Trường ĐH Khoa Học Tự Nhiên Tp. Hồ Chí Minh
TRUNG TÂM TIN HỌC

Đồ án tốt nghiệp Data Science

Topic: ***Recommender System***

GVHD: Cô Khuất Thùy Phương

HVTH: Đỗ Thị Phượng

2022

Project Overview

❑ Business Objective/Problem

- Tiki là một hệ sinh thái thương mại "all in one", trong đó có tiki.vn, là một website thương mại điện tử đứng top 2 của Việt Nam, top 6 khu vực Đông Nam Á.
- Trên trang này đã triển khai nhiều tiện ích hỗ trợ nâng cao trải nghiệm người dùng và họ muốn xây dựng nhiều tiện ích hơn nữa.
- Giả sử công ty này chưa triển khai Recommender System và bạn được yêu cầu triển khai hệ thống này, bạn sẽ làm gì?

Project Overview

Product	
Item_id	Mã sản phẩm
Name	Tên sản phẩm
Description	Mô tả tính năng sản phẩm
Rating	Điểm đánh giá chung sản phẩm
Price	Giá bán sản phẩm
List_Price	Giá sản phẩm (chưa giảm giá hoặc khuyến mãi)
Brand	Thương hiệu
Group	Nhóm phân loại sản phẩm

Review	
Customer_id	Mã khách hàng
Product_id	Mã sản phẩm
Name	Tên tài khoản
Full_name	Tên khách hàng
Created_time	Thời điểm phản hồi được tạo
Rating	Điểm đánh giá sản phẩm của khách hàng
Title	Tiêu đề đánh giá
Content	Nội dung đánh giá

Preprocessing

- Dữ liệu do giảng viên cung cấp bao gồm 2 files “ProductRaw.csv” và “ReviewRaw.csv” chứa thông tin sản phẩm, review và rating cho các sản phẩm thuộc các nhóm hang hóa như Mobile_Tablet, TV_Audio, Laptop, Camera, Accessory
- Dữ liệu về products có 4.404 dòng là thông tin về 4.404 sản phẩm
- Dữ liệu về reviews bao gồm 364.099 đánh giá sản phẩm

```
products.head()
```

	item_id	name	description	rating	price	list_price	brand	group	url	image
0	48102821	Tai nghe Bluetooth Inpods 12 - Cảm biến vân ta...	THÔNG TIN CHI TIẾT\nDung lượng pin 300\nThời g...	4.0	77000	300000	OEM	Thiết Bị Số - Phụ Kiện Số/Thiết Bị Âm Thanh và...	https://tai-nghe-bluetooth-inpods-12-cam-bien-...	https://salt.tikicdn.com/cache/280x280/ts/prod...
1	52333193	Tai nghe bluetooth không dây F9 True wireless ...	THÔNG TIN CHI TIẾT\nDung lượng pin 2000mah\nTh...	4.5	132000	750000	OEM	Thiết Bị Số - Phụ Kiện Số/Thiết Bị Âm Thanh và...	https://tai-nghe-bluetooth-khong-day-f9-true-w...	https://salt.tikicdn.com/cache/280x280/ts/prod...

```
reviews.head()
```

	customer_id	product_id	name	full_name	created_time	rating	title	content
0	709310	10001012	Lân Nguyễn Hoàng	Lân Nguyễn Hoàng	NaN	3	Ko dùng đc thẻ nhớ	Lúcđầu quên thông tin nên dùng 512gb thì ko đc...
1	10701688	10001012	Nguyễn Khánh Hòa	Nguyễn Khánh Hòa	NaN	5	Cực kì hài lòng	Tiki giao hàng nhanh. Sản phẩm đúng như mô tả,...

Preprocessing

- Products:
- Loại bỏ dữ liệu trùng, còn lại 4.373 dòng.
- Dữ liệu null lại description (3 records), các cột khác không null

```
products.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4404 entries, 0 to 4403
Data columns (total 10 columns):
 #   Column       Non-Null Count  Dtype  
---  ------       --------------  -----  
 0   item_id      4404 non-null   int64  
 1   name         4404 non-null   object 
 2   description  4401 non-null   object 
 3   rating       4404 non-null   float64
 4   price        4404 non-null   int64  
 5   list_price   4404 non-null   int64  
 6   brand        4404 non-null   object 
 7   group        4404 non-null   object 
 8   url          4404 non-null   object 
 9   image        4404 non-null   object 
dtypes: float64(1), int64(3), object(6)
memory usage: 344.2+ KB
```

```
# loại bỏ DL trùng:
print('Trước khi drop, số sản phẩm là: ', products.shape[0])
products = products.drop_duplicates()
print('Sau khi drop, số sản phẩm là: ', products.shape[0])
```

```
Trước khi drop, số sản phẩm là:  4404
Sau khi drop, số sản phẩm là:  4373
```

```
# kiểm tra dữ liệu null:
products.isnull().sum()
```

```
item_id        0
name           0
description    3
rating         0
price          0
list_price     0
brand          0
group          0
url            0
image          0
dtype: int64
```

Preprocessing

- Reviews:
- Loại bỏ dữ liệu trùng, còn lại 361.750 dòng.

```
reviews.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 364099 entries, 0 to 364098
Data columns (total 8 columns):
 #   Column        Non-Null Count   Dtype
---  ------        --------------   -----
 0   customer_id   364099 non-null  int64
 1   product_id    364099 non-null  int64
 2   name          363669 non-null  object
 3   full_name     329471 non-null  object
 4   created_time  117340 non-null  object
 5   rating        364099 non-null  int64
 6   title         364070 non-null  object
 7   content       165794 non-null  object
dtypes: int64(3), object(5)
memory usage: 22.2+ MB
```

```
# loại bỏ DL trùng:
print('Trước khi drop, records = ', reviews.shape[0])
reviews.drop_duplicates(inplace = True)
print('Sau khi drop, records = ', reviews.shape[0])
```

```
Trước khi drop, records =  364099
Sau khi drop, records =  361750
```

```
# kiểm tra dữ liệu null:
reviews.isnull().sum()
```

```
customer_id          0
product_id           0
name               428
full_name        34603
created_time    245525
rating               0
title               29
content         196064
dtype: int64
```

Preprocessing

- Dữ liệu null nhiều ở created_time, content.
- Cột title có giá trị null nhưng số lượng ít.
- name và full_name (nếu có) khá giống nhau. full_name có dữ liệu null tương đối (9.5%)
- Loại bỏ cột full_name và created_time

	feature	Missing_Value	Percentage
0	name	428	0.118314
1	full_name	34603	9.565446
2	created_time	245525	67.871458
3	content	196064	54.198756
4	title	29	0.008017

name	full_name
Lân Nguyễn Hoàng	Lân Nguyễn Hoàng
Nguyễn Khánh Hòa	Nguyễn Khánh Hòa
Toàn Phạm Khánh	Toàn Phạm Khánh
Nguyen Quang Minh	NaN
Phạm Bá Đức	Phạm Bá Đức

