گزارش ۶ درس هوش مصنوعی

ارائه یک سیستم توصیه گر برای یک مسئله ی کاربردی

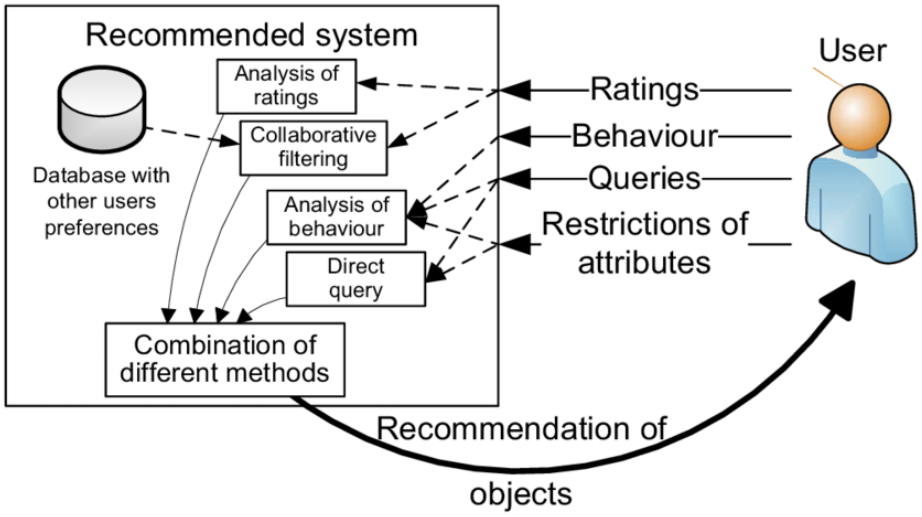
نویسنده:
امیررضا رادجو

استاد
دکتر مهدی قطعی

اردیبهشت ۱۴۰۰

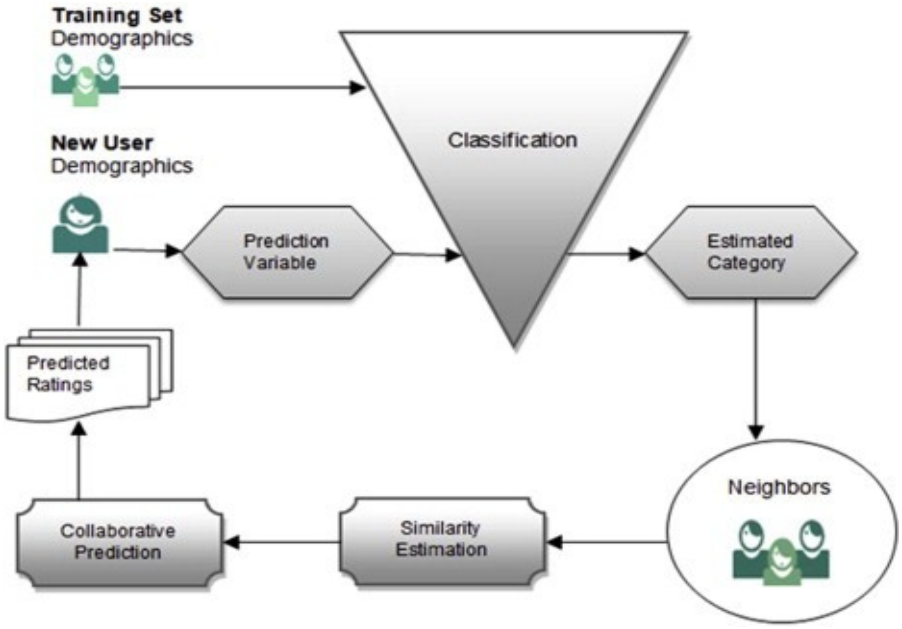
سیستم‌های توصیه گر:

سیستم‌های توصیه گر به این شکل می‌باشند که بر اساس داده‌های موجود(امتیاز ها-رفتارها-سرچ ها و ...) سعی میکنند به کاربر بهترین بیشنهاد ممکن را ارایه دهند. استفاده از سیستم‌های توصیه گر میتواند مزیت‌های بسیار زیادی برای کسب و کارهای مختلف ایجاد کند.
مثلاً فرض کنید به لباس فروشی رفته‌اید و این لباس فروشی با توجه به سلیقه شما تمامی ست های مورد علاقه تان را برایتان لیست میکند و لازم نیست وقت زیادی صرف بیدا کردن لباس و یا ست مورد علاقه خود نمایید. همچنین کیفیت بیشنهادهایی که به شما داده می‌شود به مرور زمان بهتر و بهتر شود. قطعاً خرید از این بوتیک برای شما بسیار راحت‌تر و لذت بخش تر خواهد بود.
سیستم‌های توصیه گر میتواند در همه جا به کار آید. از رستورانها و کافی شاب ها گرفته تا فروشگاههای کتاب‌های کمیک و سایت‌های مخصوص به آهنگ. درواقع در هرچیزی که میتواند به سلیقه شما دخیل باشد و تنوع زیادی داشته باشد سیستم‌های توصیه گر میتواند نقش کلیدی بسیار مهمی را ایفا کند.

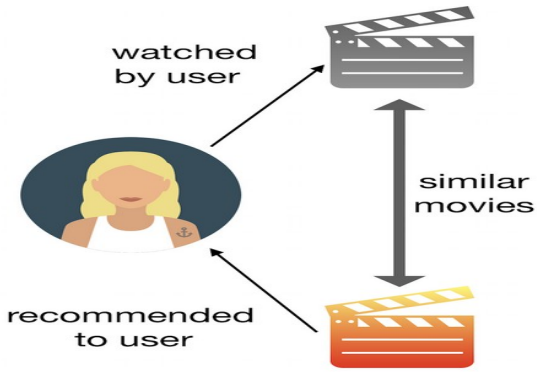


در اینجا قصد داریم تا یک سیستم توصیه گر فیلم طراحی کنیم که بتواند به کاربر فیلم‌هایی که ممکن است دوست داشته باشد را معرفی کند.
در این راستا چندین چالش خواهیم داشت. یک اینکه وقتی که کاربری برای بار اول وارد سیستم میشود چگونه میتوانیم به او فیلم بیشنهاد دهیم(cold start). یکی از ساده‌ترین و در عین حال بهترین کارهایی که میتوان انجام داد استفاده از آمار های قبلی که راجع به دیتاهای فیلم‌ها داشته‌ایم می باشد.
به طور مثال میتوان لیست ۱۰ فیلم برتر از دید کاربران را برای فرد به نمایش گذاشت.
البته وقتی کاربر جدیدی در سیستم ثبت نام می‌کند اطلاعات کمی از او به دست ما می‌رسد که همین اطلاعات نیز میتواند مفید باشد. به طور مثال این فرد ملیت ایرانی دارد بنابرین سایت میتواند لیست فیلم‌هایی که کاربران ایرانی مورد بسندشان است را به این کاربر جدید ارایه دهند و سلیقه مردمی که در استرالیا زندگی میکنند را دخیل نکنند. همچنین ممکن است دیتاهایی که بسیار بی‌اهمیت به نظر برسند بتوانند کمک شایانی به ما در این امر کنند. به طور مثال فرض کنید شخصی که به تازگی در سایت ما ثبت نام کرده است اغلب در ساعت ۳ بعد از ظهر به سایت سر میزند و کلیک های او بیشتر در ژانر کمدی بوده و تنها یک بار از ۲۰۰ بار روی ژانر ترسناک کلیک کرده است. سایت با ذخیره کردن این اطلاعات میتواند تغییراتی را در فیلم بیشنهادی برای آنی کاربر در نظر بگیرد و اغلب فیلم‌هایی به او

