Projekat iz veštačke inteligencije

**Recommender sistemi**

Naziv tima: SuPerB

Članovi:

Miljana Simić 17401

Uroš Pešić 17356

Januar, 2022. Miljana Bogdanović 17019

1. Odeljak – Kratak opis problema

Recommender systems (sistemi za preporuke) su softverski alati i tehnike koji pružaju mogućnost raznovrsnih preporuka stavki koje mogu biti korisne za konkretnog korisnika. Predstavljaju podvrstu sistema za filtriranje informacija čiji je cilj da iskoriste dostupne informacije o preferencama korisnika za obezbeđivanje informisanih preporuka.

Danas se mehanizmi za preporuke koriste u raznim oblastima, kao što su online kupovina proizvoda, preporuka filmova, restorana, putovanja, generisanje muzičke play liste i mnoge druge. Ovi mehanizmi su prvenstveno usmereni ka pojednicima koji nemaju dovoljno ličnog iskustva da procene veliki broj stavki koje Web sajt može da ponudi. Sistemi za preporuke prikupljaju od korisnika njegove preferencije koje su izražene ili eksplicitno, recimo ocenom određenih proizvoda, ili implicitno određenim postupcima korisnika, na primer navigacija do određene web strane može da podrazumeva zainteresovanost za proizvode prikazane na toj stranici.

Mogućnost sistema za preporučivanje da daje personalizovane preporuke čini ih, možda, najvažnijom tehnologijom u kontekstu sveta online kupovine.

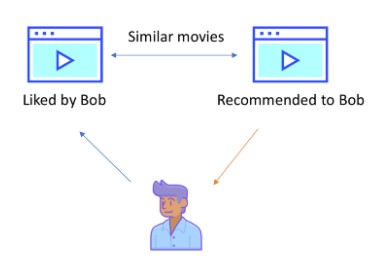
1. Odeljak – Pregled tehnika

Postoji nekoliko različitih tipova sistema za preporučivanje koji se razlikuju u adresnom domenu, korišćenom znanju, ali pre svega u algoritmu preporuke, odnosno kako se vrši predviđanje korisnosti preporuke. Neke od zastupljenih tehnika su:

* 1. Mehanizmi za preporučivanje zasnovani na sadržaju

Osnovna ideja ovog mehanizma je preporučivanje stavki slične onima za koje je korisnik prethodno pokazao interesovanje. Efikasnost ovog mehaniza zavisna je od mogućnosti određivanja sličnosti stavke sa drugim stavkama.

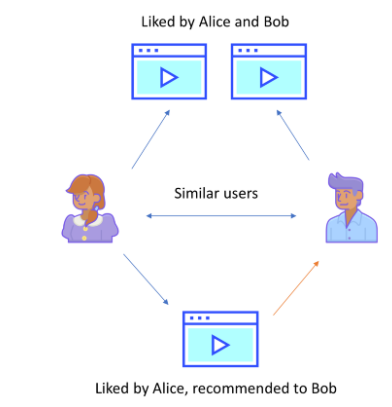
Na primer, ako korisnik oceni pozitivno film koji pripada jednom žanru, sistem će korisniku preporučiti filmove istog žanra.



* 1. Mehanizmi za preporučivanje kolaborativnim filtriranjem

Ovaj mehanizam preporučuje aktivnom korisniku stavke koje su se prethodno dopale sličnim korisnicima. Sličnost ukusa dva korisnika se ogleda u sličnosti istorije njihovih interesovanja. Ako se posmatra istorija sličnih korisnika, stavke koje se ne poklapaju postaju osnova za buduće preporuke.

Na primer, ako dva korisnika ocene pozitivno iste filmove, sistem će korisniku preporučiti filmove koje je njemu sličan korisnik ocenio pozitivno.



* 1. Mehanizmi za preporučivanje zasnovani na demografiji

Ideja ovog mehanizma je da preporučuje na osnovu demografskog profila korisnika. Na primer korisnicima se preporučuje određeni sadržaj na osnovu njihovog jezika, države u kojoj žive ili njihovom uzrastu.

