BỘ GIÁO DỤC VÀ ĐÀO TẠO
ĐẠI HỌC KINH TẾ TP HỒ CHÍ MINH
TRƯỜNG CÔNG NGHỆ VÀ THIẾT KẾ

ĐỒ ÁN MÔN HỌC

ĐỀ TÀI:

THU THẬP DỮ LIỆU (CRAWL) TỪ PLAYLIST TRÊN SPOTIFY & XÂY DỰNG RECOMMENDATION SYSTEM

Học phần: DỮ LIỆU LỚN VÀ ỨNG DỤNG

Nhóm Sinh Viên:

1. NGÔ THỊ HUYỀN
2. NGUYỄN TRỊNH THU HUYỀN
3. MAI TRẦN MỸ UYÊN

Chuyên Ngành: KHOA HỌC DỮ LIỆU
Khóa: K46
Giảng Viên: TS. Đặng Nhân Cách

TP. Hồ Chí Minh, Ngày 04 tháng 04 năm 2023

Lời cảm ơn

Để có thể hoàn thành được bài báo cáo cuối kỳ không chỉ có riêng sự cố gắng của các thành viên trong nhóm mà còn nhờ vào sự hỗ trợ của rất nhiều của thầy Đặng Nhân Cách. Chúng em xin được gửi lời cảm ơn chân thành đến thầy đã tận tình hướng dẫn chúng em về cách thức tiến hành đề tài nghiên cứu và qua các bài giảng trên lớp để có thể hoàn thành tốt bài báo cáo này

Trong quá trình phát triển đề tài do còn hạn chế về kiến thức cũng như kinh nghiệm vì vậy bài làm sẽ không tránh khỏi những sai sót. Em mong nhận được sự thông cảm và góp ý phê bình từ phía Thầy.

Trân Trọng

MỤC LỤC

MỤC LỤC ẢNH

MỤC LỤC BẢNG

CHƯƠNG 1. TỔNG QUAN

1.1. Tổng Quan Về Bài Toán

Hệ thống gợi ý (Recommender systems – RS) đang từng bước trở thành một lĩnh vực nghiên cứu quan trọng và được ứng dụng khá thành công trong thực tiễn, giúp người dùng đối phó với vấn đề quá tải thông tin [1]. Hiện nay, RS được nghiên cứu và ứng dụng trong nhiều lĩnh vực khác nhau như: thương mại điện tử (bán hàng trực tuyến), giải trí (phim ảnh, âm nhạc,...), giáo dục đào tạo (gợi ý nguồn tài nguyên học tập như sách, báo,..),… Trên thế giới, đã có nhiều công ty, tổ chức đã áp dụng thành công hệ thống gợi ý nhằm gợi ý các dịch vụ, sản phẩm và các thông tin cần thiết đến người dùng như: website mua sắm trực tuyến Amazon.com gợi ý cho mỗi khách hàng những sản phẩm mà họ có thể quan tâm, YouTube.com giới thiệu các video clip cho người xem, gợi ý phim ảnh của Netflix.com,... Điều này góp phần làm tăng doanh số bán hàng hoặc số lượng truy cập, download của hệ thống, đồng thời giúp cho khách hàng có thể tìm kiếm được những thông tin thú vị hoặc những sản phẩm mà họ mong muốn dễ dàng hơn. Cùng với sự phát triển mạnh mẽ của các loại hình truyền thông đa phương tiện thì âm nhạc là một trong những nội dung khá phổ biến và được xem như là một nhu cầu không thể thiếu trong cuộc sống, có thể chia sẻ bởi nhiều người từ nhiều quốc gia có ngôn ngữ và nền văn hóa khác nhau. Tuy nhiên, số lượng bài nhạc đang ngày càng tăng lên, đa dạng và phong phú cả về nội dung lẫn thể loại. Vì vậy, vấn đề đặt ra là khi một người sử dụng muốn tìm nghe những bài nhạc mà mình yêu thích, người sử dụng sẽ mất rất nhiều thời gian để tìm kiếm (điều này tốn thời gian mà lại không hiệu quả). Do đó, nhu cầu cần có một hệ thống gợi ý có khả năng dự đoán mức độ ưa thích của người sử dụng với từng bản nhạc và gợi ý cho họ các bản nhạc mới mà hệ thống cho là phù hợp.

1.2. Lý Do Chọn Lựa Đề Tài

Mục tiêu chính là phát triển một hệ thống đề xuất cho danh sách phát tự động. Hệ thống đề xuất có thể tạo danh sách các bản nhạc từ bộ dữ liệu được cào từ trước để dự đoán và đề xuất chính xác các bản nhạc cho danh sách phát tương ứng.

Vấn đề cần giải quyết là xác định các cách khác nhau để đề xuất bài hát tiếp theo cho người dùng hoặc danh sách phát.Các bài hát được đề xuất càng chính xác thì người dùng càng có nhiều khả năng tiếp tục sử dụng dịch vụ âm nhạc đó. Vì tập dữ liệu rất lớn và có nhiều thuộc tính nên không thể có một phương pháp duy nhất nào hoạt động hiệu quả và cho kết quả khả quan. Do đó, chúng tôi đã thử nghiệm nhiều phương pháp để đạt được mục tiêu.

Hai phương pháp chính là hệ thống lấy danh sách xếp hạng người dùng các bài hát khác nhau và sau đó đề xuất các bài hát tương ứng. Nhưng vấn đề là không phải lúc nào cũng yêu cầu người dùng xếp hạng các bài hát. Do đó, phương pháp thứ hai sử dụng các bài hát được đề xuất dựa trên các thuộc tính khác nhau mà người dùng yêu thích. Kết quả tạo ra của mô hình khá tốt, mặc dù các mô hình có thể được đào tạo tốt hơn về mặt thời gian và kiến trúc hệ thống tốt hơn. Kết quả tạo ra phù hợp với sự lựa chọn của người dùng cũng như trên các quan điểm thống kê. Thông tin thêm về dự án và các phương pháp được sử dụng được trình bày chi tiết trong báo cáo.

CHƯƠNG 2. CƠ SỞ LÝ THUYẾT

2.1. Dữ liệu lớn

Năm 2014 Gartner chính thức giới thiệu mô hình Big Data "5Vs": Mô hình này được bắt đầu bằng 5 chữ V mô tả năm tính chất quan trọng của Big Data bao gồm: Khối lượng dữ liệu (Volume); Tốc độ (Velocity); Giá trị (Value); Độ tin cậy/chính xác (Veracity); Đa dạng (Variety). [1]

- Khối lượng dữ liệu (Volume)

Đối với hệ thống âm nhạc Spotify có gần 96 triệu người dùng (tạo ra một lượng dữ liệu khổng lồ mỗi ngày. Thông qua thông tin này, nền tảng dựa trên đám mây sẽ tự động tạo các bài hát được đề xuất, thông qua một động cơ khuyến nghị thông minh dựa trên lượt thích, chia sẻ, lịch sử tìm kiếm

Spotify hiện có hơn 456 triệu người dùng hoạt động hàng tháng. Với hơn 195 triệu người đăng ký trả tiền. Tính đến năm 2023, có 82 triệu bài hát và 4.7 triệu podcasts miễn phí trên nền tảng Spotify [2]tạo ra một khối lượng dữ liệu khổng lồ mỗi ngày

- Tốc độ (Velocity) Tốc độ có thể hiểu theo 2 khía cạnh:

Mỗi ngày, người dùng Spotify tạo ra hơn 600GB dữ liệu nghe, trong khi Spotify tạo ra hơn 4TB dữ liệu lưu trữ nhạc mới và các tài sản khác. Tổng cộng, Spotify có hơn 28 petabyte dung lượng lưu trữ trải rộng trên bốn trung tâm dữ liệu toàn cầu.

- (3) Đa dạng (Variety)

Spotify sử dụng nhiều loại dữ liệu có cấu trúc từ người dùng và nhóm người dùng, chẳng hạn như: số lượng bài hát được phát bởi người dùng trong một ngày, số lượng người dùng nghe một album cụ thể tại một thời điểm, v.v. Sau đó, họ có thể sử dụng dữ liệu này để tạo ra những dự đoán về người dùng muốn nghe vào thời điểm nào trong ngày, những nghệ sĩ mới mà người dùng có thể thích, v.v. Tóm lại, Spotify xử lý nhiều loại dữ liệu hữu ích cho phân tích hành vi người dùng [3]

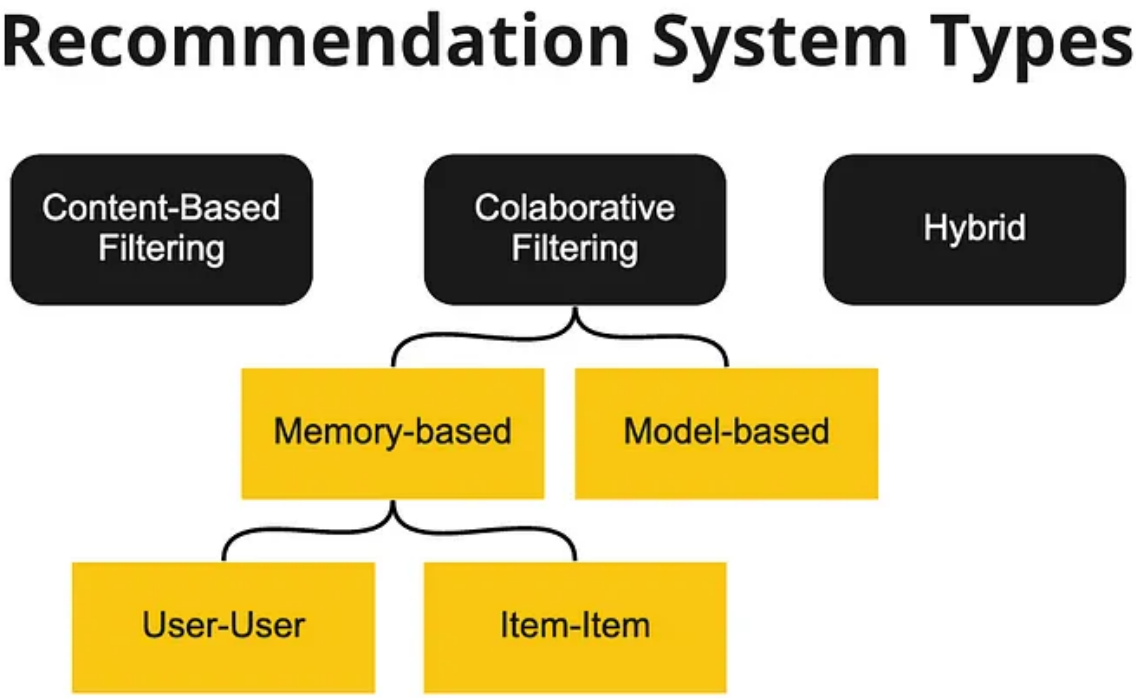
- Độ tin cậy/chính xác (Veracity)

Spotify dựa vào dữ liệu để cung cấp thông tin chi tiết và đưa ra quyết định hoàn toàn dựa trên dữ liệu. Tính xác thực (độ tin cậy) của dữ liệu là cần thiết trong bối cảnh này. Vì Spotify thu thập hầu hết dữ liệu từ hoạt động của người dùng và thông tin qua internet, dữ liệu họ thu thập thường có thể được coi là đáng tin cậy.

- Giá trị (Value)

Một trong những tính năng của Spotify là 'Wrapped'. Mỗi tháng mười hai, 'Wrapped' cung cấp cho người dùng một loạt các bài hát / nghệ sĩ yêu thích của người nghe hoặc được nghe nhiều nhất trong cả năm. 'Wrapped' cho người dùng biết liệu họ có nằm trong 1% người nghe trung thành nhất của nghệ sĩ hay không. Thông tin này được trình bày bằng cách sử dụng trực quan hóa dữ liệu để tạo một câu chuyện được cá nhân hóa cho tất cả người dùng. Spotify khuyến khích sự tương tác của người dùng với tính năng "Wrapped" bằng cách cấp huy hiệu cho người dùng. Ví dụ: nếu danh sách phát của người dùng có được số lượng người theo dõi mới, họ có thể được trao huy hiệu Tastemaker. Nếu người dùng nghe một bài hát hit trước bất kỳ ai khác, họ sẽ được cấp huy hiệu "Pioneer".

2.1. Hệ thống gợi ý (Recommender systems – RS)



Hình 1. Các dạng hệ thống đề xuất

Các hệ thống gợi ý thường sử dụng là kỹ thuật lọc cộng tác (collaborative filtering) để đưa ra các dự đoán về sở thích của người dùng (user) đối với các mục tin (items – như sản phẩm, sách, báo, phim,...) mà hệ thống cho là phù hợp nhất thông qua việc sử dụng những xếp hạng (rating/feedback) trong quá khứ của người dùng và/hoặc những xếp hạng của những người dùng khác đã có trong cơ sở dữ liệu. Lọc cộng tác thường được tiếp cận

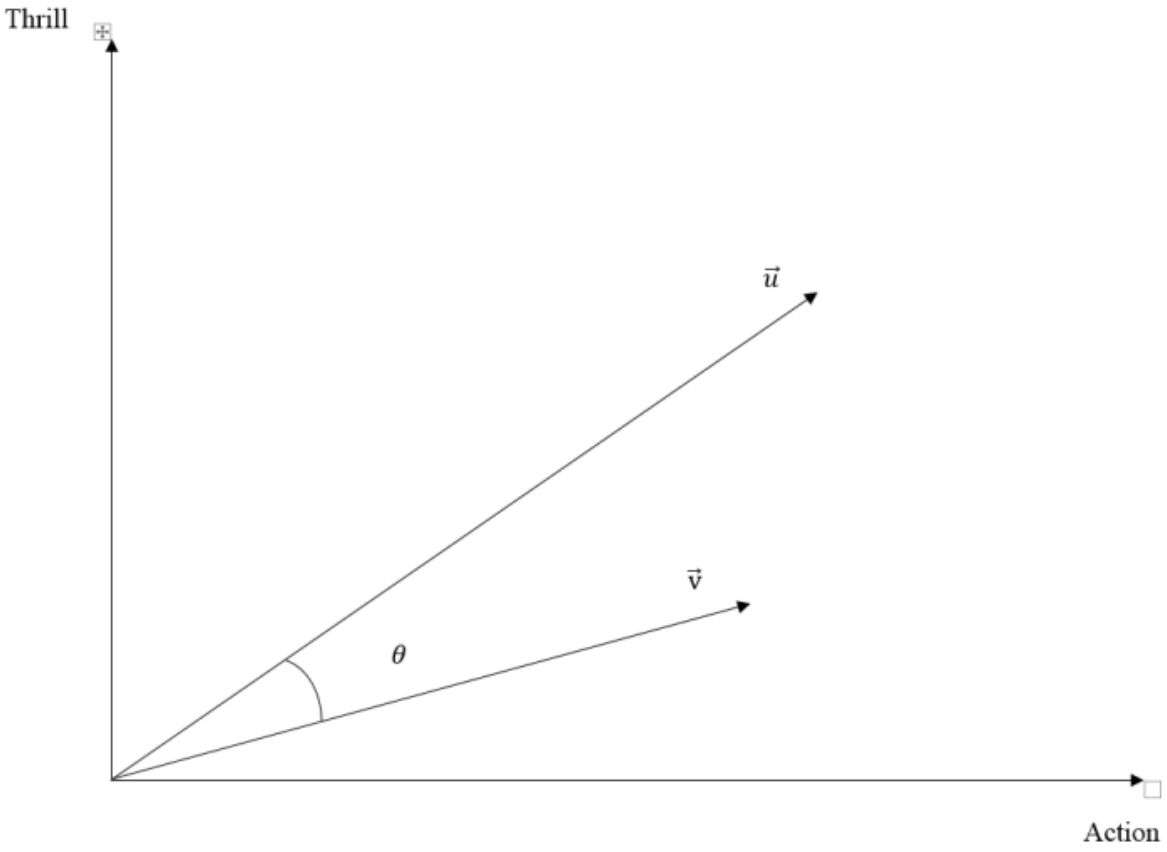
theo 2 dạng: lọc cộng tác dựa vào bộ nhớ, cụ thể là tiếp cận dựa trên người dùng (user-based) và dựa trên các mục tin (item-based). Dạng thứ 2 là tiếp cận theo mô hình (như mô hình Bayes,...) và gần đây là các mô hình nhân tố tiềm ẩn (latent factor models) như mô hình phân rã ma trận [4] (matrix factorization - đã đạt được những thành công đáng kể).

