TRƯỜNG ĐẠI HỌC ĐẠI NAM

**KHOA CÔNG NGHỆ THÔNG TIN**

Logo, company name

Description automatically generated

**BÀI TẬP LỚN**

**HỌC PHẦN: DỮ LIỆU LỚN**

**ĐỀ TÀI: DỰ ĐOÁN XU HƯỚNG PHIM**

**Giảng viên: Lê Thị Thùy Trang**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **TT** | **Mã sv** | **Họ và Tên** | **Ngày Sinh** | **Lớp** |
| **1** | **1671020062** | **Trần Trí Duy** |  |  |
| **2** | **1671020349** | **Nguyễn Công Uẩn** |  |  |
| **3** | **1671020152** | **Nguyễn Nam Hưng** |  |  |
| **4** | **1671020274** | **Mẫn Bá Sâm** |  |  |

**Hà Nội, năm 2025**

TRƯỜNG ĐẠI HỌC ĐẠI NAM

**KHOA CÔNG NGHỆ THÔNG TIN**

Logo, company name

Description automatically generated

**BÀI TẬP LỚN**

**HỌC PHẦN: DỮ LIỆU LỚN**

**ĐỀ TÀI: DỰ ĐOÁN XU HƯỚNG PHIM**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **TT** | **Mã sv** | **Họ và Tên** | **Ngày Sinh** | **Điểm** | |
| **Bằng Số** | **Bằng Chữ** |
| **1** | **1671020062** | **Trần Trí Duy** |  |  |  |
| **2** | **1671020349** | **Nguyễn Công Uẩn** |  |  |  |
| **3** | **1671020152** | **Nguyễn Nam Hưng** |  |  |  |
| **4** | **1671020274** | **Mẫn Bá Sâm** |  |  |  |

|  |  |
| --- | --- |
| **CÁN BỘ CHẤM THI 1**  **Lê Thị Thùy Trang** | **CÁN BỘ CHẤM THI 2** |

**Hà Nội, năm 2025**

# LỜI NÓI ĐẦU

Trong kỷ nguyên số hiện nay, dữ liệu lớn (Big Data) đã trở thành một yếu tố quyết định trong mọi lĩnh vực, từ kinh doanh, y tế đến giáo dục và giải trí. Ngành công nghiệp điện ảnh, với khối lượng dữ liệu khổng lồ từ các bộ phim, phản hồi từ khán giả và thông tin thị trường, đang đứng trước cơ hội lớn để tận dụng sức mạnh của dữ liệu lớn nhằm tối ưu hóa quy trình sản xuất và tiếp thị phim. Việc phân tích dữ liệu này không chỉ giúp hiểu rõ hơn về thị trường mà còn tạo ra những chiến lược hiệu quả hơn cho nhà sản xuất.

Đề tài "Dự đoán xu hướng phim" không chỉ nhằm phân tích các yếu tố ảnh hưởng đến sự thành công của một bộ phim mà còn sử dụng các phương pháp học máy để dự đoán xu hướng tiêu dùng trong ngành điện ảnh. Bằng việc khai thác dữ liệu từ các nguồn như Kaggle, chúng tôi sẽ áp dụng các kỹ thuật tiền xử lý, phân tích và mô hình hóa để tìm ra mối liên hệ giữa các đặc điểm của phim, như thể loại, doanh thu, và phản hồi từ khán giả, với kết quả kinh doanh. Điều này sẽ giúp nắm bắt được xu hướng và sở thích của khán giả một cách chính xác hơn.

Nghiên cứu này không chỉ góp phần nâng cao hiểu biết về hành vi của người tiêu dùng mà còn tạo ra những thông tin hữu ích cho các nhà sản xuất và nhà phân phối phim trong việc định hướng chiến lược phát triển sản phẩm. Hy vọng rằng nghiên cứu này sẽ mở ra những hướng đi mới cho việc ứng dụng dữ liệu lớn vào ngành công nghiệp điện ảnh, đồng thời cung cấp cái nhìn sâu sắc về xu hướng tiêu dùng trong lĩnh vực giải trí.

# MỤC LỤC

[LỜI NÓI ĐẦU 3](#_Toc192970765)

[MỤC LỤC 4](#_Toc192970766)

[CHƯƠNG 1. GIỚI THIỆU 9](#_Toc192970767)

[CHƯƠNG 2: MỤC TIÊU DỰ ÁN 10](#_Toc192970768)

[2.1. Tổng Quan và Ý Nghĩa Dự Án 10](#_Toc192970769)

[2.2. Mục Tiêu Tổng Quát 10](#_Toc192970770)

[2.3. Mục Tiêu Cụ Thể 10](#_Toc192970771)

[2.3.1. Tiền Xử Lý Dữ Liệu 10](#_Toc192970772)

[2.3.2. Tạo Đặc Trưng và Phân Tích Dữ Liệu 11](#_Toc192970773)

[2.3.3. Xây Dựng và Đánh Giá Mô Hình 11](#_Toc192970774)

[2.3.4. Ứng Dụng và Triển Khai 11](#_Toc192970775)

[2.4. Đóng Góp của Dự Án 11](#_Toc192970776)

[CHƯƠNG 3: TỔNG QUAN DỮ LIỆU VÀ XỬ LÝ DỮ LIỆU BAN ĐẦU 13](#_Toc192970777)

[3.1. Giới thiệu về dữ liệu 13](#_Toc192970778)

[3.2. Đánh giá chất lượng dữ liệu 13](#_Toc192970779)

[3.3. Quy trình xử lý dữ liệu ban đầu 14](#_Toc192970780)

[3.3.5. Tóm tắt và nhận xét 16](#_Toc192970790)

[CHƯƠNG 4: TẠO BIẾN TARGET ‘TREND’ 17](#_Toc192970791)

[4.1. Mục đích của biến 'trend' 17](#_Toc192970792)

[4.2. Lý do chọn 'profit' làm cơ sở để xác định 'trend' 17](#_Toc192970793)

[4.3. Quy trình tạo biến 'trend' 17](#_Toc192970794)

[4.4. Ưu điểm của phương pháp này 19](#_Toc192970795)

[4.5. Nhược điểm và cách khắc phục 20](#_Toc192970796)

[4.6. Kết quả cuối cùng 20](#_Toc192970797)

[CHƯƠNG 5: XỬ LÝ CỘT GENRE 22](#_Toc192970799)

[5.1. Xử lý cột genre 22](#_Toc192970800)

[5.2. Tóm tắt logic xử lý cột genre 26](#_Toc192970804)

[5.3. Kết quả cuối cùng 26](#_Toc192970805)

[CHƯƠNG 6: OVERSAMPLING LỚP MINORITY 28](#_Toc192970806)

[6.1. Mô tả vấn đề mất cân bằng dữ liệu 28](#_Toc192970807)

[6.1.1. Tác động của dữ liệu mất cân bằng 28](#_Toc192970808)

[6.1.2.Lý do cần xử lý mất cân bằng 28](#_Toc192970809)

[6.2. Phương pháp Oversampling 29](#_Toc192970810)

[6.2.1. Lý do chọn Random Oversampling 29](#_Toc192970811)

[6.3. Các bước thực hiện Oversampling 29](#_Toc192970812)

[6.4. Ưu và nhược điểm của Random Oversampling 31](#_Toc192970818)

[6.4.1.Ưu điểm 31](#_Toc192970819)

[6.4.2. Nhược điểm 32](#_Toc192970820)

[6.5. Kết quả sau Oversampling 32](#_Toc192970821)

[6.6. Các phương pháp thay thế để xử lý mất cân bằng 33](#_Toc192970822)

[6.7. Đánh giá hiệu quả của Oversampling 33](#_Toc192970823)

[6.8. Kết luận 34](#_Toc192970824)

[CHƯƠNG 7: HUẤN LUYỆN VÀ ĐÁNH GIÁ MÔ HÌNH 35](#_Toc192970825)

[7.1 Lựa chọn mô hình học máy 35](#_Toc192970826)

[7.1.1 Mô hình được chọn 35](#_Toc192970827)

[7.1.2 Lý do lựa chọn 35](#_Toc192970828)

[7.1.3 Tham số cấu hình 35](#_Toc192970829)

[7.2 Chuẩn bị dữ liệu cho mô hình 36](#_Toc192970830)

[7.2.1 Kết hợp đặc trưng bằng VectorAssembler 36](#_Toc192970831)

[7.2.2 Chia dữ liệu thành tập huấn luyện và tập kiểm tra 36](#_Toc192970833)

[7.3 Oversampling lớp thiểu số 37](#_Toc192970834)

[7.3.1 Vấn đề dữ liệu mất cân bằng 37](#_Toc192970835)