* 1. Mehanizmi za preporučivanje zasnovani na znanju

Ovaj mehanizam preporučuje stavke na osnovu znanja o specifičnom domenu, kako određene karakteristike stavke zadovoljaju potrebe korisnika i koliko je ta stavka korisna za korisnika.

* 1. Mehanizmi za preporučivanje zasnovani na zajednici

Ideja ovog mehanizma je da se stavke preporučuju korisniku na osnovu preferencija njegovih prijatelja. Statistika pokazuje da se ljudi više oslanjuju na preporuke prijatelja nego na preporuke anonimnih pojedinaca.

* 1. Hibridni mehanizmi za preporučivanje

Ovaj mehanizam je zasnovan na kombinaciji 1. i 2. mehanizma. Teži da koristi prednosti mehanizma zasnovanog na sadržaju, a da popravi nedostatke kolaborativnog filtriranja. Na primer, mehanizam kolaborativnog filtriranja sprečava da se korisnicima preporučuju novi proizvodi, što nije slučaj sa 1. mehanizmom.

Kolaborativno filtriranje je jedna od najčešće korišćenih tehnika koj se koriste za kreiranje inteligentnih recommender sistema koji daju sve bolje preporuke sa prilivom novih informacija o korisnicima. U daljem teksu ćemo obraditi jedan recommender sistem koristeći kolaborativno filtriranje na primeru sistema za preporuku filmova.

Pre svega, za testiranje algoritama za preporuku neophodan nam je skup podataka koji ćemo obrađivati tzv. dataset. Struktura datasetova može biti vrlo raznolika, ali jedna od formi koja se vrlo često sreće jeste matrična notacija. Na primeru recenzija filmova, vrste matrice bi predstavljale korisnike, a kolone filmove. U preseku vrste i kolone nalazile bi se ocene tog korisnika za konkretan film. S obzirom da broj filmova koje bi korisnici ocenili nije veliki, retko posednuta matrica bi bila dobar izbor za čuvanje podataka.

Jedan od prvih koraka u kolaborativnom filtritanju jeste nalaženja sličnih korisnika, a potom pretpostavljanje ocena koje bi korisnik dao filmovima koje još uvek nije recenzirao, na osnovu prethodnih ocena. S obzirom na izabranu metodu, za sličnost korisnika se neće koristiti žanr filma, godine, pol itd. Dakle nijedna infomacija o korisnicima niti filmova, sem ocene koju je korisnik dao za određeni film. Takođe je potrebno voditi računa o tačnosti prethodno generisanih prepostavki.

Razmotrićemo dve kategorije algoritama za generisanje pretpostavki Memory Based i Model based.

1. Memory Based

Memory based algoritme karakterišu statičke metode koje se primenjuju na ceo dataset kako bi se generisale prepostavke. Ukoliko želimo da procenimo ocenu O koju bi dao korisnik K za film F, potražičemo sve korisnike Ki koji su slični korisniku K, a da su pritom ocenili fim O. Zatim ćemo na osnovu njihovih ocena, generisati ocenu koju bi korinik K mogao da da.

