detaljnije biti opisano kolaborativno filtriranje koje je tradicionalno i moćno sredstvo za razvoj sistema za preporuku.

Dve najznačajnije metode kolaborativnog filtriranja su najbliže susedstvo i faktorizacija matrice. U praktičnom delu ovog rada korišćeno je kolaborativno filtriranje zasnovano na metodi najbližeg susedstva.

2.1.Najbliže susedstvo

Standardni metod kolaborativnog filtriranja poznat je kao algoritam najbližeg sustedstva (engl. *Nearest Neighbor Algorithm*). Postoji kolaborativno filtriranje zasnovano na korisniku i kolaborativno filtriranje zasnovano na stavkama.

Razmotrimo prvo kolaborativno filtriranje zasnovano na korisniku. Neka je data matrica ocena dimenzije n x m, sa korisnicima u_i, i = 1, ...n i stavkama p_j, j=1, …m. Želimo da predvidimo ocenu r_{ij} ako ciljni korisnik i nije pogledao ili ocenio stavku j. Proces je izračunavanje sličnosti između ciljnog korisnika i svih ostalih korisnika, odabir X najsličnijih korisnika i uzimanje težinskog proseka ocena tih X korisnika, gde sličnost korisnika određuje faktor težine. Izračunavanje ocene ciljnog korisnika i na određenu stavku j, r_{ij}, vrši se po formuli:

$$r_{ij} = \frac{\sum_k Sličnost\ (u_i\ ,\ u_k) r_{kj}}{broj\ ocena}$$

Proces kolaborativnog filtriranja zasnovanog na korisniku prikazano je na slici 1. [4]

Slika 1: Kolaborativno filtriranje zasnovano na korisniku

Iako različiti ljudi mogu imati različite polazne vrednosti prilikom davanja ocena, neki ljudi uglavnom daju visoke ocene, neki su prilično strogi iako su zadovoljni stavkama. Da bismo izbegli ovu pristrasnost, možemo da oduzmemo prosečnu ocenu svih stavki svakog korisnika prilikom izračunavanje težinskog proseka i da je vratimo za ciljanog korisnika, kao što je prikazano u formuli ispod.

$$r_{ij} = \bar{r}_i + \frac{\sum_k Sličnost\ (u_i,u_k)(r_{kj} - \bar{r}_k)}{broj\ ocena}$$

Dva načina za izračunavanje sličnosti su **Pearsonova korelacija** (engl. *Pearson Correlation*) i **kosinusna sličnost** (engl. *Cosine Similarity*). Formule za izračunavanje ovih sličnosti su prikazane ispod.

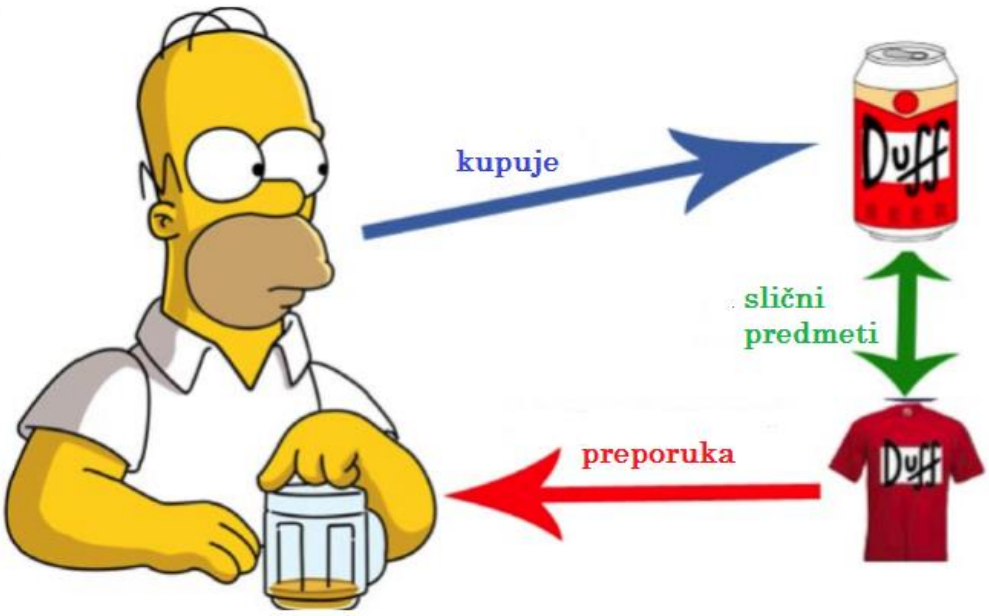
$$Pearsonova\ korelacija : Sličnost\ (u_i, u_k) = \frac{\sum_j (r_{ij} - r_i)(r_{kj} - r_k)}{\sqrt{\sum_j (r_{ij} - r_i)^2 \sum_j (r_{kj} - r_k)^2}}$$

$$Kosinusna\ sličnost : Sličnost\ (u_i, u_k) = \frac{r_i * r_k}{|r_i||r_k|} = \frac{\sum_{j=1}^{m} r_{ij} r_{kj}}{\sqrt{\sum_{j=1}^{m} {r_{ij}}^2 \sum_{j=1}^{m} {r_{kj}}^2}}$$

U osnovi, ideja je pronaći korisnike koji su slični ciljnom korisniku (najbliže susede) i dodeliti težinu njihovim ocenama stavke, i na osnovu toga predvideti ocenu te stavke za ciljanog korisnika.

I kada ne posedujemo nikakvo znanje o stavkama i samim korisnicima, smatramo da su dva korisnika slična kada daju istoj stavci slične ocene. Analogno tome, za kolaborativno filtriranje zasnovano na stavkama kažemo da su dve stavke slične kada su od istog korisnika dobile slične ocene. Zatim ćemo napraviti predviđanje za ciljanog korisnika za stavku izračunavanjem težinskog proseka ocena za X najsličnijih stavki ovog korisnika. Jedna od ključnih prednosti kolaborativnog filtriranja zasnovanog na stavkama je stabilnost koja se sastoji u tome što se ocene date stavke vremenom neće značajno promeniti, za razliku od ukusa ljudi.

