باسمه تعالی

درس: داده کاوی (فاز۲)

نام و نام خانوادگی: پارسا هدایت نیا ، سروش پسندیده

شماره دانشجویی: ۹۸۲۲۷۶۲۰۳۹ ، ۹۹۱۲۷۶۲۱۰۹

در این بخش از پروژه از دیتاست ترکیب شده دو دیتا ست google play و play store استفاده میکنیم که این ترکیب به واسطه inner join بر روی دو ستون App Name از دیتاست اول و App از دیتاست دوم صورت گرفته است. سپس ستون هایی با نام مشابه ایجاد شده و ستون هایی که یک مفهوم دارند را ترکیب کرده تا به یک دیتاست مرجع با نام Combined_df برسیم که برای ورودی این بخش استفاده کنیم.

فهرست مطالب

در ادامه این بخش به منظور دسته بندی به سراغ انتخاب ویژگی های مرتبط با پیش بینی Rating دارند می‌رویم که ابتدا یک تحلیل خطی به واسطه ماتریس همبستگی بر روی داده ها انجام داده و تحلیل میکنیم و سپس normal Rating را به ordinal categorical تبدیل می‌کنیم تا از آن در naive_bayes and decision tree and random forest استفاده کنیم.

سپس مدل خود را آموزش و نهایتا به واسطه چهار معیار (Accuracy, Precision, Recall, F1 Score) رفته تا مدل آموزش دیده خود را ارزیابی کنیم.

توضیحات بخش اول:

در داده کاوی، الگوهای مکرر تکنیکی است که برای شناسایی الگوهایی که اغلب در یک مجموعه داده ظاهر می شوند استفاده می شود. این فرآیند برای کارهایی مانند تجزیه و تحلیل سبد بازار بسیار مهم است، جایی که درک اینکه کدام اقلام اغلب با هم خریداری می شوند می تواند تصمیمات تجاری را تعیین کند. در اینجا به تفکیک مفهوم و اجزای آن می پردازیم:

اهداف کاوی الگوی مکرر

- Identify Frequent Itemsets: مجموعه ای از اقلام را پیدا کنید که اغلب با هم در تراکنش ها ظاهر می شوند.
- کشف قوانین انجمن: قوانینی را تعیین کنید که چگونگی حضور اقلام خاص در یک تراکنش را نشان می دهد که وجود موارد دیگر وجود دارد.

مفاهیم اصلی

- Itemset : مجموعه ای از یک یا چند مورد.
- Support : فراوانی (یا نسبت) تراکنش ها در پایگاه داده که شامل مجموعه آیتم های خاصی است. یک مجموعه آیتم در صورتی مکرر در نظر گرفته می شود که پشتیبانی آن از یک آستانه از پیش تعریف شده فراتر رود.

- Confidence : احتمال اینکه یک تراکنش حاوی آیتم A نیز شامل آیتم B باشد. برای ارزیابی قواعد تداعی استفاده می شود.
- Association Rule : یک عبارت ضمنی از شکل $A \rightarrow B$ به این معنی که اگر مجموعه آیتم A رخ دهد، مجموعه آیتم B نیز احتمالا رخ خواهد داد.

الگوریتم‌هایی برای استخراج الگوهای مکرر

- ❖ Apriori Algorithm
- ❖ FP-Growth (Frequent Pattern Growth)

مراحل استخراج الگوهای مکرر

۱. پیش پردازش داده: مجموعه داده را پاک کرده و آن را به یک قالب مناسب تبدیل کنید.

۲. ایجاد مجموعه‌های مکرر: از الگوریتم‌هایی مانند Apriori یا FP-Growth برای یافتن همه مجموعه‌هایی که حداقل آستانه پشتیبانی را دارند، استفاده کنید.

۳. ایجاد قوانین انجمن: از مجموعه اقلام مکرر، قوانینی را ایجاد کنید که حداقل آستانه اطمینان را برآورده کنند.

توضیح روند انجام پروژه به شرح زیر است:

همانطور که گفته شد در این بخش میخواهیم الگو های پرتکرار را استخراج کنیم پس ابتدا کتابخانه مد نظر را import کرده و سپس دیتاست ترکیبی مذکور را بارگیری کرده و حال به سراغ یک تکنیک پیش پردازش می‌رویم.

Binning یک تکنیک پیش پردازش داده است که در تجزیه و تحلیل داده ها و یادگیری ماشین برای تبدیل متغیرهای عددی پیوسته به دسته های گسسته یا "bins" استفاده می شود. هدف اصلی binning کاهش تأثیر خطاهای مشاهده جزئی و نویز، و مدیریت توزیع‌های اریب است که تجزیه و تحلیل و تفسیر داده‌ها را آسان‌تر می‌کند.

Binning می تواند به ویژه برای موارد زیر مفید باشد:

- ⊕ ساده سازی مدل با کاهش تعداد مقادیر منحصر به فرد.
- ⊕ کاهش تأثیر عوامل پرت.
- ⊕ بهبود عملکرد برخی از الگوریتم های یادگیری ماشین با تبدیل داده های پیوسته به داده های طبقه بندی شده.

انواع Binning به شرح زیر است:

- Equal-Width Binning
- Equal-Frequency Binning (Quantile Binning)
- Custom Binning

در بخش نخست کد ستون های عددی (numerical columns) با توزیع خوب (well-distributed) و اریب (skew-distributed) را در سه دسته قرار می دهد: کم، متوسط و زیاد.

این فرآیند binning به ساده‌سازی داده‌ها، مدیریت توزیع‌های اریب، و به طور بالقوه بهبود عملکرد و تفسیرپذیری تحلیل‌های بعدی یا مدل‌های یادگیری ماشین و همچنین یافتن الگوهای مکرر کمک می‌کند.

