به نام خدا

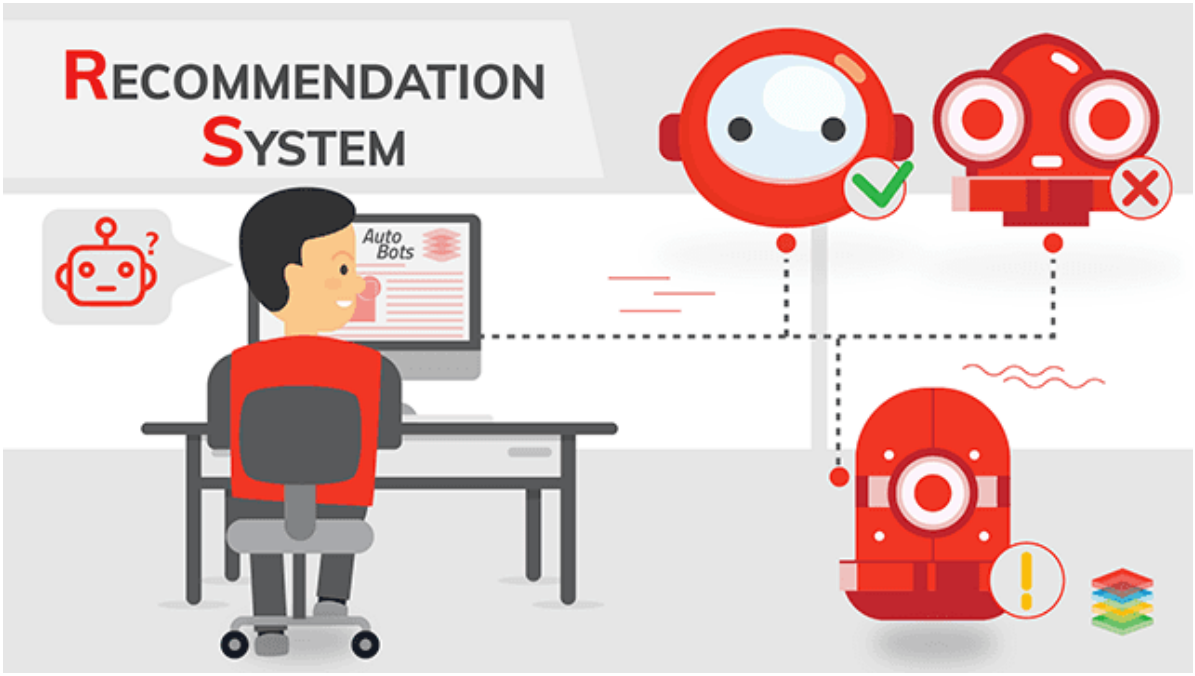
گزارش پروژه نهایی درس هوش محاسباتی

رضا مرادی 9623100

سپهر قمری 9623090

محمد مهدی شجاعی فر 9623065

Recommender system by collaborative filter



زمستان 1399

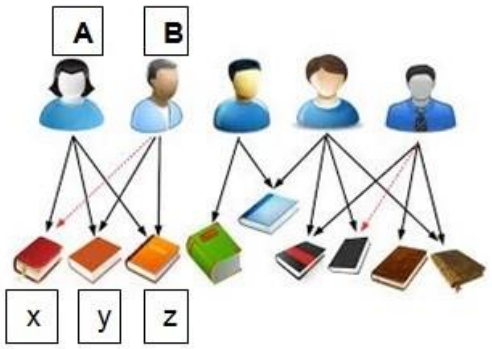
مقدمه

در نرم افزارها و سایت های مختلف اینترنتی که خدماتی را در اختیار کاربران قرار میدهند ، برای یک کاربر نوعی تعداد انتخاب ها بسیار زیاد است و حجم زیادی از اطلاعات هم ممکن است اصلا مورد نیاز و مورد علاقه کاربر مذکور نباشد. بنابراین برای رفع مشکل اضافه بار اطلاعات ، نیاز به فیلتر کردن ، اولویت بندی و ارائه کارآمد اطلاعات یکی از مسالِ بسیار مهم محسوب می گردد. از این رو سایت های معروفی از جمله Netflix و Facebook و ... ، از سیستم های recommender جهت قرار دادن اطلاعات اولویت بندی شده و متناسب با سلیقه و نیاز کاربر استفاده می کنند. یکی از مهم ترین کاربرد های سیستم های recommender ، در سایت های کتابخوانی و جهت ارائه کتاب های متناسب با سلیقه و نیاز کاربر می باشد.

