Januar 2020
Univerzitet u Nišu, Elektronski Fakultet

RECOMMENDER SISTEMI

Projekat iz veštačke inteligencije

RBP grupa:
Petar Mićić 16232
Radiša Marković 16187
Branko Simović 16326

Sadržaj

Uvod

Recommender sistem je, prosto rečeno, sistem koji treba da što je bolje moguće odredi prioritete nekih entita (proizvoda, filmova, muzike, itd.) koji bi se trebali preporučiti nekom korisniku. Primena recommender sistema je danas veoma široka, uz masovan porast elektronske trgovine, online prodavnica i raznih drugih servisa pa i online marketinga, precizni i „pametni“ sistemi preporuka proizvoda/servisa krajnjim korisnicima daju veliki doprinos unapređenju poslovanja.

Pristupi

Bez obzira na konkretnu oblast primene, postoje dva osnovna pristupa primenjiva bilo gde. To su **kolaborativno flitriranje** i **filtriranje na osnovu sadržaja**. Ključna razlika između njih je ta da kolaborativno filtriranje ne uzima u obzir nikakve informacije o samom proizvodu već daje preporuke čisto na osnovu raspodela ocena proizvoda između različitih korisnika dok filtriranje na osnovu sadržaja je bazirano na sličnostima između novih proizvoda i proizvoda koji su se korisniku ranije dopali.

Kolaborativno filtriranje

Ideja je ta da se za ljude kojima se poklapaju ocene većina proizvoda može pretpostaviti da će se poklapati i ocene za ostale proizvode.

Kolaborativno filtriranje se može vizualizovati **matricom interkacje**. Za skup korisnika K, skup proizvoda P, i matricu interakcije $M_{|K|\times|P|}$ element matrice interakcije o_{ij} je „ocena“ koju je korisnik K_i „dao“ proizvodu P_j. Ocena ne mora biti ocena u bukvalnom eksplicitnom smislu, to može biti vrednost izračunata na osnovu implicitnih interakcija korisnika sa tim proizvodom (npr. vreme koje je korisnik proveo na stranici tog proizvoda u online prodavnici). Takođe, element matrice može biti prazan u slučaju da ne postoji dovoljno informacija o interakciji tog korisnika i proizvoda da bi se dodelila ocena.

Matrica interakcije (izvor: Google Developers)

Algoritam k-najbližih suseda

Najprostiji algoritam preporuke kod kolaborativnog filtriranja je **algoritam k-najbližih suseda**. Potrebno je pre svega izračunati sličnosti između korisnika kome tražimo preporuku i ostalih korisnika. Kao funkcija računanja sličnosti može se iskoristiti formula kosinusne sličnosti (ukoliko je ocena realan broj):

$$slicnost(x,y) = \frac{\vec{x} \cdot \vec{y}}{\|\vec{x}\| \times \|\vec{y}\|} = \frac{\sum_{i \in I_{xy}} r_{x,i}\, r_{y,i}}{\sqrt{\sum_{i \in I_{xy}} r_{x,i}^2} \sqrt{\sum_{i \in I_{xy}} r_{y,i}^2}}$$

Gde je I_{xy} skup svih proizvoda koji su ocenili oba korisnika, $\vec{x}$ i $\vec{y}$ vektori ocena korisnika x i y, a r_{xi} i r_{yi} ocena korisnika x ili y koja odgovara proizvodu i.

Nakon toga, treba izdvojiti skup k-najbližih suseda i izračunati njihovu prosečnu ocenu za svaki proizvod koji korisnik kojem dajemo preporuku nije ocenio.

Preporučiti N proizvoda sa najvećom prosečnom ocenom.

Pseudokod algoritma:

```
N – broj proizvoda koji treba preporuciti
k – broj suseda koji treba uzeti u obzir
u – korisnik kome treba dati preporuke
P – skup svih proizvoda
K – skup svih korisnika bez u

knn(N, k, u, P, K)
      slicnosti = {}
      foreach korisnik in K
            add(slicnosti, {korisnik, slicnost(u, korisnik)})
      sort_descending(slicnosti)
      slicnosti = top(k, slicnosti)

      prosecne_ocene = {}
      foreach proizvod in P
            if proizvod not in u.ocenjeni_proizvodi then
                  add(prosecne_ocene, {proizvod, prosek(proizvod,
                  slicnosti)})
      sort_descending(prosecne_ocene)

      return top(N, prosecne_ocene)
```

Prednosti

Nisu potrebne nikakve informacije o proizvodima, bazira se samo na interakciji između korisnika i proizvoda.

