TRƯỜNG ĐẠI HỌC MỞ THÀNH PHỐ HỒ CHÍ MINH
KHOA CÔNG NGHỆ THÔNG TIN

BÁO CÁO GIỮA KỲ

ĐỀ TÀI: KHAI PHÁ BỘ DỮ LIỆU TIKI BOOKS

Môn học: **Khai phá dữ liệu**

Giảng viên: **Nguyễn Văn Bảy**

Lớp: **IT2003**

SINH VIÊN THỰC HIỆN:

Nguyễn Đặng Tuyết Nhi – 2051050318 – Trưởng nhóm

Lê Nguyễn Tiến Vững – 2051052156

Bảo Khiêm – 2051052062

TP. HỒ CHÍ MINH, 2023

LỜI CẢM ƠN

Lời đầu tiên, nhóm em xin trân trọng gửi lời cảm ơn chân thành đến thầy – người đã hỗ trợ, giúp đỡ về mặt kiến thức và tinh thần chúng em trong quá trình nhóm làm bài và hoàn thành bài báo cáo.

Do kiến thức của chúng em còn nhiều hạn chế nên sẽ không tránh khỏi những thiếu sót nhất định. Chúng em mong nhận được sự đóng góp ý kiến của thầy để chúng em có thêm kinh nghiệm để có thể làm tốt hơn trong những môn học sau

NHẬN XÉT CỦA GIÁO VIÊN HƯỚNG DẪN

MỤC LỤC

DANH MỤC ẢNH

DANH MỤC BẢNG

Chương 1. TỔNG QUAN

1.1. Giới thiệu về bộ dataset

Đây là bộ dữ liệu sách của công ty thương mại điện tử Tiki Việt Nam. Bộ dữ liệu cung cấp thông tin về 2024 cuốn sách bán chạy nhất, dữ liệu bao gồm:

- Thông tin về các cuốn sách
- Các đường dẫn tới bìa sách của các cuốn sách đó
- Và khoảng gần 50 comments cho mỗi cuốn sách

1.2. Ý tưởng nghiên cứu

Với một số định hướng nghiên cứu Dataset như nhận diện tác giả được ưa thích nhất, tìm hiểu xem cuốn sách nào sẽ có cơ hội cao được mua và phân tích cảm xúc của các comment qua việc xử lý ngôn ngữ tự nhiên. Nhóm chúng em cuối cùng cũng đưa ra quyết định là sẽ khai phá bộ dataset với hướng tiếp cận là tìm xem cuốn sách nào sẽ có cơ hội cao được mua. Vì theo hướng nghiên cứu đó, nên nhóm em tập trung nhiều vào dữ liệu Sách (file book_data.csv) trong bộ dataset được cung cấp.

1.3. Mô tả chi tiết dữ liệu

Bộ dataset Tiki Books gồm có:

- **book_data.csv**
  - Tổng cộng có 1801 dòng dữ liệu.
  - Với các cột: product_id, title, authors, original_price, current_price, quantity, category, n_review, avg_rating, pages, manufacturer, cover_link.

-

Column	Description
product_id	id of the product in the Tiki database (unique)
title	name of the book, maybe contain republish time
authors	same with its name
original_price	price at the first time
current_price	price at present if having a discount
quantity	total number of books sold of all time
category	kind of book
n_review	number of reviews
avg_rating	average rating (max 5.0)
pages	total pages of each book

- **book_id.csv**
  - Tổng cộng có 2024 dòng dữ liệu.
  - Chỉ với một cột: id
  -

Column	Description
id	id of all products

- **comments.csv**
  - Tổng cộng có 141,281 dòng dữ liệu.
  - Với các cột: product_id, comment_id, title, thanks_count, customer_id, rating, content.
  -

Column	Description
product_id	same with book_data file
comment_id	each comment has individual id
title	keywork of comment
thank_count	number of like of other people
customer_id	each customer has individual id

Column	Description
rating	average rating of the comment
content	same with its name

Chương 2. TIỀN XỬ LÝ DỮ LIỆU

2.1. Giới thiệu quy trình tiền xử lý dữ liệu

2.1.1. Giới thiệu quy trình

Tiền xử lý dữ liệu là bước quan trọng trong việc giải quyết bất kỳ vấn đề nào trong lĩnh vực Khai phá dữ liệu. Hầu hết các bộ dữ liệu được sử dụng trong các vấn đề liên quan đến Khai phá dữ liệu cần được xử lý, làm sạch và biến đổi trước khi một thuật toán xử lý dữ liệu có thể được huấn luyện trên những bộ dữ liệu này. Các kỹ thuật tiền xử lý dữ liệu phổ biến hiện nay bao gồm: xử lý dữ liệu bị khuyết (missing data), mã hóa các biến nhóm (encoding categorical variables), chuẩn hóa dữ liệu (standardizing data), co giãn dữ liệu (scaling data), … Những kỹ thuật này tương đối dễ hiểu nhưng sẽ có nhiều vấn đề phát sinh khi chúng ta áp dụng vào các dữ liệu thực tế. Bởi lẽ các bộ dữ liệu ứng với các bài toán trong thực tế rất khác nhau và mỗi bài toàn thì đối mặt với những thách thức khác nhau về mặt dữ liệu.

2.1.2. Ngôn ngữ Python

Trong việc nghiên cứu sắp tới để khai phá dữ liệu thì nhóm đã sử dụng ngôn ngữ Python là công cụ chính xuyên suốt để giải quyết các vấn đề như: Làm sạch dữ liệu (Data cleaning), hiển thị dữ liệu (Data visualization) và các thuật toán phục vụ đến việc khái phá dữ liệu (K-mean, KNN, Naïve-Bayes)… Một số thư viện của ngôn ngữ Python trong việc xử lý dữ liệu như: Pandas, Numpy, Matplotlib, sklearn, v.v.

