دانشگاه صنعتی امیرکبیر
دانشکده ریاضی و علوم کامپیوتر

پروژه درس هوش مصنوعی

پیشنهاد محصول فروشگاه آنلاین

اشکان ودادی گرگری، زهرا کشاورز رضایی، بهناز محمدی

۹۷۱۲۰۴۵ ،۹۷۱۲۰۳۹ ،۹۷۱۳۰۳۲

استاد درس: دکتر محمد اکبری

فروردین ۱۴۰۱

چکیده

یکی از چالش‌های امروزه‌ی فروشگاه‌های آنلاین دنیا، پیشنهاد محصول به خریداران خودشان است. در سال‌های اخیر، راهکارها و الگوریتم‌های متعددی در راستای تسهیل کردن این مورد ارائه شده است. ما قصد داریم یک روش برای حل این مسئله با الگوریتم‌های موجود ارائه دهیم و در نهایت آن را پیاده‌سازی کنیم. روشی که ما برای حل این مسئله انتخاب کردیم، سیستم‌های توصیه‌گر(Recommender System) با الگوریتم فیلترینگ مشارکتی (Collaborative Filtering) است.

مقدمه

برای پیشنهاد محصول بر اساس امتیازات خریداران، روش‌های معتددی وجود دارد که ما از میان این روش‌ها باتوجه به مقایسه روش‌های گوناگون، از سیستم‌های توصیه‌گر استفاده می‌کنیم. سیستم‌های توصیه‌گر ابزارهایی هستند که به کاربر کمک می‌کنند تا گزینه مورد نظر خود را راحت‌تر پیدا کنند (در این مسئله گزینه‌های ما همان محصولات ما هستند) و پیشنهادهایی به کاربر می‌دهد که احتمالا مورد علاقه کاربر هستند.

۱. تعریف الگوریتم

سیستم‌های توصیه‌گر ابزارهایی هستند که به کاربر کمک می‌کنند تا گزینه مورد نظر خود را راحت‌تر پیدا کنند (در این مسئله گزینه‌های ما همان محصولات ما هستند) و پیشنهادهایی به کاربر می‌دهد که احتمالا مورد علاقه کاربر هستند.

۱.۱. انواع سامانه های توصیه گر

به طور کلی سیستم‌های توصیه‌گر بر اساس عملکردشان به سه دسته کلی تقسیم می‌شوند:

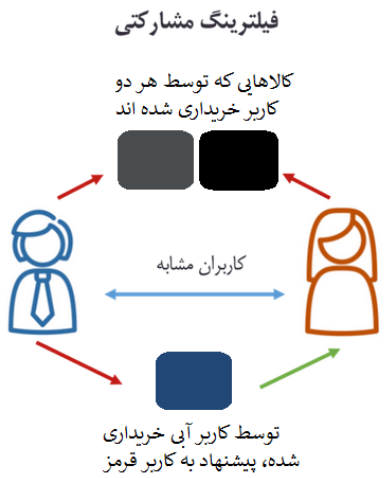
1. فیلترینگ محتوا محور (content-based filtering)
2. فیلترینگ مشارکتی(collaborative filtering)
3. مدل ترکیبی(hybrid)

ما برای حل این مسئله از فیلترینگ مشارکتی استفاده می‌کنیم.

۲.۱ فیلترینگ مشرکتی

فیلترنگ مشارکتی، تشابه بین کاربر و گزینه را به طور همزمان در نظر می‌گیرد. مدل های فیلترینگ مشارکتی می‌توانند گزینه ای به کاربر الف پیشنهاد دهد که بر اساس تشابه های بین کاربر الف و کاربر ب است.

برای مثال کاربر الف وارد سایت خرید آنلاین می‌شود، مدل فیلترینگ مشارکتی باید کالاهای مشابه کالاهایی که کاربر الف در گذشته نگاه کرده است را پیشنهاد دهد و همچنین کالاهایی که کاربران، مشابه کاربر الف خریداری کرده است.