Pomaže korisnicima u proširivanju svojih interesovanja. Sistem ne zna kategorije proizvoda niti da li je korisnik stvarno zainteresovan za određeni proizvod ali će ga sve jedno preporučiti ako su slični korisnici zainteresovani.

Mane

Glavni problem je tzv. **hladan početak**. Za bilo kojeg novog korisnika ili novi proizvod ne postoje podaci koji bi mogli da se iskoriste za računanje preporuke.

Drugi problem je generalna retkost matrice. Broj proizvoda i korisnika je obično veoma veliki a većina korisnika će oceniti samo nekoliko proizvoda tako da će proizvodi, čak i oni najpopularniji, imati mali broj ocena u odnosu na broj korisnika.

Filtriranje na osnovu sadržaja

Ovaj pristup se bazira na sličnosti između samih proizvoda. Na primer, logična pretpostavka je da neko ko je gledao mnogo akcionih filmova sa Arnoldom Švarcenegerom voli da gleda akcione filmove sa tim glumcem i da mu ih treba preporučiti.

Algoritam

Za filtriranje na osnovu sadržaja potrebna je dobra kategorizacija, tzv. „**tagovanje**“, proizvoda. Ako imamo skup tagova (prideva i osobina) T, svakom proizvodu iz skupa proizvoda P možemo dodeliti podskup skupa T koji odgovara tom proizvodu.

Svakom korisniku takođe treba dodeliti tagove koje preferiraju, bilo eksplicitno (gde oni sami biraju) ili implicitno na osnovu prethodnih podataka (implicitni način ima problem hladnog početka) ili kao kombinaciju oba.

Sledeći korak je proći kroz sve proizvode kojima se tagovi poklapaju sa bar jednim tagom proizvoda, sortirati ih u opadajući redosled po broju poklopljenih tagova i preporučiti prvih N korisniku.

```
N – broj proizvoda koji treba preporuciti
u – korisnik kome treba dati preporuke
P – skup svih proizvoda koje korisnik jos nije video

sadrzajFilter(N, u, P)
      potencijalne_preporuke = {}
      foreach proizvod in P
            s = 0
            foreach tag in P.tagovi
                  if tag in u.tagovi then
                        s = s + 1
            if s > 0 then
                  add(potencijalne_preporuke, {proizvod, s})
      sort_descending(potencijalne_preporuke)

      return top(N, potencijalne_preporuke)
```

Prednosti

Budući da ne koristi informacije o drugim korisnicima, može preporučiti nepopularne ili gotovo nepoznate proizvode ako se poklapaju sa preferencama korisnika kome traži preporuku.

Izbegava hladan početak za proizvode pošto se podrazumeva da su svi proizvodi već kategorizovani.

Mane

Kategorizacija proizvoda mora biti urađena ručno, sistem je dobar onoliko koliko je dobro urađena kategorizacija.

Sistem može samo preporučivati proizvode bazirano na interesovanjima korisnika, nije dobar za proširivanje interesovanja.

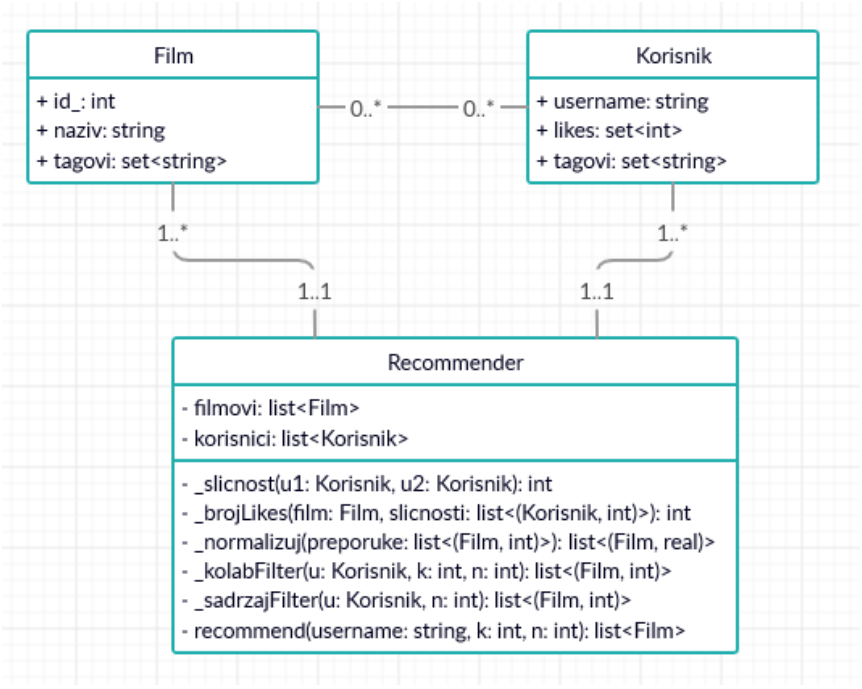
Projektovanje recommender sistema za servis streamovanja filmova