Hiện tại, trong RS có rất nhiều giải thuật được đề xuất, tuy nhiên có thể gom chung vào trong các nhóm chính: nhóm giải thuật lọc theo nội dung (content-based filtering),nhóm giải thuật lọc cộng tác (collaborative filtering), nhóm giải thuật lai ghép (hybrid filtering). Lọc cộng tác thường được tiếp cận theo 2 dạng: lọc cộng tác dựa vào bộ nhớ, cụ thể là tiếp cận dựa trên người dùng (user-based) và dựa trên các mục tin (item-based). Dạng thứ 2 là tiếp cận theo mô hình (như mô hình Bayes,...) và gần đây là các mô hình nhân tố tiềm ẩn (latent factor models) như mô hình phân rã ma trận [5]

2.1.1. Lọc dựa trên nội dung: Content-based filtering

Đây là một phương pháp đề xuất các các bài hát tương tự với các bài hát khác trong tập dữ liệu

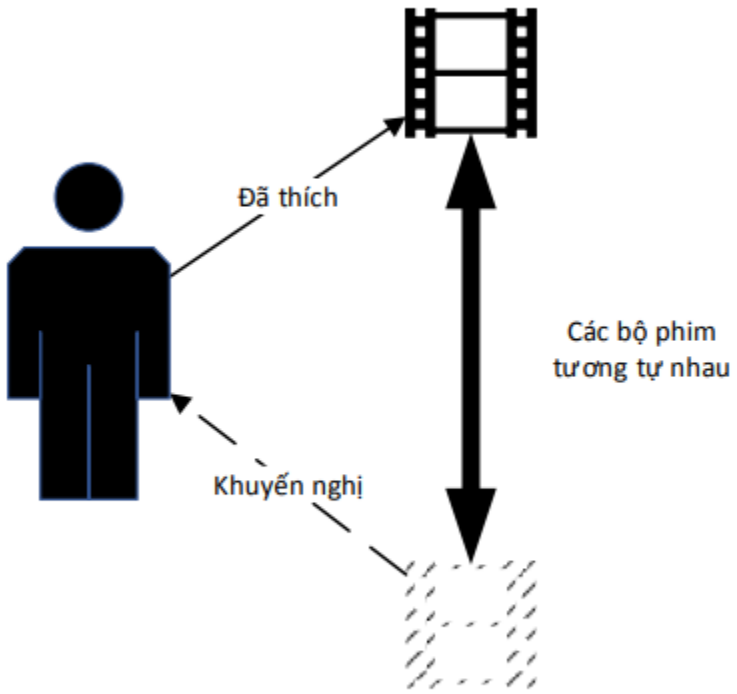
Hệ thống khuyến nghị ghi nhận user profile dưới dạng vector $v = (v_1, v_2, \dots, v_n)$, trong đó v_i là trọng số thể hiện mức độ quan tâm của người dùng đối với từng thuộc tính của sản phẩm. Vector item profile $u = (u_1, u_2, \dots, u_n)$ biểu diễn thông tin sản phẩm thông qua các thuộc tính u_i. Như vậy thông tin về sở thích của người dùng và thông tin của sảnphẩm được ánh xạ vào cùng không gian vector các thuộc tính của sản phẩm, sự phù hợp giữa sở thích của người dùng và sản phẩm được đo bằng góc lệch giữa 2 vector profile như minh họa trong Hình 2. [6]



Hình 2. Cách thức hoạt động của lọc cộng tác

Để đánh giá độ tương đồng giữa 2 vector u và v để đưa ra khuyến nghị, hệ thống so sánh bằng cosine góc lệch giữa 2 vector

$$similarity = cos(\theta) = \frac{\vec{u}\vec{v}}{\|\vec{u}\| \cdot \|\vec{v}\|} = \frac{\sum_{i=1}^{n} u_i . v_i}{\sqrt{\sum_{i=1}^{n} u_i^2} \cdot \sqrt{\sum_{i=1}^{n} v_i^2}} \quad (2)$$



Hình 1.3: Cơ chế hoạt động lọc theo nội dung

Hình 3. Cơ chế hoạt động lọc theo nội dung

- Những ưu điểm của phương pháp khuyến nghị dựa trên nội dung:

- Một là, hệ thống không yêu cầu cần có quá nhiều dữ liệu từ những người dùng khác để đạt được độ chính xác khuyến nghị chấp nhận được. Đối với phương pháp lọc dựa trên cộng tác hệ thống RS cần phải phân tích toàn bộ dữ liệu tương tác để tìm ra các quy luật (là những sản phẩm được người dùng ưa thích) thì mới có thể đưa ra gợi ý, tuy nhiên với phương pháp lọc dựa trên nội dung thì hệ thống chỉ dựa trên thông tin nội dung sản phẩm và dữ liệu tương tác của từng người dùng cụ thể để đưa ra gợi ý.
- Hai là, có thể đưa ra khuyến nghị về các sản phẩm mới dựa trên thông tin mô tả có sẵn mà không cần phải có dữ liệu tương tác từ người dùng. Điều này xuất phát từ cơ chế hoạt động của lọc nội dung dựa trên phân tích về nội dung cấu tạo nên sản phẩm để đưa ra gợi ý, do đó khi một sản phẩm mới xuất hiện trong hệ thống thì

chỉ cần nội dung của sản phẩm này "tương tự" với những sản phẩm trước đó đã được người dùng ưa thích sẽ được hệ thống khuyến nghị cho người dùng.

- Ba là, trong trường hợp người dùng có những sở thích đặc thù dựa trên nội dung của sản phẩm nhưng những sản phẩm này lại không phổ biến trong dữ liệu ma trận tương tác thì hệ thống vẫn có khả năng khám phá được nhờ thông tin nội dung.
- Bốn là, mô hình có khả năng giải thích tốt (self-explainable) về những nhân tố có ảnh hưởng đến sở thích của người dùng thông qua các giá trị trọng số của vector profile của người dùng. Giá trị trọng số càng lớn chứng tỏ mức độ quan tâm càng cao của người dùng đối với yếu tố nội dung của sản phẩm, nhờ vậy nhà kinh doanh có thể dựa vào thông tin này để tăng cường những đặc tính/nội dung phù hợp với người dùng. Bên cạnh những ưu điểm đã nêu, phương pháp khuyến nghị dựa trên nội dung gặp một số nhược điểm như sau: Hệ thống yêu cầu chất lượng thông tin nội dung của sản phẩm cao: Đối với các sản phẩm multimedia (phim ảnh, âm nhạc,…) hiện nay chưa có các phương pháp rút trích thông tin có hiệu quả với những loại dữ liệu này, do đó việc xây dựng vector thuộc tính cho những sản phẩm dạng này gặp rất nhiều khó khăn, chất lượng thông tin không tốt. Khi chất lượng thông tin mô tả đối tượng có chất lượng kém hoặc bị lỗi thì phương pháp khuyến nghị dựa trên nội dung hoạt động không hiệu quả. Kết quả khuyến nghị do hệ thống đưa ra gặp hiện tượng "cứng nhắc" (Overspecialization): Hệ thống chỉ gợi ý được các sản phẩm có các đặc tính đúng với profile của từng người dùng mà không tận dụng được thông tin từ những người dùng khác có sở thích tương tự.

2.1.2. Lọc cộng tác: Collaborative filtering

Lọc cộng tác là kỹ thuật dự đoán đối với nội dung không thể được mô tả dễ dàng và đầy đủ bởi siêu dữ liệu (metadata) như những bộ phim và nhạc. Kỹ thuật lọc cộng tác hoạt động bằng cách xây dựng một cơ sở dữ liệu (ma trận người dùng-mục tin) sở thích về các mục tin theo những người dùng. Sau đó kết hợp những người dùng với các sở thích và mối quan tâm thích hợp bằng cách tính toán các độ tương tự giữa những hồ sơ người dùng để tạo các gợi ý. Các kỹ thuật lọc cộng tác có thể được chia thành hai loại: dựa bộ nhớ (memory-based) và dựa mô hình (model-based) [7]:

- Lọc cộng tác dựa trên bộ nhớ (Memory-based Collaborative Filtering)

Dựa trên giá trị xếp hạng của người dùng trong ma trận Người dùng – Sản phẩm, hệ thống tính toán độ tương đồng giữa người dùng hiện tại với những người dùng tương tự theo thủ tục gồm 02 bước như sau:

Bước 1: Hệ thống tính toán độ tương tự giữa những người dùng.

$$similarity(x,y) = cos(\vec{x},\vec{y}) = \frac{\vec{x}\cdot\vec{y}}{\|\vec{x}\|_2 \times \|\vec{y}\|_2} = \frac{\Sigma r_{x,i}\cdot r_{y,i}}{\sqrt{\Sigma r_{x,i}^2}\sqrt{\Sigma r_{y,i}^2}} \quad (3)$$

Bước 2: Tính toán giá trị xếp hạng dự đoán theo công thức:

$$r(a,i) = \overline{r_a} + \frac{\Sigma_{i=1}^{n}\left(r_{u,i} - \overline{r_u}\right) \times similarity(x,y)}{\Sigma_{i=1}^{n} similarity(x,y)} \quad (4)$$

- Những ưu điểm của phương pháp lọc cộng tác dựa trên bộ nhớ:

- Một là, hệ thống RS không yêu cầu cần có tri thức chuyên biệt miền để xây dựng mô hình người dùng/sản phẩm. Bản chất của phương pháp lọc dựa trên cộng tác là khai thác thông tin về sở thích của nhóm người dùng có sở thích giống nhau về những sản phẩm nhất định, do đó khi phát hiện được những sản phẩm được ưa thích thì hệ thống RS hoàn toàn có thể đưa ra gợi ý trực tiếp đến sản phẩm đó mà không cần quan tâm đến bản chất (nội dung) của sản phẩm đó. Điều này giúp cho hệ thống có khả năng mở rộng khuyến nghị cho nhiều loại đối tượng có bản chất khác nhau.
- Hai là, có khả năng tận dụng thông tin của những người dùng tương tự để đưa ra khuyến nghị (khai thác xu hướng, sở thích của những nhóm người dùng giống nhau).

+ User-based collaborative filtering : Trong mô hình này, các sản phẩm được khuyến nghị cho người dùng dựa trên thực tế là các sản phẩm đã được người dùng yêu thích tương tự như người dùng. Ví dụ: nếu Derrick và Dennis thích những bộ phim giống nhau và một bộ phim mới ra mắt mà Derick thích, thì chúng tôi có thể giới thiệu bộ phim đó cho Dennis vì Derrick và Dennis có vẻ thích những bộ phim tương tự.

+ Item-based collaborative filtering : Các hệ thống này xác định các mục tương tự dựa trên xếp hạng trước đây của người dùng. Ví dụ: nếu người dùng A, B và C xếp hạng 5 sao cho sách X và Y thì khi người dùng D mua sách Y, họ cũng nhận

được đề xuất mua sách X vì hệ thống xác định sách X và Y giống nhau dựa trên xếp hạng của người dùng A, B và C

+ Kỹ thuật dựa bộ nhớ (Memorybased) (còn gọi là Phương pháp láng giềng - Neighborhood-based): có thể đạt được theo hai cách gồm các kỹ thuật dựa vào người dùng (user-based) và mục tin (item-based), trong đó hoặc là dựa trên dữ liệu quá khứ của người dùng "tương tự - similarity" (user-based approach), hoặc là dựa trên dữ liệu quá khứ của những mục tin "tương tự" (itembased approach).

- Kỹ thuật dựa trên mô hình (Model-based): quy trình xây dựng mô hình có thể được thực hiện bằng cách dùng các kỹ thuật khai phá dữ liệu và học máy. Các kỹ thuật này liên quan đến việc xây dựng các mô hình dự đoán dựa trên dữ liệu thu thập được trong quá khứ. Ví dụ những kỹ thuật này gồm luật kết hợp, phân cụm, mạng Bayesian, mạng nơron.

Các phương pháp lọc cộng tác thường gặp phải ba vấn đề: Cold Start, khả năng mở rộng và sự thưa thớt (sparsity).

- Cold Start: Các hệ thống này thường yêu cầu một lượng lớn dữ liệu hiện có của người dùng để đưa ra các đề xuất chính xác.
- Khả năng mở rộng: Trong nhiều môi trường mà các hệ thống này đưa ra các khuyến nghị, có hàng triệu người dùng và sản phẩm. Do đó, một lượng lớn công suất tính toán thường là cần thiết để tính toán các gợi ý.
- Sparsity: Số lượng các mặt hàng được bán trên các trang web thương mại điện tử lớn là cực kỳ lớn. Những người dùng tích cực nhất sẽ chỉ đánh giá một tập con nhỏ của cơ sở dữ liệu tổng thể. Do đó, ngay cả những mặt hàng phổ biến nhất cũng có rất ít xếp hạng.

2.2. Data preprocessing

Hai phương pháp trích xuất đặc trưng được xem xét sử dụng như túi từ (Bag of Words) và TF-IDF. Hai phương pháp này đơn giản nhưng hiệu quả đối với việc biểu diễn dữ liệu văn bản. TF (Term Frequency) là tần suất xuất hiện của từ trong một đoạn văn bản. TF của một từ được tính bằng cách lấy số lần xuất hiện của từ đó chia cho tổng số từ có trong trong đoạn văn

Phương pháp TF-IDF được giới thiệu và áp dụng cho dữ liệu thể loại. TF-IDF, là một công cụ để định lượng các từ trong một bộ tài liệu. Mục tiêu của TF-IDF là thể hiện tầm quan trọng của một từ trong tài liệu và kho ngữ liệu. Công thức chung để tính TF-IDF là:

$$w_{i,j} = \text{tf}_{i,j} \text{ x } \log(\frac{N}{df_i})$$

$Tf_{i,j:}$ số lần i có mặt trong j

df_i : số văn bản chứa i

N: tổng số từ của văn bản

TF đếm số lần một thuật ngữ xuất hiện trong một tài liệu chia cho tổng số từ trong tài liệu đó, trong khi IDF lấy giá trị log của tần số tài liệu, đó là tổng số tài liệu mà một thuật ngữ có mặt.

Mục đích của TF-IDF là xác định các từ quan trọng trong mỗi tài liệu và toàn bộ tập hợp tài liệu. Trong dự án này, các tài liệu tương đương với các bài hát. Do đó, chúng tôi tính toán thể loại quan trọng nhất trong mỗi bài hát và sự phổ biến của chúng trong toàn bộ các bài hát để xác định trọng số của thể loại. Phương pháp này tốt hơn mã hóa one-hot vì nó tính đến tầm quan trọng và sự phổ biến của mỗi thể loại, thay vì đại diện cho mỗi thể loại là có hoặc không có. Động lực là tìm những từ không chỉ quan trọng trong mỗi tài liệu mà còn chiếm toàn bộ ngữ liệu. Giá trị nhật ký được sử dụng để giảm tác động của N lớn, điều này sẽ dẫn đến IDF rất lớn so với TF. TF tập trung vào tầm quan trọng của một từ trong tài liệu, trong khi IDF tập trung vào tầm quan trọng của một từ trong tài liệu

2.3. Phân cụm K-Means

Phân cụm thuộc nhóm phương pháp học không giám sát (unsupervised learning) vì không biết trước được số nhóm (khác với bài toán phân lớp). Quá trình này sẽ gom cụm các đối tượng có đặc điểm tương đồng vào các cụm tương ứng. Thuật toán sẽ tìm cách phân các đối tượng thảnh từng cụm có đặc điểm tương tự, đồng thời độ tương đồng giữa các cụm thấp (khác biệt cao). Hai phương pháp phân cụm sẽ được áp dụng đối với bộ dữ liệu trên bao gồm: phân cụm phân cấp và phân cụm phân hoạch. [8]

Phân cụm dữ liệu nhằm mục đích chính là thành lập các nhóm dữ liệu từ tập dữ liệu lớn, theo đó, cho phép ta khai phá và tìm kiếm thông tin tiềm ẩn, hữu ích phục vụ cho ra quyết định. Đây là một phương pháp xử lý thông tin quan trọng, phổ biến, nó nhằm khám phá mối liên hệ giữa các mẫu dữ liệu bằng cách tổ chức chúng thành các cụm tương tự.

Các bước trong thuật toán K-Means

- Đầu vào: Cho tập dữ liệu D, với K là số cụm, phép đo khoảng cách giữa 2 điểm dữ liệu là d(x,y)

- Khởi tạo: Khởi tạo K điểm dữ liệu trong D làm các điểm trung tâm (centroid)
- Lặp lại các bước sau đến khi hội tụ:
    - *Bước 1*: Với mỗi điểm dữ liệu, gán điểm dữ liệu đó vào cluster có khoảng cách đến điểm trung tâm của cluster là nhỏ nhất.
    - *Bước 2:* Với mỗi cluster, xác định lại điểm trung tâm của tất cả các điểm dữ liệu được gán vào cluster đó.