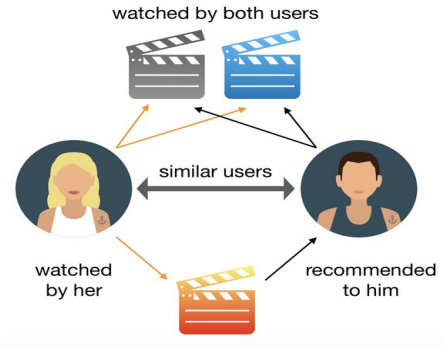
بیشنهاد کند که ترسنتاک نیستند. اما باز بعد از مدتی با رأی دادن کاربر این سیستم میتواند کاملتر شود.



مدتی بعد که کاربر فیلم‌های بیشتری دید و نظرات خود را درباره آن‌ها ابراز کرد سیستم میتواند با توجه به همین تصمیمات برای او تصمیم گیری نماید. مثلاً فرض کنید شخصی به ۵ فیلم امتیاز بالایی داده است که ۳ فیلم از آن‌ها که ساخته کوبریک است و دو فیلم دیگر که امتیاز بالا گرفتند نیز در ژانر جنایی بودند. و همچنین جک نیکلسون در دوتا از این فیلم‌ها بازیگر نقش اول بوده است.
سیستم توصیه گر با توجه به همین اطلاعات میتواند بگوید که این شخص از فیلمی که کارگردان آن کوبریک بوده و ساختار جنایی داشته باشد و همچنین جک نیکلسون در آن ایفای نقش کرده باشد خوشش خواهد آمد. بنابرین در لیست فیلمهایی که به او بیشنهاد می‌دهد فیلم شاینینگ در رتبه نخست قرار خواهد گرفت زیرا تمامی این ویژگی‌ها را داراست.
اما باز این سیتستم بعد از مدتی میتواند کمی کاملتر شود.



وقتی کاربر تعداد قابل قبولی فیلم دیده باشد و همچنین تعداد کاربران موجود در سایت مذکور به اندازه کافی زیاد باشند این امکان برای سایت وجود دارد که با توجه به اطلاعاتی که ذخیره کرده است تشخیص دهد که سلیقه فیلمی کاربر مورد نظر نزدیک به کدام یک از کاربران دیگر سایت می‌باشد و از همین طریق فیلم‌هایی که این کاربر ندیده ولی فرد شبیه به آن در سیستم مشاهده کرده و امتیاز بالایی داده است را به او معرفی کند. سیستم‌های توصیه گر میتوانند این سه روش را با یکدیگر ترکیب کنند و با قرار دادن وزنی برای هر کدام از آن‌ها سیستم توصیه گر خود را روز به روز بهبود دهند.



در ادامه ما تلاش میکنیم تا یک سیستم توصیه گر با هر کدام از روش‌های گفته شده در بالا بنویسیم و نتایج آن‌ها را مشاهده نماییم و ببینیم این نوع سیستم‌ها چگونه عمل میکنند و تا چه حد میتوانند قابل اعتماد باشند.
ابتدا نیاز است که دیتاستی متناسب با این کار بیابیم. متأسفانه نتوانستیم دیتاستی بیابیم که تمامی بارامتر های موردنظر را داشته باشد تا بتوان هر سه نوع سیستم را به طور کامل بیاده سازی کرد. به همین دلیل از دو دیتا ست مختلف استفاده کردیم تا بتوان هر دو نوع سیستم توصیه گر را به طور عملی هرچند به طور ناقص نشان داد. لینک دو دیتاست را در خط زیر مشاهده می‌کنید.

https://www.kaggle.com/tmdb/tmdb-movie-metadata
https://www.kaggle.com/rounakbanik/the-movies-dataset

