به نام خدا

تمرین اول – مبانی علم داده **نیایش خانی ۹۹۵۲۱۲۳۵**

سوال ۱) MLE از روش‌های آماری مهم برای تخمین پارامترهای یک مدل است. در این روش، فرض می‌شود که مجموعه‌ای از داده‌های مشاهده‌شده از یک توزیع احتمالی پیروی می‌کنند، که پارامترهای آن توزیع ناشناخته‌اند. هدف MLE این است که مقادیری برای این پارامترهای ناشناخته پیدا کند که احتمال مشاهده داده‌های واقعی را بیشینه کند. برای مثال، اگر فرض کنیم داده‌ها از یک توزیع نرمال پیروی می‌کنند، MLE به ما کمک می‌کند تا میانگین و واریانس این توزیع را به گونه‌ای برآورد کنیم که داده‌های موجود را به بهترین نحو توصیف کند.

فرایند بیشینه‌سازی در MLE با استفاده از تابع درست‌نمایی انجام می‌شود. این تابع، احتمال وقوع داده‌های مشاهده‌شده را به‌عنوان تابعی از پارامترهای مدل در نظر می‌گیرد. سپس با مشتق‌گیری از این تابع نسبت به پارامترها و برابر قرار دادن آن با صفر، مقادیری برای پارامترها به‌دست می‌آید که این احتمال را بیشینه می‌کند. در برخی موارد، این فرایند به‌صورت تحلیلی قابل حل است، اما در موارد پیچیده‌تر، نیاز به روش‌های عددی دارد. این روش به دلیل خصوصیات مطلوبی که دارد مانند ناریبی و کارایی در نمونه‌های بزرگ، یکی از ابزارهای اصلی در آمار و یادگیری ماشین برای تخمین پارامترهاست.

سوال ۲) از آنجایی که برای $x<0$ تابع داریم، پس تابع چگالی فقط برای $x<0$ اعمال می شود.

با استفاده تابع *likelihood* خواهیم داشت:

$$L(\theta) = \prod_{i=1}^{n} f(X_i|\theta) = \prod_{i=1}^{n} \frac{-\theta X_i^{\theta-1}}{2^{\theta}}$$

سپس اگر از این *log* بگیریم:

$$\ell(\theta) = \log L(\theta) = \sum_{i=1}^{n} log\left(\frac{-\theta X_i^{\theta-1}}{2^{\theta}}\right)$$

$$\ell(\theta) = \sum_{i=1}^{n} (\log(-\theta) + (\theta-1)\log(-X_i) - \theta\log(2))$$

$$= nlog(-\theta) + (\theta-1)\sum_{i=1}^{n} \log(-X_i) - n\theta\log(2)$$

برای بدست آوردن *MLE* از $\ell(\theta)$ نسبت به θ مشتق می‌گیریم:

$$\frac{d\ell(\theta)}{d\theta} = \frac{n}{\theta} + \sum_{i=1}^{n} \log(-X_i) - nlog(2) = 0$$

$$\frac{n}{\theta} = nlog(2) - \sum_{i=1}^{n} \log(-X_i)$$

$$\theta = \frac{n}{nlog(2) - \sum_{i=1}^{n} \log(-X_i)}$$

بنابراین *MLE* برای θ برحسب n خواهد بود:

$$\hat{\theta} = \frac{n}{nlog(2) - \sum_{i=1}^{n} \log(-X_i)}$$

سوال ۳)

الف) احتمال آمدن عدد ۱ با استفاده از روش *Maximum Likelihood*:

$$L(p) = p^{15} - (1-p)^1$$

از این تابع *log* میگیریم:

$$\ell(p) = \log\big(L(p)\big) = 15\log(p) + \log(1-p)$$

سپس نسبت به p مشتق می‌گیریم:

$$\frac{d\ell(p)}{dp} = \frac{15}{p} - \frac{1}{1-p} = 0 \rightarrow 15(1-p) = p \rightarrow 15 - 15p = p \rightarrow 16p = 15$$

$$\rightarrow p = \frac{15}{16} \approx 0.93$$

در اینجا از *MLE* استفاده کردیم زیرا به ما این امکان را می دهد که پارامتری را بیابیم که بیشترین تطابق را باداده های مشاهده شده دارد و احتمال مشاهده داده های داده شده را به حداکثر می رساند.

