

**ĐẠI HỌC DUY TÂN**

**KHOA CÔNG NGHỆ THÔNG TIN**

--------------🙖🙐✰🙖🙐-------------



**MÔN: ĐỒ ÁN CDIO – CS347**

**Tên đề tài:**

**XÂY DỰNG WEBSITE NGHE NHẠC TRỰC TUYẾN TÍCH HỢP HỆ THỐNG ĐỀ XUẤT NHẠC**



**GVHD: Ts. LÊ THANH LONG**

**Tên nhóm: Nhóm 3**

* **Ngô Tấn Vĩnh (28211445639)**
* **Trần Văn Vỹ (27211248535)**
* **Lê Trần Ánh Nhung (28201302124)**
* **Cao Minh (27217801951)**
* **Nguyễn Nhật Minh (27217801937)**

**Đà nẵng, 13/10/2024**

**Mục lục**

[PHẦN MỞ ĐẦU 1](#_heading=h.gjdgxs)

[CHƯƠNG I: Ý TƯỞNG DỰ ÁN 5](#_heading=h.1fob9te)

[1.1. Đề xướng ý tưởng sản phẩm (Cá nhân) 5](#_heading=h.3znysh7)

[1.1.1. Thành viên 1 5](#_heading=h.2et92p0)

[1.1.2. Thành viên 2 6](#_heading=h.tyjcwt)

[1.1.3. Thành viên 3 7](#_heading=h.3dy6vkm)

[1.1.4. Thành viên 4 9](#_heading=h.1t3h5sf)

[1.1.5. Thành viên 5 11](#_heading=h.4d34og8)

[1.2. Ý tưởng đề xuất 13](#_heading=h.2s8eyo1)

[CHƯƠNG II: CƠ SỞ LÝ THUYẾT 14](#_heading=h.17dp8vu)

[2.1. Machine Learning 14](#_heading=h.3rdcrjn)

[2.2. Deep Learning 15](#_heading=h.26in1rg)

[2.3. Ngôn ngữ và công cụ lập trình 16](#_heading=h.lnxbz9)

[2.4. Giới thiệu thư viện 17](#_heading=h.35nkun2)

[KẾT LUẬN CHƯƠNG II 18](#_heading=h.1ksv4uv)

[CHƯƠNG III: GIỚI THIỆU THUẬT TOÁN 19](#_heading=h.44sinio)

[3.1. Collaborative Filtering (Lọc cộng tác) 19](#_heading=h.2jxsxqh)

[3.1.1. User-based Collaborative Filtering (Lọc cộng tác dựa trên người dùng) 19](#_heading=h.z337ya)

[3.1.2. Item-based Collaborative Filtering (Lọc cộng tác dựa trên bài hát) 19](#_heading=h.3j2qqm3)

[3.2. K-nearest neighbor 20](#_heading=h.1y810tw)

[3.3. Convolutional Neural Networks (CNN) 21](#_heading=)

[KẾT LUẬN CHƯƠNG III 23](#_heading=h.4i7ojhp)

[CHƯƠNG IV: XÂY DỰNG CHƯƠNG TRÌNH 24](#_heading=h.2xcytpi)

[4.1. Dữ liệu(Dataset) 24](#_heading=h.1ci93xb)

[4.2. Xây dựng mô hình 26](#_heading=h.3whwml4)

[4.2.1. Cài đặt thư viện: 26](#_heading=h.2bn6wsx)

[4.2.2. Xử lý dữ liệu 26](#_heading=h.qsh70q)

[4.2.3. Xác thực và truy xuất dữ liệu từ Spotify thông qua API 31](#_heading=h.3as4poj)

[4.2.4. Tạo Playlist Vector 34](#_heading=h.1pxezwc)

[4.2.5. Model CNN VGG16 35](#_heading=h.49x2ik5)

[4.2.6 Mô hình CNN trong hệ thống gợi ý nhạc: 36](#_heading=h.u4t7mpfyt3mj)

[4.3. Demo 37](#_heading=h.2p2csry)

[PHẦN KẾT LUẬN 43](#_heading=h.147n2zr)

[1. Kết quả đạt được 43](#_heading=h.3o7alnk)

[2. Hạn chế của đề tài 43](#_heading=h.23ckvvd)

[3. Hướng phát triển 43](#_heading=h.ihv636)

[TÀI LIỆU THAM KHẢO 44](#_heading=h.1hmsyys)

# PHẦN MỞ ĐẦU

**1. Lý do chọn đề tài**

Trong bối cảnh thế giới đang không ngừng chuyển mình với tốc độ chóng mặt của công nghệ số, ngành công nghiệp âm nhạc cũng đang trải qua những biến chuyển sâu sắc để thích nghi và phát triển. Nghe nhạc trực tuyến, với khả năng kết nối người yêu nhạc với kho tàng âm nhạc khổng lồ, đa dạng thể loại, đã trở thành một phần không thể thiếu trong đời sống tinh thần của con người hiện đại. Tuy nhiên, giữa biển nhạc mênh mông ấy, việc tìm kiếm và tận hưởng những giai điệu thực sự chạm đến trái tim, phù hợp với sở thích và tâm trạng của mỗi cá nhân lại trở thành một bài toán đặt ra cho các nhà phát triển nền tảng âm nhạc.

Các nền tảng nghe nhạc trực tuyến hiện nay, dù sở hữu kho nhạc đồ sộ, vẫn còn nhiều hạn chế trong việc cá nhân hóa trải nghiệm người dùng. Việc gợi ý bài hát dựa trên thuật toán đơn giản, thiếu đi sự thấu hiểu sâu sắc về gu âm nhạc, khiến người nghe dễ rơi vào vòng lặp nhàm chán, bỏ lỡ cơ hội khám phá những bản nhạc mới lạ, phù hợp với mình. Chính trong bối cảnh đó, trí tuệ nhân tạo (AI), với khả năng xử lý dữ liệu và học hỏi phi thường, nổi lên như một giải pháp đột phá, mở ra kỷ nguyên mới cho trải nghiệm âm nhạc cá nhân hóa.

Sức mạnh của AI, đặc biệt là thuật toán lọc cộng tác, nằm ở khả năng phân tích dữ liệu khổng lồ về thói quen, gu âm nhạc của cộng đồng người dùng để từ đó đề xuất những bài hát phù hợp một cách chính xác và tinh tế cho từng cá nhân. Website nghe nhạc trực tiếp tích hợp AI không chỉ dừng lại ở việc mang đến trải nghiệm âm nhạc cá nhân hóa cho người dùng, mà còn kiến tạo một hệ sinh thái âm nhạc bền vững, nơi các giá trị nghệ thuật và lợi ích kinh tế cùng song hành. Bài hát của các nghệ sĩ mới, thay vì bị lãng quên giữa biển nhạc mênh mông, có thể tiếp cận đúng đối tượng khán giả tiềm năng. Dữ liệu người dùng được phân tích để thấu hiểu thị hiếu âm nhạc, từ đó tối ưu hóa chiến lược quảng bá, phát hành sản phẩm âm nhạc và gia tăng doanh thu cho cả nghệ sĩ và nền tảng.

Chính vì lẽ đó, chúng tôi lựa chọn đề tài "Xây dựng website nghe nhạc trực tuyến tích hợp hệ thống đề xuất nhạc" với mong muốn góp phần kiến tạo một không gian thưởng thức âm nhạc thông minh, nơi mỗi người dùng đều có thể tìm thấy "chất riêng" của chính mình trong thế giới âm nhạc vô tận, đồng thời góp phần thúc đẩy sự phát triển bền vững cho chính ngành công nghiệp âm nhạc.

**2. Mục đích và ý nghĩa của đề tài**

**2.1. Mục đích của đề tài**

* Nắm vững kiến thức nền tảng về lập trình web, cơ sở dữ liệu và quy trình xây dựng website hoàn chỉnh.
* Hiểu rõ cách ứng dụng AI, vận dụng thuật toán học máy để phát triển hệ thống đề xuất bài hát.
* Vận dụng kiến thức và AI để cá nhân hóa trải nghiệm người dùng, nâng cao hiệu quả sử dụng website.
* Biết cách phân tích, đánh giá hiệu quả hệ thống AI dựa trên dữ liệu thực tế, rút ra bài học kinh nghiệm.
* Nâng cao nhận thức về vai trò của AI trong việc phát triển hệ sinh thái âm nhạc bền vững, tạo tiền đề cho các dự án tương lai.

**2.2. Ý nghĩa của đề tài**

**Về mặt lý luận của đề tài**

Góp phần phát triển lĩnh vực trí tuệ nhân tạo và học máy: Đề tài này sẽ sử dụng các phương pháp và thuật toán tiên tiến trong lĩnh vực trí tuệ nhân tạo và học máy để xây dựng hệ thống đề xuất. Qua đó, nó góp phần bổ sung kiến thức và ứng dụng của AI trong đời sống.

Nâng cao hiểu biết về phân tích dữ liệu người dùng: Việc nghiên cứu các mô hình và phương pháp phân tích dữ liệu người dùng sẽ giúp làm sáng tỏ cách thức mà các hệ thống thông minh có thể hiểu và dự đoán hành vi của con người.

**Về mặt thực tiễn của đề tài**

Cải thiện trải nghiệm người dùng: Hệ thống đề xuất nhạc sẽ giúp người dùng tìm thấy các bài hát và nghệ sĩ phù hợp với sở thích cá nhân, từ đó tạo ra một trải nghiệm nghe nhạc thú vị và độc đáo.

Hỗ trợ nghệ sĩ và nhà sản xuất âm nhạc: Hệ thống có thể giúp các nghệ sĩ và nhà sản xuất tiếp cận đến khán giả mục tiêu một cách hiệu quả hơn, đặc biệt là những tài năng mới và chưa được biết đến rộng rãi.

Tăng cường sự cạnh tranh của nền tảng âm nhạc: Sở hữu một hệ thống đề xuất mạnh mẽ là lợi thế cạnh tranh lớn, giúp nền tảng thu hút và giữ chân người dùng, từ đó phát triển bền vững trong thị trường âm nhạc số.

**3. Đối tượng nghiên cứu**

Đối tượng nghiên cứu của đề tài này bao gồm các thuật toán đề xuất nhạc, dữ liệu hành vi và sở thích nghe nhạc của người dùng, và các yếu tố ảnh hưởng đến sự hài lòng và gắn bó của người dùng với nền tảng âm nhạc.

**4. Phạm vi nghiên cứu**

Phạm vi nghiên cứu của đề tài bao gồm việc thu thập và phân tích dữ liệu từ các nền tảng âm nhạc trực tuyến, phát triển và kiểm thử các mô hình đề xuất nhạc, và đánh giá hiệu quả của hệ thống đề xuất thông qua các chỉ số như độ chính xác, tính cá nhân hóa, và mức độ hài lòng của người dùng.

**5. Phương pháp nghiên cứu**

- Tìm hiểu các tài liệu liên quan (ngôn ngữ Python,CNN,LSTM …)

- Phân tích và lựa chọn giải pháp.  
- Tổng hợp các tài liệu liên quan đến xây dựng và phát triển phần mềm.

- Góp ý của thầy giáo hướng dẫn.

**6. Đối tượng sử dụng**

Đối tượng sử dụng của hệ thống đề xuất nhạc bao gồm:

1. Người dùng cá nhân: Đây là những người dùng trực tiếp các nền tảng âm nhạc để nghe nhạc hàng ngày. Họ mong muốn có trải nghiệm nghe nhạc cá nhân hóa, được gợi ý những bài hát và nghệ sĩ phù hợp với sở thích và thói quen nghe nhạc của mình.
2. Nền tảng âm nhạc trực tuyến: Các công ty cung cấp dịch vụ nghe nhạc trực tuyến (như Spotify, Apple Music, YouTube Music) có thể tích hợp hệ thống đề xuất nhạc vào nền tảng của họ để cải thiện trải nghiệm người dùng, giữ chân người dùng và thu hút người dùng mới.
3. Nghệ sĩ và nhà sản xuất âm nhạc: Hệ thống đề xuất nhạc giúp các nghệ sĩ và nhà sản xuất âm nhạc đưa tác phẩm của họ đến với khán giả mục tiêu một cách hiệu quả hơn. Điều này đặc biệt hữu ích cho các nghệ sĩ mới hoặc chưa được biết đến rộng rãi.
4. Nhà nghiên cứu và phát triển trong lĩnh vực AI và dữ liệu: Các nhà nghiên cứu và phát triển có thể sử dụng hệ thống này như một trường hợp nghiên cứu để khám phá và cải tiến các thuật toán trí tuệ nhân tạo và học máy trong lĩnh vực đề xuất cá nhân hóa.
5. Các nhà quảng cáo và tiếp thị: Hệ thống đề xuất nhạc cung cấp dữ liệu và phân tích về sở thích âm nhạc của người dùng, giúp các nhà quảng cáo và tiếp thị đưa ra chiến lược quảng cáo nhắm đến đối tượng cụ thể một cách chính xác hơn.

