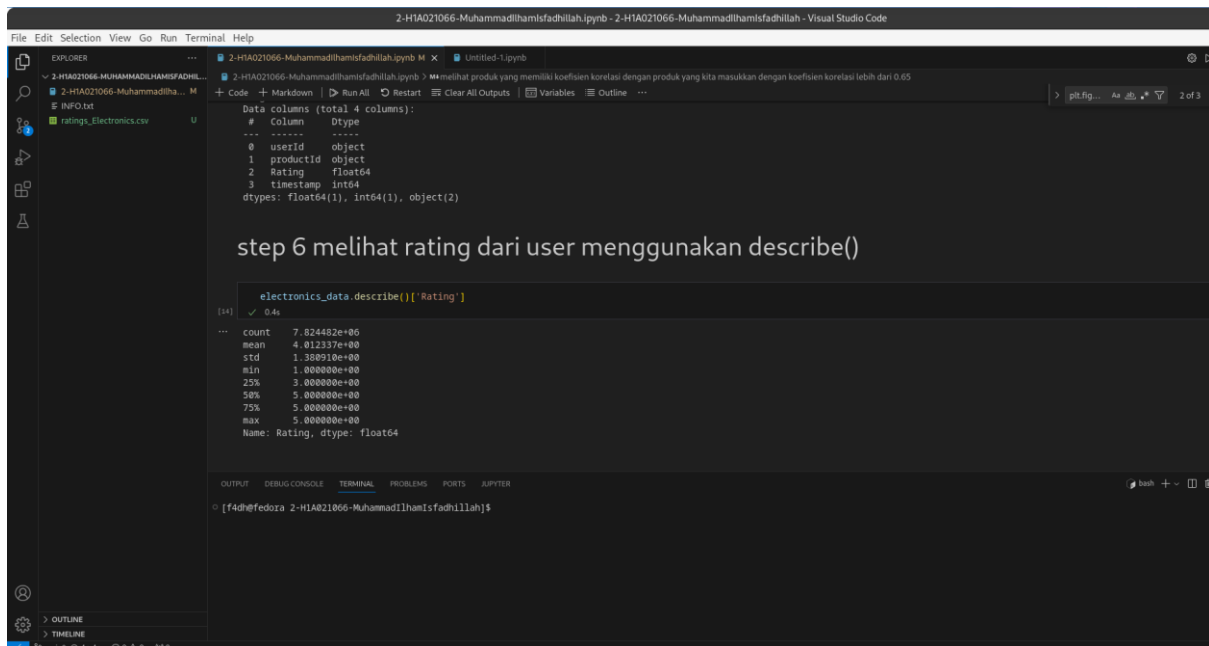


Kemudian, selain itu juga kita dapat melihat berapa baris dan kolom yang tersedia pada dataset kita, yaitu dengan cara menggunakan method `shape()` pada variabel dataset kita, dan akan tertampil untuk indeks ke 0 berupa jumlah baris dan indeks ke 1 berupa jumlah kolom. Kemudian kita juga dapat melihat tipe data dari setiap data kolom dengan menggunakan `dtypes()`, sehingga ditampilkan tipe data dari kolom `userID`, `productID`, `Rating` dan juga `timestamp`. Kemudian untuk melihat informasi umum tentang variabel dataset kita, kita menggunakan method `info()`, sehingga akan dilihat secara detail informasi dataset yang kita yang berada pada `DataFrame`. Kemudian, jika kita ingin menghapus beberapa informasi dari method `info()`, kita dapat menggunakan memberikan nilai `False` dalam parameter mengenai data yang tidak ingin kita tampilkan. Kemudian, dengan bantuan library `pandas` juga kita dapat melihat informasi lebih detail mengenai kolom tertentu. Misal kita ingin mengetahui tentang kolom `Rating`, maka dengan method `describe()` kita dapat melihatnya, yaitu sebagai berikut