[7.3.2 Giải pháp: Oversampling 37](#_Toc192970836)

[7.3.3 Lý do chọn oversampling thay vì undersampling 38](#_Toc192970838)

[7.4 Huấn luyện mô hình 38](#_Toc192970839)

[7.4.1 Khởi tạo mô hình 38](#_Toc192970840)

[7.4.2 Huấn luyện 38](#_Toc192970842)

[7.5 Dự đoán trên tập kiểm tra 38](#_Toc192970843)

[7.6 Đánh giá mô hình 39](#_Toc192970844)

[7.6.1 Độ chính xác (Accuracy) 39](#_Toc192970845)

[7.6.2 Ma trận nhầm lẫn (Confusion Matrix) 39](#_Toc192970847)

[7.6.3 Báo cáo phân loại (Classification Report) 40](#_Toc192970850)

[7.6.4 Đường cong ROC và AUC 40](#_Toc192970851)

[7.6.5. Mô hình Feature Importance 42](#_Toc192970854)

[CHƯƠNG 8: PHÂN TÍCH TẦM QUAN TRỌNG CỦA CÁC ĐẶC TRƯNG 43](#_Toc192970857)

[8.1. Giới thiệu về tầm quan trọng của đặc trưng 43](#_Toc192970858)

[8.2. Cách tính tầm quan trọng của đặc trưng trong Random Forest 43](#_Toc192970859)

[8.3. Trích xuất và trực quan hóa tầm quan trọng của đặc trưng 44](#_Toc192970860)

[8.4. Phân tích kết quả chi tiết 44](#_Toc192970861)

[8.4.1. Các đặc trưng quan trọng nhất 45](#_Toc192970862)

[8.4.2. Các đặc trưng thể loại (genres) 45](#_Toc192970863)

[8.4.3. Các đặc trưng thời gian 45](#_Toc192970864)

[8.4.4. Các đặc trưng ít quan trọng 46](#_Toc192970865)

[8.5. Ý nghĩa thực tiễn và ứng dụng của phân tích 46](#_Toc192970866)

[8.6. Hạn chế của phân tích tầm quan trọng 46](#_Toc192970867)

[8.7. Giải pháp khắc phục hạn chế 47](#_Toc192970868)

[CHƯƠNG 9: KHÁM PHÁ DỮ LIỆU (EDA) 48](#_Toc192970869)

[9.1 Phân phối của các biến số 48](#_Toc192970870)

[9.1.1 Biến vote\_average (Điểm đánh giá trung bình) 48](#_Toc192970871)

[9.1.2 Biến vote\_count (Số lượng đánh giá) 49](#_Toc192970874)

[9.2 Phân tích các thể loại phim (genres) 49](#_Toc192970875)

[9.2.1 Số lượng phim theo thể loại 49](#_Toc192970876)

[9.3 Mối quan hệ giữa các biến 51](#_Toc192970879)

[9.3.1 Mối quan hệ giữa vote\_count và vote\_average 51](#_Toc192970880)

[9.6 Kết quả và nhận xét từ EDA 52](#_Toc192970882)

[9.7 Tác động của EDA đến dự án 52](#_Toc192970883)

[CHƯƠNG 10: KẾT LUẬN 53](#_Toc192970884)

# MỤC LỤC HÌNH ẢNH

[CHƯƠNG 3: TỔNG QUAN DỮ LIỆU VÀ XỬ LÝ DỮ LIỆU BAN ĐẦU 13](#_Toc192970777)

[Hình 3.1: Code đọc dữ liệu 14](#_Toc192970781)

[Hình 3.2: Kết quả đọc dữ liệu 14](#_Toc192970782)

[Hình 3.3: Code loại bỏ các cột không liên quan 14](#_Toc192970783)

[Hình 3.4: Kết quả sau khi loại bỏ các cột không liên quan 14](#_Toc192970784)

[Hình 3.5: Code chuyển đổi dữ liệu 15](#_Toc192970785)

[Hình 3.6: Code xử lý dữ liệu trùng lặp 15](#_Toc192970786)

[Hình 3.7: Kết quả Số lượng dòng trùng lặp được ghi lại và loại bỏ 15](#_Toc192970787)

[Hình 3.8: Code xử lý giá trị thiếu 16](#_Toc192970788)

[Hình 3.8: Code tạo các đặc trưng mới 16](#_Toc192970789)

[CHƯƠNG 4: TẠO BIẾN TARGET ‘TREND’ 17](#_Toc192970791)

[Hình 4.1: Kết quả thông tin sau DataFrame sau xử lý 21](file:///C:\CONG%20NGHE%20THONG%20TIN\Dữ%20Liệu%20Lớn\BTL_BigData\Nhom7BIGDATA.docx#_Toc192970798)

[CHƯƠNG 5: XỬ LÝ CỘT GENRE 22](file:///C:\CONG%20NGHE%20THONG%20TIN\Dữ%20Liệu%20Lớn\BTL_BigData\Nhom7BIGDATA.docx#_Toc192970799)

[Hình 5.1: Code Tách chuỗi genres 22](file:///C:\CONG%20NGHE%20THONG%20TIN\Dữ%20Liệu%20Lớn\BTL_BigData\Nhom7BIGDATA.docx#_Toc192970801)

[Hình 5.2: Code xử lý cột genres (tách chuỗi) 23](file:///C:\CONG%20NGHE%20THONG%20TIN\Dữ%20Liệu%20Lớn\BTL_BigData\Nhom7BIGDATA.docx#_Toc192970802)

[Hình 5.3: Code xử lý cột genres (gom các cột genre hiếm) 25](file:///C:\CONG%20NGHE%20THONG%20TIN\Dữ%20Liệu%20Lớn\BTL_BigData\Nhom7BIGDATA.docx#_Toc192970803)

[CHƯƠNG 6: OVERSAMPLING LỚP MINORITY 28](#_Toc192970806)

[Hình 6.1: Code ví dụ (sử dụng PySpark) 30](file:///C:\CONG%20NGHE%20THONG%20TIN\Dữ%20Liệu%20Lớn\BTL_BigData\Nhom7BIGDATA.docx#_Toc192970813)

[HÌnh 6.2 : Code kiểm tra 30](file:///C:\CONG%20NGHE%20THONG%20TIN\Dữ%20Liệu%20Lớn\BTL_BigData\Nhom7BIGDATA.docx#_Toc192970814)

[Hình 6.3: Code Tính tỷ lệ oversampling 30](file:///C:\CONG%20NGHE%20THONG%20TIN\Dữ%20Liệu%20Lớn\BTL_BigData\Nhom7BIGDATA.docx#_Toc192970815)

[Hình 6.4 : Code Thực hiện oversampling 31](file:///C:\CONG%20NGHE%20THONG%20TIN\Dữ%20Liệu%20Lớn\BTL_BigData\Nhom7BIGDATA.docx#_Toc192970816)

[Hình 6.5 : Code Kết hợp lại tập huấn luyện 31](file:///C:\CONG%20NGHE%20THONG%20TIN\Dữ%20Liệu%20Lớn\BTL_BigData\Nhom7BIGDATA.docx#_Toc192970817)

[CHƯƠNG 7: HUẤN LUYỆN VÀ ĐÁNH GIÁ MÔ HÌNH 35](#_Toc192970825)

[Hình 7.1: Code tạo vector features 36](file:///C:\CONG%20NGHE%20THONG%20TIN\Dữ%20Liệu%20Lớn\BTL_BigData\Nhom7BIGDATA.docx#_Toc192970832)

[Hình 7.2 : Code Oversampling 37](file:///C:\CONG%20NGHE%20THONG%20TIN\Dữ%20Liệu%20Lớn\BTL_BigData\Nhom7BIGDATA.docx#_Toc192970837)

[Hình 7.3 : Code huấn luyện đánh giá 38](file:///C:\CONG%20NGHE%20THONG%20TIN\Dữ%20Liệu%20Lớn\BTL_BigData\Nhom7BIGDATA.docx#_Toc192970841)

[Hình 7.4 : Code độ chính xác (Accuracy) 39](file:///C:\CONG%20NGHE%20THONG%20TIN\Dữ%20Liệu%20Lớn\BTL_BigData\Nhom7BIGDATA.docx#_Toc192970846)

[Hình 7.5: Code quy trình ma trận Confusion Matrix 39](file:///C:\CONG%20NGHE%20THONG%20TIN\Dữ%20Liệu%20Lớn\BTL_BigData\Nhom7BIGDATA.docx#_Toc192970848)

[Hình 7.6: Confusion Matrix 40](file:///C:\CONG%20NGHE%20THONG%20TIN\Dữ%20Liệu%20Lớn\BTL_BigData\Nhom7BIGDATA.docx#_Toc192970849)