# 

# CHƯƠNG I: Ý TƯỞNG DỰ ÁN

## 1.1. Đề xướng ý tưởng sản phẩm (Cá nhân)

### 1.1.1. Thành viên 1

- Họ & Tên (sinh viên): Ngô Tấn Vĩnh

- Chuyên ngành: Data Science

- Tên ý tưởng (sản phẩm): Web nghe nhạc trực tuyến

***1.1.1.1. Mô tả ý tưởng sản phẩm***

Tạo 1 trang web nhạc: Khi like 1 bài bất kì -> bài hát được post lên model -> model sẽ đưa xuống các bài có chung 1 cơ sở nào đó với bài vừa like và hiển thị vào 1 mục nào đó trên web.

***1.1.1.2. Đánh giá cơ sở kiến thức của sinh viên***

* Có thể vận dụng các ngôn ngữ lập trình như C++/C, Java, JavaScript, HTML, CSS, Python,...
* Tạo được 1 web đơn giản có các chức năng cần thiết.

***1.1.1.3. Đánh giá khả năng phát triển của sản phẩm***

| Ước tính số lượng người sẽ hưởng lợi từ sản phẩm của bạn | 100000 |
| --- | --- |
| Ước tính số lượng người sẽ bỏ tiền ra mua sản phẩm của bạn | 100 |
| Liệt kê các loại đối tượng sẽ được hưởng lợi từ ý tưởng sản phẩm của bạn (ít nhất là 1, nhiều nhất là 5): | Khách hàng có nhu cầu mua sắm (các phụ huynh muốn quản lý cho con cái). |

|  | Trung bình | Khá | Cao | Rất cao |
| --- | --- | --- | --- | --- |
| Đánh giá khả năng các nhà đầu tư sẽ bỏ tiền ra cho ý tưởng sản phẩm của bạn: |  |  | X |  |

|  | Trung bình | Khá | Cao | Rất cao |
| --- | --- | --- | --- | --- |
| Ước tính độ khó để phát triển thành công sản phẩm của bạn: |  | X |  |  |

| Ước lượng thời gian cần thiết để phát triển sản phẩm của bạn: | 4 tháng |
| --- | --- |

### 1.1.2. Thành viên 2

- Họ & Tên (sinh viên): Trần Văn Vỹ

- Chuyên ngành: Khoa Học Máy Tính

- Tên ý tưởng (sản phẩm): Ứng dụng AI nhận diện biển số xe

***1.1.2.1. Mô tả ý tưởng sản phẩm***

Ứng dụng này sử dụng công nghệ trí tuệ nhân tạo (AI) và học máy (machine learning) để nhận diện và phân tích biển số xe trong thời gian thực. Nó có thể được sử dụng cho nhiều mục đích, bao gồm quản lý giao thông, an ninh, và kiểm soát bãi đỗ xe.

***1.1.2.2. Đánh giá cơ sở kiến thức của sinh viên***

Kiến thức Cơ bản về AI: Các khái niệm cơ bản về trí tuệ nhân tạo và học máy. Hiểu biết về các thuật toán nhận diện hình ảnh (như CNN - Convolutional Neural Networks).

Quy trình Nhận diện Biển số:

* Các bước trong quy trình nhận diện biển số xe: thu thập dữ liệu, tiền xử lý hình ảnh, nhận diện ký tự.
* Các công nghệ và phần mềm phổ biến được sử dụng trong ứng dụng này.

Ứng dụng Thực tiễn: Các lĩnh vực ứng dụng của nhận diện biển số xe (quản lý giao thông, an ninh, bãi đỗ xe). Những lợi ích và thách thức khi triển khai ứng dụng này.

***1.1.2.3. Đánh giá khả năng phát triển của sản phẩm***

| Ước tính số lượng người sẽ hưởng lợi từ sản phẩm của bạn. | 1000 |
| --- | --- |
| Ước tính số lượng người sẽ bỏ tiền ra mua sản phẩm của bạn | 20 |
| Liệt kê các loại đối tượng sẽ được hưởng lợi từ ý tưởng sản phẩm của bạn (ít nhất là 1,nhiều nhất là 5) | Những người có nhu cầu đăng hình ảnh liên tục |

|  | Trung bình | Khá | Cao | Rất cao |
| --- | --- | --- | --- | --- |
| Đánh giá khả năng các nhà đầu tư sẽ bỏ tiền ra cho ý tưởng sản phẩm của bạn: |  | x |  |  |

|  | Trung bình | Khá | Cao | Rất cao |
| --- | --- | --- | --- | --- |
| Ước tính độ khó để phát triển thành công sản phẩm của bạn: |  | x |  |  |

| Ước lượng thời gian cần thiết để phát triển sản phẩm của bạn: | 1 tháng |
| --- | --- |

### 1.1.3. Thành viên 3

- Họ & Tên (sinh viên): Lê Trần Ánh Nhung

- Chuyên ngành: Khoa Học Máy Tính

- Tên ý tưởng (sản phẩm): Ứng dụng AI phân loại ảnh khối u não

***1.1.3.1. Mô tả ý tưởng sản phẩm***

Ứng dụng AI phân loại khối u não dựa trên thuật toán Convolutional Neural Networks (CNN) được phát triển nhằm giải quyết những thách thức trong việc chẩn đoán các loại khối u dựa trên hình ảnh MRI. Quá trình chẩn đoán thủ công hiện nay có thể tốn thời gian và dễ gặp sai sót do khối lượng lớn hình ảnh và tính phức tạp của các đặc điểm bệnh lý. Ứng dụng này không chỉ hỗ trợ các bác sĩ trong việc phân loại chính xác các loại khối u như u lành tính hay u ác tính, mà còn giúp giảm tải công việc, tăng hiệu suất chẩn đoán và đảm bảo bệnh nhân được điều trị kịp thời hơn. Đây là một bước tiến quan trọng trong việc áp dụng trí tuệ nhân tạo vào y học, góp phần cải thiện chất lượng chăm sóc sức khỏe.

Convolutional Neural Networks (CNN) là công nghệ cốt lõi của ứng dụng, với khả năng vượt trội trong việc nhận diện và phân loại hình ảnh. CNN sử dụng các lớp chập để trích xuất và phân tích tự động các đặc trưng trong hình ảnh MRI, giúp phát hiện các mẫu bệnh lý mà đôi khi mắt thường không nhận ra được. Quy trình phát triển ứng dụng bao gồm thu thập và xử lý dữ liệu MRI, sau đó huấn luyện mô hình CNN trên các tập dữ liệu đã gắn nhãn để nhận diện các đặc điểm của khối u. Khi được tích hợp vào hệ thống, ứng dụng sẽ cho phép người dùng tải lên hình ảnh MRI và nhận kết quả phân loại nhanh chóng, từ đó hỗ trợ bác sĩ trong việc ra quyết định chẩn đoán và điều trị.

Tuy nhiên, hạn chế của ứng dụng này nằm ở yêu cầu độ chính xác cực cao khi áp dụng vào thực tế lâm sàng. Trong lĩnh vực y tế, một sai sót nhỏ cũng có thể dẫn đến những hậu quả nghiêm trọng, do đó việc triển khai AI trong phân loại khối u cần được thử nghiệm và kiểm chứng kỹ lưỡng. Việc ứng dụng CNN vào chẩn đoán khối u não hiện vẫn đối mặt với thách thức trong việc đạt được độ tin cậy tuyệt đối để có thể áp dụng rộng rãi trong thực tế. Điều này đòi hỏi sự đầu tư lớn về dữ liệu và thời gian để đảm bảo rằng hệ thống có thể đưa ra các quyết định chính xác và an toàn cho bệnh nhân.

***1.1.3.2. Đánh giá khả năng phát triển của sản phẩm***

| Ước tính số lượng người sẽ hưởng lợi từ sản phẩm của bạn | 10000 |
| --- | --- |
| Ước tính số lượng người sẽ bỏ tiền ra mua sản phẩm của bạn | 100 |
| Liệt kê các loại đối tượng sẽ được hưởng lợi từ ý tưởng sản phẩm của bạn (ít nhất là 1,nhiều nhất là 5): | * Bác sĩ * Bệnh nhân mắc bệnh u não |

|  | Trung bình | Khá | Cao | Rất cao |
| --- | --- | --- | --- | --- |
| Đánh giá khả năng các nhà đầu tư sẽ bỏ tiền ra cho ý tưởng sản phẩm của bạn: |  |  | X |  |

|  | Trung bình | Khá | Cao | Rất cao |
| --- | --- | --- | --- | --- |
| Ước tính độ khó để phát triển thành công sản phẩm của bạn: |  |  |  | X |

| Ước lượng thời gian cần thiết để phát triển sản phẩm của bạn: | 6 tháng |
| --- | --- |

### 1.1.4. Thành viên 4

- Họ & Tên (sinh viên): Cao Minh

- Chuyên ngành: Khoa học máy tính

- Tên ý tưởng (sản phẩm): Trợ lý ngôn ngữ thông minh cho người câm điếc

***1.1.4.1. Mô tả ý tưởng sản phẩm***

Sản phẩm là một ứng dụng trí tuệ nhân tạo giúp người câm điếc giao tiếp dễ dàng với người bình thường thông qua việc dịch ngôn ngữ ký hiệu thành văn bản hoặc giọng nói. Ứng dụng có thể nhận diện cử chỉ tay và gương mặt của người dùng, sau đó chuyển đổi thành câu văn hoặc âm thanh. Ngoài ra, ứng dụng còn cung cấp các bài học tự luyện ngôn ngữ ký hiệu cho người mới học.

***1.1.4.2. Đánh giá cơ sở kiến thức của sinh viên***

Có nền tảng vững chắc về trí tuệ nhân tạo, xử lý ngôn ngữ tự nhiên và phát triển ứng dụng di động. Kỹ năng lập trình Python, TensorFlow, và thiết kế giao diện người dùng giúp đủ khả năng hiện thực hóa sản phẩm này.

***1.1.4.3. Đánh giá khả năng phát triển của sản phẩm***

| Ước tính số lượng người sẽ hưởng lợi từ sản phẩm của bạn | 15000 |
| --- | --- |
| Ước tính số lượng người sẽ bỏ tiền ra mua sản phẩm của bạn | 100 |
| Liệt kê các loại đối tượng sẽ được hưởng lợi từ ý tưởng sản phẩm của bạn (ít nhất là 1,nhiều nhất là 5): | * Người câm điếc * Người già gặp khó khăn trong việc nói * Bệnh nhân sau phẫu thuật ảnh hưởng đến giọng nói * Người học ngôn ngữ ký hiệu * Nhân viên chăm sóc người khuyết tật |

|  | Trung bình | Khá | Cao | Rất cao |
| --- | --- | --- | --- | --- |
| Đánh giá khả năng các nhà đầu tư sẽ bỏ tiền ra cho ý tưởng sản phẩm của bạn: |  | X |  |  |

|  | Trung bình | Khá | Cao | Rất cao |
| --- | --- | --- | --- | --- |
| Ước tính độ khó để phát triển thành công sản phẩm của bạn: |  |  | X |  |

| Ước lượng thời gian cần thiết để phát triển sản phẩm của bạn: | 9 tháng |
| --- | --- |

### 1.1.5. Thành viên 5

- Họ & Tên (sinh viên): Nguyễn Nhật Minh

- Chuyên ngành: Khoa học máy tính

- Tên ý tưởng (sản phẩm): Ứng dụng phân tích và phân loại văn bản cảm xúc

***1.1.5.1. Mô tả ý tưởng sản phẩm***

**Ứng dụng phân tích và phân loại văn bản cảm xúc** là một dự án khoa học máy tính nhằm sử dụng các kỹ thuật xử lý ngôn ngữ tự nhiên (NLP) và học máy để phân tích cảm xúc được biểu đạt trong văn bản, chẳng hạn như các bài đăng trên mạng xã hội, đánh giá sản phẩm hoặc bình luận của người dùng.

Ứng dụng sẽ nhận đầu vào là văn bản từ người dùng hoặc một tập hợp văn bản và sau đó phân loại chúng theo các nhãn cảm xúc như tích cực, tiêu cực, hoặc trung lập. Đối với các dự án nâng cao hơn, cảm xúc có thể được phân loại thành nhiều nhóm hơn như giận dữ, vui vẻ, buồn bã, ngạc nhiên, v.v.

Có thể thu thập dữ liệu từ các nguồn như Twitter (thông qua API), các trang đánh giá sản phẩm, bình luận trên YouTube, v.v. Ngoài ra, các tập dữ liệu công khai như IMDb reviews hoặc Yelp reviews cũng rất hữu ích.