```
File Edit Selection View Go Run Terminal Help
2-HIA021066-Muhammadiyahfadhilah.ipynb - 2-HIA021066-Muhammadiyahfadhilah - Visual Studio Code

EXPLORER
2-HIA021066-Muhammadiyahfadhilah.ipynb
INFO.txt
ratings_Electronics.csv

Data columns (total 4 columns):
# Column Dtype
---
0 userid object
1 productid object
2 Rating float64
3 timestamp int64
dtypes: float64(1), int64(1), object(2)

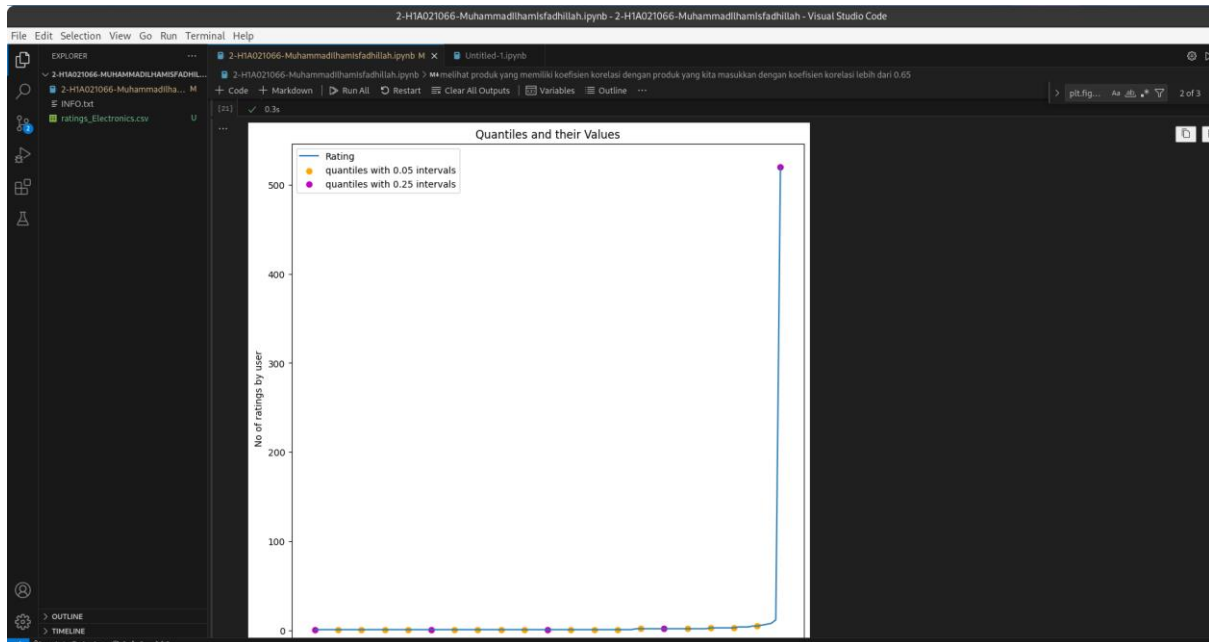
step 6 melihat rating dari user menggunakan describe()

electronics_data.describe()['Rating']
[14] ✓ 0.4s
...
count 7.824482e+06
mean 4.812337e+00
std 1.388910e+00
min 1.000000e+00
25% 3.000000e+00
50% 5.000000e+00
75% 5.000000e+00
max 5.000000e+00
Name: Rating, dtype: float64

OUTPUT DEBUG CONSOLE TERMINAL PROBLEMS PORTS JUPYTER
[f4dhe@fedora 2-HIA021066-Muhammadiyahfadhilah]$
```

Gambar 2 menunjukkan informasi dari kolom rating menggunakan describe()