[Hình 7.7: Code tạo mô hình ROC 41](file:///C:\CONG%20NGHE%20THONG%20TIN\Dữ%20Liệu%20Lớn\BTL_BigData\Nhom7BIGDATA.docx#_Toc192970852)

[Hình 7.8: Mô hình ROC Curve 41](file:///C:\CONG%20NGHE%20THONG%20TIN\Dữ%20Liệu%20Lớn\BTL_BigData\Nhom7BIGDATA.docx#_Toc192970853)

[Hình 7.9: Code Mô hình Feature Importance 42](file:///C:\CONG%20NGHE%20THONG%20TIN\Dữ%20Liệu%20Lớn\BTL_BigData\Nhom7BIGDATA.docx#_Toc192970855)

[Hình 7.10: Feature Importance 42](file:///C:\CONG%20NGHE%20THONG%20TIN\Dữ%20Liệu%20Lớn\BTL_BigData\Nhom7BIGDATA.docx#_Toc192970856)

[CHƯƠNG 8: PHÂN TÍCH TẦM QUAN TRỌNG CỦA CÁC ĐẶC TRƯNG 43](#_Toc192970857)

[CHƯƠNG 9: KHÁM PHÁ DỮ LIỆU (EDA) 48](#_Toc192970869)

[Hình 9.1 : Code vẽ biểu đồ vote\_average 48](#_Toc192970872)

[Hình 9.2: Biểu đồ phân phối của vote\_average 48](#_Toc192970873)

[Hình 9.3: Code tạo biểu dồ số lượng phim thể loại 50](#_Toc192970877)

[Hình 9.4: Số lượng phim theo thể loại 50](#_Toc192970878)

[Hình 9.5: Biểu đồ mối quan hệ giữa vote\_count và vote\_average 51](#_Toc192970881)

# CHƯƠNG 1. GIỚI THIỆU

Trong bối cảnh ngành công nghiệp điện ảnh phát triển mạnh mẽ, việc dự đoán xu hướng phim giúp các nhà sản xuất, phân phối và đầu tư đưa ra quyết định chiến lược chính xác. Dự án này khai thác dữ liệu lớn từ TMDB để phân tích các yếu tố ảnh hưởng đến thành công của phim và xây dựng mô hình dự báo xu hướng dựa trên Spark MLlib.

Bằng cách xử lý dữ liệu, tạo đặc trưng như lợi nhuận, thể loại phim (one-hot encoding) và áp dụng kỹ thuật oversampling để cân bằng dữ liệu, chúng tôi xây dựng mô hình Random Forest nhằm dự đoán xu hướng phim. Quá trình đánh giá mô hình dựa trên các chỉ số như Confusion Matrix, ROC Curve và Feature Importance.

Việc kết hợp Pandas và Spark giúp tối ưu hóa khả năng xử lý dữ liệu lớn, đồng thời việc trực quan hóa bằng biểu đồ EDA giúp phân tích sâu hơn về xu hướng phim. Dự án này không chỉ chứng minh khả năng ứng dụng học máy trong điện ảnh mà còn cung cấp thông tin quan trọng để hỗ trợ các chiến lược đầu tư và sản xuất phim hiệu quả.

# CHƯƠNG 2: MỤC TIÊU DỰ ÁN

## 2.1. Tổng Quan và Ý Nghĩa Dự Án

Trong bối cảnh ngành công nghiệp điện ảnh ngày càng phát triển mạnh mẽ, việc phân tích và dự đoán xu hướng phim đóng vai trò quan trọng không chỉ trong việc giúp các nhà sản xuất, phân phối có cái nhìn tổng quan về thị trường mà còn hỗ trợ các chiến lược marketing, định hướng sản xuất, và đầu tư hiệu quả. Dự án “Dự đoán xu hướng phim” được xây dựng nhằm khai thác dữ liệu lớn (Big Data) từ các nguồn thông tin phim để phân tích, xử lý và xây dựng mô hình học máy từ đó đưa ra những nhận định khách quan về xu hướng của các bộ phim trong tương lai.

Ý nghĩa của dự án không chỉ nằm ở khả năng phân tích dự đoán một cách tự động mà còn góp phần mở rộng kiến thức và ứng dụng của các công nghệ xử lý dữ liệu lớn, học máy và trực quan hóa dữ liệu vào lĩnh vực điện ảnh. Qua đó, dự án hướng đến việc tạo ra một công cụ hỗ trợ ra quyết định dựa trên dữ liệu, từ đó giúp các bên liên quan có những chiến lược phù hợp để tận dụng tối đa cơ hội của thị trường đầy biến động này.

## 2.2. Mục Tiêu Tổng Quát

Mục tiêu tổng quát của dự án là xây dựng một hệ thống dự đoán xu hướng phim dựa trên dữ liệu thực tế được thu thập từ các nguồn khác nhau. Hệ thống sẽ thực hiện các bước tiền xử lý dữ liệu, tạo đặc trưng và xây dựng mô hình dự đoán sử dụng thuật toán học máy (đặc biệt là Random Forest trong Spark MLlib). Qua đó, dự án mong muốn:

* **Xác định các yếu tố then chốt:** Tìm ra các đặc trưng quan trọng ảnh hưởng đến xu hướng thành công của phim như lợi nhuận, số lượng bình chọn, điểm đánh giá và thông tin thể loại phim.
* **Dự đoán xu hướng:** Phân loại các bộ phim thành nhóm “xu hướng tăng” và “xu hướng giảm” dựa trên các chỉ số đã xử lý và các đặc trưng được trích xuất.
* **Hỗ trợ ra quyết định:** Cung cấp thông tin dự báo khách quan giúp các nhà sản xuất, phân phối và các nhà đầu tư có thể cân nhắc và đưa ra các quyết định chiến lược phù hợp.

## 2.3. Mục Tiêu Cụ Thể

Để đạt được mục tiêu tổng quát, dự án được triển khai theo một chuỗi các bước xử lý và phân tích dữ liệu, với các mục tiêu cụ thể như sau:

### 2.3.1. Tiền Xử Lý Dữ Liệu

* **Thu thập và làm sạch dữ liệu:** Sử dụng Python để đọc dữ liệu từ file CSV chứa thông tin phim, tiến hành loại bỏ các cột không liên quan, xử lý dữ liệu trùng lặp và chuyển đổi kiểu dữ liệu phù hợp (ví dụ chuyển đổi cột ngày phát hành sang định dạng datetime, các cột số sang định dạng numeric).
* **Xử lý giá trị thiếu:** Áp dụng phương pháp imputation cho các cột số bằng giá trị trung vị nhằm đảm bảo tính liên tục của dữ liệu. Đối với cột “release\_date”, các dòng thiếu dữ liệu sẽ được loại bỏ để đảm bảo chất lượng dữ liệu ngày tháng.

### 2.3.2. Tạo Đặc Trưng và Phân Tích Dữ Liệu

* **Trích xuất đặc trưng:** Tạo ra các đặc trưng mới từ dữ liệu gốc như “năm”, “tháng” từ cột “release\_date”, tính toán “lợi nhuận” từ hiệu số giữa doanh thu và chi phí, và xử lý cột “genres” để tạo các biến nhị phân biểu diễn sự hiện diện của từng thể loại phim.
* **Phân tích khám phá dữ liệu (EDA):** Sử dụng các biểu đồ trực quan như histogram, scatter plot, bar chart để phân tích sự phân phối và mối quan hệ giữa các biến. Phân tích này giúp làm rõ những xu hướng, đặc điểm và mối liên hệ giữa các yếu tố ảnh hưởng đến thành công của phim.

### 2.3.3. Xây Dựng và Đánh Giá Mô Hình

* **Thiết lập biến mục tiêu:** Sử dụng biến “profit” để tính toán giá trị trung vị, từ đó xác định nhãn “trend” cho từng bộ phim (1: xu hướng tăng, 0: xu hướng giảm).
* **Tiền xử lý và cân bằng dữ liệu:** Loại bỏ các cột gây rò rỉ thông tin và áp dụng kỹ thuật oversampling cho lớp minority nhằm cân bằng dữ liệu, đảm bảo mô hình học máy không bị lệch về phía lớp chiếm ưu thế.
* **Huấn luyện mô hình học máy:** Sử dụng thuật toán Random Forest trên nền tảng Spark MLlib để xây dựng mô hình dự đoán. Quá trình này bao gồm việc tạo vector đặc trưng, chia tập dữ liệu thành tập huấn luyện và tập kiểm tra, sau đó đánh giá mô hình qua các chỉ số như Accuracy, Confusion Matrix và ROC Curve.
* **Đánh giá và tối ưu hóa:** Phân tích kết quả mô hình thông qua việc trực quan hóa các chỉ số đánh giá và xác định các đặc trưng quan trọng (Feature Importance) nhằm từ đó có thể cải tiến mô hình trong các giai đoạn sau.