از دیتاست اول دو نوع اول سیستم‌های توصیه گر را بیاده سازی می نماییم.
این دیتا ست حاوی دو فایل credits و movies می‌باشد که در فایل csv اول برای هر فیلم اطلاعاتی از قبیل بازیگران فیلم و همچنین دست اندرکاران آن فیلم وجود دارد.

	movie_id	title	cast	crew
0	19995	Avatar	[{"cast_id": 242, "character": "Jake Sully", "...	[{"credit_id": "52fe48009251416c750aca23", "de...
1	285	Pirates of the Caribbean: At World's End	[{"cast_id": 4, "character": "Captain Jack Spa...	[{"credit_id": "52fe4232c3a36847f800b579", "de...
2	206647	Spectre	[{"cast_id": 1, "character": "James Bond", "cr...	[{"credit_id": "54805967c3a36829b5002c41", "de...
3	49026	The Dark Knight Rises	[{"cast_id": 2, "character": "Bruce Wayne / Ba...	[{"credit_id": "52fe4781c3a36847f81398c3", "de...
4	49529	John Carter	[{"cast_id": 5, "character": "John Carter", "c...	[{"credit_id": "52fe479ac3a36847f813eaa3", "de...

در فایل دوم اطلاعات دیگری از همان فیلم موجود است. اطلاعاتی از قبیل خلاصه داستان فیلم ژانر امتیاز کاربران تعداد رأی های داده شده بودجه ساخت محبوبیت تاریخ انتشار و ..... قرار دارد.

[8]:

	budget	genres	homepage	id	keywords	original_language	original_title	overview	popularity	production_companies	production_countries	release_date	revenue	runtime	spoken_languages	status	tagline	title	vote_average	vote_coun
0	237000000	[{"id": 28, "name": "Action"}, {"id": 12, "nam...	http://www.avatarmovie.com/	19995	[{"id": 1463, "name": "culture clash"}, {"id":...	en	Avatar	In the 22nd century, a paraplegic Marine is di...	150.437577	[{"name": "Ingenious Film Partners", "id": 289...	[{"iso_3166_1": "US", "name": "United States o...	2009-12-10	2787965087	162.0	[{"iso_639_1": "en", "name": "English"}, {"iso...	Released	Enter the World of Pandora.	Avatar	7.2	1180
1	300000000	[{"id": 12, "name": "Adventure"}, {"id": 14, "...	http://disney.go.com/disneypictures/pirates/	285	[{"id": 270, "name": "ocean"}, {"id": 726, "na...	en	Pirates of the Caribbean: At World's End	Captain Barbossa, long believed to be dead, ha...	139.082615	[{"name": "Walt Disney Pictures", "id": 2}, {"...	[{"iso_3166_1": "US", "name": "United States o...	2007-05-19	961000000	169.0	[{"iso_639_1": "en", "name": "English"}]	Released	At the end of the world, the adventure begins.	Pirates of the Caribbean: At World's End	6.9	450
2	245000000	[{"id": 28, "name": "Action"}, {"id": 12, "nam...	http://www.sonypictures.com/movies/spectre/	206647	[{"id": 470, "name": "spy"}, {"id": 818, "name...	en	Spectre	A cryptic message from Bond's past sends him o...	107.376788	[{"name": "Columbia Pictures", "id": 5}, {"nam...	[{"iso_3166_1": "GB", "name": "United Kingdom"...	2015-10-26	880674609	148.0	[{"iso_639_1": "fr", "name": "Français"}, {"is...	Released	A Plan No One Escapes	Spectre	6.3	446
3	250000000	[{"id": 28, "name": "Action"}, {"id": 80, "nam...	http://www.thedarkknightrises.com/	49026	[{"id": 849, "name": "dc comics"}, {"id": 853,...	en	The Dark Knight Rises	Following the death of District Attorney Harve...	112.312950	[{"name": "Legendary Pictures", "id": 923}, {"...	[{"iso_3166_1": "US", "name": "United States o...	2012-07-16	1084939099	165.0	[{"iso_639_1": "en", "name": "English"}]	Released	The Legend Ends	The Dark Knight Rises	7.6	910
4	260000000	[{"id": 28, "name": "Action"}, {"id": 12, "nam...	http://movies.disney.com/john-carter	49529	[{"id": 818, "name": "based on novel"}, {"id":...	en	John Carter	John Carter is a war-weary, former military ca...	43.926995	[{"name": "Walt Disney Pictures", "id": 2}]	[{"iso_3166_1": "US", "name": "United States o...	2012-03-07	284139100	132.0	[{"iso_639_1": "en", "name": "English"}]	Released	Lost in our world, found in another.	John Carter	6.1	212