Proces kolaborativnog filtriranja zasnovanovanog na stavkama prikazan je na slici 2. [5]



Slika 2: Kolaborativno filtriranje zasnovano na stavkama

Postoji nekoliko ograničenja ove metode. Ova metoda ne funkcioniše dobro kada su matrice ocena retko posednute, tj. kada niko u susedstvu nije ocenio stavku koju sistem pokušava da preporuči ciljnom korisniku. Takođe, nije računarski efikasan kada broj stavki i korisnika raste. [2]

2.2.Faktorizacija matrice

Budući da su oskudnost i skalabilnost dva najveća izazova za standardnu metodu kolaborativnog filtriranja, dolazi naprednija metoda koja razlaže originalnu retko posednutu matricu na nisko-dimenzionalne matrice sa latentnim faktorima/karakteristikama i sa manje propusnosti. To je faktorizacija (dekompozicija) matrice.

Pored rešavanja problema oskudnosti i skalabilnosti, postoji i intuitivno objašnjenje zašto su nam potrebne nisko-dimenzionalne matrice koje predstavljaju preference korisnika. Na primer, korisnik je dao dobre ocene filmu *Avatar*, *Gravitacija* i *Početak*. Nisu to nužno 3 odvojena mišljenja, ali pokazuju da bi ovi korisnici mogli biti zainteresovani za naučnofantastične filmove i možda postoji mnogo više naučnofantastičnih filmova koje bi ovaj korisnik želeo da pogleda. Za razliku od određenih filmova, latentne karakteristike izražene su atributima višeg nivoa, a *Sci-Fi*

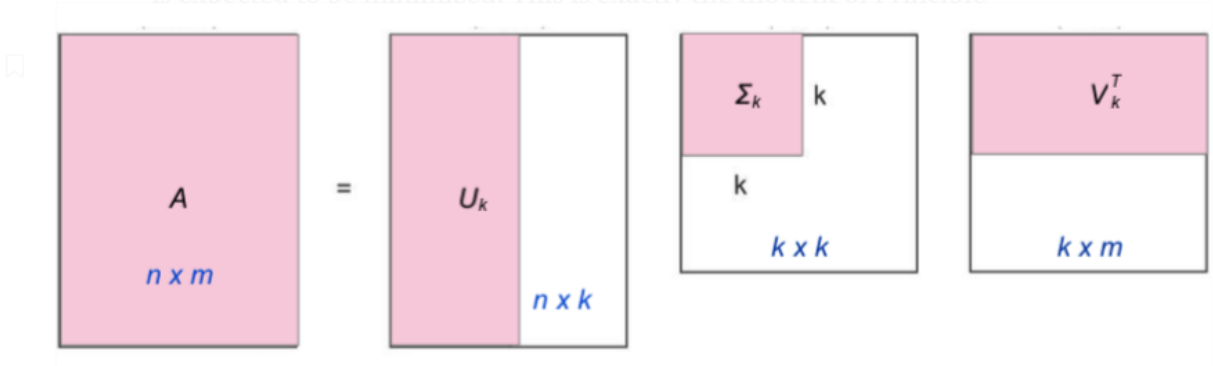
kategorija je jedna od latentnih karakteristika u ovom sličaju. Ono što nam faktorizacija matrice na kraju daje je koliko je korisnik svrstan u skup latentnih karakteristika i koliko se film uklapa u ovaj skup latentnih karakteristika. Prednost toga u odnosu na standardni algoritam najbližih suseda je u tome što iako dva korisnika nisu ocenila nijedan isti film, ipak je moguće pronaći sličnost između njih ako dele slične osnovne ukuse, tj. latentne karateristike.

Da bi se shvatilo kako se matrica faktoriše (dekomponuje), prvo što treba razumeti je **dekompozicija singularne vrednosti** (engl. *Singular Value Decomposition* - SVD). Na osnovu linearne algebre, bilo koja stvarna matrica R se može razložiti na 3 matrice U, Σ i V. Nastavljajući primer sa fimovima od malopre, U je n x r matrica karakteristika koja je latentna za korisnika, V je matrica latentnih karakteristika filmova dimenzija m x r. Σ je r x r dijagonalna matrica koja sadrži pojedinačne vrednosti izvorne matrice, jednostavno predstavljajući koliko je određena karakteristika važna za predviđanje korisničkih preferencija.

$$R = U\Sigma V^T$$

$$U \in IR^{nxr}, \qquad \Sigma \in IR^{rxr}, \qquad V \in IR^{rxm}$$

Da bismo sortirali vrednosti Σ smanjenjem apsolutne vrednosti i odsekli matricu Σ na prvih k dimenzija (k singularnih vrednosti), možemo rekontruisati matricu kao matricu A. Izbor dimenzije k treba da bude takav da A sigurno može da uhvati najveći deo varijanse unutar originalne matrice R, tako da je A aproksimacija R, A ≈ R. Razlika između A i R je greška za koju se očekuje da bude minimalizovana. Faktorizacija matrice A prikazana je na slici 3.



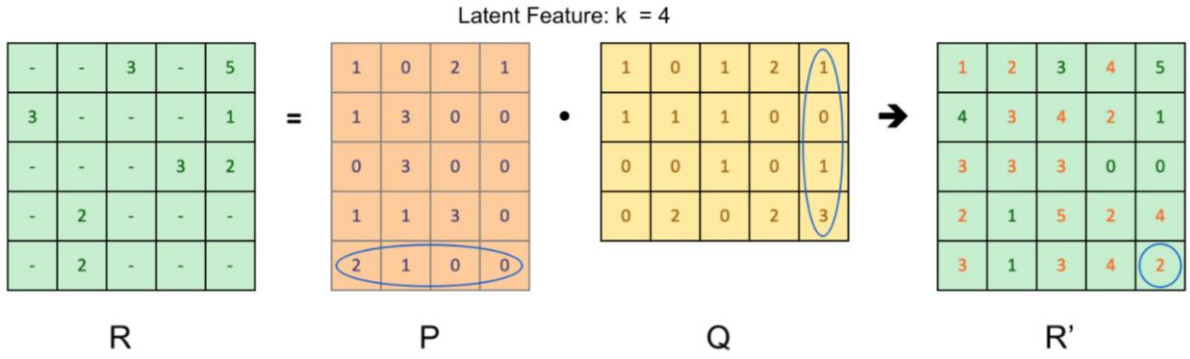
Slika 3: Faktorizacija matrice A

Kada je matrica R gusta, U i V se mogu lako analitički faktorisati. Međutim, matrica gledanosti filmova je izuzetno oskudna. Iako postoje neke metode za popunjavanje nedostajućih vrednosti, ovde se ne razmatra slučaj kada su nedostajuće vrednosti dopunjene. Umesto da faktorišemo R preko SVD-a, pokušavamo da pronađemo direktno U i V sa ciljem da kada se U i V ponovo pomnože, izlazna matrica R' bude najbliža aproksimacija R i da nema više proređene matrice.

