ỦY BAN NHÂN DÂN THÀNH PHỐ HỒ CHÍ MINH

TRƯỜNG ĐẠI HỌC SÀI GÒN

KHOA CÔNG NGHỆ THÔNG TIN

-----◆◆◆-----

ĐỀ TÀI ĐỒ ÁN: PHÂN TÍCH DỮ LIỆU CÁC VIDEO THỊNH HÀNH TRÊN NỀN TẢNG YOUTUBE

Sinh viên thực hiện: **Trịnh Vĩnh Phúc**

Mã số sinh viên: **3119410318**

Học phần: **Khai phá dữ liệu**

Giảng viên hướng dẫn: **TS. Vũ Ngọc Thanh Sang**

Thành phố Hồ Chí Minh, tháng 04 năm 2023

MỤC LỤC

NHẬN XÉT CỦA GIẢNG VIÊN HƯỚNG DẪN

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

Danh mục các bảng

Danh mục sơ đồ hình ảnh

LỜI MỞ ĐẦU

Công nghệ ngày càng phổ biến và không ai có thể phủ nhận được tầm quan trọng và những hiệu quả mà nó đem lại cho cuộc sống chúng ta. Sự phát triển nhanh chóng của mạng Internet đã sinh ra một khối lượng khổng lồ các dữ liệu dạng siêu văn bản. Cùng với sự thay đổi và phát triển hàng ngày, hàng giờ về nội dung cũng như số lượng các trang web trên Internet thì vấn đề tìm kiếm thông tin đối với người sử dụng lại vô cùng khó khăn. Có thể nói nhu cầu tìm kiếm thông tin trên một cơ sở dữ liệu phi cấu trúc đã được phát triển mạnh mẽ cùng với sự bành trướng của Internet. Thật vậy, với Internet, con người đã dần làm quen với các trang web cùng với vô văn các thông tin. Trong những năm gần đây Internet đã trở thành một trong những kênh về khoa học, thông tin kinh tế, thương mại và quảng cáo chính, ảnh hưởng đến đời sống, kinh thế, giáo dục và ngay cả chính trị. Có thể nói, Internet như là cuốn từ điển Bách khoa toàn thư với thông tin đa dạng về mặt nội dung cũng như hình thức được trình bày dưới dạng văn bản, hình ảnh, âm thanh,...

Tuy nhiên, cùng với sự đa dạng và số lượng lớn thông tin như vậy, Internet đã nảy sinh vấn đề quá tải thông tin. Sự tích lũy dữ liệu này xảy ra với một tốc độ bùng nổ. Người ta ước tính rằng lượng thông tin trên toàn cầu tăng gấp đôi sau khoảng hai năm và theo đó số lượng cũng như kích cỡ của các cơ sở dữ liệu cũng tăng lên một cách nhanh chóng. Khai phá dữ liệu ra đời như một hướng giải quyết hữu hiệu cho vấn đề quá tải thông tin và xử lý thông tin nhiễu loạn. Khai phá dữ liệu là một tập hợp các kỹ thuật được sử dụng để tự động khai thác và tìm ra các mối quan hệ lẫn nhau của dữ liệu trong một tập hợp dữ liệu khổng lồ và phức tạp, đồng thời cũng tìm ra các mẫu tiềm ẩn trong tập dữ liệu đó.

Em xin cảm ơn TS. Vũ Ngọc Thanh Sang – giảng viên trường Đại học Sài Gòn khoa Công nghệ thông tin đã tạo điều kiện cho em có cơ hội thực hiện và tận tình giúp đỡ em hoàn thành dự án này, qua đó gặt hái được những kinh nghiệm và kỹ năng quý báu song hành với những kiến thức đã được học ở trường. Tuy nhiên, trong quá trình học tập, em sẽ không thể tránh khỏi sai sót nhưng sẽ cố gắng cải thiện và tiếp thu ý kiến từ mọi người để em có thêm động lực tiếp tục trên con đường chinh phục công nghệ.

CHƯƠNG 1: GIỚI THIỆU ĐỀ TÀI

1.1. Tìm hiểu đề tài

YouTube là một nền tảng chia sẻ video trực tuyến của Mỹ có trụ sở chính tại San Bruno, California. Nền tảng này được tạo ra vào tháng 2 năm 2005 và đã được Google mua lại vào tháng 11 năm 2006 với giá 1,65 tỷ đô la Mỹ và hiện hoạt động như một trong những công ty con của Google. YouTube là trang web được truy cập nhiều thứ hai sau Google Tìm kiếm.

Với sức ảnh hưởng và mức độ phủ sóng mạnh mẽ, YouTube đã dần chiếm lĩnh thị trường Hoa Kỳ. Tại Mỹ, việc tìm kiếm, xem - nghe video trên YouTube đã trở thành thói quen hằng ngày của rất nhiều người, đồng thời cũng không nằm ngoài xu thế chung trên thế giới. Sức ảnh hưởng này còn nằm ở bảng xếp hạng các video xu hướng nhất của YouTube. YouTube duy trì danh sách các video thịnh hành nhất xét theo từng mốc thời gian nhất định. Theo tạp chí Variety, "Để xác định các video thịnh hành nhất trong năm, YouTube sử dụng kết hợp nhiều yếu tố bao gồm đo lường tương tác của người dùng (số lượt xem, lượt chia sẻ, nhận xét và lượt thích)".

Ở đề tài này, chúng ta sẽ tìm hiểu cách để một video trở nên thịnh hành dựa trên những thông số tương tác của người dùng trên video đó.

1.2. Mục đích đề tài

Dựa và các phương pháp khai phá và phân tích dữ liệu, chúng ta sẽ tìm ra mối tương quan giữa các yếu tố (views, likes, dislikes) trong việc xác định video nào có thể trở thành xu hướng. Chúng ta tiến hành phân tích sự liên kết và quan hệ giữa thuật toán thịnh hành của YouTube với những thông số tương tác của người dùng (lượt xem, số lượng thích, số lượng không thích,...).

1.3. Phạm vi đề tài

Phạm vi của đề tài là dữ liệu của các video thịnh hành trên YouTube tại Hoa Kỳ từ ngày 14/11/2017 đền ngày 14/06/2018.

Dữ liệu bao gồm các thuộc tính cơ bản của một video trên YouTube: Chỉ mục video, ngày thịnh hành, tiêu đề, ngày xuất bản, số lượng lượt xem, số lượng yêu thích, số lượng không thích, số lượng bình luận, mô tả video.

CHƯƠNG 2: MÔ TẢ BỘ DỮ LIỆU

2.1. Nguồn gốc dữ liệu

Tập dữ liệu này là bản ghi từ ngày 14/11/2017 đến ngày 14/06/2018 về các video thịnh hành nhất trên YouTube tại thị trường Hoa Kỳ. Nguồn từ website:

https://www.kaggle.com/datasets/datasnaek/youtube-new?select=USvideos.csv

Để có được dữ liệu trending video từ YouTube này, ta có thể sử dụng các thư viện hoặc công cụ hỗ trợ như Google API, YouTube API,…. Dưới đây là các bước để crawl dữ liệu trending video từ YouTube:

- Đăng ký tài khoản Google API và YouTube API và lấy API key.
- Sử dụng API key để truy cập dữ liệu từ YouTube API, chẳng hạn như danh sách các video đang hot, thông tin về kênh, bình luận, thẻ,…
- Sử dụng Scrapy hoặc BeautifulSoup để crawl dữ liệu từ trang web của YouTube, chẳng hạn như tiêu đề video, tác giả, số lượt xem, số lượt like, số lượt dislike, thời gian đăng, thời gian cập nhật,…
- Lưu trữ dữ liệu được crawl vào một tệp hoặc cơ sở dữ liệu để phân tích và xử lý dữ liệu sau.