Sử dụng các mô hình phân loại phổ biến như Logistic Regression, Naive Bayes, hoặc các mô hình nâng cao hơn như Support Vector Machine (SVM) hoặc Random Forest. Với lượng dữ liệu lớn và phức tạp hơn, bạn có thể sử dụng các mô hình học sâu như LSTM hoặc BERT. Dữ liệu sau khi tiền xử lý sẽ được sử dụng để huấn luyện mô hình phân loại cảm xúc. Mô hình này học từ các văn bản đã được gán nhãn để có thể dự đoán cảm xúc cho các văn bản mới.

Ứng dụng có thể tổng hợp cảm xúc từ nhiều nguồn dữ liệu khác nhau để xác định xu hướng cảm xúc của người dùng trên mạng xã hội hoặc đối với một sản phẩm cụ thể.

Ứng dụng thực tế:

* Phân tích cảm xúc người dùng đối với sản phẩm/dịch vụ: Dựa trên đánh giá sản phẩm, phản hồi của khách hàng để cải thiện chất lượng.
* Phân tích tâm lý xã hội: Đo lường và dự đoán tâm lý của người dùng trên các nền tảng mạng xã hội.
* Dự đoán xu hướng thị trường: Sử dụng phân tích cảm xúc để dự đoán xu hướng tiêu dùng.

***1.1.5.2. Đánh giá khả năng phát triển của sản phẩm***

| Ước tính số lượng người sẽ hưởng lợi từ sản phẩm của bạn | 10000 |
| --- | --- |
| Ước tính số lượng người sẽ bỏ tiền ra mua sản phẩm của bạn | 200 |
| Liệt kê các loại đối tượng sẽ được hưởng lợi từ ý tưởng sản phẩm của bạn (ít nhất là 1,nhiều nhất là 5): | * Doanh nghiệp và công ty thương mại * các nền tảng mạng xã hội * Các nhà nghiên cứu xã hội và tâm lý học * Công ty tư vấn * Nhà phát triển ứng dụng |

|  | Trung bình | Khá | Cao | Rất cao |
| --- | --- | --- | --- | --- |
| Đánh giá khả năng các nhà đầu tư sẽ bỏ tiền ra cho ý tưởng sản phẩm của bạn: |  | X |  |  |

|  | Trung bình | Khá | Cao | Rất cao |
| --- | --- | --- | --- | --- |
| Ước tính độ khó để phát triển thành công sản phẩm của bạn: |  |  | X |  |

| Ước lượng thời gian cần thiết để phát triển sản phẩm của bạn: | 10 tháng |
| --- | --- |

## 1.2. Ý tưởng đề xuất

Tên sản phẩm: **XÂY DỰNG WEBSITE NGHE NHẠC TRỰC TUYẾN KẾT HỢP AI ĐỀ XUẤT BÀI HÁT**

* Tên nhóm: Nhóm 3

| STT | Họ và tên | Vai trò | Công việc |
| --- | --- | --- | --- |
| 1 | Ngô Tấn Vĩnh | Thành viên | Thiết kế giao diện người dùng, làm báo cáo word |
| 2 | Trần Văn Vỹ | Thành viên | Phân tích, thiết kế mô hình AI, làm báo cáo word |
| 3 | Lê Trần Ánh Nhung | Thành viên | Quản lý tiến độ, đánh giá mô hình, làm báo cáo word và powerpoint |
| 4 | Cao Minh | Thành viên | Phân tích, xây dựng mô hình AI, làm báo cáo word |
| 5 | Nguyễn Nhật Minh | Thành viên | Thu thập và xử lý dữ liệu, làm báo cáo word |

* Thời gian triển khai:

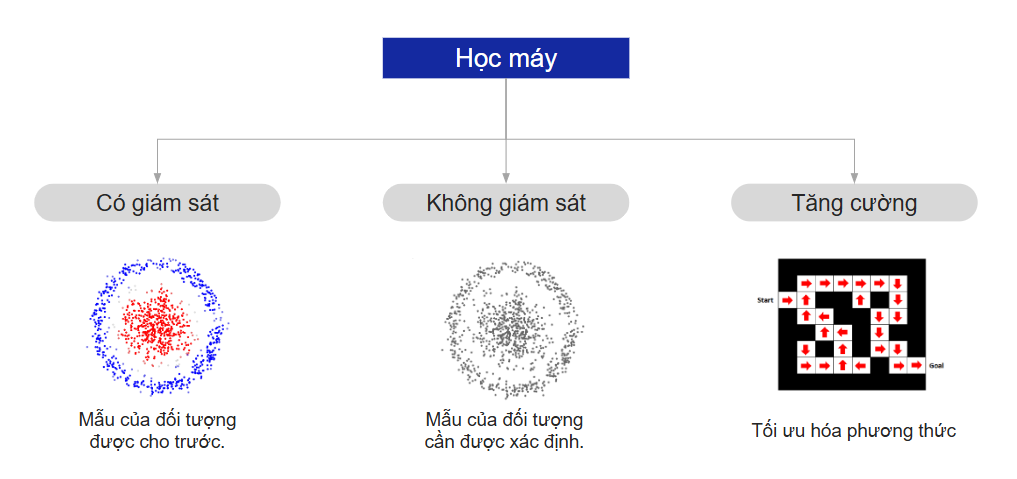
Ngày bắt đầu: 25/08/2024

Ngày kết thúc: 13/10/2024

# CHƯƠNG II: CƠ SỞ LÝ THUYẾT

## 2.1. Machine Learning

Machine Learning (học máy) là một lĩnh vực của trí tuệ nhân tạo (AI) tập trung vào việc phát triển các thuật toán và mô hình giúp máy tính học hỏi từ dữ liệu để thực hiện các nhiệm vụ mà không cần lập trình cụ thể cho từng trường hợp. Trong bối cảnh hệ thống đề xuất nhạc theo sở thích của người dùng, các thuật toán học máy giúp phân tích dữ liệu nghe nhạc của người dùng và đưa ra các gợi ý phù hợp. Có ba loại học máy chính:



* Học có giám sát (Supervised Learning): Sử dụng các dữ liệu có gắn nhãn để huấn luyện mô hình. Trong hệ thống đề xuất nhạc, các dữ liệu này có thể là các bài hát mà người dùng đã nghe và đánh giá. Các thuật toán như hồi quy tuyến tính, cây quyết định, và các mạng nơ-ron sâu có thể được sử dụng.
* Học không giám sát (Unsupervised Learning): Phân tích các dữ liệu không có nhãn để tìm ra các mẫu và cấu trúc ẩn. Ví dụ, phân cụm (clustering) có thể được sử dụng để nhóm các người dùng có sở thích âm nhạc tương tự, từ đó đưa ra các đề xuất phù hợp.
* Học tăng cường (Reinforcement Learning): Tương tác với môi trường và nhận phản hồi dưới dạng phần thưởng hoặc hình phạt để tối ưu hóa chiến lược đề xuất. Ví dụ, hệ thống có thể học cách điều chỉnh các gợi ý dựa trên phản hồi của người dùng.

Các khái niệm như overfitting, underfitting, cross-validation, và hyperparameter tuning cũng quan trọng trong việc đảm bảo mô hình học máy hoạt động chính xác và hiệu quả trên các dữ liệu mới.

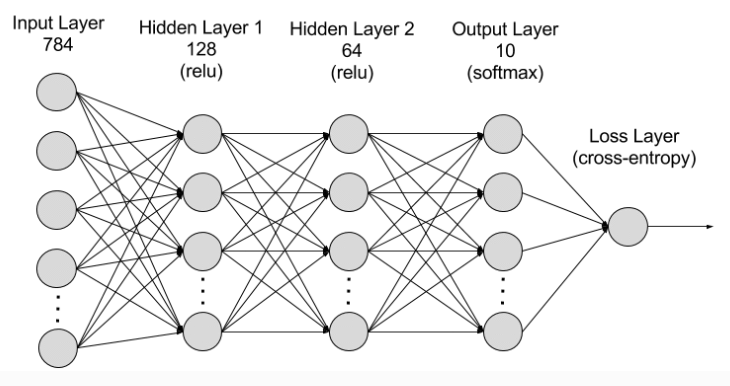
## 2.2. Deep Learning

Deep Learning có thể được xem là một lĩnh vực con của Machine Learning. Nó được xây dựng dựa trên các khái niệm phức tạp và chủ yếu hoạt động với các mạng nơ-ron nhân tạo để bắt chước khả năng tư duy và suy nghĩ của bộ não con người.

Các khái niệm liên quan đến mạng nơ ron nhân tạo và Deep Learning đã xuất hiện từ khoảng những năm 1960, tuy nhiên nó lại bị giới hạn bởi khả năng tính toán và số lượng dữ liệu lúc bấy giờ. Cho đến hiện tại, nhờ những tiến bộ trong phân tích dữ liệu lớn đã cho tạo điều kiện cho mạng nơ ron phát triển mạnh mẽ. Đây được xem là một trong những bước đột phá lớn nhất trong Machine Learning.

Các nơ ron nhân tạo là những mô đun phần mềm được gọi là nút, sử dụng các phép toán để xử lý dữ liệu. Một mạng nơ ron gồm có 3 lớp chính:

* Lớp đầu vào (Input Layer): Đây là các nút để nhập dữ liệu đầu vào cho hệ thống.
* Lớp ẩn (Hidden Layer): Dữ liệu đầu vào được chuyển đến các lớp ẩn để xử lý thông tin ở các cấp độ khác nhau, thích ứng với hành vi của mình khi nhận được thông tin mới.
* Lớp đầu ra (Output Layer): Gồm các nút xuất dữ liệu.



So với học máy truyền thống, mạng học sâu đem lại những lợi ích sau:

* Xử lý hiệu quả dữ liệu phi cấu trúc và đưa ra các nhận xét chung mà không cần trích xuất tính năng thủ công.
* Khám phá những mối quan hệ và mẫu ẩn, phân tích sâu hơn một lượng lớn dữ liệu và hé lộ những thông tin chuyên sâu mới có thể đưa vào đào tạo.
* Học không giám sát, tự học hỏi và cải thiện dần theo thời gian dựa trên hành vi của người dùng mà không yêu cầu các tập dữ liệu được dán nhãn và phải có tính biến động cao.

Nhưng bên cạnh đó, học sau là một công nghệ tương đối mới nên cũng sẽ có những thách thức khi triển khai thực tiễn. Các thuật toán học sâu cần được đào tạo trên dữ liệu chất lượng cao theo số lượng lớn nên khi giá trị ngoại lai hoặc sai sót trong tập dữ liệu đầu vào thì sẽ ảnh hưởng đáng kể đến kết quả ra. Và bởi vì khả năng xử lý lớn thiên về điện toán nên yêu cầu có cơ sở hạ tầng đủ năng lực điện toán để hoạt động đúng cách, nếu không sẽ mất rất nhiều thời gian để xử lý kết quả.

## 2.3. Ngôn ngữ và công cụ lập trình

Để xây dựng hệ thống đề xuất nhạc, cần sử dụng các ngôn ngữ và công cụ lập trình mạnh mẽ và linh hoạt. Một số ngôn ngữ và công cụ phổ biến bao gồm:

**Python**

Đây là ngôn ngữ lập trình phổ biến nhất trong lĩnh vực học máy và phân tích dữ liệu. Python có nhiều thư viện mạnh mẽ hỗ trợ học máy như Scikit-Learn, TensorFlow, PyTorch và Pandas. Python cũng rất linh hoạt và dễ học, giúp nhanh chóng phát triển và triển khai các mô hình học máy.

**SQL (Structured Query Language)**

SQL được sử dụng để truy vấn và thao tác dữ liệu trong các cơ sở dữ liệu quan hệ. Trong hệ thống đề xuất nhạc, SQL có thể được sử dụng để thu thập, lưu trữ và xử lý dữ liệu người dùng cũng như dữ liệu âm nhạc.

**HTML (HyperText Markup Language)**

HTML là ngôn ngữ chuẩn để tạo ra các trang web. Trong hệ thống đề xuất nhạc, HTML được sử dụng để xây dựng giao diện người dùng, cho phép người dùng tương tác với hệ thống thông qua trình duyệt web.

**JavaScript (JS)**

JavaScript là ngôn ngữ lập trình phía khách hàng được sử dụng rộng rãi để tạo ra các tính năng tương tác và động trên trang web. Trong hệ thống đề xuất nhạc, JavaScript có thể được sử dụng để làm cho giao diện người dùng trở nên sinh động và phản ứng nhanh với các thao tác của người dùng.