در این پروژه از روش collaborative filtering جهت پیاده سازی سیستم recommender برای یک سامانه کتابخانه استفاده شده است.

Book recommendation system

این برنامه یک نوع سیستم پیشنهاد دهنده ی کتاب است که براساس یک سری دیتا این کار را انجام میدهد دیتا یا اطلاعات خام ورودی در واقع شامل کاربر ها و امتیاز هایی است که هر کاربر به هر کتاب داده است.



Types of Recommendation Systems

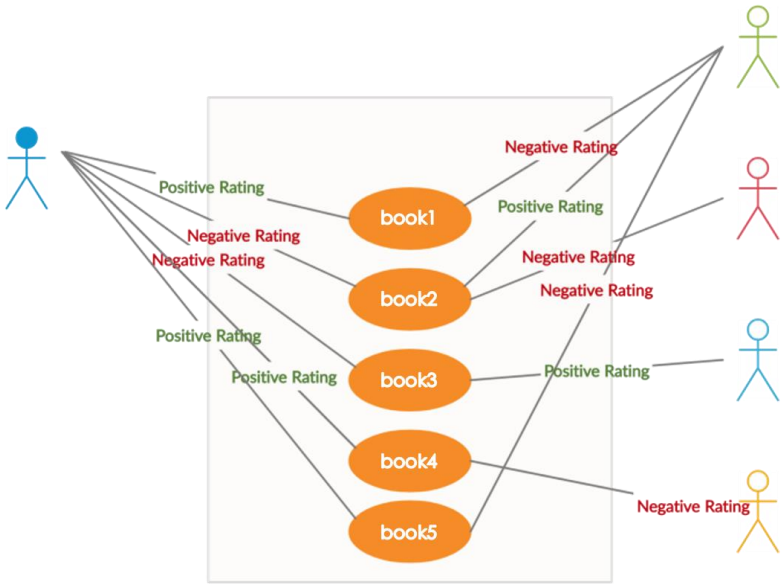
دو نوع سیستم recommendation **وجود دارد:**

- Content filtering recommender systems.
- Collaborative filtering based recommender systems.

Content filtering recommender systems

در این روش فیلترینگ محتوای مورد نظر با استفاده از از اطلاعات جانبی مانند خصوصیات یک آهنگ(نام آهنگ ، نام خواننده ، نام فیلم ، زبان فیلم و...) صورت میگیرد. در این روش حتی اگر موراد جدید هم به کتابخانه اضافه شوند ، سسیتم عملکرد مناسبی خواهد داشت.

Collaborative filtering based recommender systems



ایده ای که در پشت سر این نوع سیستم وجود دارد این است که ابتدا امتیازات کاربر های مختلف را بر روی کتاب های مختلف لحاظ میکند سپس بر اساس امتیازات قبلی که کاربر نسبت به کتاب های مختلف داده و با بررسی امتیازات کاربر های مشابه، بهترین کتاب را پیشنهاد میدهد.

Why Collaborative?

Pros

- **نیازی به دانستن هیچ ویژگی خاصی از کتاب ها نیستیم مانند تاریخی بودن یا رمان یا اجتمایی و ...**
- **اطلاعات نهایی را براساس کاربر های دیگر پیش بینی میکند پس سرعت بالایی دارد.**

Cons

- **در مورد کتاب جدید یا کتابی که هیچ امتیازی به آن داده نشده نمیتوانیم پیشنهاد خاصی بدهیم.**
- **به اطلاعات مجموعه از کاربر ها حداقل نیاز داریم و اگر امتیاز ها پراکنده باشند با مشکل پراکندگی نیز روبرو می شویم.**

بنابراین در این پروژه از روش collaborative **استفاده شده و مشکل کاربرها و کتاب های جدید هم برطرف گردیده است.**

Similarity function

همان طور که گفته شد در این روش کاری که سیستم انجام میدهد ، پیدا کردن میزان شباهت بین دو یا چند المان است . بنابراین لازک است در ابتدا روش مناسب جهت محاسبه میزان شباهت دو المان ارائه شود.

در حالت کلی از دو روش میتوان شباهت بین دو المان را مورد بررسی قرار داد:

- **روش اقلیدسی**

برای تشابه بین دو کاربر a **و** b **داریم:**

$$sim(a,b) = \frac{\sum_{p \in P}(r_{a,p} - \bar{r}_a)(r_{b,p} - \bar{r}_b)}{\sqrt{\sum_{p \in P}(r_{a,p} - \bar{r}_a)^2}\sqrt{\sum_{p \in P}(r_{b,p} - \bar{r}_b)^2}}$$

Where:

a, b = Users

r(a, p)= Rating of user a for item p

P = Set of items. Rated by both users a and b

- **تشابه کسینوسی**

تشابه کسینوسی معیاری است که برای یافتن شباهت بین دو مورد صرف نظر از اندازه آنها استفاده می شود. ما کسینوس یک زاویه را با اندازه گیری بین هر دو بردار در یک فضای چند بعدی محاسبه می کنیم که بر اساس ابعاد قابل استفاده است.