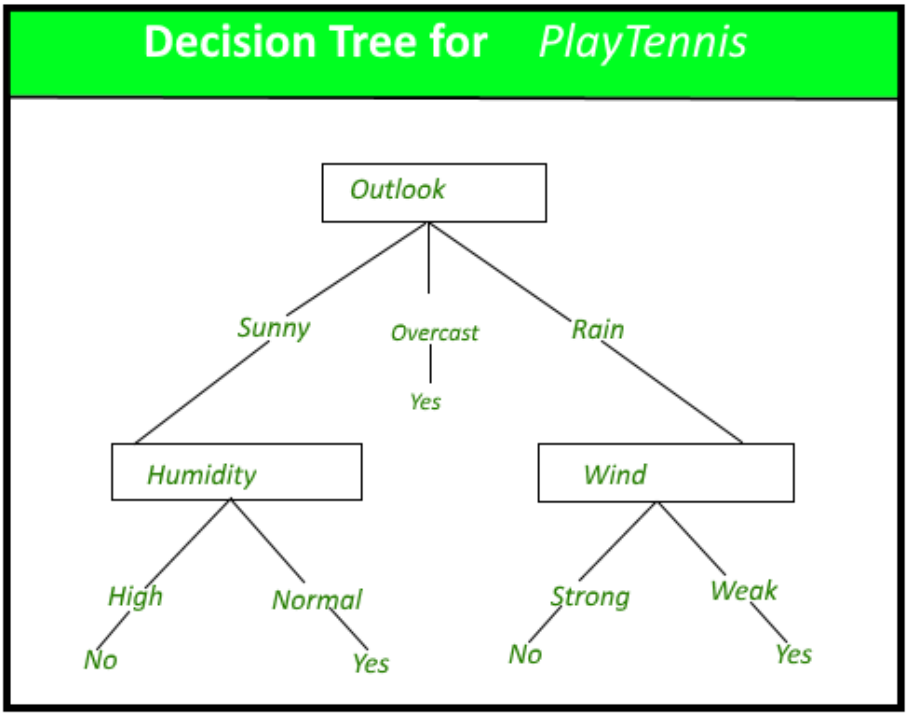
2.4. Các mô hình phân lớp

2.4.1. Decision Tree

https://trituenhantao.io/kien-thuc/decision-tree/

a. Khái niệm

Cây quyết định (Decision Tree) là công cụ mạnh mẽ và phổ biến nhất để phân loại và dự đoán. Cây quyết định là một cấu trúc cây giống như lưu đồ, trong đó mỗi nút bên trong biểu thị một phép thử trên một thuộc tính, mỗi nhánh biểu thị một kết quả của phép thử và mỗi nút lá (nút đầu cuối) giữ một nhãn lớp. [9]



Hình 4. Cây phân lớp

b. Xây dựng cây quyết định

Một cây có thể được "học" bằng cách chia tập hợp nguồn thành các tập con dựa trên kiểm tra giá trị thuộc tính. Quá trình này được lặp lại trên mỗi tập con dẫn xuất theo cách đệ quy được gọi là phân vùng đệ quy. Đệ quy được hoàn thành khi tất cả các tập hợp con tại một nút có cùng giá trị của biến mục tiêu hoặc khi việc tách không còn thêm giá trị cho các dự đoán. Việc xây dựng bộ phân loại cây quyết định không yêu cầu bất kỳ cài đặt tham số hoặc kiến thức miền nào và do đó phù hợp với khám phá kiến thức khám phá. Cây quyết định có thể xử lý dữ liệu nhiều chiều. Nhìn chung, bộ phân loại cây quyết định có độ chính xác tốt. Quy nạp cây quyết định là một phương pháp quy nạp điển hình để học kiến thức về phân loại.

c. Biểu diễn cây quyết định

Cây quyết định phân loại các thể hiện bằng cách sắp xếp chúng xuống cây từ gốc đến một số nút lá, cung cấp sự phân loại của thể hiện. Một thể hiện được phân loại bằng cách bắt đầu từ nút gốc của cây, kiểm tra thuộc tính được chỉ định bởi nút này, sau đó di

chuyển xuống nhánh cây tương ứng với giá trị của thuộc tính như trong hình trên. Quá trình này sau đó được lặp lại cho cây con bắt nguồn từ nút mới.

d. Ưu và nhược điểm

Ưu điểm:

- Cây quyết định có thể tạo ra các quy tắc dễ hiểu.
- Cây quyết định thực hiện phân loại mà không cần tính toán nhiều.
- Cây quyết định có thể xử lý cả biến liên tục và phân loại.
- Cây quyết định cung cấp một dấu hiệu rõ ràng về trường nào là quan trọng nhất để dự đoán hoặc phân loại.

Nhược điểm:

- Cây quyết định ít thích hợp hơn cho các nhiệm vụ ước tính trong đó mục tiêu là dự đoán giá trị của một thuộc tính liên tục.
- Cây quyết định dễ mắc lỗi trong các bài toán phân loại với nhiều lớp và một số ví dụ huấn luyện tương đối nhỏ.
- Cây quyết định có thể tốn kém về mặt tính toán để đào tạo. Quá trình phát triển một cây quyết định rất tốn kém về mặt tính toán. Tại mỗi nút, mỗi trường phân tách ứng cử viên phải được sắp xếp trước khi có thể tìm thấy phân tách tốt nhất của nó. Trong một số thuật toán, sự kết hợp của các trường được sử dụng và phải thực hiện tìm kiếm để có trọng số kết hợp tối ưu. Các thuật toán cắt tỉa cũng có thể tốn kém vì nhiều cây con ứng cử viên phải được hình thành và so sánh.

2.4.2. Support Vector Machine

a. Khái niệm

Support Vector Machine (SVM) là một hệ thống học có giám sát và được sử dụng cho các bài toán phân loại và hồi quy. Máy vectơ hỗ trợ được nhiều người cực kỳ ưa chuộng vì nó tạo ra độ chính xác đáng chú ý với sức mạnh tính toán ít hơn. Nó chủ yếu được sử dụng trong các vấn đề phân loại. Chúng tôi có ba loại học tập được giám sát, không giám sát và học tập tăng cường. Máy vectơ hỗ trợ là một bộ phân loại chọn lọc được xác định chính thức bằng cách chia siêu phẳng. [10]

Đưa ra dữ liệu đào tạo được gắn nhãn, thuật toán tạo ra siêu phẳng tốt nhất để phân loại các ví dụ mới. Trong không gian hai chiều, siêu phẳng này là một đường chia mặt phẳng thành hai phần mà mỗi lớp nằm ở hai bên. Mục đích của thuật toán máy vectơ hỗ trợ là tìm một siêu phẳng trong không gian N chiều phân loại riêng các điểm dữ liệu.

b. Ưu và nhược điểm

Ưu điểm:

- SVM hoạt động tương đối tốt khi có một biên độ phân ly dễ hiểu giữa các lớp.
- Nó hiệu quả hơn trong không gian nhiều chiều.
- Nó có hiệu quả trong trường hợp số lượng kích thước lớn hơn số lượng mẫu vật.
- SVM có hệ thống bộ nhớ tương đối.

Nhược điểm:

- Thuật toán SVM không được chấp nhận đối với các tập dữ liệu lớn.
- Nó không thực thi tốt khi tập dữ liệu có nhiều âm thanh hơn, tức là các lớp mục tiêu đang chồng chéo.
- Trong trường hợp số lượng thuộc tính cho mỗi điểm dữ liệu vượt quá số lượng mẫu dữ liệu đào tạo, máy vectơ hỗ trợ sẽ hoạt động kém.
- Vì phân loại SVM hoạt động bằng cách đặt các điểm dữ liệu, bên trên và bên dưới siêu phẳng phân loại nên không có sự làm rõ xác suất nào cho việc phân loại.

c. Ứng dụng

- Quan sát khuôn mặt
- Sắp xếp văn bản và siêu văn bản
- Nhóm các bức chân dung
- Ghi nhớ chữ viết tay
- Kiểm soát dự đoán tổng quát [11]

2.4.3. KNN

KNN là một mô hình đơn giản và trực quan nhưng vẫn có hiệu quả cao vì nó không tham số; mô hình không đưa ra giả định nào về việc phân phối dữ liệu. Hơn nữa, nó có thể được sử dụng trực tiếp để phân loại đa lớp. [12]

Thuật toán KNN có nhiều ứng dụng trong ngành đầu tư, bao gồm dự đoán phá sản, dự đoán giá cổ phiếu, phân bổ xếp hạng tín dụng trái phiếu doanh nghiệp, tạo ra chỉ số vốn và trái phiếu tùy chỉnh.

Lưu ý khi sử dụng thuật toán KNN

- Một thách thức quan trọng của KNN là xác định thước đo để tính khoảng cách giữa đối tượng cần phân lớp với các đối tượng còn lại trong cơ sở dữ liệu. Vì đây là lựa chọn mang tính chủ quan nên nếu chọn thước đo không phù hợp thì mô hình sẽ không hiệu quả. Ví dụ, nếu một nhà phân tích đang xem xét sự tương đồng về hiệu suất của các cổ phiếu khác nhau trên thị trường, anh ta có thể xem xét sử

dụng mối tương quan của lợi nhuận trong quá khứ giữa các cổ phiếu như một thước đo.

- Kiến thức về dữ liệu và hiểu biết về các mục tiêu của phân tích là các bước quan trọng trong quá trình xác định đặc điểm tương đồng. Kết quả KNN có thể nhạy cảm với việc bao gồm các đặc điểm không liên quan hoặc tương quan, do đó cần phải chọn các đặc điểm một cách thủ công. Bằng cách đó, nhà phân tích loại bỏ thông tin ít giá trị hơn để giữ thông tin liên quan và phù hợp nhất. Nếu được thực hiện chính xác, quá trình này sẽ tạo ra một thước đo khoảng cách điển hình hơn. Các thuật toán KNN có xu hướng hoạt động tốt hơn với một số lượng nhỏ các tính năng.
- Số k là siêu tham số của mô hình (hyperparameter), các giá trị khác nhau của k có thể dẫn đến các kết luận khác nhau. Ví dụ, để dự đoán xếp hạng tín dụng của trái phiếu chưa được xếp hạng, k nên là 3, 15 hay 50 trái phiếu tương tự nhất với trái phiếu chưa được xếp hạng? Nếu k là số chẵn, có thể không có phân loại rõ ràng. Chọn giá trị cho k quá nhỏ sẽ dẫn đến tỉ lệ lỗi cao và độ nhạy đối với các điểm dữ liệu bất thường mang tính cục bộ. Nhưng chọn giá trị cho k quá lớn sẽ làm giảm đi tính chất khái niệm láng giềng gần nhất vì lấy trung bình quá nhiều kết quả. Trên thực tế, một số kĩ thuật khác nhau có thể được sử dụng để xác định giá trị tối ưu cho k, cần chú ý đến số lượng các loại (categories) và phân vùng của chúng trong mô hình.

a. Ưu và nhược điểm

Ưu điểm của KNN

- Độ phức tạp tính toán của quá trình training là bằng 0.
- Việc dự đoán kết quả của dữ liệu mới rất đơn giản.
- Không cần giả sử gì về phân phối của các class.

Nhược điểm của KNN

- KNN rất nhạy cảm với nhiễu khi K nhỏ.
- Như đã nói, KNN là một thuật toán mà mọi tính toán đều nằm ở khâu test. Trong đó việc tính khoảng cách tới *từng* điểm dữ liệu trong training set sẽ tốn rất nhiều thời gian, đặc biệt là với các cơ sở dữ liệu có số chiều lớn và có nhiều điểm dữ

liệu. Với K càng lớn thì độ phức tạp cũng sẽ tăng lên. Ngoài ra, việc lưu toàn bộ dữ liệu trong bộ nhớ cũng ảnh hưởng tới hiệu năng của KNN.

2.4.4. Random Forest

Random Forests (RF) chỉ định một họ các phương thức học máy (Machine Learning), bao gồm các thuật toán khác nhau để tạo ra một tập hợp các cây quyết định, như thuật toán Breiman Forest được trình bày bởi Breiman [8] và thường được sử dụng trong tài liệu như một mô hình chuẩn. Bản chất của thuật toán rừng ngẫu nhiên đó là có thể kết hợp được nhiều cây quyết định thay vì chỉ đưa ra lựa chọn dựa trên quyết định của một cây. Từ đó, RF có thể giảm lỗi dự đoán và cải thiện hiệu suất dự đoán. Các bước học tập bao gồm xây dựng một tập hợp các cây quyết định, mỗi nhóm được điều khiển từ một tập hợp con ‘bootstrap, từ tập học ban đầu, tức là sử dụng nguyên tắc đóng bao và sử dụng phương pháp cảm ứng cây gọi là cây ngẫu nhiên. Một thuật toán cảm ứng như vậy, thường dựa trên thuật toán cây phân loại và hồi quy [13]

Phương pháp rừng ngẫu nhiên cho phép học song song từ nhiều cây quyết định được xây dựng và huấn luyện ngẫu nhiên với nhiều tập con chứa các mẫu khác nhau. Mỗi cây trong rừng được huấn luyện bởi một tập hợp con với dữ liệu được phân phối ngẫu nhiên theo nguyên tắc đóng bao và cũng có các tính năng ngẫu nhiên. Các kết quả cuối cùng được đưa ra dưới dạng giá trị trung bình của mỗi cây quyết định cho bài toán hồi quy hoặc được xác định bằng kết quả đa số cho bài toán phân loại dữ liệu. Với nhiều ưu điểm của rừng ngẫu nhiên (RF), thuật toán này đã được áp dụng rộng rãi trong nhiều ứng dụng khác nhau [14]

2.5. Các mô hình hồi quy

2.5.1. Linear Regression

Hồi quy tuyến tính là một loại thuật toán học máy có giám sát được sử dụng để dự đoán biến đầu ra dạng số liên tục (còn gọi là biến phụ thuộc) dựa trên một hoặc nhiều biến đầu vào (còn gọi là biến độc lập).

Trong hồi quy tuyến tính, mục tiêu là tìm mối quan hệ tuyến tính giữa các biến đầu vào và biến đầu ra, mối quan hệ này có thể được biểu diễn dưới dạng phương trình có dạng:

$$y = \beta0 + \beta1x1 + \beta2x2 + ... + \beta nxn + \varepsilon$$

Trong đó: y là biến đầu ra dự đoán, x1, x2, ..., xn là biến đầu vào, β0, β1, β2, ..., βn là các hệ số (còn gọi là trọng số hoặc tham số) xác định độ dốc của đường , và ε là thuật ngữ lỗi.

Thuật toán hồi quy tuyến tính học các giá trị tối ưu của các hệ số từ dữ liệu huấn luyện, giúp giảm thiểu tổng sai số bình phương giữa giá trị dự đoán và giá trị thực tế. Sau khi mô hình được đào tạo, nó có thể được sử dụng để đưa ra dự đoán về dữ liệu mới.

Hồi quy tuyến tính có thể được sử dụng cho nhiều ứng dụng, bao gồm:

- Dự đoán giá cổ phiếu, giá nhà ở hoặc dữ liệu tài chính khác.
- Dự báo doanh thu hoặc doanh thu dựa trên dữ liệu lịch sử.
- Phân tích tác động của quảng cáo đến doanh số.
- Dự đoán số lượng khách hàng sẽ ghé thăm một cửa hàng hoặc trang web.
- Xác định các yếu tố chính ảnh hưởng đến sự hài lòng của khách hàng hoặc sự gắn kết của nhân viên.

2.5.2. Random Forest Regressor

Random Forest Regressor là một thuật toán học máy có giám sát, sử dụng một tập hợp các cây quyết định để dự đoán một biến đầu ra dạng số liên tục (còn gọi là biến phụ thuộc) dựa trên một hoặc nhiều biến đầu vào (còn gọi là biến độc lập).

Trong Random Forest Regressor, thuật toán xây dựng nhiều cây quyết định bằng cách sử dụng tập con ngẫu nhiên của dữ liệu huấn luyện và tập con ngẫu nhiên của các biến đầu vào. Mỗi cây quyết định trong tập hợp đưa ra dự đoán về biến đầu ra và dự đoán cuối cùng được tính bằng cách lấy trung bình các dự đoán của tất cả các cây quyết định.

Công cụ hồi quy rừng ngẫu nhiên là một cải tiến so với cây quyết định đơn lẻ vì nó giảm tình trạng thừa và tăng độ chính xác cũng như độ bền của mô hình. Nó cũng cung cấp thước đo tầm quan trọng của từng biến đầu vào trong việc đưa ra dự đoán.

Tương tự như Linear Regression thì Random Forest Regressor cũng được dùng cho việc dự đoán:

- Dự đoán giá nhà ở hoặc dữ liệu tài chính khác.
- Dự báo doanh thu hoặc doanh thu dựa trên dữ liệu lịch sử.
- Phân tích tác động của các chiến dịch tiếp thị đến doanh số bán hàng.
- Dự đoán tỷ lệ rời bỏ khách hàng hoặc giá trị trọn đời.
- Xác định các yếu tố chính ảnh hưởng đến sự hài lòng của khách hàng hoặc sự gắn kết của nhân viên.

2.6. Mô hình Keras Neural Network

Keras là một open source cho Neural Network được viết bởi ngôn ngữ Python. Nó là một library được phát triển vào năm 2005 bởi Francois Chollet, là một kỹ sư nghiên cứu Deep Learning. Keras có thể sử dụng chung với các thư viện nổi tiếng như Tensorflow, CNTK, Theano. Một số ưu điểm của Keras như:

- Dễ sử dụng, dùng đơn giản hơn Tensor, xây dựng model nhanh.
- Run được trên cả CPU và GPU.
- Hỗ trợ xây dựng CNN , RNN hoặc cả hai. Với những người mới tiếp cận đến Deep như mình thì mình chọn sử dụng Keras để build model vì nó đơn giản,dễ nắm bắt hơn các thư viện khác. Dưới đây mình xin giới thiệu một chút về API này. [15]

Models

Trong Keras có hỗ trợ 2 cách dựng models là Sequential model và Function API. Với Sequential ta sử dụng như sau:

```
from keras.models import Sequential
from keras.layers import Dense, MaxPooling2D, Flatten, Convolution2D


model= Sequential()
model.add(Convolution2D(32,3,3, input_shape=(64,64,3), activation='relu'))
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Flatten())
model.add(Dense(output_dim=128,activation='relu'))
model.add(Dense(output_dim=1, activation='sigmoid'))

model.compile(optimizer='adam', loss='binary_crossentropy', metrics=['accuracy'])

model.fit(x_train, y_train,
                    batch_size = batch_size,
                    epochs = eposhs,
                    verbose =1,
                    validation_data =(x_test, y_test))
```

Hình 5. Sequential model trong Keras

Nội dung đoạn code trên như sau:

1. Khởi tạo models Sequential ( )
2. Tạo Convolutionnal Layers : Conv2D là convolution dùng để lấy feature từ ảnh với các tham số :

- filters : số filter của convolution
- kernel_size : kích thước window search trên ảnh
- strides : số bước nhảy trên ảnh

- activation : chọn activation như linear, softmax, relu, tanh, sigmoid. Đặc điểm mỗi hàm các bạn có thể search thêm để biết cụ thể nó ntn.
- padding : có thể là "valid" hoặc "same". Với same thì có nghĩa là padding =1.