ب) مراحل بالا را تکرار می‌کنیم:

$$L(p) = p^7 - (1-p)^1$$

$$\ell(p) = \log\big(L(p)\big) = 7\log(p) + \log(1-p)$$

$$\frac{d\ell(p)}{dp} = \frac{7}{p} - \frac{1}{1-p} = 0 \rightarrow 7(1-p) = p \rightarrow 7 - 7p = p \rightarrow 8p = 7$$

$$\rightarrow p = \frac{7}{8} \approx 0.87$$

ج) برای مقایسه تخمین‌های به دست آمده از ۱۶ پرتاب و ۸ پرتاب، مشاهده می کنیم که تخمین p با داده های بیشتر (۱۶ پرتاب) در مقایسه با نمونه کوچکتر (۸ پرتاب) کمی بالاتر است. می‌توانیم بگوییم تخمین با داده های بیشتر، دقیق تر است زیرا بر اساس تعداد بیشتری از مشاهدات است و واریانس برآورد را کاهش می دهد بنابراین با یک نمونه بزرگتر، ما تخمین

قابل اعتمادتری از p را داریم. اما با حجم نمونه کوچکتر ممکن است دقت ما به احتمال واقعی نزدیک نباشد زیرا نمونه های کوچکتر بیشتر مستعد نوسانات هستند و ممکن است به درستی احتمال اصلی را نشان ندهند.

بنابراین حجم نمونه بزرگتر، قابلیت اطمینان و دقت *MLE* را افزایش می دهد، زیرا اطلاعات بیشتری در مورد توزیع ارائه می دهد و تأثیر تغییرات تصادفی در نمونه را کاهش می دهد.

سوال عملی ۱)

- ***what is the usuage of the 'bins' parameter?*** : *Bins* تعداد *bar* ها را مشخص می کند.

- ***Do you see any limits?*** : یکی از محدودیت هایی که برای ترسیم هیستوگرام وجود دارد، حساسیت اندازه *bins* است زیرا اگر مقدار کمی برای آن قرار دهیم، ممکن است داده ها را بیش از حد *smooth* کند و جزئیات مهم در مورد توزیع را قابل مشاهده نباشد. همچنین داشتن *Bins* زیاد هم می‌تواند *Plot* را خیلی شلوغ و شناسایی الگوها دشوار کند.
  مشکل دیگر هنگام ترسیم *Dream Weight* در مقابل *Actual Weight* است که همپوشانی داده ها تفسیر مقادیر را چالش برانگیز می‌کند.
  همچنین هیستوگرام برای داده های پیوسته مناسب است و اگر ستونی مقادیر تکراری زیادی داشته باشد، هیستوگرام ممکن است بیشتر شبیه یک نمودار میله‌ای شود که در نتیجه توزیع را نادرست نشان می‌دهد. بنابراین مقایسه به درستی انجام نمی‌شود.

- ***different values for 'method' parameter in df.corr:***
  pearson: Standard correlation for linear relationships.
  kendall: Measures ordinal correlation.
  spearman: Measures rank correlation.

- ***What is axis parameter?*** : این پارامتر مشخص می‌کند که عملیات باید روی ردیف‌ها یا ستون‌ها اعمال شود. اگر *axis = 0* باشد، این عملیات روی ردیف ها و در صورتی که *1* باشد، عملیات روی ستون ها اعمال خواهد شد.