Kemudian pada tahapan ini juga kita dapat melihat nilai tertinggi dan nilai terendah dari kolom Rating dengan cara menambahkan method min() dan max() , yang mana ini sangat berguna jika kita ingin mengetahui secara cepat berapakah nilai terbesar dan terkecil dibandingkan kita diharuskan untuk ngescroll dan menceknya satu satu. Kemudian juga , kita dapat mengecek apakah terdapat parameter yang null dan akan kita dapat menggunakan sum() untuk menghitung jumlah yang null jika ada. Ini merupakan tahapan awal kita untuk mengecek apakah ada data yang null sebelum data tersebut kita proses dalam bentuk grafik untuk menghindari error yang terjadi. Kemudian untuk mengecek ada berapa sih data mengenai kolom kolom yang ada di dataset kita , seperti berapa rating yang ada, berapa produk yang ada dan berapa user yang ada di dataset kita, kita dapat menggunakan len(np.unique()) dari kolom yang ingin kita coba cek. Selain itu juga, semisal dalam sebuah dataset terdapat terlalu banyak informasi yang tidak kita butuhkan, maka kolom data tersebut dapat kita hapusnya menggunakan method drop(), seperti pada percobaan ini , kita menghapus kolom timestamp karena kedepannya tidak terlalu berfungsi saat kita coba olah dalam bentuk grafik. Kemudian, kita juga dapat membuat sebuah pengelompokkan dalam dataset kita, semisal kita ingin menampilkan salah satu kolom dalam dataset kita, kita dapat menggunakan method groupby(). Selain itu, disini kita juga dapat menampilkan sebuah dataset yang data kolomnya telah disortir sesuai kemauan kita dengan menggunakan sort\_values(). Selain itu, kita juga dapat mendapatkan nilai kuantil dari data kolom di dataset kita menggunakan method quantile().



Gambar 3 Quantiles dari data

## 2.2.2 Recommending Products Based on the Product Popularity

Kemudian, pada percobaan selanjutnya kita mulai masuk mencoba untuk membuat sebuah rekomendasi produk berdasarkan popularitas produk yang ada di dataset kita. Langkah yang perlu kita lakukan disini adalah melakukan penyortiran produk kita, kita akan menyortir sebuah produk yang memiliki Rating lebih dari dan sama dengan 50 kali dan kemudian menyortirnya berdasarkan Rating terendah terlebih dahulu dan melakukan plotting grafiknya. Kemudian, kita dapat melakukan penyortiran berdasarkan rata rata rating, jadi disini kita membuat sebuah tabel yang terdiri dari sebuah produk dan rata rata ratingnya dengan menggunakan groupby() dan mean() atau menyortir produk berdasarkan produk yang sering dirating yang nanti hasil akhirnya dapat kita sortir untuk menjadi lebih rapih.

```

# calculate the average rating of each product
new_df.groupby('productId')['Rating'].mean().head()

...
productId
0972683275    4.470988
1400501466    3.560000
1400501520    4.243902
1400501776    3.884892
1400532020    3.684211
Name: Rating, dtype: float64

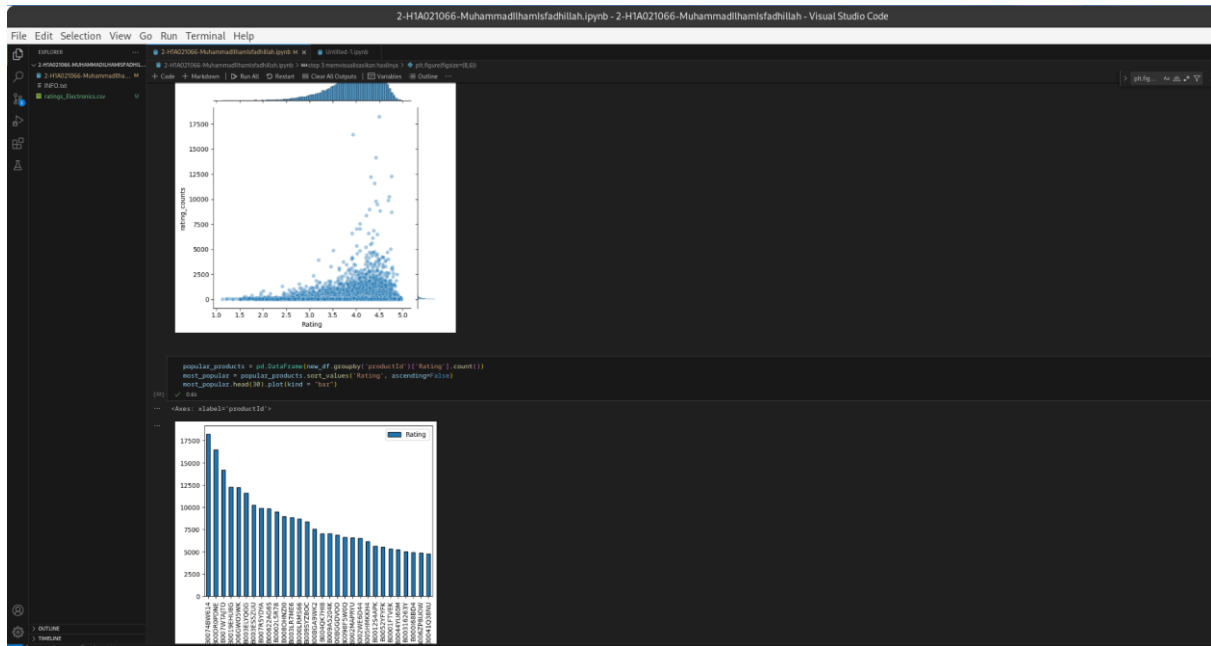
# Obtain the rankings of the products sorted by the number of rating times.
new_df.groupby('productId')['Rating'].count().sort_values(ascending=False).head()

...
productId
B0074BWS14    18244
B000R0P0NE    10454
B007ATATJ0    14172
B0019EHUS6    12285
B006GWS0MK    12226
Name: Rating, dtype: int64