### 2.3.4. Ứng Dụng và Triển Khai

* **Phân tích dự báo:** Cung cấp báo cáo trực quan kết hợp với các biểu đồ, số liệu thống kê để dễ dàng theo dõi xu hướng và nhận diện các mẫu hình thị trường phim.
* **Đưa ra khuyến nghị:** Dựa trên kết quả dự đoán, hệ thống sẽ đưa ra các khuyến nghị cụ thể giúp các nhà sản xuất và phân phối phim có thể điều chỉnh chiến lược kinh doanh, marketing và đầu tư phù hợp với xu hướng hiện tại của thị trường.
* **Tích hợp và triển khai:** Hướng đến việc xây dựng một giao diện thân thiện cho người dùng cuối, từ đó tích hợp hệ thống vào quá trình ra quyết định của các doanh nghiệp liên quan trong ngành điện ảnh.

## 2.4. Đóng Góp của Dự Án

Dự án “Dự đoán xu hướng phim” mang lại nhiều giá trị thiết thực như:

* **Cải thiện quy trình ra quyết định:** Nhờ vào hệ thống dự đoán khách quan dựa trên dữ liệu, các bên liên quan có thể dễ dàng nắm bắt xu hướng thị trường, từ đó tối ưu hóa chiến lược sản xuất và quảng bá phim.
* **Nâng cao hiệu quả kinh doanh:** Việc dự đoán xu hướng một cách chính xác giúp giảm thiểu rủi ro đầu tư, đồng thời tăng khả năng thành công của các sản phẩm phim.
* **Đóng góp cho nghiên cứu khoa học:** Kết quả của dự án sẽ cung cấp một khung tham chiếu và phương pháp tiếp cận mới trong việc xử lý dữ liệu lớn và ứng dụng học máy trong lĩnh vực giải trí, mở ra hướng nghiên cứu và ứng dụng trong các lĩnh vực khác.

# CHƯƠNG 3: TỔNG QUAN DỮ LIỆU VÀ XỬ LÝ DỮ LIỆU BAN ĐẦU

## 3.1. Giới thiệu về dữ liệu

Trong đề tài “Dự đoán xu hướng phim”, dữ liệu được thu thập từ nhiều nguồn liên quan đến thông tin của các bộ phim như doanh thu, chi phí, đánh giá, ngày phát hành, và thông tin thể loại. Tập dữ liệu ban đầu được lưu trữ dưới định dạng CSV, bao gồm hàng chục cột thông tin với số lượng dòng dữ liệu khá lớn. Mục đích chính của việc thu thập dữ liệu là tạo ra một cơ sở dữ liệu đồng nhất, đầy đủ và chính xác nhằm phục vụ cho quá trình phân tích và xây dựng mô hình dự đoán xu hướng phim.

Dữ liệu thu thập ban đầu bao gồm các thông tin sau:

* **Thông tin nhận dạng và mô tả phim:** Gồm các trường như id, title, overview, tagline, homepage… Đây là các thông tin mô tả bản chất của phim nhưng không được sử dụng trực tiếp trong quá trình dự đoán.
* **Thông tin tài chính:** Bao gồm các trường như doanh thu (revenue), chi phí (budget) và lợi nhuận (profit – được tính toán sau). Đây là các thông tin quan trọng để đánh giá khả năng sinh lời của bộ phim.
* **Thông tin đánh giá và phổ biến:** Các trường vote\_average, vote\_count, popularity… cho biết mức độ yêu thích và tần suất được bình chọn của phim.
* **Thông tin thời gian:** Trường release\_date được lưu dưới dạng chuỗi ban đầu, sau đó chuyển đổi sang định dạng datetime để trích xuất các đặc trưng về năm và tháng.
* **Thông tin thể loại:** Trường genres cung cấp danh sách các thể loại phim dưới dạng chuỗi, với các thể loại được phân cách nhau bằng dấu phẩy.

Những thông tin này không chỉ giúp đánh giá được tiềm năng thành công của phim mà còn đóng vai trò là cơ sở để tạo ra các đặc trưng mới phục vụ cho quá trình xây dựng mô hình học máy.

## 3.2. Đánh giá chất lượng dữ liệu

Khi tiếp cận tập dữ liệu ban đầu, việc đánh giá chất lượng dữ liệu là bước không thể bỏ qua. Trong quá trình kiểm tra, các vấn đề chính được phát hiện bao gồm:

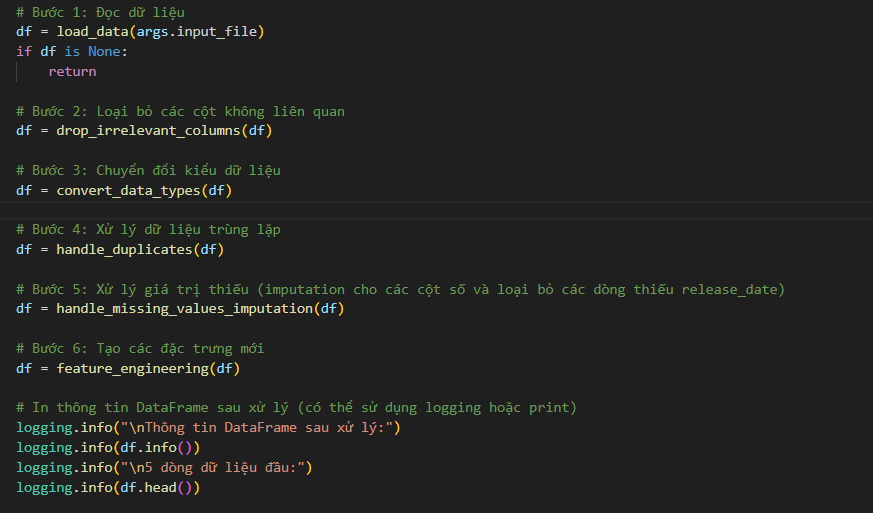
* **Thiếu dữ liệu:** Một số cột số như revenue, budget, vote\_average, vote\_count, runtime, popularity có giá trị bị thiếu. Điều này ảnh hưởng trực tiếp đến khả năng phân tích và tính toán các chỉ số như trung vị, trung bình hay tỷ lệ phần trăm.
* **Dữ liệu trùng lặp:** Có một số dòng dữ liệu xuất hiện nhiều lần do lỗi nhập liệu hoặc do quá trình thu thập dữ liệu từ nhiều nguồn.
* **Định dạng dữ liệu không đồng nhất:** Các trường như release\_date ban đầu ở dạng chuỗi (string) và cần được chuyển đổi sang định dạng datetime để có thể xử lý và phân tích thời gian chính xác.
* **Các cột không liên quan:** Tập dữ liệu ban đầu chứa nhiều cột chỉ mang tính chất mô tả thông tin và không ảnh hưởng trực tiếp đến việc dự đoán xu hướng phim (như các trường về mô tả, đường dẫn hình ảnh…).

## 3.3. Quy trình xử lý dữ liệu ban đầu

Để đảm bảo dữ liệu được sử dụng cho quá trình xây dựng mô hình đạt chất lượng cao, các bước xử lý dữ liệu ban đầu được thực hiện một cách có hệ thống như sau:

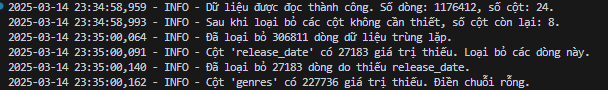
***Bước 1: Đọc dữ liệu***

* **Mô tả:** Đọc file CSV bằng pandas và hiển thị thông tin cơ bản (số dòng, số cột).
* **Code:**



##### Hình 3.1: Code đọc dữ liệu

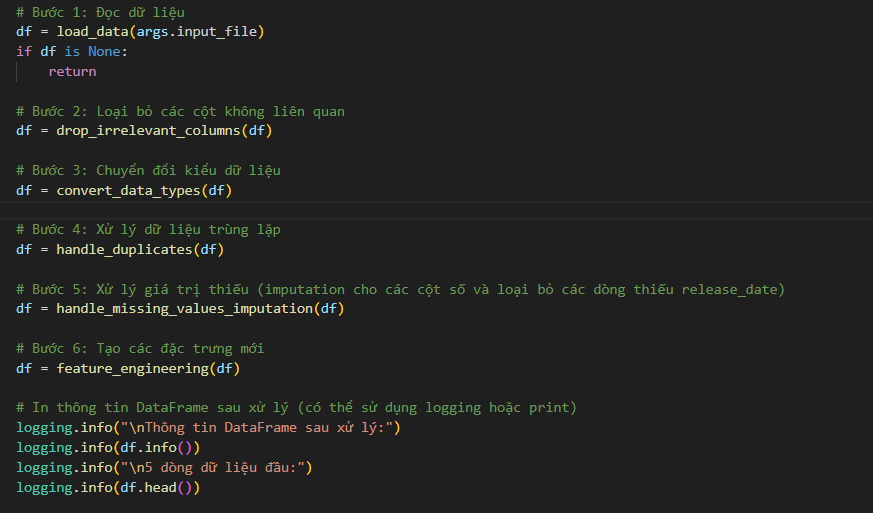
* **Kết quả:** Tập dữ liệu được tải thành công với thông tin số dòng và số cột được ghi lại qua logging.