2.2. Kích thước bộ dữ liệu

Tập dữ liệu bao gồm 40949 hàng và 16 thuộc tín. Trong đó, các thuộc tính gồm có:

- *video_id*: Chỉ mục video
- *trending_date*: Ngày thịnh hành của video
- *title*: Tiêu đề video
- *channel_title*: Tên kênh sở hữu
- *category_id*: Chỉ mục danh mục
- *publish_time*: Thời gian công khai
- *tags*: Các nhãn video
- *views*: Số lượng xem
- *likes*: Số lượng thích
- *dislikes*: Số lượng không thích
- *comment_count*: Số lượng bình luận
- *thumbnail_link*: Địa chỉ hình ảnh thu nhỏ
- *comments_disabled*: Vô hiệu bình luận
- *ratings_disabled*: Vô hiệu đánh giá
- *video_error_or_removed*: Video lỗi hoặc bị gỡ
- *description*: Mô tả video

CHƯƠNG 3: TIỀN XỬ LÝ DỮ LIỆU

3.1. Sử dụng những thư viện và tệp tin dữ liệu

Những thư viện được sử dụng trong dự án:

- *numpy*: bổ sung hỗ trợ cho các mảng lớn, đa chiều, cùng và một bộ các hàm toán học cấp cao.
- *pandas*: bổ sung các thao tác phân tích dữ liệu, cấu trúc dữ liệu và các phép toán để thao tác với các bảng số và chuỗi thời gian.
- *matplotlib.pyplot*: trực quan hóa dữ liệu và vẽ đồ thị.
- *seaborn*: xây dựng những hình ảnh trực quan đẹp mắt. Nó có thể được coi là một phần mở rộng của một thư viện khác có tên là Matplotlib.
- *pca*: giảm chiều và tăng khả năng trực quan hóa dữ liệu.

```
[ ]  import numpy as np
     import pandas as pd
     import matplotlib.pyplot as plt
     import seaborn
     import pca
```

Hình 3.1 Chèn các thư viện được sử dụng

Trong dự án này, chúng ta sử dụng Google Colab, là một sản phẩm từ Google Research, nó cho phép chạy các dòng code python thông qua trình duyệt, đặc biệt phù hợp với Data analysis, machine learning và giáo dục. Để liên kết với Google Drive nhằm dễ dàng xử lý tập tin ngay trên Drive, ta sử dụng câu lệnh:

```
from google.colab import drive
drive.mount('/content/drive')
```

Tạo đường dẫn đến tệp dữ liệu (“USvideos.csv”) và đọc file csv truyền vào:

```
path_data = '/content/drive/MyDrive'
df = pd.read_csv(f'{path_data}/USvideos.csv', header=0, index_col=0)
```

Hiển thị thông tin các thuộc tính của tập dữ liệu:

```
[5]  #chi tiết của dataframe
     df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 40949 entries, 2kyS6SvSYSE to ooyjaVdt-jA
Data columns (total 15 columns):
 #   Column                  Non-Null Count  Dtype
---  ------                  --------------  -----
 0   trending_date           40949 non-null  object
 1   title                   40949 non-null  object
 2   channel_title           40949 non-null  object
 3   category_id             40949 non-null  int64
 4   publish_time            40949 non-null  object
 5   tags                    40949 non-null  object
 6   views                   40949 non-null  int64
 7   likes                   40949 non-null  int64
 8   dislikes                40949 non-null  int64
 9   comment_count           40949 non-null  int64
 10  thumbnail_link          40949 non-null  object
 11  comments_disabled       40949 non-null  bool
 12  ratings_disabled        40949 non-null  bool
 13  video_error_or_removed  40949 non-null  bool
 14  description             40379 non-null  object
dtypes: bool(3), int64(5), object(7)
memory usage: 4.2+ MB
```

Hình 3.2 Thông tin chi tiết của bộ dữ liệu

3.2. Loại bỏ các trường thông tin không cần thiết

Sau khi phân tích, ta nhận thấy các trường thông tin không cần thiết:

- *category_id*
- *thumbnail_link*
- *comment_count*
- *tags*
- *ratings_disabled*
- *video_error_or_removed*
- *comments_disabled*
- *channel_title*
- *publish_time*

Chúng ta không sử dụng toàn bộ các biến đầu vào mà cần lọc ra những trường cần thiết cho việc phân tích và nghiên cứu. Những trường dữ liệu trên không có chức năng trong quá trình phân tích và không đóng vai trò quan trọng. Ta có thể loại bỏ chúng nhằm hoàn thiện dữ liệu, tiêu tốn ít bộ nhớ lưu trữ và giảm thời gian huấn luyện.

Việc giảm chiều dữ liệu từ không gian cao chiều xuống không gian thấp chiều như trên vẫn giữ được những đặc trưng chính của dữ liệu nhưng có thể tiết kiệm được chi phí huấn luyện và dự báo.

Thông qua đó, chúng ta nhận thấy được mức độ quan trọng của các trường dữ liệu:

- video_id
- trending_date
- title
- tags
- views
- likes
- dislikes

```
[ ]  df = df.drop('category_id', axis='columns')
     df = df.drop("thumbnail_link", axis='columns')
     df = df.drop("comment_count", axis='columns')
     df = df.drop("tags", axis='columns')
     df = df.drop("ratings_disabled", axis='columns')
     df = df.drop("video_error_or_removed", axis='columns')
     df = df.drop("comments_disabled", axis='columns')
     df = df.drop("channel_title", axis='columns')
     df = df.drop("publish_time", axis='columns')
     df.head()
```

Hình 3.3 Xóa các cột không cần thiết trong bộ dữ liệu

3.3. Xử lý dữ liệu thiếu

Nhằm tạo nên sự hoàn thiện của dữ liệu, ta cần tìm những dữ liệu bị thiếu. Trong tập dữ liệu trên, chúng ta có thể thấy có những dữ liệu thiếu xuất hiện dưới dạng dữ liệu trống (`null`). Cách tốt nhất để giải quyết dữ liệu dạng này là hàm `isnull()` và hàm `dropna()`.

Để xác định trong tập dữ liệu có xuất hiện dữ liệu `null` hay không, ta có thể sử dụng hàm `isnull()` để tìm kiếm:

```
[ ]  df.isnull().sum()

trending_date      0
title              0
views              0
likes              0
dislikes           0
description      570
dtype: int64
```

Hình 3.4 Hiển thị số lượng dữ liệu trống

Sau khi đã tìm được những trường dữ liệu `null`, ta tiến hành loại bỏ chúng khỏi tập dữ liệu bằng hàm `dropna()`:

```
[ ]  df = df.dropna()
     df.isnull().sum()
```

```
trending_date    0
title            0
views            0
likes            0
dislikes         0
description      0
dtype: int64
```

Hình 3.5 Hiển thị lại số lượng dữ liệu trống sau khi loại bỏ thành công

3.4. Kiểm tra các dữ liệu trùng

Trong quá trình làm sạch dữ liệu, việc loại bỏ dữ liệu trung là vô cùng cần thiết. Việc này giúp tăng tính chính xác và hiệu quả cho việc phân tích dữ liệu. Loại bỏ dữ liệu trùng lặp cũng giảm bớt chi phí cho việc xử lý và huấn luyện sau này.

```
[ ]  #Kiểm tra các dữ liệu trùng nhau
     #Có 47 dòng dữ liệu trùng nhau
     Dup_Rows = df[df.duplicated()]
     Dup_Rows.count()
```

```
trending_date    47
title            47
views            47
likes            47
dislikes         47
description      47
dtype: int64
```

Hình 3.6 Hiển thị số lượng dữ liệu trùng nhau

Sau khi sử dụng hàm `duplicated()` để tìm kiếm những hàng có dữ liệu trùng lặp với nhau, kết quả cho ta thấy trong tập dữ liệu có 47 dòng trùng với nhau. Để xử lý việc này vô cùng đơn giản, ta có thể sử dụng `drop_duplicates(keep='first')`. Trong đó, thông số `keep='first'` nhằm giữ lại dòng trùng lặp đầu tiên xóa tất cả hàng còn lại.

```
[ ]  df_count = df.count()
     DF_RM_DUP = df.drop_duplicates(keep='first')
     df_count_remove_duplicate = DF_RM_DUP.count()

     print(f"Số dòng Dataframe trước loại bỏ Duplicate: {df_count}")
     print(f"Số dòng Dataframe sau loại bỏ Duplicate: {df_count_remove_duplicate}")
```

```
Số dòng Dataframe trước loại bỏ Duplicate: trending_date    40379
title            40379
views            40379
likes            40379
dislikes         40379
description      40379
dtype: int64
 Số dòng Dataframe sau loại bỏ Duplicate: trending_date    40332
title            40332
views            40332
likes            40332
dislikes         40332
description      40332
dtype: int64
```

Hình 3.7 Xóa dữ liệu trùng lặp

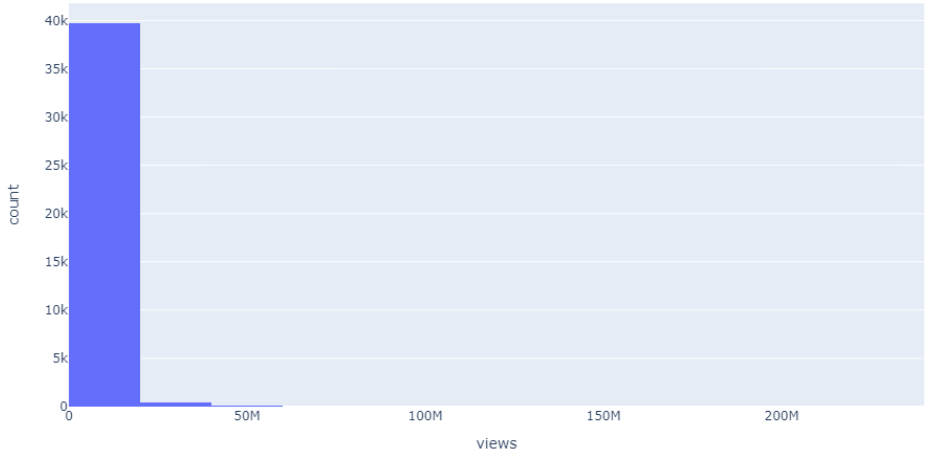
Sau khi xóa bỏ dữ liệu trùng, tập dữ liệu giảm từ 40379 dòng xuống 40332 dòng. Dữ liệu đã được làm sạch tương đối, tiếp theo ta cần xử lý dữ liệu ngoại lai và dữ liệu nhiễu.