Preprocessing

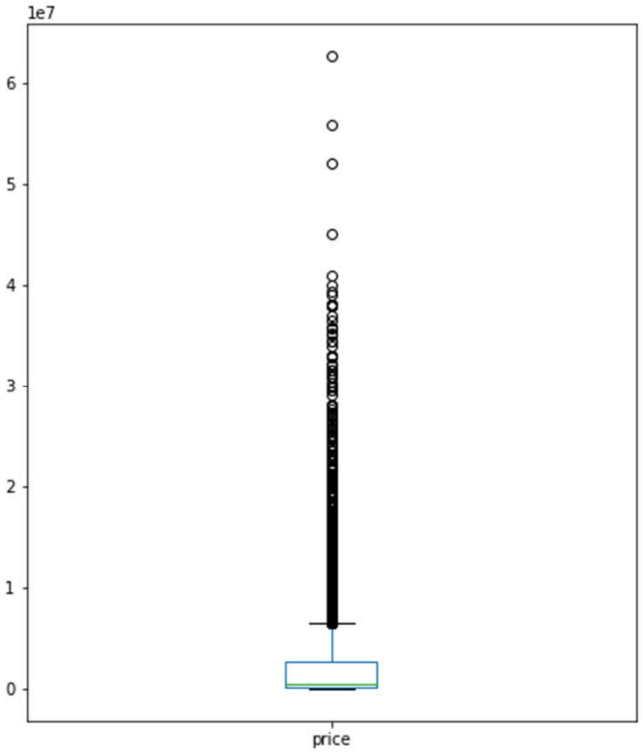
- Dữ liệu null nhiều ở created_time, content.
- Cột title có giá trị null nhưng số lượng ít.
- name và full_name (nếu có) khá giống nhau. full_name có dữ liệu null tương đối (9.5%)
- Loại bỏ cột full_name và created_time
- Loại bỏ 660 reviews có product_id không tồn tại trong bảng products còn lại 361090 đánh giá cho 4214 sản phẩm

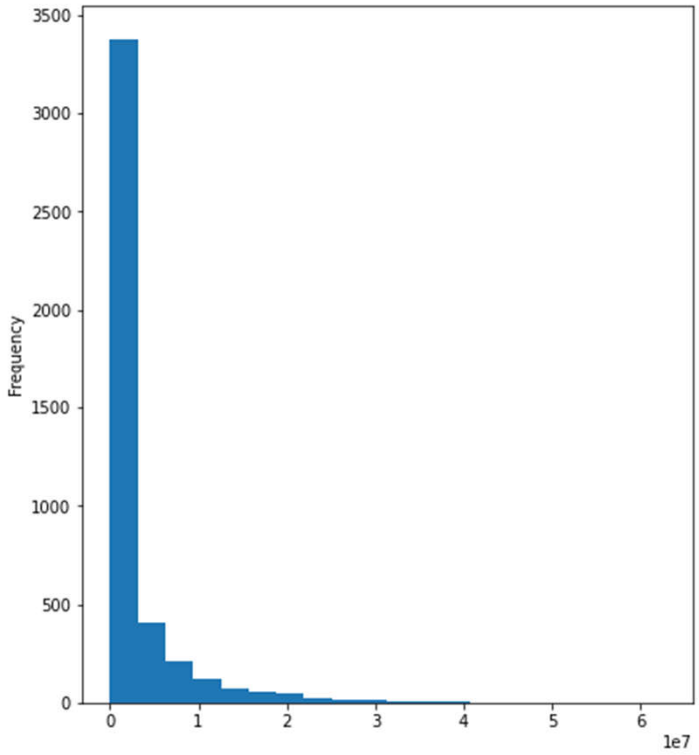
	feature	Missing_Value	Percentage
0	name	428	0.118314
1	full_name	34603	9.565446
2	created_time	245525	67.871458
3	content	196064	54.198756
4	title	29	0.008017

name	full_name
Lân Nguyễn Hoàng	Lân Nguyễn Hoàng
Nguyễn Khánh Hòa	Nguyễn Khánh Hòa
Toàn Phạm Khánh	Toàn Phạm Khánh
Nguyen Quang Minh	NaN
Phạm Bá Đức	Phạm Bá Đức

EDA

	count	mean	std	min	25%	50%	75%	max
price	4,373.00	2,763,501.13	5,544,076.83	7,000.00	150,000.00	487,000.00	2,680,000.00	62,690,000.00
list_price	4,373.00	3,893,684.92	7,900,791.25	12,000.00	279,000.00	790,000.00	3,590,000.00	82,990,000.00



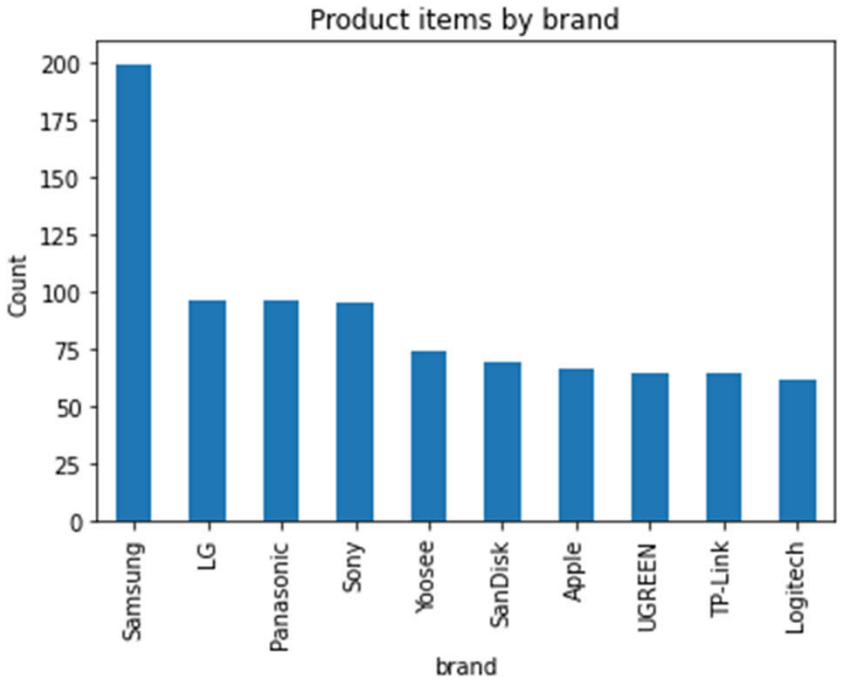


- **Products**:
- Giá sp có range rất rộng trong khoảng 7000-62.690.000
- Giá gốc sp có range rất rộng trong khoảng 12.000-82.990.000
- Phần lớn giá sp tập trung < 3.000.000