##### Hình 3.2: Kết quả đọc dữ liệu

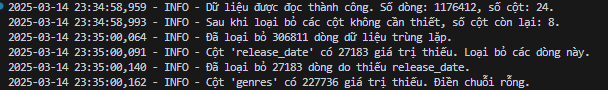
***Bước 2: Loại bỏ các cột không cần thiết***

* **Mô tả:** Loại bỏ các cột không liên quan đến việc dự đoán xu hướng như id, title, status, adult, backdrop\_path, homepage, imdb\_id, original\_language, original\_title, overview, poster\_path, tagline, production\_companies, production\_countries, spoken\_languages, keywords.
* **Code:**



##### Hình 3.3: Code loại bỏ các cột không liên quan

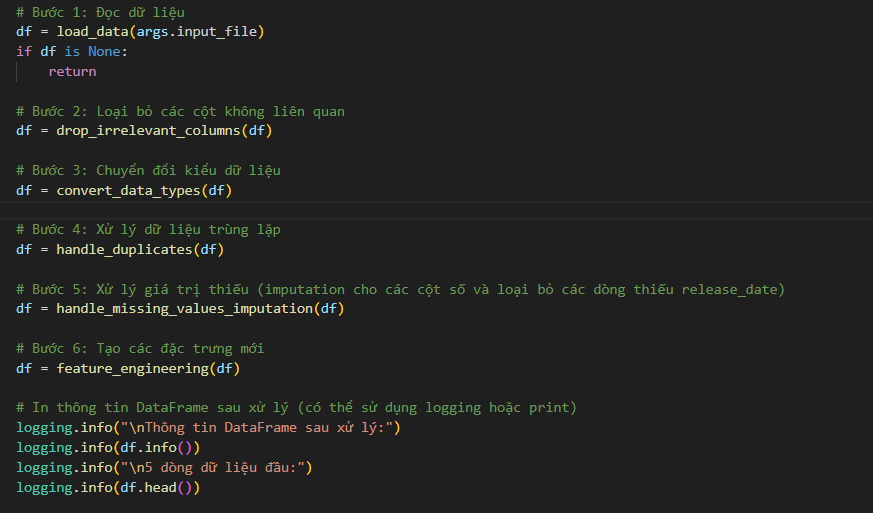
* **Lý do:** Các cột này không cung cấp thông tin định lượng hoặc phân loại trực tiếp liên quan đến xu hướng phim.
* **Kết quả:**



##### Hình 3.4: Kết quả sau khi loại bỏ các cột không liên quan

***Bước 3: Chuyển đổi kiểu dữ liệu***

* **Mô tả:**
  + Chuyển cột release\_date sang kiểu datetime.
  + Chuyển các cột số như revenue, budget, vote\_average, vote\_count, runtime, popularity sang kiểu numeric.
* **Code:**

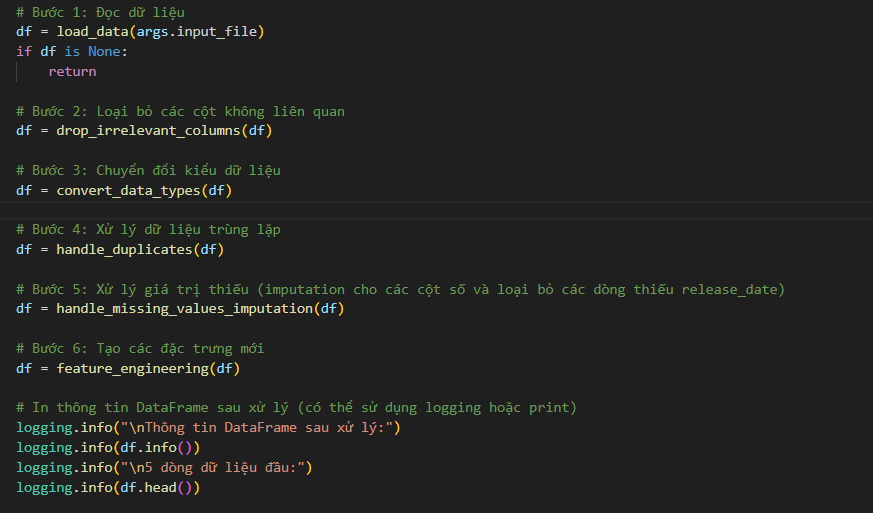


##### Hình 3.5: Code chuyển đổi dữ liệu

* **Kết quả:** Đảm bảo dữ liệu có định dạng phù hợp cho các phép tính và phân tích sau này.

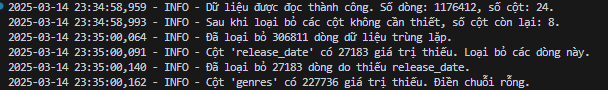
***Bước 4: Xử lý dữ liệu trùng lặp***

* **Mô tả:** Loại bỏ các dòng trùng lặp để đảm bảo tính duy nhất của mỗi bản ghi.
* **Code:** Hàm handle\_duplicates(df).



##### Hình 3.6: Code xử lý dữ liệu trùng lặp

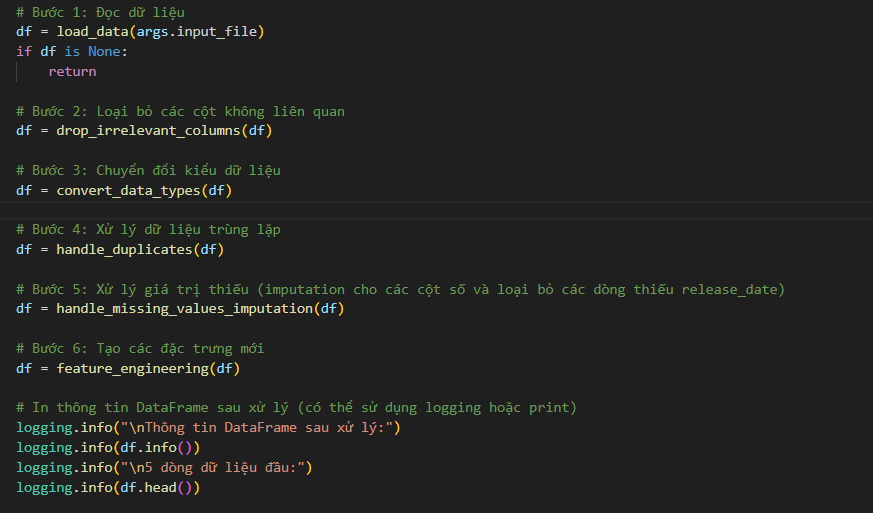
* **Kết quả:** Số lượng dòng trùng lặp được ghi lại và loại bỏ.



##### Hình 3.7: Kết quả Số lượng dòng trùng lặp được ghi lại và loại bỏ

***Bước 5: Xử lý giá trị thiếu***

* **Mô tả:**
  + Đối với các cột số (revenue, budget, vote\_average, vote\_count, runtime, popularity): Điền giá trị thiếu bằng **giá trị trung vị** (median).
  + Đối với cột release\_date: Loại bỏ các dòng thiếu vì imputation cho ngày không hợp lý.
  + Đối với cột genres: Điền giá trị thiếu bằng chuỗi rỗng ("").
* **Code:** Hàm handle\_missing\_values\_imputation(df).