. متأسفانه به دلیل در اختیار نداشتن اطلاعات افراد رأی دهنده(منطقه سکونت و...) امکان انجام عملیات های گفته شده وجود ندارد و برای قسمت اول تنها میتوانیم مقدار محبوبیت فیلم‌ها را در نظر بگیریم. حتی در این حالت نیز به مشکلاتی بر خواهیم خورد. به طور مثال فرض کنید از اکران یک فیلم مدت زمان بسیار کمی گذشته باشد و فقط یک نفر به آن رأی داده باشد و رأی او هم امتیاز کامل به فیلم بوده باشد. یقیناً نمیتوان این فیلم را برتر از فیلم‌هایی که میلیون‌ها نفر به آن رأی داده‌اند و ۹۰ درصد امتیاز کامل را گرفته است دانست. بنابرین برای بهبود این وضعیت و ایجاد حالتی منطقی‌تر از امتیازبندی که imdb استفاده میکند کمک میگیریم. این امتیازبندی به این شکل است که تعداد افراد رأی دهنده را نیز در رتبه بندی فیلم‌ها دخیل میکند.
فرمولی که توسط این سایت استفاده می‌شود به شکل زیر است:

$$WeightedRating(\mathbf{WR}) = \left(\frac{\mathbf{v}}{\mathbf{v}+\mathbf{m}} \cdot \mathbf{R}\right) + \left(\frac{\mathbf{m}}{\mathbf{v}+\mathbf{m}} \cdot \mathbf{C}\right)$$

با استفاده از این فرمول میتوان امتیازبندی منصفانه تری را انجام داد.
V نشان دهنده تعداد رأی های داده شده به کتاب می‌باشد و m مینیمم رأی هایی است که باید یک کتاب بیاورد تا بتواند در لیست قرار بگیرد.
R میانگین رأی های داده شده به آن فیلم و همچنین C میانگین رأی های همه فیلم‌های موجود در دیتاست خواهد بود.
این نوع محاسبه به ما کمک خواهد کرد که با دادن وزن به امتیاز ها یک تعادل معقول بین امتیاز و تعداد رأی دهندگان ایجاد نماییم.
حال با استفاده از امتیاز های داده شده داده‌های خود را مرتب میکنیم و فیلمی را به کاربران بیشنهاد میدهیم که امتیازی که با این فرمول به دست آورده باشد مقدار ماکسیمم خود باشد.
این روش صددرصد روش کامل و قابل اعتمادی نیست اما میتوان گفت که اکثر سلیقه ها را در بر میگیرد بنابرین برای فردی که به تازگی وارد سایت شده و ما اطلاعاتی از وی نداریم میتواند راه حل مناسبی باشد. هرچند داشتن اطلاعاتی از قبیل موقعیت مکانی و ... میتوانست کمک بیشتری به سیستم توصیه گر ما کند.

Content Based:

حال فرض کنید این کاربر یک فیلم مشاهده میکند و از آن خوشش میاید. سیستم توصیه گر ما باید بتواند با استفاده از همین داده کوچک فیلمی به کاربر بیشنهاد بدهد.
برای این کار ابتدا کمی داده‌های خود را بررسی میکنیم. ما لیست بازیگران- دست اندرکاران- خلاصه فیلم-ژانر و... را در اختیار داریم. حال کمی فکر کنید که به چه شکل میتوان با استفاده از این اطلاعات فیلمی به کاربر معرفی کرد.