حال مجموعه آیتم های مکرر را با استفاده از الگوریتم Apriori با حداقل آستانه support ۰/۰۵ پیدا می‌کنیم. این بدان معنی است که هر مجموعه ای که حداقل در ۵٪ از معاملات ظاهر شود، مکرر در نظر گرفته می شود.

	support	itemsets
2	0.94256	(Normal_Revenue_binned: Zero)
1	0.94240	(Normal_Price_binned: Zero)
7	0.94240	(Normal_Revenue_binned: Zero, Normal_Price_binned: Zero)
3	0.93056	(Type: 0)
8	0.93056	(Normal_Price_binned: Zero, Type: 0)
9	0.93056	(Normal_Revenue_binned: Zero, Type: 0)
13	0.93056	(Normal_Revenue_binned: Zero, Type: 0, Normal_Price_binned: Zero)
0	0.84240	(Content Rating: Everyone)
5	0.79520	(Content Rating: Everyone, Normal_Revenue_binned: Zero)
4	0.79504	(Content Rating: Everyone, Normal_Price_binned: Zero)
10	0.79504	(Normal_Revenue_binned: Zero, Content Rating: Everyone, Normal_Price_binned: Zero)
6	0.78352	(Content Rating: Everyone, Type: 0)
11	0.78352	(Content Rating: Everyone, Normal_Price_binned: Zero, Type: 0)
12	0.78352	(Content Rating: Everyone, Normal_Revenue_binned: Zero, Type: 0)
14	0.78352	(Normal_Revenue_binned: Zero, Content Rating: Everyone, Normal_Price_binned: Zero, Type: 0)

Frequent itemsets sample

پس از یافتن الگو های مکرر با استفاده از الگوریتم Apriori، itemset های maximal را شناسایی میکنیم.

مجموعه اقلام ماکسیمال، بزرگترین مجموعه اقلام مکرر هستند (یعنی زیرمجموعه هیچ مجموعه اقلام مکرر دیگری نیستند).

	support	itemsets
14	0.78352	(Normal_Revenue_binned: Zero, Content Rating: Everyone, Normal_Price_binned: Zero, Type: 0)

Maximal

در نهایت association rules را از مجموعه آیتم های maximal با استفاده از آستانه confidence ۰/۷ استخراج کرده ایم که این بدان معنی است که برای یک قانون A -> B، حداقل ۷۰٪ از تراکنش های حاوی A نیز حاوی B هستند.

	antecedents	consequents	antecedent support	consequent support	support
0	(Content Rating: Everyone, Normal_Revenue_binne...	(Type: 0)	NaN	NaN	0.78352
1	(Content Rating: Everyone, Normal_Revenue_binne...	(Normal_Price_binned: Zero)	NaN	NaN	0.78352
2	(Normal_Revenue_binned: Zero, Type: 0, Normal_P...	(Content Rating: Everyone)	NaN	NaN	0.78352
3	(Content Rating: Everyone, Normal_Price_binned:...	(Normal_Revenue_binned: Zero)	NaN	NaN	0.78352
4	(Content Rating: Everyone, Normal_Revenue_binne...	(Normal_Price_binned: Zero, Type: 0)	NaN	NaN	0.78352
5	(Normal_Revenue_binned: Zero, Normal_Price_binn...	(Content Rating: Everyone, Type: 0)	NaN	NaN	0.78352
6	(Normal_Revenue_binned: Zero, Type: 0)	(Content Rating: Everyone, Normal_Price_binned: Zero)	NaN	NaN	0.78352
7	(Content Rating: Everyone, Normal_Price_binned:...	(Normal_Revenue_binned: Zero, Type: 0)	NaN	NaN	0.78352
8	(Content Rating: Everyone, Type: 0)	(Normal_Revenue_binned: Zero, Normal_Price_binned: Zer...	NaN	NaN	0.78352
9	(Normal_Price_binned: Zero, Type: 0)	(Content Rating: Everyone, Normal_Revenue_binned: Zero)	NaN	NaN	0.78352
10	(Normal_Revenue_binned: Zero)	(Content Rating: Everyone, Normal_Price_binned: Zero, ...	NaN	NaN	0.78352
11	(Content Rating: Everyone)	(Normal_Revenue_binned: Zero, Type: 0, Normal_Price_bi...	NaN	NaN	0.78352
12	(Normal_Price_binned: Zero)	(Content Rating: Everyone, Normal_Revenue_binned: Zero...	NaN	NaN	0.78352
13	(Type: 0)	(Content Rating: Everyone, Normal_Revenue_binned: Zero...	NaN	NaN	0.78352

Association Rules

توضیحات بخش دوم:

بخش اول:

در این بخش به انتخاب خود می‌خواهیم دو الگوریتم یادگیری بدون ناظر (Kmeans و DBSCAN) را بر روی داده های خود پیاده سازی کنیم تا در بخش بعدی به الگوهای مناسبی دست بیابیم.

برای این امر ابتدا به واسطه کتابخانه sklearn و ماژول های Kmeans و DBSCAN را import کرده حال داده های دیتاست ترکیبی خود را بارگیری کرده و داخل یک دیتا فریم با نام df_cluster ذخیره می‌کنیم.

سپس یک تحلیل آماری بر روی ستون های عددی خود که شامل ستون های Rating Count، Rating، Reviews، Size، Installs، Price، Revenue، popularity می‌باشند انجام داده که این تحلیل شما مینیمم، ماکزیزمم و تعداد داده های آنهاست که به ما در تصمیم گیری بخش های بعدی کمک خواهد کرد.

در ادامه ستون های عددی خود را پردازش کرده و به داده های categorical تغییر می‌دهیم. هدف این کد، پیش پردازش ستون های عددی در دیتا فریم df_cluster با محاسبه فواصل bin بهینه با استفاده از قانون Freedman-Diaconis و سپس گسسته سازی (binning) این ستون ها است که به ما یک binding پویا را ارائه می‌دهد. همچنین میتوانیم از binding ثابت (static) نیز بهره ببریم که در این بخش ما هر دو مورد را پیاده سازی کرده ایم.