3. Pooling Layers: sử dụng để làm giảm param khi train, nhưng vẫn giữ được đặc trưng của ảnh.

- pool_size : kích thước ma trận để lấy max hay average
- Ngoài ra còn có : MaxPooling2D, AvergaPooling1D, 2D (lấy max , trung bình) với từng size.

4. Dense ( ): Layer này cũng như một layer neural network bình thường, với các tham số sau:

- units : số chiều output, như số class sau khi train (chó , mèo, lợn, gà).
- activation : chọn activation đơn giản với sigmoid thì output có 1 class.
- use_bias : có sử dụng bias hay không (True or False)

5. Hàm compile: Ở hàm này chúng tôi sử dụng để training models như thuật toán train qua optimizer như Adam, SGD, RMSprop,..

- learning_rate : dạng float , tốc độc học, chọn phù hợp để hàm số hội tụ nhanh.

6. Hàm fit ():

- Bao gồm data train, test đưa vào training.
- Batch_size thể hiện số lượng mẫu mà Mini-batch GD sử dụng cho mỗi lần cập nhật trọng số .
- Epoch là số lần duyệt qua hết số lượng mẫu trong tập huấn luyện.
- Giả sử ta có tập huấn luyện gồm 55.000 hình ảnh chọn batch-size là 55 images có nghĩa là mỗi lần cập nhật trọng số, ta dùng 55 images. Lúc đó ta mất 55.000/55 = 1000 iterations (số lần lặp) để duyệt qua hết tập huấn luyện (hoàn thành 1 epochs). Có nghĩa là khi dữ liệu quá lớn, chúng tôi không thể đưa cả tập data vào train được, ta phải chia nhỏ data ra thành nhiều batch nhỏ hơn.

CHƯƠNG 3. CÁC KẾT QUẢ THỰC NGHIỆM

3.1. Bộ Dữ Liệu

Để crawl được data trên spotify đầu tiên cần phải đăng nhập vào tài khoản dưới trang web dành cho lập trình viên trên Spotify, sau đó tiến hành Create an app và điền đầy đủ các thông tin về dự án, và tiếp đó chúng tôi sẽ được Spotify cấp cho các loại mã như Client Id, Client Secret.

Bước 1. Import các thư viện cần thiết

```
import spotipy
from spotipy.oauth2 import SpotifyClientCredentials
import json
import pandas as pd
```

Hình 6. Thư viện cần thiết để crawl data

Bước 2. Dùng các thông tin vừa được cấp để kết nối đến Spotify

```
credentials = json.load(open('authorization.json'))
client_id = credentials['client_id']
client_secret = credentials['client_secret']

playlist_index = 0

playlists = json.load(open('playlist.json'))
playlist_uri = playlists[playlist_index]['uri']
like = playlists[playlist_index]['like']
```

```
client_credentials_manager = SpotifyClientCredentials(client_id=client_id,client_secret=client_secret)
sp = spotipy.Spotify(client_credentials_manager=client_credentials_manager, requests_timeout=10, retries=5)
```

```
uri = playlist_uri
username = uri.split(':')[2]
playlist_id = uri.split(':')[4]
```

Hình 7. Kết nối đến Spotify

Bước 3. Hàm để lấy dữ liệu từ Spotify

```
def get_playlist_tracks(username, playlist_id):
    results = sp.user_playlist_tracks(username,playlist_id)
    tracks = results['items']
    while results['next']:
        results = sp.next(results)
        tracks.extend(results['items'])
    results = tracks

    playlist_tracks_id = []
    playlist_tracks_titles = []
    playlist_tracks_artists = []
    playlist_tracks_first_artists = []
    playlist_tracks_first_release_date = []
    playlist_tracks_popularity = []
    playlist_tracks_album_name = []
    playlist_tracks_explicit = []

    for i in range(len(results)):
        print(i)
        if i == 0:
            playlist_tracks_id = results[i]['track']['id']
            playlist_tracks_titles = results[i]['track']['name']
            playlist_tracks_first_release_date = results[i]['track']['album']['release_date']
            playlist_tracks_album_name =  results[i]['track']['album']['name']
            playlist_tracks_popularity = results[i]['track']['popularity']
            playlist_tracks_explicit = results[i]['track']['explicit']

            artist_list = []
            for artist in results[i]['track']['artists']:
                artist_list= artist['name']
            playlist_tracks_artists = artist_list


            features = sp.audio_features(playlist_tracks_id)
            features_df = pd.DataFrame(data=features, columns=features[0].keys())
```

```
    features_df['title'] = playlist_tracks_titles
    features_df['all_artists'] = playlist_tracks_artists
    features_df['album_name'] = playlist_tracks_album_name
    features_df['popularity'] = playlist_tracks_popularity
    features_df['release_date'] = playlist_tracks_first_release_date
    features_df['explicit'] = playlist_tracks_explicit

    features_df = features_df[['id', 'title', 'all_artists','album_name','popularity','release_date','explicit','speechiness',
                               'danceability', 'energy', 'key', 'loudness',
                               'mode', 'acousticness', 'instrumentalness',
                               'liveness', 'valence', 'tempo',
                               'duration_ms', 'time_signature']]
    continue
else:
    try:
        playlist_tracks_id = results[i]['track']['id']
        playlist_tracks_titles = results[i]['track']['name']
        playlist_tracks_first_release_date = results[i]['track']['album']['release_date']
        playlist_tracks_album_name =  results[i]['track']['album']['name']
        playlist_tracks_popularity = results[i]['track']['popularity']
        playlist_tracks_explicit = results[i]['track']['explicit']

        artist_list = []
        for artist in results[i]['track']['artists']:
            artist_list= artist['name']
        playlist_tracks_artists = artist_list


        features = sp.audio_features(playlist_tracks_id)
        new_row = {'id':[playlist_tracks_id],
       'title':[playlist_tracks_titles],
       'all_artists':[playlist_tracks_artists],
       'album_name':[playlist_tracks_album_name],
       'popularity':[playlist_tracks_popularity],
       'release_date':[playlist_tracks_first_release_date],
       'explicit':[playlist_tracks_explicit],
       'speechiness':[features[0]['speechiness']],
                'danceability':[features[0]['danceability']],
                'energy':[features[0]['energy']],
                'key':[features[0]['key']],
                'loudness':[features[0]['loudness']],
                'mode':[features[0]['mode']],
                'acousticness':[features[0]['acousticness']],
                'instrumentalness':[features[0]['instrumentalness']],
                'liveness':[features[0]['liveness']],
                'valence':[features[0]['valence']],
                'tempo':[features[0]['tempo']],
                'duration_ms':[features[0]['duration_ms']],
                'time_signature':[features[0]['time_signature']]
                }

                dfs = [features_df, pd.DataFrame(new_row)]
                features_df = pd.concat(dfs, ignore_index = True)
            except:
                continue

    return features_df
```

Hình 8. Hàm lấy dữ liệu từ Spotify

Bước 4. Tiến hành lấy dữ liệu

```
df = get_playlist_tracks(username, playlist_id)
```

```
9942
9943
9944
9945
9946
9947
9948
9949
9950
9951
9952
9953
9954
9955
9956
9957
9958
9959
9960
9961
9962
9963
9964
9965
9966
9967
9968
9969
9970
9971
9972
9973
9974
```

Bước 5. Lưu df vào file csv

Dữ liệu crawl được gồm có: 9901 dòng và 23 cột

STT	Tên biến	Mô tả

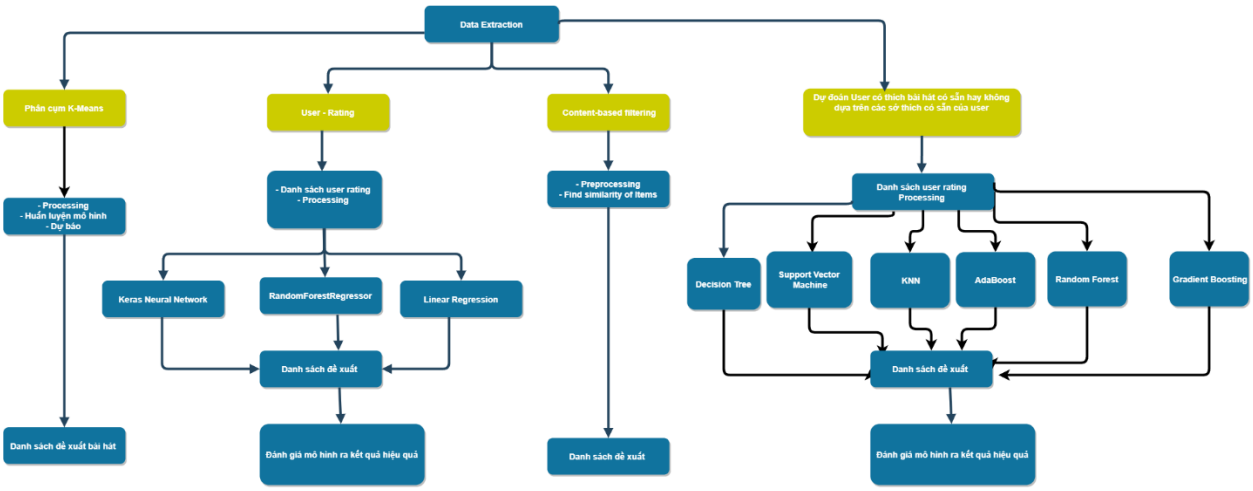
1	id	
2	title	Tên bản nhạc
3	all_artists	Nghệ sĩ trình diễn bản nhạc
4	album_name	Tên album chứa bản nhạc
5	popularity	Loại biến rời rạc, giá trị từ 0 đến 100, mô tả mức độ phổ biến của bài hát, càng cao càng phổ biến
6	release_date	Ngày phát hành
7	explicit	Cho biết bài hát có ẩn chứa từ ngữ, nội dung nhảy cảm liên quan đến tình dục, chém giết, chính trị tôn giáo,… hay không. Giá trị "TRUE" là "Có" và giá trị "FALSE" là "Không"
8	speechiness	Xác suất thể hiện có sự hiện diện của tiếng nói (khác với tiếng hát) trong bài hát hay không. Có giá trị trong khoảng 0.0 - 1.0, giá trị càng cao thì đây khả năng cao là bài diễn thuyết, sách nói,...
9	danceability	Mô tả bài hát có phù hợp để nhảy hay không dựa trên sự kết hợp, độ mạnh yếu của nhịp độ, nhịp điệu bài hát, có giá trị từ 0 - 1.0
10	energy	Mô tả độ dồn dập của bài hát, có giá trị từ 0.0 - 1.0, càng gần 1.0 thì bài hát càng tạo cảm giác dồn dập, dồn nén cao và ngược lại
11	key	Cao độ trung bình của hát, giá trị là số nguyên, tính theo chuẩn Pitch Class Notation, nếu không có cao độ giá trị là -1
12	loudness	Độ to trung bình của bài hát (tính theo đơn vị dB), giá trị thường rơi vào khoảng -60.0 - 0.0 dB
13	mode	Loại biến rời rạc, mô tả điệu thức của bài hát, chỉ có hai giá trị là 0 (giọng Thứ) và 1 (giọng Trưởng)
14	acousticness	Cường độ Acoustic của bài hát hay nói cách khác là xác suất bài hát này có tính chất acoustic, có giá trị từ 0.0 - 1.0

15	instrumentalness	Độ đo thể hiện tính instrumental (không lời) của bài hát, có giá trị trong khoảng 0.0 - 1.0, giá trị càng cao thì càng ít giọng hát trong bài hát
16	liveness	Độ đo thể hiện tính live (nhạc sống, có sự hiện diện của khán giả trong lúc thu âm) của bài hát, có giá trị trong khoảng 0.0 - 1.0
17	valence	Biểu thị tính tích cực của bài hát, giá trị càng cao thì bài hát càng có tính chất tích cực (vui, phấn khởi), càng thấp thì bài hát càng buồn
18	tempo	Mô tả tốc độ của bài hát, càng cao chứng tỏ bài đó có nhịp càng nhanh và ngược lại
19	duration_ms	Thời lượng bài hát
20	time_signature	Số chỉ nhịp của bài
21	artist_pop	Độ phổ biến của ca sĩ
22	genres_list	List thể loại nhạc của bài hát
23	genres	Thể loại của bài hát

Bảng 1. Mô tả các biến có trong tập dữ liệu

3.2. Các Kết Quả Thực Nghiệm

Để trực quan hoá bài nghiên cứu này, chúng tôi tóm tắt những bước cần làm trong sơ đồ sau:



Hình 9. Sơ đồ bài làm

Dữ liệu cần được tiền xử lý trước khi thực hiện tiến hành mô hình hoá bằng các thuật toán. Vì dữ liệu được nhập ban đầu là dữ liệu danh sách phát Spotify nên điều quan trọng là phải xóa các bài hát sao chép tồn tại giữa nhiều danh sách phát. Quá trình này bao gồm việc thu thập tên nghệ sĩ và tên bản nhạc để chúng tôi không vô tình xóa các bài hát có cùng tên nhưng của các nghệ sĩ khác nhau:

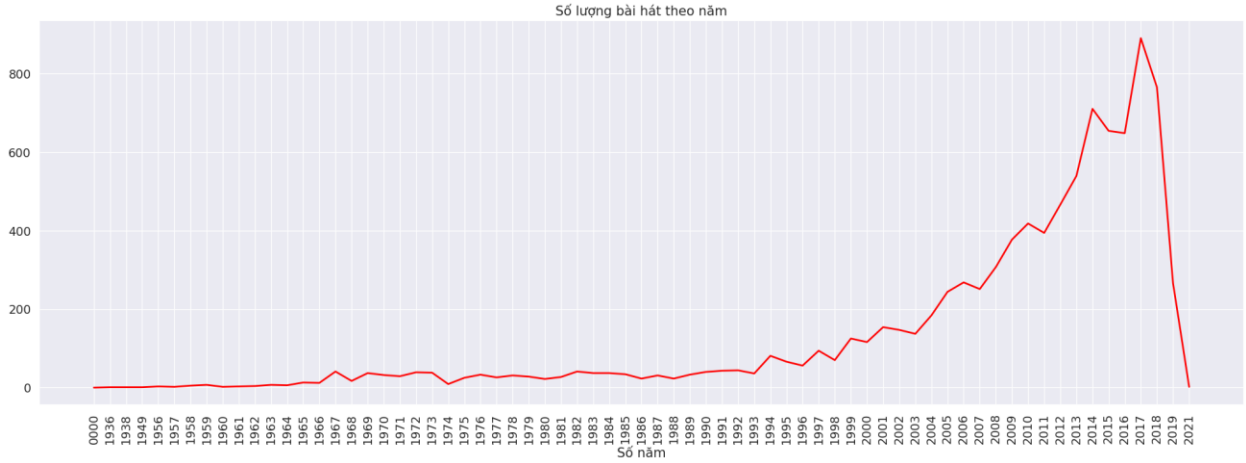
```
# Drop song duplicates
def drop_duplicates(df):
    '''
    Drop duplicate songs
    '''
    df['artists_song'] = df.apply(lambda row: row['all_artists']+row['title'],axis = 1)
    return df.drop_duplicates('artists_song')

songDF = drop_duplicates(playlistDF)
print("Are all songs unique: ",len(pd.unique(songDF.artists_song))==len(songDF))
```

Hình 10. Xử lý dữ liệu trùng lặp

Dữ liệu ban đầu có 9901 dòng, sau khi thực hiện bước xoá những bài hát trùng lặp thì còn lại 9353 dòng. Tiếp theo chúng tôi tiến hành lựa chọn những trường dữ liệu cần thiết phục vụ cho quá trình phân cụm dữ liệu:

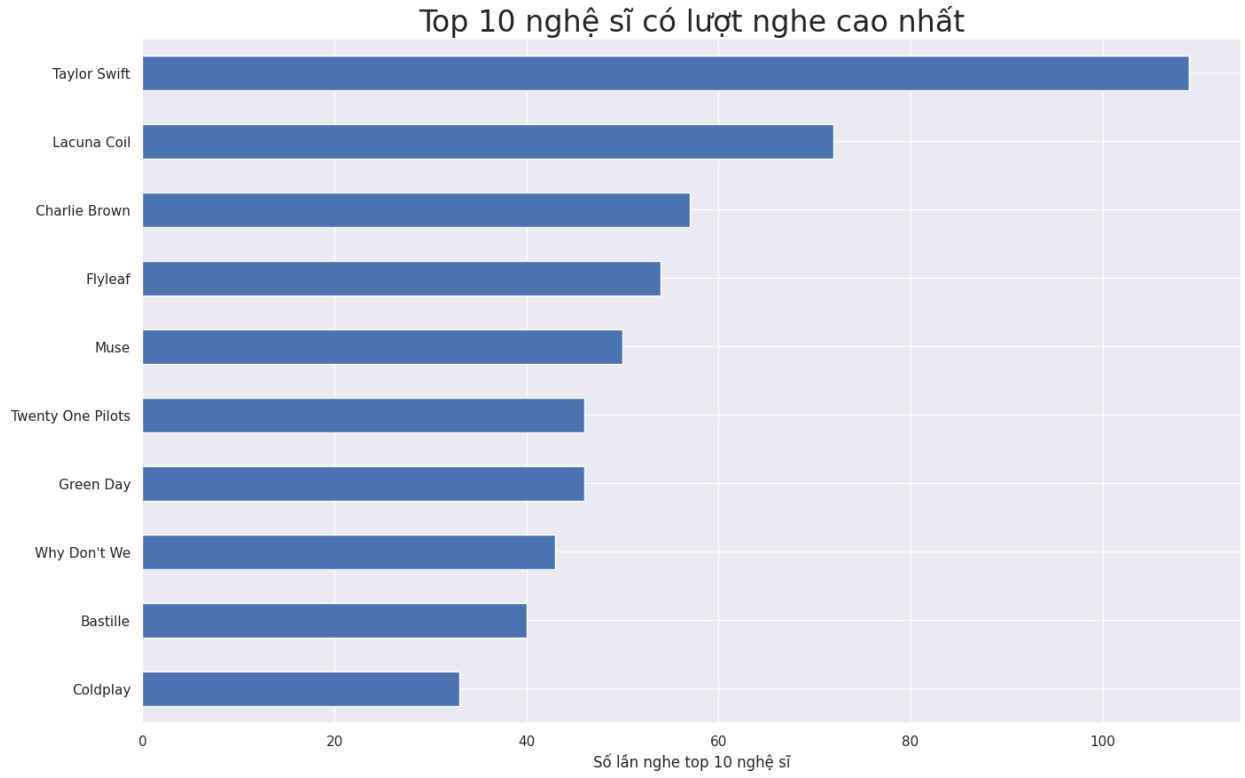
3.2.1. Exploratory Data Analysis (EDA)

Dữ liệu thống kê mô tả về dữ liệu Spotify



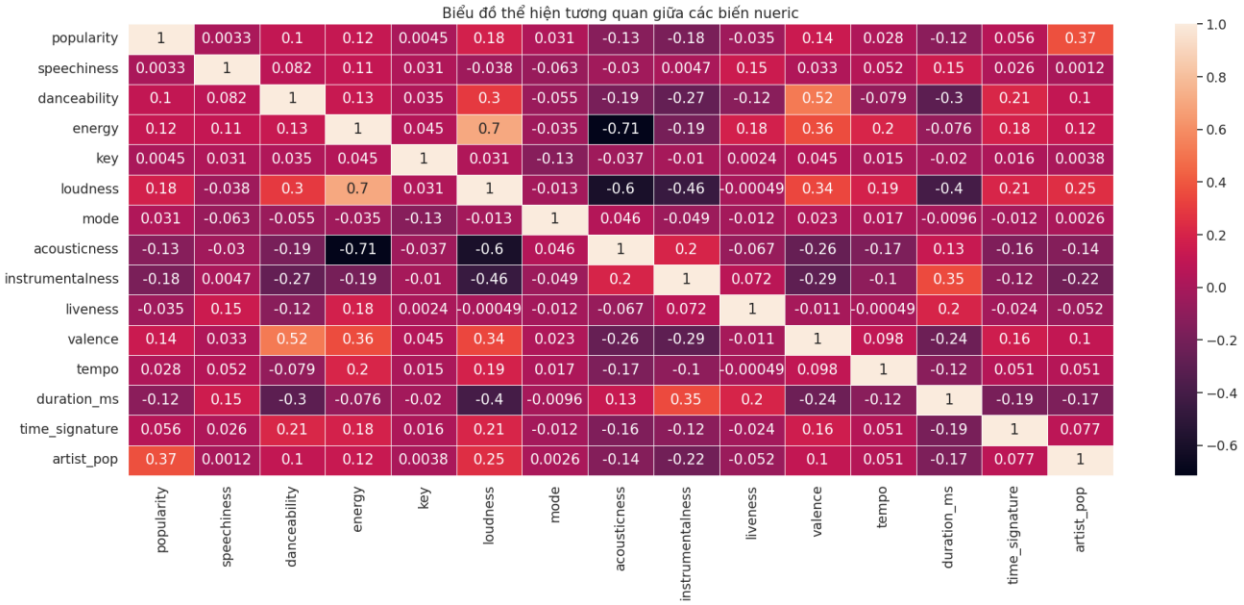
Hình 11. Số lượng bài hát theo hàng năm

Năm 2017 có lượt nghe cao nhất là 890 lượt nghe, năm 2018 có lượt nghe đứng thứ 2 là 765 lượt nghe



Hình 12. Top 10 nghệ sĩ có lượt nghe nhiều nhất

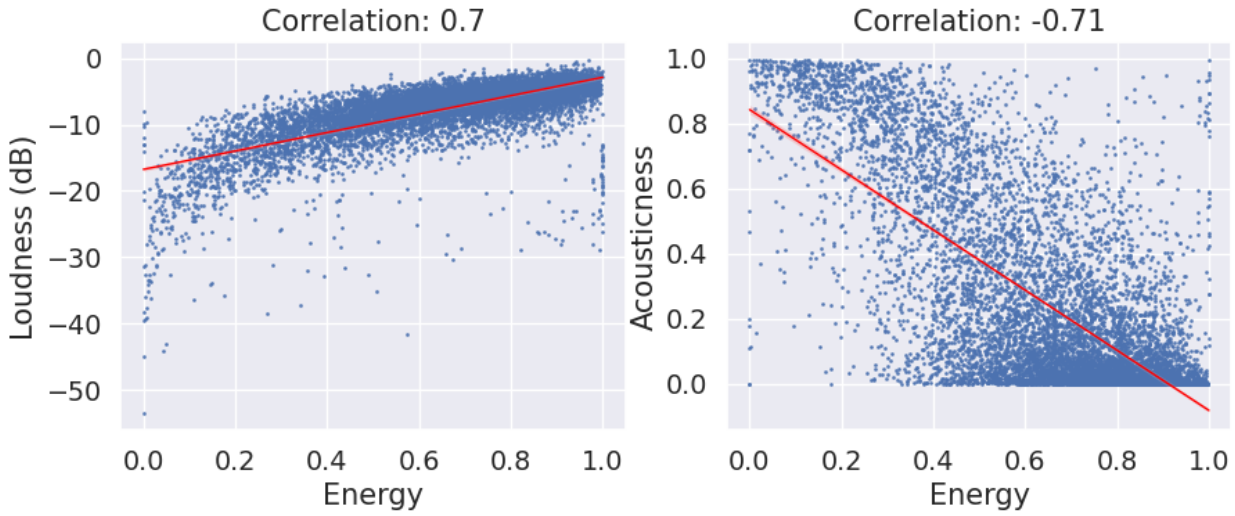
Nghệ sĩ Taylor Swift có lượt nghe cao nhất là 109 lượt nghe. Lacuna Coil là nghệ sĩ có 72 lượt nghe đứng sau Taylor Swift



Hình 13. Biểu đồ nhiệt thể hiện mối tương quan giữa các biến numeric

Bản đồ nhiệt tương quan dưới đây cho thấy các tính năng âm thanh Spotify liên quan đến nhau. Trong khi hầu hết các cặp tính năng có các hệ số tương quan (r) gần 0, cho thấy không có mối quan hệ tuyến tính có ý nghĩa, một số ít các cặp tính năng yếu đến liên quan đến vừa phải:

- Energy vs. loudness: hệ số tương quan r = 0,7
- Acousticness vs. energy có hệ số tương quan r = -0.71



Hình 14. Đặc trưng của âm thanh theo hệ số tương quan lớn nhất và nhỏ nhất với biến energy

Độ ồn(Loudness) của một bài hát được tính là trung bình của các giá trị âm lượng trong toàn bộ bài hát, và âm nhạc dày hơn, nhanh hơn tự nhiên dẫn đến độ ồn trung bình cao hơn nên energy cao hơn

Mối quan hệ giữa âm thanh (Acoustic) và năng lượng (energy) sẽ được xấp xỉ tốt hơn với mô hình phi tuyến tính, nhưng mối quan hệ cơ bản vẫn giữ được; Tỷ lệ các nhạc cụ âm thanh (acoustic) càng lớn trong một bài hát, năng lượng của bài hát càng thấp.

3.2.2. Đề xuất bài hát dựa trên phương pháp phân cụm K-Means

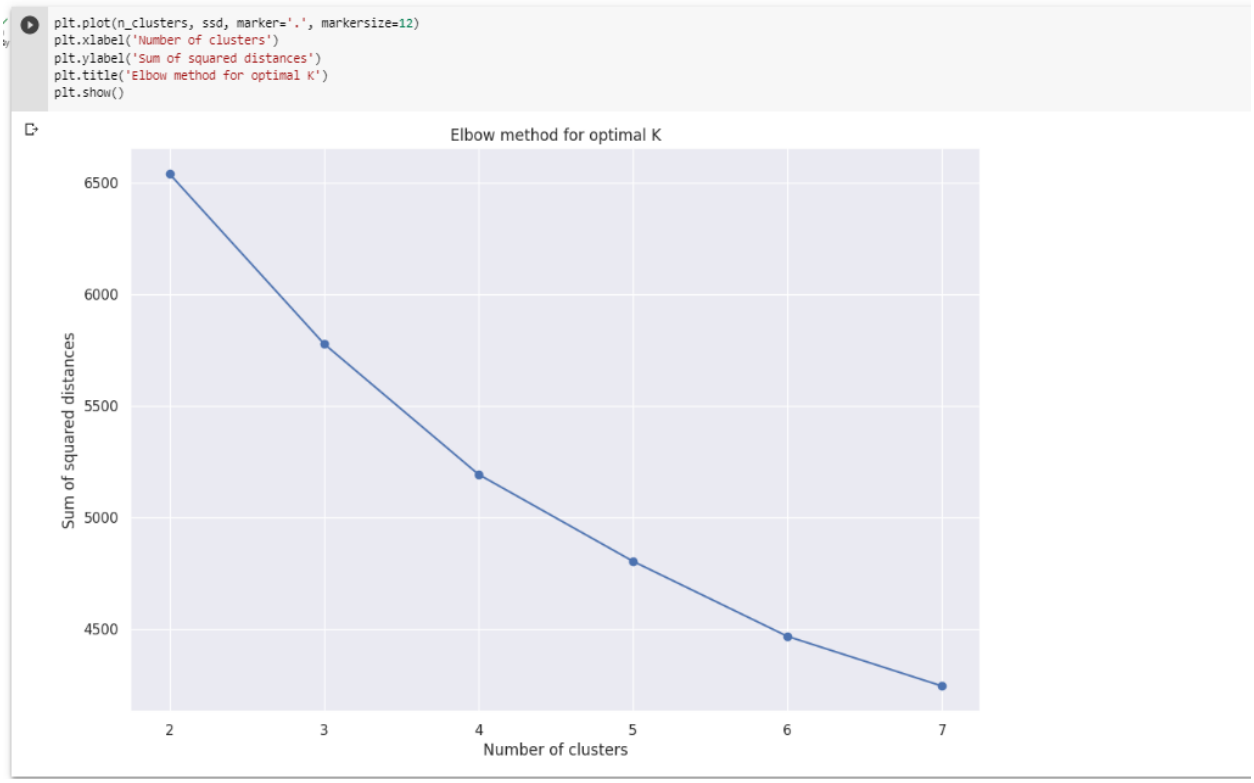
Chúng tôi tiến hành chuẩn hóa dữ liệu đã được chọn bằng phương pháp MinMaxScaler. Dữ liệu sau khi đã được chuẩn hoá như sau:

acousticness_scaled	danceability_scaled	energy_scaled	instrumentalness_scaled	liveness_scaled	valence_scaled	tempo_scaled	speechiness_scaled	loudness_scaled	popularity_scaled	explicit_scaled	key_scaled	m
0.066834	0.627551	0.779	0.000000	0.156883	0.457576	0.666370	0.141066	0.883785	0.000000	0.0	0.181818	
0.100503	0.701020	0.845	0.000000	0.045749	0.817172	0.366387	0.060188	0.922987	0.681818	0.0	0.636364	
0.034171	0.834694	0.803	0.000000	0.154858	0.638384	0.445510	0.083281	0.924643	0.875000	0.0	0.090909	
0.073869	0.636735	0.876	0.000000	0.330972	0.788889	0.416244	0.104493	0.941723	0.875000	0.0	0.818182	
0.610050	0.626531	0.379	0.000464	0.099798	0.203030	0.449754	0.049739	0.808051	0.943182	0.0	0.363636	
...	...	...	...	...	...	...	...	...	...	...	...	
0.019899	0.418367	0.761	0.084785	0.250000	0.194949	0.577709	0.048589	0.845166	0.000000	0.0	0.818182	
0.314573	0.414286	0.306	0.007908	0.073583	0.073737	0.563382	0.038871	0.808014	0.568182	0.0	0.272727	
0.017487	0.369388	0.507	0.216216	0.114372	0.460606	0.610732	0.041170	0.793962	0.500000	0.0	0.090909	
0.089146	0.337755	0.733	0.001742	0.958502	0.383838	0.535228	0.048903	0.842081	0.000000	0.0	0.181818	
0.001065	0.348980	0.932	0.000000	0.094939	0.228283	0.617008	0.090700	0.936644	0.000000	0.0	0.090909	

Hình 15. Dữ liệu sau khi scale

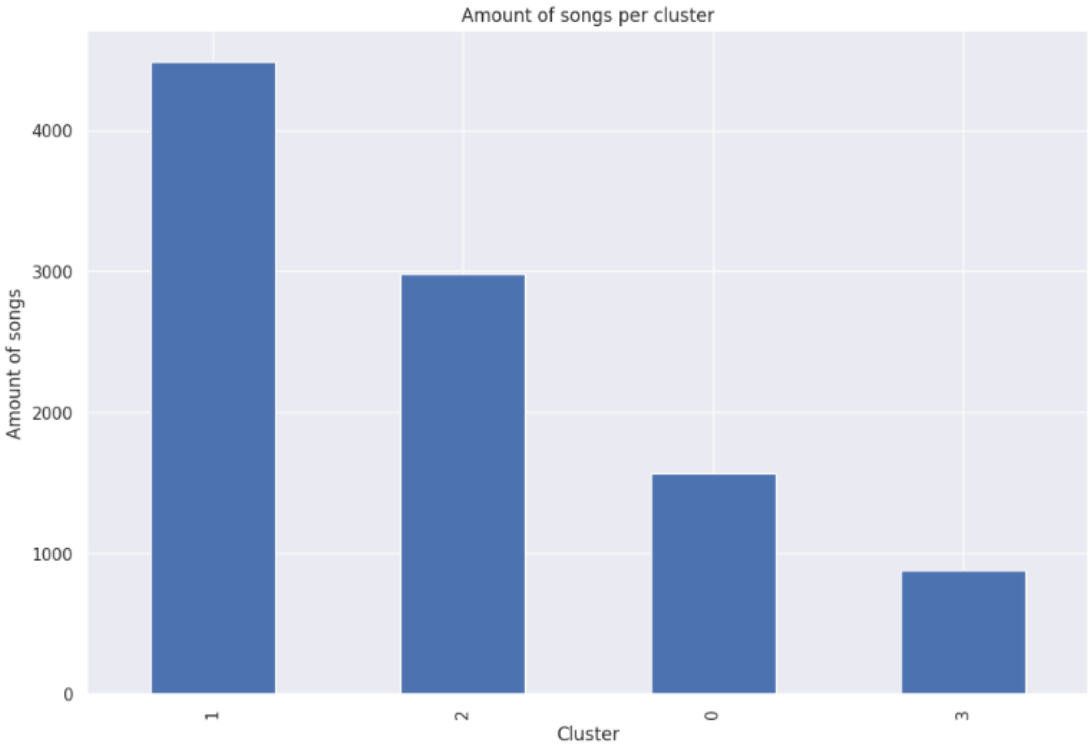
Để tiến hành phân cụm dữ liệu, bước đầu tiên chúng tôi cần chọn số cụm phù hợp, thực hiện huấn luyện bộ dữ liệu bao gồm danh sách những bài hát đã lấy được từ Spotify,

dựa vào kết quả chỉ số silhouette và phương pháp Elbow, chúng tôi đã quyết định tiến hành training dữ liệu với số cụm bằng 4



Hình 16. Elbow method

```
[28] df_songs_scaled['cluster'] = model.labels_

     df_songs_scaled['cluster'].value_counts().plot(kind='bar')
     plt.xlabel('Cluster')
     plt.ylabel('Amount of songs')
     plt.title('Amount of songs per cluster')
     plt.show()
```




```
display(df_songs_scaled['cluster'].value_counts())
minor_cluster = df_songs_scaled['cluster'].value_counts().tail(1)
print("Amount of songs in the smallest cluster: ", int(minor_cluster.values))
```

```
1    4487
2    2976
0    1561
3     877
Name: cluster, dtype: int64
Amount of songs in the smallest cluster:  877
```