3.5. Xử lý dữ liệu ngoại lai

Ở quá trình này, ta sử dụng thư viện đồ họa mới plotly.express. Thư viện này giúp tạo ra các biểu đồ tương tác, đồ thị chất lượng cao trong xử lý dữ liệu. Việc xử lý giá trị ngoại lại là xác định và loại bỏ các giá trị khác xa với phần còn lại của các giá trị trong trường đó. Các giá trị này có tần xuất xảy ra vô cùng thấp trong cột dữ liệu. Đây chính là dữ liệu ngoại lai. Ta xác định những trường dữ liệu ta cần xử lý bao gồm: “views”, “likes” và “dislikes”.

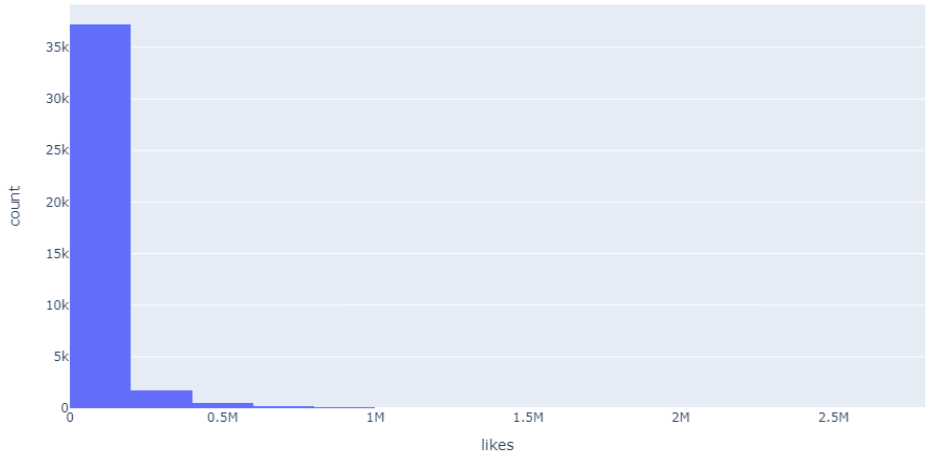
Đầu tiên, ta mô hình hóa những trường “views”, “likes” và “dislikes” để tìm ra những giới hạn giá trị có tần tần xuất xảy ra thấp nhất:

```
fig = px.histogram(df,x="views", nbins=20)
fig.show()
```



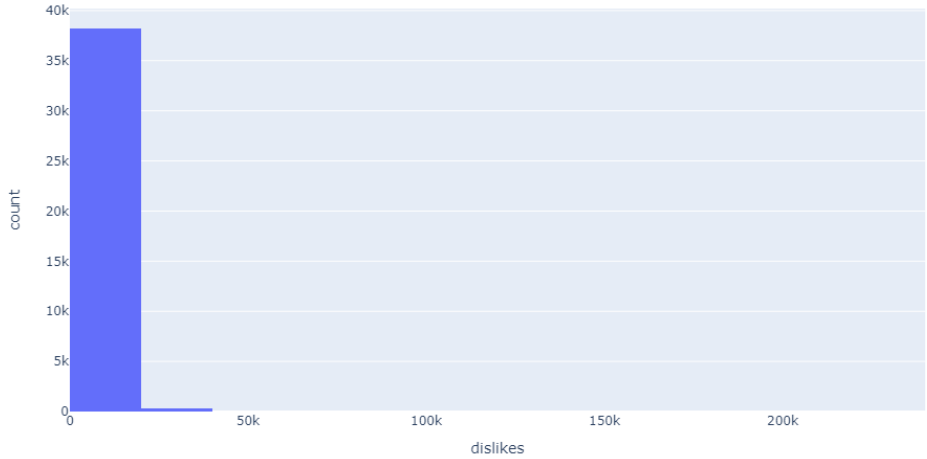
Hình 3.8 Biểu đồ số lượng view của bộ dữ liệu

```
fig = px.histogram(df,x="likes",nbins = 20)
fig.show()
```



Hình 3.9 Biểu đồ số lượng like của bộ dữ liệu

```
fig = px.histogram(df,x="dislikes", nbins=20)
fig.show()
```



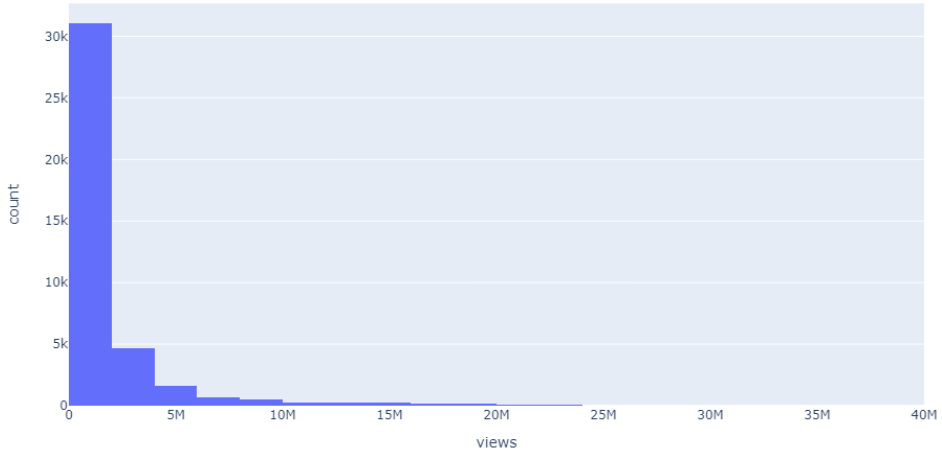
Hình 3.10 Biểu đồ số lượng dislike của bộ dữ liệu

Từ những mô hình trên, ta thấy được giới hạn ngoại lai của từng trường dữ liệu:

- views: 40 000 000
- likes: 400 000
- dislikes: 39 000

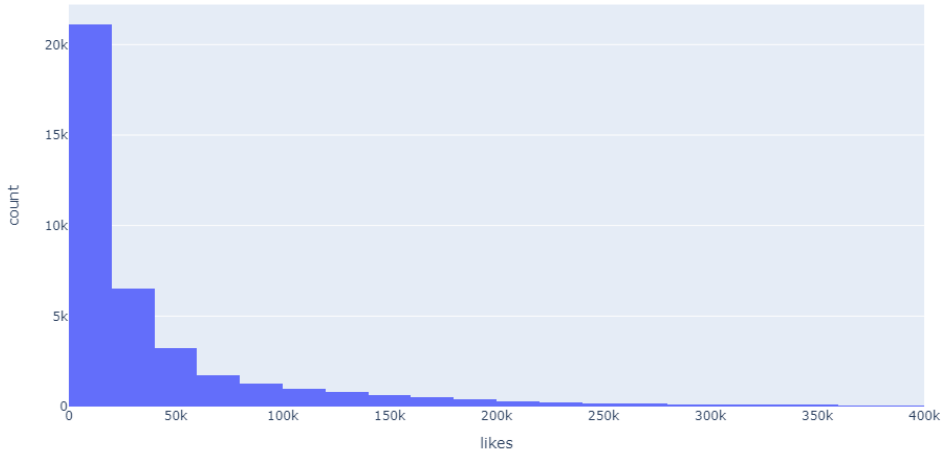
Tiếp theo, ta loại bỏ những dữ liệu nằm ngoài giới hạn ngoại lai:

```
df = df.drop(df[df.views >=40000000].index)
fig = px.histogram(df,x="views", nbins=20)
fig.show()
```