Ova numerička aproksimacija obično se postiže **ne-negativnom faktorizacijom matrice** (engl. *Non-Negative Matrix Factorization*) za sisteme za preporuku, jer nema negativnih vrednosti u ocenama.

Ako je stavka označena kao vektor q_i, a korisnik kao vektor p_u, predviđena ocena koju bi korisnik u dao stavci i računa se kao skalarni proizvod ova dva vektora, kao što je prikazano na formuli ispod. Ova vrednost je predstavljena u matrici R' u redu u i koloni i. Proces kreiranja izlazne matrice R' prikazan je na slici 4.

$$Predviđena\ ocena: r'_{ui} = {p_u}^T q_i$$



Slika 4: Matrice gledanosti filmova

Kako pronaći optimalne vrednosti za q_i i p_u? Kao i većina zadataka mašinskog učenja, funkcija gubitka (engl. *loss function*) je definisana da minimizira troškove grešaka.

$$\min_{q,p} \sum (r_{ui} - {p_u}^T q_i)^2 + \lambda(\|p_u\|^2 + \|q_i\|^2)$$

r_{ui} je prava ocena iz originalne matrice korisnik-stavka. Proces optimizacije je pronalaženje optimalne matrice P sastavljene od vektora p_u i matrice Q sastavljene od vektora q_i kako bi se minimizovala greška kvadratnog zbira između predviđenih ocena r_{ui}' i stvarnih ocena r_{ui}. Takođe, dodata je L2 regularizacija da bi se sprečilo prekomerno overfitovanje korisnika i predmeta. Takođe, sasvim je uobičajeno dodavanje izraza pristrasnosti koji obično ima 3 glavne komponente: prosečna ocena svih stavki μ, prosečna ocena stavke i minus (-) μ (zabeležena kao b_u), prosečna ocena koju daje korisnik u minus (-) u (zabeležena kao b_i). [2]

$$\min_{p,q,b_u,b_i} \sum (r_{ui} - ({p_u}^T q_i + \mu + b_u + b_i))^2 + \lambda\left(\|p_u\|^2 + \|q_i\|^2 + {b_u}^2 + {b_i}^2\right)$$

2.3.Korišćenje kolaborativnog filtriranja

Kolaborativno filtriranje koristi interakcije koje korisnici imaju sa stavkama. Evo nekoliko tačaka koje treba imati na umu prilikom odlučivanja da li se u određenim situacijama može koristiti kolaborativno filtriranje:

- Kolaborativno filtriranje ne zahteva da budu poznate karakteristike predmeta ili korisnika. Pogodan je za skup različitih vrsta predmeta, na primer, proizvodi iz supermarketa gde se mogu dodati predmeti različitih kategorija. U skupu sličnih predmeta, poput onog u knjižari, poznate karakteristike poput pisaca i žanrova mogu biti korisne i sistemi mogu imati koristi od pristupa zasnovanog na sadržaju ili od hibridnih pristupa.
- Kolaborativno filtriranje može pomoći sistemima za preporuku da se ne specijalizuju preterano za korisničke profile i preporučuju stavke koje se potpuno razlikuju od onoga što su ranije videli. Ukoliko se očekuje da sistem za preporuku ne predlaže par patika nekome ko je upravo kupio jedan par sličnih patika, dodavanje kolaborativnog filtriranja u sistem za preporuku je dobro rešenje.

Iako se kolaborativno filtriranje vrlo često koristi kod sistema za preporuku, neki od izazova sa kojima se suočavaju tokom korišćenja su sledeći:

- Kolaborativno filtriranje može dovesti do nekih problema poput hladnog starta za nove stavke koje se dodaju u listu. Dok ih neko ne oceni, stavke se ne dobijaju kao preporuke.
- Proređenost podataka (engl. *data sparsity*) može uticati na kvalitet preporuka i takođe izaziva gore pomenuti problem hladnog starta.
- Skaliranje može biti izazov za rastuće skupove podataka jer složenost može postati prevelika. Sistemi za preporuku zasnovani na stavkama brži su od korisničkih kada je skup podataka veliki.

Pošto svaki od tipova algoritama koji se koriste kod sistema za preporuku ima svoje prednosti i nedostatke, obično je najbolje praviti hibridne sisteme za preporuku. Prednosti više algoritama koji rade zajedno mogu vam pomoći da postavite tačnije sisteme za preporuku. Na primer, rešenje dobitnika *Netflix* nagrade je bila složena kombinacija više algoritama.

3.Primer primene kolaborativnog filtriranja za izgradnju sistema za preporuku filmova

Praktični deo ovog rada je sistem za preporuku filmova zasnovan na kolaborativnom filtriranju korišćenjem metode najbližeg susedstva. Sistem ne uzima u obzir karakteristike filmova, tj. ne podrazumeva nikakvu analizu sadržaja.

Sistem za preporuku filmova je razvijen u programskom jeziku Pajton (engl. *Python*).

Za razvoj sistema za preporuku filmova kao praktičnog dela ovog rada, izabran je baš programski jezik Pajton jer je zbog brojnih i bogatih biblioteka, ovaj jezik našao široku primenu u projektima iz oblasti veštačke inteligencije i mašinskog učenja.