**CSS (Cascading Style Sheets)**

CSS được sử dụng để định dạng và bố trí các yếu tố trên trang web. Trong hệ thống đề xuất nhạc, CSS giúp tạo ra giao diện người dùng hấp dẫn và dễ sử dụng, đảm bảo trải nghiệm tốt cho người dùng.

## 2.4. Giới thiệu thư viện

**Scikit-Learn**

Scikit-Learn là một thư viện mạnh mẽ cho học máy trong Python, nổi bật với khả năng cung cấp một bộ công cụ phong phú cho các tác vụ như phân loại, hồi quy, phân cụm, và giảm chiều. Được xây dựng dựa trên các thư viện như NumPy và SciPy, Scikit-Learn cho phép người dùng dễ dàng triển khai và tinh chỉnh các mô hình học máy thông qua giao diện trực quan và đơn giản.

Thư viện này không chỉ hỗ trợ các thuật toán hiện đại mà còn cung cấp các phương pháp đánh giá mô hình và tối ưu hóa tham số, giúp cải thiện hiệu suất của các giải pháp học máy. Với khả năng tích hợp tốt với các công cụ phân tích dữ liệu khác, Scikit-Learn là lựa chọn lý tưởng cho các nhà nghiên cứu và nhà phát triển trong lĩnh vực khoa học dữ liệu.

**Numpy**

Numpy (Numeric Python): là một thư viện toán học phổ biến của Python. Cho phép làm việc hiệu quả với ma trận và mảng, đặc biệt là dữ liệu ma trận và mảng lớn với tốc độ xử lý nhanh hơn nhiều lần khi chỉ sử dụng “core Python” đơn thuần.

**Pandas**

Thư viện Pandas trong Python là một thư viện mã nguồn mở cung cấp một loạt các công cụ và cấu trúc dữ liệu, chủ yếu là DataFrame, để thực hiện các nhiệm vụ phân tích và xử lý dữ liệu dễ dàng hơn trong Python. Thư viện này đặc biệt phù hợp cho các công việc liên quan đến khoa học dữ liệu và phân tích dữ liệu.

**Tensorflow**

Đây là một thư viện mã nguồn mở do Google phát triển, được thiết kế để xây dựng và huấn luyện các mô hình học máy và mạng nơron. TensorFlow hỗ trợ nhiều loại mô hình như mạng nơron sâu (DNN), mạng tích chập (CNN), mạng tuần hoàn (RNN), và nhiều phương pháp học sâu (Deep Learning) khác. Với khả năng xử lý linh hoạt, nó được sử dụng rộng rãi trong các lĩnh vực như thị giác máy tính, xử lý ngôn ngữ tự nhiên, và trí tuệ nhân tạo nói chung. TensorFlow cũng hỗ trợ chạy trên nhiều nền tảng khác nhau, bao gồm CPU, GPU, và TPU để tăng tốc quá trình tính toán.

**Streamlit**

Đây là một thư viện mã nguồn mở dành cho Python, giúp dễ dàng xây dựng các ứng dụng web tương tác với giao diện người dùng trực quan chỉ bằng vài dòng code. Điểm mạnh của Streamlit là khả năng phát triển các dashboard hoặc giao diện trực quan để hiển thị kết quả của các mô hình machine learning hoặc dữ liệu phức tạp mà không cần phải viết mã HTML, CSS, hay JavaScript.

**Ipython**

IPython là một thư viện Python mạnh mẽ, chủ yếu được sử dụng trong các môi trường phát triển như Jupyter Notebook. Nó cung cấp một giao diện tương tác nâng cao, cho phép người dùng thực hiện các lệnh Python ngay lập tức và quản lý lịch sử lệnh. IPython hỗ trợ tính năng gỡ lỗi, magic commands (các lệnh đặc biệt như %timeit, %run), và khả năng hiển thị đa phương tiện. Thư viện này rất hữu ích cho việc thử nghiệm mã, giảng dạy, và nghiên cứu khoa học dữ liệu, giúp người dùng tương tác hiệu quả hơn với các thư viện như Pandas, NumPy và Matplotlib.

## KẾT LUẬN CHƯƠNG II

Qua chương này đã cung cấp cái nhìn tổng quan về những khái niệm cốt lõi liên quan đến máy học, ngôn ngữ lập trình, và các thư viện quan trọng trong việc xây dựng hệ thống đề xuất nhạc. Máy học đã trở thành một phần không thể thiếu trong quá trình phát triển các ứng dụng thông minh, cho phép hệ thống phân tích và học hỏi từ dữ liệu để đưa ra những gợi ý cá nhân hóa cho người dùng. Thông qua các thuật toán và phương pháp học máy, hệ thống có thể nhận diện và nắm bắt những sở thích, xu hướng, và hành vi nghe nhạc của từng người dùng, từ đó tạo ra những trải nghiệm âm nhạc độc đáo và phong phú.

# CHƯƠNG III: GIỚI THIỆU THUẬT TOÁN

## 3.1. Collaborative Filtering (Lọc cộng tác)

Lọc cộng tác là một trong những phương pháp phổ biến nhất để xây dựng hệ thống đề xuất, đặc biệt trong lĩnh vực âm nhạc. Phương pháp này dựa trên giả định rằng nếu người dùng A có sở thích tương tự với người dùng B trong quá khứ, thì những bài hát mà B thích có thể sẽ được A thích. Lọc cộng tác có thể chia thành hai loại chính:

### 3.1.1. User-based Collaborative Filtering (Lọc cộng tác dựa trên người dùng)

Trong phương pháp này, hệ thống đề xuất nhạc dựa trên sự tương đồng giữa người dùng hiện tại và các người dùng khác. Các bước thực hiện như sau:

1. Tính toán độ tương đồng giữa các người dùng: Sử dụng các số liệu như cosine similarity, Pearson correlation để đo độ tương đồng giữa các người dùng dựa trên lịch sử nghe nhạc và đánh giá của họ.
2. Tìm nhóm người dùng tương tự: Xác định một nhóm nhỏ người dùng có độ tương đồng cao với người dùng hiện tại.
3. Đưa ra đề xuất: Dựa trên sở thích của nhóm người dùng tương tự, hệ thống sẽ đề xuất những bài hát mà họ đã nghe và đánh giá cao nhưng người dùng hiện tại chưa nghe.

Ví dụ: Nếu người dùng A và B đều thích các bài hát của nghệ sĩ X, và người dùng B cũng thích các bài hát của nghệ sĩ Y, hệ thống sẽ đề xuất các bài hát của nghệ sĩ Y cho người dùng A.

### 3.1.2. Item-based Collaborative Filtering (Lọc cộng tác dựa trên bài hát)

Phương pháp này tập trung vào việc tìm kiếm sự tương đồng giữa các bài hát, thay vì giữa các người dùng. Các bước thực hiện như sau:

1. Tính toán độ tương đồng giữa các bài hát: Sử dụng các số liệu như cosine similarity để đo độ tương đồng giữa các bài hát dựa trên lịch sử nghe nhạc và đánh giá của người dùng.
2. Xác định các bài hát tương tự: Đối với mỗi bài hát mà người dùng đã nghe và đánh giá cao, xác định các bài hát tương tự.
3. Đưa ra đề xuất: Dựa trên sự tương đồng giữa các bài hát, hệ thống sẽ đề xuất những bài hát tương tự với những bài hát mà người dùng đã nghe và đánh giá cao.

Ví dụ: Nếu người dùng A thích bài hát X, và bài hát Y có độ tương đồng cao với bài hát X, hệ thống sẽ đề xuất bài hát Y cho người dùng A.

## 3.2. K-nearest neighbor

K lân cận (K-nearest neighbor hay KNN) là một trong những thuật toán học có giám sát đơn giản nhất. Mọi tính toán được thực hiện tại pha kiểm tra. KNN có thể được áp dụng vào các bài toán phân loại và hồi quy. KNN còn được gọi là một thuật toán lười học, instance-based [AKA91], hoặc memory-based learning.

KNN là thuật toán đi tìm đầu ra của một điểm dữ liệu mới chỉ dựa trên thông tin của K điểm dữ liệu gần nhất trong tập huấn luyện.

A diagram of a circle with different colored squares and a question mark

Description automatically generated

Vì KNN ra quyết định dựa trên các điểm gần nhất nên có hai vấn đề ta cần lưu tâm. Thứ nhất, khoảng cách được định nghĩa như thế nào. Thứ hai, cần phải tính toán khoảng cách như thế nào cho hiệu quả.

* Với vấn đề thứ nhất, mỗi điểm dữ liệu được thể hiện bằng một vector đặc trưng, khoảng cách giữa hai điểm chính là khoảng cách giữa hai vector đó. Có nhiều loại khoảng cách khác nhau tuỳ vào bài toán, nhưng khoảng cách được sử dụng nhiều nhất là khoảng cách Euclid.
* Vấn đề thứ hai cần được lưu tâm hơn, đặc biệt với các bài toán có tập huấn luyện lớn và vector dữ liệu có kích thước lớn. Giả sử các vector huấn luyện là các cột của ma trận X ∈ R d×N với d và N lớn. KNN sẽ phải tính khoảng cách từ một điểm dữ liệu mới z ∈ R d đến toàn bộ N điểm dữ liệu đã cho và chọn ra K khoảng cách nhỏ nhất. Nếu không có cách tính hiệu quả, khối lượng tính toán sẽ rất lớn.

Với bài toán hồi quy, chúng ta cũng hoàn toàn có thể sử dụng phương pháp tương tự: đầu ra của một điểm được xác định dựa trên đầu ra của các điểm lân cận và khoảng cách tới chúng. Giả sử x1, . . . , xK là K điểm lân cận của một điểm dữ liệu z với đầu ra tương ứng là y1, . . . , yK. Giả sử các trọng số ứng với các lân cận này là w1, . . . , wK. Kết quả dự đoán đầu ra của z có thể được xác định bởi

A math equations with black and white numbers

Description automatically generated with medium confidence

**Ưu điểm**

+ Độ phức tạp tính toán của quá trình huấn luyện gần như bằng 0. Việc tính bình phương `2 norm của mỗi điểm dữ liệu huấn luyện có thể được thực hiện trước trong bước này.

+ Việc dự đoán kết quả của dữ liệu mới tương đối đơn giản sau khi đã xác định được các điểm lân cận.

+ KNN không không cần giả sử về phân phối của từng nhãn

**Nhược điểm**

+ KNN nhạy cảm với nhiễu khi K nhỏ.

+ Khi sử dụng KNN, phần lớn tính toán nằm ở pha kiểm tra. Trong đó việc tính khoảng cách tới từng điểm dữ liệu huấn luyện tốn nhiều thời gian, đặc biệt là với các cơ sở dữ liệu có số chiều lớn và có nhiều điểm dữ liệu. K càng lớn thì độ phức tạp càng cao. Ngoài ra, việc lưu toàn bộ dữ liệu trong bộ nhớ cũng ảnh hưởng tới hiệu năng của KNN

## 3.3. Convolutional Neural Networks (CNN)

Convolutional Neural Network (CNN – Mạng nơ-ron tích chập) là một trong những mô hình Deep Learning tiên tiến để nhận dạng và phân loại hình ảnh. Trong đó, xác định đối tượng và nhận dạng khuôn mặt là một trong số những lĩnh vực mà CNN được sử dụng rộng rãi.

**Convolution Layer**

Đây là lớp cốt lõi của CNN, chịu trách nhiệm trích xuất các đặc trưng từ hình ảnh đầu vào. Quá trình này diễn ra bằng cách trượt một bộ lọc (filter) hay kernel trên toàn bộ hình ảnh, thực hiện phép nhân tích chập giữa các giá trị pixel của hình ảnh và bộ lọc. Kết quả của mỗi phép nhân tích chập tạo ra một feature map, thể hiện mức độ tương đồng giữa bộ lọc và vùng ảnh tương ứng.

A close-up of a grid

Description automatically generated

* Bộ lọc (Filter/Kernel): Là một ma trận nhỏ chứa các trọng số, được học trong quá trình huấn luyện. Mỗi bộ lọc được thiết kế để phát hiện một đặc trưng cụ thể trong hình ảnh, chẳng hạn như cạnh, góc cạnh, hoặc hình dạng phức tạp hơn.
* Bước trượt (Stride): Xác định khoảng cách di chuyển của bộ lọc sau mỗi lần tích chập. Bước trượt lớn hơn sẽ tạo ra feature map nhỏ hơn, giảm độ phân giải không gian nhưng tăng tốc độ tính toán.
* Đệm (Padding): Bổ sung thêm các pixel xung quanh viền của hình ảnh đầu vào, giúp kiểm soát kích thước của feature map đầu ra và tránh mất thông tin ở rìa ảnh.

Giả sử input của 1 convolutional layer tổng quát là tensor kích thước H \* W \* D. Kernel có kích thước F \* F \* D (kernel luôn có depth bằng depth của input và F là số lẻ), stride: S, padding: P. Convolutional layer áp dụng K kernel.