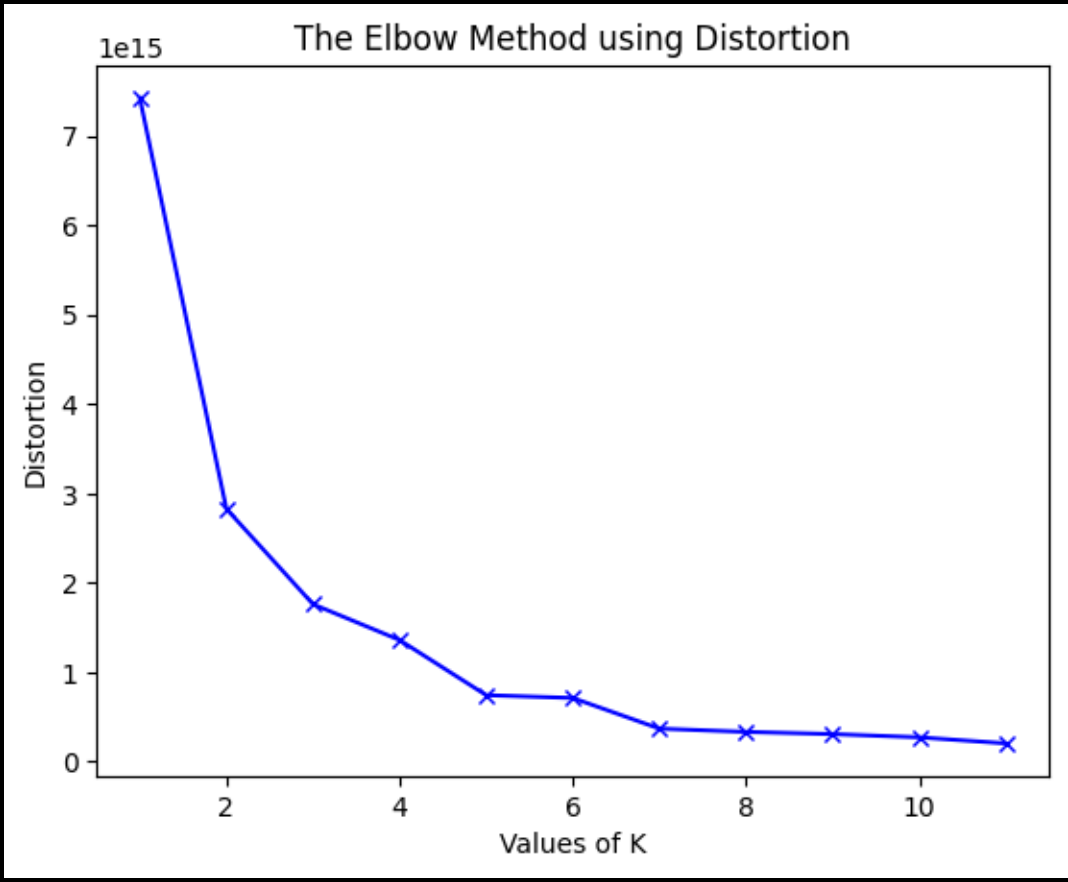
این ترکیب بندی می تواند به آماده سازی داده ها برای خوشه بندی کمک کند، زیرا داده های عددی را با گروه بندی آنها در فواصل گسسته ساده می کند. علاوه بر این، یک ستون بولی را پردازش می کند و با انداختن ستون های غیر ضروری DataFrame را برای خوشه بندی آماده می کند.

- انتخاب ستون های عددی مربوطه
- محاسبه فواصل بهینه bin با استفاده از قانون Freedman-Diaconis.
- گسسته سازی (باین بندی) ستون های عددی به فواصل.
- تبدیل یک ستون خاص به بولی
- دور انداختن ستون های غیر ضروری برای حفظ تنها ستون های مورد نیاز برای خوشه بندی.

سپس دیتاست ما آماده خوشه بندی است پس باید متناسب با الگوریتم اول خوشه بندی (kmeans) یک K مناسب برای آن بیابیم که این کار با استفاده از elbow method صورت گرفته که از k ۱ تا ۱۲ تست و نتایج را نمایش می‌دهد.

به طور کلی مراحل انجام آن به شرح زیر است:

۱. با انتخاب ستون های مربوطه، داده ها را برای خوشه بندی آماده می کند.
۲. خوشه بندی K-means را برای محدوده ای از مقادیر K (۱ تا ۱۱) اعمال می کند.
۳. distortion و اینرسی(inertia) را برای هر مقدار K محاسبه می کند.
۴. distortion ها را در برابر مقادیر K ترسیم می کند تا روش Elbow را تجسم کند و به تعیین تعداد بهینه خوشه ها برای مجموعه داده کمک کند.