۳.۱. پیدا کردن تشابه بین کاربران و محصولات

فرض کنیم امتیازهایی که هر کاربر به هر کالا داده است را جمع‌آوری کرده و ماتریس کاربران و کالاها را ایجاد کرده ایم (جدول زیر). در کل چهار کاربر داریم و هفت کالا و هر کاربر امتیازی بین صفر تا ۵ به هر کالا که خریداری کرده است، داده است و درایه‌هایی که امتیاز ندارند، توسط کاربر خریداری نشده اند.

	خنک کننده کامپیوتر	کیس کامپیوتر	پرینتر	مانیتور	کارت حافظه	هارد اکسترنال	فلش مموری
الف	۴			۵	۱		
ب	۵	۵	۴				
ج				۲	۵	۴	
د		۳					۳

با توجه به ماتریسی که می‌بینیم کاربر الف، به کالاهایی که مرتبط به تجهیزات جانبی کامپیوتر است امتیاز بیشتری داده‌است و به کالاهایی که مرتبط به تجهیزات ذخیره سازی است امتیاز کمی داده‌است که می‌توان تشابهی بین کیس کامپیوتر و خنک کننده کامپیوتر و پرینتر و مانیتور دید و تضادی بین آن‌ها و کارت حافظه، هارد اکسترنال و فلش مموری است. کاربر ب نیز مانند کاربر الف به لوازم جانبی کامپیوتر امتیاز بالایی داده است.

حال برای پیشنهاد دادن کالاها به روش فیلترینگ مشارکتی، باید بتوان روشی دقیق برای پیدا کردن تشابه بین کاربران و کالاها پیدا کرد که در مدل‌های یادگیری ماشین نیز قابل استفاده باشد. یکی از مرسوم ترین روش‌های پیدا کردن تشابه، مشابهت کسینوسی (Cosine similarity) است.

برای پیدا کردن تشابه بین دو کاربر، سطر هر کاربر را به عنوان یک بردار در نظر می‌گیریم و امتیازهایی که وجود ندارند را به ناچار، صفر قرار می‌دهیم. برای مثال، بردار کاربر الف به شکل زیر است:

الف	۴	۰	۰	۵	۱	۰	۰

تابع تشابه در شکل زیر قابل مشاهده است که حاصل تقسیم ضرب داخلی بردارها بر ضرب اندازه بردارها است:

$$Similarity(A,B) = \cos(A,B) = \frac{A \cdot B}{\|A\| \times \|B\|}$$

حال اگر بخواهیم تشابه بین کاربران الف، ب و ج را با استفاده از تابع مشابهت کسینوسی استفاده کنیم، نتایج زیر بدست می‌آیند:

$$Similarity(A,B) = 0.38 > Similarity(A,C) = 0.32$$

کاربران الف و ب کالاهایی که مرتبط با تجهیزات جانبی کامپیوتر هستند را دوست داشتند و کاربر ج کالاهایی که مرتبط با تجهیزات ذخیره سازی ، و تشابه بین کاربران الف و ب بیشتر است.

۴.۱ چالش‌های فیلترینگ مشارکتی

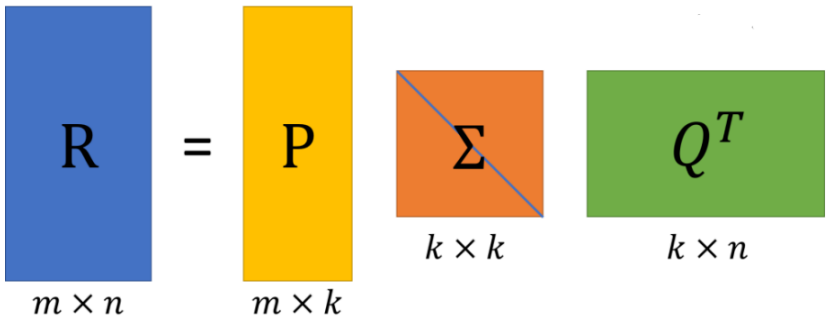
مشکل اصلی فیترینگ مشارکتی، هزینه محاسباتی آن است. برای مثال در مساله ما ۴۶۵۵۴ کاربر داریم و ۳۴۴۶ کالا. اگر بخواهیم شباهت بین آن‌ها را بیابیم. اندازه ماتریس M*N است. (تعداد کاربران و کالاها) پیچیدگی در بدترین حالت به O(M*N) می‌رسد.