U nastavku sledi opis projektovanja prostog recommender sistema za filmove u programskom jeziku python. Za pokretanje potreban je python 3.7+, pokrenuti fajl `main.py`.

Sistem je baziran na kombinaciji kolaborativnog filtriranja i filtriranja na osnovu sadržaja i koristiće unapred popunjenu bazu podataka filmova i korisnika. U svrhu jednostavnosti podaci su predstavljeni u JSON obliku.

Kao ocenu filma koristićemo sistem „lajkova“ gde korisnici imaju listu filmova koji ima se dopadaju i koje podrazumevamo da su već gledali.

Opis programa



Klasa Film sadrži jedinstveni identifikator filma, naziv filma, i listu tagova koji odgovaraju tom filmu.

Klasa Korisnik sadrži username korisnika, listu identifikatora filmova kojima je dao like i listu tagova koje taj korisnik preferira.

Klasa Recommender ima kao atribute liste filmova i korisnika i implementira metode za preporuku.

Kolaborativni filter

Kolaborativni filter (`_kolabFilter`) koristi algoritam k-najbližih suseda (opisan gore). Za funkciju sličnosti između dva korisnika (`_slicnost`) računa se broj filmova kojima su oba korisnika dala like. Funkcija vraća listu N uređenih parova (Film, Težina), gde je težina broj lajkova (`_brojLikes`) koji je odabranih k korisnika dalo za dati film.

Filter sadržaja

Filter sadržaja (`_sadrzajFilter`) implementira algoritam naveden u odeljku „Filtriranje na osnovu sadržaja“ izmenjen tako da neće preporučivati filmove kojima je korisnik dao like. Takođe vraća listu N uređenih parova (Film, Težina), ovaj put je težina broj tagova filma koji se poklapaju sa preferencama korisnika.

Recommender

Sam recommender (`recommend`) kombinuje izlaze dva filtera tako što normalizuje težine filmova na interval [0, 1] (`_normalizuj`), soritra po normalizovanim težinama i izbacuje duple preporuke (ostavljajući onu sa većom težinom). Uzima prvih N i vraća kao listu objekata Film.

Formula normalizacije:

$$norm_i = \frac{t_i - \min(T)}{\max(T) - \min(T)}$$

Gde je t_i težina a T lista svih težina.

Glavni program

Glavni program učitava podatke iz fajlova `filmovi.json` i `korisnici.json` i za uneti username, broj preporuka i broj sličnih korisnika koji treba gledati ispisuje listu preporučenih filmova

Primer rada

Uzmimo za primer korisnika sa username-om „pera“.

Njemu se dopadaju sledeći filmovi:

- Last Christmas
- The Matrix
- The Godfather
- Reservoir Dogs

Kao i tagovi:

- komedija
- krimi

- fantastika

Preporučimo mu 3 filma gledajući 3 sličnih korisnika. Algoritam k-najbližih će izračunati da su njemu najbliži korisnici:

- „stefan“ sa dva poklapanja
- „branko“ sa jednim poklapanjem
- „jelena“ sa jednim poklapanjem

Između njih tri, algoritam je našao jedan film koji ima dva lajka: „Noelle“, ostali svi imaju po jedan ili nula.

Algoritam preporuke na osnovu sadržaja će naći dva filma koja nije gledao a koja se poklapaju sa dva od tri njegova navedena taga: „Bad boys for life“ i „Dolittle“.

Tako smo došli do tri filma koji će nakon normalizacije imati najveće težine:

- Noelle
- Bad boys for life
- Dolittle

Njih ćemo preporučiti korisniku.

Literatura

- https://developers.google.com/machine-learning/recommendation
- https://towardsdatascience.com/introduction-to-recommender-systems-6c66cf15ada
- https://medium.com/recombee-blog/machine-learning-for-recommender-systems-part-1-algorithms-evaluation-and-cold-start-6f696683d0ed