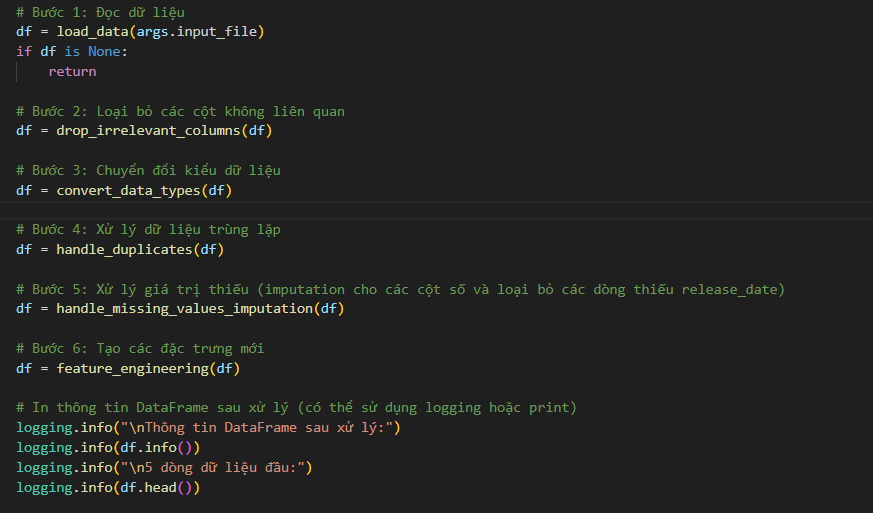


##### Hình 3.8: Code xử lý giá trị thiếu

* **Lý do:**
  + Giá trị trung vị ít bị ảnh hưởng bởi các giá trị ngoại lai so với trung bình cộng.
  + release\_date cần chính xác để trích xuất đặc trưng như year, month.

***Bước 6: Tạo các đặc trưng mới***

* **Mô tả:**
  + Tạo cột year và month từ release\_date.
  + Tạo cột profit = revenue - budget.
  + Xử lý cột genres: Tách chuỗi thành danh sách và áp dụng **one-hot encoding** để biến đổi thành các cột nhị phân (ví dụ: genre\_Action, genre\_Drama).
* **Code:** Hàm feature\_engineering(df).



##### Hình 3.8: Code tạo các đặc trưng mới

* **Kết quả:** Tăng số lượng đặc trưng để mô hình học máy có thể sử dụng.

### 3.3.5. Tóm tắt và nhận xét

Qua quá trình xử lý dữ liệu ban đầu, tập dữ liệu đã được “làm sạch” và chuẩn hóa, tạo nền tảng vững chắc cho các bước tiếp theo trong quá trình xây dựng mô hình dự đoán. Việc loại bỏ các cột không cần thiết giúp giảm thiểu nhiễu, trong khi các thao tác chuyển đổi dữ liệu và xử lý giá trị thiếu đảm bảo rằng thông tin tài chính, đánh giá và các thông số khác được thể hiện chính xác.  
Những bước xử lý này không chỉ cải thiện chất lượng dữ liệu mà còn giúp tiết kiệm thời gian và tài nguyên tính toán trong quá trình huấn luyện mô hình học máy, góp phần nâng cao độ chính xác của dự đoán xu hướng phim.

# CHƯƠNG 4: TẠO BIẾN TARGET ‘TREND’

## 4.1. Mục đích của biến 'trend'

Trong dự án dự đoán xu hướng phim, biến 'trend' đóng vai trò là **biến mục tiêu (target variable)** mà mô hình học máy sẽ học để dự đoán. Đây là một biến nhị phân, nhận giá trị **0** hoặc **1**, nhằm phân loại một bộ phim có xu hướng thành công hay không dựa trên hiệu quả tài chính của nó. Cụ thể:

* **trend = 1**: Bộ phim được coi là có xu hướng thành công.
* **trend = 0**: Bộ phim không được coi là thành công.

Việc xác định "thành công" ở đây sẽ dựa trên **lợi nhuận (profit)** của bộ phim, một chỉ số quan trọng phản ánh hiệu quả tài chính, từ đó giúp các nhà sản xuất phim đánh giá xu hướng thị trường.

## 4.2. Lý do chọn 'profit' làm cơ sở để xác định 'trend'

Để tạo biến 'trend', chúng ta cần một tiêu chí rõ ràng và có ý nghĩa để đánh giá thành công của một bộ phim. Trong trường hợp này, **profit** được chọn vì các lý do sau:

1. **Tính trực quan**: Profit là kết quả trực tiếp của doanh thu (revenue) trừ đi ngân sách (budget), cho thấy một bộ phim có sinh lời hay không.
2. **Tầm quan trọng thực tế**: Với các nhà sản xuất phim, yếu tố tài chính thường là mối quan tâm hàng đầu khi đánh giá sự thành công của một dự án.
3. **Khả năng đo lường**: Dữ liệu về doanh thu và ngân sách thường có sẵn trong các tập dữ liệu phim, giúp dễ dàng tính toán profit mà không cần dựa vào các yếu tố chủ quan như cảm nhận của khán giả.

Tuy nhiên, profit không phải là yếu tố duy nhất định nghĩa sự thành công của một bộ phim (ví dụ: một bộ phim có thể thành công về mặt nghệ thuật hoặc văn hóa nhưng không sinh lời). Dù vậy, trong phạm vi dự án này, chúng ta tập trung vào khía cạnh tài chính để đơn giản hóa và đảm bảo tính khách quan.

## 4.3. Quy trình tạo biến 'trend'

Để tạo biến 'trend', chúng ta sẽ thực hiện một chuỗi các bước cụ thể từ việc tính toán profit đến phân loại phim dựa trên một ngưỡng hợp lý. Dưới đây là quy trình chi tiết:

***Bước 1: Tính toán profit***

Profit là giá trị cơ bản để đánh giá hiệu quả tài chính của một bộ phim.

* **revenue**: Doanh thu toàn cầu của bộ phim (thường tính bằng USD).
* **budget**: Ngân sách sản xuất của bộ phim (cũng tính bằng USD).

Trong mã nguồn, giả sử dữ liệu được lưu trong một DataFrame (ví dụ sử dụng thư viện Spark hoặc Pandas), chúng ta có thể tạo cột 'profit' như sau:

df["profit"] = df["revenue"] - df["budget"]

* Nếu một bộ phim có doanh thu nhỏ hơn ngân sách, profit sẽ là số âm, cho thấy phim bị lỗ.
* Ngược lại, profit dương cho thấy phim có lãi.

Ví dụ:

* Phim A: revenue = 200 triệu USD, budget = 150 triệu USD → profit = 50 triệu USD.
* Phim B: revenue = 80 triệu USD, budget = 100 triệu USD → profit = -20 triệu USD.

***Bước 2: Xác định ngưỡng để phân loại 'trend'***

Để biến profit thành một nhãn nhị phân (0 hoặc 1), chúng ta cần một **ngưỡng (threshold)** để so sánh. Các phim có profit vượt ngưỡng sẽ được gán nhãn 1, và các phim dưới ngưỡng sẽ được gán nhãn 0.

***Lựa chọn ngưỡng: Giá trị trung vị (median) của profit***

Ngưỡng được chọn ở đây là **giá trị trung vị (median)** của cột 'profit' trong toàn bộ tập dữ liệu. Lý do chọn median thay vì các phương pháp khác (như trung bình hoặc một giá trị cố định) bao gồm:

* **Kháng ngoại lai (robust to outliers)**: Trong dữ liệu phim, có thể tồn tại các giá trị profit cực đại (ví dụ: các bom tấn như Avengers) hoặc cực tiểu (các phim thất bại thảm hại). Giá trị trung bình (mean) dễ bị ảnh hưởng bởi những ngoại lai này, trong khi median thì không.
* **Cân bằng dữ liệu**: Median chia tập dữ liệu thành hai phần bằng nhau: 50% phim có profit cao hơn median và 50% thấp hơn. Điều này đảm bảo số lượng phim ở hai lớp (trend = 0 và trend = 1) gần như tương đương, tránh tình trạng mất cân bằng lớp (class imbalance) trong mô hình học máy.
* **Tính thích nghi**: Median được tính trực tiếp từ dữ liệu, nên phương pháp này có thể áp dụng cho bất kỳ tập dữ liệu phim nào mà không cần điều chỉnh thủ công.

***Các phương pháp thay thế và lý do không chọn***

* **Trung bình (mean)**: Như đã đề cập, mean nhạy cảm với outliers, có thể làm ngưỡng bị lệch.
* **Ngưỡng cố định (ví dụ: profit > 0)**: Chỉ xét phim có lãi (profit > 0) có thể không phản ánh đúng xu hướng, vì một số phim có profit dương nhưng rất thấp vẫn không được coi là thành công so với kỳ vọng.
* **Top X% (ví dụ: top 25%)**: Tuy có thể khả thi, nhưng đòi hỏi phải xác định tỷ lệ phần trăm trước, làm tăng độ phức tạp không cần thiết.