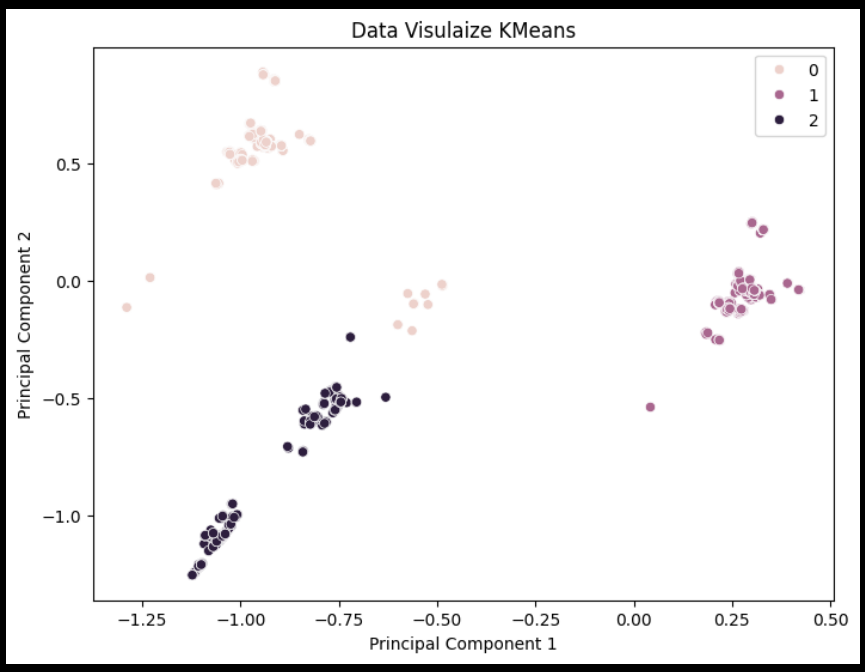
تصویر ۱ : elbow method

پس از انتخاب درست k میخواهیم الگوریتم k-means را بر روی داده های فیلتر شده خود اعمال کنیم تا خوشه بندی خوبی داشته باشیم.

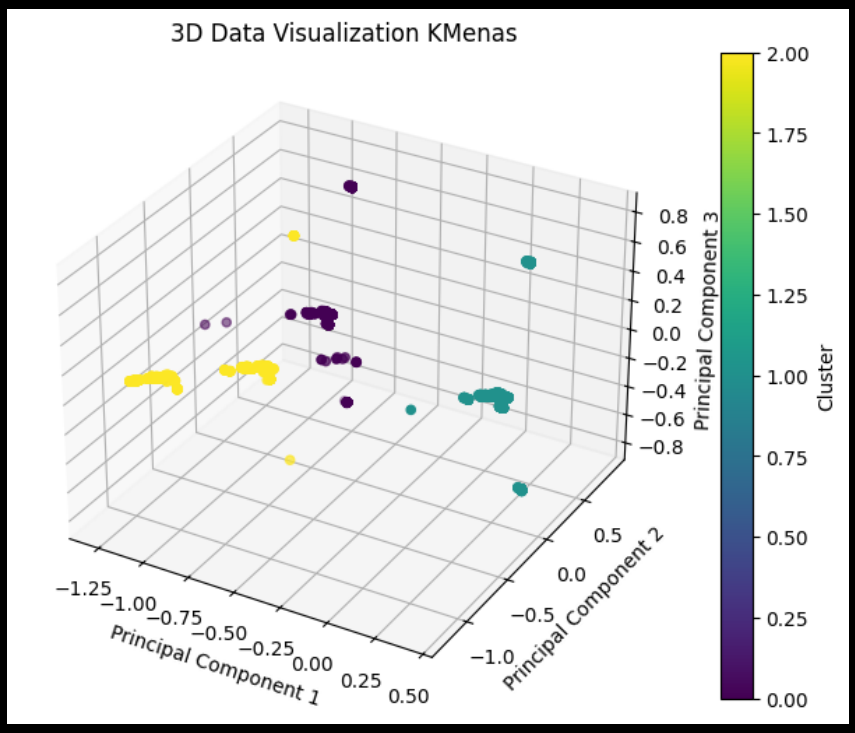
حال به سراغ دومین الگوریتم خوشه بندی خود میرویم که DBSCAN است.

حال باید الگو هایی که در خوشه ها شناخته می‌شوند را مرود بررسی قرار بدیم که بدین منظور ابتدا مشابه بخش الگو های پرتکرار یک تابع برای یافتن الگو های پرتکرار زده که در تابع get_frequent_itemsets

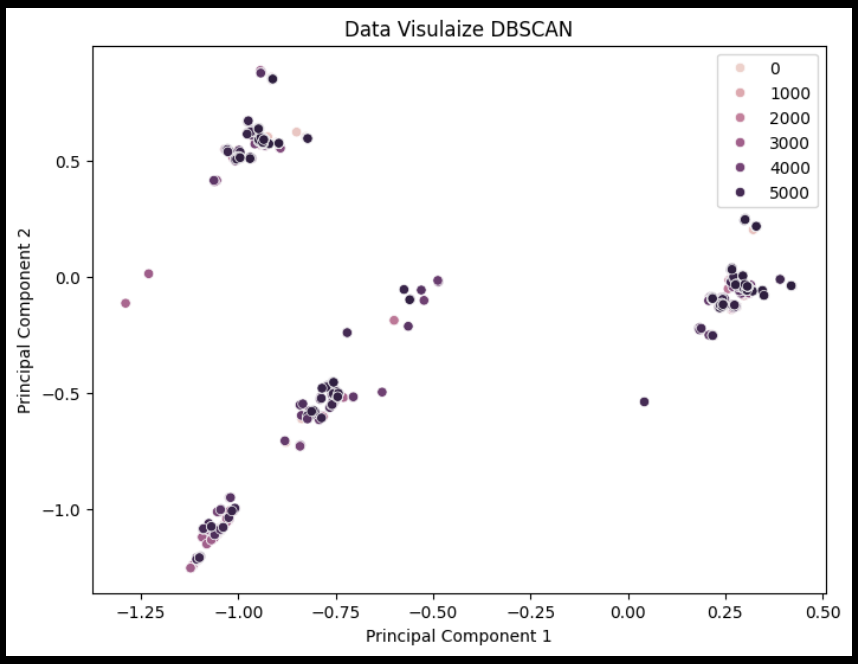
برای یافتن مجموعه آیتم های مکرر در یک مجموعه داده با استفاده از الگوریتم Apriori طراحی شده است.
این تابع برای کشف الگوها و ارتباط بین دو ستون مشخص شده در مجموعه داده است.