- ***What is inplace parameter?*** : این پارامتر تعیین می کند که عملیات *DataFrame* اصلی را تغییر دهد یا یک مورد جدید بسازد. اگر *inplace=True*، این عملیات به طور مستقیم *DataFrame* اصلی را تغییر می دهد و هیچ *DataFrame* جدیدی برگردانده نمی شود و در صورتی که *inplace=False* (پیش فرض) *DataFrame* اصلی را تغییر نمی‌دهدو در عوض یک کپی تغییر یافته را برمی‌گرداند.

سوال عملی ۲)

- ***How we can detect anomalies based on z-score?*** **:** برای تعیین اینکه چه چیزی به عنوان یک ناهنجاری یا پرت در یک مجموعه داده در نظر گرفته شود، یک مقدار آستانه (*threshold*) در نظر می‌گیریم. معمولا در *Z-score* از دو *threshold* زیر استفاده می شود:

- $|z - score| > 2$
- $|z - score| > 3$

- ***plot the PODs for abnormal users*** **:**

تعدادی از کاربران غیرعادی مقادیر *POD* پایینی دارند که در حدود ۰/۲-۰/۳ است. همچنین تعدادی نیز مقادیر *POD* نزدیک به ۰/۷-۰/۹ دارند. بنابراین نمودار به دو دسته مقادیر *POD* پایین و *POD* بالاتر تقسیم شده است. اگر در نظر بگیریم که *POD* مخفف *Proof of Delivery* باشد، می توان این تفسیر را داشت که برای مقادیر پایین *POD*، کاربران ممکن است فرکانس پایینی از تحویل تایید شده داشته باشند. آنها ممکن است سفارش های بزرگ یا پراکنده انجام دهند که می تواند به معنای رفتار خرید نامنظم یا غیر معمول باشد که چنین کاربرانی ممکن است الگوهای هزینه غیرقابل پیش بینی داشته باشند که منجر به طبقه بندی آنها به عنوان *anomaly* می شود. همچنین برای مقادیر بالای *POD* نیز ممکن است نشان دهنده کاربران وفادار و با فرکانس بالا باشد که هم بیشتر هزینه می کنند و هم تعداد تحویل تایید شده بالایی دارند.

سوال عملی ۳)

- بارگذاری و آماده‌سازی داده‌ها :

ابتدا دیتاست با استفاده از *panadas* لود شد و جداکننده مطابق دیتاست روی `;` تنظیم شد. این کار به ما اجازه می‌دهد تا ستون‌ها را به‌عنوان «*User-ID*»، «*ISBN*» و «*Book-Ratings*» داشته باشیم.