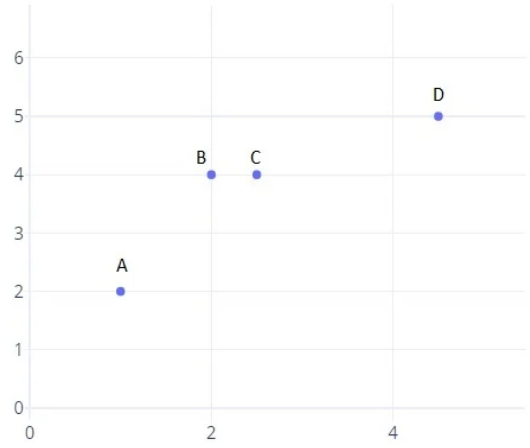
$$\text{Similarity}(p,q) = \cos\theta = \frac{p \cdot q}{\|p\|\|q\|} = \frac{\sum_{i=1}^{n} p_i q_i}{\sqrt{\sum_{i=1}^{n} p_i^{\,2}}\sqrt{\sum_{i=1}^{n} q_i^{\,2}}}$$

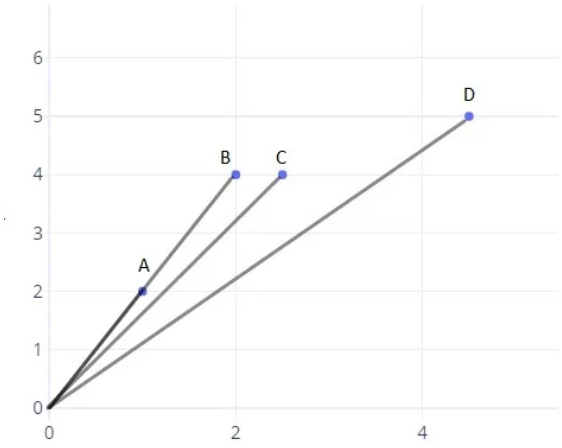
Where:

Cosine is an angle calculated between -1 to 1 where -1 denotes dissimilar items, and 1 shows items which are a correct match.

Cos p. q — gives the dot product between the vectors.

||p|| ||q|| — represents the product of vector's magnitude





همان طور که مشاهده می گردد ، در روش اقلیدسی ، فاصله بین نقاط مورد ارزیابی قرار می گیرد در حالی که در روش کسینوس ، علاوه بر فاصله زاویه نیز مورد بررسی قرار میگرد که میتواند نقایسه جامع تر و درست تری را ارائه دهد. به علاوه در روش کسینوسی ، شباهت در بازه –1 تا 1 ارائه می شود و این مقیاس بندی کار را برای تحلیل و مقایسه بین داده های مختلف بسیار تسهیل میکند.

Different techniques of Collaborative filtering

- User-based nearest neighbor
- Item-based nearest neighbor

User-based Nearest Neighbor

این تکنیک را مثلا برای هومن در نظر میگیریم:

هومن کتاب x را تا به حال نخوانده است در اینجا این الگوریتم به این صورت کار می کند:

- **این تکنیک کاربر هایی را پیدا میکند که در گذشته مانند هومن آیتم های مشابهی را امتیاز داده اند و به کتاب x نیز امتیاز داده اند.**
- **الگوریتم پیش بینی میکند.**
- **برای تمام کتاب هایی که هومن تا به حال نخوانده این تکنیک اجرا میشود.**

Perdition function for User-based Nearest Neighbor

پیش بینی برای آیتم p برای کاربر a به صورت زیر تعریف می شود:

$$pred(a,p) = \overline{r_a} + \frac{\sum_{b \in N} sim(a,b) * (r_{b,p} - \overline{r_b})}{\sum_{b \in N} sim(a,b)}$$

Challenges

برای مقدار قابل توجهی از داده ها ، الگوریتم با عملکرد شدید و مقیاس بزرگ مقابله می کند.

گستردگی محاسباتی O (MN) می تواند در بدترین حالت روبرو شود. جایی که M تعداد کاربر ها و N تعداد کتاب هاست یا آیتم ها است.