هر چند این چالش تنها در سامانه‌های توصیه‌گر وجود ندارد، قبل‌تر از سامانه‌های توصیه‌گر ، این مشکل در جامعه بازیابی اطلاعات (Information Retrieval) نیز مطرح بود که برای حل این مشکل از روشی به نام SVD (Singular value decomposition) استفاده شد. به طور کلی، هدف SVD این است که فضای ماتریس را به فضای کوچک‌تر و فشرده‌تری تجزیه کند.

۱.۴.۱ SVD

SVD روشی برای تجزیه ماتریس در جبرخطی است که می‌توان ماتریس را به اجزای سازنده آن تجزیه کرد. از همان ایده در تجزیه ماتریس استفاده می‌کنیم. فرمول SVD در شکل زیر آمده است که ماتریس R به چند ماتریس دیگر تجزیه شده است که حاصلضرب آن ماتریس‌ها همان ماتریس R است.

$$R = P\Sigma Q^T$$



که در مثال ما :

- R همان ماتریس امتیازها است که هر کاربر به کالاها داده است.
- P ماتریس ویژگی‌های کاربر نام دارد. هر سطر از P، یک بردار است که نشان‌دهنده‌ی علایق یک کاربر است.
- Q ماتریس ویژگی‌های گزینه (همان کالاها) نام دارد، که هر سطر ازQ، ویژگی‌های مرتبط به یک کالا را در خود دارد.

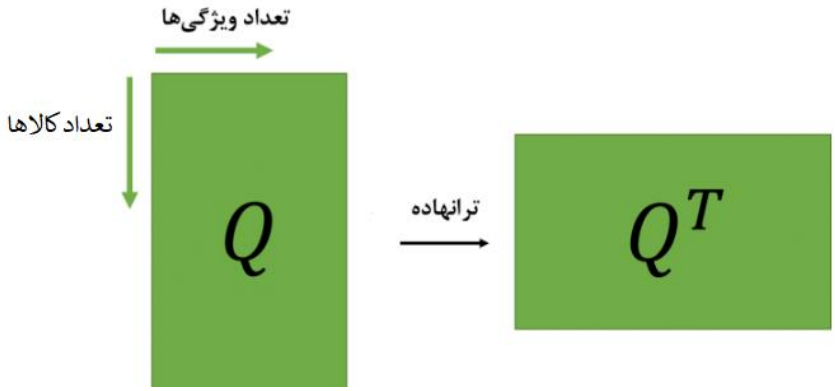
- ماتریس قطری است که داریه‌های آن به جز قطرش، صفر هستند. این ماتریس نشان‌دهنده‌ی وزن مرتبط به ویژگی‌هاست.

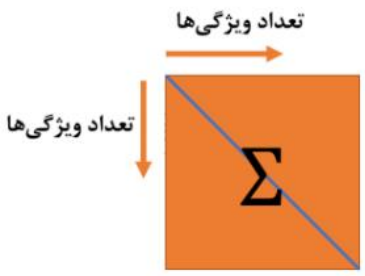
با این تجزیه، ویژگی‌هایی برای کاربر و کالاها بدست می‌آید که لزوما قابل توصیف برای انسان نیست. برای مثال ما به کالاها ویژگی‌هایی مانند اینکه جزو کدام دسته از لوازم الکترونیکی هست(جزو لوازم جانبی و یا تجهیزات ذخیره سازی) نسبت می‌دهیم، اما ویژگی‌هایی که در ماتریس Q برای کالاها بدست آوردیم ممکن است قابل توصیف نباشد. هدف اصلی از انجام این تجزیه، کاهش ابعاد ماتریس اولیه R است که با ضرب هر سطر از P وQ (که نشان دهنده ویژگی‌های یک کاربر و یک کالا است) تا بتوانیم پیش‌بینی کنیم که آن کاربر چه امتیازی به آن کالا خواهد داد.

