Sistem de recomandare al filmelor

Facultatea de Automatică și Calculatoare

Ingineria Sistemelor

02.06.2021

1. Introducere

Prezicerea continutului pe care un utilizator il doreste este in zilele de astazi o problema foarte importanta pentru site-uri, magazine, platforme de filme si multe altele. Platforme precum Netflix, Amazon, Youtube au dezvoltat sisteme foarte sofisticate de recomandare a continutului nou si relevant pentru utilizatori. Aceste sisteme sunt una dintre cele mai de valoare bunuri ale unei companii, asa cum a fost demonstrat de Neflix prin competitia soponsorizata cu un premiu de 1 milion de dolari, pentru imbunatatirea sistemului lor. Aceste sisteme sunt cunoscute dupa numele de „Sisteme de recomandare”.

Sistemele de recomandare pot avea diferite abordari de obtinere a rezultatelor. O sarcina importanta pe care un sistem de recomandare trebuie sa o faca este de a face o prezicere despre cum unui utilizator anume s-ar putea sa-i placa un produs, bazat pe comportamentul anterior al acestuia dar si al altor useri. Cele mai comune tehnici dezvoltate sunt:

1. Filtrarea bazata pe continut (Content-based filtering)
2. Filtrarea colaborativa (Collaborative filtering)

Ambele au avantajele si dezavantajele lor. Filtrarea bazata pe continut compara atributele produselor si face recomandari gasind produse care sunt similare cu cele pe care utilizatorul le-a placut anterior. Pe de alta parte, filtrarea colaborativa incearca sa gaseasca utilizatori care au gusturi similare si fac predictii pe baza modului in care acesti utilizatori interactioneaza cu diferite produse. Filtrarea colaborativa se foloseste de relatiile utilizator-produs pentru a face preziceri pentru utilizator. Aceasta abordare este foarte puternica, deoarece sistemului nu-i sunt necesare informatii despre atributele produselor. Prin urmare, se poate aplica foarte usor oricarui set de date cu relatii user-item.

Metodele de filtrare colaborativa, pot fi impartite in doua categorii: memory-based si modelbased. Abordarile de tip memory-based saunt de obicei simplu de implementat si pot furniza socruri de predictie foarte bune. Cu toate acestea, aceste tehnici s-au dovedit a fi ineficiente si greu de scalat pentru seturi de date foarte mari, comune in aplicatiile de zi cu zi.

In acest studiu, vom investiga, prin urmare, abordarea de tip model-based in vederea construirii unui sistem de recomandare pentru filme. Filtrarea colaborativa bazata pe model foloseste algoritmi de machine learning pentru a crea un model bazat pe datele de antrenare, dupa care foloseste modelul pentru a face predictii. Vom compara doua implementari ale factorizarii de matrici, folosind stochastic gradient descent (SGD), si alternating least squares (o variatie a metodei CMMP).

2. Background

2.1 Sistem de recomandare

Un sistem de recomandare este un sistem ce recomanda produse unui anumit client. In general, dandu-se un produs si un user sistemului de recomandare acesta trebuie sa gaseasca similaritati user-item si sa prezica cat de mult ii va placea produsul utilizatorului respectiv. Datele sistemelor de recomandare includ useri, produse (melodii, retete, filme, haine, electronice, etc.), caracteristicile produselor si interactiunile user-item. Aceste interactiuni includ date precum vizualizarea, cumpararea, like/dislike, rating (1-5), sau timp vizionat.