* Pronalaženje sličih korisnika

Recimo da imamo 4 korisnika i 2 filma tako da su njihove ocene sledeće

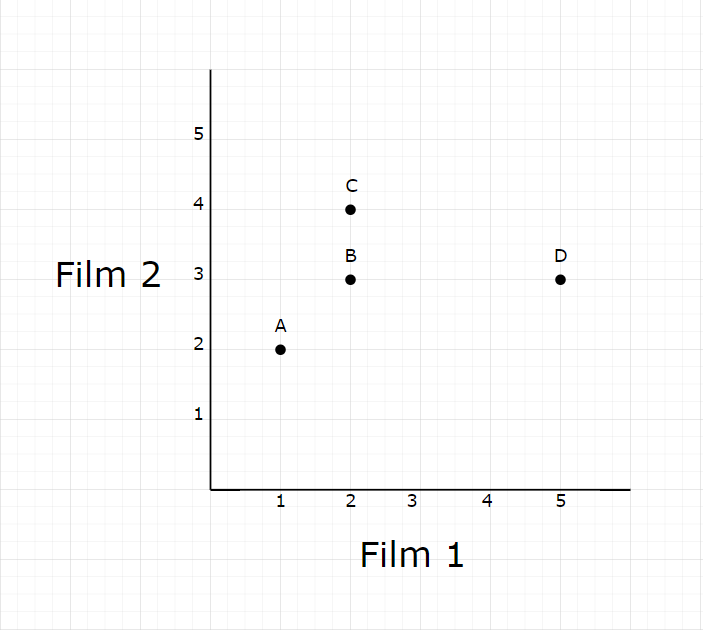
A: 1.0, 1.0

B: 2.0, 4.0

C: 4.0, 4.0

D: 5.0, 3.0

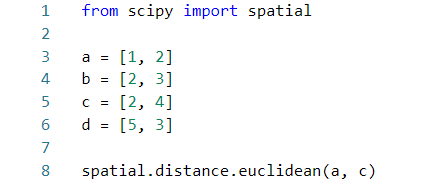
Zbog lakše vizualizacije izabrana su 2 filma, u realnosti, taj broj bi bio dosta veći.



Za procenjivanje sličnosti, mogli bismo da iskoristimo udaljenost dvaju tačaka, odnosno Euklidsku razdaljenost. Za dve tačke A(ax, a y) i C(cx, cy), udaljenost bi se računala kao:

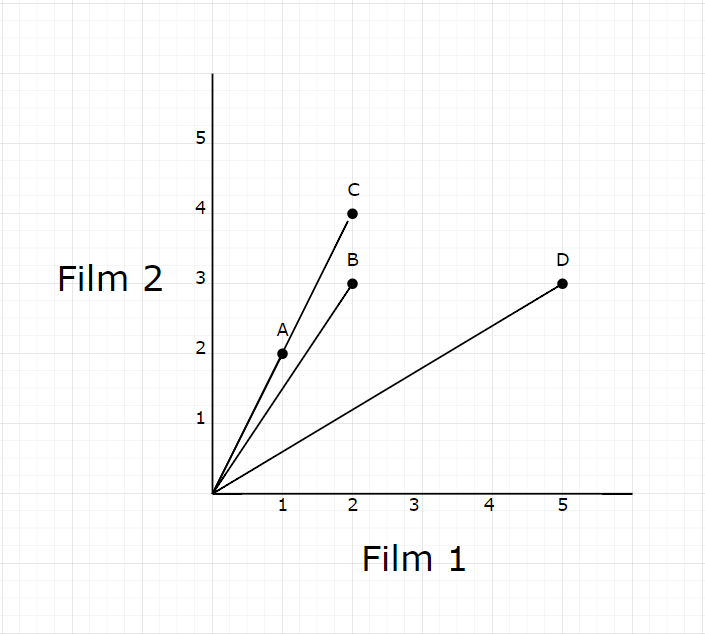
A u opštem slučaju, u Euklidskom n-prostoru:

U programskom jeziku Python uz pomoć biblioteke „scipy“ to bi bilo urađeno na sledeći način:



Na osnovu udaljenosti, reklo bi se da su A i B najsličniji korisnici, ali zapravo se korisnicima A i C drugi film dopao duplo više nego prvi. Međutim to je nešto što euklidska udaljenost ne može da uoči. Stoga uvodimo drugu tehniku za ocenu sličnosti – kosinusnu sličnost.

Kosinusna sličnost ne bi u obzir uzimala udaljenost tačaka, već ugao koji one zaklapaju. Ukoliko je ugao manji, sličnost bi bila veća.