همانطور که در بالاتر گفته شد برای این کار میتوان این اطلاعات را در نظر گرفت و فیلم‌هایی شبیه به این اطلاعات را استخراج نمود. مثلاً اگر کاربر فیلمی راجع به زندگی مافیایی دیده است همه فیلم‌های مافیایی در امتیاز بندی برای انتخاب فیلم بعدی امتیاز خواهند گرفت. یا اگر بازگر اصلی فیلم اول مارگوت رابی بوده باشد همه فیلم‌هایی که مارگوت رابی در آن‌ها ایفای نقش کرده‌اند امتیاز خواهند گرفت. نحوه توزیع این امتیازها و همچنین وزنی که به آن‌ها داده شود دلخواه می باشد و میتوان بر اساس معیار های مختلف آن‌ها را مشخص کرد. ما در این بیاده سازی اولویت اول را با کارگردان فیلم قرار دادیم. سپس بازیگران و بعد از آن کلمات کلیدی که در خلاصه فیلم آمده است.

روند این کار به این صورت می‌باشد که که رشته ای از کلمات کلیدی و نام کارگردان و بازیگران اصلی میسازیم و با بررسی شباهت‌ها بین رشته‌های ایجاد شده برای هر فیلم فیلم مشابه را به کاربر برمیگردانیم. در اینجا چون ارزش کارگردان و بازیگران برای ما بیشتر بوده است به ترتیب ۳ و ۲ بار آن‌ها را در رشته تکرار میکنیم.
حال نوبت به بررسی رشته‌ها و امتیازدهی بین آن‌ها می‌باشد. برای این کار چندین روش موجود است که طبق تست های انجام شده بهترین روش شباهت کسینوسی و بعد از آن فاصله اقلیدسی می‌باشد که هر دو داخل کد قرار داده شده‌اند. اما به دلیل کارکرد بهتر تشابه کسینوسی از آن استفاده کرده‌ایم.
در صورت تمایل میتوانید با عوض کردن جای آن‌ها تغییرات در صورت استفاده از روش فاصله اقلیدسی را نیز مشاهده نمایید.

کدهای مربوط به این قسمت:

```
def get_director(x):
    for i in x:
        if i['job'] == 'Director':
            return i['name']
    return np.nan
```

```
def get_list(x):
    if isinstance(x, list):
        names = [i['name'] for i in x]
        if len(names) > 3:
            names = names[:3]
        return names

    return []
```

```
def clean_data(x):
    if isinstance(x, list):
        return [str.lower(i.replace(" ", "")) for i in x]
    else:
        if isinstance(x, str):
            return str.lower(x.replace(" ", ""))
        else:
            return ''
```

```
def create_soup(x):
    return ' '.join(x['keywords']) + ' ' + ' '.join(x['cast']) + ' ' + x['director'] + ' ' +
' '.join(x['genres']) + ' ' + x['director'] + ' ' + x['director'] + ' ' + ' '.join(x['cast'])
```