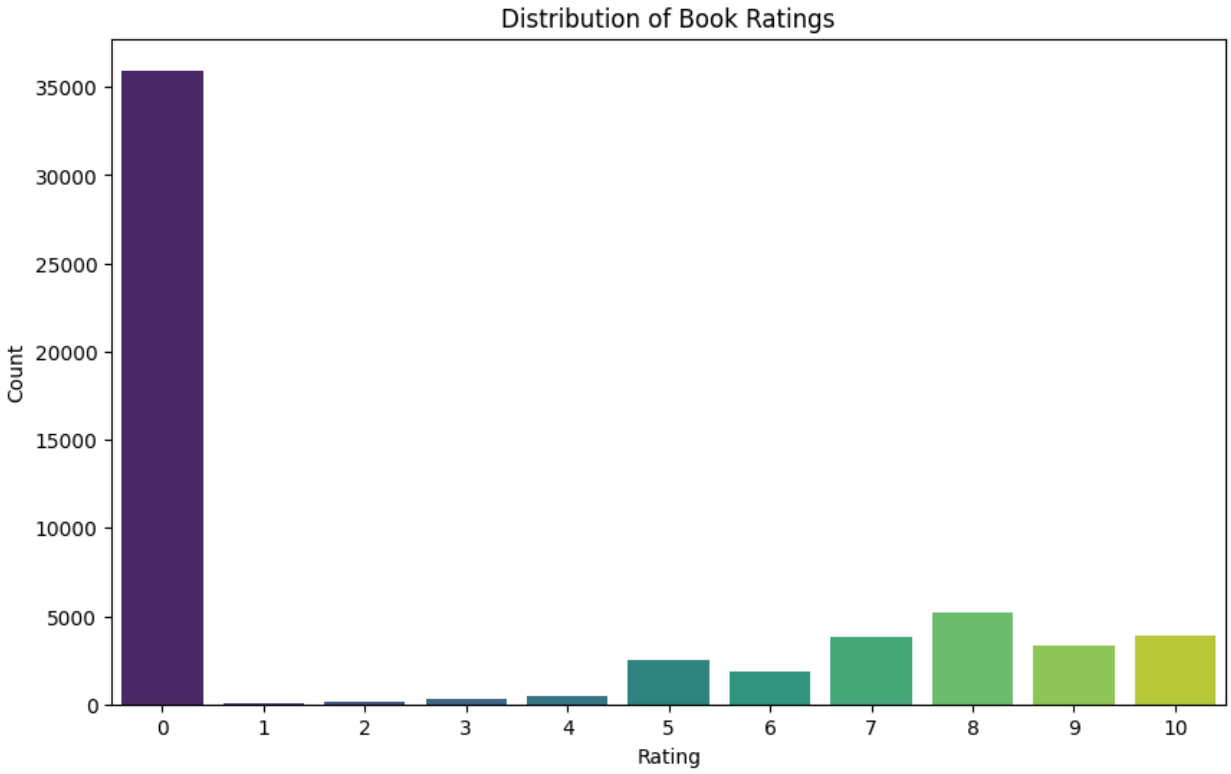
Note : به دلیل بزرگ بودن دیتاست، ۵٪ از دیتا را به صورت رندوم به عنوان نمونه انتخاب کردیم.

برای آماده‌سازی دیتا، یک ماتریس امتیازدهی ایجاد کردیم که در آن ایندکس نشان‌دهنده کاربران، ستون شناسه کتاب (*ISBN*) و مقادیر آنها را امتیاز کتاب‌ها قرار دادیم. از آنجایی که ماتریس ما دارای مقادیر *Nan* است، این مقادیر را با میانگین امتیازها پر کردیم.

- تحلیل داده‌ها:

ابتدا با استفاده از *dtypes* تایپ فیچرها را فهمیدیم. سپس تعداد کاربران و کتاب‌های منحصربه‌فرد را محاسبه کردیم تا اندازه کلی مجموعه داده و میزان تنوع کاربر و آیتم را ببینیم.

سپس *distribution* امتیازها را بررسی کردیم و تعداد هر *rate* را بدست آوردیم. همچنین برای اینکه درک بهتری از این توزیع داشته باشیم، نمودار آن را رسم کردیم.



همانطور که مشاهده می‌شود مقدار قابل توجهی از امتیازات صفر بوده است.

پراکندگی زیاد ماتریس به این معنی است که اکثر کاربران تنها زیر مجموعه کوچکی از کتاب های موجود را رتبه بندی کرده اند که توصیه های مبتنی بر شباهت را چالش برانگیز می کند زیرا موارد مشترک کمی بین کاربران وجود دارد.

- پیاده‌سازی *Collaborative Filtering* :
  با توجه به بزرگ بودن دیتاست، ران تایم *correlation* بسیار زیاد بود بنابراین از *Cosine Similarity* استفاده کردیم.

  همچنین مجموعه داده‌های پراکنده می‌تواند منجر به مشکلاتی در محاسبه *correlation* بین کاربران شود، به ویژه در هنگام استفاده از متود *pearson*، زیرا بر آیتم هایی که *overlap* دارند متکی است. *Cosine Similarity* پراکندگی را بهتر کنترل می‌کند زیرا فقط جهت *preferences* را در نظر می گیرد.