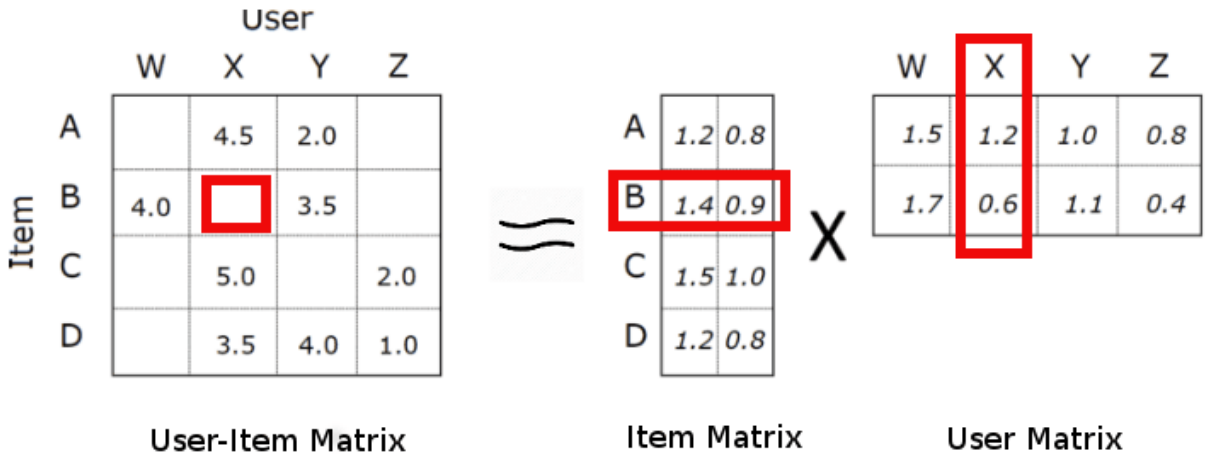
Sistemele de recomandare sunt folositoare atunci cand exista o abundenta de continut care nu este relevant unui user. Scopul este de a creste numarul de produse vandute, timpul petrecut interactionand cu platforma, vanzarea de produse variate sau de crestere a satisfactiei clientilor.

2.2 Collaborative filtering (CF)

Sistemele de recomandare bazate pe filtrarea colaborativa fac predictii pe basa relatiilor user-item, fara a detine informatii despre useri sau produse. Doua metode comune in abordarea acestei probleme sunt reprezentate de neighborhood-based CF si latent factor models. Metodele Neighborhood-based fac predictii calculand similaritati intre useri si produse bazate pe relatia user-item. Abordarea orientata pe user se concentreaza pe alti useri cu interactiuni similare atunci cand face predictii. Metoda orientata pe produs, pe de alta parte, calculeaza o similaritate produs – produs si face predictii cantarind produsele similare carora utilizatorul le-a acordat un rating anterior. Modelul ce foloseste latent factors, foloseste o matrice user-item ce contine ratingurile userilor pentru fiecare item, incercand sa caracterizeze atat userii cat si produsele cu un numar de

caracteristici latente. Numarul de factori poate, in mod normal, varia intre 10-100 pentru fiecare user si item. Vectorii factori pentru un user si item pot fi inmultiti pentru a gasi un rating prezis pentru item. Acesti factori pot fi vazuti ca niste caracteristici deduse despre un produs sau utilizator. In contextul filmelor, o caracteristica poate fi reprezentata de gen, varsta grupului tinta, lucruri mai putin evidente, precum timpul de dezvoltare a unui personaj, etc. O metoda ce foloseste vectorii latenti pentru sistemele de recomandare se numeste **matrix factorization** (factorizarea de matrici).

3. Matrix factorization



Aceastea este o tehnica care calculeaza modelul factorilor latenti al unui siste bazat pe interactiunea dintre utilizator si filme. Aceasta relatie user-film poate fi reprezentata ca o matrice cu useri pe coloane si filme pe linie. Interactiunile sunt reprezentate de ratinguri date de useri filmelor (de la 1 la 5). Elementele din aceasta matrice sunt foarte imprastiate, deoarece utilizatorii dau rating doar unei mici parti din filmele prezente in sistem. Aceasta metoda a aratat ca este capabila sa faca predictii foarte bune si in cazul matricelor carora le lipsesc foarte multe elemente. Metoda reduce dimensiunea matricei cu ratinguri Y, factorizand-o intr-un produs de doua matrice, P, pentru filme si Q pentru utilizatori.