تابع get_director: استخراج اسم کارگردان از لیست دست اندرکاران
تابع get_list: دریافت اسم سه بازیگر اول از لیست بازیگران
تابع clean data: مربوط به تمیز کردن و مرتب کردن داده‌ها شامل حذف کردن فاصله بین اسم ها(در صورتی که فاصله حذف نشود ممکن است دو شخص متفاوت به دلیل داشتن first name یکسان دارای تشابه تشخیص داده شوند و یک تشابه به غلط در نظر گرفته شود) همچنین همه حروف را به حالت کوچک در میاورد تا تشابهی در این بین به دلیل تفاوت در حرف بزرگ و کوچک از بین نرود.
تابع create soup: این تابع یک برای هر فیلم با توجه به مقادیری که دریافت کرده است یک رشته میسازد که با توجه به این رشته مقدار شباهت بین فیلم‌های مختلف را تشخیص دهد. در این قسمت است که ما ۳ بار کارگردان و ۲ بار اسم بازیگران را وارد میکنیم تا با درجه اهمیت بیشتری وارد مقایسه‌ها شوند.

```
tfidf = TfidfVectorizer(stop_words='english')
df_movies['overview'] = df_movies['overview'].fillna('')
tfidf_matrix = tfidf.fit_transform(df_movies['overview'])
```

```
cosine_sim = linear_kernel(tfidf_matrix, tfidf_matrix)
euclidean = -1 * euclidean_distances(tfidf_matrix, tfidf_matrix)
```

```
indices = pd.Series(df_movies.index, index=df_movies['title']).drop_duplicates()
```

سلول اول: استخراج کلمات کلیدی هر فیلم با در نظر گرفتن حروف برتکرار در زبان انگلیسی و قرار دادن آن‌ها در یک ستون جدا به اسم overview

سلول دوم: دو روش متفاوت برای مقایسه ماتریس های کلمات که روش اول روش تشابه کسینوسی و روش دوم فاصله اقلیدسی میباشد که ما در این سیستم از تشابه کسینوسی استفاده کرده‌ایم. فرمول های مربوط به آن‌ها را میتوانید در زیر مشاهده کنید.

$$\text{similarity} = \cos(\theta) = \frac{\mathbf{A} \cdot \mathbf{B}}{\|\mathbf{A}\|\|\mathbf{B}\|} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2}\sqrt{\sum_{i=1}^{n} B_i^2}},$$

$$E(x, y) = \sqrt{\sum_{i=0}^{n} (x_i - y_i)^2}$$

Collaborative:

بعد از اینکه کاربر مدتی در سیستم فعالیت داشت میتوان سیستم توصیه گر را به این حالت ارتقا داد. این حالت به اینگونه عمل می‌کند که سعی میکند با توجه به بیدا کردن سلیقه مشترک بین شما و بقیه کاربران مقدار امتیازی که به فیلم‌های مختلف خواهید داد را حدس بزند. و به این ترتیب میتواند سلیقه های مشترک با شما را یافته و فیلمی که برای شما مناسب‌تر است(این بار بر اساس سلیقه افرادی که به نظر می‌رسد با شما سلیقه یکسانی دارند) انتخاب کند. این نوع سیستم دارای مزیت‌ها و معایب مختص به خودش می‌باشد.
اشکال اصلی در این نوع سیستم توصیه گر وجود داده به اندازه کافی می‌باشد. برای مثال اگر شما شروع به ساختن یک سیستم توصیه گر برای سایت جدیدتان کنید یقیناً در ابتدا نمیتوانید این نوع سیستم را با استفاده از دیتای کاربران خودتان بیاده سازی کنید و نیازمند دیتا و زمان بیشتری برای جمع آوری دیتای مورد نیاز خواهید بود اما در صورتی که شما دیتای کافی داشته باشید این روش میتواند بهترین روش برای شما باشد.
به طور مثال در سیستم توصیه گر قبلی کسی که فیلم سینمایی green mile را مشاهده کرده است احتمالاً علاقه ندارد تا بعد از آن فیلمی مانند monsters ball که تقریباً شباهت‌های بسیار زیادی در خلاصه داستان بین آن‌ها وجود دارد ببیند. بلکه چیزی که برای این شخص جذاب است مفهوم و حسی است که از فیلم میگیرد. شاید انتخاب فیلم از روی کارگردان بتواند این قضیه را بهبود دهد ولی یقیناً به اندازه کافی مناسب نخواهد بود. اما وقتی از رأی افراد شبیه به خودتان استفاده میکنید کسی که رأی داده دارای عواطف انسانی می‌باشد و چیزهایی را در فیلم تشخیص میدهد که توسط کامبیوتر قابل تشخیص نیست و یا اینکه در دیتاست موجود نیست. به همین دلیل وقتی دیتاست غنی در اختیار داشته باشیم این روش می‌تواند یک روش عالی باشد.
البته میتوان ترکیبی از این روش‌ها را نیز استفاده کرد که بسته به سلیقه فردی که سیستم توصیه گر را انتخاب میکند می باشد.
یک استراتژی جالب برای این کار آن است که به ترتیب به سه استراتژی ضریب های a b c بدهیم. در ابتدا که دیتایی از کاربر نداریم a=1 b=0 c=0 قرار دهیم و هرچه اطلاعات و دیتای بیشتری از کاربر به دست می آوریم ضریب ها را تغییر دهیم به گونه‌ای که رفته‌رفته a کم شود و مقدار دو متغیر دیگر اضافه شود تا بتوان سیستم توصیه گر مناسب تری را به کاربران ارایه داد