- فانکشن پیشنهاد کتاب:
  ابتدا شباهت *user_id* مورد نظر را با کاربران *user_similarity_df* (که حاوی مقادیر شباهت از پیش محاسبه شده بین همه جفت کاربر است) پیدا می کنیم.
  سپس *rating_matrix* را می گیریم و هر ردیف (*user*) را در امتیاز شباهت مربوطه از *user_similarities* ضرب می کنیم. با این کار رتبه بندی هر کاربر بر اساس شباهت آنها به *user_id* مقیاس بندی می شود.
  کاربرانی که شباهت بیشتری دارند تأثیر بیشتری روی توصیه خواهند داشت.
  سپس مجموع وزن دار رتبه‌بندی‌ها را برای هر کتاب با جمع‌بندی رتبه‌بندی‌های همه کاربران (پس از مقیاس‌بندی بر اساس شباهت) برای هر کتاب محاسبه می‌کنیم.

این به ما یک "امتیاز پیش بینی شده" ترکیبی برای هر کتاب بر اساس ترجیحات کاربران مشابه *user_id* می دهد. ما این مقدار را بر مجموع نمرات شباهت تقسیم می‌کنیم تا نتایج را نرمالایز کنیم، بنابراین تعداد کاربرانی که به یک کتاب خاص رتبه‌بندی کرده‌اند، مغرضانه نیستند.
سپس کتاب‌هایی را که *user_id* قبلاً رتبه‌بندی کرده است شناسایی می‌کنیم و آنها را از لیست توصیه‌ها حذف می‌کنیم. این مانع از توصیه کتاب‌هایی می‌شود که کاربر قبلاً به آنها *rate* داده است.
در نهایت از *nlargest(n_recommendations)* استفاده می کنیم تا کتاب های برتر *n_recommendations* را با بالاترین امتیازات پیش بینی شده بدست آوریم و لیست این کتاب ها را برمی‌گردانیم.

```
# Choose a random user ID from the sample data
random_user_id = df_sample['User-ID'].iloc[random.randint(0, rating_matrix.shape[0])]
print(f"Testing recommendations for Random User ID: {random_user_id}")

# Get recommended books for the randomly selected user
recommended_books = recommend_books(random_user_id, rating_matrix_filled, user_similarity_df)
print(f"Recommended books for User {random_user_id}: {recommended_books}")
```

✓ 9.4s

```
Testing recommendations for Random User ID: 67840
Recommended books for User 67840: ['0446611778', '0451179803', '0060217863', '0689835825', '0836280660']
```

برای تست عملکرد این سیستم، با استفاده تابع *random* ، از میان *user id* های موجود، یکی را انتخاب می‌کنیم و لیست کتاب های پیشنهادی این کاربر چاپ می شود.

همانطور که مشاهده می‌شود برای کاربر با آیدی رندوم ۶۷۸۴۰ ، *ISBN* کتاب های توصیه شده چاپ شده است.

مقایسه دو روش:

Cosine Similarity : این متود با مجموعه داده ما بهتر عمل کرد زیرا ماتریس ما پراکنده بود. شباهت کسینوس کمتر تحت تأثیر کاربرانی قرار می گیرد که در مقیاس های مختلف امتیاز می دهند (به عنوان مثال، کاربرانی که اکثراً بالا یا اکثراً پایین امتیاز می دهند).

Pearson Correlation: در حالی که این روش می‌تواند شباهت رتبه‌بندی کاربران را با دقت بیشتری ثبت کند، با پراکندگی مجموعه داده ما مشکل داشت.

نتیجه گیری : *Cosine Similarity* به دلیل ماهیت پراکنده داده‌ها مؤثرتر بود، در حالی که *Correlation* ممکن است در مجموعه داده‌های متراکم‌تر که در آن کاربران بسیاری از موارد مشابه را رتبه‌بندی می‌کنند، بهتر عمل کند.