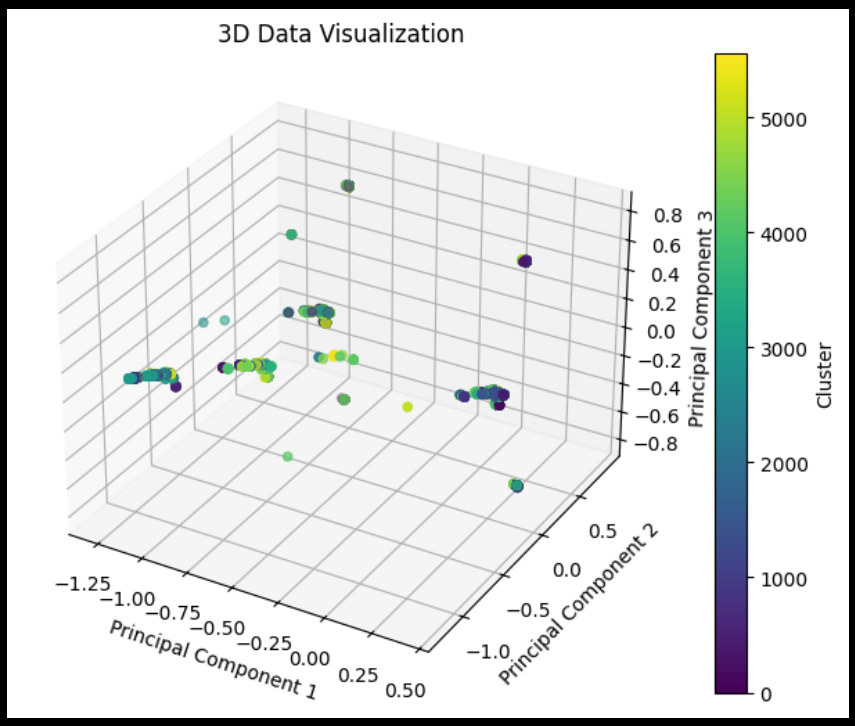
تصویر ۲ : k-means 2D



تصویر ۳ : k-means 3D



تصویر ٤: DBSCAN 2D



تصویر ٥: DBSCAN 3D

پس به واسطه این تابع به سراغ بررسی الگو های پرتکرار داخل هر کلاستر می‌پردازیم که در آن به استخراج قوانین ارتباطی مبتنی بر خوشه بر روی مجموعه داده ای است که توسط df_dummies نشان داده شده پرداخته ایم.

مراحل بخش فوق به شرح زیر است:

- ستون های بولی را در هر خوشه شناسایی کنید.
- مجموعه آیتم های مکرر برای جفت ستون های بولی را محاسبه کنید.
- قوانین ارتباط را از این مجموعه آیتم های مکرر استخراج کنید.
- قوانین کشف شده را به همراه ابرداده (شماره خوشه) در result_set ذخیره کنید.
- به طور کلی، هدف کد کشف ارتباطات معنی‌دار بین ویژگی‌های بولی در هر خوشه از مجموعه داده است، که به طور بالقوه الگوهای جالب یا همبستگی‌های خاص هر خوشه شناسایی شده توسط الگوریتم خوشه‌بندی (kmeans_labels) را آشکار می‌کند.

حال به سراغ چاپ نتایج به دست آمده از فرآیند استخراج قوانین انجمنی در هر خوشه می‌رویم.

هدف و نحوه عملکرد آن را به صورت زیر است:

۱. Header Print Statement
۲. Iterating Over Results
۳. Printing Association Rules
۴. Formatting

تکرار روی قوانین: برای هر تاپل filtered_df_tuple (که شامل قوانین تداعی به صورت تاپل است)، از طریق هر قانون (قانون) تکرار می شود.

Printing Rule Details: معیارهای مختلف مربوط به هر قانون مرتبط را چاپ می کند:

antecedent and consequent: موارد دخیل در قاعده.

support: پشتیبانی از قانون در مجموعه داده.

confidence: اطمینان از قاعده.

lift: بالابر قاعده.

leverage, conviction, added_value, synergy, complementary: معیارهای اضافی که ممکن است بینشی در مورد قدرت و ماهیت انجمن ارائه دهد.