دیتاست این قسمت:

این قسمت حاوی یک دیتا ست جداگانه برای خود می‌باشد که در شکل زیر فرم کلی آن را مشاهده میکنید. این دیتاست حاوی آیدی کاربران آیدی فیلم امتیاز و زمان رأی می‌باشد ما برای استفاده از این دیتا ست در حالت ساده خود به سه ستون اول نیاز داریم.
ساختار دیتاست را در عکس زیر مشاهده می‌کنید.

	userId	movieId	rating	timestamp
0	1	31	2.5	1260759144
1	1	1029	3.0	1260759179
2	1	1061	3.0	1260759182
3	1	1129	2.0	1260759185
4	1	1172	4.0	1260759205

توضیحات کد مربوط به این قسمت:

[32]:
```
data = Dataset.load_from_df(ratings[['userId', 'movieId', 'rating']], reader)
```

[33]:
```
svd = SVD()
cross_validate(svd, data, measures=['RMSE', 'MAE'], cv=5)
```

[33]:
```
{'test_rmse': array([0.89436598, 0.89260879, 0.89748886, 0.8907387 , 0.9100275 ]),
 'test_mae': array([0.68891284, 0.68823508, 0.69215914, 0.68859321, 0.69862209]),
 'fit_time': (5.27518105506897,
  5.764752626419067,
  6.771573781967163,
  5.065270662307739,
  5.307448387145996),
 'test_time': (0.22551250457763672,
  0.19232559204101562,
  0.19416284561157227,
  0.16683316230773926,
  0.19488978385925293)}
```

[34]:
```
svd.fit(data.build_full_trainset())
```

[34]:
```
<surprise.prediction_algorithms.matrix_factorization.SVD at 0x7fd95d9986a0>
```

[36]:
```
list_of_movies=[]
for i in range(0,10000):
    tt = svd.predict(1, i, 500)
    if(tt[3]>3.3):
        if not q_movies['title'][q_movies['id']==tt[1]].empty:
            print(q_movies['title'][q_movies['id']==tt[1]])
```

در این قسمت با استفاده از الگوریتم singular value decomposition برای ماتریس رأی های کاربران می‌باشد که توضیحات کامل آن را میتوانید در وبلاگ فرادرس مشاهده کنید. به طور خلاصه با استفاده از آن این ماتریس را به سه ماتریس دیگر تجزیه میکنیم. و با کمک آن مقادیر دیگر را حدس زده و برای هر کاربر با توجه به رأی دیگران بهترین فیلم موجود را انتخاب میکنیم.

منابع:

۱- https://www.kaggle.com/gspmoreira/recommender-systems-in-python-101

۲- https://www.kaggle.com/rounakbanik/movie-recommender-systems

۳- https://www.kaggle.com/ibtesama/getting-started-with-a-movie-recommendation-system