2.2. Làm sạch dữ liệu (file book_data.csv)

2.2.1. Xử lý các ô dữ liệu trống và tạo biểu đồ mô hình hóa dữ liệu

```
data_df.isnull().sum()
```

```
product_id          0
title               0
authors           143
original_price      0
current_price       0
quantity           45
category            0
n_review            0
avg_rating          0
pages             250
manufacturer      273
cover_link          0
dtype: int64
```

Hình 1. Tìm dữ liệu bị trống

Ở đây ta thấy các cột dữ liệu như authors, quantity, pages, manufacturer có các ô dữ liệu trống nên ta sẽ tới từng cột để xử lý.

- **Authors:**
  - Hiển thị các giá trị của cột và tần suất xuất hiện:

```
data_df.authors.value_counts()
```

```
Nguyễn Nhật Ánh     24
Higashino Keigo     20
.                   18
Thích Nhất Hạnh     16
Haruki Murakami     15
                    ..
Urako Kanamori       1
Cổ Viên              1
Robert Winston       1
Yongchul Kwon        1
John C. Maxwell      1
Name: authors, Length: 1083, dtype: int64
```

Hình 2. Đếm số lần xuất hiện tên các tác giả

- Thay các ô dữ liệu trống thành giá trị “Unknown”:

```
# Thay thế giá trị "." ở cột Authors thành "Unknown"
data_df.loc[data_df.authors == '.' , 'authors'] = "Unknown"
```

```
data_df.authors = data_df.authors.fillna("Unknown")
```

Hình 3. Thay thế các ô giá trị "." thành "Unknown"

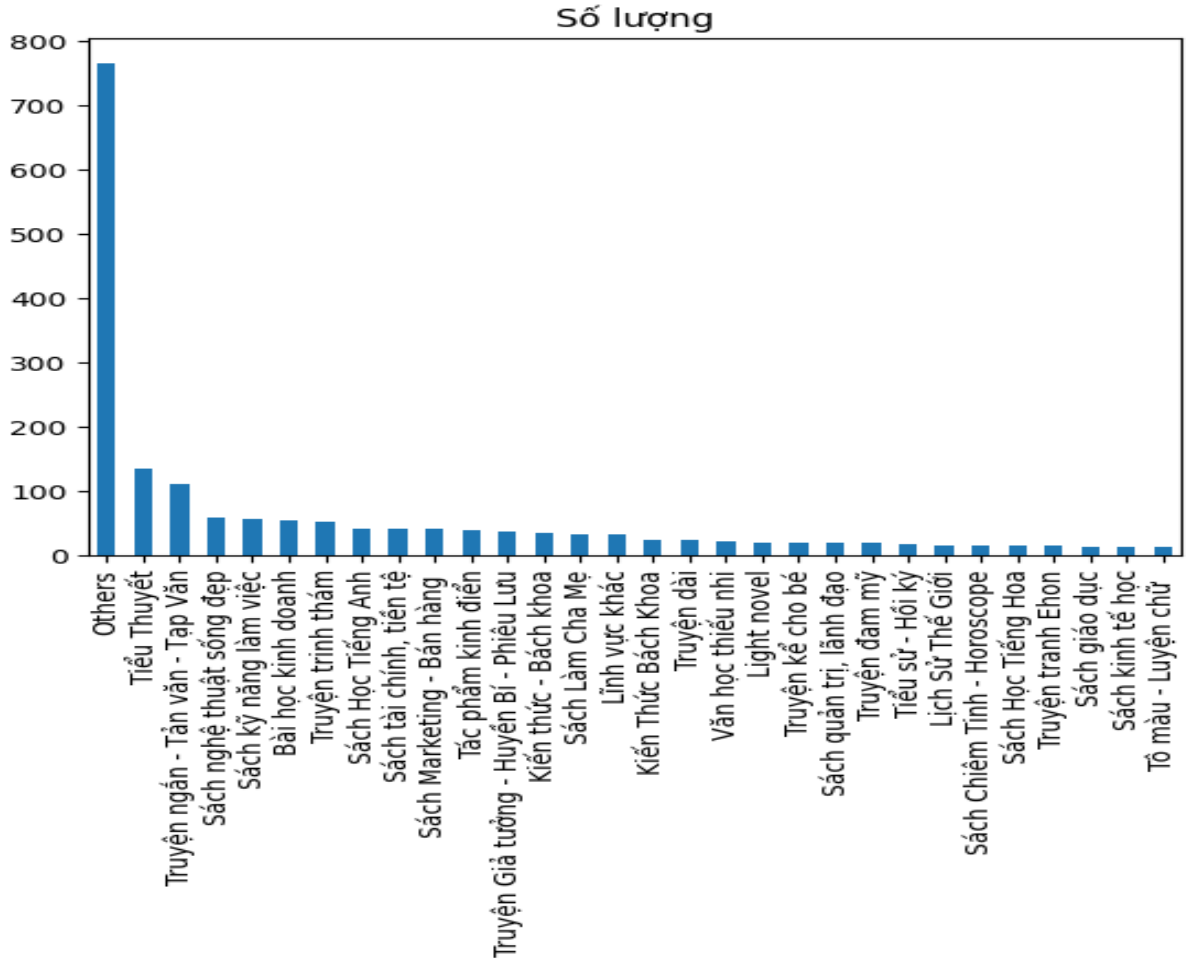
- **Quantity:**
  Thay các ô giá trị trống thành giá trị trung bình được tính từ cột Quantity

```
import numpy as np
data_df.quantity = data_df.quantity.fillna(np.mean(data_df.quantity))
```