EDA

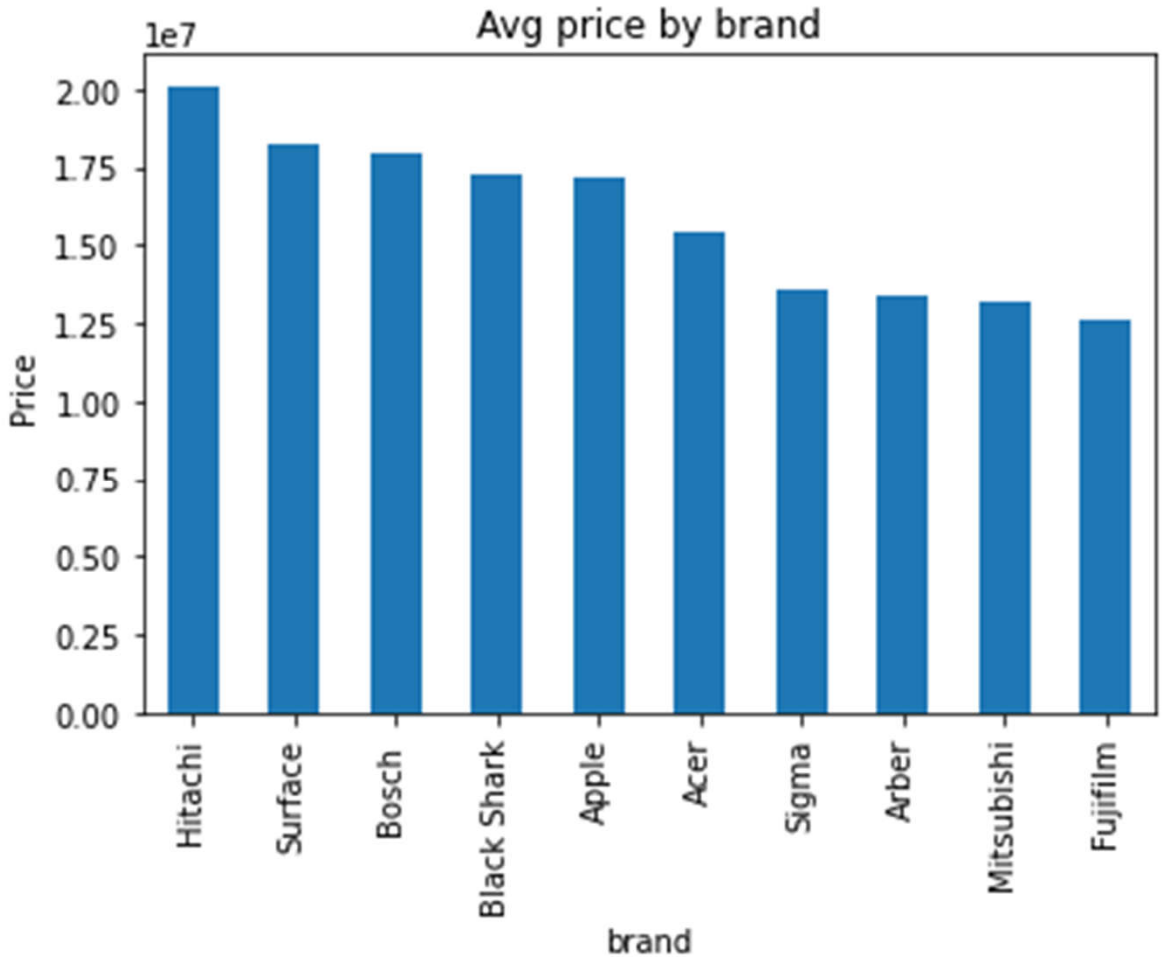
- Ngoại trừ OEM có số lượng mã sp vượt trội (1115) thì trong top 10 có Samsung có số lượng mã sp nhiều nhất các thương hiệu khác có số lượng mã sp tương đương nhau.

```
brand
OEM              1115
Samsung           199
LG                 96
Panasonic          96
Sony               95
                  ...
KTV                 1
KBVISION-USA        1
KAPUSI              1
Joy Collection      1
xMOWI               1
Name: item_id, Length: 521, dtype: int64
```



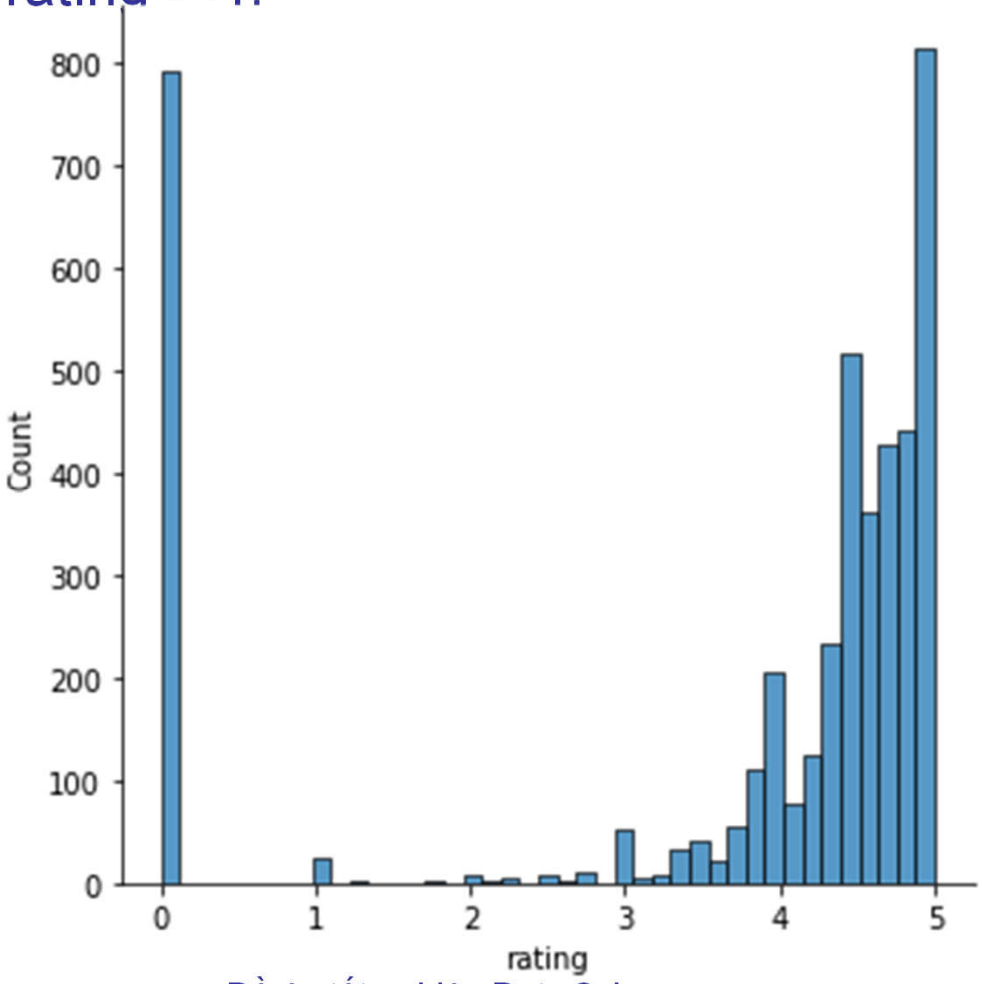
EDA

- Về giá bán thì thương hiệu Hitachi có trung bình giá bán sản phẩm cao nhất.