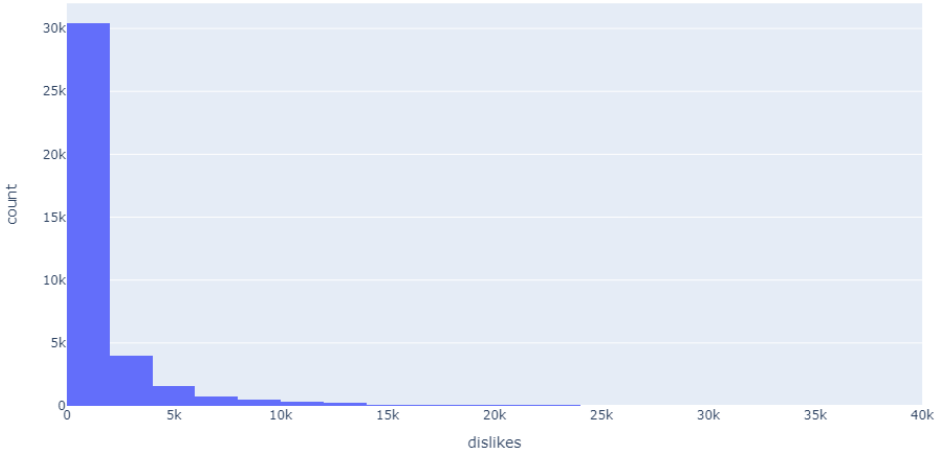
Hình 3.11 Biểu đồ số lượng view sau khi xử lý ngoại lai

```
df = df.drop(df[df.likes >=400000].index)
fig = px.histogram(df,x="likes", nbins = 20)
fig.show()
```



Hình 3.12 Biểu đồ số lượng like sau khi xử lý ngoại lai

```
df = df.drop(df[df.dislikes >=39000].index)
fig = px.histogram(df,x="dislikes", nbins = 20)
fig.show()
```



Hình 3.13 Biểu đồ số lượng dislike sau khi xử lý ngoại lai

3.6. Xử lý dữ liệu nhiễu

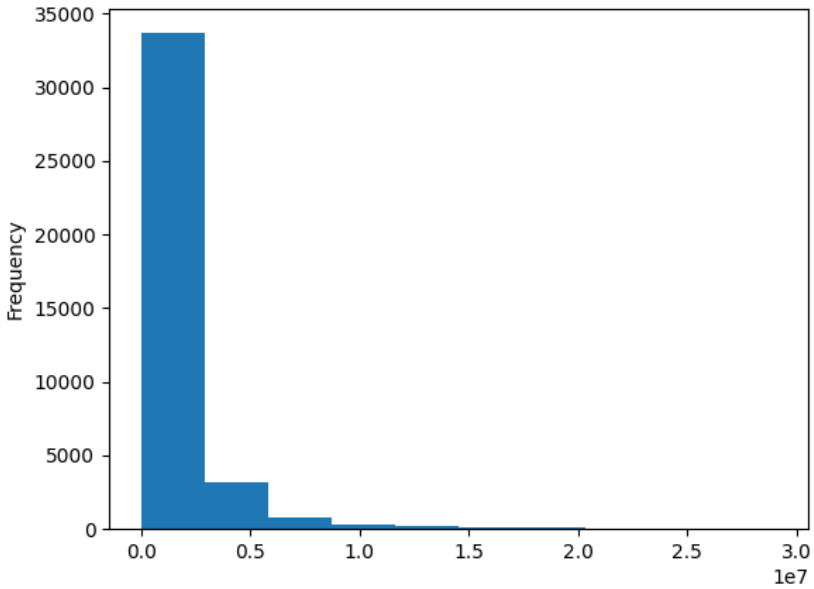
Phương pháp bining data là một kỹ thuật xử lý dữ liệu nhiễu bằng cách chia dữ liệu thành các phân khúc rời rạc và đặt chúng vào các nhóm (bins) tương ứng. Việc này giúp giảm thiểu ảnh hưởng của dữ liệu nhiễu lên quá trình phân tích và giúp tăng độ chính xác trong việc đưa ra quyết định.

Có thể áp dụng phương pháp bining data cho các biến số liên tục hoặc phân loại. Để thực hiện phương pháp này, trước tiên cần xác định số lượng các nhóm cần tạo ra. Số lượng nhóm này có thể được chọn dựa trên kinh nghiệm hoặc các phương pháp thống kê như phân phối tần suất của dữ liệu. Sau đó, dữ liệu sẽ được phân bố vào các nhóm tương ứng.

Việc chọn số lượng và kích thước của các nhóm cũng là một yếu tố quan trọng trong phương pháp bining data. Nếu chọn quá nhiều nhóm, dữ liệu sẽ bị chia nhỏ và không còn có tính đại diện, trong khi chọn quá ít nhóm thì sẽ mất mát thông tin và không thể phân tích chi tiết hơn.

Đối với tập dữ liệu này, ta cần phân tích ba biến số: “views”, “likes” và “dislike”. Trước tiên, ta phân tích cột “views”.

```
df['views'].plot.hist(bins=10)
```



Hình 3.14 Biểu đồ tần số cột views

```
maxrange = int(np.ceil(max(df['views'])))
minrange = int(np.floor(min(df['views'])))
ageRange = maxrange - minrange
bins = 10
binwidth = int(np.round(ageRange/bins))

intervals = [ views for views in range(minrange, maxrange + binwidth,
binwidth)]
binlabels = ["bin" + str(i) for i in range (1, int(len(intervals)))]

df['Views_Cut'] = pd.cut(df['views'], bins = intervals, labels = None,
include_lowest=True)

df.groupby('Views_Cut')['views'].count().plot.bar()
plt.xticks(rotation=52)
plt.ylabel('observation count')
```

Ở trên, ta thực hiện tính toán giá trị cận trên của phạm vi "views" bằng cách lấy giá trị tối đa của cột "views" và làm tròn lên đến số nguyên gần nhất. Tương tự, ta tính toán giá trị cận dưới của phạm vi "views" bằng cách lấy giá trị tối thiểu của cột "views" và làm tròn xuống đến số nguyên gần nhất.

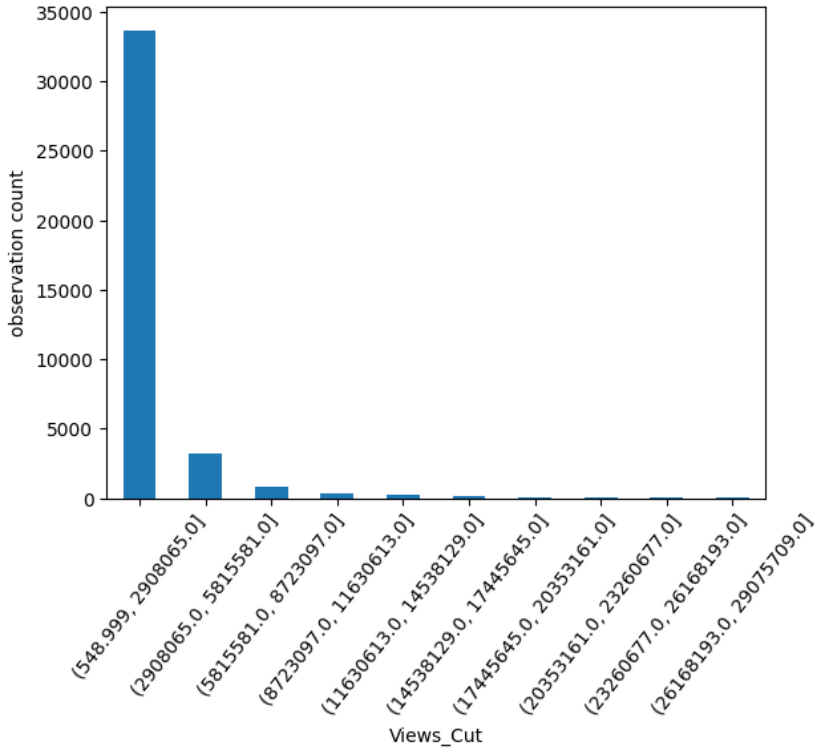
Sau đó, mã tính toán phạm vi "views" bằng cách tính hiệu giữa cận trên và cận dưới của phạm vi. Biến bins được đặt bằng 10 để chỉ định số lượng khoảng chia phạm

vi "views". Mã tính toán độ rộng của mỗi khoảng bằng cách chia phạm vi "views" cho số lượng khoảng và làm tròn đến số nguyên gần nhất.

Sau đó, mã tạo ra một danh sách intervals các khoảng bằng cách sử dụng hàm `range()` từ giá trị cận dưới đến giá trị cận trên của phạm vi "views", với độ rộng của mỗi khoảng là binwidth.