۲.۴.۱. نگاه دقیق‌تر به ماتریس‌هایSVD

یکی از ماتریس‌های بدست آمده از تجزیه، P است که ماتریس ویژگی‌های کاربران است (شکل روبه‌رو). این ماتریس برخلاف ماتریسR ، یک ماتریس کامل است و تمام داریه‌های آن مقدار دارد. هر سطر آن نشان‌دهنده‌ی یک کاربر است و هر ستون آن یک ویژگی منتسب به کاربران است. هرچند که این ویژگی‌ها لزوما قابل توصیف نیست، اما فرض کنیم یک ستون نمایان‌گر سن کاربر باشد. بنابراین برای کاربران جوان این ستون مقدار کمتری نسبت به کاربران کهن‌سال دارد.

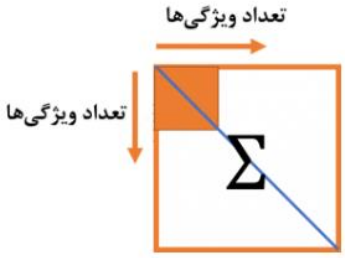
در شکل زیر نیز ماتریس Q نمایش داده شده است که ماتریس ویژگی‌های منتسب به گزینه‌ها (کالاها) است. این ماتریس نیز کامل است و در ضرب، ترانهاده آن قرار می گیرد. هر سطر آن بردار ویژگی یک کالا است. در اینجا نیز برای مثال ممکن است یک ستون نشان‌دهنده‌ی جزء لوازم الکتریک بودن برای یک کالا باشد و برای کالایی مانند میز که جزء لوازم الکتریکی نیست، مقدار این ویژگی آن کم خواهد بود.





اما سیگما، ماتریس قطری وزن‌ها است که تنها قطر این ماتریس مقدار دارد و بقیه مقادیر آن صفر است. هر کدام از این وزن‌ها منتسب به یک ویژگی است و مقدار وزن آن نشان‌دهنده‌ی میزان اهمیت آن ویژگی برای پیش‌بینی علایق کاربر است.

۳.۴.۱. کاهش ابعاد



یکی از اهداف اصلی که از SVD استفاده می‌کنیم، کاهش ابعاد ماتریس رتبه‌ها (R) است و سعی داریم با کمترین میزان داده، بتوانیم ماتریس اولیه را تخمین بزنیم و نمایش دهیم ولی SVD به شکل اولیه، تنها ماتریس را به ماتریس‌های دیگری تجزیه می‌کند و برای کاهش ابعاد قدمی دیگری نیاز است. از آن جا که وزن‌های ماتریس سیگما نشان‌دهنده‌ی اهمیت هر ویژگی است، می‌توانیم فقط بزرگ‌ترین وزن‌ها را نگه داریم و بقیه را صفر در نظر بگیریم. با صفر کردن وزن‌های کم اهمیت، ویژگی‌های مرتبط با آن‌ها نیز در ماتریس P و Q نیز صفر خواهد شد که باعث می‌شود ماتریس‌های بسیار کوچک‌تری داشته باشیم و از طرفی تنها مهم‌ترین ویژگی‌ها را برای کاربران و محصولات درنظر بگیریم.

۲. دادگان (dataset)

این مخزن داده شامل چندین دادگان است. برای این مسئله ما از مجموعه وسایل الکترونیکی(electronics) و فقط امتیاز دهی شده‌اند را استفاده می‌کنیم. این دادگان فقط شامل نام کاربرها، کالاها، امتیازدهی و زمان ثبت امتیازاست. هر سطر متعلق به یک کاربر و یک کالا است که امتیاز کاربر و رمان خرید را به ما نشان می‌هد.

به عنوان مثال :

	userId	productId	ratings	timestamp
0	AKM1MP6P0OYPR	0132793040	5.0	1365811200
1	A2CX7LUOHB2NDG	0321732944	5.0	1341100800
2	A2NWSAGRHCP8N5	0439886341	1.0	1367193600
3	A2WNBOD3WNDNKT	0439886341	3.0	1374451200
4	A1GI0U4ZRJA8WN	0439886341	1.0	1334707200

۳. پیاده‌سازی:

برای بخش پیاده سازی الگوریتم ۶ گام مهم در آن موجود است که به ترتیب هر یک از موارد را مورد بررسی قرار می‌دهیم. شما می‌توانید کد زده شده در پروژه را با استفاده از لینک زیر باز کنید.