```

Gambar 4 pensortiran produk berdasarkan rata rata dari rating

Kemudian, kita juga dapat menambahkan kolom baru dalam dataset yaitu dengan menambahkan jumlah rating / rating\_counts dalam pada tabel dataset. Setelah itu, kita dapat memvisualisasikan hasilnya menggunakan hist() , yang mana parameter di dalamnya dapat kita ganti nilai bins nya.



Gambar 5 membentuk histogram dengan bins = 50 dari data frame

### 2.2.3 Recommending Products Based on Collaborative Filtering

Kemudian, pada percobaan terakhir barulah kita membuat sebuah rekomendasi produk berdasarkan pada Filtering Kolaboratif, dimana tahapannya disini kita perlu memilih 10000 sampel dan menggunakan method pivot\_table() untuk membuat sebuah hubungan antara productID dan userID dan di dalamnya kita dapat memberikan default value pada kolom yang berupa NaN dengan mengubah nilai fill value pada parameter pivot\_table() seperti pada gambar berikut. Selain itu kita juga dapat menukar kolom dan baris pada tabel dengan menggunakan object attributes T. Selanjutnya, pada percobaan ini digunakan algoritma Singular Value Decomposition (SVD) untuk mereduksi dimensi tabel tersebut dan mengidentifikasi fitur-fitur produk yang paling penting (10 fitur dalam contoh ini). Dengan mereduksi dimensi, data menjadi lebih ringkas namun masih mempertahankan informasi yang relevan. Langkah berikutnya adalah membangun matriks korelasi antara produk dengan corrcoeff().

```

2-HIA021066-MuhammadHamsifadhilah.ipynb - 2-HIA021066-MuhammadHamsifadhilah - Visual Studio Code

step 2 mendekompos tabel menggunakan algoritma SVC dari library klearn

from sklearn.decomposition import TruncatedSVD # Import the SVD algorithm.
SVD = TruncatedSVD(n_components=10) # Construct an SVD model to combine the number of features (that is, the number of columns) into 10 important combined features.
decomposed_matrix = SVD.fit_transform(X) # Transform the table.
decomposed_matrix.shape # View the size of the table after conversion.
[27]: 0.0s
... (76, 10)

step 3 mencari kesamaan/similarities dari produk menggunakan corrcoeff()

correlation_matrix = np.corrcoef(decomposed_matrix)
correlation_matrix.shape
[28]: 0.0s
... (76, 76)

step 4 merekomendasikan produk berdasarkan kesamaan produk

X.index[20] # Select the 20th product.
[29]: 0.0s
... '9984984354'

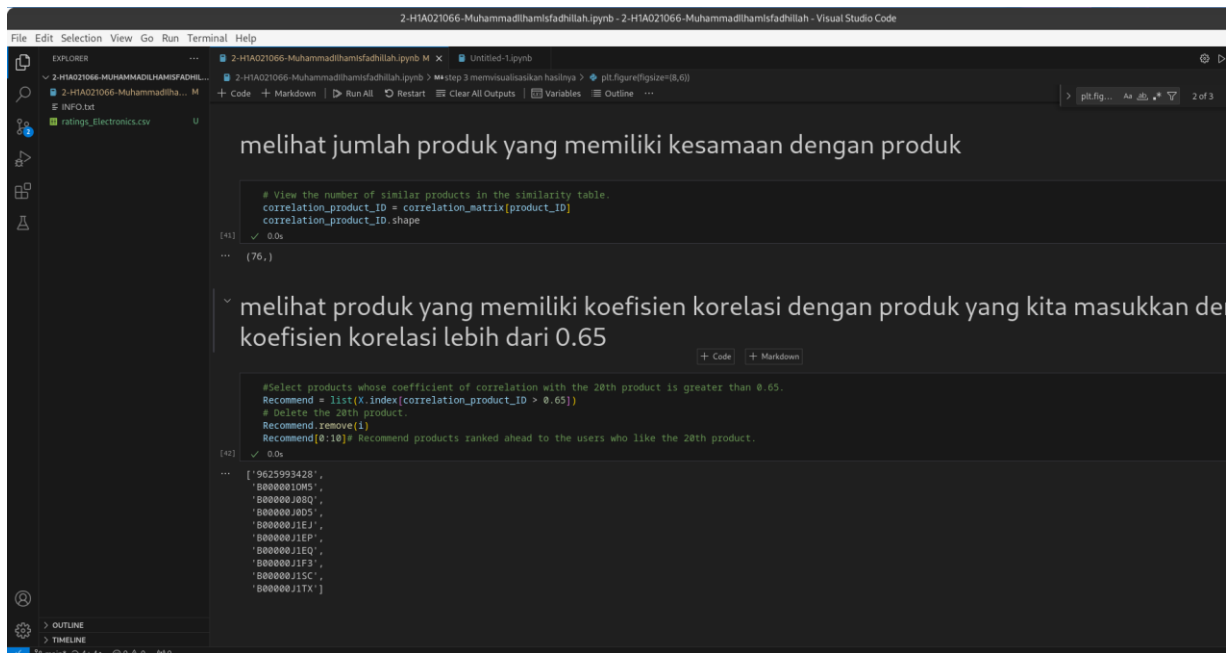
# Determine whether the product is unique.
i = '9984984354'
product_names = list(X.index)
recomendation = product_names[indices[i]]