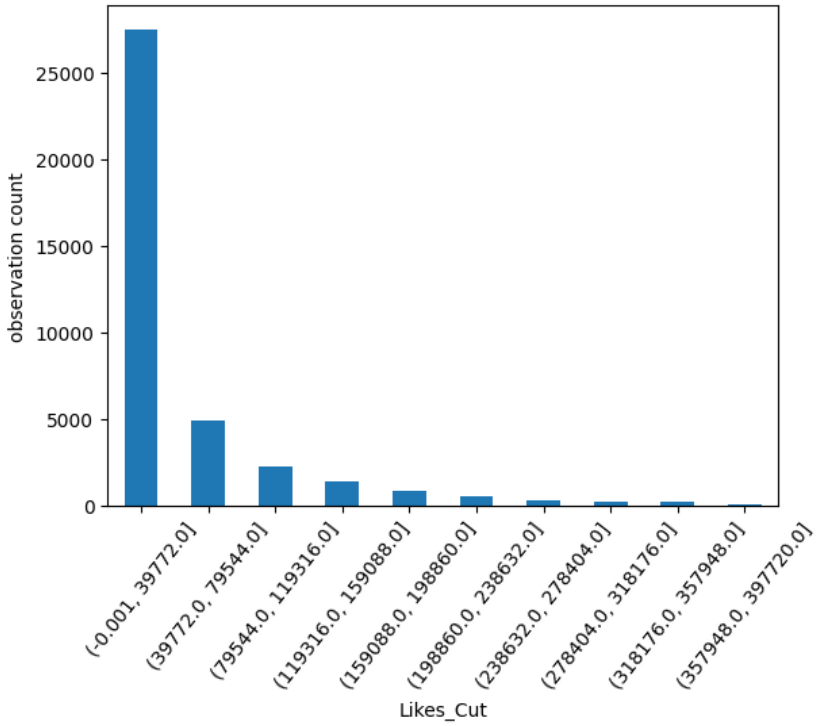
Ta tạo ra một cột mới trong DataFrame df có tên là "Views_Cut" bằng cách sử dụng hàm `pd.cut()` để chia cột "views" thành các khoảng bằng với các khoảng được xác định trước đó bằng intervals, với các nhãn được xác định bằng binlabels.

Cuối cùng, mã sử dụng hàm `groupby()` để nhóm dữ liệu theo cột "Views_Cut" và đếm số lượng quan sát trong mỗi khoảng.

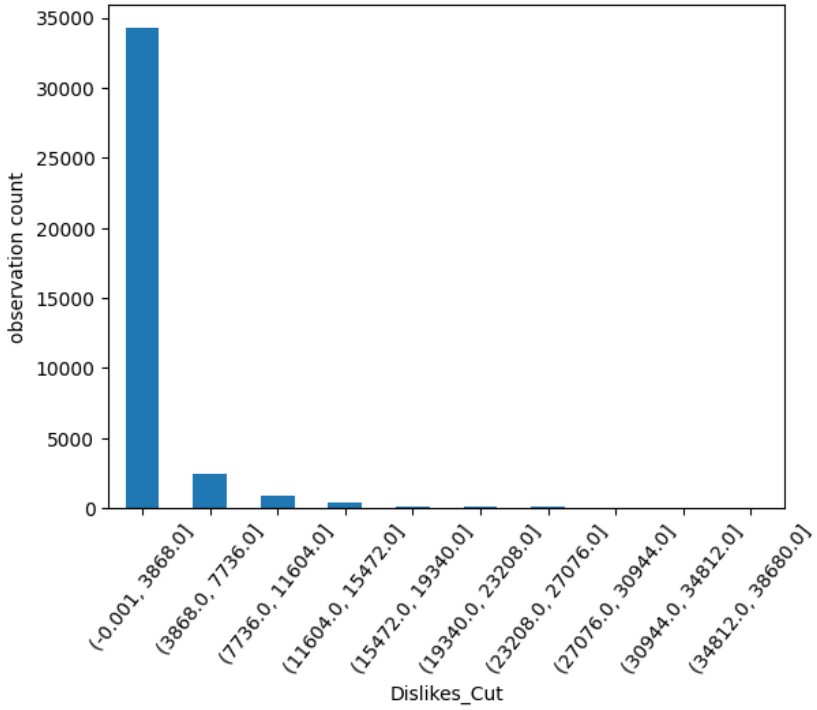


Hình 3.15 Biểu đồ tần số cột views sau khi xử lý dữ liệu nhiễu

Tương tự đối với hai cột “like” và “dislike”, ta tiếp tục phân chia dữ liệu và nhóm chúng theo các bins. Ta được:



Hình 3.16 Biểu đồ tần số cột likes sau khi xử lý dữ liệu nhiễu



Hình 3.17 Biểu đồ tần số cột dislike sau khi xử lý dữ liệu nhiễu

Sau khi đã hoàn thành việc phân nhóm dữ liệu, ta xóa các cột “Views_Cut”, “Dislikes_Cut” và “Likes_Cut” để làm cho DataFrame nhỏ hơn và dễ quản lý hơn.

```
df = df.drop('Dislikes_Cut', axis='columns')
df = df.drop('Views_Cut', axis='columns')
df = df.drop('Likes_Cut', axis='columns')
```

Trên là quá trình tiền xử lý dữ liệu, quá trình này là quá trình quan trọng trong khai phá dữ liệu, giúp chuẩn bị dữ liệu trước khi được áp dụng các phương pháp phân tích. Quá trình này nhằm loại bỏ các giá trị nhiễu, các giá trị bị thiếu, chuẩn hóa các giá trị dữ liệu, rút trích các đặc trưng quan trọng của dữ liệu và chuyển đổi chúng thành các định dạng phù hợp cho việc phân tích.

Việc thực hiện các bước tiền xử lý dữ liệu đảm bảo rằng dữ liệu đã được sạch và phù hợp để sử dụng cho các mô hình phân tích và khai thác dữ liệu.

CHƯƠNG 4: PHÂN TÍCH DỮ LIỆU KHÁM PHÁ

4.1. Phân tích đơn biến

Các đặc tính mô tả của dữ liệu là các thống kê mô tả mà được sử dụng để tóm tắt và mô tả các tính chất của dữ liệu. Các đặc tính mô tả này giúp ta hiểu được cách dữ liệu được phân bố và hình thành trong tập dữ liệu, từ đó có thể rút ra những kết luận và quyết định phù hợp.

Các đặc tính mô tả thường được sử dụng bao gồm:

- Giá trị trung bình.
- Giá trị trung vị.
- Độ lệch chuẩn.
- Phân vị (biểu diễn dưới dạng biểu đồ line).

4.1.1. Thống kê giá trị trung bình

```
[43] df.mean(axis=0, numeric_only=True)

views       1.392780e+06
likes       4.096883e+04
dislikes    1.703419e+03
dtype: float64
```

Hình 4.1 Giá trị trung bình của các cột thuộc tính

Đây là giá trị trung bình của ba thuộc tính trong tập dữ liệu (views, likes, dislikes). Trung bình được sử dụng để biểu thị mức độ trung thành của dữ liệu đối với một số giá trị cụ thể:

- Views: 1.392780×10^6
- Likes: 4.096883×10^4
- Dislikes: 1.703419×10^3

4.1.2. Thống kê giá trị trung vị

```
[44] df.median(axis=0, numeric_only=True)

views       629429.0
likes        16667.0
dislikes       585.0
dtype: float64
```

Hình 4.2 Giá trị trung vị của các cột thuộc tính

Đây là giá trị ở giữa tập dữ liệu, nghĩa là có nửa số giá trị trong tập dữ liệu lớn hơn trung vị và nửa còn lại nhỏ hơn:

- Views: 629429.0

- Likes: 16667.0
- Dislikes: 585.0

4.1.3. Độ lệch chuẩn của các dữ liệu

Là căn bậc hai của phương sai, đây là một độ đo phổ biến để đo sự phân tán của dữ liệu. Sử dụng phương thức std() trong thư viện Numpy của Python để tính độ lệch chuẩn.

```
[45] df.std(axis=0, numeric_only=True)

views       2.271627e+06
likes       6.145914e+04
dislikes    3.384368e+03
dtype: float64
```

Hình 4.3 Độ lệch chuẩn của các cột thuộc tính

- Views: 2.271627×10^6
- Likes: 6.145914×10^4
- Dislikes: 3.384368×10^3

4.1.4. Phân vị

Phân vị được sử dụng để phân loại các giá trị trong tập dữ liệu. Nó đại diện cho một phần trăm của các giá trị trong tập dữ liệu, với ý nghĩa là giá trị đó bé hơn hoặc bằng phần trăm đó các giá trị khác trong tập dữ liệu.

index	views	likes	dislikes
count	38467	38467	38467
mean	1392780.363	40968.83407	1703.418853
std	2271627.117	61459.13838	3384.368136
min	549	0	0
25%	232608	5223.5	192
50%	629429	16667	585
75%	1565474	46971	1635
max	29075706	397715	38675

Bảng 4.1 Bảng phân vị của các cột thuộc tính

Thống kê mô tả cho thấy:

- Trung bình lượt views là 1392780.363, số lượt views thấp nhất là 549 và cao nhất là 29075706. Độ lệch chuẩn so với giá trị trung bình là 2271627.117. Đây là chuỗi có xu hướng phân phối lệch về bên trái vì median (mức phân vị 50%) nhỏ hơn mean.
- Trung bình lượt likes là 40968.83407, số lượt views thấp nhất là 0 và cao nhất là 397715. Độ lệch chuẩn so với giá trị trung bình là 61459.13838. Đây là chuỗi

có xu hướng phân phối lệch về bên trái vì median (mức phân vị 50%) nhỏ hơn mean.