EDA

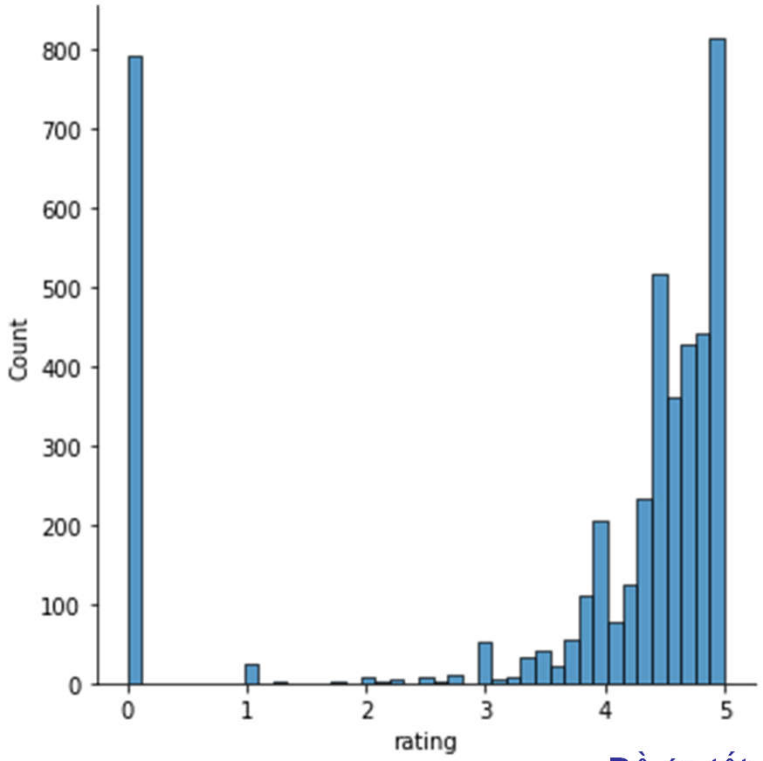
- Rating sp trong khoảng 0-5
- Sp có rating là 0 và 5 là tương đương nhau và có số lượng khá lớn
- Phần lớn sp có rating > 4.

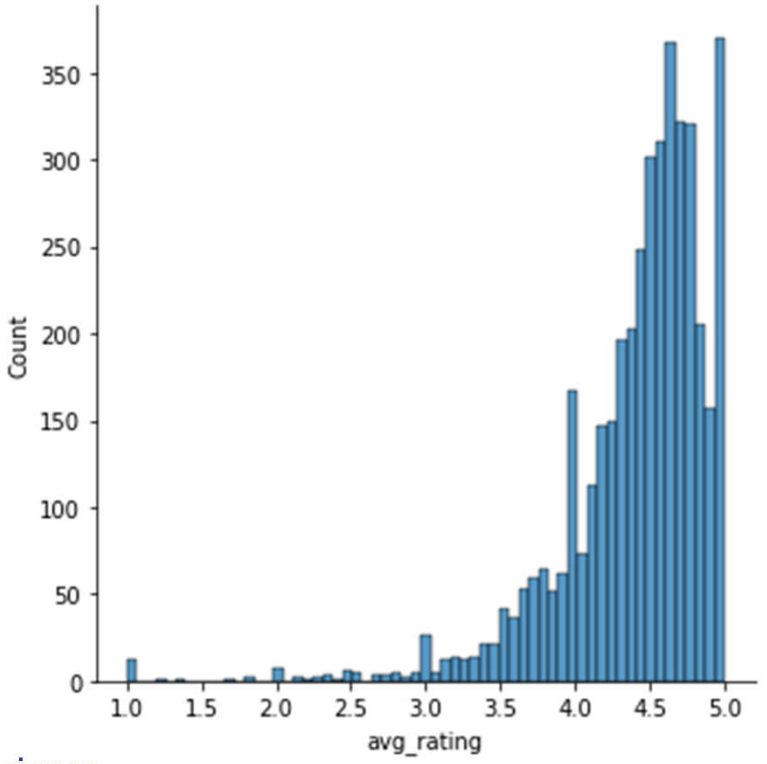


EDA

	product_id	avg_rating
0	54665	4.60
1	55897	4.63
2	104180	4.48
3	116897	4.24
4	122012	4.49

- Xem xét rating trong review của KH
- Rating của sp trong review của KH >0. Có thể kết luận điểm rating = 0 trong product là do thiếu dữ liệu.

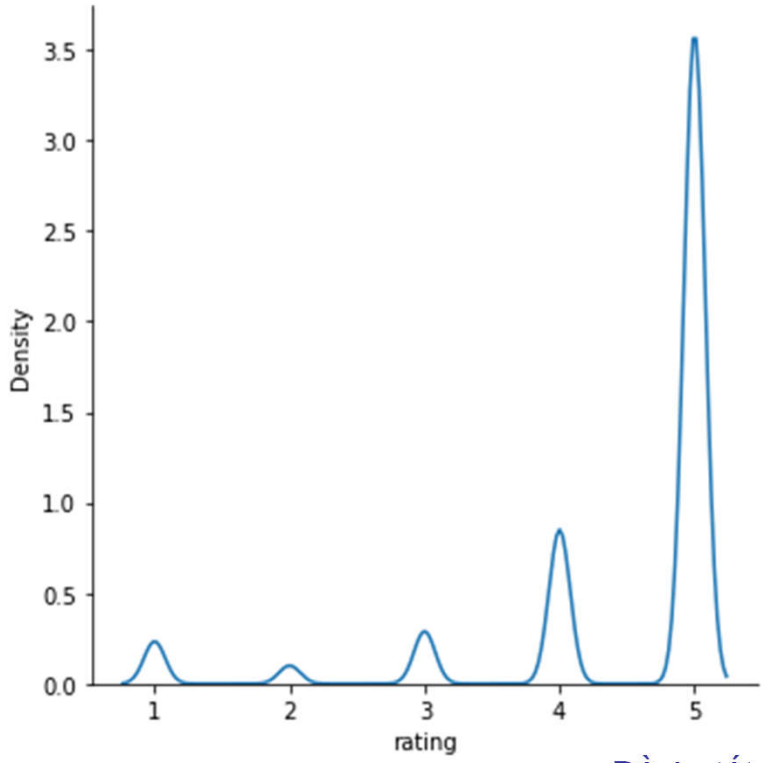




EDA

Reviews:

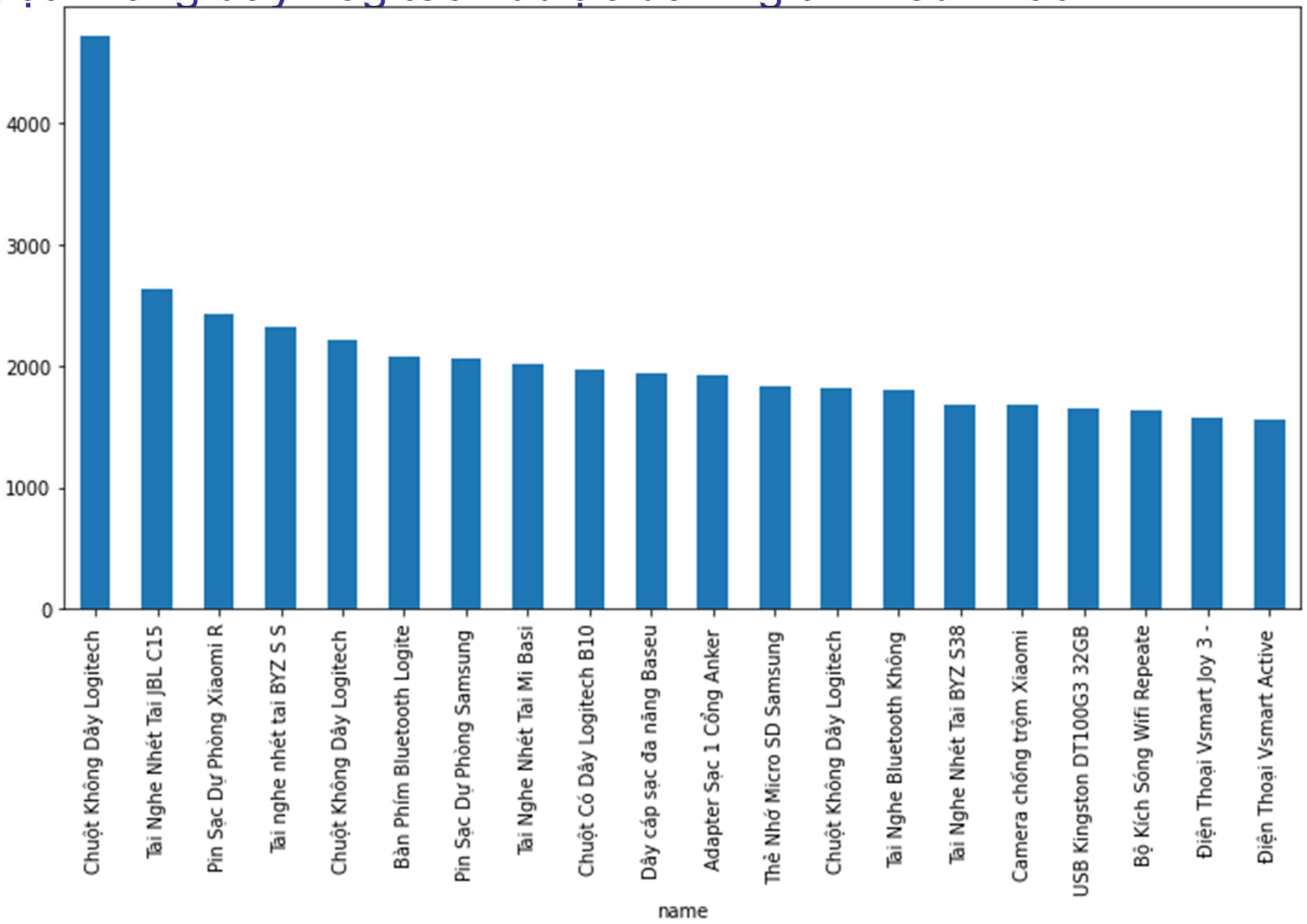
- Phần lớn KH phản hồi tích cực về sp
- Nguyên nhân: SP có chất lượng tốt hoặc do KH dễ tính??
- Phần lớn đánh giá cho rating 5




```
rating
1     16616
2      7098
3     20600
4     60565
5    256211
dtype: int64
```

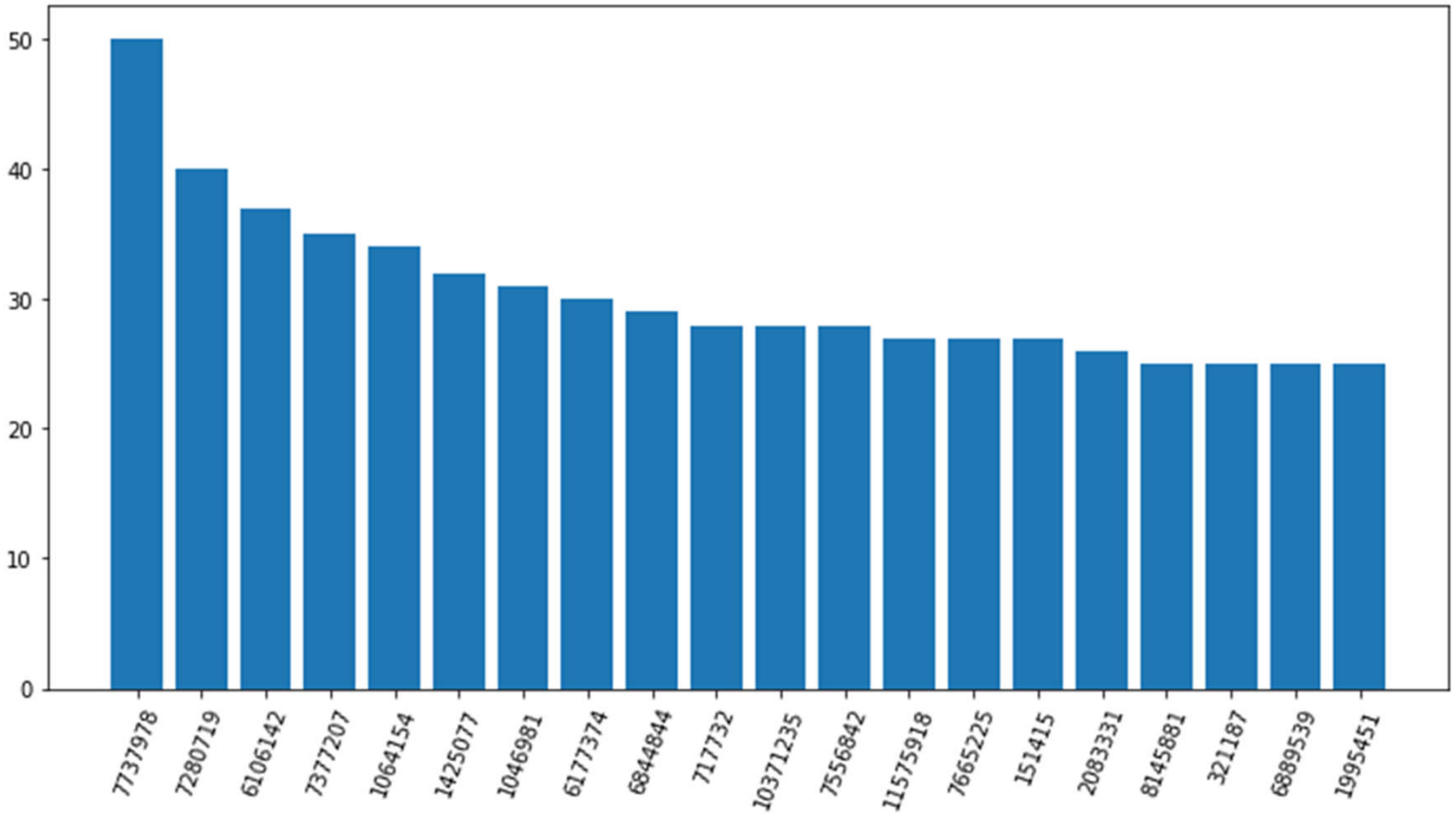
EDA

- Các sp được đánh giá nhiều nhất là phụ kiện điện thoại, máy tính
- Chuột không dây Logitech được đánh giá nhiều nhất.



EDA

- KH có mã số 7737978 thực hiện nhiều đánh giá nhất (cho 50 sp)
- Top 20 chủ yếu thực hiện đánh giá cho khoảng 30-40 sp.