Output của layer là tensor 3 chiều có kích thước:

A white square with black text

Description automatically generated

**Pooling Layer**

Lớp gộp được sử dụng để giảm kích thước của feature map, từ đó giảm số lượng tham số, tăng tốc độ tính toán và kiểm soát overfitting. Các phương pháp gộp phổ biến bao gồm: Max pooling và Average pooling

**Activation Layer**

Lớp này áp dụng một hàm kích hoạt phi tuyến (nonlinear activation function) lên đầu ra của lớp tích chập hoặc lớp gộp. Hàm kích hoạt giúp giới thiệu tính phi tuyến vào mô hình, cho phép CNN học cách biểu diễn phức tạp hơn từ dữ liệu. Các hàm kích hoạt phổ biến bao gồm:

* ReLU (Rectified Linear Unit): Giữ nguyên giá trị dương và gán giá trị 0 cho giá trị âm.
* Sigmoid: Chuyển đổi giá trị đầu vào thành giá trị trong khoảng từ 0 đến 1.
* Tanh (Hyperbolic Tangent): Chuyển đổi giá trị đầu vào thành giá trị trong khoảng từ -1 đến 1.

**Fully Connected Layer**

Lớp này thường nằm ở cuối mạng CNN, nhận đầu vào là một vector đặc trưng được làm phẳng từ các lớp trước đó. Mục đích của lớp kết nối đầy đủ là thực hiện phân loại hoặc hồi quy dựa trên vector đặc trưng này.

Quá trình huấn luyện CNN:

1. Lan truyền xuôi (Forward propagation): Đưa dữ liệu huấn luyện qua mạng CNN để tính toán giá trị dự đoán.
2. Tính toán hàm mất mát: So sánh giá trị dự đoán với giá trị thực tế để tính toán mức độ sai lệch.
3. Lan truyền ngược (Backpropagation): Lan truyền ngược lỗi từ đầu ra đến đầu vào, tính toán gradient của hàm mất mát theo từng trọng số.
4. Cập nhật trọng số: Sử dụng thuật toán tối ưu hóa (ví dụ như Gradient Descent) để cập nhật các trọng số dựa trên gradient đã tính, nhằm giảm thiểu hàm mất mát.

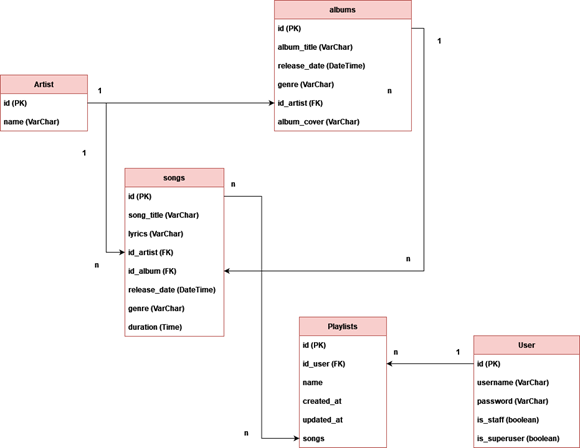
## KẾT LUẬN CHƯƠNG III

Việc thấu hiểu hành vi và sở thích của người dùng là yếu tố then chốt trong việc xây dựng một hệ thống đề xuất nhạc hiệu quả. Hai phương pháp chính của lọc cộng tác, dựa trên người dùng và dựa trên bài hát, giúp hệ thống nắm bắt mối liên kết giữa người dùng và âm nhạc, từ đó đưa ra những gợi ý phù hợp. Bên cạnh đó, kỹ thuật K-nearest neighbor (KNN) là công cụ hữu ích trong việc xác định các bài hát tương đồng dựa trên khoảng cách trong không gian đặc trưng. Đặc biệt, mô hình mạng nơron tích chập (CNN) còn được sử dụng để trích xuất các đặc trưng từ ảnh bìa album, giúp hệ thống bổ sung thêm yếu tố hình ảnh trong quá trình đề xuất nhạc, từ đó làm tăng tính cá nhân hóa và tính chính xác trong gợi ý. Sự kết hợp nhuần nhuyễn giữa các phương pháp lọc cộng tác, KNN và CNN hứa hẹn mang lại một hệ thống đề xuất thông minh, nâng cao trải nghiệm người dùng trên nền tảng website nghe nhạc trực tuyến.

Dựa trên cơ sở lý thuyết vững chắc này, chương tiếp theo sẽ tập trung vào việc triển khai hệ thống đề xuất thực tế. Các thuật toán đã được trình bày sẽ được ứng dụng để huấn luyện mô hình với tập dữ liệu cụ thể, nhằm xây dựng một hệ thống đề xuất nhạc hiệu quả và đáng tin cậy.

# CHƯƠNG IV: XÂY DỰNG CHƯƠNG TRÌNH

## 4.1. Dữ liệu(Dataset)



**Các Thực Thể và Thuộc Tính:**

1. Artist (Nghệ sĩ):

* id (PK): Khóa chính
* name (VarChar): Tên của nghệ sĩ

2. Albums (Album):

* id (PK): Khóa chính
* album\_title (VarChar): Tên của album
* release\_date (DateTime): Ngày phát hành của album
* genre (VarChar): Thể loại của album
* id\_artist (FK): Khóa ngoại tham chiếu đến nghệ sĩ đã tạo ra album
* album\_cover (VarChar): Đường dẫn của ảnh bìa album

3. Songs (Bài hát):

* id (PK): Khóa chính
* song\_title (VarChar): Tên của bài hát
* lyrics (VarChar): Lời bài hát
* id\_artist (FK): Khóa ngoại tham chiếu đến nghệ sĩ biểu diễn bài hát
* id\_album (FK): Khóa ngoại tham chiếu đến album chứa bài hát
* release\_date (DateTime): Ngày phát hành của bài hát
* genre (VarChar): Thể loại của bài hát (có thể thừa hưởng từ album)
* duration (Time): Thời lượng của bài hát

4. Playlists (Danh sách phát):

* id (PK): Khóa chính
* id\_user (FK): Khóa ngoại tham chiếu đến người dùng đã tạo danh sách phát
* name (VarChar): Tên của danh sách phát
* created\_at (DateTime): Thời điểm danh sách phát được tạo
* updated\_at (DateTime): Thời điểm danh sách phát được cập nhật lần cuối
* songs: Chỉ ra mối quan hệ nhiều-nhiều với thực thể bài hát, tức là một danh sách phát có thể chứa nhiều bài hát, và một bài hát có thể xuất hiện trong nhiều danh sách phát

5. User (Người dùng):

* id (PK): Khóa chính
* username (VarChar): Tên đăng nhập của người dùng
* password (VarChar): Mật khẩu của người dùng
* is\_staff (boolean): Biến boolean cho biết người dùng có quyền quản trị hay không
* is\_superuser (boolean): Biến boolean cho biết người dùng có quyền siêu quản trị hay không

**Mối Quan Hệ:**

* Một-Nhiều:
* Mỗi nghệ sĩ có thể có nhiều album (1 Nghệ sĩ -> n Album)
* Mỗi nghệ sĩ có thể có nhiều bài hát (1 Nghệ sĩ -> n Bài hát)
* Mỗi album có thể chứa nhiều bài hát (1 Album -> n Bài hát)
* Mỗi người dùng có thể có nhiều danh sách phát (1 Người dùng-> n Danh sách phát)
* Nhiều-Nhiều:
* Một danh sách phát có thể có nhiều bài hát, và một bài hát có thể thuộc nhiều danh sách phát (n Danh sách phát <-> n Bài hát)

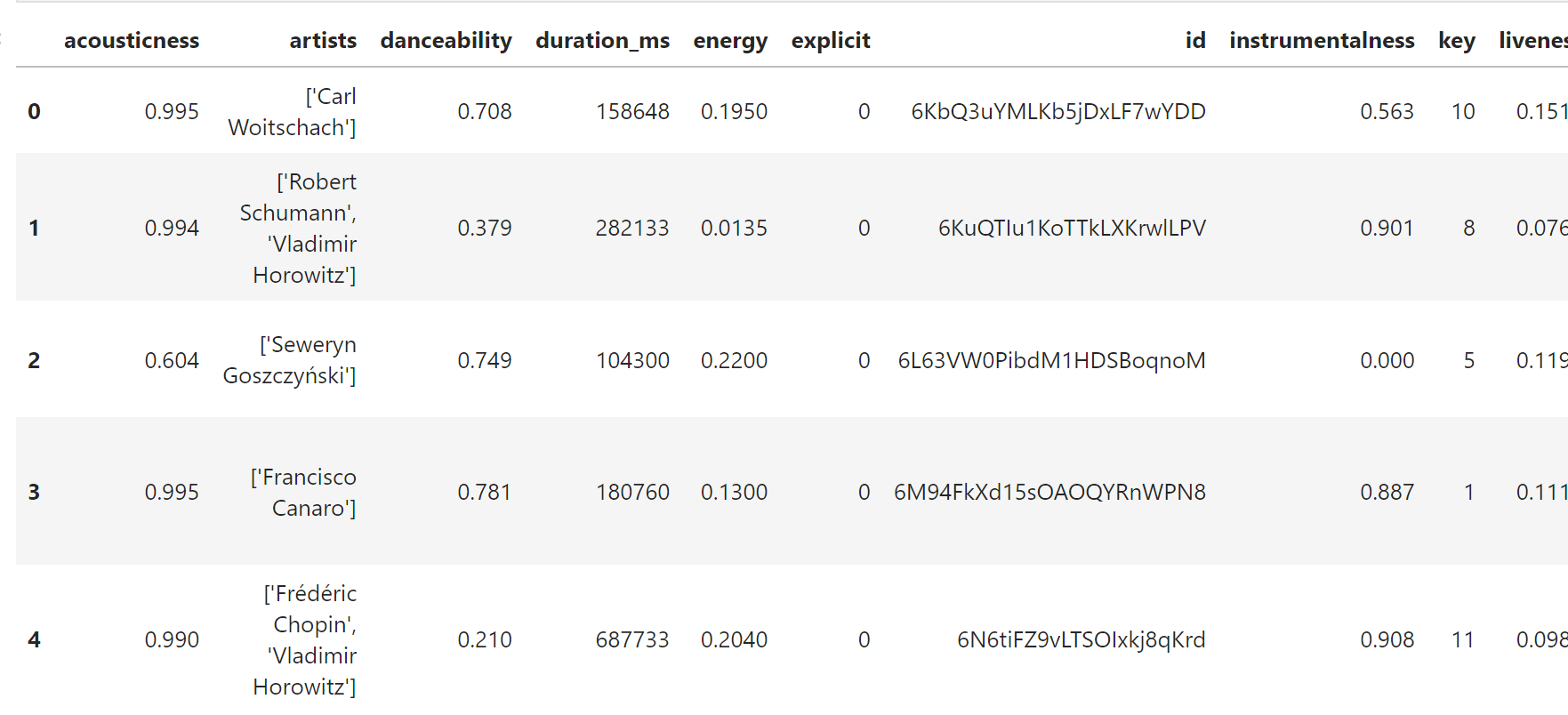
## 4.2. Xây dựng mô hình

### 4.2.1. Cài đặt thư viện:

| import pandas as pd  import numpy as np  import json  import re  import sys  import itertools  from sklearn.feature\_extraction.text import TfidfVectorizer  from sklearn.metrics.pairwise import cosine\_similarity  from sklearn.preprocessing import MinMaxScaler  import matplotlib.pyplot as plt  import spotipy  from spotipy.oauth2 import SpotifyClientCredentials, SpotifyOAuth  import spotipy.util as util  import warnings  from IPython.core.display import display, HTML  import IPython.display as display |
| --- |