- Trung bình lượt dislikes là 1703.418853, số lượt views thấp nhất là 0 và cao nhất là 38675. Độ lệch chuẩn so với giá trị trung bình là 3384.368136. Đây là chuỗi có xu hướng phân phối lệch về bên trái vì median(mức phân vị 50%) nhỏ hơn mean.

4.2. Phân tích đa biến

4.2.1. Ma trận tương quan

Ma trận tương quan để hiểu tương quan giữa các biến số trong phân tích đa biến. Nó được sử dụng để mô tả mức độ liên quan giữa hai hay nhiều biến số trong tập dữ liệu. Khi hai biến số có tương quan cao, điều đó có nghĩa là khi giá trị của một biến số thay đổi, giá trị của biến số kia sẽ có xu hướng thay đổi theo cùng một hướng.

Ma trận tương quan của tập dữ liệu là một bảng chứa các hệ số tương quan giữa tất cả các cặp biến số trong tập dữ liệu. Các hệ số tương quan này có giá trị từ 0 đến 1, trong đó giá trị 1 biểu thị mối tương quan dương hoàn hảo giữa hai biến và giá trị 0 biểu thị không có tương quan nào giữa hai biến.

Sau khi chuẩn bị dữ liệu, ta tính toán ma trận tương quan bằng cách sử dụng công thức tính toán tương quan giữa hai biến:

$$corr(X,Y) = \frac{cov(X,Y)}{\sigma_X \sigma_Y}$$

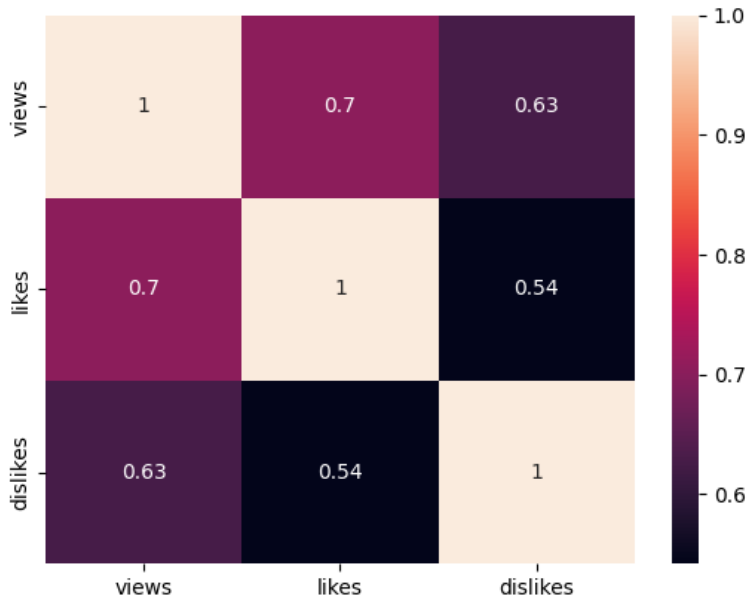
Trong đó $cov(X,Y)$ là hiệp phương sai giữa hai biến X và Y, và σ_X và σ_Y lần lượt là độ lệch chuẩn của X và Y. Ma trận tương quan sẽ có kích thước bằng số lượng biến, trong đó phần tử tại hàng i và cột j sẽ là tương quan giữa biến i và biến j.

```
correlationMatrix  = df[["views","likes","dislikes"]].corr()
print(correlationMatrix)
```

	views	**likes**	**dislikes**
views	1	0.70437	0.62741
likes	0.70437	1	0.54201
dislikes	0.62741	0.54201	1

Bảng 4.2 Bảng ma trận tương quan

Để trực quan hóa ma trận tương quan, ta có thể sử dụng biểu đồ heatmap (đồ thị nhiệt) để hiển thị mức độ tương quan giữa các biến dưới dạng màu sắc. Các giá trị tương quan lớn sẽ được đại diện bởi màu đậm, còn các giá trị tương quan nhỏ sẽ được đại diện bởi màu nhạt.



Hình 4.4 Đồ thị nhiệt giữa ba thuộc tính chính của tập dữ liệu

Dựa trên ma trận tương quan và biểu đồ heatmap, ta có thể phân tích mối tương quan giữa các biến để tìm hiểu mức độ ảnh hưởng của chúng lên nhau. Các biến có tương quan lớn hơn 0.7 thường được coi là có mối quan hệ mạnh, các biến có tương quan bằng 1 được coi là có mối quan hệ rất mạnh còn các biến có tương quan nhỏ hơn 0.3 thường được coi là không có mối quan hệ đáng kể với nhau:

- Views: số lượng video có lượt views lên top trending có độ tương quan rất cao
- Likes: số lượng video có lượt likes lên top trending có độ tương quan cao
- Dislikes: số lượng video có lượt dislikes lên top trending có độ tương quan trung bình

Hệ số tương quan > 0.5 cho thấy được mối tương quan giữa lượt views, lượt likes và dislikes là mạnh. Số lượng video có lượt views cao sẽ kéo theo lượt likes và dislikes tăng lên.

Phân tích ma trận tương quan giúp chúng ta hiểu được mức độ tương quan giữa các biến số trong tập dữ liệu và giúp chúng ta loại bỏ các biến có tương quan cao hoặc có thể kết hợp chúng lại để giảm số lượng biến và cải thiện hiệu suất của mô hình.

4.2.2. Phân tích chuỗi thời gian

Chúng ta sẽ phân tích khoảng thời gian những ngày hình thành xu hướng của các video trong bộ dữ liệu.

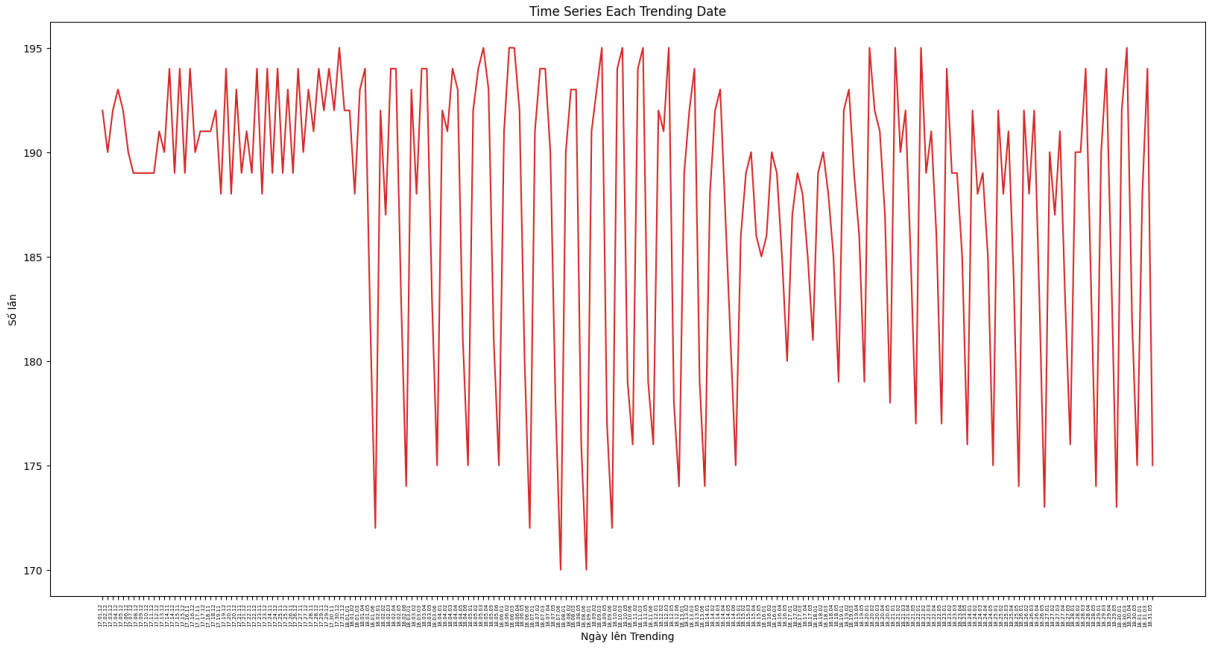
```
time_series = df.groupby(["trending_date"])["trending_date"].count()
time_series = pd.DataFrame({"Trending Date": time_series.index, "Count":
time_series.values})
time_series.head(10)
```

index	Trending Date	Count
0	17.01.12	192
1	17.02.12	190
2	17.03.12	192
3	17.04.12	193
4	17.05.12	192
5	17.06.12	190
6	17.07.12	189
7	17.08.12	189
8	17.09.12	189
9	17.10.12	189

Bảng 4.3 Bảng phân bố các ngày hình thành xu hướng của video

Cách phân bố ngày lên trending của các video lên xu hướng là bình thường. Ta thấy số lần video lên top trending trung bình 189 lần.