EDA

```
# Lựa chọn thuộc tính cần thiết cho bài toán:
products = products[['item_id', 'name', 'description']]
reviews = reviews[['id', 'customer_id', 'product_id', 'rating']]
```

	item_id	name	description
0	48102821	Tai nghe Bluetooth Inpods 12 - Cảm biến vân ta...	THÔNG TIN CHI TIẾT\nDung lượng pin 300\nThời g...
1	52333193	Tai nghe bluetooth không dây F9 True wireless ...	THÔNG TIN CHI TIẾT\nDung lượng pin 2000mah\nTh...

id	customer_id	product_id	rating
0	709310	10001012	3
1	10701688	10001012	5
2	11763074	10001012	5
3	9909549	10001012	5
4	1827148	10001012	5

ContentBased_cosine

- Tạo cột name_description bằng cách nối name và description
- Sử dụng underthesea để xử lý text tiếng Việt. Sử dụng word_tokenize tạo column name_description_pre

```
products.head(2)
```

	item_id	name	description	name_description	name_description_pre
0	48102821	Tai nghe Bluetooth Inpods 12 - Cảm biến vân ta...	THÔNG TIN CHI TIẾT\nDung lượng pin 300\nThời g...	Tai nghe Bluetooth Inpods 12 - Cảm biến vân ta...	Tai_nghe Bluetooth_Inpods 12 - Cảm_biến vân ta...
1	52333193	Tai nghe bluetooth không dây F9 True wireless ...	THÔNG TIN CHI TIẾT\nDung lượng pin 2000mah\nTh...	Tai nghe bluetooth không dây F9 True wireless ...	Tai_nghe bluetooth không dây F9_True wireless ...

- Áp dụng Cosine similarities
- Sử dụng TF-IDF, kết hợp stopwords để tạo ma trận tfidf matrix

```
tf = TfidfVectorizer(analyzer= 'word', min_df = 0, stop_words=stop_words)
```

```
tfidf_matrix = tf.fit_transform(products.name_description_pre)
```

ContentBased_cosine

- Tạo array cosine_similarities từ tfidf_matrix. Gồm 4.370 dòng x 4.370 cột tương ứng với 4.370 sản phẩm. Mỗi giá trị c[i, j] là sự tương đồng giữa sản phẩm i và sản phẩm j

```
cosine_similarities = cosine_similarity(tfidf_matrix, tfidf_matrix)
```

```
cosine_similarities
```

```
array([[1.        , 0.35929855, 0.08313389, ..., 0.01771502, 0.0095737 ,
        0.05917742],
       [0.35929855, 1.        , 0.08321854, ..., 0.01520767, 0.02756314,
        0.08715747],
       [0.08313389, 0.08321854, 1.        , ..., 0.03603776, 0.02342419,
        0.06634264],
       ...,
       [0.01771502, 0.01520767, 0.03603776, ..., 1.        , 0.0063396 ,
        0.03202514],
       [0.0095737 , 0.02756314, 0.02342419, ..., 0.0063396 , 1.        ,
        0.04105841],
       [0.05917742, 0.08715747, 0.06634264, ..., 0.03202514, 0.04105841,
        1.        ]])
```

```
cosine_similarities.shape
```

```
(4370, 4370)
```

ContentBased_cosine

- Viết function recommendation_cosine.
- Input là id_view_product đang xem, products data (bao gồm item_id, name, description, name_description, name_description_pre); matrix_file là ma trận cosine_similarities
- Output là (5 sp tương tự với sản phẩm đang xem gồm score, item_id, name, name_description_pre) và text là ghép nối name_description_pre của 5sp này

```
def recommendation_cosine(id_view_product,data_file,matrix_file):
    df = data_file
    cosine_sim = matrix_file
    matrix = pd.DataFrame(cosine_sim,columns=df.item_id.values,index=df.item_id.values).reset_index()
    df_sim = matrix.loc[matrix[id_view_product]>=0.0,["index",id_view_product]].sort_values(id_view_product,ascending=False)
    result = pd.merge(df_sim, df, left_on="index",right_on="item_id")
    result.rename(columns={id_view_product:"score"},inplace=True)
    result = result[["score","item_id","name","name_description_pre"]].reset_index(drop=True).iloc[1:6,:]
    text = " ".join(content for content in result.name_description_pre)
    return result, text
```

ContentBased_cosine

- Ví dụ: Đề xuất 5 sản phẩm tương tự của sản phẩm đang xem có id là 48102821

item_id	name	description	name_description	name_description_pre
48102821	Tai nghe Bluetooth Inpods 12 - Cảm biến vân ta...	THÔNG TIN CHI TIẾT\nDung lượng pin 300\nThời g...	Tai nghe Bluetooth Inpods 12 - Cảm biến vân ta...	Tai_nghe Bluetooth_Inpods 12 - Cảm_biến vân ta...

```
id_view_product = 48102821
df_recommend_cosine, text = recommendation_cosine(id_view_product=id_view_product,
                            data_file = products,
                            matrix_file = cosine_similarities)

df_recommend_cosine
```

	score	item_id	name	name_description_pre
1	0.434966	56365197	Tai nghe bluetooth không dây i12 TWS 5.0, thiế...	Tai_nghe bluetooth không dây i12 TWS_5.0 , thi...
2	0.387165	22413470	Tai Nghe Bluetooth Air.podes Cảm Ứng Công Nghệ...	Tai_Nghe Bluetooth_Air ._podes Cảm_Ứng Công_Ng...
3	0.383320	50319688	Tai Nghe Bluetooth Mini I12 Tws V5.0 (Trắng) N...	Tai_Nghe Bluetooth Mini I12 Tws V5 ._0 (Trắng...
4	0.371535	72928043	Tai Nghe Bluetooth Amoi F9 kèm Củ Sạc 1A và Cá...	Tai_Nghe Bluetooth_Amoi_F9 kèm Củ_Sạc_1A và Cá...
5	0.370487	56885678	Tai Nghe Bluetooth TWS F9 Tai Nghe Nhét Hai T...	Tai_Nghe Bluetooth TWS F9 Tai_Nghe Nhét Hai Ta...

ContentBased_cosine

- Ví dụ: WordCloud 5 sản phẩm tương tự của sản phẩm đang xem có id là 48102821

ContentBased_Gensim

- Áp dụng Gensim

```
products.head(2)
```

	item_id	name	description	name_description	name_description_pre
0	48102821	Tai nghe Bluetooth Inpods 12 - Cảm biến vân ta...	THÔNG TIN CHI TIẾT\nDung lượng pin 300\nThời g...	Tai nghe Bluetooth Inpods 12 - Cảm biến vân ta...	Tai_nghe Bluetooth_Inpods 12 - Cảm_biến vân ta...
1	52333193	Tai nghe bluetooth không dây F9 True wireless ...	THÔNG TIN CHI TIẾT\nDung lượng pin 2000mah\nTh...	Tai nghe bluetooth không dây F9 True wireless ...	Tai_nghe bluetooth không dây F9_True wireless ...

```
# Tokenize the sentences into words
intro_products =  [[text for text in x.split()] for x in products.name_description_pre]
```

```
intro_products[:1]
```

```
[['Tai_nghe',
  'Bluetooth_Inpods',
  '12',
  '-',
  'Cảm_biến',
  'vân',
  'tay',
  ',',
  'chống',
  'nước',
  ',',
  'màu_sắc',
  'đa_dạng',
```

- Split từng từ trong name_description_pre thành 1 list chứa các từ riêng lẻ. intro_products bao gồm 4.370 list con tương ứng với từng sản phẩm