3.1.Skup podataka

Pre nego što napravimo bilo koji model mašinskog učenja, od vitalnog je značaja da razumemo šta su podaci i šta pokušavamo da postignemo. Istraživanje podataka otkriva skrivene trendove i uvide, a preprocesiranje podataka čini podatke spremnim za upotrebu u algoritmima mašinskog učenja. [3]

Da biste eksperimentisali sa algoritmima za preporuku, biće vam potrebni podaci koji sadrže skup stavki i skup korisnika koji su reagovali na neke od stavki.

Reakcija može biti eksplicitna (ocena na skali od 1 do 5, sviđanja ili nesviđanja) ili implicitna (pregled predmeta, dodavanje na listu želja, vreme provedeno na članku).

Dok radite sa takvim podacima, uglavnom ćete ih videti u obliku matrice koja se sastoji od reakcija koje skup korisnika daje na neke stavke iz skupa stavki. Svaki red bi sadržao ocene koje je dao korisnik, a svaka kolona ocene koje je dobila neka stavka. Matrica sa pet korisnika i pet predmeta mogla bi izgledati kao na slici 5.

	i_1	i_2	i_3	i_4	i_5
u_1	5		4	1	
u_2		3		3	
u_3		2	4	4	1
u_4	4	4	5		
u_5	2	4		5	2

Slika 5: Matrica ocena korisnika i stavki

Matrica prikazuje pet korisnika koji su neke od predmeta ocenili na skali od 1 do 5. Na primer, prvi korisnik je dao ocenu 4 trećoj stavci.

U većini slučajeva ćelije u matrici su prazne, jer korisnici ocenjuju samo nekoliko stavki. Teško je da će svaki korisnik oceniti ili reagovati na svaku dostupnu stavku. Matrica sa uglavnom praznim ćelijama naziva se retko posednuta matrica, a suprotna od nje (uglavnom popunjena matrica) naziva se gusto posednuta matrica.

Za praktični deo ovog rada korišćen je skup podataka *MovieLens* koji je prikupio *GroupLens Reserach*. Konkretno, MovieLens 100k skup podataka je stabilan referentni skup podataka sa 100836 ocena koje je 610 korisnika dalo za 9742 filma, pri čemu je svaki korisnik ocenio najmanje 20 filmova.

Ovaj skup podataka se sastoji od nekoliko datoteka koje sadrže informacije o filmovima, korisnicima i ocenama koje su korisnici dali filmovima koje su gledali. Najbitnije su sledeće datoteke:

- *movies*: lista filmova
- *ratings*: lista ocena koje su dali korisnici

Datoteka *movies* sadrži ID-jeve filmova, nazive i godine publikacije filmova kao i žanrove filmova odvojene zarezima. Prvih nekoliko redova ove datotke izgleda kao na slici 6.

	MovieID	TitleAndYearOfPublication	Genres
0	1	Toy Story (1995)	Adventure|Animation|Children|Comedy|Fantasy
1	2	Jumanji (1995)	Adventure|Children|Fantasy
2	3	Grumpier Old Men (1995)	Comedy|Romance
3	4	Waiting to Exhale (1995)	Comedy|Drama|Romance
4	5	Father of the Bride Part II (1995)	Comedy

Slika 6: Prvih nekoliko redova iz datoteke sa filmovima

Datoteka *ratings* koja sadrži ocene je lista korisničkih ID-jeva, ID-jeva filmova, ocene i vremenske oznake odvojene zarezima. Prvih nekoliko redova ove datoteke izgleda kaona slici 7.

	UserID	MovieID	Rating	Timestamp
0	1	1	4.0	964982703.0
1	1	3	4.0	964981247.0
2	1	6	4.0	964982224.0
3	1	47	5.0	964983815.0
4	1	50	5.0	964982931.0

Slika 7: Prvih nekoliko redova iz datoteke sa ocenama

Kao što je prikazano gore, datoteka govori koju ocenu je korisnik dodelio određenom filmu. Ova datoteka sadrži više od 100000 takvih ocena, koje će se koristiti za predviđanje ocena filmova koje korisnici nisu videli. [1]

3.2.Preprocesiranje podataka

Prvo učitavamo skup podataka, to jest datoteke *movies*, *ratings*, *tags* i *links*, kao što je prikazano na slici 8 za primer učitavanja movies datoteka.

```
[4] #Loading data
    movies = pd.read_csv('movies.csv', sep=',', error_bad_lines=False, encoding="latin-1")
    movies.columns = ['MovieID', 'TitleAndYearOfPublication', 'Genres']
    movies.head()
```

	MovieID	TitleAndYearOfPublication	Genres
0	1	Toy Story (1995)	Adventure|Animation|Children|Comedy|Fantasy
1	2	Jumanji (1995)	Adventure|Children|Fantasy
2	3	Grumpier Old Men (1995)	Comedy|Romance
3	4	Waiting to Exhale (1995)	Comedy|Drama|Romance
4	5	Father of the Bride Part II (1995)	Comedy

Slika 8: Učitavanje podataka iz movies datoteke

Nakon učitavanja datoteka možemo videti da skup podataka sadrži 100836 ocena za 9742 filma, što je prikazano na slici 9.

```
[6] # Print the number of records and the total number of movies
    print('Dataset sadrzi ', len(ratings), ' ocena za ', len(movies), ' filmova.')

    Dataset sadrzi  100836  ocena za  9742  filmova.
```

Slika 9: Karakteristike skupa podataka

Svaki od ovih fajlova je, jedan po jedan, istražen i podaci su pripremljeni za primenu algoritama masinskog učenja.

3.2.1.Datoteka *movies*

Započinjemo sa datotekom sa filmovima. Inicijalno, datoteka *movies* izgleda kaona slici 6. Možemo videti da su naziv filma i godina publikacije zapamćeni unutar iste kolone, kao i da neki filmovi imaju više žanrova. Kolonu sa nazivom i godinom publikacije razbijamo na dve kolone, prvu koja pamti samo naziv filma (*Title*) i drugu koja pamti godinu publikacije (*YearOfPublication*). Takođe, svaki žanr smeštamo u posebnu kolonu, ukoliko film pripada većem broju žanrova.