Hình 17. Kết quả huấn luyện dữ liệu

Kết quả phân cụm cho ta thấy trong dữ liệu gồm 9900 bài hát đã được đưa vào huấn luyện, số lượng các bài hát được xếp lần lượt từ cụm 9 tới cụm 3 là : 1561, 4487, 2976, 877. Dưới đây là danh sách chi tiết một số bài hát được phân vào các cụm:

all_artists	title	album_name
Ben's Brother	Let Me Out	Let Me Out
P C III	Fortress	Sgons, Vol. 1
Faye	Screams and Dreams	Screams and Dreams
Teddy Sky	Fireproof	Fireproof
Willamette Stone	Heart Like Yours	If I Stay: Original Motion Picture Soundtrack:...
Kay Cook	Upgrade	Upgrade
Charles Vald	Nature Sounds - Rain, Birds and Thunder (60 Mi...	The Sounds of Rain - 6 Hours of Nature Sounds
Kraftwerk	Autobahn	3-D The Catalogue
James Horner	Never an Absolution	Titanic: Original Motion Picture Soundtrack - ...
Crowder	Come As You Are	Neon Steeple

all_artists	title	album_name
Taylor & Mad.S	Omerta 2016	Omerta 2016
Janove	Me Vokser Aldri Opp	Artisten & Marlene
Taylor Swift	Enchanted	Speak Now
Kid Rock	All Summer Long	Rock n Roll Jesus
Trace Adkins	Hillbilly Bone	Cowboy's Back In Town (Deluxe Edition)
In Flames	Trigger	Reroute to Remain (Reissue 2014)
Jason Derulo	Whatcha Say	Jason Derulo (Deluxe Audio)
Janove	I Natt Blir Du Fri	Artisten & Marlene
Charlie Brown	Henpecked	The Strongest Man
Sabrina Carpenter	Bad Time	Singular Act I

Hình 18. Danh sách bài hát được phân vào các cụm

Để việc huấn luyện mô hình đạt kết quả tốt hơn chúng tôi cũng có thể dùng phương pháp giảm chiều dữ liệu PCA. Phương pháp này không yêu cầu phải loại bỏ bất kì một biến đầu vào nào mà tất cả chúng sẽ được tận dụng nhằm tạo ra những biến được tổ hợp tuyến tính từ chúng. Chúng tôi tiến hành giảm chiều dữ liệu xuống còn 3 chiều và tiến hành huấn luyện mô hình, kết quả huấn luyện mô hình như sau:

```
[ ]  pca = PCA(n_components=3, random_state=42)
     songs_pca = pca.fit_transform(songs_scaled)
     pca.explained_variance_ratio_.sum()
```

```
0.5571589526831334
```

```
df_pca = pd.DataFrame(songs_pca, columns=['C1', 'C2', 'C3'])
df_pca['cluster'] = model.labels_
df_pca.head()
```

	C1	C2	C3	cluster
0	-0.369089	-0.103291	0.183379	1
1	-0.309729	-0.501823	-0.164496	1
2	-0.398227	-0.546084	0.410513	1
3	-0.280418	-0.638762	-0.297341	1
4	-0.410800	0.112326	0.176965	0

```
#Undersampling clusters
```

Hình 19. PCA giảm chiều dữ liệu

Một cách khác nữa cũng được sử dụng để giảm chiều dữ liệu là t-SNE, chúng tôi cũng thực hiện tương tự như quá trình PCA, kết quả phân cụm như sau:

Applying t-SNE to visualizing the clusters

```
[ ]  #tsne = TSNE(n_components=2, perplexity=50, n_iter=100, random_state=42, learning_rate=190)
     tsne = TSNE(n_components=2, perplexity=50, random_state=42)
     songs_tsne = tsne.fit_transform(songs_scaled)
```

```
[ ]  df_tsne = pd.DataFrame(songs_tsne, columns=['C1', 'C2'])
     df_tsne['cluster'] = model.labels_
     df_tsne.head()
```

	C1	C2	cluster
0	1.478166	15.801890	1
1	-35.511589	-34.481472	1
2	-2.044168	-28.611538	1
3	-43.134937	-26.854790	1
4	-56.525826	5.251867	0

Hình 20. t-SNE (giảm chiều dữ liệu)

Kết quả phân cụm được thể hiện dưới dạng biểu đồ:

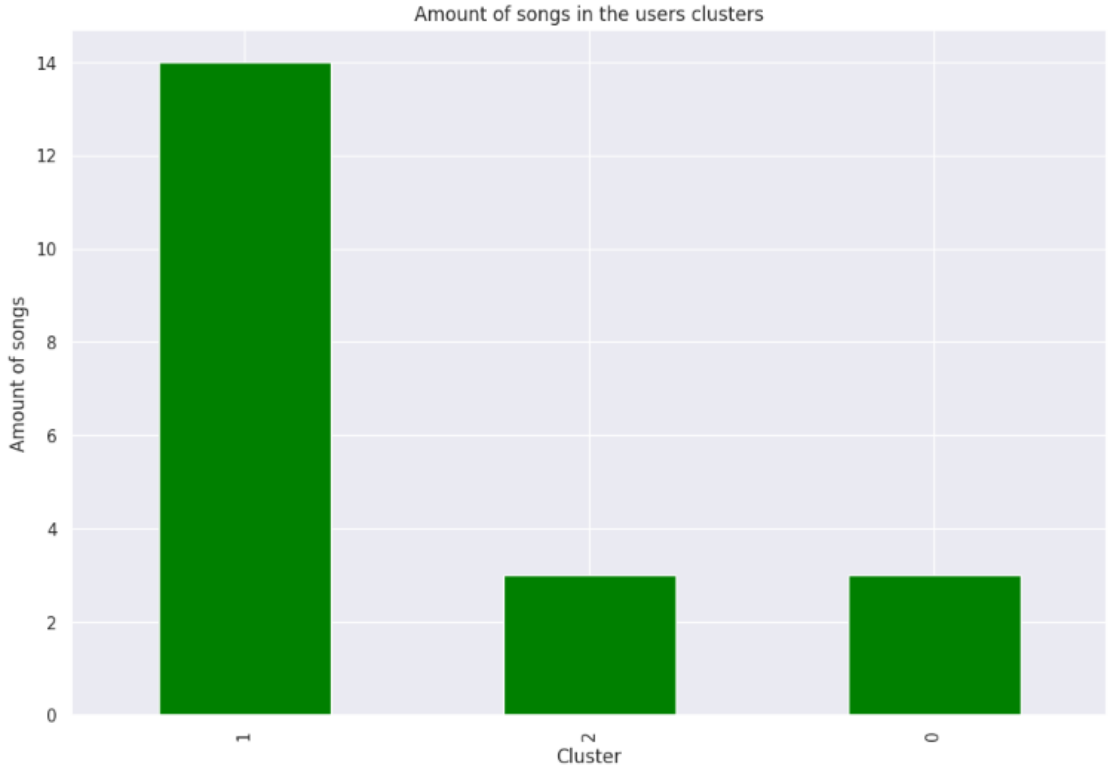
Hình 21. Trực quan hoá kết quả phân cụm

Sau khi đã huấn luyện mô hình, chúng tôi tiến hành đưa dữ liệu test gồm 20 bài hát được chọn ngẫu nhiên từ danh sách phát, và tiến hành dự đoán lần lượt 20 bài hát đó thuộc cụm nào trong số 4 cụm mà chúng tôi đã tiến hành huấn luyện:

```
user_pred = model.predict(user_scaled)
print('20  users clusters: ', user_pred[:20])
```

```
20  users clusters:  [1 2 1 2 1 1 1 1 1 0 1 1 1 0 1 1 0 2 1 1]
```

```
user_cluster = pd.DataFrame(user_scaled, columns=columns_to_cluster_scaled)
user_cluster['cluster'] = user_pred

user_cluster['cluster'].value_counts().plot(kind='bar', color='green')
plt.xlabel('Cluster')
plt.ylabel('Amount of songs')
plt.title('Amount of songs in the users clusters')
plt.show()
```



Hình 22. Kết quả phân cụm 20 bài hát

```
#Sample of songs in each cluster
df_user_songs_joined = pd.concat([playlistDF_test,user_cluster], axis=1).set_index('cluster')
for cluster in user_cluster['cluster'].unique():
    display(df_user_songs_joined.loc[cluster, ['all_artists','title','album_name']].sample(frac=1).head(10))
```

cluster	all_artists	title	album_name
1	Miley Cyrus	The Climb	Hannah Montana The Movie
1	Taylor Swift	I Know Places	1989 (Deluxe Edition)
1	Taylor Swift	The Outside	Taylor Swift
1	Taylor Swift	Crazier	Hannah Montana The Movie
1	Taylor Swift	Treacherous	Red (Deluxe Edition)
1	Taylor Swift	Long Live	Speak Now
1	Taylor Swift	The Lucky One	Red
1	Taylor Swift	Gorgeous	reputation
1	Taylor Swift	22	Red
1	Taylor Swift	Gorgeous	Gorgeous

cluster	all_artists	title	album_name
2	SZA	What Lovers Do (feat. SZA)	Red Pill Blues (Deluxe)
2	Taylor Swift	Look What You Made Me Do	reputation
2	Taylor Swift	I Did Something Bad	reputation

cluster	all_artists	title	album_name
0	RHODES	Let It All Go	Let It All Go
0	Colbie Caillat	Breathe	Fearless
0	Taylor Swift	State Of Grace - Acoustic	Red (Deluxe Edition)

Hình 23. Kết quả chi tiết danh sách 20 bài hát thuộc các cụm

```
#Analysing the percentage of songs per cluster
df_user_songs_joined.reset_index(inplace=True)
cluster_pct = df_user_songs_joined.cluster.value_counts(normalize=True)*100
display(cluster_pct)
print('Total songs: ', int(cluster_pct.round(0).sum()))
```

```
1    70.0
2    15.0
0    15.0
Name: cluster, dtype: float64
Total songs:  100
```

Hình 24. Phần trăm các bài hát có trong mỗi cụm

Kết quả dự đoán trong 20 bài hát được đưa vào dự đoán có 15% bài hát thuộc cụm 0, 70% bài hát thuộc cụm 1, 15% bài hát thuộc cụm 2, không có bài hát nào trong số 20 bài hát đưa vào huấn luyện cụm 3. Kết quả này chính là cơ sở để chúng tôi tiến hành lấy số lượng % bài hát để đề xuất cho danh sách nhạc tự động.

```
#Getting mixed samples from the original dataset and inserting in a new playlist, based on the percentage of songs present in each user cluster
playlist = pd.DataFrame()

for ncluster, pct in cluster_pct.items():
    songs = df_songs_joined[df_songs_joined['cluster'] == ncluster].sample(n=int(round(pct, 0)))
    playlist = pd.concat([playlist,songs], ignore_index=True)
    if len(playlist) > 10 :
        flag = 10 - len(playlist)
        playlist = playlist[:flag]

playlist = playlist[['all_artists','title','album_name','genres_list']]
```

```
playlist
```

	all_artists	title	album_name	genres_list
0	Sugar Ray	When It's Over - David Kahne Main	Sugar Ray	[alternative_metal, alternative_rock, funk_met...
1	LOONA/yyxy	love4eva (feat. Grimes)	beauty&thebeat	[k-pop_girl_group]
2	Bombay Bicycle Club	Shuffle	A Different Kind Of Fix	[alternative_dance, british_indie_rock, indie_...
3	George Ezra	Pretty Shining People	Staying at Tamara's	[folk-pop, neo-singer-songwriter]
4	Bruno Mars	Nothin' on You (feat. Bruno Mars)	B.o.B Presents: The Adventures of Bobby Ray	[atl_hip_hop, dance_pop, hip_hop, pop, pop_rap...
5	Flyleaf	Traitor	Between The Stars	[alternative_metal, christian_alternative_rock...
6	The Doors	Roadhouse Blues	Morrison Hotel	[acid_rock, album_rock, classic_rock, psychede...
7	Reel Big Fish	Your Girlfriend Sucks	Candy Coated Fury	[modern_ska_punk, pop_punk, punk, ska, ska_pun...
8	Quinn XCII	Always Been You	The Story of Us	[electropop, indie_pop_rap, pop, pop_rap]
9	Galantis	Runaway (U & I)	Pharmacy	[dance_pop, edm, electro_house, pop, pop_dance...

Hình 25. Danh sách bài hát được đề xuất từ K Means

Từ danh sách 20 bài hát được đưa vào huấn luyện, chúng tôi đã đề xuất ra 10 bài hát khác tưởng đồng với 20 bài hát đó từ danh sách những bài hát được cào từ Spotify.

3.2.3. Đề xuất bài hát dựa trên danh sách nhạc và đánh giá của người nghe

Chúng tôi có 1 user có sở thích nhạc pop và có được rating của user đó cho 20 bài hát thuộc thể loại nhạc pop.

```
indices = np.arange(user_database_scaled.shape[0])
random_indices = np.random.choice(indices, size=20, replace=False)
helen_X = user_database_scaled.loc[random_indices]
indices = [index for index in helen_X.index]
top_ids = user.loc[indices].id.values
for track_id in top_ids:
    print('https://open.spotify.com/track/{}'.format(track_id))
```

```
https://open.spotify.com/track/0CLnkxpa3M4eA3HeGaERat
https://open.spotify.com/track/6XDBA3QWX51lDJ0oZbaJJN
https://open.spotify.com/track/5P4wWhUYWM0IaVYLuZxdar
https://open.spotify.com/track/49mWEy5MgtNujgT7xU3emT
https://open.spotify.com/track/5vyxXfD5gLlyPxGZMEjtmd
https://open.spotify.com/track/28M2gifMU282QBM3fKajIS
https://open.spotify.com/track/4lIxdJw6W3Fg4vUIYCB0S5
https://open.spotify.com/track/2X2J0BhxaLTmnxO4pPUhSd
https://open.spotify.com/track/4svZDCRz4cJoneBpjpx8DJ
https://open.spotify.com/track/5G9AVKld9q7DCrmoY42raf
https://open.spotify.com/track/4nYsmWkuTaowTMy4gskmBw
https://open.spotify.com/track/5ybJm6GczjQOgTqmJ0BomP
https://open.spotify.com/track/1P17dC1amhFzptugyAO7Il
https://open.spotify.com/track/0XfOV7qY3834QpFVwOb6CC
https://open.spotify.com/track/2zfgVd034GlUvk7LqBHl6u
https://open.spotify.com/track/4WXzzCof26KJLTK5kK53dS
https://open.spotify.com/track/1ZY1PqizIl78geGM4xWlEA
https://open.spotify.com/track/70K0ezmzYEZeqoSaZMyP7o
https://open.spotify.com/track/2QA3IixpRcKyOdG7XDzRgv
https://open.spotify.com/track/0V8FYVlBFuXXTIvRnMbZyS
```

Hình 26. Danh sách 20 bài nhạc và rating

Còn đây là điểm (rating) mà user đã chấm cho 20 bài hết trên nằm trong thang từ 0 đến 10

Điểm mà user chấm cho 20 bài hát trên

```
[ ]  user_y = [8, 7.5, 8, 9, 6.5, 7.5, 7, 6, 9, 6, 7, 4, 9, 6, 8, 3, 6, 7.5, 6, 8.5]
```

Hình 27. Điểm chấm cho 20 bài hát

Tiếp theo chúng tôi loại bỏ đi những trường dữ liệu không cần thiết bằng cách dùng hàm drop và scale dữ liệu để phục vụ cho việc đề xuất danh sách phát

```
[ ]  X_database = df.drop(['all_artists','album_name', 'release_date', 'id', 'title', 'genres','genres_list'], axis=1)
     X_database.shape

     (9441, 17)
```

```
[ ]  X_database_scaled.head()
```

	popularity	speechiness	danceability	energy	key	loudness	acousticness	instrumentalness	liveness	valence	tempo	duration_ms	time_signature	artist_pop	release_year
0	-1.139116	0.875976	0.287847	0.550093	-0.899605	0.261705	-0.583898	-0.392560	-0.196752	-0.090842	1.410357	-0.200322	0.181937	2.072261	0.780670
1	1.042384	-0.160346	0.725922	0.846080	0.494867	0.743018	-0.467832	-0.392560	-0.913672	1.347492	-1.161299	-0.250568	0.181937	0.540240	0.780670
2	1.660475	0.135554	1.522975	0.657725	-1.178500	0.763342	-0.696499	-0.392560	-0.209811	0.632365	-0.483003	-0.150596	0.181937	1.459453	0.694716
3	1.660475	0.407355	0.342606	0.985104	1.052656	0.973051	-0.559645	-0.392560	0.926292	1.234364	-0.733892	-0.304809	0.181937	1.582014	0.780670
4	1.878625	-0.294238	0.281762	-1.243764	-0.341816	-0.668126	1.288752	-0.390605	-0.565006	-1.108988	-0.446621	-0.058318	0.181937	1.582014	0.522808

Hình 28. Dữ liệu sau khi đã scale

Linear Regression

Đầu tiên chúng tôi sử dụng mô hình hồi quy **Linear Regression** để tiến hành dự báo danh sách bài hát

```
simple_LR_model = LinearRegression().fit(helen_X, user_y)
preds_LR = simple_LR_model.predict(X_database_scaled)
database_full = X_database_scaled.copy() # df to hold both predictors and predicted ratings
database_full['predicted_ratings'] = preds_LR
database_sorted = database_full.sort_values(by=['predicted_ratings']) # sort from worst to best ratings
top5_lr = database_sorted[database_sorted.shape[0] - 5:] # top 5 ratings (this number can be increased to get more songs)
top5_lr
```

	danceability	energy	key	loudness	acousticness	instrumentalness	liveness	valence	tempo	duration_ms	time_signature	artist_pop	release_year	explicit	mode	predicted_ratings
	-3.026312	-2.270747	-1.178500	-1.623131	2.570676	3.412106	-0.622464	-1.760279	-1.732092	1.486016	0.181937	0.233836	0.952579	False	0.0	49.651816
	-3.058559	-2.648802	1.052656	-3.620443	2.369726	3.677548	-0.773944	-1.383726	-1.983659	-0.289141	0.181937	0.049993	-0.164825	False	0.0	49.690969
	-2.861426	-1.638412	-0.899605	-1.201866	2.082159	3.546934	-0.642052	-1.725532	-1.159478	-0.020181	0.181937	-0.011288	0.350900	False	0.0	49.915663
	-2.176325	-2.140692	-0.620711	-1.932382	2.518706	3.542720	-0.536277	-1.678665	-0.492465	0.141833	0.181937	0.907925	0.608762	False	0.0	50.174182
	-2.907058	-2.216931	0.215973	-2.216458	2.428625	3.761815	-0.758927	-1.796641	-1.658793	-0.286803	3.169178	-0.440254	0.007083	False	1.0	50.677639

Hình 29. Huấn luyện mô hình

```
df_top
```

	id	genres	title
0	0Vc8lKVpeK7SeUULvYVncH	australian_indie	Love Letter
1	4w0mTXaAAMhNH4qF0fniCC	deep_groove_house house pop_dance tropical_hou...	Touch
2	21AHQlJa5Q0s1Jx2PeulUD	brostep dance_pop edm electro_house electronic...	Love in the Middle of a Firefight (feat. Brend...
3	05wFHt1ZAsFU8GKbHV7nxo	alternative_rock anti-folk art_rock baroque_po...	The Party Line
4	5jOMxC9ezgFPGSMSn1zJ5S	permanent_wave punk	Peacemaker

+ Mã + Văn bản

*Hình 30. Danh sách đề xuất bài hát từ Mô hình **Linear Regression***

Danh sách bài hát được đề xuất có tổng ⅗ bài thuộc thể loại nhạc pop, chỉ số này khá là thấp so với kỳ vọng của chúng tôi.