$$\begin{bmatrix} y_{11} & \cdots & y_{1u} \\ \vdots & \ddots & \vdots \\ y_{i1} & \cdots & y_{iu} \end{bmatrix} = \begin{bmatrix} p_1 \\ p_2 \\ \vdots \\ p_i \end{bmatrix} \text{ x } [q_1 \quad \dots \quad q_u]$$

i x u i x f f x u

$$Y = PQ^T \tag{1}$$

f reprezinta numarul de caracteristici extrase, u – numarul de useri, iar i – numarul de filme. Fiecare rand p_i este un vector de caracteristici pentru un film, iar fiecare coloana q_u este un vector de caracteristici pentru un user. Produsul acestora creeaza o estimare a ratingului origina

$$y_{iu} = p_i \; x \; q_u^T \tag{2}$$

Exista o multime de procedee de factorizare a matricelor in mai multe componente folosite in machine learning sau statistica, dar majoritate nu functioneaza atunci cand din matrice lipsesc foarte multe elemente. O metoda este reprezentata de factorizarea matricei folosind ratingurile deja stiute si incercarea de minimizare a erorii patratice:

$$min \sum_{i,u} (y_{iu} - p_i q_u^T)^2 \tag{3}$$

Totusi, asta poate rezulta in overfittarea datelor de training. Pentru a preveni acest lucru, se introduce un termen de regularizare erorii patratice. Impactul acestuia este controlat de constanta β.

$$min \sum_{i,u} (y_{iu} - p_i q_u^T)^2 + \beta \, (\|p_i\|^2 + \|q_u\|^2) \tag{4}$$

, unde $\| . \|$ reprezinta norma frobenius. Aceasta abordare s-a dovedit a fi foarte eficienta si in acelasi timp scalabila din punct de vedere al timpului asupra seturilor de date cuprinzatoare. Prin urmare pasii acestei metode sunt urmatorii:

1. Initializarea matricelor P si Q de dimensiuni i x f, respectiv u x f, unde i, u, f, reprezinta numarul de filme, useri si caracteristici, cu valori aleatoare in intervalul [0, 1].
2. Se itereaza peste toate ratingurile deja stiute din setul de date:
    a) Se calculeaza ratingul prezis corespunzator unui rating deja stiut;
    b) Se calculeaza eroarea ratingului prezis;
    c) Se actualizeaza P si Q, corespunzator cu eroarea.

3.1 Stochastic gradient descent

Algoritmul Stochastic gradient descent (SGD) rezolva aceasta problema de optimizare (4). Acest algoritm parcurge fiecare rating din datele de training, incearca sa prezica ratingul si calculeaza eroarea de predictie:

$$e_{iu} = y_{iu} - p_i q_u^T \tag{5}$$

Dupa aceea, se actualizeaza vectorii p_i si q_u cu un factor proportional cu o constanta α, care reprezinta pasul de invatare (learning rate).

$$Iteratia\ Metodei\ Gradient: x_{k+1} = x_k - \alpha \nabla f(x_k)$$

$$q_u = q_u - 2\alpha(e_{iu} p_i - \beta q_u) \tag{6}$$

$$p_i = p_i - 2\alpha(e_{iu} q_u - \beta p_i) \tag{7}$$

Iteratiile peste date continua, calculand eroarea si actualizand vectorii p_i si q_u, pana la convergenta sau o eroare de aproximatie satisfacatoare fata de matricea initiala este atinsa.

3.2 Alternating Least Squares

Alternating least squares (ALS, o variatie a algoritmului CMMP) este o varianta alternativa de rezolvare a problemei de optimizare (4). Aceastea metoda functioneaza luand alternativ vectorii p_i si q_u, fixandu-l pe unul si optimizandu-l pe altul rezolvand problema celor mai mici patrate pana cand convergenta este gasita. Abordarea aceasta este inceata in comparatie cu SGD, dar are avantajul ca poate fi paralelizata.