Sau đó, ta trực quan hóa dữ liệu bằng đồ thị đường để biểu diễn dữ liệu theo chuỗi thời gian:



Hình 4.5 Biểu đồ phân bố video vào các ngày xu hướng

CHƯƠNG 5: KHAI PHÁ DỮ LIỆU

5.1. Đặt vấn đề

Dựa vào tập dữ liệu hiện có, với các thuộc tính: ngày thịnh hành, tiêu đề, số lượt xem, số lượt thích, số lượt không thích, ngày phát hành và mô tả. Chúng ta có thể đặt ra những câu hỏi để tiến hành khai phá tập dữ liệu.

Đầu tiên, để hiểu rõ hơn về đa dạng và đặc điểm của các video thịnh hành, chúng ta sẽ phân nhóm chúng dựa trên các thuộc tính như lượt xem, lượt thích, lượt không thích,.... Mục tiêu là xác định các nhóm dữ liệu có đặc điểm chung về mức độ quan tâm từ người xem và phản hồi từ cộng đồng YouTube. Vậy chúng ta đặt ra vấn đề, ***"Có bao nhiêu nhóm các video dựa trên các thuộc tính như lượt xem, lượt thích, lượt không thích?".***

Tiếp theo, số lượt xem, lượt thích và lượt không thích là những chỉ số quan trọng trong việc đánh giá sự thành công của một video trên nền tảng YouTube. Tuy nhiên, ***"Video nào có khả năng thu hút nhiều lượt xem hơn, video có lượt thích cao hay video có lượt không thích thấp?"***. Câu hỏi này cũng rất quan trọng trong việc giúp các chủ sở hữu video có thể hiểu rõ hơn về yếu tố quan trọng để tạo ra một video thành công và thu hút được nhiều lượt xem trên YouTube. Chính vì vậy, việc tìm ra câu trả lời cho câu hỏi này sẽ mang lại nhiều giá trị và hỗ trợ cho các chủ sở hữu video đưa ra các chiến lược quảng cáo và tiếp cận được nhiều khán giả hơn trên nền tảng YouTube.

Vì vậy, chúng ta sẽ tiến hành phân tích tập dữ liệu và tìm câu trả lời cho hai câu hỏi:

- ***Có bao nhiêu nhóm các video dựa trên các thuộc tính như lượt xem, lượt thích, lượt không thích?***
- ***Video nào có khả năng thu hút nhiều lượt xem hơn, video có lượt thích cao hay video có lượt không thích thấp?***

5.2. Thuật toán sử dụng

5.2.1. Phân nhóm các video dựa trên các thuộc tính của chúng

Để giải quyết vấn đề này, phân nhóm các video dựa trên các thuộc tính như lượt xem, lượt thích, lượt không thích, một lựa chọn phù hợp là sử dụng thuật toán K-means Clustering.

K-means Clustering là một thuật toán phân cụm phổ biến và đơn giản, phân chia dữ liệu thành các nhóm dựa trên sự tương đồng về khoảng cách giữa các điểm dữ liệu. Trong trường hợp này, chúng ta muốn phân nhóm các video dựa trên các thuộc tính như lượt xem, lượt thích, lượt không thích. K-means Clustering là một lựa chọn hợp lý để xác định các nhóm video tương tự về các thuộc tính này. K-means Clustering là một

thuật toán đơn giản và hiệu quả trong việc phân cụm dữ liệu. Nó có khả năng xử lý dữ liệu lớn và hoạt động tốt với các thuộc tính số.

Ta có thể sử dụng thư viện scikit-learn trong Python để áp dụng thuật toán K-means Clustering. Đầu tiên, tiền xử lý dữ liệu bằng cách chọn các thuộc tính cần thiết và chuẩn hóa dữ liệu nếu cần. Sau đó, xác định số lượng nhóm mong muốn và áp dụng thuật toán K-means Clustering để phân nhóm các video. Bạn có thể sử dụng khoảng cách Euclidean hoặc khoảng cách cosine để đo độ tương đồng giữa các điểm dữ liệu.

Trong quá trình áp dụng, bạn cần chú ý đến số lượng nhóm (k) mà bạn muốn tạo ra và các phương pháp khởi tạo khác nhau (như ngẫu nhiên hoặc chọn ngẫu nhiên các điểm dữ liệu làm trung tâm ban đầu) có thể ảnh hưởng đến kết quả của thuật toán. Để đánh giá hiệu suất của phân cụm, bạn có thể sử dụng các độ đo như Silhouette Score hoặc Calinski-Harabasz Index.

5.2.2. Tìm hiểu về khả năng thu hút nhiều lượt xem của video

Để giải quyết vấn đề này, tìm hiểu video nào có khả năng thu hút nhiều lượt xem hơn, video có lượt thích cao hay video có lượt không thích thấp, một lựa chọn phù hợp là sử dụng thuật toán Random Forests.

Random Forests là một thuật toán phân loại rừng ngẫu nhiên dựa trên việc kết hợp nhiều cây quyết định. Với vấn đề này, chúng ta cần dự đoán xem một video có khả năng thu hút nhiều lượt xem hơn dựa trên các thuộc tính hiện có. Random Forests có khả năng xử lý dữ liệu lớn, xử lý cả dữ liệu số và dữ liệu rời rạc, và tỏ ra hiệu quả trong việc phân loại.

Random Forests được chứng minh là một thuật toán mạnh mẽ trong phân loại. Nó có khả năng xử lý dữ liệu lớn, giảm thiểu overfitting, và có thể đánh giá quan trọng đặc trưng của các thuộc tính dựa trên tần suất xuất hiện trong các cây quyết định.

Ta có thể sử dụng thư viện scikit-learn trong Python để áp dụng thuật toán Random Forests. Đầu tiên, tiền xử lý dữ liệu bằng cách chọn các thuộc tính cần thiết và chuẩn hóa dữ liệu nếu cần. Sau đó, chia dữ liệu thành tập huấn luyện và tập kiểm tra. Tiếp theo, huấn luyện mô hình Random Forests trên tập huấn luyện và đánh giá hiệu suất trên tập kiểm tra bằng các độ đo như độ chính xác, độ phủ, độ chính xác dương tính, và $F1 - score$.

Trong quá trình áp dụng, bạn cần chú ý đến các thông số của thuật toán Random Forests như số lượng cây quyết định ($n\_estimators$), độ sâu của cây (max_depth), và số lượng đặc trưng được xem xét khi tìm hiểu cây ($max\_features$). Tùy thuộc vào tập dữ liệu của bạn, bạn có thể thử nghiệm và điều chỉnh các thông số này để đạt được kết quả tốt nhất.

CHƯƠNG 6: ĐÁNH GIÁ VÀ CHỌN THUẬT TOÁN

6.1. Áp dụng thuật toán

6.1.1. Thuật toán K-means Clustering

Trước tiên, ta tạo tập dữ chứa các trường cần thiết: views, likes, dislikes

```
features = ["views", "likes", "dislikes"]
data = df[features].copy()
```

Bảng 6.1 Tạo tập data chứa dữ liệu cần thiết

```
[ ]  data
```

video_id	views	likes	dislikes
2kyS6SvSYSE	748374	57527	2966
1ZAPwfrtAFY	2418783	97185	6146
5qpjK5DgCt4	3191434	146033	5339
puqaWrEC7tY	343168	10172	666
d380meD0W0M	2095731	132235	1989
...	...	...	...
_QWZvU7VCn8	5564576	46351	2295
7UoP9ABJXGE	5534278	45128	1591
BZt0qjTWNhw	1685609	38160	1385
D6Oy4LfoqsU	1066451	48068	1032
oV0zkMe1K8s	5660813	192957	2846

38467 rows × 3 columns

Hình 6.1 Dữ liệu được trả về

Nhằm dễ dàng dữ lý dữ liệu, ta có thể đưa các giá trị dữ liệu về một khoảng giá trị cụ thể để dễ dàng so sánh và xử lý. Ta thực hiện bằng cách chuẩn hóa dữ liệu theo phạm vi từ 1 đến 10.

```
data = ((data - data.min()) / (data.max() - data.min())) * 9 + 1
```