RandomForestRegressor

Với mô hình Random Forest Regression, chúng tôi cũng tiến hành fit dữ liệu và dự đoán danh sách nhạc được đề xuất

```
from sklearn.ensemble import RandomForestRegressor

est_rf = RandomForestRegressor(random_state=15).fit(helen_X, user_y)
preds_rf = est_rf.predict(X_database_scaled)

database_rf = X_database_scaled.copy()
database_rf['predicted_ratings'] = preds_rf

database_rf_sorted = database_rf.sort_values(by=['predicted_ratings'])
top5_rf = database_rf_sorted[database_rf_sorted.shape[0]-5:]
top5_rf
```

danceability	energy	key	loudness	acousticness	instrumentalness	liveness	valence	tempo	duration_ms	time_signature	artist_pop	release_year	explicit	mode	predicted_ratings
-0.046794	0.106114	0.215973	-0.014057	1.108589	-0.392560	-0.451396	-0.632237	-1.110636	-0.296497	0.181937	0.540240	0.866624	False	1.0	8.345
-0.052878	0.541124	-1.178500	0.590356	0.259747	-0.392560	-0.523218	-0.591835	-1.195432	-0.135976	0.181937	0.969206	0.522808	False	0.0	8.350
-0.770834	-0.257142	-1.457395	-0.322846	0.339434	-0.384217	-0.536277	-0.506989	1.596444	-0.286088	0.181937	1.459453	0.780670	False	0.0	8.420
-0.326675	-0.046364	-1.178500	-0.287741	-0.131759	-0.392463	-0.583288	-0.721123	-1.136700	-0.130618	0.181937	2.010980	0.866624	False	1.0	8.430
0.263509	-0.871538	-1.457395	0.123131	1.118983	-0.392555	-0.353456	-0.397902	-2.173031	-0.180660	0.181937	1.275610	0.866624	True	0.0	8.440

Hình 31. Huấn luyện mô hình

```
df_top
```

	id	genres	title
0	4Tid4MwqgR1CfKCun3tFon	electropop indie_poptimism pop pop_dance pop_e...	Kings of Summer - Single Version
1	2lYTJK94hb0fd1LQtb6Dhk	big_room dance_pop edm pop pop_dance	Don't Leave Me Alone (feat. Anne-Marie)
2	0iwsQWgtjSq2kUXuZwTDAL	alt_z dance_pop pop post-teen_pop	Home with You
3	715sEYWkafd4xvl87dwZgu	album_rock classic_rock mellow_gold new_wave_p...	The Heart Of Rock And Roll
4	2RjpDxBii8EvC6p6wxwCSz	pop	Harder To Breathe

Hình 32. Danh sách đề xuất bài hát từ Mô hình Mô hình Random Forest Regression

Kết quả của mô hình Random Forest Regression cho ra ⅘ bài hát được đề xuất đều là thể loại nhạc pop. Vì vậy mô hình này hoạt động khá tốt.

Keras Neural Network

Đầu tiên, chúng tôi tiến hành xây dựng mô hình, chọn thuật toán cập nhật nghiệm, xây dựng loss và phương pháp đánh giá mô hình.

```
[ ]  helen_X_nn = np.array(helen_X, dtype=np.float32)
     user_y_nn = np.array(user_y, dtype=np.float32)
```

```
[ ]  model = Sequential([
         Dense(200, input_shape=(17,), activation='relu'),
         Dense(150, activation='relu'),
         Dense(100, activation='relu'),
         Dense(1, activation='relu')
     ])
```

```
[ ]  model.compile(loss='mean_absolute_error', optimizer='adam')
     model.summary()
```

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 dense (Dense)               (None, 200)               3600

 dense_1 (Dense)             (None, 150)               30150

 dense_2 (Dense)             (None, 100)               15100

 dense_3 (Dense)             (None, 1)                 101

=================================================================
Total params: 48,951
Trainable params: 48,951
Non-trainable params: 0
_________________________________________________________________
```

Hình 33. Xây dựng mô hình huấn luyện

Tiếp theo chúng tôi sẽ huấn luyện mô hình và tiến hành dự báo:

```
[ ] model.fit(helen_X_nn, user_y_nn, epochs=50, batch_size=32, validation_split = .2)

    Epoch 1/50
    1/1 [==============================] - 2s 2s/step - loss: 6.9436 - val_loss: 6.9157
    Epoch 2/50
    1/1 [==============================] - 0s 59ms/step - loss: 6.7998 - val_loss: 6.7391
    Epoch 3/50
    1/1 [==============================] - 0s 85ms/step - loss: 6.6064 - val_loss: 6.5411
    Epoch 4/50
    1/1 [==============================] - 0s 57ms/step - loss: 6.4053 - val_loss: 6.3274
    Epoch 5/50
    1/1 [==============================] - 0s 79ms/step - loss: 6.1986 - val_loss: 6.1038
    Epoch 6/50
    1/1 [==============================] - 0s 37ms/step - loss: 5.9799 - val_loss: 5.8656
    Epoch 7/50
    1/1 [==============================] - 0s 37ms/step - loss: 5.7448 - val_loss: 5.6030
    Epoch 8/50
    1/1 [==============================] - 0s 55ms/step - loss: 5.4872 - val_loss: 5.3094
    Epoch 9/50
    1/1 [==============================] - 0s 41ms/step - loss: 5.2035 - val_loss: 4.9895
    Epoch 10/50
    1/1 [==============================] - 0s 35ms/step - loss: 4.8864 - val_loss: 4.6424
    Epoch 11/50
    1/1 [==============================] - 0s 35ms/step - loss: 4.5333 - val_loss: 4.2637
    Epoch 12/50
    1/1 [==============================] - 0s 36ms/step - loss: 4.1391 - val_loss: 3.8437
    Epoch 13/50
```

	id	genres	title
0	15Trb1S2FDZSMLDzWfnlbg	irish_rock pop_rock	Promises
1	5BhMoGrz5KzG2fA5uzHjZ1	beatlesque british_invasion classic_rock merse...	Don't Let Me Down - Naked Version / Remastered...
2	5clz8BOBulxx7q2yHxqOCk	album_rock birmingham_metal classic_rock hard_...	Children of the Grave - 2014 Remaster
3	1febXMopmNMoPDsOLAbAP3	black_metal death_metal metal pagan_black_meta...	No Sympathy For Fools
4	48JM82SHQTBYkhEYhMaOaa	beatlesque british_invasion classic_rock merse...	She Loves You - Remastered 2009

Hình 34. Dự báo

Kết quả mô hình Neural network cho ra 1 bài thể loại pop trong top 5 bài có predict ratings cao. Do đó mô hình này được đánh giá không hiệu quả.

3.2.4. Dự đoán xem liệu user có thích bài hát được đưa vào hay không dựa trên các sở thích có sẵn của user

Đầu tiên chúng tôi tiến hành đưa tập dữ liệu danh sách nhạc, ở đây ta sẽ giả định cột Liked với 1 là ý thể hiện user thích bài hát và 0 là thể hiện user không thích bài hát.

```
df_non_ratings['Liked'].value_counts()
```

```
1    7591
0    1850
Name: Liked, dtype: int64
```

Hình 35. Dữ liệu đánh giá sở thích theo bài hát

Chúng tôi có 1850 bài hát user không thích và 7591 bài hát mà user thích. Sau đó chúng tôi tiến hành loại bỏ những trường dữ liệu không cần thiết cho việc dự đoán mô hình, tiến hành scale những trường dữ liệu cần thiết:

```
df_scale = df_non_ratings.drop(['all_artists','album_name', 'release_date', 'id', 'title', 'genres','genres_list','key'], axis=1)
df_scale.shape
```

```
(9441, 17)
```

Hình 36. Loại bỏ những trường dữ liệu không cần thiết

Chúng tôi tiến hành lần lượt xây dựng các mô hình: Decision Tree, Support Vector Machine, KNN, AdaBoost, Random Forest, Gradient Boosting và tiến hành dự báo, đánh giá hiệu quả mô hình. Kết quả đánh giá mô hình được thể hiện trong bảng sau:

Score	Decision Tree	Support Vector Machine	KNN	AdaBoost	Random Forest	Gradient Boosting
Accuracy	0.650079	0.79513	0.7946	0.7946	0.789836	0.79513
Precision	0.789401	0.79513	0.795021	0.795334	0.794667	0.795443
f1_score	0.776311	0.885874	0.885546	0.885478	0.88244	0.885807
AUC	0.486475	0.5	0.499667	0.500626	0.498589	0.500959
Recall	0.763648	1	0.999334	0.998668	0.992011	0.999334

Hình 37. Chỉ số đánh giá các mô hình

Precision: tỷ lệ giữa dự đoán ta đoán sẽ xảy ra là đúng sự thật sẽ xảy ra trong thực tế

Recall Sensitivity: trong số những trường hợp đã xảy ra trong thực tế thì có bao nhiêu trường hợp ta dự đoán đúng được thực hiện bởi mô hình của ta ứng dụng

Accuracy: tỷ lệ dự đoán đúng trong số tất cả những dự đoán của chúng ta đặt ra. Nhưng Accuracy chỉ thật sự đúng để đánh giá mô hình khi dự đoán ta đưa ra so với thực tế là gần như bằng nhau

Trong các phương pháp trên, theo chỉ số đánh giá độ chính xác của mô hình (Accuracy), chúng tôi nhận thấy các mô hình dự đoán hầu đa đều có kết quả khá tốt, trong đó mô hình SVM và Gradient Boosting có chỉ số Accuracy cao nhất.

3.2.5. Phương pháp Lọc dựa trên nội dung: Content-based filtering

Đầu tiên, tiến hành phân loại các biến trong dữ liệu thành ba loại dựa trên nguồn dữ liệu, đó là metadata (siêu dữ liệu), audio (dữ liệu âm thanh) và text (dữ liệu văn bản).

Các trường dữ liệu cụ thể được chọn ra và phân loại như sau:

- Metadata
  - id
  - genres_list
  - artist_pop
  - track_pop
- Audio
  - Mood: Danceability, Valence, Energy, Tempo
  - Properties: Loudness, Speechiness, Instrumentalness
  - Context: Liveness, Acousticness
  - metadata: key, mode
- Text
  - title

Đối với dữ liệu dạng **text**, chúng tôi lựa chọn trường dữ liệu tên bài hát (title) và tiến hành sentiment analysis để phân tích theo 2 yếu tố:

- Subjectivity: thước đo xem văn bản chứa bao nhiêu ý kiến hoặc cảm xúc cá nhân, trái ngược với sự thật khách quan. Nó được thể hiện dưới dạng giá trị, low, high và medium trong đó low cho biết văn bản hoàn toàn khách quan (không chứa ý kiến cá nhân) và high cho biết rằng văn bản hoàn toàn chủ quan (chỉ chứa ý kiến cá nhân, và medium cho biết bài hát thuộc dạng trung lập, không hoàn toàn khách quan cũng không hoàn toàn chủ quan
- Polarity: là thước đo mức độ tình cảm được thể hiện trong văn bản, có tính đến sự phủ định. Nó cho biết tên bài hát thể hiện sự tiêu cực, tích cực, hoặc là không tích cực cũng không tiêu cực

	title	all_artists	subjectivity	polarity
0	...Ready For It?	Taylor Swift	low	Neutral
1	Life Changes	Thomas Rhett	low	Neutral
2	24K Magic	Bruno Mars	high	Positive
3	Galway Girl	Ed Sheeran	low	Neutral
4	Photograph	Ed Sheeran	low	Neutral
...	...	...	...	...
9895	Remember the Future, Pt. 1	Nektar	low	Neutral
9896	Funeral For A Friend / Love Lies Bleeding	Elton John	high	Positive
9897	Fool's Overture	Supertramp	low	Neutral
9898	Heart of the Sunrise - 2003 Remaster	Yes	low	Neutral
9899	Venus And Mars / Rock Show / Jet - Live / Rema...	Wings	high	Positive

9353 rows × 4 columns

Hình 38. Phân tích tình cảm tên bài hát

Mục tiêu của phân tích tình cảm là trích xuất các tính năng bổ sung từ các bản nhạc. Bằng cách đó, chúng tôi có thể trích xuất dữ liệu cảm xúc cho các tính năng âm thanh khác thông qua thông tin văn bản. Ví dụ: nếu tâm trạng chung của tiêu đề bài hát trong danh sách phát là tích cực, thì điều này có thể được sử dụng để đề xuất các bài hát tích cực.

Đối với dữ liệu dạng metadata (siêu dữ liệu), chúng tôi tiến hành sử dụng phương pháp TF-IDF cho trường dữ liệu genres (thể loại) để tính toán thể loại nổi bật nhất trong mỗi bài hát và mức độ phổ biến của chúng trong các bài hát và xác định trọng số của thể loại. Để thực hiện điều này, chúng tôi đã sử dụng chức năng TfidfVectorizer() từ thư viện Scikit learn.

Đối với dữ liệu về mức độ phổ biến ‘artist_pop’ , tôi đã sử dụng nó như một biến liên tục và chỉ chuẩn hóa nó thành một phạm vi từ 0 đến 1. Ý tưởng là những bài hát phổ biến có khả năng được nghe bởi những người thích ca sĩ nổi tiếng, trong khi những bài hát ít phổ biến hơn có thể sẽ được nghe bởi những người có cùng sở thích.

```
[ ]  pop = songDF[["artist_pop"]].reset_index(drop = True)
     scaler = MinMaxScaler()
     pop_scaled = pd.DataFrame(scaler.fit_transform(pop), columns = pop.columns)
     pop_scaled.head()
```

	artist_pop
0	1.00
1	0.75
2	0.90
3	0.92
4	0.92

Hình 39. Scale trường dữ liệu "artist_pop"

Đối với các dữ liệu thuộc Audio chúng tôi cũng tiến hành mã hóa bằng phương pháp MinMaxScaler từ thư viện scikit learn

Sau khi đã xử lý các bước cần thiết cho tất cả dữ liệu của từng bài hát, bây giờ chúng tôi sẽ tiến hành quá trình thực hiện thuật toán lọc dựa trên nội dung. Quá trình này bao gồm 2 bước và 2 bước này là cần thiết mỗi khi một số người yêu cầu danh sách phát bài hát mới.

Bước đầu tiên, chúng tôi sẽ tóm tắt tất cả các bài hát trong danh sách phát thành một vectơ có thể so sánh với tất cả các bài hát khác trong bộ dữ liệu để tìm điểm tương đồng của chúng. Chúng tôi sẽ nhập một khung dữ liệu danh sách phát. Điều duy nhất cần thiết trong khung dữ liệu là id bản nhạc. Có dữ liệu id bản nhạc, trước tiên chúng tôi có thể tách biệt giữa các bài hát có trong danh sách phát và những bài hát không có. Điều quan trọng là phải loại trừ các bài hát trong danh sách phát vì chúng tôi không muốn đề xuất các bài hát hiện có. Cuối cùng, chúng tôi cộng tất cả các giá trị đặc trưng của từng bài hát trong danh sách phát lại với nhau dưới dạng một vectơ tóm tắt.

Bước thứ 2: Sau khi truy xuất vector tóm tắt danh sách phát và các bài hát không có trong danh sách phát, chúng tôi dùng độ đo cosine để tìm ra sự tương đồng giữa các bài hát trong danh sách phát Độ đo cosine là một giá trị toán học đo lường độ tương tự giữa các vectơ.