***Bước 3: Tính toán median của profit***

Để tính median, chúng ta sử dụng hàm phù hợp tùy theo công cụ xử lý dữ liệu:

* Trong **Pandas**:

median\_profit = df["profit"].median()

* Trong **PySpark** (dành cho dữ liệu lớn):

median\_profit = df.approxQuantile("profit", [0.5], 0.01)[0]

* + [0.5]: Chỉ định quantile thứ 50%, tức là median.
  + 0.01: Độ chính xác tương đối, giúp tăng tốc độ tính toán trên dữ liệu lớn.

Giả sử median\_profit được tính ra là 30 triệu USD. Điều này có nghĩa là 50% phim trong tập dữ liệu có profit lớn hơn 30 triệu USD, và 50% còn lại nhỏ hơn.

***Bước 4: Tạo cột 'trend' dựa trên ngưỡng***

Dựa trên giá trị median, chúng ta tạo cột 'trend' với logic:

* Nếu profit >= median\_profit, trend = 1 (phim thành công).
* Nếu profit < median\_profit, trend = 0 (phim không thành công).

Trong mã nguồn (sử dụng PySpark):



Ví dụ:

* Phim A: profit = 50 triệu USD, median = 30 triệu USD → trend = 1.
* Phim B: profit = -20 triệu USD, median = 30 triệu USD → trend = 0.

## 4.4. Ưu điểm của phương pháp này

Phương pháp tạo biến 'trend' dựa trên profit và median có nhiều lợi ích:

1. **Cân bằng lớp tự nhiên**: Vì median chia dữ liệu thành hai phần bằng nhau, mô hình học máy sẽ không bị thiên lệch do số lượng mẫu ở một lớp quá lớn.
2. **Đơn giản và dễ triển khai**: Quy trình chỉ cần vài bước tính toán cơ bản, không đòi hỏi các giả định phức tạp hay kiến thức chuyên môn sâu về ngành phim.
3. **Tính linh hoạt**: Median được tính từ dữ liệu thực tế, nên phương pháp này có thể áp dụng cho bất kỳ tập dữ liệu phim nào, bất kể quy mô hay phạm vi thời gian.
4. **Khách quan**: Việc sử dụng profit và median loại bỏ yếu tố chủ quan, đảm bảo kết quả dựa hoàn toàn trên số liệu.

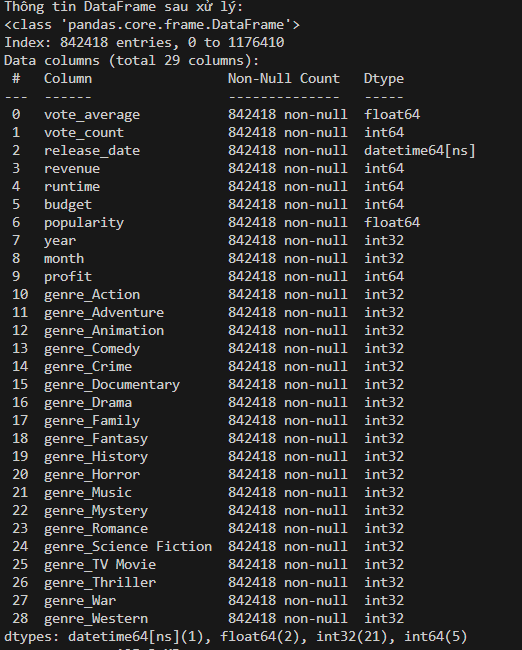
## 4.5. Nhược điểm và cách khắc phục

Mặc dù phương pháp này hiệu quả, nó cũng có một số hạn chế:

1. **Bỏ qua các yếu tố phi tài chính**: Thành công của một bộ phim không chỉ nằm ở profit mà còn ở các yếu tố như điểm đánh giá (rating), giải thưởng, hoặc tầm ảnh hưởng văn hóa.
   * **Khắc phục**: Nếu cần, có thể kết hợp thêm các yếu tố này vào ngưỡng (ví dụ: profit cao + rating > 8.0), nhưng sẽ làm phức tạp hóa quy trình.
2. **Dữ liệu có thể chứa outliers**: Dù median ít bị ảnh hưởng bởi outliers, những giá trị cực đoan vẫn có thể làm lệch phân bố profit.
   * **Khắc phục**: Có thể tiền xử lý dữ liệu bằng cách loại bỏ hoặc điều chỉnh các outliers trước khi tính median.
3. **Ngưỡng cố định không linh hoạt**: Sử dụng median có thể không phù hợp nếu mục tiêu là xác định "top phim" (ví dụ: top 10% thay vì 50%).
   * **Khắc phục**: Có thể thay đổi ngưỡng thành quantile khác (ví dụ: 75%) tùy theo yêu cầu cụ thể của dự án.

## 4.6. Kết quả cuối cùng

Sau khi hoàn thành các bước trên, tập dữ liệu sẽ có thêm cột 'trend' với giá trị 0 hoặc 1, đại diện cho xu hướng thành công của từng bộ phim. Biến này sẽ được sử dụng làm nhãn mục tiêu trong quá trình huấn luyện và đánh giá mô hình học máy. Ví dụ về kết quả trong DataFrame:



##### Hình 4.1: Kết quả thông tin sau DataFrame sau xử lý

# CHƯƠNG 5: XỬ LÝ CỘT GENRE

## 5.1. Xử lý cột genre

Cột genres trong tập dữ liệu ban đầu chứa thông tin về các thể loại phim, được biểu diễn dưới dạng một chuỗi văn bản, ví dụ: "Action, Adventure, Science Fiction". Đây là một đặc trưng quan trọng để dự đoán xu hướng phim, nhưng dữ liệu dạng chuỗi này không thể sử dụng trực tiếp trong các mô hình học máy, vốn yêu cầu đầu vào là số. Vì vậy, quá trình xử lý cột genres cần được thực hiện qua nhiều bước để chuyển đổi dữ liệu thành dạng số hóa hiệu quả, đồng thời xử lý các trường hợp đặc biệt như thể loại hiếm để tối ưu hóa cho việc huấn luyện mô hình. Dưới đây là các bước chi tiết:

***Bước 1: Tách chuỗi genres thành danh sách các thể loại riêng lẻ***

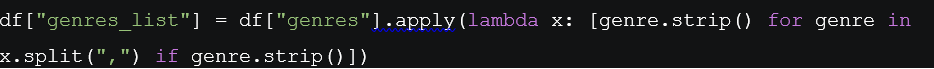
**Mô tả:**

Mỗi giá trị trong cột genres là một chuỗi văn bản chứa nhiều thể loại phim, được phân tách bằng dấu phẩy và có thể kèm theo khoảng trắng dư thừa (ví dụ: "Action, Adventure, Science Fiction"). Để xử lý, chúng ta cần tách chuỗi này thành một danh sách các thể loại riêng biệt, đồng thời loại bỏ các khoảng trắng không cần thiết để chuẩn hóa dữ liệu.

***Cách thực hiện***

* Sử dụng phương thức split(",") trong Python để tách chuỗi tại dấu phẩy.
* Áp dụng hàm strip() để loại bỏ khoảng trắng ở đầu và cuối mỗi thể loại.
* Lọc bỏ các giá trị rỗng (nếu có) để đảm bảo danh sách chỉ chứa các thể loại hợp lệ.

***Code:***

******

##### Hình 5.1: Code Tách chuỗi genres

* Hàm lambda này nhận từng giá trị trong cột genres, tách chuỗi thành danh sách, và xử lý từng phần tử.

**Ví dụ:**

* **Đầu vào:** "Action, Adventure, Science Fiction"
* **Đầu ra:** ["Action", "Adventure", "Science Fiction"]
* **Trường hợp đặc biệt:** Nếu có giá trị như "Comedy, , Drama" (có dấu phẩy thừa), kết quả sẽ là ["Comedy", "Drama"], vì phần tử rỗng bị loại bỏ.

Lý do

Việc tách chuỗi thành danh sách là bước đầu tiên để chuẩn bị dữ liệu cho các kỹ thuật mã hóa tiếp theo, vì hầu hết các phương pháp mã hóa (như one-hot encoding) yêu cầu đầu vào là danh sách hoặc tập hợp các giá trị riêng lẻ.