Hình 4. Thay các ô giá trị trống thành giá trị trung bình được tính từ cột Quantity

- **Category:**
  Đếm tần suất xuất hiện của các giá trị trong cột category, lấy top 30 giá trị xuất hiện nhiều nhất và gộp các thể loại còn lại thành “Others”.

```
def handle_category(category):
    if category not in keeping_values:
        return "Others"
    return category

data_df.category = data_df.category.apply(lambda category: handle_category(category))

data_df.category.value_counts().plot(kind='bar', title='Số lượng')
```



Hình 5. Biểu đồ số lượng các thể loại sách

Nhận xét: Do việc gộp các thể loại còn lại thành “Others” nên cột “Others” sẽ chiếm số lượng vượt bậc hơn so với các thể loại còn lại. Nếu xét tổng quan hơn thì thể loại “Tiểu thuyết” có số lượng sách nhiều nhất, kế đó là “Truyện ngắn - Tản văn - Tạp Văn”. Hai thể loại này chiếm số lượng nhiều hơn và gần như là gấp đôi so với các thể loại sách còn lại cho thấy nó có khả năng được độc giả ưa chuộng hơn và chọn mua nhiều nên phần “cung” mới chênh lệch nhiều hơn như vậy.

- **Manufacturer:**
  Lấp các ô dữ liệu rỗng:

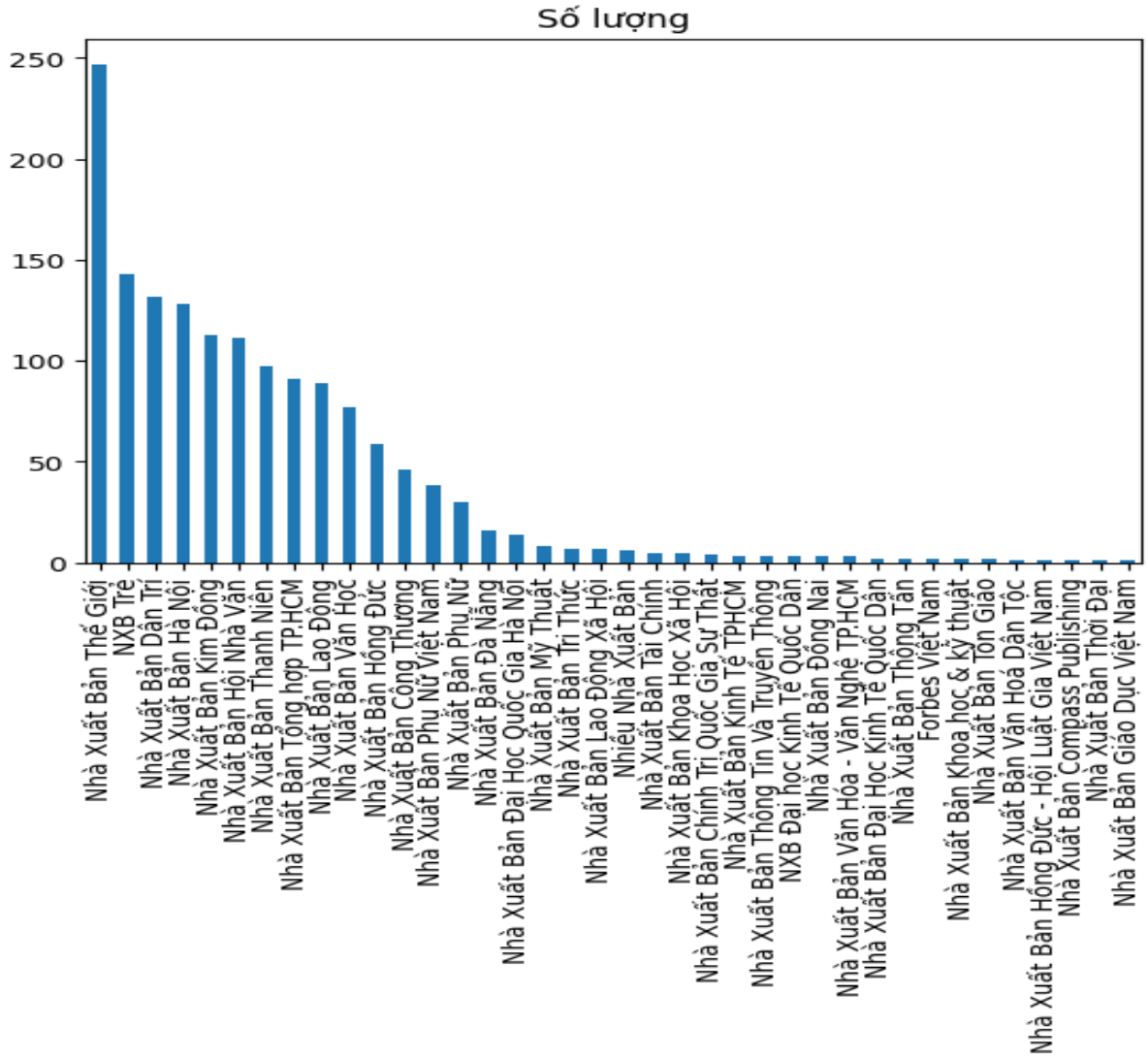
```
data_df.manufacturer = data_df.manufacturer.fillna("Unknown")
```

Sửa lỗi giá trị nhà xuất bản:

```
def handle_manufactuner(manufactuner):
    if manufactuner == "hong duc":
        return "Nhà Xuất Bản Hồng Đức"
    elif manufactuner == "NXB Dân Trí":
        return "Nhà Xuất Bản Dân Trí"
    elif manufactuner == "ĐHQG Hà Nội":
        return "Nhà Xuất Bản Đại Học Quốc Gia Hà Nội"
    else:
        return manufactuner


data_df.manufacturer = data_df.manufacturer.apply(lambda manufacturer:
handle_manufactuner(manufacturer))
```