Daca pentru minimizarea functiei obiectiv aplicam metoda celor mai mici patrate, dar o facem alternativ, fixand p, calculand q, si invers, atunci solutiile acestor probleme vor fi date de ecuatiile urmatoare:

$$p_i = (y_{iu} q_u \, (\, q_u^T q_u + \, \beta I))^{-1} \tag{8}$$

$$q_u = (y_{iu} q_u \, (\, p_i^T p_i + \, \beta I))^{-1} \tag{9}$$

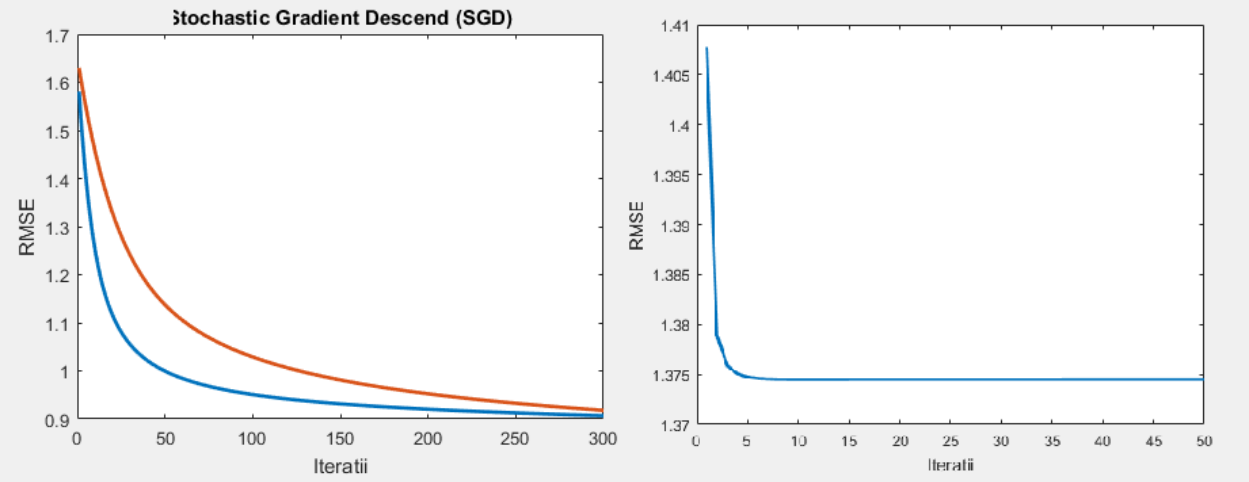
4. Evaluare

Pentru evauarea algoritmilor, a fost folosita eroarea medie patratica (RMSE), ce reprezinta deviatia standard dintre un set de valori estimate si valorile actuale. Intr-un sistem de recomandari se foloseste pentru a masura cat de departe de valorile reale au fost valorile prezise.

$$RMSE = \sqrt{\frac{1}{n} \sum (\, t_{iu} - \, y_{ui} \,)^2} \tag{10}$$

, unde t reprezinta matricea cu datele de testare, iar y, matricea cu datele prezise.

SGD – Train Data vs. Test Data **ALS**



Tabel comparație SGD - ALS

	SGD		ALS	
Iterații	RMSE	Timp (s)	RMSE	Timp (s)
5	1.4292	0.86	1.3868	23.14
10	1.2819	1.72	1.3801	45.55
50	0.9993	17.77	1.3744	340.93
100	0.9516	38.27	-	-
150	0.9331	54.94	-	-

Functia fminunc() minimizeaza eroarea patratica in 5.57, cu maxim 300 de iteratii si cu RMSE = 1.5096.