AI-finalpoject.ipynb - Colaboratory (google.com)

۱.۳ بخش اول – بارگزاری کتابخانه‌های مورد نیاز:

باتوجه به مسئله که باید با داده‌های فراوانی در طی پیاده‌سازی کار کنیم، کتابخانه‌های مربوط به خواندن داده، کار با داده و ... را اضافه کردیم. همچنین یک سری کتابخانه‌ها توسط توضیحات داده‌گانی که استفاده کردیم نیز استفاده کردیم.

```
#import the reqired libraries
import numpy as np
import pandas as pd
import json
from sklearn.model_selection import train_test_split
from scipy.sparse.linalg import svds
```

- کتابخانه numpy:

برای انجام محاسبات گوناگون ریاضی، از کتابخانه numpy استفاده می‌کنیم. NumPy یک کتابخانه برای زبان برنامه نویسی پایتون (Python) است. با استفاده از این کتابخانه امکان استفاده از آرایه‌ها و ماتریس‌های بزرگ چند بعدی فراهم می‌شود.

- کتابخانه pandas:

یک کتابخانه متن‌باز (Open Source) که کارایی بالا، ساختاری با قابلیت استفاده آسان و ابزارهای تحلیل داده برای برنامه نویسی پایتون را فراهم می‌کند. پانداس یک کتابخانه قدرتمند برای تحلیل و پردازش داده‌ها است.

- کتابخانه JSON:

داده‌ها را با استفاده از کتابخانه JSON داریم به صورت مورد نیازمان تبدیل می‌کنیم.

- train_test_split:

از این تابع از کتابخانه sklearn، برای تقسیم داده‌ها به دو مجموعه train و test استفاده کردیم.

- svds:

از این تابع از کتابخانه scipy برای محاسبه SVD استفاده کردیم.

۲.۳. بخش دوم – خواندن دیتاست:

مقدماتی‌ترین چالش در مسائلی که با داده‌های بزرگ کار می‌کنند، گرفتن و خواندن داده‌ها و ذخیره آن‌ها در یک ماتریس (آرایه) است.

```
from google.colab import drive
drive.mount('/content/drive')
```

در این قسمت فقط برای دسترسی گوگل کولب به گوگل درایو دیتاست استفاده کردیم و میتوانید آن را درنظر نگیرید ویا دیتاست را در گوگل درایو خودتان قرار دهید. لینک دیتاست نیز در آدرس زیر موجود است.

http://snap.stanford.edu/data/amazon/productGraph/categoryFiles/ratings_Electronics.csv

با توجه به توضیحات مربوط به دیتاست داده شده، ستون‌های userId، productId و ratings را می‌گیریم و در لیست electronics_dataset قرار می‌دهیم. سپس ۵ سطر اول آن را با electronics_dataset.head()و اطلاعات داده را با electronics_dataset.info() نمایش می‌دهیم.

```
# Import the dataset and give the column names
columns=['userId', 'productId', 'ratings','timestamp']
electronics_dataset=pd.read_csv('/content/drive/MyDrive/AI Final
 Project/Copy of ratings_Electronics.csv',names=columns)
electronics_dataset.drop('timestamp',axis=1,inplace=True)

electronics_dataset.head()
electronics_dataset.info()
```

برای اطلاعات بیشتر نیز دستورات اضافه‌تری نیز قرار می‌دهیم مانند تعداد سطر و ستون دیتاست و جنس داده‌های موجود در هر ستون دیتاست (این بخش‌ها بودنشان اختیاری است و برای بدست آوردن یک تخمینی از حجم داده‌ها استفاده کردیم).

```
#Check the number of rows and columns
print('shape of the dataset (row,col):',electronics_dataset.shap
e)

#Check the datatypes
electronics_dataset.dtypes
```

یکی از کارهایی که با توجه به بزرگ بودن داده‌ها در منابع گوناگون پیشنهاد کردند، انتخاب یک زیرمجموعه‌ای از داده‌ها است. در این قسمت ما ۵۰۰۰۰ سطر اول مجموعه دیتاستمان را گرفتیم و در متغیر electronics_dataset_subset ذخیره کردیم.