Vẽ biểu đồ:

```
data_df.manufacturer.value_counts().plot(kind='bar', title='Số lượng')
```



Hình 6. Biểu đồ số lượng các Nhà Xuất Bản

Nhận xét: Qua biểu đồ trên, ta thấy Nhà Xuất Bản Thế Giới là nơi phát hành sách nhiều nhất và chênh lệch hơn hẳn so với các nhà xuất bản còn lại. Khách quan mà nói, ta có thể nhận định rằng việc mua sách do Nhà Xuất Bản Thế Giới phát hành có thể đáp ứng được yêu cầu của độc giả hơn vì NXB này phát hành nhiều sách hơn, có thể đa dạng về thể loại và đáp ứng đủ về số lượng sách cần mua hơn so với các NXB còn lại.

2.3. Biến đổi dữ liệu bằng cách chuẩn hóa dữ liệu

2.3.1. Giới thiệu về phương pháp chuẩn hóa dữ liệu

Chuẩn hóa (Normalization) là phương pháp chuyển đổi giá trị thuộc tính vào một miền giá trị. Chuẩn hóa có một số phương pháp thông dụng hay được sử dụng như:

- Z-Score Normalization: Sử dụng giá trị trung bình và phương sai để chuẩn hóa dữ liệu

$$Z=\frac{x-\mu}{\sigma}$$

Với: x là giá trị đầu vào

μ là giá trị trung bình

σ là phương sai

- Decimal Normalization: với j là số nguyên nhỏ nhất để Max$|U_i| < 1$

$$U_i = \frac{V_i}{10^j}$$

- Min-Max Normalization:

$$x_{scaled} = \frac{x - x_{min}}{x_{max} - x_{min}}$$

Trong phần này, nhóm chúng em xử dụng phương pháp Min-Max Normalization để xử lý dữ liệu. Đối với những dữ liệu có giá trị là số thì sử dụng công thức được nêu ở trên. Còn đối với những dữ liệu có giá trị định danh thì chúng em dựa trên tần suất xuất hiện của các giá trị và sau đó áp dụng công thức để tìm ra giá trị min-max.

2.3.2. Chuẩn hóa dữ liệu

- **Bước 1: Chọn ra các cột cần thiết trong dataset để chuẩn hóa dữ liệu**

```
# Xóa các cột không cần tính
df = df.drop({'No', 'title', 'authors', 'original_price', 'pages', 'cover_link'}, axis=1)
```

Dataframe sau khi đã xóa các cột không cần tính:

product_id	current_price	quantity	category	n_review	avg_rating	manufacturer
74021317	64800	53075	Tiểu Thuyết	11481	5.0	Nhà Xuất Bản Hội Nhà Văn
184466860	59900	7929	Others	780	4.8	Nhà Xuất Bản Thế Giới
73787185	126000	17896	Lĩnh vực khác	3623	4.8	Nhà Xuất Bản Hội Nhà Văn
52789367	47400	24668	Tác phẩm kinh điển	5131	5.0	Nhà Xuất Bản Hà Nội
147920903	81000	10000	Tiểu Thuyết	1636	4.8	Nhà Xuất Bản Hội Nhà Văn

- **Bước 2: Chuẩn hóa dữ liệu**
  - Với các cột chứa dữ liệu số:
    - Cột *quantity*:

```
min_quan = df_book['quantity'].min()
max_quan = df_book['quantity'].max()

df_book['quantity_new'] =
(((df_book[['quantity']] -min_quan)) /(max_quan -min_quan)).round(3)
```

    - Cột *current_price*:

```
min_price = df_book['current_price'].min()
max_price = df_book['current_price'].max()

df_book['current_price_new'] =
(((df_book[['current_price']] -min_price)) /(max_price -min_price)).round(3)
```

- Cột *n_review*:

```
min_review = df_book['n_review'].min()
max_review = df_book['n_review'].max()

df_book['n_review_new'] =
(((df_book[['n_review']] -min_review)) /(max_review -min_review)).round(3)
```

- Cột *n_review*:

```
min_review = df_book['n_review'].min()
max_review = df_book['n_review'].max()

df_book['n_review_new'] =
(((df_book[['n_review']] -min_review)) /(max_review -min_review)).round(3)
```

- Cột *avg_rating*:

```
min_rating = df_book['avg_rating'].min()
max_rating = df_book['avg_rating'].max()

df_book['avg_rating_new'] =
(((df_book[['avg_rating']]-min_rating))/(max_rating-min_rating)).round(3)
```

- Với các cột chứa dữ liệu là giá trị định danh:
  - Cột *manufacturer*:
    Tạo một bảng thống kê tần suất xuất hiện của các giá trị trong cột:

```
# Thống kê tần suất xuất hiện
frequency_table = df_book['manufacturer'].value_counts().sort_index().reset_index()
frequency_table.columns = ['manufacturer', 'Tần suất xuất hiện']


# Tính giá trị nhỏ nhất
min_frequency = frequency_table['Tần suất xuất hiện'].min()
# Tính giá trị lớn nhất
max_frequency = frequency_table['Tần suất xuất hiện'].max()


frequency_table['Tần suất xuất hiện'] = frequency_table['Tần suất xuất hiện'].apply(lambda x:
(x-min_frequency)/(max_frequency-min_frequency)).round(4)
```

Kết quả hiển thị (không chụp được hết bảng kết quả):