Daljom analizom utvrđeno je da 12 redova iz tabele imaju null vrednosti za godinu publikacije. S obzirom da znamo naslove filmova za koje godina publikacije nije navedena u skupu podataka, pretraživanjem sajta IMDb, pronalazimo nedostajuće vrednosti za godine publikacije I dopunjujemo nedostajuće vrednosti. Na sličan način je utvrđeno da za neke filmove iz skupa podataka nisu izlistani žanrovi. Na isti način na koji su nađene nedostajuće godine publikacije, sa *IMDb* sajta su pronađeni žanrovi za filmove za koje nisu bili navedeni, i nedostajući žanrovi su dopunjeni.

U sklopu preprocesiranja datoteke *movies* proveravamo tipove podataka za svaku od kolona i prilagođavamo širinu kolone tako da prikaže puni tekst kolona.

Nakon modifikacija koje su izvršene nad podacima iz datoteke sa filmovima, tabela filmovi izgleda kao na slici 10.

```
[33] movies.head()
```

	MovieID	Title	YearOfPublication	Genre 0	Genre 1	Genre 2	Genre 3	Genre 4	Genre 5	Genre 6	Genre 7	Genre 8	Genre 9
0	1	Toy Story	1995	Adventure	Animation	Children	Comedy	Fantasy	NaN	NaN	NaN	NaN	NaN
1	2	Jumanji	1995	Adventure	Children	Fantasy	NaN	NaN	NaN	NaN	NaN	NaN	NaN
2	3	Grumpier Old Men	1995	Comedy	Romance	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
3	4	Waiting to Exhale	1995	Comedy	Drama	Romance	NaN	NaN	NaN	NaN	NaN	NaN	NaN
4	5	Father of the Bride Part II	1995	Comedy	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN

Slika 10: Tabela sa filmovima nakon preprocesiranja

3.2.2.Datoteka *ratings*

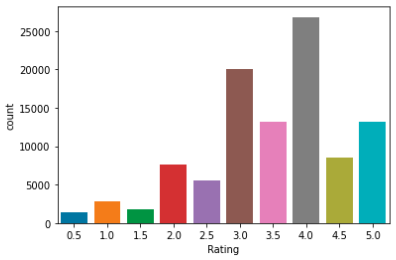
Proveravamo oblik podataka i prvih nekoliko redova u skupu ocena otkriva da će naša matrica ocena korisnika biti vrlo proređena, jer je svaki korisnik ocenio mnogo manji broj filmova u odnosu na broj filmova u skupu podataka. Inicijalno, datoteka *ratings* izgleda kaona slici 7.

Analizom skupa podataka sa ocenama utvrđeno je da su eksplicitne ocene predstavljene sa 0.5, 1.0, 1.5, 2.0, 2.5, 3.0, 3.5, 4.0, 4.5 i 5,0, a ukoliko korisnik nije ocenio određeni film kolona sa ocenom ima vrednost *nan*.

Za izgradnju sistema za preporuku filmova koji je razvijen kao praktični deo ovog rada koriste se samo eksplicitne ocene, dok se nan vrednosti, implicitne ocene, predstavljaju vrednošću 0, čime su korisnici razdvojeni na one koji su određeni film ocenili eksplicitno (tj. dali neku ocenu filmu) i one čije je ponašanje implicino zabeleženo.

Brojni prikaz (engl. countplot) *rating* atributa sa slike 11 pokazuje da su veće ocene češće među korisnicima, a ocena 4 je data najveći broj puta.

```
[39] #plotting count of movieRating
     sns.countplot(data=ratings , x='Rating')
     plt.show()
     #It can be seen that higher ratings are more common among users and rating 4 has been rated highest number of times
```



Slika 11: Countplot rating atributa

3.3.Jednostavan sistem za preporuke zasnovan na popularnosti

U ovom trenutku može se izgraditi jednostavan sistem preporuka zasnovan na popularnosti, kao što je prikazano na slici 12. Ovaj sistem se bazira na sumiranju korisničkih ocena za različite filmove.

```
[40] #At this point , a simple popularity based recommendation system can be built based on count of user ratings for different movies
     ratings_count = pd.DataFrame(ratings.groupby(['MovieID'])['Rating'].sum())
     top10 = ratings_count.sort_values('Rating', ascending = False).head(10)
     print("Following movies are recommended")
     top10.merge(movies, left_index = True, right_on = 'MovieID')
     #Given below are top 10 recommendations based on popularity.
```

```
Following movies are recommended
```

	Rating	MovieID	Title	YearOfPublication	Genre 0	Genre 1	Genre 2	Genre 3	Genre 4	Genre 5	Genre 6	Genre 7	Genre 8	Genre 9
277	1404.0	318	Shawshank Redemption, The	1994	Crime	Drama	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
314	1370.0	356	Forrest Gump	1994	Comedy	Drama	Romance	War	NaN	NaN	NaN	NaN	NaN	NaN
257	1288.5	296	Pulp Fiction	1994	Comedy	Crime	Drama	Thriller	NaN	NaN	NaN	NaN	NaN	NaN
1939	1165.5	2571	Matrix, The	1999	Action	Sci-Fi	Thriller	NaN	NaN	NaN	NaN	NaN	NaN	NaN
510	1161.0	593	Silence of the Lambs, The	1991	Crime	Horror	Thriller	NaN	NaN	NaN	NaN	NaN	NaN	NaN
224	1062.0	260	Star Wars: Episode IV - A New Hope	1977	Action	Adventure	Sci-Fi	NaN	NaN	NaN	NaN	NaN	NaN	NaN
97	955.5	110	Braveheart	1995	Action	Drama	War	NaN	NaN	NaN	NaN	NaN	NaN	NaN
2226	931.5	2959	Fight Club	1999	Action	Crime	Drama	Thriller	NaN	NaN	NaN	NaN	NaN	NaN
461	929.5	527	Schindler's List	1993	Drama	War	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN
418	892.5	480	Jurassic Park	1993	Action	Adventure	Sci-Fi	Thriller	NaN	NaN	NaN	NaN	NaN	NaN

Slika 12: Jednostavan sistem za preporuku filmova zasnovan na popularnosti

Očigledno je da su filmovi sa liste filmova koje je preporučio ovaj sistem za preporuke zasnovan na popularnosti, filmovi koji imaju visoke ocene na *IMDb* sajtu. Na primer, najpopularniji film koji ovaj sistem prepoznaje je Bekstvo iz Šošenka (engl. *The Shawshank Redemption*), a ovaj film je takođe najbolje ocenjen na IMDb listi. Drugi po popularnosti je film Forest Gump, koji je na

IMDb-ovoj listi popularnosti 12. Film Petparačke priče (engl. *Pulp Fiction*) je treći po popularnosti u našem skupu podataka, dok je na IMDb-ovoj listi popularnosti 7... I ostali filmovi koje je ovaj sistem preporučio su visoko rangirani na IMDb-ovoj listi popularnosti.