```

Gambar 6 mendekompos tabel menggunakan SVD dan corrcoeff()

Matriks ini mengukur kesamaan antara produk berdasarkan pola rating pengguna. Semakin tinggi korelasi antara dua produk, semakin mirip mereka dalam hal preferensi pengguna. Dalam hal ini, kita

dapat merekomendasikan produk-produk kepada pengguna berdasarkan kesamaan produk yang telah mereka sukai sebelumnya. Rekomendasi yang dihasilkan didasarkan pada pola rating produk oleh pengguna lain yang memiliki preferensi serupa, seperti pada gambar berikut



The screenshot shows a Jupyter Notebook with two cells. The first cell contains code to view the shape of the correlation matrix for a specific product ID. The second cell contains code to select products with a correlation coefficient greater than 0.65 and to delete the 20th product from the recommendation list.

```
# View the number of similar products in the similarity table.
correlation_product_ID = correlation_matrix[product_ID]
correlation_product_ID.shape

[41]: 0.0s
(76,)
```

melihat produk yang memiliki koefisien korelasi dengan produk yang kita masukkan de koefisien korelasi lebih dari 0.65

```
#Select products whose coefficient of correlation with the 20th product is greater than 0.65.
Recommend = list(X.index[correlation_product_ID > 0.65])
# Delete the 20th product.
Recommend.remove(1)
Recommend[0:10]# Recommend products ranked ahead to the users who like the 20th product.

[42]: 0.0s
['9625993428',
 'B0000010M5',
 'B0000010BQ',
 'B0000010D5',
 'B0000011E',
 'B0000011EP',
 'B0000011EQ',
 'B0000011F3',
 'B0000011SC',
 'B0000011TK']
```

Gambar 7 mencari produk serupa dengan koefisien korelasi lebih dari 0.65