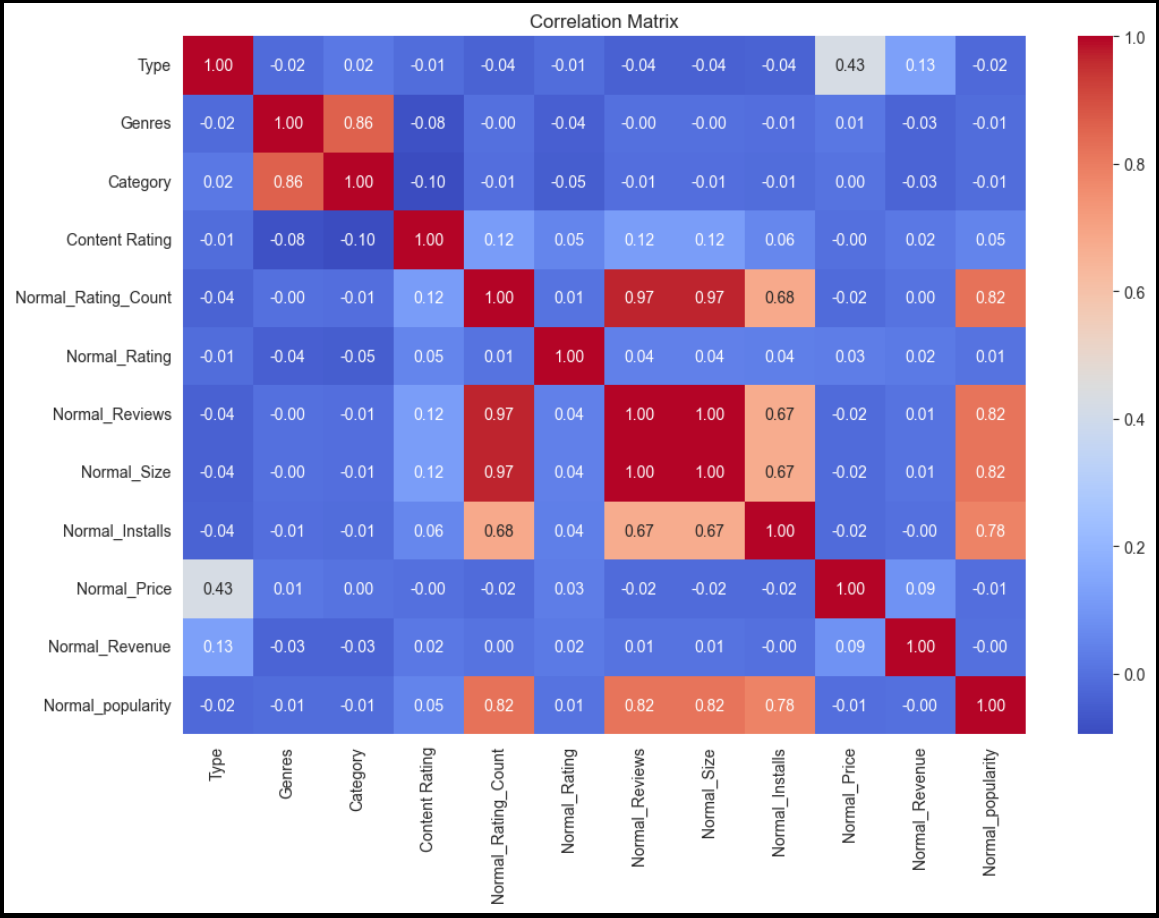
پس از خوشه بندی، تحلیل نوبت مصور سازی است که با کاهش بعد به صورت دو نمودار دو بعدی و سع بعدی نمایش داده شده است که ملموس تر باشد.

بخش دوم:

حال دربخش انتهایی به سراغ یادگیری با ناظر می‌رویم که به الگوریتم های دسته بندی (classification) معروف هستند.

اولین کاری که برای این بخش انجام داده ایم رمزگذاری برچسب برای داده های دسته بندی ترتیبی یا به عبارتی Label Encoding for ordinal categorical data که توسط ماژول LableEncoder از کتابخانه sklearn بهره برده ایم.

سپس به سراغ انتخاب ویژگی رفته ایم که برای این کار دیتافریم df_relevant را برای ذخیره سازی داده های مرتبط ساخته سپس به واسطه ماتریس همبستگی (correlation matrix) می توانیم میزان همبستگی میان Rating و سایر بخش ها را مشاهده کنیم.



تصویر ٦: ماتریس همبستگی (correlation Matrix)

بنابراین بر اساس نتیجه همبستگی، به نظر می‌رسد که رتبه‌بندی با سایر ویژگی‌ها همبستگی پایینی دارد، بنابراین رابطه خطی قوی با دیگران ندارد، بنابراین باید آن را با استفاده از روش‌هایی مانند درخت تصمیم یا جنگل تصادفی (نه رگرسیون خطی) طبقه‌بندی کنیم، پس ویژگی های مرتبطی را انتخاب کرده که نقش عمده ای در پیش بینی متغیر هدف دارند.

حال با این نتیجه گیری به سراغ تبدیل normal_rating به نوع طبقه بندی ترتیبی برای استفاده از درخت تصمیم و جنگل تصادفی رفته ایم.

برای این کار ما نیاز داریم نخست داده های خود را تقسیم بندی کنیم.

برای ارزیابی عملکرد مدل، مجموعه داده را به مجموعه های آموزشی و آزمایشی تقسیم کردیم.

ویژگی ها را برای استانداردسازی محدوده متغیرهای مستقل مقیاس میکنیم.

پس از این کار وارد فاز آموزش شده و سه مدل برای خود می سازیم برای DecisionTree، RandomForest و نهایتا GaussianNB (naive_bayes) سپس مراحل زیر به ترتیب اجزا شده اند.

با استفاده از Greed Search برای hyperParameter Tuning انواع حالات را تعریف میکنیم و مدل ها را آموزش میدهیم تا بهترین hyperparameter هارا بیابیم.

چند طبقه‌بندی‌کننده جنگل تصادفی را آموزش داده ایم و روی مجموعه آزمایشی پیش‌بینی کرده.

چند طبقه بندی کننده ساده بیز را آموزش داده ایم و روی مجموعه آزمایشی پیش بینی کرده.

چند طبقه‌بندی درخت تصمیم را آموزش داده ایم و روی مجموعه آزمایشی پیش‌بینی کرده.

```
Training Decision Tree...
Fitting 5 folds for each of 60 candidates, totalling 300 fits
Training Random Forest...
Fitting 5 folds for each of 144 candidates, totalling 720 fits
Training Naive Bayes...
Fitting 5 folds for each of 8 candidates, totalling 40 fits
```

Hyperparameter tuning

و سپس برای هر یک از آنها یک _grid_search در نظر گرفته و سپس Score(امتیاز) را برای هر یک از مدل ها محاسبه کرده و بهترین مدل را انتخاب میکنیم.

```
DecisionTreeClassifier(criterion='entropy', max_depth=5, min_samples_leaf=5,
                       random_state=42)
RandomForestClassifier(max_depth=20, min_samples_split=5, random_state=42)
GaussianNB(var_smoothing=0.0001)
```