3.4.Sistem preporuka zasnovan na kolaborativnom filtriranju

Prvi ključni korak u izgradnji sistema za preporuku zasnovanih na kolaborativnom filtriranju je generisanje matrice ocena korisnika iz tabele ocena, kao što je prikazano na slici 13.

```
[43] #Generating ratings matrix from ratings table
     ratings_matrix = ratings.pivot(index='UserID', columns='MovieID', values='Rating')
     userID = ratings_matrix.index
     movieID = ratings_matrix.columns
     print(ratings_matrix.shape)
     ratings_matrix.head()
     #Notice that most of the values are NaN (undefined) implying absence of ratings
```

(554, 8757)

MovieID	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	24	25	26	
UserID																											
1	4.0	NaN	4.0	NaN	NaN	4.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	N
2	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	N
3	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	N
4	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	3.0	NaN	NaN	NaN	NaN	NaN	N
5	4.0	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	NaN	4.0	NaN	NaN	NaN	NaN	NaN	N

5 rows × 8757 columns

Slika 13: Generisanje matrice ocena korisnika

Primećuje se da su većina vrednosti u matrici ocena NaN, što ukazuje na odsustvo ocena i, prema tome, na retkost podataka. Takođe imajte na umu da su ovde uzete u obzir samo eksplicitne ocene. Kako većina algoritama mašinskog učenja ne može da obrađuje NaN vrednosti, zamenjujemo ih sa 0, što sada ukazuje na odsustvo ocene.

3.4.1.Kolaborativno filtriranje zasnovano na korisnicima

U nastavku su objašnjene funkcije koje su korišćene u sistemu za preporuku filmova zasnovanom na kolaborativom filtriranju, koji je razvijen u okviru praktičnog dela ovog rada.

Fukncija *findksimilarusers* koja je prikazana na slici 14 prima *userID* i matricu ocena kao parametre. Ova funkcija kreira model najbližih suseda (promenljiva *model_knn*). Model se trenira prosleđivanjem matrice ocena funkciji *fit*. Povratna vrednost funkcije su sličnost između korisnika i indeksi k najsličnijih korisnika.

```
[47] #This function finds k similar users given the user_id and ratings matrix
     #These similarities are same as obtained via using pairwise_distances
     def findksimilarusers(user_id, ratings, metric = metric, k=k):
         similarities=[]
         indices=[]
         model_knn = NearestNeighbors(metric = metric, algorithm = 'brute')
         model_knn.fit(ratings)
         loc = ratings.index.get_loc(user_id)
         distances, indices = model_knn.kneighbors(ratings.iloc[loc, :].values.reshape(1, -1), n_neighbors = k+1)
         similarities = 1-distances.flatten()

         return similarities,indices
```

Slika 14: Izgled funkcije za pronalaženje k sličnih korisnika

Funkcija *predict_userbased* koja je prikazana na slici 15 predviđa ocenu za navedenu kombinaciju korisnika i filma na osnovu korisničkog pristupa. Id korisnika i filma se prosleđuju funkciji kao argumenti poziva. Pozivom funkcije *findksimilrusers*, ova fukncija nalazi k korisnika koji su najsličniji korisniku čiji je id prosleđen. Nakon pronalaženja k sličnih korisnika, funkcija računa težinski prosek ocena koje su slični korisnici dali filmu čiji je id prosleđen. Izračunati težinski prosek predstavlja ocenu koju sistem predviđa kao ocenu koju bi određeni korisnik dao određenom filmu. Predviđena ocena je povratna vrednost funkcije.

```
#This function predicts rating for specified user-item combination based on user-based approach
def predict_userbased(user_id, item_id, ratings, metric = metric, k=k, print_flag = True):
    prediction=0
    user_loc = ratings.index.get_loc(user_id)
    item_loc = ratings.columns.get_loc(item_id)
    similarities, indices=findksimilarusers(user_id, ratings,metric, k) #similar users based on cosine similarity
    mean_rating = ratings.iloc[user_loc,:].mean() #to adjust for zero based indexing
    sum_wt = np.sum(similarities)-1
    product=1
    wtd_sum = 0

    for i in range(0, len(indices.flatten())):
        if indices.flatten()[i] == user_loc:
            continue;
        else:
            ratings_diff = ratings.iloc[indices.flatten()[i],item_loc]-np.mean(ratings.iloc[indices.flatten()[i],:])
            product = ratings_diff * (similarities[i])
            wtd_sum = wtd_sum + product

    #in case of very sparse datasets, using correlation metric for collaborative based approach may give negative ratings
    #which are handled here as below
    if prediction <= 0:
        prediction = 1
    elif prediction >5:
        prediction = 5

    prediction = round(mean_rating + (wtd_sum/sum_wt), 1)
    prediction = round(prediction * 2) / 2

    #prediction = int(round(mean_rating + (wtd_sum/sum_wt)))
    if (print_flag == True):
        print ('\nPredicted rating for user {0} -> item {1}: {2}'.format(user_id,item_id,prediction))

    return prediction
```

```
[49] predict_userbased(260,3000,ratings_matrix);

     Predicted rating for user 260 -> item 3000: 1.5
```