ContentBased_Gensim

- Loại bỏ ký tự không phải chữ (dấu câu, số,…), đưa về chữ thường lower, loại bỏ stopword, loại bỏ một số từ vô nghĩa,… được intro_products_re

```
intro_products_re[:1]
```

```
[['tai_nghe',
  'bluetooth_inpods',
  'cảm_biến',
  'vân',
  'tay',
  'chống',
  'nước',
  'màu_sắc',
  'đa_dạng',
  'màu_sắc',
  'lựa',
  'chọnthông',
  'chi_tiết',
  'dung_lượng',
  'pin',
  'pin',
  'nhạc',
  'liên_tục',
```

ContentBased_Gensim

```
dictionary = corpora.Dictionary(intro_products_re)
```

```
dictionary.token2id
```

```
{'airpod': 0,
 'apple': 1,
 'bao_gồm': 2,
 'bluetooth': 3,
 'bluetooth_.': 4,
 'bluetooth_inpods': 5,
 'bấm': 6,
 'chi_tiết': 7,
 'chuẩn': 8,
 'chạm': 9,
 'chất': 10,
 'chọnthông': 11,
 'chống': 12,
 'chờ': 13,
 'cải_thiện': 14,
 'cảm_biến': 15,
 'cảm_ứng': 16,
 'cắm': 17,
 'dock': 18,
 'dung_lượng': 19,
```

- Tạo 1 từ điển dictionary từ intro_products_re bao gồm từ và chỉ số của từ đó. Gồm 40.879 từ

ContentBased_Gensim

```
corpus[0]
```

```
[(0, 1),
 (1, 1),
 (2, 1),
 (3, 2),
 (4, 3),
 (5, 1),
 (6, 1),
 (7, 1),
 (8, 1),
 (9, 1),
 (10, 1),
 (11, 1),
 (12, 1),
 (13, 2),
 (14, 1),
 (15, 1),
 (16, 1),
 (17, 1),
 (18, 2),
 (19, 1),
 (20, 2),
 (21, 1),
 (22, 1),
```

- Tạo 1 corpus là ma trận thưa thớt với mỗi sản phẩm sẽ có chỉ số và số lần xuất hiện của từ trong intro_products_re. Chỉ những từ nào có xuất hiện thì mới liệt kê

ContentBased_Gensim

- Dùng models có sẵn của Gensim, áp dụng TfidfModel để tạo ra tfidf của corpus
- Tính toán sự tương đồng trong ma trận thưa thớt lưu vào index

```
tfidf = models.TfidfModel(corpus)
# tính toán sự tương đồng trong ma trận thưa thớt
index = similarities.SparseMatrixSimilarity(tfidf[corpus],
                                             num_features = feature_cnt)
```

- Viết function recommender để đề xuất sản phẩm
- Input: name_description_pre của sản phẩm đang xem, dictionary bao gồm từ và các chỉ số, tfidf của corpus, index là sự tương đồng trong ma trận thưa thớt
- Output: 6 item có điểm tương đồng cao nhất bao gồm chính nó

ContentBased_Gensim

- Results của sản phẩm đang xem có id 0

```
results = recommender(name_description_pre, dictionary, tfidf, index)
```

```
View product's vector:
[(5, 1), (15, 1), (48, 1), (67, 1)]
5 highest scores:
      id     score
0      0  0.286694
75    75  0.226480
23    23  0.144560
719  719  0.129248
587  587  0.126589
245  245  0.125531
Ind to list:
[0, 75, 23, 719, 587, 245]
```

ContentBased_Gensim

- Viết function recommendation:
- Input: product_ID là mã sp đang xem
- Output: 5 item có điểm tương đồng cao nhất không bao gồm chính nó

```
def recommendation(product_ID):
  product = products[products.item_id == product_ID].head(1)
  name_description_pre = product['name_description_pre'].to_string(index = False)
  results = recommender(name_description_pre, dictionary, tfidf, index)
  results = results[results.item_id != product_ID]
  return results
```

ContentBased_Gensim

- Ví dụ: Đề xuất cho sản phẩm đang xem có mã 48102821

	index	item_id	name	id	score
75	75	35607267	Tai nghe Bluetooth Inpods 12 Thời trang	75	0.226480
23	23	35373097	Tai Nghe Bluetooth True Wireless AMOI F9 5.0 C...	23	0.144560
719	719	48273751	Tai nghe Blutooth 5.0 kiêm dock sạc dự phòng- ...	719	0.129248
587	587	79965318	Tai nghe Bluetooth Lanith – Tai Nghe Không Dây...	587	0.126589
245	245	58291928	Tai Nghe True Wireless Earbuds QCY T7 Bluetoot...	245	0.125531

Kết luận

- Nhìn chung 2 thuật toán cho kết quả đề xuất tương đối tốt.
- Gensim tốn ít không gian lưu trữ hơn.

CollaborativeFiltering_ALS

- CollaborativeFiltering áp dụng ALS

```
data = spark.read.csv('Reviews.csv',inferSchema=True,header=True)
```

```
data.show(5, False)
```

```
+---+---+-----------+----------+------+
|_c0|id |customer_id|product_id|rating|
+---+---+-----------+----------+------+
|0  |0  |709310     |10001012  |3     |
|1  |1  |10701688   |10001012  |5     |
|2  |2  |11763074   |10001012  |5     |
|3  |3  |9909549    |10001012  |5     |
|4  |4  |1827148    |10001012  |5     |
+---+---+-----------+----------+------+
only showing top 5 rows
```

```
data.printSchema()
```

```
root
 |-- _c0: integer (nullable = true)
 |-- id: integer (nullable = true)
 |-- customer_id: integer (nullable = true)
 |-- product_id: integer (nullable = true)
 |-- rating: integer (nullable = true)
```

CollaborativeFiltering_ALS

- Dữ liệu gồm 361.090 reviews của 251.149 KH cho 4.214 SP
- Độ thưa thớt ~ 99.97%

```
display(numerator, customers, products)
```

```
361090
251149
4214
```

```
denominator = customers* products
denominator
```

```
1058341886
```

```
# độ thưa thớt của dữ liệu
sparsity = 1 - (numerator*1.0/denominator)
print('Sparsity: ', sparsity)
```

```
Sparsity:  0.999658815355627
```

CollaborativeFiltering_ALS

- KH chấm điểm rating khá cao. Nhiều nhất là rating = 5 với 256.211 reviews

```
+------+------+
|rating| count|
+------+------+
|     1| 16616|
|     3| 20600|
|     5|256211|
|     4| 60565|
|     2|  7098|
+------+------+
```

- Chia DL thành 2 tập training và test

```
(training, test) = data.randomSplit([0.8, 0.2])
```

CollaborativeFiltering_ALS

```
als = ALS(maxIter = 20,
          regParam = 0.1,
          userCol = 'customer_id',
          itemCol = 'product_id',
          ratingCol = 'rating',
          coldStartStrategy = 'drop',
          nonnegative = True)
model = als.fit(training)
```

```
predictions = model.transform(test)
```

```
predictions.show(5)
```

```
+-----+-----+-----------+----------+------+----------+
|  _c0|   id|customer_id|product_id|rating|prediction|
+-----+-----+-----------+----------+------+----------+
|48567|48567|    7523016|   1675793|     5| 3.7366116|
|48573|48573|    2225318|   1675793|     5| 2.9967952|
|48568|48568|   13098881|   1675793|     4|  2.769885|
|68613|68613|   13676965|   2069769|     4| 2.8575258|
|68597|68597|   11137984|   2069769|     2| 2.0279472|
+-----+-----+-----------+----------+------+----------+
only showing top 5 rows
```