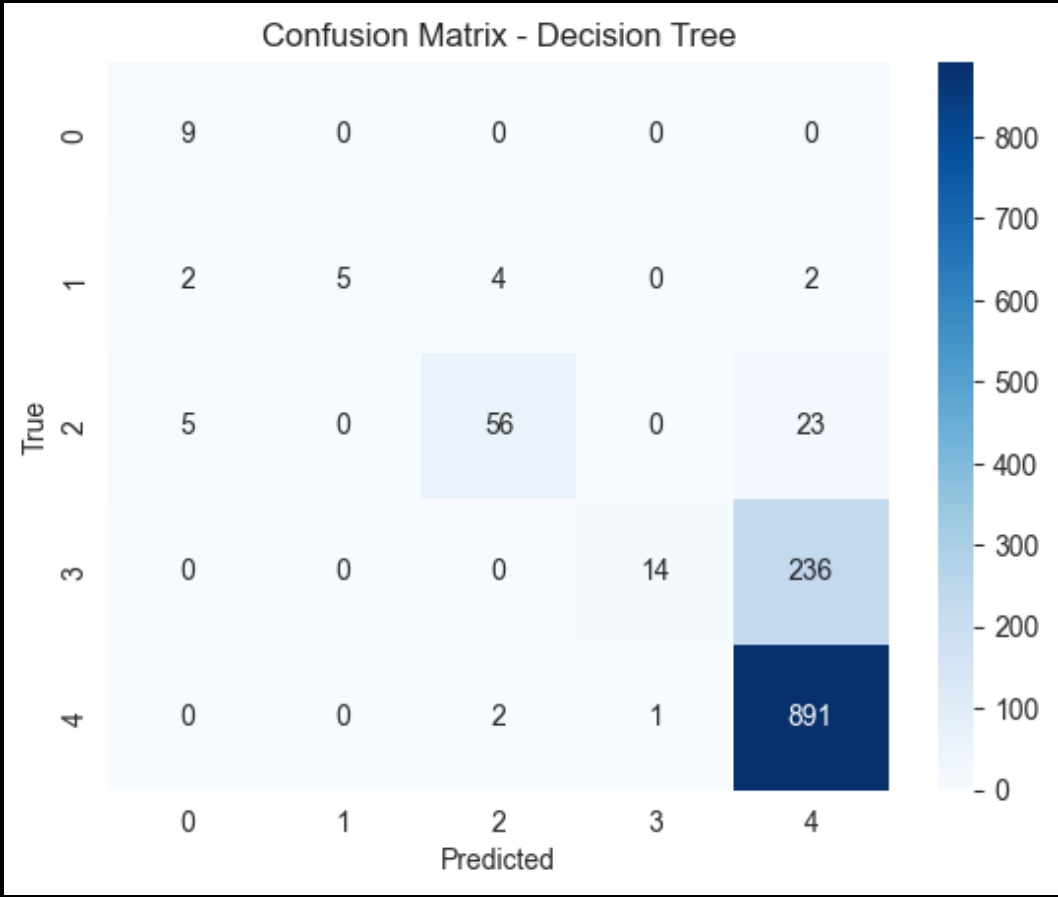
Find best hyperparameters for each model

پس از آن یک پیش بینی (predictions) بر روی داده های تست خود برای هر سه مدل پیاده سازی کرده و پس از آن توسط معیار های ارزیابی مذکور محاسبه می‌کنیم.

نتایج حاصل به شرح زیر است:

```
Decision Tree - train_acc: 0.78
Decision Tree - Accuracy: 0.78
Decision Tree - Precision: 0.8149759408602151
Decision Tree - Recall: 0.78
Decision Tree - F1 Score: 0.7065606672132407
```