Slika 15: Izgled funckije za predviđanje ocene

Funkcija *recommendItem* koja je prikazana na slici 16 koristi gore navedene funkcije za preporučivanje filmova za pristup zasnovan na korisniku ili stavci (zasnovan na odabranoj kombinaciji pristupa i metrike). Daju se preporuke za filmove koji nisu već ocenjeni. Funkcija kao argument poziva prima id korisnika kome treba preporučiti filmove, matricu ocena i metriku koja će se koristiti za računanje sličnosti između korisnika. Korisnik može da iz padajućeg menija odabere metriku sličnosti (kosinus/korelacija), kao i pristup (korisnik/stavka) dok poziva ovu funkciju. Funkcija *recommendItem* poziva funkcije *predict_userbased* ili

predict_itembased u zavisnosti od odabranog pristupa, sa odgovarajućim parametrima, i predviđenu ocenu koju ove funkcije vrate dodaje u listu predikcija. Nakon što izračuna predviđene ocene koje bi korisik čiji je id prosleđen funkciji kao argument dao svim filmovima iz skupa podataka, sortira se lista predviđenih ocena i 10 filmova sa najboljim ocenama se ispisuju na ekranu kao preporuke filmova korisniku.

```
#This function utilizes above functions to recommend items for item/user based approach and cosine/correlation.
#Recommendations are made if the predicted rating for an item is >= to 6,and the items have not been rated already
def recommendItem(user_id, ratings, metric=metric):
    if (user_id not in ratings.index.values) or type(user_id) is not int:
        print("User id should be a valid integer from this list :\n\n {} ".format(re.sub('[\[\]]', '', np.array_str(ratings_matrix.index.values))))
    else:
        ids = ['Select','Item-based (correlation)','Item-based (cosine)','User-based (correlation)','User-based (cosine)']
        select = widgets.Dropdown(options=ids, value=ids[0],description='Select approach', width='1000px')
        def on_change(change):
            clear_output(wait=True)
            prediction = []
            if change['type'] == 'change' and change['name'] == 'value':
                if (select.value == 'Item-based (correlation)') | (select.value == 'User-based (correlation)') :
                    metric = 'correlation'
                else:
                    metric = 'cosine'
                #with suppress_stdout():
                if (select.value == 'Item-based (correlation)') | (select.value == 'Item-based (cosine)'):
                    for i in range(ratings.shape[1]):
                        if (ratings[ratings.columns[i]][user_id] != 0): #not rated already
                            prediction.append(predict_itembased(user_id, ratings.columns[i] ,ratings, metric))
                        else:
                            prediction.append(-1) #for already rated items
                else:
                    for i in range(ratings.shape[1]):
                        if (ratings[ratings.columns[i]][user_id] != 0): #not rated already
                            prediction.append(predict_userbased(user_id, ratings.columns[i] ,ratings, metric))
                        else:
                            prediction.append(-1) #for already rated items
                prediction = pd.Series(prediction)
                prediction_sorted = prediction.sort_values(ascending=False)
                recommended = prediction_sorted[:10]

                print("As per {0} approach....Following movies are recommended...".format(select.value))
                for i in range(len(recommended)):
                    print("{0}. {1}".format(i + 1, movies.Title[recommended.index[i]].encode('utf-8')))

        select.observe(on_change)
        display(select)
```

Slika 16: Izgled funkcije za preporuku filmova

Na slici 17 prikazano je 10 najboljih preporuka filmova za korisnika 220 zasnovane na korisničkom pristupu.

```
[71] #checking for incorrect entries
     predicted_ratings = recommendItem(220,ratings_matrix)

     As per Item-based (correlation) approach....Following movies are recommended...
     1. b'Star Wars: Episode IV - A New Hope '
     2. b'Princess Bride, The '
     3. b'Arsenic and Old Lace '
     4. b'Some Kind of Wonderful '
     5. b'Toy Story '
     6. b'Fall '
     7. b'Mission: Impossible '
     8. b"Cheech and Chong's Up in Smoke "
     9. b'Once Upon a Time in the West '
     10. b'All of Me '
```

Slika 17: Preporuke filmova za korisnika 220 zasnovane na korisničkom pristupu

3.4.2.Kolaborativno filtriranje zasnovano na stavkama

Slične funkcije su napisane za kolaborativno filtriranje zasnovano na stavkama za pronalaženje k sličnih filmova i predviđanje korisničkih ocena za sve filmove. Ista funkcija *recommendItem* se

može koristiti za preporučivanje knjiga na osnovu pristupa zasnovanog na stavkama i odabrane metrike. Daju se preporuke ako filmovi nisu već ocenjeni.

Slika 18 prikazuje 10 najboljih preporuka filmova za korisnika 220 zasnovane na pristupu baziranom na stavkama. Vidimo da se preporuke zasnovane na stavkama u izvesnoj meri razlikuju od onih koje je preporučio sistem korišćenjem pristupa zasnovanom na korisnicima.

```
[75] #checking for incorrect entries
     predicted_ratings = recommendItem(220,ratings_matrix)

     As per User-based (correlation) approach....Following movies are recommended...
     1. b'Toy Story '
     2. b'Godfather, The '
     3. b'Star Wars: Episode IV - A New Hope '
     4. b'Shawshank Redemption, The '
     5. b'Home Alone 2: Lost in New York '
     6. b"Cheech and Chong's Up in Smoke "
     7. b'Cooler, The '
     8. b'Once Upon a Time in the West '
     9. b'Rififi '
     10. b'Star Wars: Episode V - The Empire Strikes Back '
```

Slika 18: Preporuke filmova za korisnika 220 zasnovane napristupu baziranom na stavkama

3.5.Evaluacija sistema za preporuku filmova zasnovanog na kolaborativnom filtriranju

Sistemi za preporuku mogu se evaluirati na mnogo različitih, često neuporedivih načina. U praktičnom delu ovog rada za evaluaciju razvijenog modela korišćena je *r2_score* funkcija iz *sklearn.metrics* biblioteke.