### 4.2.2. Xử lý dữ liệu

| spotify\_df = pd.read\_csv('data.csv')  spotify\_df.head() |
| --- |

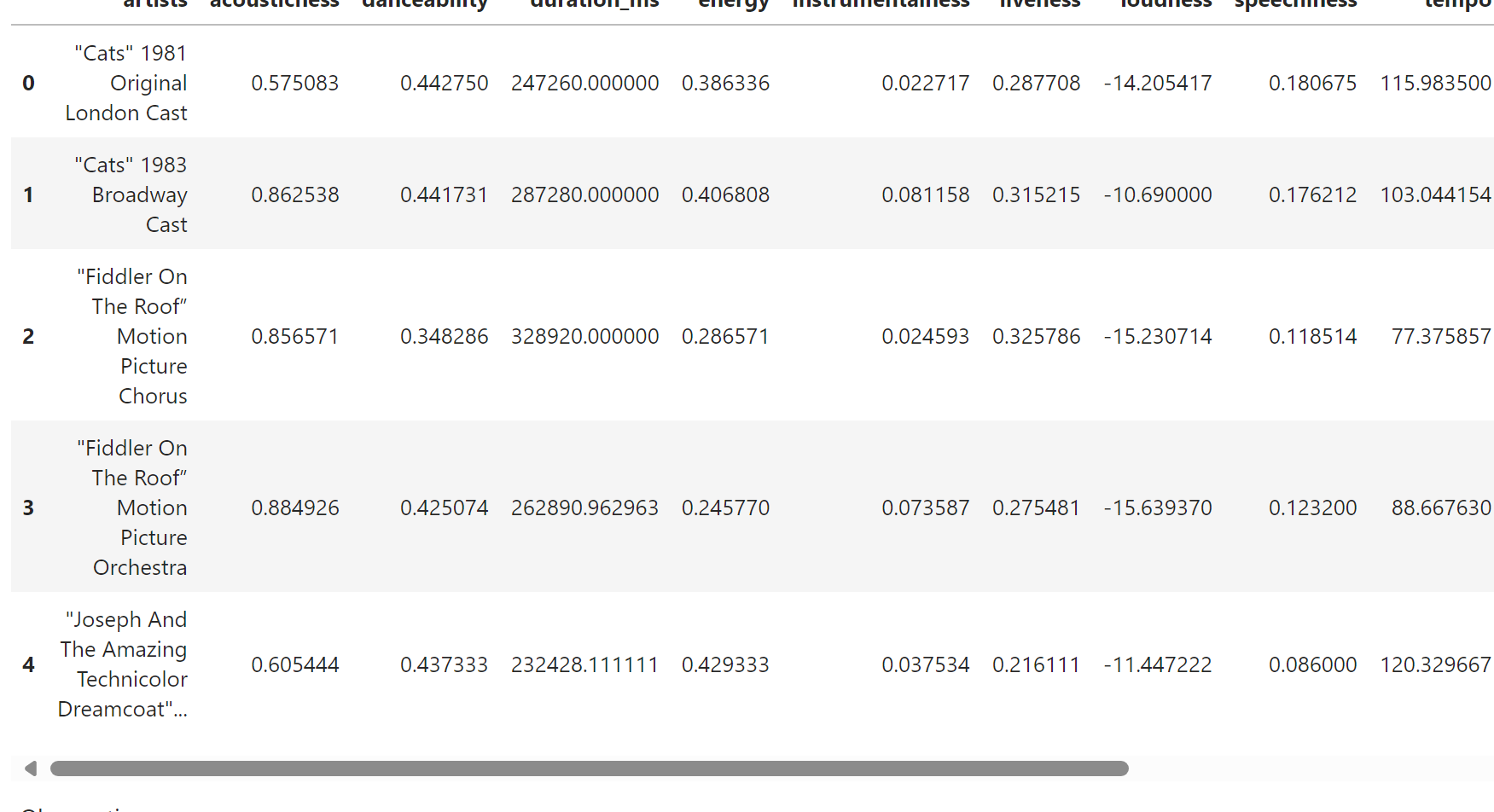


*Dữ liệu trong data.csv*

#Nhận xét:

* Dữ liệu này là một **song level**.
* Có nhiều giá trị số mà chúng em có thể sử dụng để so sánh giữa các bài hát (như "liveness", "tempo", "valence", v.v.).
* Ngày phát hành sẽ hữu ích, nhưng chúng em cần tạo biến OHE cho ngày phát hành với các khoảng thời gian 5 năm.
* Tương tự như trên, chúng em cũng cần tạo các biến OHE cho độ phổ biến. Chúng em cũng sẽ sử dụng các khoảng thời gian 5 năm ở đây.
* Không có thông tin nào liên quan đến thể loại của bài hát, điều này sẽ rất hữu ích. Dữ liệu này không đủ để giúp chúng ta tìm nội dung phù hợp vì đây là hệ thống gợi ý dựa trên nội dung. May mắn thay, có một file data\_w\_genres.csv có thể chứa một số thông tin hữu ích.

| data\_w\_genre **=** pd**.**read\_csv('data\_w\_genres.csv')  data\_w\_genre**.**head() |
| --- |



*Dữ liệu trong data*\_w\_genres.csv

#Nhận xét:

* Dữ liệu này ở cấp độ nghệ sĩ.
* Có các biến liên tục tương tự như trong tập dữ liệu ban đầu của chúng ta, nhưng chúng em sẽ không sử dụng chúng. Chúng em sẽ chỉ sử dụng các giá trị trong tập dữ liệu trước.
* Thể loại sẽ rất hữu ích ở đây và chúng em cần sử dụng nó trong các bước tiếp theo. Hiện tại, cột thể loại dường như ở định dạng danh sách, nhưng theo kinh nghiệm của chúng em thì có thể không phải như vậy. Hãy cùng điều tra thêm về vấn đề này.

**Xử lý cột:**

| data\_w\_genre['genres\_upd'] = data\_w\_genre['genres'].apply(lambda x: [re.sub(' ','\_',i) for i in re.findall(r"'([^']\*)'", x)]) |
| --- |

Sử dụng regex để trích xuất thể loại từ chuỗi ký tự và lưu dưới dạng danh sách các thể loại.Bước tiếp theo là liên kết dữ liệu bài hát và thể loại:

* Tách cột nghệ sĩ trong dữ liệu bài hát thành từng dòng riêng.
* Kết hợp dữ liệu bài hát với dữ liệu thể loại.

| spotify\_df['artists\_upd\_v1'] = spotify\_df['artists'].apply(lambda x: re.findall(r"'([^']\*)'", x))  spotify\_df['artists\_upd\_v2'] = spotify\_df['artists'].apply(lambda x: re.findall('\"(.\*?)\"',x))  spotify\_df['artists\_upd'] = np.where(spotify\_df['artists\_upd\_v1'].apply(lambda x: not x), spotify\_df['artists\_upd\_v2'], spotify\_df['artists\_upd\_v1']) |
| --- |

Xử lý cột artists bằng cách tách tên nghệ sĩ từ chuỗi ký tự. Tiếp theo là tạo mã định danh duy nhất cho mỗi bài hát và xử lý dữ liệu bị trùng lặp:

| spotify\_df['artists\_song'] = spotify\_df.apply(lambda row: row['artists\_upd'][0]+row['name'],axis = 1)  spotify\_df.drop\_duplicates('artists\_song', inplace=True) |
| --- |

Tách dòng và kết hợp với dữ liệu thể loại:

| artists\_exploded = spotify\_df[['artists\_upd','id']].explode('artists\_upd')  artists\_exploded\_enriched = artists\_exploded.merge(data\_w\_genre, how = 'left', left\_on = 'artists\_upd',right\_on = 'artists') |
| --- |

Nhóm thể loại lại với nhau:

| artists\_genres\_consolidated = artists\_exploded\_enriched\_nonnull.groupby('id')['genres\_upd'].apply(list).reset\_index()  artists\_genres\_consolidated['consolidates\_genre\_lists'] = artists\_genres\_consolidated['genres\_upd'].apply(lambda x: list(set(list(itertools.chain.from\_iterable(x))))) |
| --- |

Hoàn thiện việc ghép dữ liệu:

| spotify\_df = spotify\_df.merge(artists\_genres\_consolidated[['id','consolidates\_genre\_lists']], on = 'id', how = 'left') |
| --- |

***Xử lý các đặc trưng:***

* Chuẩn hóa các biến số.
* Biến đổi (OHE) cột year và popularity.
* Tạo đặc trưng TF-IDF cho thể loại âm nhạc.

| spotify\_df.tail()  spotify\_df['year'] = spotify\_df['release\_date'].apply(lambda x: x.split('-')[0])  float\_cols = spotify\_df.dtypes[spotify\_df.dtypes == 'float64'].index.values  ohe\_cols = 'popularity'  spotify\_df['popularity'].describe()  # create 5 point buckets for popularity  spotify\_df['popularity\_red'] = spotify\_df['popularity'].apply(lambda x: int(x/5))  spotify\_df['consolidates\_genre\_lists'] = spotify\_df['consolidates\_genre\_lists'].apply(lambda d: d if isinstance(d, list) else [])  spotify\_df.head()  def ohe\_prep(df, column, new\_name):  tf\_df = pd.get\_dummies(df[column])  feature\_names = tf\_df.columns  tf\_df.columns = [new\_name + "|" + str(i) for i in feature\_names]  tf\_df.reset\_index(drop = True, inplace = True)  return tf\_df |
| --- |

Tạo ra tổ hợp có chứa các đặc trưng cần thiết:

| #function to build entire feature set  def create\_feature\_set(df, float\_cols):  tfidf = TfidfVectorizer()  tfidf\_matrix = tfidf.fit\_transform(df['consolidates\_genre\_lists'].apply(lambda x: " ".join(x)))  genre\_df = pd.DataFrame(tfidf\_matrix.toarray())  genre\_df.columns = ['genre' + "|" + i for i in tfidf.get\_feature\_names()]  genre\_df.reset\_index(drop = True, inplace=True)  #explicity\_ohe = ohe\_prep(df, 'explicit','exp')  year\_ohe = ohe\_prep(df, 'year','year') \* 0.5  popularity\_ohe = ohe\_prep(df, 'popularity\_red','pop') \* 0.15  #scale float columns  floats = df[float\_cols].reset\_index(drop = True)  scaler = MinMaxScaler()  floats\_scaled = pd.DataFrame(scaler.fit\_transform(floats), columns = floats.columns) \* 0.2  #concanenate all features  final = pd.concat([genre\_df, floats\_scaled, popularity\_ohe, year\_ohe], axis = 1)  #add song id  final['id']=df['id'].values  return final  complete\_feature\_set = create\_feature\_set(spotify\_df, float\_cols=float\_cols)#.mean(axis = 0)  complete\_feature\_set.head() |
| --- |

→ Kết quả cuối cùng là một **DataFrame hoàn chỉnh** chứa thông tin về các bài hát, nghệ sĩ, và thể loại nhạc, sẵn sàng để sử dụng cho bước tiếp theo.

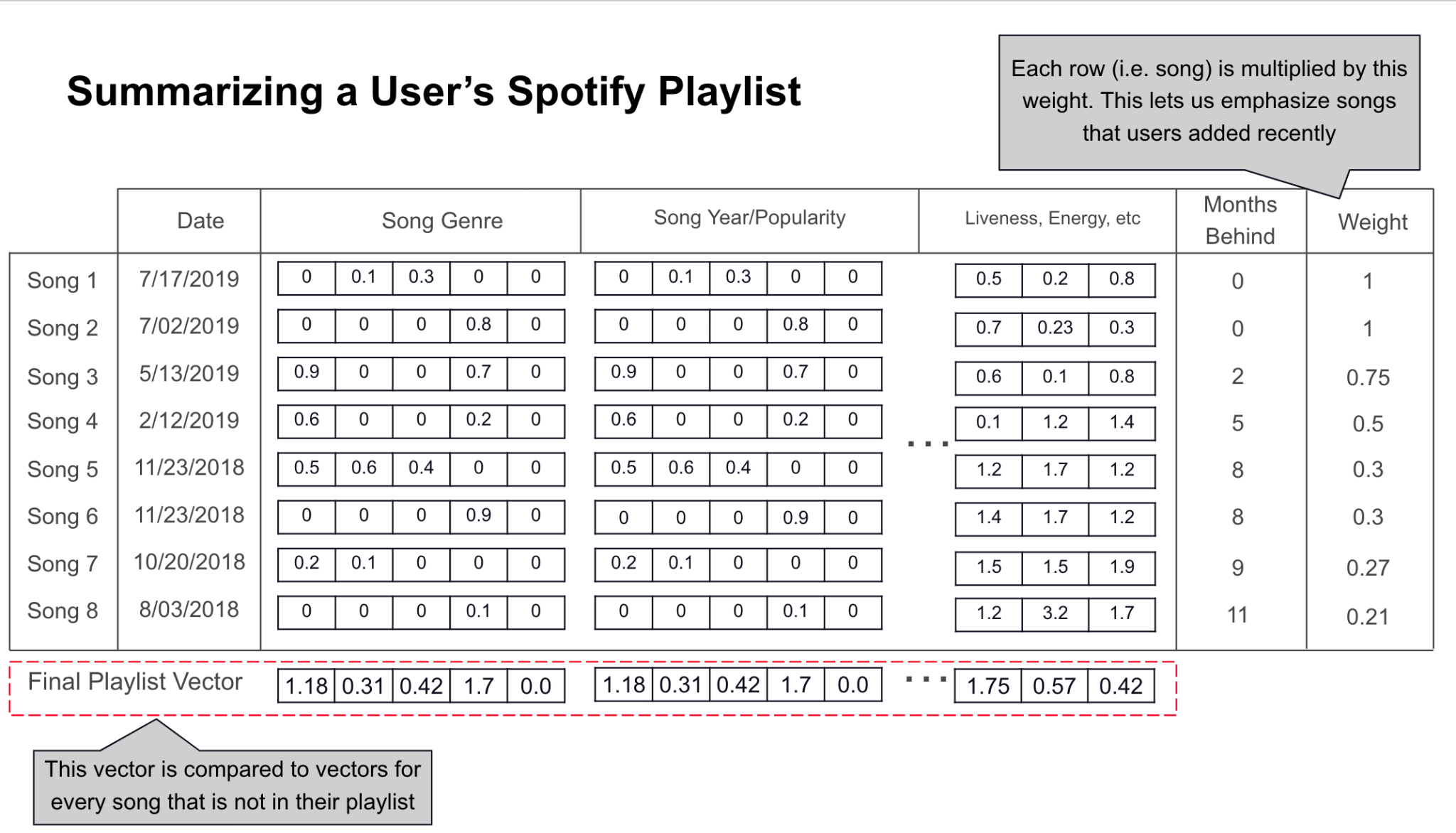
### 4.2.3. Xác thực và truy xuất dữ liệu từ Spotify thông qua API