```
def generate_playlist_recos(df, features, nonplaylist_features):
    '''
    Generated recommendation based on songs in aspecific playlist.
    ---
    Input:
    df (pandas dataframe): spotify dataframe
    features (pandas series): summarized playlist feature (single vector)
    nonplaylist_features (pandas dataframe): feature set of songs that are not in the selected playlist

    Output:
    non_playlist_df_top_40: Top 40 recommendations for that playlist
    '''

    non_playlist_df = df[df['id'].isin(nonplaylist_features['id'].values)]
    # Find cosine similarity between the playlist and the complete song set
    non_playlist_df['sim'] = cosine_similarity(nonplaylist_features.drop('id', axis = 1).values, features.values.reshape(1, -1))[:,0]
    non_playlist_df_top_40 = non_playlist_df.sort_values('sim',ascending = False).head(40)

    return non_playlist_df_top_40
```

Hình 40. Tìm điểm tương đồng giữa các bài hát

Trong đoạn mã này, chúng tôi đã sử dụng hàm cosine_similarity() từ thư viện scikit learn để đo mức độ giống nhau giữa mỗi bài hát và vectơ danh sách phát được tóm tắt.

	id	title	all_artists	album_name	popularity	release_date	explicit	speechiness	danceability	energy	key	loudness	mode	acousticness	instrumentalness
1106	2R7C9dDqv1UPycvepBFziA	Superman	Taylor Swift	Speak Now (Deluxe Edition)	51	2010-10-25	False	0.0324	0.582	0.765	7	-3.648	1	0.02680	0.000002
1049	0dBW6ZsW8skfvoRfgeerBF	Mine	Taylor Swift	Speak Now	63	2010-10-25	False	0.0296	0.624	0.757	7	-2.940	1	0.00265	0.000002
1057	24DefNCFiWTP8OjYWiXuYe	Speak Now	Taylor Swift	Speak Now	61	2010-10-25	False	0.0304	0.709	0.599	7	-3.734	1	0.09500	0.000000
48	1fo2ctLqj3zBhRQKOXprol	Style	Taylor Swift	1989 (Deluxe Edition)	62	2014-10-27	False	0.0402	0.588	0.791	7	-5.595	1	0.00245	0.002580
42	3bIxTsfeNMO7Nt2J3EUKrA	22	Taylor Swift	Red (Deluxe Edition)	55	2012-10-22	False	0.0378	0.658	0.729	7	-6.561	1	0.00215	0.001300
1070	3rnI1UCyGJvUTVvT97VQr5	Tell Me Why	Taylor Swift	Fearless	42	2008-11-11	False	0.0380	0.604	0.855	7	-3.097	1	0.05710	0.000004
649	32mVHdy0bi1XKgr0ajsBlG	Picture To Burn	Taylor Swift	Taylor Swift	64	2006-10-24	False	0.0323	0.658	0.877	7	-2.098	1	0.17300	0.000000
1103	5AsmAjBSSQjRPWgAYIp8tm	The Moment I Knew	Taylor Swift	Red (Deluxe Edition)	40	2012-10-22	False	0.0312	0.620	0.506	7	-7.327	1	0.18700	0.000015
574	4t0OI7XrODjSkAu3bTPmWj	Fifteen	Taylor Swift	Fearless	54	2008-11-11	False	0.0266	0.556	0.651	7	-4.396	1	0.06700	0.000000
11	7AEAGTc8cReDqcbPoY9gwo	We Are Never Ever Getting Back Together	Taylor Swift	Red	57	2012-10-22	False	0.0916	0.628	0.676	7	-5.911	1	0.00957	0.000027

anceability	energy	key	loudness	mode	acousticness	instrumentalness	liveness	valence	tempo	duration_ms	time_signature	genres	artist_pop	genres_list	subjectivity	polarity	sim
0.582	0.765	7	-3.648	1	0.02680	0.000002	0.1030	0.559	131.982	275960	4	pop	100	[pop]	low	Neutral	0.941643
0.624	0.757	7	-2.940	1	0.00265	0.000002	0.1890	0.658	121.070	230707	4	pop	100	[pop]	low	Neutral	0.941419
0.709	0.599	7	-3.734	1	0.09500	0.000000	0.0973	0.735	118.975	240760	4	pop	100	[pop]	low	Neutral	0.941222
0.588	0.791	7	-5.595	1	0.00245	0.002580	0.1180	0.487	94.933	231000	4	pop	100	[pop]	low	Neutral	0.941100
0.658	0.729	7	-6.561	1	0.00215	0.001300	0.0752	0.668	104.007	230133	4	pop	100	[pop]	low	Neutral	0.940970
0.604	0.855	7	-3.097	1	0.05710	0.000004	0.3430	0.510	99.960	200547	4	pop	100	[pop]	low	Neutral	0.940774
0.658	0.877	7	-2.098	1	0.17300	0.000000	0.0962	0.821	105.586	173067	4	pop	100	[pop]	low	Neutral	0.940654
0.620	0.506	7	-7.327	1	0.18700	0.000015	0.1020	0.275	126.015	285560	4	pop	100	[pop]	low	Neutral	0.940586
0.556	0.651	7	-4.396	1	0.06700	0.000000	0.1450	0.203	95.485	294333	4	pop	100	[pop]	low	Neutral	0.940528
0.628	0.676	7	-5.911	1	0.00957	0.000027	0.1020	0.750	85.984	191880	4	pop	100	[pop]	low	Neutral	0.940279

Hình 41. Điểm tương đồng giữa các bài hát theo độ đo Cosine

Bằng cách sử dụng độ đo cosine, chúng tôi đã trích xuất ra được danh sách phát gồm 10 bài hát phù hợp nhất cho người dùng. Có thể thấy với cosine similarity thì khả năng user sẽ được hệ thống đề xuất những bài nhạc thể loại pop cao hơn những mô hình như Linear Regression, Random Forest Regression, Keras Neural Network

3.3. Phân Tích và Đánh Giá

Đối với phương pháp phân cụm K-Means, chúng tôi đã tiến hành xây dựng 1 danh sách phát gồm 10 hát được đề xuất sau khi tiến hành huấn luyện dữ liệu với danh sách nhạc đưa vào gồm 20 bài hát. Tuy nhiên chỉ số Silhouette của mô hình này khá thấp (<0.3) nên chúng tôi đang cân nhắc đến với không tin tưởng mô hình này để xây dựng hệ thuật đề xuất danh sách nhạc từ Spotify.

Có thể thấy với cosine_similarity thì khả năng user sẽ được hệ thống đề xuất những bài nhạc thể loại pop cao hơn những mô hình như Linear Regression, Random Forest Regression, Keras Neural Network. Với top 10 bài hát đề xuất ở cosine_similarity ta thấy chỉ số rất cao (đều lớn hơn 0.9) trong khi những mô hình như Linear Regression, Random Forest Regression, Keras Neural Network lại cho ra những đề xuất không hẳn là về thể loại pop nó bao gồm nhiều thể loại khác trong đó như dance_pop, new_wave_pop,... có thể thấy ở Hệ thống đề xuất theo sở thích người nghe và ratings thì cosine_similarity đang làm tốt hơn các mô hình hồi quy và neural network.

Còn về Hệ thống dự đoán xem liệu user có thích hay không thích các bài hát dựa trên sở thích của user các mô hình phân lớp đều đưa ra độ chính xác khá cao (cao nhất là SVM và Gradient Boosting với khoảng 80% accuracy). Cho thấy khả năng dự đoán khá tốt của mô hình phân lớp. Song với đó chúng tôi cũng cần phải thu thập thêm nhiều dữ liệu để có thể xây dựng mô hình với khả năng dự đoán chính xác hơn nữa.

CHƯƠNG 4. KẾT LUẬN

4.1. Các Kết Quả Đạt Được

Sau khi thực hiện nghiên cứu này, về bước đầu chúng tôi đã dần thành công với các hệ thống đề xuất dựa trên cosine_similarity với chỉ số cosine similarity trong top 10 đều lớn hơn 0,9 và cũng đang khá thành công với Hệ thống đề xuất bằng Random Forest Regressor khi cả 4 trong top 5 bài hát với predict ratings cao đều mang thể loại pop (thể loại mà người dùng thích nghe).

Về hệ thống dự đoán các bài hát xem người dùng có thích các bài hát đó hay không thì chúng tôi đã thu được độ chính xác của mô hình khá cao (đạt tới gần 80%).

Với việc xây dựng hệ thống đề xuất một trong những hệ thống tiên tiến và hiệu quả nhất trong ngành truyền phát nhạc. Và dưới đây là một số điều mà hệ thống đề xuất này có thể thực hiện:

1. Khám phá âm nhạc mới: Hệ thống đề xuất giúp người dùng khám phá các bài hát và nghệ sĩ mới dựa trên lịch sử nghe, các bài hát và danh sách phát đã lưu cũng như các thể loại ưa thích của họ.
2. Tạo danh sách phát được cá nhân hóa: dựa vào hệ thống đề xuất này chúng tôi cũng có thể tạo danh sách phát được cá nhân hóa cho người dùng, chẳng hạn như Bản phối hàng ngày, Khám phá hàng tuần và Radar phát hành, được cập nhật thường xuyên dựa trên hành vi nghe của người dùng.
3. Cải thiện đài phát thanh: Tính năng đài phát thanh trong Spotify được cung cấp bởi hệ thống đề xuất, sử dụng thuật toán học máy để xác định các bài hát tương tự và tạo danh sách phát các bài hát có khả năng thu hút người dùng.
4. Tìm các bài hát và nghệ sĩ tương tự: chúng tôi cũng có thể xác định các bài hát và nghệ sĩ tương tự dựa trên sở thích và lịch sử nghe của người dùng, cho phép họ khám phá âm nhạc mới phù hợp với sở thích của mình.
5. Nâng cao kết quả tìm kiếm: hệ thống cũng có thể nâng cao kết quả tìm kiếm bằng cách đề xuất các bài hát, nghệ sĩ và danh sách phát có liên quan dựa trên truy vấn của người dùng.

4.2. Những Hạn Chế và Hướng Phát Triển

4.2.1. Những hạn chế

Chúng tôi đề xuất hướng phát triển Hệ thống đề xuất của Spotify là một trong những hệ thống đề xuất phổ biến và thành công nhất trong ngành truyền phát nhạc. Tuy

nhiên, giống như tất cả các hệ thống khuyến nghị, nó có những hạn chế và lĩnh vực cần phát triển. Dưới đây là một số hạn chế và hướng phát triển tiềm năng cho hệ thống khuyến nghị của Spotify:

Đối với phương pháp đề xuất bài hát dựa trên danh sách phát có đánh giá, khi người dùng hoặc mục mới vào hệ thống sẽ không có xếp hạng nào để đề xuất. Nếu người dùng có sở thích khác thường thì xếp hạng sẽ ít và người dùng không có nhiều những danh sách nhạc khác để so sánh, nên các đề xuất sẽ kém. Việc cá nhân hóa sẽ bị suy yếu do các bài hát nổi tiếng dù sao cũng sẽ được đề xuất. Ngoài ra, với việc thiếu xếp hạng hoặc ngay từ đầu không có một lượng lớn dữ liệu, các dự đoán sẽ rất kém. Cuối cùng, một hệ thống giới thiệu hoàn hảo không nên đòi hỏi quá nhiều nỗ lực của con người, bởi vì không phải lúc nào người dùng cũng sẵn sàng xếp hạng. Dữ liệu không phải lúc nào cũng mang tính đại diện do xếp hạng ngày càng tăng đối với những người xếp hạng.

Dữ liệu hạn chế: Chất lượng của các đề xuất phụ thuộc rất nhiều vào số lượng và chất lượng của dữ liệu có sẵn. Mặc dù Spotify có quyền truy cập vào một lượng dữ liệu khổng lồ, bao gồm hành vi của người dùng, lịch sử nghe và siêu dữ liệu âm nhạc, nhưng vẫn có giới hạn về lượng dữ liệu mà hệ thống có thể thu thập. Ngoài ra, một số dữ liệu người dùng có thể bị hạn chế do lo ngại về quyền riêng tư.

Thiếu sự đa dạng: Hệ thống đề xuất của Spotify được biết đến với việc đề xuất các nghệ sĩ và bài hát tương tự với những gì người dùng đã nghe trước đây. Điều này có thể dẫn đến sự thiếu đa dạng trong các đề xuất, điều này có thể hạn chế đối với những người dùng muốn khám phá các thể loại và phong cách khác nhau.

Thiếu bối cảnh: Hệ thống đề xuất của Spotify được thiết kế để đề xuất âm nhạc dựa trên hành vi và lịch sử nghe của người dùng, nhưng không phải lúc nào hệ thống cũng tính đến bối cảnh mà người dùng đang nghe nhạc. Ví dụ: nếu người dùng đang nghe nhạc trong khi tập thể dục, hệ thống có thể đề xuất nhạc có năng lượng cao, ngay cả khi người dùng thường thích nhạc êm dịu hơn.

Cá nhân hóa hạn chế: Mặc dù hệ thống đề xuất của Spotify cá nhân hóa các đề xuất ở một mức độ nào đó, nhưng vẫn còn chỗ cần cải thiện trong lĩnh vực này. Ví dụ: hệ thống có thể tính đến các tùy chọn cụ thể hơn của người dùng, chẳng hạn như nhạc cụ yêu thích hoặc phong cách giọng hát.

4.2.2. Hướng phát triển

Để giải quyết những hạn chế này và cải thiện hiệu suất tổng thể của hệ thống đề xuất, Spotify có thể xem xét các hướng phát triển sau:

Mở rộng nguồn dữ liệu: Ngoài lịch sử nghe và hành vi của người dùng, Spotify có thể kết hợp các nguồn dữ liệu khác, chẳng hạn như hoạt động trên mạng xã hội hoặc danh sách phát do người dùng tạo, để cải thiện chất lượng của các đề xuất.

Tăng tính đa dạng: Spotify có thể khám phá các cách đề xuất các nghệ sĩ và bài hát đa dạng hơn, chẳng hạn như kết hợp nhiều thể loại hơn hoặc kết hợp các đề xuất từ các chuyên gia trong các lĩnh vực âm nhạc khác nhau.

Kết hợp bối cảnh: Để cải thiện mức độ phù hợp của các đề xuất, Spotify có thể kết hợp dữ liệu theo ngữ cảnh, chẳng hạn như thời gian trong ngày, địa điểm và hoạt động hiện tại, vào thuật toán đề xuất.

Cải thiện khả năng cá nhân hóa: Spotify có thể kết hợp các tùy chọn chi tiết hơn của người dùng, chẳng hạn như nhạc cụ hoặc phong cách giọng hát yêu thích, vào thuật toán đề xuất để cung cấp các đề xuất được cá nhân hóa hơn. Ngoài ra, hệ thống có thể cho phép người dùng cung cấp phản hồi về các đề xuất để cá nhân hóa hơn nữa trải nghiệm của họ.

TÀI LIỆU THAM KHẢO

[1] Cao Tuan, “Big Data va mo hinh 5V”.

[2] Deyan Georgiev, “21+ Spotify Revenue and User Statistics for 2023,” 2023.

[3] J. K. J. S. N. W. Buse Dikici, “Big Data: Spotify,” 2016.

[4] R. Bell, “Matrix factorization techniques for recommender systems.,” *Computer Journal,* pp. 30-37, 2009.

[5] L. R. &. B. S. Francesco Ricci, Introduction to Recommender Systems Handbook, 2010.

[6] M. v. S. Robin van Meteren, “Using Content-Based Filtering for Recommendation”.

[7] J. Singh, “Collaborative Filtering based Hybrid Music Recommendation System,” *2020 3rd International Conference on Intelligent Sustainable Systems (ICISS),* pp. 186-190, 2020.

[8] N. Đ. Quý, “Quy's blog,” 2021. [Trực tuyến]. Available: https://ndquy.github.io/posts/thuat-toan-phan-cum-kmeans/.

[9] “trituenhantao.io,” 2019. [Trực tuyến]. Available: https://trituenhantao.io/kien-thuc/decision-tree/.

[10] H. C. Trung, “VIBLO,” 20 08 2020. [Trực tuyến]. Available: https://viblo.asia/p/gioi-thieu-ve-support-vector-machine-svm-6J3ZgPVElmB.

[11] “Machine Learning cơ bản,” 04 09 2017. [Trực tuyến]. Available: https://machinelearningcoban.com/2017/04/09/smv/.

[12] “Machine Learning cơ bản,” 01 08 2017. [Trực tuyến]. Available: https://machinelearningcoban.com/2017/01/08/knn/.

[13] L. Breiman, “Classification and Regression Trees,” *Routledge,* 2017.

[14] I. C. YEH, “36 MATERIALS SCIENCE; COMPRESSION STRENGTH; CONCRETES; MATHEMATICAL MODELS; NEURAL NETWORKS;

PREDICTION EQUATIONS; VALIDATION," *Cement and Concrete Research*, tập 28, số 12, 1998.

[15] T. D. Thang, "VIBLO," 2020. [Trực tuyến]. Available: https://viblo.asia/p/lam-quen-voi-keras-gGJ59mxJ5X2.

[16] N. Đ. Quý, "Quy's blog," 2021. [Trực tuyến]. Available: https://ndquy.github.io/posts/gioi-thieu-machine-learning/.

[17] Y. R. B. a. C. V. Koren, ""Matrix factorization techniques for recommender systems.","*Computer*, pp. 30-37, 2009.