با استفاده از روش کاهش ابعاد می توان عملکرد را افزایش داد. با این حال ، کیفیت سیستم را کاهش می دهد.

Item-based Nearest Neighbor

این روش بر اساس شباهت بین کتاب ها یا موارد مختلف ، پیش بینی هایی را ایجاد می کند. به این صورت که به جای پیدا کردن افراد مشابه به هومن ، کتاب هایی مشابه با کتاب هایی که هومن پیش از این مطالعه کرده است را پیشنهاد می دهد.

Perdition function for Item-based Nearest Neighbor

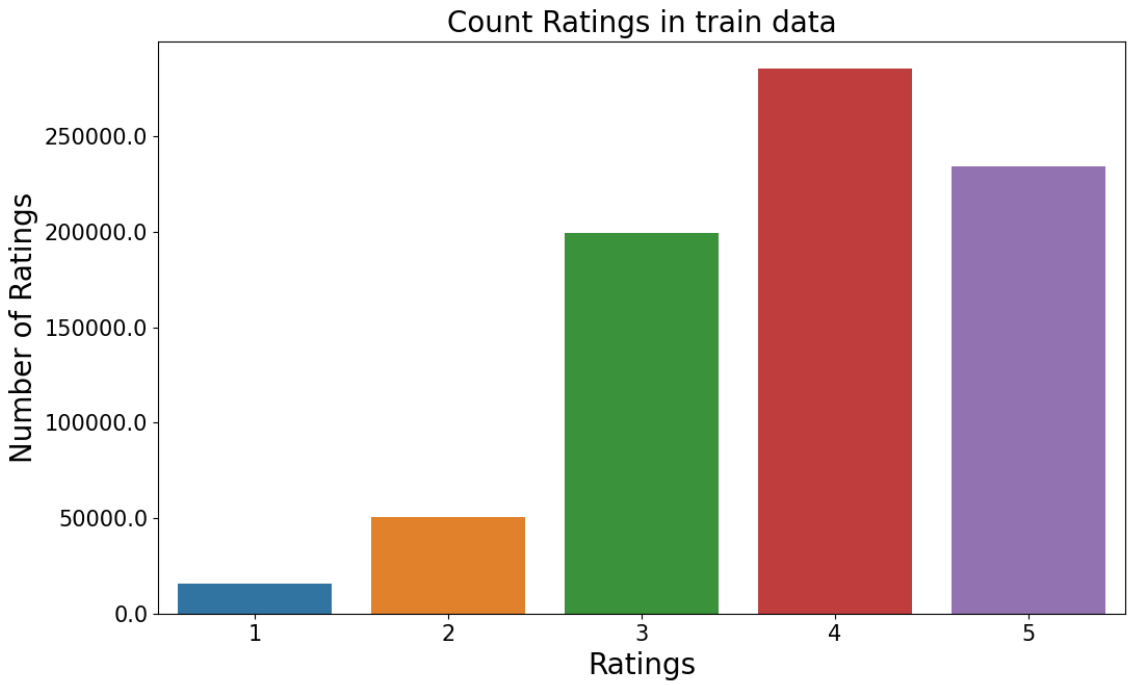
پیش بینی برای کاربر u و آیتم i از مجموع وزنی رتبه بندی های کاربر u برای مواردی که بیشترین شباهت به i را دارند تشکیل شده است.

$$pred(u,i) = \frac{\sum_{j \in ratedItems(u)} sim(i,j) \cdot r_{ui}}{\sum_{j \in ratedItems(u)} sim(i,j)}$$

بررسی سیستم طراحی شده

در ابتدا همه داده ها را از دیتاست میخوانیم و 80 درصد انها را در ترین و مابقی را درتست قرار می دهیم. توجه کنید که هیچ امتیاز صفری در این دیتاست نیست و همه رای ها عدد های صحیحی بین 1 تا 5 می باشد. لازم به ذکر است که تمام کاربر ها امتیاز نداده اند و ممکن است کاربری به هیچ کتابی امتیاز نداده باشد. همجنین ممکن است کتابی تا به حال امتیاز نگرفته باشد.

سپس داده هایمان را پلات میکنیم تا مشخص باشد که از چه تعداد از هر رای 1 تا 5 به کتاب ها داده شده است.



در قسمت بعدی ماتریسی دوبعدی میسازیم که ستون های این ماتریس کتاب ها و ردیف های ان کاربران می باشد. تقاطع هر کاربر و کتاب، امتیازی است که ان کاربر به کتاب داده است. این تابع به کامنت # matrix user/item/rating مشخص شده است. کتاب هایی که تا به حال امتیاز دهی نشده اند، بدیهتا ستون صفر هستد و کاربرانی که تا به حال رای نداده اند ردیف صفر هستند.