- Build model với traning data và thuật toán ALS gồm các tham số như hình bên
- Dự đoán giá trị rating cho dữ liệu test, tạo được cột prediction

CollaborativeFiltering_ALS

- Đánh giá model
- RMSE đang rất cao, 1.27 so với std là 1.017 => tuning parameter để cải thiện model.

```
evaluator = RegressionEvaluator(metricName='rmse',
                                labelCol = 'rating',
                                predictionCol = 'prediction')
rmse = evaluator.evaluate(predictions)
rmse
```

```
1.2721176230827271
```

```
data.describe(['rating']).show()
```

```
+-------+------------------+
|summary|            rating|
+-------+------------------+
|  count|            361090|
|   mean| 4.475136392589105|
| stddev|1.0166716679634682|
|    min|                 1|
|    max|                 5|
+-------+------------------+
```

CollaborativeFiltering_ALS

- Tunning Parameter
- Lựa chọn model cho RMSE thấp nhất
- Recommender có sự cải thiện lớn về sai số, giảm từ 1.27 xuống 0.295.

```
# initialize the ALS model
als_model = ALS(userCol='customer_id', itemCol='product_id', ratingCol='rating', coldStartStrategy='drop', nonnegative=True)

# create the parameter grid
params = ParamGridBuilder().addGrid(als_model.regParam, [.01, .05, .1, .15]).addGrid(als_model.rank, [5, 10, 50, 100]).build()

# create crossvalidator estimator
cv = CrossValidator(estimator = als_model, estimatorParamMaps= params, evaluator= evaluator, parallelism = 4)
best_model = cv.fit(data)
model = best_model.bestModel
```

```
predictions = model.transform(test)
rmse = evaluator.evaluate(predictions)
print('RMSE:', rmse)
```

```
RMSE: 0.2951857150515815
```

CollaborativeFiltering_ALS

- Đề xuất 5 sp có dự đoán rating cao nhất cho tất cả users

```
user_recs = model.recommendForAllUsers(5)
```

```
user_recs.printSchema()
```

```
root
 |-- customer_id: integer (nullable = false)
 |-- recommendations: array (nullable = true)
 |    |-- element: struct (containsNull = true)
 |    |    |-- product_id: integer (nullable = true)
 |    |    |-- rating: float (nullable = true)
```

CollaborativeFiltering_ALS

- Ví dụ: Đề xuất 5 sp có dự đoán rating cao nhất cho customer_id = 997878

```
customer_id = 997878
result = user_recs.filter(user_recs['customer_id'] == customer_id)
result.show(truncate = False)
```

```
+-----------+--------------------------------------------------------------------------------------------------------------------------------------+
|customer_id|recommendations                                                                                                                       |
+-----------+--------------------------------------------------------------------------------------------------------------------------------------+
|997878     |[[49078809, 7.3777604], [44579109, 6.8818226], [55366883, 6.872575], [58545325, 6.819825], [72969822, 6.6251655]]                    |
+-----------+--------------------------------------------------------------------------------------------------------------------------------------+
```

```
result = result.select(result.customer_id, explode(result.recommendations))
result = result.withColumn('product_id', result.col.getField('product_id'))\
               .withColumn('rating', result.col.getField('rating'))
result.show()
```

```
+-----------+--------------------+----------+---------+
|customer_id|                 col|product_id|   rating|
+-----------+--------------------+----------+---------+
|     997878|[49078809, 7.3777...|  49078809|7.3777604|
|     997878|[44579109, 6.8818...|  44579109|6.8818226|
|     997878|[55366883, 6.872575]|  55366883| 6.872575|
|     997878|[58545325, 6.819825]|  58545325| 6.819825|
|     997878|[72969822, 6.6251...|  72969822|6.6251655|
+-----------+--------------------+----------+---------+
```

CollaborativeFiltering_ALS

- Lưu kết quả dưới dạng Parquet để tiến hành đưa lên website đề xuất cho KH

```
user_recs.write.parquet('Rec_U.parquet', mode = 'overwrite')
```

CollaborativeFiltering_Surprise

- Dữ liệu bao gồm 361.090 reviews của 251.149 KH cho 4.214 SP

```
n_ratings = len(df)
n_products = len(df['product_id'].unique())
n_customers = len(df['customer_id'].unique())
```

```
display(n_ratings, n_products, n_customers)
```

```
361090
4214
251149
```

- Sử dụng Reader() để parse 1 data frame thành dữ liệu chuẩn để dụng surprise

```
reader = Reader()
data = Dataset.load_from_df(df[['customer_id', 'product_id', 'rating']], reader)
```

CollaborativeFiltering_Surprise

- Đưa 6 thuật toán có thể sử dụng để build collaborative filtering

```
algorithm = [SVD(), SVDpp(), NMF(), SlopeOne(), CoClustering(), BaselineOnly()]
```

- Lựa chọn BaselineOnly() vì có RMSE thấp nhất và thời gian thực hiện ngắn

Model	test_rmse	fit_time	test_time
BaselineOnly	0.969870	2.840784	0.645969
SVD	0.973352	19.935855	0.850277
SVDpp	0.984872	36.628077	1.524906
CoClustering	1.050876	28.428150	0.638937
SlopeOne	1.077203	3.654448	0.804788
NMF	1.122492	42.227506	0.584286

CollaborativeFiltering_Surprise

- BaselineOnly() kết hợp Tuning Parameter: Sử dụng GridSearchCV để chọn bộ tham số tốt nhất, có RMSE = 0.971

```
bsl_options = {'method': 'als', 'n_epochs': 10, 'lr_all': 0.002, 'reg_all': 0.4}
algorithm = BaselineOnly(bsl_options=bsl_options)
```

```
trainset = data.build_full_trainset()
algorithm.fit(trainset)
```

CollaborativeFiltering_Surprise

- Viết hàm recommender:
- Input là customer_id
- Output là data frame bao gồm 5 product_id và EstimateScore (điểm rating dự đoán) > 3

```
def recommender(customer_id):
  df_score = df[['product_id']]
  df_score['EstimateScore'] = df_score['product_id'].apply(lambda x: algorithm.predict(customer_id, x).est)
  df_score = df_score.sort_values(by=['EstimateScore'], ascending=False)
  df_score = df_score.drop_duplicates()
  df_score = df_score[df_score.EstimateScore >= 3.0]
  results = df_score.head()
  return results
```

CollaborativeFiltering_Surprise

- Ví dụ: Đề xuất cho KH có customer_id = 14188390

```
recommender_14188390 = recommender(14188390)
recommender_14188390
```

	product_id	EstimateScore
237331	53080935	4.641777
305513	68025746	4.641678
324562	73179180	4.633438
319924	71896003	4.630561
313577	70771651	4.627334

CollaborativeFiltering_Surprise

Kết luận:

- Sau khi xây dựng model theo 2 cách: ALS (BigData) và BaseLineOnly(Machine Learning) thì em sử dụng ALS để đề xuất cho người dùng vì có RMSE thấp hơn. (ALS là 0.295 và BaseLineOnly là 0.97)