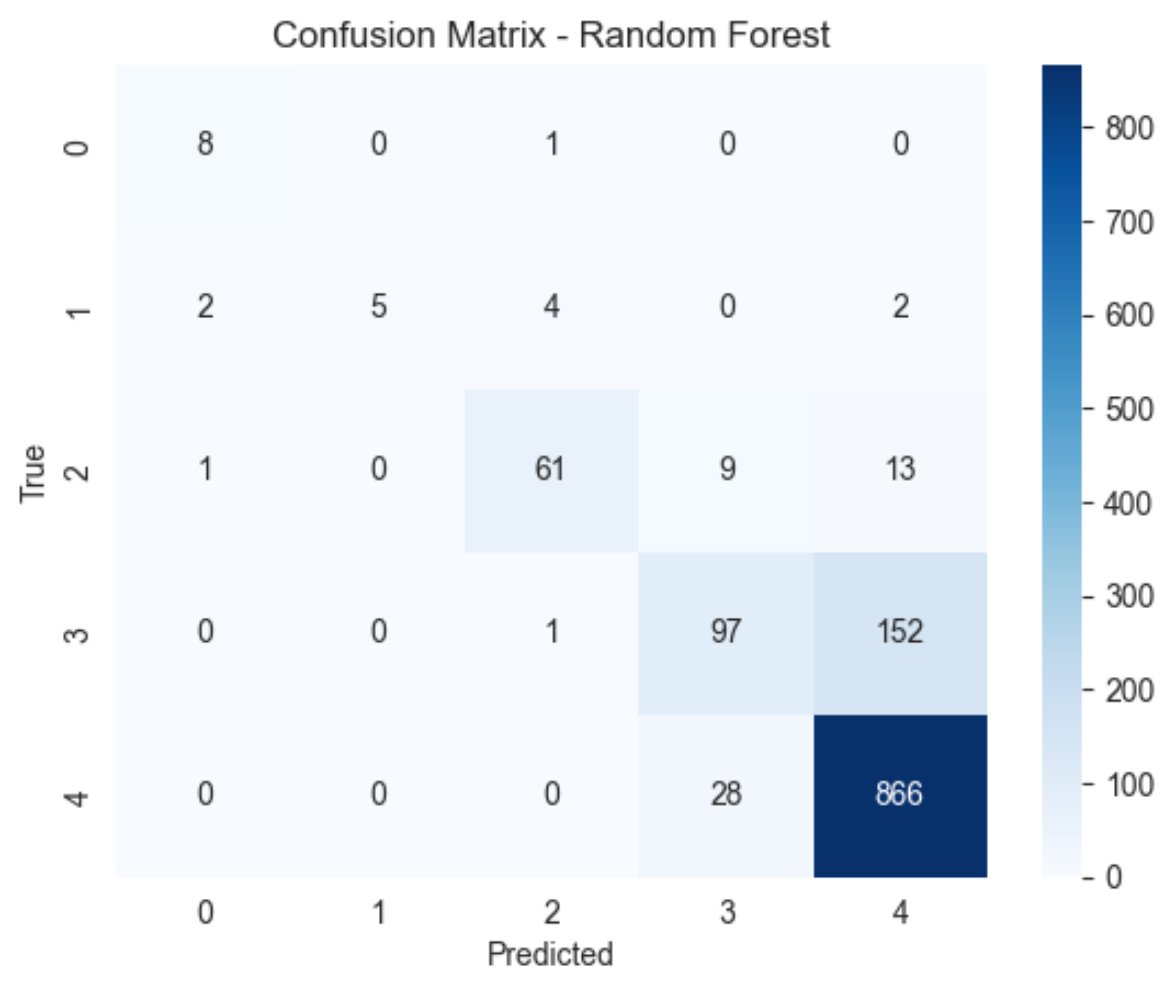
تصویر ۷: نتایج ارزیابی Decision Tree



تصویر ۸: ماتریس گمراهی Decision Tree

```
Random Forest - train_acc: 0.8296
Random Forest - Accuracy: 0.8296
Random Forest - Precision: 0.8211717265122072
Random Forest - Recall: 0.8296
Random Forest - F1 Score: 0.8096998464006129
```

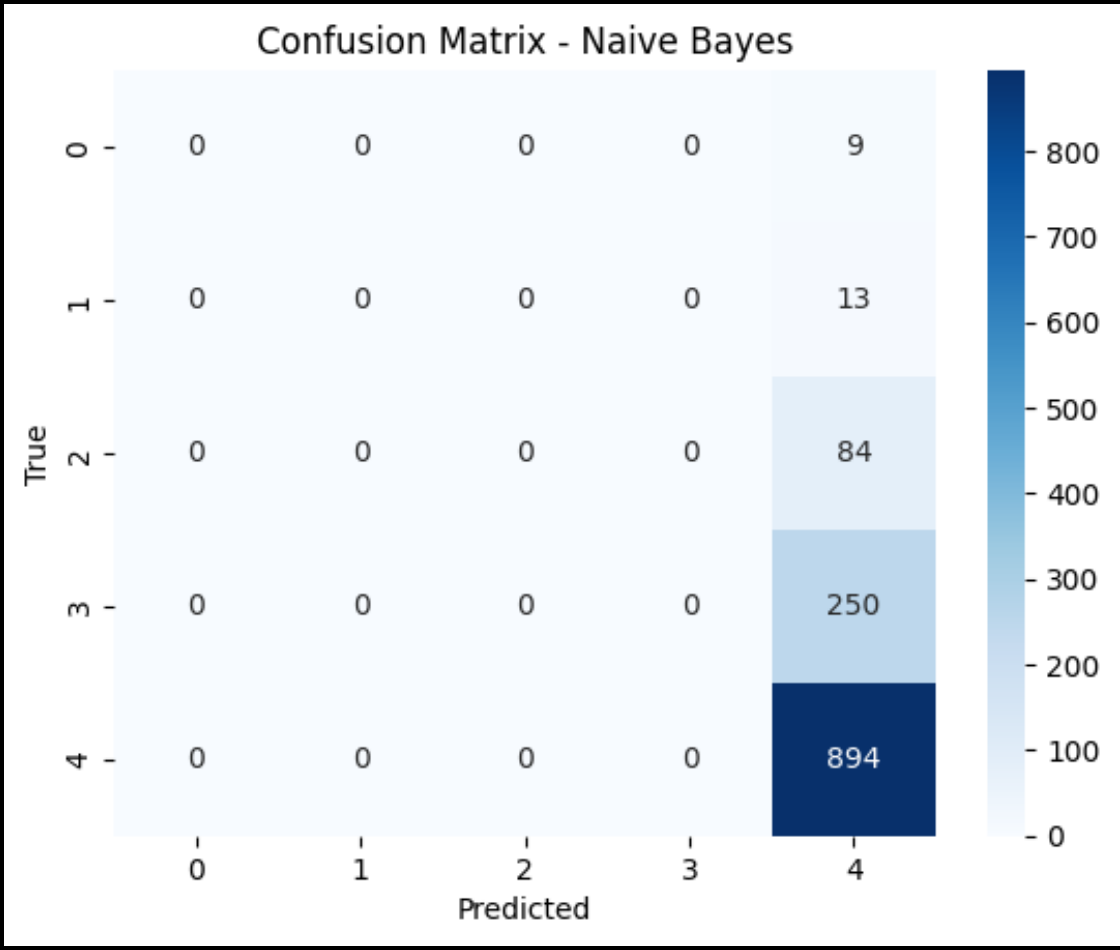
تصویر ۹: : نتایج ارزیابی random forest



تصویر ۱۰ : ماتریس گمراهی Random Forest

```
Naive Bayes - Accuracy: 0.7152
Naive Bayes - Precision: 0.5115110399999999
Naive Bayes - Recall: 0.7152
Naive Bayes - F1 Score: 0.596444776119403
```

تصویر ۱۱ : : نتایج ارزیابی Navie Bayes



تصویر ۱۲ : ماتریس گمراهی Navie Bayes

نتایج حاصل نشان دهنده ی این است که **Random forest** بهترین گزینه و **Navie Bayes** بدترین گزینه برای انجام طبقه بندی می‌باشد.