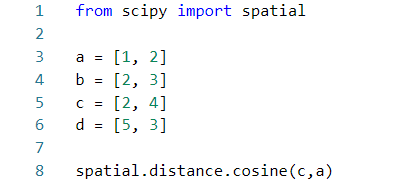


Kosinusna udaljenost se računa po sledećoj formuli:

Sc(A, C)

A kosinusna sličnost se računa kao 1-Sc.

U programskom jeziku Python to bi bilo urađeno na sledeći način:



Međutim, kosinusna distanca i dalje nije dovoljno dobar način za određivanje pretpostavki, jer za korisnike sa različitim preferencijama vraća apsolutnu sličnost, što nije tako retko. Zbog toga je nephodno dovesti sve korisnike na isti nivo, odnosno eliminisati njihovu pristrasnost. To ćemo uraditi tako što ćemo za svakog korisnika izračunati prosečnu ocenu za sve filmove koje je ocenio, a potom tu vrednost oduzeti od od svake ocene koju je dao.

* Procena ocene koju bi dao korisnik

Kada smo pronašli slične korisnike, potrebno je da na osnovu njihovih ocena procenimo koju ocenu bi dao korisnik K. Jedan od načina za to bi bila prosečna vrednost. Međutim ona ne bi dala realnu sliku jer postoji mogućnost da je samo par korisnika dosta slično korisniku K, a ostali i ne toliko. Stoga, prilikom računjanja prosečne vrednosti unosimo i težine koje te ocene nose. Ukoliko bi sličost korisnika Ki sa korisnikom K bila Si, a ocena koju je dao Oi, prosečnu ocenu bismo računali na sledeći način:

Do sada smo razmatrali samo sličnost korisnika, međutim, moguće je uzeti u obzir i sličnost filmova. Naime, ukoliko postoji više korisnika nego filmova, ovaj drugi pristup bi bio mnogo brži. Recimo da za film F pronađemo slične filmove, a onda ocenu koju bi korisnik dao za taj film, odredimo na osnovu filmova sličnih filmu F a da ih je ocenio korisnik K.

1. Model Based

Model Based pristup uključuje smanjivanje odnosno kompresovanje matrice podataka primenom faktorizacije. Dimenzije matrice određene su brojem korisnika i brojem filmova. Kao što je već pomenuto, zbog slabe popunjenosti matrice, mnoga polja ostaju prazna, te smanjivanje dimenzija matrice znatno utiče na performanse algoritma. Dakle matricu dimenzija mxn moguće je prestaviti kao proizvod matrica mxp i pxn. Takve matrice predstavljaju korisnike i filmove pojedinačno. Za matricu mxp gde je m broj korisnika, kolone predstavljaju latentni faktor i pokazuju karakteristike korisnika. Analogno i za filmove. Npr. Recimo da za jednog korisnika imamo vektor (2, -2). Taj vektor bi predstavljao korisnika koji voli dokumentarne filmove i ostavlja im pozitivne ocene (2), a ne voli romantične filmove i ocenjuje ih negativno. Što se tiče filmova, vektor (2, 1) predstavlja koliko taj film pripada kojoj kategoriji. Ocenu bismo mogli da dobijemo kao 2\*2+1\*(-2)=2. U realnosti, ovi vektori su mnogo veći. Što je veći broj latentnih faktora, personalizacija filmova je bolja.

Razmotrićemo jedan algoritam za faktorizaciju matrica – SVD.

* Singular Value Decomposiition

Singular Value dekompozicija matricu M mxn dekomponuje kao:

Gde je U matrica mxm, je dijagonalna matrica dimenzija mxn, a

Je nxn matrica. Matrice U I su ortonormalne matrice, što znači da važi

, odnosno . Matrica je pravougaona i sadrzi opisuje jačinu svakog latentnog faktora, dok matrice U i V prikazuju vezu između korisnika i latentnih faktora, tj. filmova i latentnih faktora, respektivno.

U Pythonu dunkcija svd iz paketa numpy.linalg određuje dekompoziciju matrica, s tim što za dijagonalnu matricu vraća vrednosti koje se nalaze na njenoj dijagonali.