Koeficijent determinacije, označen kao R^2, "R na kvadrat", (engl. *R-sqared*) predstavlja procenat varijanse u zavisnoj promenljivoj koja se predviđa iz nezavisnih promenljivih.[6]

Vrednosti ovog koeficijenta kreću se od 0 do 1 i obično se navode kao procenti, od 0% do 100%. Što se vrednost više približava 1, to je sistem precizniji. Ako je R^2 jednak jedinici, to znači da su sve predikcije tačne. [7]

Model se trenira nad celim skupom podataka, a testira nad 100 random određenih korisnika. R^2 skor je izračunat za sve kombinacije pristup – metrika (pristup zasnovan na stavkama/korelacija, prisut zasnovan na korisniku/korelacija, pristup zasnovan na stavkama/kosinusna sličnost, pristup zasnovan na korisniku/kosinusna sličnost), za svakog korisnika iz test skupa podataka. Srednje vrednosti R^2 skora za sve kombinacije pristupa I metrike prikazane su na slici 19.

```
[69] print("R2 score for Item-based (correlation) approach: ", "%.2f" % (r2_correlation_item_score*100), "%")
     print("R2 score for User-based (correlation) approach: ", "%.2f" % (r2_correlation_user_score*100), "%")
     print("R2 score for Item-based (cosine) approach: ", "%.2f" % (r2_cosine_item_score*100), "%")
     print("R2 score for User-based (cosine) approach: ", "%.2f" % (r2_cosine_user_score*100), "%")

     R2 score for Item-based (correlation) approach:  80.43 %
     R2 score for User-based (correlation) approach:  60.58 %
     R2 score for Item-based (cosine) approach:  80.43 %
     R2 score for User-based (cosine) approach:  62.07 %
```

Slika 19: Rezultati evaluacije sistema za preporuku filmova

Vrednosti prikazane na slici 19 predstavljaju rezultate uspešnosti izražene u procentima sistema za preporuku filmova razvijenog u okviru praktičnog dela ovog rada. Kao što je već pomenuto, što je vrednost R^2 skora bliža 1 (odnosno 100%), sistem uspešnije predviđa ocene korisnika na određeni film, i samim tim sistem daje bolje preporuke. Na osnovu dobijenog rezultata možemo zaključiti da razvijeni sistem za preporuku filmova radi značajno bolje ukoliko se koristi pristup zasnovan na stavkama. Metrika ne utiče značajno na uspešnost razvijenog sistema, tj. slični rezultati su dobijeni i kad se koristi korelacija i kad se koristi kosinusnu sličnost.

4.Zaključak

Kolaborativno filtriranje pruža snažnu moć predviđanja za sisteme za preporuku i istovremeno zahteva najmanje podataka. To je tehnika koja se najčešće koristi za razvoj sistema za preporuku.

Većina web sajtova, kao što su Amazon, YouTube i Netflix koriste kolaborativno filtriranje kao deo svojih sofisticiranih sistema preporuka.

Dve najznačajnije metode kolaborativnog filtriranja su najbliže susedstvo i faktorizacija matrice.U praktičnom delu ovog rada korišćeno je kolaborativno filtriranje zasnovano na metodi najbližeg susedstva.

Postoji kolaborativno filtriranje zasnovano na korisniku i kolaborativno filtriranje zasnovano na stavkama.

Najveći izazov sa kojim se standardna metoda kolaborativnog filtriranja suočava je skalabilnost i oskudnost matrica. Metod faktorizacije matrice je naprednija tehnika za razvoj sistema za preporuku koja razlaže originalnu retko posednutu matricu na nisko-dimenzionalne matrice sa latentnim faktorima/karakteristikama.

Prednost kolaborativnog filtriranja je da ono ne zahteva da budu poznate karakteristike predmeta ili korisnika. Takođe model može pomoći korisnicima da otkriju nova interesovanja. Kolaborativno filtriranje ne zahteva nikakvo domensko znanje, što je još jedna prednost ovog pristupa. [8]

Pored navedenih prednosti, kolaborativno filtriranje se suočava i sa nekim izazovima, poput hladnog starta za nove stavke koje se dodaju u listu. Takođe, proređenost podataka može uticati na kvalitet preporuka. Skalabilnost je još jedan izazov sa kojim se kolaborativno filtriranje suočava.

Pošto svaki tip algoritma koji se koriste kod sistema za preporuku ima svoje prednosti i nedostatke, obično je najbolje praviti hibridne sisteme za preporuku.

Kao praktični deo ovog rada razvijen je sistem za preporuku filmova zasnovan na kolaborativnom filtriranju koji koristi model najbližeg susedstva prilikom treniranja modela i podržava i pristup zasnovan na korisnicima i pristup zasnovan na stavkama, kao i dve različite metrike (korelacija, kosinusna sličnost) za pronalaženje sličnih korisnika ili stavki.

Razvijeni model treniran je nad celim skupom podataka, a testiran na 100 slučajnih korisnika. Sistem je postigao uspešnost predikcije oko 80% , za pristup zasnovan na stavkama, i oko 60%

za pristup zasnovan na korisnicima. Na osnovu rezultata evaluacije zaključuje se da metrika ne utiče značajno na uspešnost predikcije.

Dobijeni rezultati se mogu dodatno poboljšati uključivanjem pristupa zasnovanog na sadržaju, tj. atributa skupa podataka u proces analize. Skup podataka je već obrađen tako da se analiza atributa može uključiti bez ili sa minimalnim modifikacijama skupa podataka.

5.Literatura

[1] Build a Recommendation Engine With Collaborative Filtering, https://realpython.com/build-recommendation-engine-collaborative-filtering/, datum poslednjeg pristupa 22.6.2021.

[2] Introduction to Recommender System, https://towardsdatascience.com/intro-to-recommender-system-collaborative-filtering-64a238194a26, datum poslednjeg pristupa 22.6.2021.

[3] My Journey to building Book Recommendation System…, https://towardsdatascience.com/my-journey-to-building-book-recommendation-system-5ec959c41847, datum poslednjeg pristupa 22.6.2021.

[4] Recommendation Engine Models, https://dzone.com/articles/recommendation-engine-models , datum poslednjeg pristupa 22.6.2021.

[5] Build Recommendation Engine Using Graph, https://medium.com/tiket-com-dev-team/build-recommendation-engine-using-graph-cbd6d8732e46,datum poslednjeg pristupa 22.6.2021.

[6] Coefficient of determination, https://en.wikipedia.org/wiki/Coefficient_of_determination, datum poslednjeg pristupa 22.6.2021.

[7] R-Squared Definition, https://www.investopedia.com/terms/r/r-squared.asp, datum poslednjeg pristupa 22.6.2021.

[8] Collaborative Filtering Advantages & Disadvantages, https://developers.google.com/machine-learning/recommendation/collaborative/summary, datum poslednjeg pristupa 22.6.2021.