در تابع بعدی میانگین امتیاز های داده شده به هرکتاب و هر ردیف اورده شده است. بدیهتا خانه هایی از ماتریس که صفر هستند، یعنی ان کاربر به کتاب رای نداده است نه اینکه ان کاربر به کتاب امتیاز صفر داده است و در این میانگین گیری محسوب نمی شوند. این تابع با کامنت # get average rating for user/book **مشخص شده است.**

در مرحله بعدی تعداد کتاب ها و کاربرانی که در دسته ترین هستند/نیستند مشخص شده است. از انجایی که هرردیف دیتاست ما شامل ایدی یک کتاب و کاربر بود، هیچ نظمی در ان وجود ندارد و ممکن است مثلا تمام امتیازاتی که یک کاربر داده در انجا باشد. این قسمت در کد با کامنت # extra information about our train dataset **مشخص شده است.**

در قسمت بعدی دیتاست کتاب خوانده و ذخیره شده است و با کامنت # read books.csv **مشخص شده است.**

در نهایت نیز تابع پیدا کردن شباهت برای یک کتاب ایجاد شده است. این تابع ایدی یک کتاب را گرفته و درصد شباهت بقیه کتاب ها به ان را برمیگرداند. این قسمت با کامنت # compute similarity of other books with specified book **مشخص شده است.**

حال برای تست این سیستم دو راهکار ما پیشنهاد میدهیم که در ارائه به خاطر کمبود وقت نتوانستیم درست ان را نشان دهیم که این قسمت با کامنت # suggest books to new user **مشخص شده است.**

- **به کاربرهای جدید میتوانیم بدون دادن هیچ اطلاعاتی، کتاب پیشنهاد دهیم. به این صورت که** 10 **کتابی که بیشترین امتیازات را داشته اند را نشان دهیم که خب ایده مناسبی نیست و با کامنت** # first way(bad way) **مشخص شده است.**

- **در روش دوم کاربر ایدی تعدادی از کتاب هایی که دوس دارد را به عنوان ورودی می دهد و سیستم ابتدا نام ان کتاها را نوشته و سپس تعدادی کتاب به آن پیشنهاد میدهد. برای مثال در کلاس کتاب هری پاتر پیشنهاد شد که ایدی ان را پیدا نکردیم. حال ان را پیدا کرده ایم و ان را تست میکنیم. این کتاب در ردیف دوم دیتاست books.csv است و ایدی ان 1 است. این تست در خط 147 تست شده است. لازم به ذکر است از قدرت این الگوریتم است که با یک کتاب هم به درستی کار میکند و نیازمند چند کتب برای تست نیست.**

```
["Harry Potter and the Philosopher's Stone"]
The Time Traveler's Wife 0.47537015629169316
Pride and Prejudice 0.475990652214073
Catching Fire 0.4802992679425578
Cien aÃ±os de soledad 0.4912931766677484
To Kill a Mockingbird 0.5023818876778888
Harry Potter and the Order of the Phoenix 0.5393441575047074
Of Mice and Men  0.5529006115448589
Twilight 0.5592650874370423
Harry Potter and the Prisoner of Azkaban 0.5872842270819664
```

Featuring

با استفاده از روش featuring **تعدادی ویژگی جدید اضافه کردیم برای اینکه بدانیم کاربر** u **به کتاب** b **چه رای ای میدهد. 5 کاربر مشابه به** u **را انتخاب کرده و نمره ی آنها به کتاب** b **جزو ویژگی ها اضافه میکنیم. همچنین 5 کتاب مشابه به** b **را انتخاب کرده و نمره ای که کاربر** u **به آنها داده است به ویژگی ها اضافه میکنیم در آخر میانگین رای های کتاب** b **و میانگین رای های کاربر** u **و میانگین کل امتیاز هارا نیز اضافه میکنیم.**

حال از regression **برای پیش بینی امتیاز ها استفاده کردیم برای پیشنهاد دادن کتاب به این صورت عمل میکنیم:**

تمام کتاب هایی که کاربر u **به آنها امتیاز نداده است را انتخاب کرده و امتیاز آنرا پیش بینی میکنیم سپس کتاب هایی که بیشترین امتیاز را برای آنها پیش بینی کرده ایم به عنوان کتاب های پیشنهادی ارائه میکنم.**

```
2.9748383 3
3.9598267 5
2.9748383 3
3.94919 4
2.9748383 3
3.9598267 5
2.9748383 3
1.9845864 2
3.94919 4
RMSE = 0.4912404165114508
```