NXB Trẻ	0.5379
NXB Đại học Kinh Tế Quốc Dân	0.0076
Nhiều Nhà Xuất Bản	0.0189
Nhà Xuất Bản Chính Trị Quốc Gia Sự Thật	0.0114
Nhà Xuất Bản Compass Publishing	0.0000
Nhà Xuất Bản Công Thương	0.1705
Nhà Xuất Bản Dân Trí	0.4962
Nhà Xuất Bản Giáo Dục Việt Nam	0.0000
Nhà Xuất Bản Hà Nội	0.4811
Nhà Xuất Bản Hồng Đức	0.2197
Nhà Xuất Bản Hồng Đức - Hội Luật Gia Việt Nam	0.0000
Nhà Xuất Bản Hội Nhà Văn	0.4167

Hình 7. Kết quả tính tần suất xuất hiện của các NXB

Dùng phương thức merge để đối chiếu với giá trị trong dataframe chính:

```
data_new = df_book.merge(frequency_table[['manufacturer', 'Tần suất xuất hiện']], on='manufacturer')
data_new = data_new.rename(columns={'Tần suất xuất hiện':'manufacturer_new'})
```

- Cột *category* (cách làm tương tự cột *manufacturer*):

```
# Thống kê tần suất xuất hiện
frequency_table_cate = df_book['category'].value_counts().sort_index().reset_index()
frequency_table_cate.columns = ['category', 'cate_freq']


# Tính giá trị nhỏ nhất
min_frequency_cate = frequency_table_cate['cate_freq'].min()
# Tính giá trị lớn nhất
max_frequency_cate = frequency_table_cate['cate_freq'].max()


frequency_table_cate['cate_freq'] = frequency_table_cate['cate_freq'].apply(lambda x:
(x-min_frequency_cate)/(max_frequency_cate-min_frequency_cate)).round(4)
```

```
data_new = data_new.merge(frequency_table_cate[['category', 'cate_freq']], on='category')
data_new = data_new.rename(columns={'cate_freq':'cate_freq_new'})
```

- **Bước 3: Chuyển các dữ liệu đã chuẩn hóa sang một file csv mới**

```
data_new = data_new[["product_id","current_price_new","quantity_new","cate_freq_new",
"n_review_new" ,"avg_rating_new","manufacturer_new",]]
data_new.to_csv('/content/drive/MyDrive/KPDL/BTL/Datanew.csv')
```

Chương 3. KHAI PHÁ DỮ LIỆU

3.1. Giới thiệu các phương pháp khai phá dữ liệu

Sau khi đã có data frame thì nhóm chúng em sẽ tiến hành các bước khai phá.

3.1.1. Gom cụm bằng K-means (clustering)

Clustering là kỹ thuật phổ biến nhất trong học tập không giám sát, nơi dữ liệu được nhóm dựa trên sự giống nhau của các điểm dữ liệu. Clustering có nhiều ứng dụng trong đời thực, nơi nó có thể được sử dụng trong nhiều tình huống khác nhau.

Nguyên tắc cơ bản đằng sau cụm là việc gán một tập hợp các quan sát nhất định thành các nhóm con hoặc cụm sao cho các quan sát hiện diện trong cùng một cụm có mức độ giống nhau. Đó là việc thực hiện khả năng nhận thức của con người để phân biệt các đối tượng dựa trên bản chất của chúng.

Ở đây nhóm chúng em quyết định gom cụm bằng phương pháp K-means với việc tính khoảng cách bằng phương pháp Manhattan (tính khoảng cách 2 điểm giữa không gian 2 chiều)

Bắt đầu bằng việc cho một dòng dữ liệu mà chúng em cho là mốc (giá trị số lương sách bán nhiều nhất) rồi dùng ngôn ngữ Python để tính khoảng cách Manhattan để ra kết quả, với 0 là gần nhất và 6 là xa nhất. Sau khi tính xong thì tiếp tục gom cụm bằng phương pháp K-means với 4 cụm và đánh dấu 4 centroid ở 4 cụm.

3.1.2. Phân lớp và train model (Classification)

Sau khi đã có 4 cụm thì nhóm em sử dụng 4 label cho dataframe: ‘top seller’, ‘bán chạy’, ‘bán vừa’ và ‘bán tệ’ và sử dụng hai phương pháp KNN và Naïve_Bayes để xem phương thức nào thích hợp nhất để chạy trên model.

3.2. Tiến hành khai phá dữ liệu

- ***Bước 1: Tính Manhattan:***

  Trong dataframe này, nhóm chúng em nhận ra có một dòng dữ liệu mà số sách bán ra có giá trị chênh lệch so với giá trị nhiều thứ 2 (chênh lệch 10 lần), nên chúng em quyết định bỏ dòng dữ liệu đó và mặc định nó là ‘Top seller’.

Unnamed: 0	product_id	current_price_new	quantity_new	cate_freq_new	n_review_new	avg_rating_new	manufacturer_new
0	74021317	0.0540	0.0791	0.8395	1.0000	1.00	0.5833
1	147920903	0.0675	0.0149	0.8395	0.1425	0.96	0.5833
2	52788072	0.0545	0.0256	0.8395	0.2642	0.96	0.5833
3	114937969	0.0750	0.0055	0.8395	0.0677	0.96	0.5833
4	70016692	0.0440	0.0100	0.8395	0.1347	0.96	0.5833
...	...	...	...	...	...	...	...
1763	29290844	0.0183	0.0000	0.9973	0.0005	1.00	0.4621
1764	110181955	0.1242	0.0008	0.9973	0.0121	0.96	0.6364
1765	169262882	0.1421	0.0000	0.9973	0.0001	1.00	0.7803
1766	1672721	0.0392	0.0042	0.9973	0.0000	0.00	0.9508
1767	1041102	0.0742	0.0006	0.9973	0.0013	0.90	0.9508