سپس براساس ستون ratings اطلاعات گوناگونی نظیر میانه، انحراف معیار و ... را چاپ میکنیم و مینیموم و ماکسیموم امتیازات را نیز چاپ میکنیم. (این بخش نیز اختیاری است)

```
#Taking subset of the dataset
electronics_dataset_subset=electronics_dataset.iloc[:50000,0:]
electronics_dataset_subset.info()
print('\n')

#Summary
electronics_dataset_subset['ratings'].describe().transpose()

#minimum and maximum ratings
print('\n')
print('Minimum:',electronics_dataset_subset.ratings.min())
print('Maximum:',electronics_dataset_subset.ratings.max())
```

همچنین از روی userId و productId نیز تعداد کل کاربران و محصولات منحصر به فرد این ۵۰۰۰۰ سطر را نمایش می‌دهیم. (این بخش نیز اختیاری است)

```
# Count of unique user and product in the subset data
print('unique users= ', electronics_dataset_subset['userId'].nun
ique())
print('unique product= ', electronics_dataset_subset['productId'
].nunique())
```

۳.۳. بخش سوم – دقیق‌تر کردن داده‌ها:

حال میخواهیم کمی داده‌ها را دقیق‌تر کنیم. به این صورت که می‌خواهیم تعداد کاربرانی که به محصولات مختلف را دادند را حساب کنیم. یعنی در مجموع هر کاربر چند دفعه رای داده است یا به بیان دیگر به چند محصول رای داده است و روی این افراد تمرکز کنیم و از میان تمام کاربران با چاپ ۱۵ نفر اول این لیست سعی میکنیم حدودی از تعداد امتیاز دادن کاربران بدست آوریم.

```
#Check the top 15 users based on ratings (Count)
unique_users=electronics_dataset_subset.groupby('userId')
most_rated=unique_users.size().sort_values(ascending=False)[:15]
print('Top 15 users based on ratings: \n',most_rated)
```

با کمک خروجی کد بالا، به این نتیجه می‌رسیم روی کاربرانی که به بیش از ۱۵ کالا رای دادن تمرکز کنیم و آن را در electronics_dataset_final ذخیره می‌کنیم و کمی اطلاعات با توجه به فایل نهایی شدمان بدست می‌آوریم.

```
counts=electronics_dataset_subset.userId.value_counts()
electronics_dataset_final=electronics_dataset_subset[electronics
_dataset_subset.userId.isin(counts[counts>=15].index)]

print('Number of users who have rated 15 or more items =', len(e
lectronics_dataset_final))
print('Number of unique users in the final data = ', electronics
_dataset_final['userId'].nunique())
print('Number of unique products in the final data = ', electron
ics_dataset_final['productId'].nunique())
```

برای استفاده از الگوریتم باید جدول pivot را طراحی کنیم. برای ساخت این جدول از تابع pivot استفاده می‌کنیم که جدولی که در سطرها userId و در ستون‌ها productId و در خانه‌های هر سطر مقدار امتیازات (ratings) را قرار می‌دهیم و آن را final_ratings_matrix می‌نامیم.

با چاپ final_ratings_matrix نشان میدهیم این ماتریس اسپارس است و میتوان از روش SVD برای تبدیل به فضای کوچکتر و فشرده‌تر استفاده کرد.

```
#constructing the pivot table for Algorithm
final_ratings_matrix = electronics_dataset_final.pivot(index = '
userId', columns ='productId', values = 'ratings').fillna(0)
final_ratings_matrix

print('Shape of final_ratings_matrix: ', final_ratings_matrix.sh
ape)
#It shows that it is a sparse matrix. So, many cells are filled
with 0 values.
```

۴.۳. بخش چهارم – تقسیم داده‌ها:

داده‌ها را مشابه تمام مسائل علوم داده، به دو مجموعه test و train (با نرخ تبدیل ۷۰:۳۰) تبدیل می‌کنیم.

```
# Split the data randomnly into train and test datasets into 70:
30 ratio
# with train_test_split function
train_data, test_data = train_test_split(electronics_dataset_fin
al, test_size = 0.3, random_state=0)

print('Shape of training data: ',train_data.shape)
print('Shape of testing data: ',test_data.shape)
```