Đoạn code dưới đây thực hiện các chức năng chính sau:

* Xác thực Spotify API: Sử dụng client\_id, client\_secret, và scope để xác thực và truy cập dữ liệu Spotify.
* Truy xuất danh sách phát: Tải tên và mã định danh (ID) của các playlist của người dùng, cũng như URL hình ảnh bìa.
* Hàm create\_necessary\_outputs: Lấy các bài hát từ một playlist cụ thể, lọc chúng theo những bài có trong tập dữ liệu Kaggle và sắp xếp theo ngày thêm vào.
* Hiển thị danh sách phát: Tạo playlist EDM, chứa các bài hát trong cả Spotify và Kaggle dataset.
* Hàm visualize\_songs: Trực quan hóa bìa album của các bài hát trong playlist bằng matplotlib.

| client\_id = 'id'  client\_secret= 'secret'  scope = 'user-library-read'  if len(sys.argv) > 1:  username = sys.argv[1]  else:  print("Usage: %s username" % (sys.argv[0],))  sys.exit()  auth\_manager = SpotifyClientCredentials(client\_id=client\_id, client\_secret=client\_secret)  sp = spotipy.Spotify(auth\_manager=auth\_manager)  token = util.prompt\_for\_user\_token(scope, client\_id= client\_id, client\_secret=client\_secret, redirect\_uri='http://localhost:8881/')  sp = spotipy.Spotify(auth=token)  #gather playlist names and images.  #images aren't going to be used until I start building a UI  id\_name = {}  list\_photo = {}  for i in sp.current\_user\_playlists()['items']:  id\_name[i['name']] = i['uri'].split(':')[2]  list\_photo[i['uri'].split(':')[2]] = i['images'][0]['url']  def create\_necessary\_outputs(playlist\_name,id\_dic, df):  """  Pull songs from a specific playlist.  Parameters:  playlist\_name (str): name of the playlist you'd like to pull from the spotify API  id\_dic (dic): dictionary that maps playlist\_name to playlist\_id  df (pandas dataframe): spotify datafram    Returns:  playlist: all songs in the playlist THAT ARE AVAILABLE IN THE KAGGLE DATASET  """  #generate playlist dataframe  playlist = pd.DataFrame()  playlist\_name = playlist\_name  for ix, i in enumerate(sp.playlist(id\_dic[playlist\_name])['tracks']['items']):  #print(i['track']['artists'][0]['name'])  playlist.loc[ix, 'artist'] = i['track']['artists'][0]['name']  playlist.loc[ix, 'name'] = i['track']['name']  playlist.loc[ix, 'id'] = i['track']['id'] # ['uri'].split(':')[2]  playlist.loc[ix, 'url'] = i['track']['album']['images'][1]['url']  playlist.loc[ix, 'date\_added'] = i['added\_at']  playlist['date\_added'] = pd.to\_datetime(playlist['date\_added'])    playlist = playlist[playlist['id'].isin(df['id'].values)].sort\_values('date\_added',ascending = False)    return playlist  playlist\_EDM = create\_necessary\_outputs('EDM', id\_name,spotify\_df)  #playlist\_chill = create\_necessary\_outputs('chill',id\_name, spotify\_df)  #playlist\_classical = create\_necessary\_outputs('Epic Classical',id\_name, spotify\_df)  from skimage import io  import matplotlib.pyplot as plt  def visualize\_songs(df):  """  Visualize cover art of the songs in the inputted dataframe  Parameters:  df (pandas dataframe): Playlist Dataframe  """    temp = df['url'].values  plt.figure(figsize=(15,int(0.625 \* len(temp))))  columns = 5    for i, url in enumerate(temp):  plt.subplot(len(temp) / columns + 1, columns, i + 1)  image = io.imread(url)  plt.imshow(image)  plt.xticks(color = 'w', fontsize = 0.1)  plt.yticks(color = 'w', fontsize = 0.1)  plt.xlabel(df['name'].values[i], fontsize = 12)  plt.tight\_layout(h\_pad=0.4, w\_pad=0)  plt.subplots\_adjust(wspace=None, hspace=None)  plt.show()  playlist\_EDM |
| --- |

### 4.2.4. Tạo Playlist Vector

****

Hàm generate\_playlist\_feature tóm tắt đặc trưng của một playlist của người dùng thành một vector duy nhất và trả về một bộ đặc trưng tổng quát, bao gồm cả các bài hát trong và ngoài playlist.

| def generate\_playlist\_feature(complete\_feature\_set, playlist\_df, weight\_factor):  complete\_feature\_set\_playlist = complete\_feature\_set[complete\_feature\_set['id'].isin(playlist\_df['id'].values)]#.drop('id', axis = 1).mean(axis =0)  complete\_feature\_set\_playlist = complete\_feature\_set\_playlist.merge(playlist\_df[['id','date\_added']], on = 'id', how = 'inner')  complete\_feature\_set\_nonplaylist = complete\_feature\_set[~complete\_feature\_set['id'].isin(playlist\_df['id'].values)]#.drop('id', axis = 1)    playlist\_feature\_set = complete\_feature\_set\_playlist.sort\_values('date\_added',ascending=False)  most\_recent\_date = playlist\_feature\_set.iloc[0,-1]  for ix, row in playlist\_feature\_set.iterrows():  playlist\_feature\_set.loc[ix,'months\_from\_recent'] = int((most\_recent\_date.to\_pydatetime() - row.iloc[-1].to\_pydatetime()).days / 30)  playlist\_feature\_set['weight'] = playlist\_feature\_set['months\_from\_recent'].apply(lambda x: weight\_factor \*\* (-x))  playlist\_feature\_set\_weighted = playlist\_feature\_set.copy()  #print(playlist\_feature\_set\_weighted.iloc[:,:-4].columns) playlist\_feature\_set\_weighted.update(playlist\_feature\_set\_weighted.iloc[:,:-4].mul(playlist\_feature\_set\_weighted.weight,0))  playlist\_feature\_set\_weighted\_final = playlist\_feature\_set\_weighted.iloc[:, :-4]  #playlist\_feature\_set\_weighted\_final['id'] = playlist\_feature\_set['id']  return playlist\_feature\_set\_weighted\_final.sum(axis = 0), complete\_feature\_set\_nonplaylist |
| --- |

### 4.2.5. Model CNN VGG16

VGG16 là một kiến trúc mạng nơ-ron tích chập (Convolutional Neural Network - CNN) bao gồm 16 lớp sâu, trong đó có 13 lớp tích chập (convolutional layers) và 3 lớp kết nối đầy đủ (fully connected layers). Kiến trúc của nó bao gồm: Các lớp tích chập (convolutional layers) sử dụng bộ lọc 3x3 để phát hiện các đặc trưng từ hình ảnh. Các lớp pooling (max pooling) giúp giảm kích thước không gian của các bản đồ đặc trưng mà vẫn giữ nguyên các đặc trưng quan trọng. Sau các lớp tích chập, có 3 lớp fully connected để đưa ra dự đoán cuối cùng.

***Huấn luyện VGG16:***

Thông thường, VGG16 được sử dụng dưới dạng một mô hình đã được huấn luyện sẵn trên tập dữ liệu **ImageNet**, nên không cần phải huấn luyện lại từ đầu mà có thể sử dụng các trọng số đã được huấn luyện (pretrained model). Dưới đây là ví dụ code huấn luyện mô hình VGG16:

Các thư viện cần thiết:

| from tensorflow.keras.applications import VGG16 from tensorflow.keras.layers import Dense, Flatten from tensorflow.keras.models import Model from tensorflow.keras.preprocessing.image import ImageDataGenerator from tensorflow.keras.optimizers import Adam |
| --- |

Tải mô hình VGG16, không bao gồm lớp fully connected cuối cùng

| vgg = VGG16(weights='imagenet', include\_top=False, input\_shape=(224, 224, 3)) |
| --- |

Đóng băng các lớp của VGG16 để không huấn luyện lại

| for layer in vgg.layers:  layer.trainable = False |
| --- |

Thêm các lớp fully connected mới:

| x = Flatten()(vgg.output)  x = Dense(512, activation='relu')(x)  x = Dense(10, activation='softmax')(x) |
| --- |

Tạo mô hình mới, biên dịch mô hình và chuẩn bị dữ liệu

| model = Model(inputs=vgg.input, outputs=x)  model.compile(optimizer=Adam(learning\_rate=0.0001), loss='categorical\_crossentropy', metrics=['accuracy'])  train\_datagen = ImageDataGenerator(rescale=1./255, rotation\_range=40, width\_shift\_range=0.2, height\_shift\_range=0.2, shear\_range=0.2, zoom\_range=0.2, horizontal\_flip=True, fill\_mode='nearest')  train\_generator = train\_datagen.flow\_from\_directory(  'path\_to\_train\_data', # Đường dẫn tới thư mục chứa dữ liệu huấn luyện  target\_size=(224, 224),  batch\_size=32,  class\_mode='categorical'  ) |
| --- |

Huấn luyện mô hình và lưu mô hình đã huấn luyện

| model.fit(train\_generator, epochs=10, steps\_per\_epoch=100)  model.save('custom\_vgg16\_model.h5') |
| --- |

### 4.2.6 Mô hình CNN trong hệ thống gợi ý nhạc:

Trong hệ thống gợi ý nhạc, mô hình VGG16 được sử dụng để trích xuất các đặc trưng từ hình ảnh bìa album của các bài hát. Mô hình VGG16 là một kiến trúc mạng nơ-ron tích chập sâu, đã được huấn luyện trên tập dữ liệu ImageNet với mục tiêu nhận diện các đối tượng từ hình ảnh.

Cách hoạt động của mô hình:

* Trích xuất đặc trưng hình ảnh: Mỗi hình ảnh bìa album của bài hát được tải về từ API của Spotify. Sau đó, hình ảnh này được đưa qua mô hình VGG16 đã huấn luyện sẵn để trích xuất các đặc trưng hình ảnh. Các đặc trưng này thể hiện thông tin về hình dạng, kết cấu, và các mẫu phức tạp trong bức ảnh.
* Kết hợp đặc trưng hình ảnh vào hệ thống gợi ý: Các đặc trưng trích xuất từ mô hình CNN giúp bổ sung thêm thông tin trực quan khi gợi ý các bài hát. Điều này cải thiện độ chính xác của hệ thống khi gợi ý những bài hát có sự tương đồng về mặt hình ảnh.

## 4.3. Demo

Trong dự án này, chúng em xây dựng một hệ thống gợi ý âm nhạc sử dụng mô hình học sâu (Deep Learning) và API của Spotify. Hệ thống này giúp người dùng tìm kiếm bài hát theo thể loại và nhận được các gợi ý bài hát dựa trên sở thích của họ. Để thực hiện điều này, chúng em đã sử dụng mô hình VGG16 để trích xuất các đặc trưng từ hình ảnh bìa album, kết hợp với KNN được sử dụng để gợi ý bài hát dựa trên ma trận tương đồng giữa các bài hát. Dưới đây là mô tả chi tiết về từng hàm trong mã nguồn.

**4.3.1 Các thư viện cần thiết**

| import pickle  import streamlit as st  import spotipy  from spotipy.oauth2 import SpotifyClientCredentials  from tensorflow.keras.applications import VGG16  from tensorflow.keras.preprocessing import image  from tensorflow.keras.applications.vgg16 import preprocess\_input  import numpy as np  import requests  from io import BytesIO |
| --- |

Trong phần này, chúng ta nhập các thư viện cần thiết, bao gồm pickle để tải dữ liệu, streamlit để tạo giao diện người dùng, spotipy để tương tác với API của Spotify, và các thư viện của Keras để sử dụng mô hình VGG16.