Với dataframe trên, chúng em chọn dòng dữ liệu có giá trị trong cột "quanity_new" lớn nhất làm chuẩn:

```
im = data[data['quantity_new'] == data['quantity_new'].max()]
```

	Unnamed: 0	product_id	current_price_new	cate_freq_new	n_review_new	avg_rating_new	manufacturer_new	quantity_new
486	486	26114399	0.0475	0.0	0.575	0.96	0.0	1.0

Dòng dữ liệu chuẩn này nằm ở dòng thứ 486.

```
# Dùng phép toán Manhattan để tính khoảng cách giữa quyển sách đến với quyển sách có khả năng bán chạy nhất
x = 486
col =['current_price_new', 'quantity_new', 'cate_freq_new', 'n_review_new' , 'avg_rating_new' ,'manufacturer_new']
def manhattan_distance(row):
  distance = 0
  for column in col:
    distance += abs(row[column] - data.loc[x, column])
  return distance
data['KQ'] = data.apply(manhattan_distance, axis=1)
data['KQ'].round(4)
data.to_csv('Test2.csv')
```

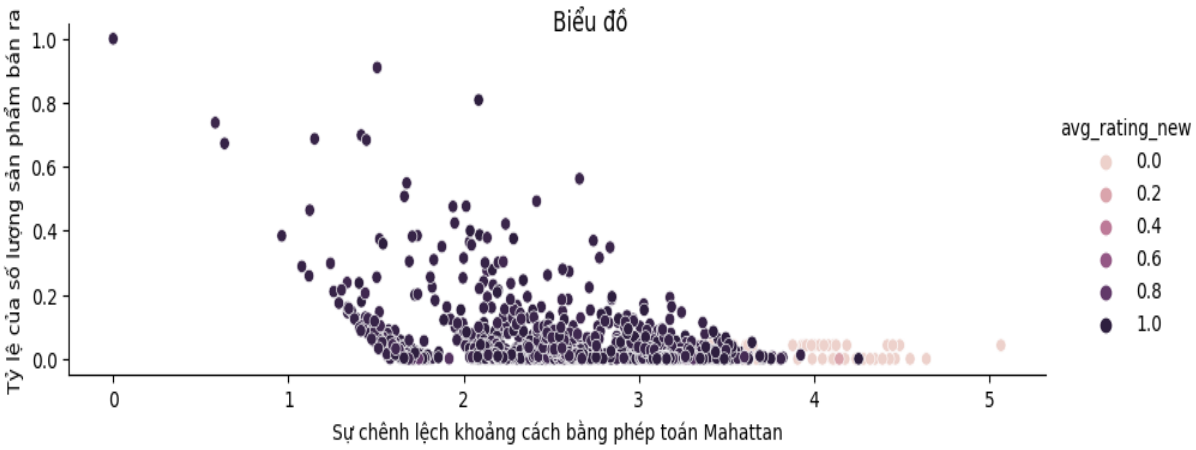
Đây là khoảng cách nhỏ nhất và lớn nhất khi tính bằng phương pháp Manhattan:

```
#Tính khoảng cách lớn nhất và nhỏ nhất
kq_min = data['KQ'].min()
kq_max = data['KQ'].max()
print(kq_min,kq_max )
```

```
0.0 5.06612
```

- ***Bước 2: Vẽ biểu đồ gom cụm và phân lớp:***

```
# Vẽ biểu đồ

import seaborn as sns

%matplotlib inline

g = sns.relplot(

    data = data,

    x = 'KQ',

    y = 'quantity_new',

    aspect= 3,

    height = 3,

    hue = 'avg_rating_new')

g.ax.set_xlabel('Sự chênh lệch khoảng cách bằng phép toán Manhattan')

g.ax.set_ylabel('Tỷ lệ của số lượng sản phẩm bán ra')

g.fig.suptitle('Biểu đồ')
```



Hình 8. Sự chênh lệch khoảng cách qua phép toán Manhattan

Nhận xét:

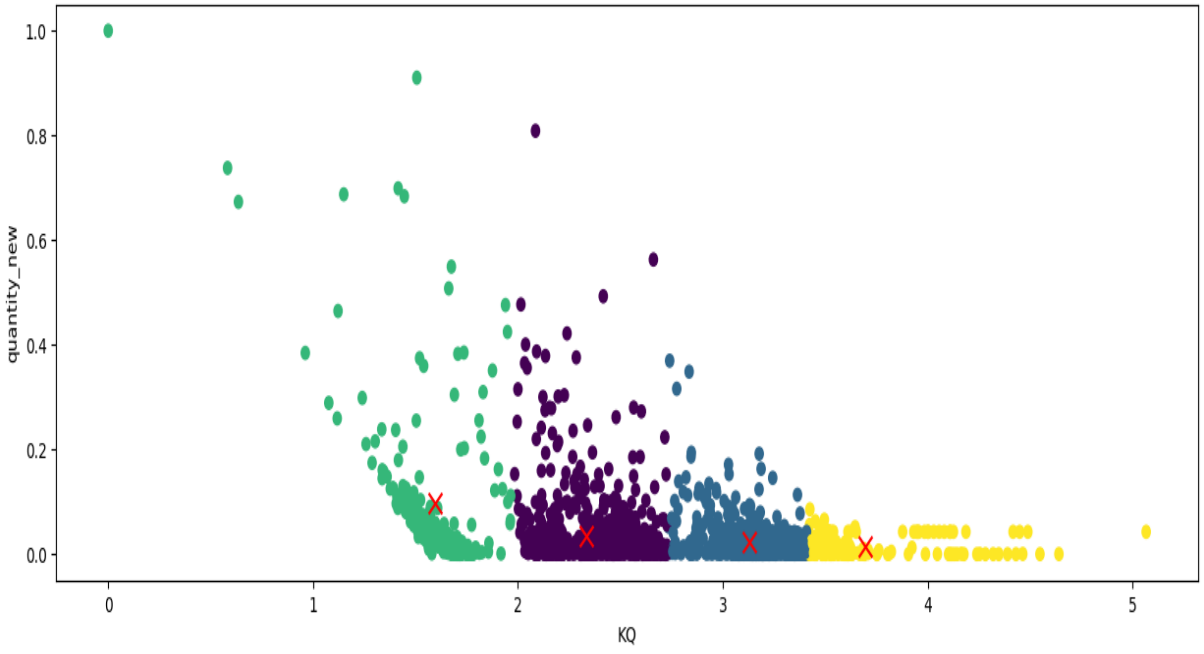
Ta có cột Y là số lượng sản phẩm bán ra của từng quyển sách đã được đổi thành khoảng cách từ [0:1] đây là cột ['quantity'] trong dữ liệu. Ở đó giá trị cao nhất 1, là quyển sách có lượt bán cao nhất là quyển 'Nóng Giận Là Bản Năng , Tĩnh Lặng Là Bản Lĩnh' với lượt bán ra là 65623 quyển với mã số là 26114399.