۵.۳. بخش پنجم – ساخت مدل فیلترینگ توصیه‌گر مشارکتی:

روش اصلی ما پیاده‌سازی این بخش است. با اتصال دوباره مجموعه train و test، مجموعه electronics_dataset_final_CF را تولید می‌کنیم.

```
electronics_dataset_final_CF = pd.concat([train_data, test_data]
).reset_index()
```

۱. حال از روی electronics_dataset_final_CF، جدول Pivot آن را تشکیل میدهیم و این همان مقدارهای واقعی ما است که آن را real_pivot_table می‌نامیم. سپس به هر سطر آن (کاربران منحصر به فرد) یک index نیز نسبت می‌دهیم.

```
# Constructing the pivot table for Algorithm
real_pivot_table = electronics_dataset_final_CF.pivot(index = 'u
serId', columns ='productId', values = 'ratings').fillna(0)

#define user index
real_pivot_table['user_index'] = np.arange(0, real_pivot_table.s
hape[0], 1)
real_pivot_table.set_index(['user_index'], inplace=True)
```

۲. تجزیه SVD ماتریس real_pivot_table به کمک تابع svds انجام میدهیم و سپس sigma را به فرم یک ماتریس قطری در میاریم.

```
# Singular Value Decomposition
P, sigma, Qt = svds(real_pivot_table, k = 10)
# Construct diagonal array in SVD
sigma = np.diag(sigma)
```

۳. حال از روی تجزیه SVD مقدار predicate_pivot_table را حساب می‌کنیم.

```
#Predicted ratings
all_user_predicted_ratings = np.dot(np.dot(P, sigma), Qt)
# Convert predicted ratings to dataframe
predicate_pivot_table = pd.DataFrame(all_user_predicted_ratings,
 columns = real_pivot_table.columns)
```

۴. هردو ماتریس real_pivot_table و predicate_pivot_table را چاپ میکنیم تا مقایسه کنید.

```
# Print Real Rating and Predicate Rating:
# so Actual ratings given by users:
print("Pivot Table:")
real_pivot_table

# and Predicate Rating given by SVD:
print("Predicate Table:")
predicate_pivot_table
```

۵. تابع توصیه‌گر مشارکتی:

در این تابع شماره کاربر (index) را میگیریم که باید ازش یکی کم کنیم چون از صفر شروع میشود. همچنین ماتریس real_pivot_table و predicate_pivot_table را حساب کنیم. تعداد پیشنهادات به فرد را با متغیر num_recommendations میگیریم.

سپس داده‌های امتیازات کاربر را بر اساس هم real_pivot_table و هم predicate_pivot_table مرتب میکنیم و سپس به تعداد num_recommendations، پیشنهاداتی که real_pivot_table در آن صفر است و predicate_pivot_table ماکسیموم است، را انتخاب می‌کنیم.

```
# Recommend the items with the highest predicted ratings
def recommend_items(userID, real_pivot_table, predicate_pivot_ta
ble, num_recommendations):
    # index starts at 0
    user_index = userID-1

    # Get and sort the user's ratings
        #sorted_user_ratings:
    sorted_user_ratings = real_pivot_table.iloc[user_index].sort
_values(ascending=False)

        #sorted_user_predictions:
    sorted_user_predictions = predicate_pivot_table.iloc[user_in
dex].sort_values(ascending=False)
```

```
    temp = pd.concat([sorted_user_ratings, sorted_user_predictio
ns], axis=1)
    temp.index.name = 'Recommended Items'
    temp.columns = ['user_ratings', 'user_predictions']
    temp = temp.loc[temp.user_ratings == 0]
    temp = temp.sort_values('user_predictions', ascending=False)
    print('\nBelow are the recommended items for user(user_id =
{}):\n'.format(userID))
    print(temp.head(num_recommendations))
```

۶. نمایش و محاسبه K محصول برتر هر فرد:

سپس برای نمایش کافی است فقط تابع مرحله ۵ را صدا بزنیم و آن را به ازای کاربر و تعداد K اجرا کنیم. برای مثال:

```
# pivod_df --> before SVD
# preds_df --> after SVD
userID = 4
num_recommendations = 5
recommend_items(userID, real_pivot_table, predicate_pivot_table,
 num_recommendations)
```

۶.۳. بخش ششم – ارزیابی:

برای بخش پایانی که هدف ارزیابی کل پروژه است، ابتدا یک نمایش از ۵ تای اول و سپس میانگین امتیازات هر محصول را مقایسه میکنیم.

```
# Actual ratings (users)
real_pivot_table.head()
# Average ACTUAL rating for each product
real_pivot_table.mean().head()

# Predicted ratings
predicate_pivot_table.head()
# Average PREDICTED rating for each product
predicate_pivot_table.mean().head()
```

و حالا با کمک کمترین مربعات (RMSE یا RMSD) می‌توانیم مقایسه کنیم و هرچه این عدد کمتر باشد به معنای دقیق‌تر بودن خروجی ما است.

$$\mathrm{RMSD} = \sqrt{\frac{\sum_{i=1}^{N} (x_i - \hat{x}_i)^2}{N}}$$

برای این کار دو جدول بالا را باهم ادغام میکنیم و سپس مولفه به مولفه در فرمول RMSE قرار میدهیم.

```
rmse_df = pd.concat([final_ratings_matrix.mean(), predicate_pivo
t_table.mean()], axis=1)
rmse_df.columns = ['Avg_REAL_ratings', 'Avg_PREDICATE_ratings']
rmse_df['item_index'] = np.arange(0, rmse_df.shape[0], 1)
rmse_df.head()
```

```
RMSE = round((((rmse_df.Avg_REAL_ratings - rmse_df.Avg_PREDICATE
_ratings) ** 2).mean() ** 0.5), 5)
print('\nRMSE = {} \n'.format(RMSE))
```

۴. تحلیل ارزیابی:

همانطور که در قسمت ششم بخش پیاده‌سازی راجع به استفاده از روش کمترین مربعات برای محاسبه و ارزیابی الگوریتم استفاده شده صحبت کردیم. در این قسمت میخواهیم مقدار ارزیابی شده را مورد بررسی قرار بدهیم. به نوعی کافی است نزدیک بودن مقدار واقعی و تخمینی را نشان بدهیم.

در خروجی زیر ۵ مقدار اول را نشان میدهیم که نزدیک بودن مقدار واقعی و تخمینی را میبینید و در پایین جدول مقدار کمترین مربعات حاصل از تمام کاربرانی که انتخاب کردیم را مشاهده میکنید که بسیار به صفر نزدیک است و دلالت بر درستی الگوریتم و دقیق بودن دارد.

productId	Avg_REAL_ratings	Avg_PREDICATE_ratings	item_index
1400599997	0.090909	0.088513	0
B00000DM9M	0.454545	0.449816	1
B00000J061	0.454545	0.558292	2
B00000J08C	0.454545	0.449816	3
B00000J0A2	0.363636	0.354053	4

```
RMSE = 0.05854
```

۵. جمع بندی:

ما برای حل این پروژه از روش‌های گوناگونی سعی کردیم استفاده کنیم و از مسائل متفاوت جستجو و ماشین لرنینگ سعی کردیم کمک بگیریم که بهترین راه حل میان تمام راه‌حل‌های موجود استفاده از توصیه‌گر مشارکتی در میان منابع مختلف بیشتر از سایر منابع پیشنهاد شده بود.

به طور کلی برای مسائل پیشنهاد محصول، کالا، فیلم، آهنگ و ... الگوریتم‌های توصیه‌گر مشارکتی بسیار استفاده می‌شوند و با توجه به خروجی ارزیابی نیز می‌توان به این نتیجه رسید که این امر بدون دلیل نیست.

منابع

1. Recommender Systems Specializati by Joseph A Konstan | Michael D.Ekstrand
2. Lecture 43 — Collaborative Filtering | Stanford University
3. Collaborative Filtering
4. Amazon review data set
5. What are Product Recommendation Engines? And the various versions of them? by Maruti Techlabs | Towards Data Science