Bảng 6.2 Chuẩn hóa giá trị trong miền từ 1 đến 10

```
[209] data.describe()
```

	views	likes	dislikes
count	38467.000000	38467.000000	38467.000000
mean	1.430955	1.927095	1.396400
std	0.703165	1.390775	0.787571
min	1.000000	1.000000	1.000000
25%	1.071832	1.118204	1.044680
50%	1.194665	1.377162	1.136134
75%	1.484411	2.062919	1.380478
max	10.000000	10.000000	10.000000

Hình 6.2 Giá trị trả về

Tiếp theo là các hàm cần thiết sử dụng trong thuật toán K-means Clustering:

- Tạo tâm ngẫu nhiên cho tập dữ liệu:

```
def random_centroids(data, k):
    centroids = []
    for i in range(k):
        centroid = data.apply(lambda x: float(x.sample()))
        centroids.append(centroid)
    return pd.concat(centroids, axis=1)
```

Bảng 6.3 Hàm random_centroids

- Tạo labels cho tập dữ liệu:

```
def get_labels(data, centroids):
    distances = centroids.apply(lambda x: np.sqrt(((data - x) **
2).sum(axis=1)))
    return distances.idxmin(axis=1)
```

Bảng 6.4 Hàm get_labels

- Tạo tâm mới:

```
def new_centroids(data, labels, k):
    centroids = data.groupby(labels).apply(lambda x:
np.exp(np.log(x).mean())).T
    return centroids
```

Bảng 6.5 Hàm new_centroids

- Vẽ đồ thị biểu diễn cho các cụm:

```
def plot_clusters(data, labels, centroids, iteration):
    pca = PCA(n_components=2)
    data_2d = pca.fit_transform(data)
    centroids_2d = pca.transform(centroids.T)
    clear_output(wait=True)
    plt.title(f'Iteration {iteration}')
    plt.scatter(x=data_2d[:,0], y=data_2d[:,1], c=labels, s=1)
    plt.scatter(x=centroids_2d[:,0], y=centroids_2d[:,1], s=10,
marker='x', color = 'r')
    plt.show()
```

Bảng 6.6 Hàm plot_clusters

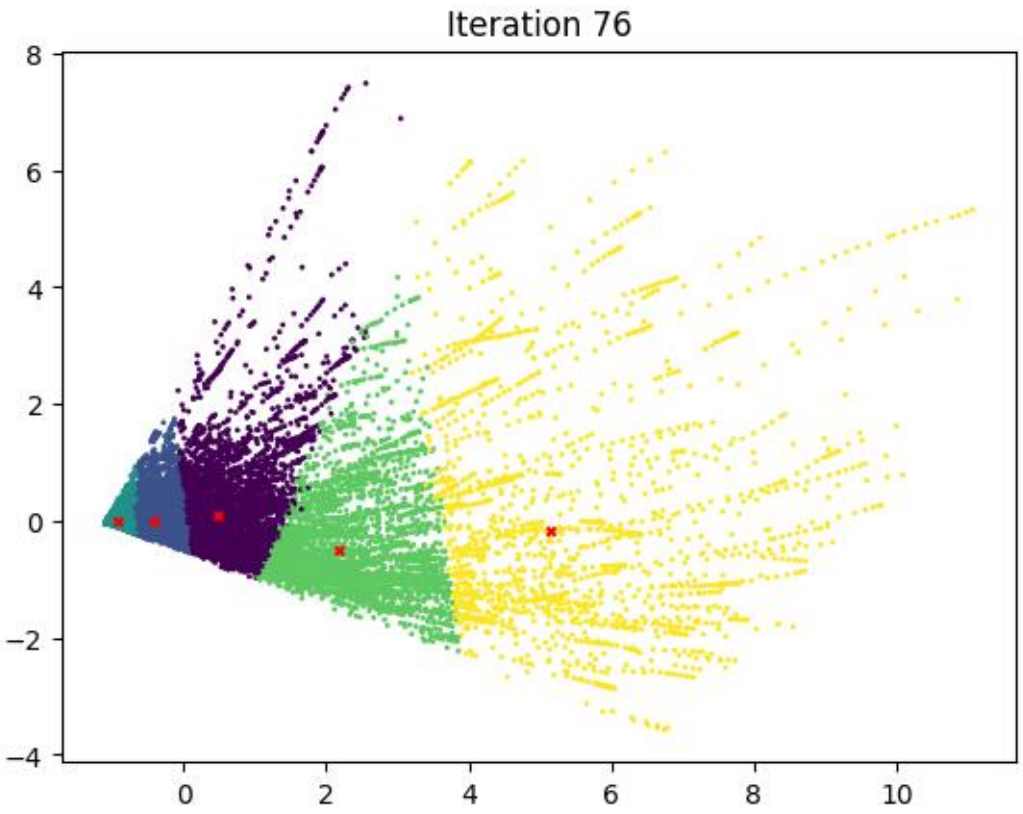
Phần chính của thuật toán K-means Clustering:

```
max_iterations = 100
centroid_count = 5

centroids = random_centroids(data, centroid_count)
old_centroids = pd.DataFrame()
iteration = 1

while iteration < max_iterations and not
centroids.equals(old_centroids):
    old_centroids = centroids

    labels = get_labels(data, centroids)
    centroids = new_centroids(data, labels, centroid_count)
    plot_clusters(data, labels, centroids, iteration)
    iteration += 1
```

Bảng 6.7 Kết hợp các hàm để phân nhóm tập dữ liệu

Để tiến hành phân nhóm, ta thực hiện các bước sau:

- Khởi tạo các centroid ban đầu: Trước khi bắt đầu thuật toán, cần khởi tạo ngẫu nhiên các centroid ban đầu.
- Gán nhãn cho các mẫu dữ liệu: Với các centroid đã được khởi tạo, cần gán nhãn cho từng mẫu dữ liệu dựa trên khoảng cách từ mẫu đó đến các centroid.
- Cập nhật centroid mới: Dựa trên nhãn đã được gán, cần tính toán lại vị trí của các centroid mới dựa trên trung bình của các mẫu dữ liệu trong cùng một nhóm.
- Lặp lại quá trình cho đến khi tiêu chí dừng được đáp ứng: Quá trình gán nhãn và cập nhật centroid mới sẽ được lặp lại cho đến khi các centroid không thay đổi hoặc đạt tới số lần lặp tối đa.
- Vẽ đồ thị các nhóm dữ liệu bằng hàm plot_clusters.



Hình 6.3 Đồ thị phân nhóm bằng K-means Clustering

```
[221] centroids
```

	0	1	2	3	4
views	1.674762	1.302966	1.086149	1.950629	3.294388
likes	2.298058	1.577104	1.129116	4.075616	6.457339
dislikes	1.629502	1.245760	1.068107	1.782034	3.015319

Hình 6.4 Dữ liệu các centroid được tạo

Sau khi hoàn tất việc phân nhóm, các video sẽ được gắn nhãn đúng với nhóm mà chúng thuộc về. Qua đó, chúng ta nhận biết sự tương đồng hoặc khác biệt giữa các video trong tập dữ liệu. Các nhóm video có thể được hiểu là các tập hợp video có đặc điểm tương tự nhau.

```
[222] df[labels == 3][["title"] + features]
```

video_id	title	views	likes	dislikes
1ZAPwfrtAFY	The Trump Presidency: Last Week Tonight with J...	2418783	97185	6146
5qpjK5DgCt4	Racist Superman | Rudy Mancuso, King Bach & Le...	3191434	146033	5339
d380meD0W0M	I Dare You: GOING BALDI?	2095731	132235	1989
5E4ZBSInqUU	Marshmello - Blocks (Official Music Video)	687582	114188	1333
ujyTQNNjjDU	G-Eazy - The Plan (Official Video)	2642930	115795	3055
...	...	...	...	...
99t4EBwlAt8	Shawn Mendes Answers the Web's Most Searched Q...	2310794	105783	558
SQsPvrev_bQ	435	2252933	129865	1550
oLDbO545aKQ	Terrible Magicians | Rudy Mancuso & Juanpa Zurita	3825440	196635	4514
Xr2rgT9uEnA	LIE DETECTOR TEST WITH MY GIRLFRIEND!	3229540	109945	3062
oV0zkMe1K8s	How Black Panther Should Have Ended	5660813	192957	2846

3338 rows × 4 columns

Hình 6.5 Ví dụ minh họa việc phân nhóm thành công

6.1.2. Thuật toán Random Forests

6.2. Tiến hành đánh giá kết quả

6.2.1. Thuật toán K-means Clustering

CHƯƠNG 7: KẾT QUẢ VÀ THẢO LUẬN

7.1. Đánh giá kết quả

7.2. Điểm mạnh của nghiên cứu

7.3. Điểm yếu cả nghiên cứu

CHƯƠNG 8: KẾT LUẬN

8.1. Tổng kết kết quả

8.2. Kết luận hiệu quả

TÀI LIỆU THAM KHẢO