Cột X là khoảng cách được tính bằng phép toán Manhattan với quyển sách trên là cột ['KQ'], khoảng cách càng xa thì khả năng bán được của quyển sách càng thấp. Ở đây khoảng cách xa nhất là quyển 'Nguyên Lý Marketing (Phiên bản mới nhất 2021)' với lượt bán là 2802 với mã số là 135350713.

Vì thuật toán Manhattan áp dụng vào dữ liệu này được tính không chỉ bằng số lượng sản phẩm bán ra, nó còn phụ thuộc vào giá, số lượng review,... mà dự đoán kết quả nên khả năng dự đoán không chỉ phụ thuộc vào ['quantity'] mà dẫn đến dự đoán sai.

```
# Gom cụm bằng phương pháp K-Means
from sklearn.cluster import KMeans
from sklearn import cluster
import matplotlib.pyplot as plt

#Phân thành 4 cụm
kmeans = KMeans(n_clusters= 4)

#Tạo tọa độ
data_ToaDo = data[['KQ','quantity_new']]

# Nạp tọa độ
kmeans.fit(data_ToaDo)

# Lấy trung tâm của các cụm
centers = kmeans.cluster_centers_
centers.round(4)
```

```
#Phân vùng bằng thư viện của K-Means
labels = kmeans.labels_

# Vẽ biểu đồ dữ liệu
plt.figure(figsize = (15,5))
plt.scatter(data['KQ'], data['quantity_new'], c = labels)
plt.scatter(centers[:, 0], centers[:, 1], c='red', marker='x', s = 100)
plt.xlabel('KQ')
plt.ylabel('quantity_new')
plt.show()

# Tạo Class
data['PhanLoai'] = labels
data['PhanLoai'] = data['PhanLoai'].replace({0: 'Top Seller', 1: 'Bán chạy', 2: 'Bán vừa',
3: 'Bán tệ'})
```



Hình 9. Biểu đồ gom cụm

- ***Bước 3: Sử dụng phương pháp phân lớp KNN và Naïve-Bayes để nhận xét mô hình phân lớp:***

```
# Sử dụng phương pháp KNN dựa trên Class

import pandas as pd

from sklearn.model_selection import  train_test_split

from sklearn.neighbors import KNeighborsClassifier

from sklearn.naive_bayes import GaussianNB

from sklearn.metrics import classification_report,confusion_matrix


# Phương pháp split test: lấy 20% dữ liệu ban đầu để test

train_df, test_df = train_test_split(data2, test_size= 0.2, random_state = 42)

knn_model = KNeighborsClassifier( n_neighbors= 5)


knn_model.fit(train_df[["current_price_new","quantity_new","cate_freq_new","n_review_new","avg_rating_new","manufacturer_new","KQ"]],train_df["PhanLoai"])

knn_prediction =
knn_model.predict(test_df[["current_price_new","quantity_new","cate_freq_new","n_review_new","avg_rating_new","manufacturer_new","KQ"]])


print("Confusion matrix: ")

print(confusion_matrix(test_df["PhanLoai"],knn_prediction))

print("KNN Classification Report: ")

print(classification_report(test_df["PhanLoai"],knn_prediction))
```

```
Confusion matrix:
[[161   0   0   0]
 [  0  29   3   0]
 [  0   1 108   0]
 [  0   0   0  52]]
KNN Classification Report:
              precision    recall  f1-score   support

    Bán chạy       1.00      1.00      1.00       161
      Bán tệ       0.97      0.91      0.94        32
     Bán vừa       0.97      0.99      0.98       109
  Top Seller       1.00      1.00      1.00        52

    accuracy                           0.99       354
   macro avg       0.98      0.97      0.98       354
weighted avg       0.99      0.99      0.99       354
```

Nhận xét:

- Kết quả trên cho ta thấy được phân lớp Top Seller và Bán chạy có precision, recall và f1-score bằng 1 là quá lí tưởng, không bỏ sót sách nào lọt qua phân lớp khác.
- Phân lớp Bán vừa có precision 0.97, recall 0.99, f1-score 0.98 cho thấy có thể có sách thuộc nhóm Bán vừa sẽ bị lọt sang nhóm khác, ví dụ như lọt sang Bán tệ thì sách đó sẽ không được độc giả chú ý đến và mua thậm chí bị độc giả bỏ qua không mua. Nhưng do recall 0.99 gần 1 nên việc bỏ sót qua sách Bán vừa thấp hơn nhiều.
- Phân lớp Bán tệ có precision 0.97, recall 0.91, f1-score 0.94. Giống như trên, sách ở nhóm Bán tệ có thể lọt sang các nhóm khác, nhưng do recall chỉ có 0.91 nên việc bỏ sót nó qua nhóm sách khác sẽ có tỉ lệ cao hơn. Nhưng việc sách Bán tệ xuất hiện ở các nhóm sách khác thì sẽ thu hút được độc giả mua sách đó hơn, nhằm có thể đẩy đi được hàng tồn, có thể tăng được doanh thu nếu bán được số lượng nhiều.
- Nhưng do chỉ lấy 20% đầu để test nên có thể sẽ có sai số trong việc phân loại các nhóm sách. Ngoài ra, dữ liệu có những giá trị chênh lệch nhiều hơn so với các giá trị còn lại do việc gom gọn dữ liệu nên sẽ không thể tránh khỏi sai sót.

```
# Sử dụng phương pháp Navie-Bayes
nb_model = GaussianNB()


nb_model.fit(train_df[["current_price_new","quantity_new","cate_freq_new",
"n_review_new","avg_rating_new","manufacturer_new","KQ"]], train_df["PhanLoai"])


nb_predicious =
nb_model.predict(test_df[["current_price_new","quantity_new","cate_freq_new",
```

```
"n_review_new","avg_rating_new","manufacturer_new","KQ"]])


print("Confusion matrix: ")
print(confusion_matrix(test_df["PhanLoai"],nb_predicious))
print("Navie-Bayes Classitification Report: ")
print(classification_report(test_df["PhanLoai"], nb_predicious))
```

```
Confusion matrix:
[[158   3   0   0]
 [  0  27   5   0]
 [  8   6  95   0]
 [  1   0   0  51]]
Navie-Bayes Classitification Report:
              precision    recall  f1-score   support

     Bán chạy       0.95      0.98      0.96       161
       Bán tệ       0.75      0.84      0.79        32
     Bán vừa        0.95      0.87      0.91       109
   Top Seller       1.00      0.98      0.99        52

    accuracy                           0.94       354
   macro avg        0.91      0.92      0.91       354
weighted avg        0.94      0.94      0.94       354
```

Nhận xét:

- Ta có thể thấy việc sử dụng thuật toán Naive Bayes sẽ đơn giản hơn, mô hình tiên đoán cũng gần như chính xác hơn vì nó thực hiện được trên bộ dữ liệu lớn với các vấn đề phức tạp.

- ***Bước 4: Tạo file csv cuối với các Lớp đã được phân:***

```
import pandas as pd

data = pd.read_csv('/content/drive/MyDrive/Nam 3/KPDL/Tiki/PhanLoaiFinal.csv')

books = pd.read_csv('/content/drive/MyDrive/Nam 3/KPDL/Tiki/Bản sao của books.csv')



im = books[books['product_id'] == 75307228]

booksFinal = books.merge(data[['product_id', 'PhanLoai']], on='product_id')



df2 = {'Unnamed: 0':133 , 'product_id':75307228, 'title' : 'OSHO - Yêu - Being In Love',
'authors' :'Osho', 'original_price':168000 , 'current_price':110800, 'quantity':671121 ,'category':
'Sách nghệ thuật sống đẹp',
```

```
'n_review': 1855, 'avg_rating': 5.0, 'pages':350, 'manufacturer': 'Nhà Xuất Bản Văn Hóa - Văn Nghệ TP.HCM',
'cover_link':'https://salt.tikicdn.com/ts/product/5e/18/24/2a6154ba08df6ce6161c13f4303fa19e.jpg', 'PhanLoai': 'Top Seller'}


booksFinal = booksFinal.append(df2, ignore_index = True)

booksFinal.to_csv('booksFinal.csv')
```

Chương 4. KẾT LUẬN

4.1. Kết quả đạt được của đề tài

Hoàn thành việc khai phá dataset Tiki Books theo hướng tìm xem cuốn sách nào sẽ có cơ hội được mua cao hơn. Chia ra được 4 phân lớp cho dataset là bán chạy, bán vừa, bán tệ và top seller. Ngoài ra, việc khai phá dataset này giúp nhóm chúng em củng cố thêm kiến thức cho môn Khai phá dữ liệu như cách gom cụm dữ liệu bằng phương pháp K-means, tính khoảng cách Manhattan, phân lớp dữ liệu bằng phương pháp KNN và Naïve-Bayes.

4.2. Hạn chế của đề tài

Chưa khai phá được sâu hơn dataset vì nhóm chúng em mới chỉ dừng ở việc tìm sách bán chạy. Ngoài ra, hai hướng khác để nghiên cứu thì nhóm chưa khai thác được.

4.3. Phát triển đề tài trong tương lai

Trong tương lai sẽ nghiên cứu thêm về hai hướng còn lại của dataset Tiki Books.

BẢNG PHÂN CÔNG

Tên Thành Viên	**MSSV**	**Nhiệm Vụ**	**Hoàn Thành**	**Ghi Chú**
Nguyễn Đặng Tuyết Nhi	2051050172	Làm sạch dữ liệu, chuẩn hóa dữ liệu, nhận xét biểu đồ, tìm hướng chia phân lớp, báo cáo đề tài.	100%	
Lê Nguyễn Tiến Vững	2051050320	Chuẩn hóa dữ liệu, gom cụm và phân lớp dữ liệu, báo cáo đề tài.	100%	
Bảo Khiêm	2051052062	Chuẩn hóa dữ liệu, nhận xét biểu đồ, tìm hướng phân lớp, báo cáo đề tài.	100%	

Bảng 1. Bảng phân công