5. Concluzie

Rezultatele arata ca factorizarea matriceala este o abordare potrivit de buna pentru implementarea sistemelor de recomandare. Aceasta abordare este si foarte scalabila, folosind putina memorie pentru calculul ratingurilor. Algoritmul SGD iterează rapid peste seturi de date relativ mici, dar pentru seturi de date mari, fiecare iterație durează foarte mult si pentru a atinge o eroare optima, este nevoie de multe iterații. Pentru seturi de date foarte mari, ALS se dovedește a fi mai practic, deoarece acesta se poate paraleliza foarte ușor.

6. *Referințe*

- Francesco Ricci, Lior Rokach, and Bracha Shapira. Introduction to recommender systems handbook. Springer, 2011.
- Gediminas Adomavicius and Alexander Tuzhilin. Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. Knowledge and Data Engineering, IEEE Transactions on, 17(6):734–749, 2005.
- Yehuda Koren, Robert Bell, and Chris Volinsky. Matrix factorization techniques for recommender systems. Computer, (8):30–37, 2009.
- https://github.com/kk289/ML-Anomaly_Detection_and_Recommender_Systems-MATLAB
- https://github.com/StephenWuHao1212/COMPSCI-571-Machine-Learning/blob/master/Project%20Report(Movie%20Recommender%20System).pdf
- https://medium.com/analytics-vidhya/math-behind-content-based-recommendation-system-a7e440c96fa
- https://upscfever.com/upsc-fever/en/data/en-exercises-25.html
- https://github.com/Mogbo/Movie-Recommender-System/blob/master/Project_Report.pdf
- https://towardsdatascience.com/recommendation-system-matrix-factorization-d61978660b4b
- https://towardsdatascience.com/paper-summary-matrix-factorization-techniques-for-recommender-systems-82d1a7ace74
- https://developers.google.com/machine-learning/recommendation/collaborative/matrix
- https://github.com/richashah1106/movie-recommendation-system/blob/master/Report/EE239AS%20Project%201%20Report%20(PDF%20Version).pdf
- https://hal.archives-ouvertes.fr/hal-01314906/document
- http://repository.bilkent.edu.tr/bitstream/handle/11693/50632/%C3%96FA_MastersThesis.pdf?sequence=1&isAllowed=y
- https://ranger.uta.edu/~heng/CSE6389_15_slides/SGD1.pdf
- https://www.holehouse.org/mlclass/16_Recommender_Systems.html

- https://stanford.edu/~rezab/classes/cme323/S16/projects_reports/baalbaki.pdf
- https://medium.com/recombee-blog/machine-learning-for-recommender-systems-part-1-algorithms-evaluation-and-cold-start-6f696683d0ed
- https://link.springer.com/article/10.1007/s12525-018-0297-2
- https://datasciencemadesimpler.wordpress.com/tag/alternating-least-squares/#ALS
- https://stanford.edu/~rezab/classes/cme323/S15/notes/lec14.pdf
- http://cs229.stanford.edu/proj2014/Christopher%20Aberger,%20Recommender.pdf
- https://datajobs.com/data-science-repo/Recommender-Systems-[Netflix].pdf
- http://web.cs.ucla.edu/~chohsieh/teaching/CS260_Winter2019/lecture13.pdf
- https://www.academia.edu/6220338/Fast_als_based_matrix_factorization_for_explicit_and_implicit_feedback_datasets
- https://hellanicus.lib.aegean.gr/bitstream/handle/11610/18038/Matrix_Factorization_techniques_for_Recommender_Systems.pdf?sequence=1&isAllowed=y
- https://www.cs.ubc.ca/~schmidtm/Courses/540-W19/L11.pdf
- https://www.stat.cmu.edu/~ryantibs/convexopt-F18/lectures/stochastic-gd.pdf
- https://stats.stackexchange.com/questions/323570/convergence-of-stochastic-gradient-descent-as-a-function-of-training-set-size
- https://www.diva-portal.org/smash/get/diva2:927190/FULLTEXT01.pdf
- https://stanford.edu/~rezab/classes/cme323/S15/notes/lec14.pdf
- https://perso.uclouvain.be/paul.vandooren/ThesisHo.pdf