**4.3.2 Khởi tạo thông tin xác thực cho Spotify**

| CLIENT\_ID = "YOUR\_CLIENT\_ID"  CLIENT\_SECRET = "YOUR\_CLIENT\_SECRET"  client\_credentials\_manager = SpotifyClientCredentials(client\_id=CLIENT\_ID, client\_secret=CLIENT\_SECRET)  sp = spotipy.Spotify(client\_credentials\_manager=client\_credentials\_manager) |
| --- |

Đoạn mã này khởi tạo thông tin xác thực cần thiết để kết nối với API của Spotify, cho phép chúng ta truy xuất thông tin về bài hát và album.

**4.3.3 Hàm trích xuất đặc trưng từ hình ảnh bìa album**

| def extract\_image\_features(img\_url):  model = VGG16(weights='imagenet', include\_top=False)  response = requests.get(img\_url)  img\_data = BytesIO(response.content)  img = image.load\_img(img\_data, target\_size=(224, 224))  x = image.img\_to\_array(img)  x = np.expand\_dims(x, axis=0)  x = preprocess\_input(x)  features = model.predict(x)  return features |
| --- |

Mục đích: Hàm này nhận URL của bìa album và trích xuất các đặc trưng hình ảnh bằng mô hình VGG16.

Quy trình: Tải ảnh từ URL, thay đổi kích thước về (224x224), tiền xử lý ảnh để phù hợp với mô hình, và cuối cùng sử dụng mô hình để trích xuất các đặc trưng.

**4.3.4 Hàm lấy URL bìa album từ API của Spotify**

| def get\_song\_album\_cover\_url(song\_name, artist\_name):  search\_query = f"track:{song\_name} artist:{artist\_name}"  results = sp.search(q=search\_query, type="track")  if results and results["tracks"]["items"]:  track = results["tracks"]["items"][0]  album\_cover\_url = track["album"]["images"][0]["url"]  track\_url = track['external\_urls']['spotify']  return album\_cover\_url, track\_url, track['id']  else:  return "https://i.postimg.cc/0QNxYz4V/social.png", None, None |
| --- |

Mục đích: Hàm này tìm kiếm bài hát theo tên và nghệ sĩ để lấy URL của bìa album và liên kết đến bài hát trên Spotify.

Quy trình: Tạo truy vấn tìm kiếm, nhận kết quả từ Spotify API và trả về URL bìa album cùng với liên kết đến bài hát.

**4.3.5 Hàm gợi ý bài hát**

| def recommend(song):  index = music[music['song'] == song].index[0]  distances = sorted(list(enumerate(similarity[index])), reverse=True, key=lambda x: x[1])  recommended\_music\_names = []  recommended\_music\_posters = []  recommended\_music\_links = []  recommended\_music\_ids = []  for i in distances[1:6]:  artist = music.iloc[i[0]].artist  album\_cover\_url, track\_url, track\_id = get\_song\_album\_cover\_url(music.iloc[i[0]].song, artist)  image\_features = extract\_image\_features(album\_cover\_url)  recommended\_music\_posters.append(album\_cover\_url)  recommended\_music\_names.append(music.iloc[i[0]].song)  recommended\_music\_links.append(track\_url)  recommended\_music\_ids.append(track\_id)  return recommended\_music\_names, recommended\_music\_posters, recommended\_music\_links, recommended\_music\_ids |
| --- |

Mục đích: Hàm này nhận một bài hát và gợi ý 5 bài hát tương tự dựa trên ma trận tương đồng đã được tính toán trước.

Quy trình: Tìm chỉ số bài hát trong DataFrame, sắp xếp các bài hát theo độ tương đồng, lấy bìa album và liên kết cho từng bài hát gợi ý.

**4.3.6 Ứng dụng gợi ý nhạc**

| st.header('Music Recommender System')  music = pickle.load(open('df.pkl', 'rb'))  similarity = pickle.load(open('similarity.pkl', 'rb'))  search\_term = st.text\_input("Search for the music genre you want", "")  if search\_term:  filtered\_music = music[music['song'].str.contains(search\_term, case=False)]  music\_list = filtered\_music['song'].values  else:  music\_list = music['song'].values  selected\_song = st.selectbox("Select a song", music\_list)  if st.button('Show Recommendations'):  recommended\_music\_names, recommended\_music\_posters, recommended\_music\_links, recommended\_music\_ids = recommend(selected\_song)  cols = st.columns(5)  for i in range(5):  with cols[i]:  st.image(recommended\_music\_posters[i], width=100)  st.text(recommended\_music\_names[i])  st.write(f"[Play on Spotify]({recommended\_music\_links[i]})") |
| --- |

Mục đích: Tạo giao diện người dùng cho ứng dụng gợi ý nhạc sử dụng Streamlit.

Quy trình: Tải dữ liệu bài hát và ma trận tương đồng, cho phép người dùng tìm kiếm và chọn bài hát, sau đó hiển thị các bài hát gợi ý.

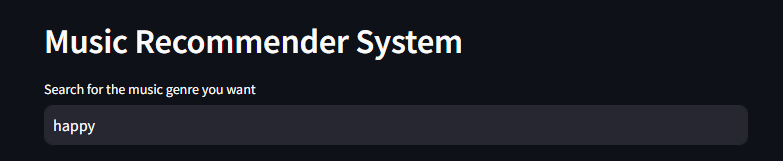
**4.3.7 Tích hợp Spotify Web Playback SDK**

| if 'track\_id' in st.session\_state:  track\_id = st.session\_state.track\_id  st.markdown(f"""  <script src="https://sdk.scdn.co/spotify-player.js"></script>  <script>  window.onSpotifyWebPlaybackSDKReady = () => {{  const token = 'YOUR\_ACCESS\_TOKEN'; // Replace with your actual token  const player = new Spotify.Player({{  name: 'Spotify Player',  getOAuthToken: cb => {{ cb(token); }},  volume: 0.5  }});  player.connect().then(success => {{  if (success) {{  console.log('Player connected to Spotify!');  }}  }});  player.addListener('ready', data => {{  player.resume().then(() => {{  player.queue.add(track\_id);  player.play();  }});  }});  }};  </script>  """, unsafe\_allow\_html=True) |
| --- |

Mục đích: Tích hợp chức năng phát nhạc trực tiếp trên ứng dụng bằng Spotify Web Playback SDK.

Quy trình: Kiểm tra xem track\_id có trong trạng thái phiên không, nếu có thì khởi tạo trình phát Spotify và phát bài hát đã chọn.

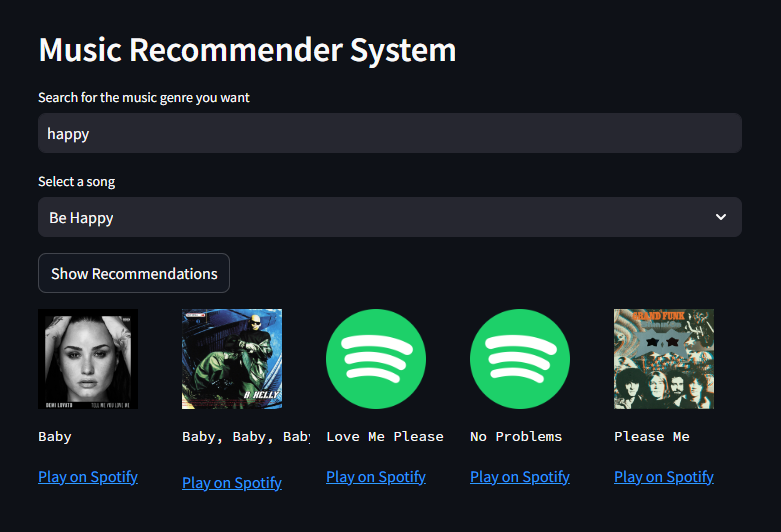
**4.3.8. Hình Ảnh Demo**

****

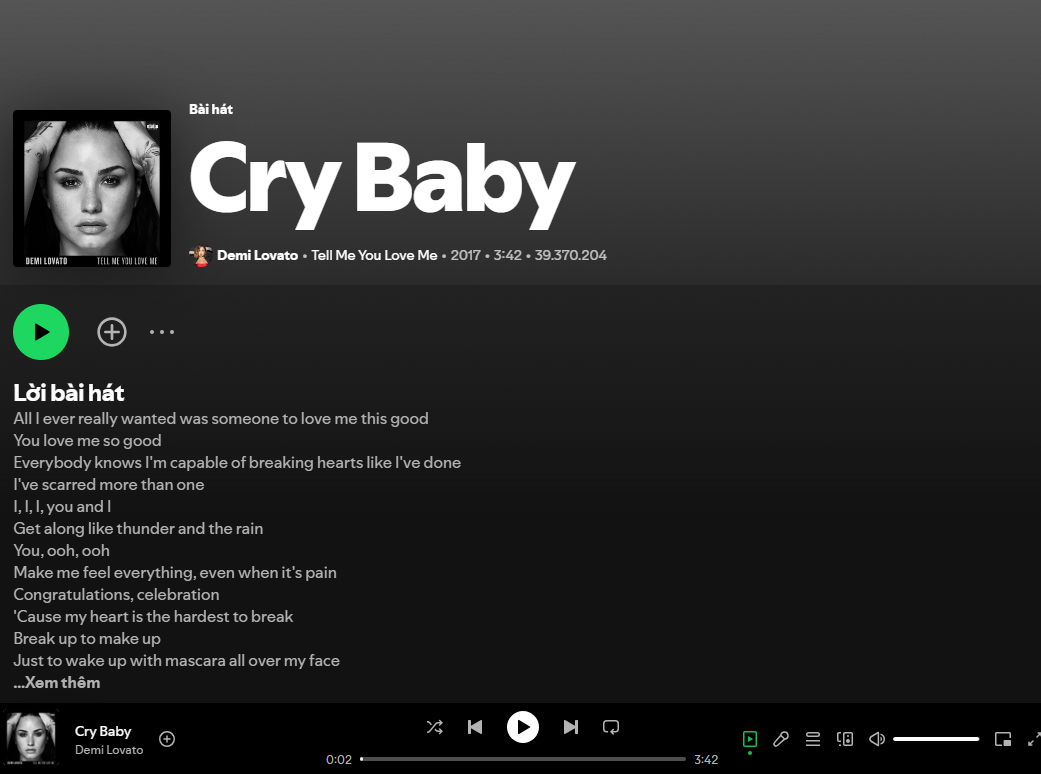
*Nhập một thể loại nhạc yêu thích*

**

*Hệ thống gợi ý các bài hát có thể loại tương tự trong danh sách bên dưới*

**

*Sau khi chọn “Show Recommendations” hệ thống sẽ đề xuất thêm 5 bài hát tương tự*

**

*Click vào “Play on Spotify” để nghe bài hát*

# PHẦN KẾT LUẬN

## 1. Kết quả đạt được

* Vận dụng được kiến thức về lập trình web, cơ sở dữ liệu và quy trình xây dựng website hoàn chỉnh với những chức năng cơ bản.
* Biết cách phân tích và xử lý dữ liệu bằng những thuật toán học máy.
* Xây dựng được hệ thống đề xuất bài hát cá nhân hóa trải nghiệm người dùng.

## 2. Hạn chế của đề tài

Vì hạn chế về mặt thời gian nên chưa xây dựng được website với đầy đủ chức năng như mục tiêu đặt ra. Mô hình đề xuất nhạc sẽ thiếu tính hiệu quả, khó khăn trong việc đưa ra gợi ý bài hát chính xác nếu đối tượng sử dụng là người dùng mới chưa có lịch sử nghe nhạc nhiều, nguồn dữ liệu ít, độ nhiễu cao.

## 3. Hướng phát triển

Phân tích sâu hơn vào lời bài hát và đặc điểm của âm thanh (tempo, âm sắc, nhịp điệu) để đưa ra các gợi ý nhạc không chỉ dựa trên thói quen của người nghe, mà còn dựa trên nội dung và cảm xúc, nguồn năng lượng mà bài hát truyền tải. Từ đó mang lại trải nghiệm thật sự thăng hoa về thanh âm và cảm xúc.

Tích hợp AI trợ lý ảo (như Alexa, Google Assistant) để tìm kiếm bài hát dựa trên giọng nói. Người dùng chỉ cần nói tên bài hát, lời bài hát, hoặc yêu cầu tìm một bản nhạc có đặc điểm cụ thể, AI sẽ tự động tìm kiếm và phát nhạc.

# TÀI LIỆU THAM KHẢO

[1]Vũ Hữu Tiệp (2020), *“Machine Learning cơ bản”*

[2] Spotify Web API, <https://developer.spotify.com/documentation/web-api>

[3] Streamlit Documentation, <https://docs.streamlit.io/>