***Bước 2: Áp dụng One-hot Encoding để chuyển đổi thành các cột nhị phân***

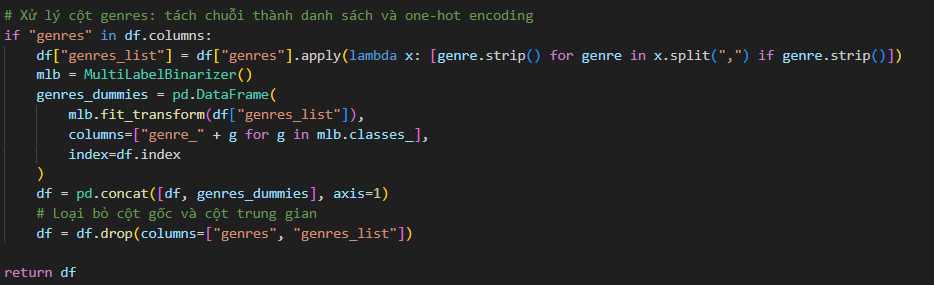
**Mô tả**

Sau khi có danh sách các thể loại cho mỗi phim, chúng ta cần chuyển đổi thông tin này thành dạng số để mô hình học máy có thể hiểu được. Phương pháp **one-hot encoding** được sử dụng để tạo ra các cột nhị phân riêng biệt cho từng thể loại phim. Mỗi cột đại diện cho một thể loại, với giá trị 1 nếu phim thuộc thể loại đó và 0 nếu không.

***Cách thực hiện:***

* Sử dụng lớp MultiLabelBinarizer từ thư viện sklearn.preprocessing để thực hiện mã hóa one-hot cho dữ liệu đa nhãn (một phim có thể thuộc nhiều thể loại).
* Tạo các cột mới với tiền tố "genre\_" để dễ nhận diện, ví dụ: genre\_Action, genre\_Adventure.
* Gắn kết quả mã hóa vào DataFrame ban đầu.

***Code:***

******

##### Hình 5.2: Code xử lý cột genres (tách chuỗi)

* mlb.fit\_transform() biến đổi danh sách thể loại thành ma trận nhị phân.
* pd.DataFrame tạo DataFrame mới từ ma trận này với tên cột được thêm tiền tố "genre\_".
* pd.concat nối các cột mới vào DataFrame chính theo trục cột (axis=1).

Ví dụ

* **Danh sách thể loại:** ["Action", "Adventure", "Science Fiction"]
* **Kết quả:**
  + genre\_Action = 1
  + genre\_Adventure = 1
  + genre\_Science Fiction = 1
  + genre\_Comedy = 0 (nếu "Comedy" không có trong danh sách)
* Tập dữ liệu sẽ có tất cả các cột tương ứng với các thể loại duy nhất xuất hiện trong toàn bộ dữ liệu.

**Lý do:**

One-hot encoding là một phương pháp phổ biến và hiệu quả để biểu diễn dữ liệu phân loại trong học máy. Nó đảm bảo rằng mỗi thể loại được xử lý độc lập, không giả định thứ tự hay mối quan hệ nào giữa các thể loại, phù hợp với đặc điểm của dữ liệu phim.

***Bước 3: Xử lý các thể loại hiếm để giảm chiều dữ liệu và tránh overfitting***

**Mô tả**

Trong tập dữ liệu, một số thể loại có thể xuất hiện rất ít (ví dụ: dưới 5% tổng số phim), như Documentary hoặc Western. Việc giữ nguyên các cột one-hot encoding cho những thể loại này có thể gây ra hai vấn đề:

1. **Tăng chiều dữ liệu:** Số lượng cột tăng lên không cần thiết, làm phức tạp mô hình.
2. **Overfitting:** Mô hình có thể học các mẫu không có ý nghĩa từ các thể loại hiếm, dẫn đến hiệu suất kém trên dữ liệu mới.

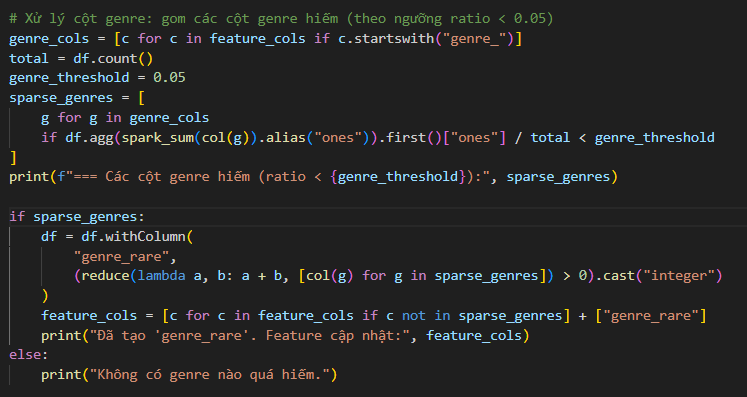
Để giải quyết, chúng ta sẽ:

* Xác định các thể loại hiếm dựa trên tỷ lệ xuất hiện.
* Gộp tất cả các thể loại hiếm vào một cột duy nhất gọi là genre\_rare.

*Cách thực hiện*

1. **Tính tỷ lệ xuất hiện:**
   * Đếm số lần xuất hiện của mỗi thể loại trong toàn bộ tập dữ liệu.
   * Chia cho tổng số phim để tính tỷ lệ.
   * Đặt ngưỡng (threshold) là 5% (có thể điều chỉnh tùy theo tập dữ liệu).
2. **Xác định thể loại hiếm:**
   * Các thể loại có tỷ lệ xuất hiện dưới ngưỡng sẽ được coi là hiếm.
3. **Gộp thành cột genre\_rare:**
   * Tạo cột mới với giá trị 1 nếu phim thuộc ít nhất một thể loại hiếm, và 0 nếu không.
   * Loại bỏ các cột riêng lẻ của thể loại hiếm khỏi tập dữ liệu.

***Code:***

**

##### Hình 5.3: Code xử lý cột genres (gom các cột genre hiếm)

Ví dụ

* **Tập dữ liệu:** 1000 phim.
* **Thể loại hiếm:** genre\_Documentary (30 phim, 3%), genre\_Western (20 phim, 2%).
* **Ngưỡng:** 5% (50 phim).
* **Kết quả:**
  + Các cột genre\_Documentary và genre\_Western bị loại bỏ.
  + Cột genre\_rare = 1 cho bất kỳ phim nào thuộc Documentary hoặc Western, ngược lại = 0.

Lý do

* **Giảm chiều dữ liệu:** Thay vì giữ hàng chục cột hiếm, chỉ còn một cột genre\_rare.
* **Tránh overfitting:** Mô hình không học các mẫu không đáng tin cậy từ dữ liệu hiếm.
* **Giữ thông tin:** Vẫn duy trì thông tin rằng phim thuộc một thể loại ít phổ biến.

***Bước 4: Loại bỏ các cột không cần thiết để làm sạch dữ liệu***

**Mô tả**

Sau khi hoàn thành các bước trên, cột genres gốc (dạng chuỗi) và cột genres\_list (danh sách trung gian) không còn cần thiết nữa, vì thông tin đã được chuyển đổi sang các cột nhị phân và cột genre\_rare. Việc loại bỏ các cột này giúp tập dữ liệu gọn gàng hơn và chỉ chứa các đặc trưng số phù hợp cho mô hình.

*Cách thực hiện*

* Sử dụng phương thức drop() của DataFrame để xóa các cột không cần thiết.

***Code:***

df = df.drop(columns=["genres", "genres\_list"])

**Ví dụ**

* **Trước khi loại bỏ:** DataFrame chứa genres, genres\_list, genre\_Action, genre\_rare, v.v.
* **Sau khi loại bỏ:** Chỉ còn các cột đặc trưng số như genre\_Action, genre\_Adventure, genre\_rare.

Lý do

* **Tối ưu hóa bộ nhớ:** Loại bỏ dữ liệu dư thừa để giảm kích thước tập dữ liệu.
* **Chuẩn bị cho mô hình:** Đảm bảo đầu vào của mô hình chỉ chứa các đặc trưng đã xử lý.

## 5.2. Tóm tắt logic xử lý cột genre

1. **Tách chuỗi genres thành danh sách:**
   * Chuyển đổi chuỗi văn bản thành danh sách các thể loại riêng lẻ, loại bỏ khoảng trắng và giá trị rỗng.
2. **One-hot encoding:**
   * Biến đổi danh sách thể loại thành các cột nhị phân (0 và 1) đại diện cho từng thể loại.
3. **Xử lý thể loại hiếm:**
   * Xác định các thể loại xuất hiện dưới ngưỡng 5%, gộp chúng thành cột genre\_rare để giảm chiều và tránh overfitting.
4. **Loại bỏ cột không cần thiết:**
   * Xóa cột genres và genres\_list để làm sạch dữ liệu, chỉ giữ lại các đặc trưng số.

## 5.3. Kết quả cuối cùng

Sau khi xử lý, cột genres ban đầu được thay thế bằng một tập hợp các cột nhị phân (ví dụ: genre\_Action, genre\_Drama) và một cột genre\_rare (nếu cần). Dữ liệu giờ đây ở dạng số hoàn toàn, sẵn sàng để đưa vào huấn luyện mô hình học máy. Quá trình này không chỉ đảm bảo tính chính xác mà còn tối ưu hóa hiệu suất bằng cách cân bằng giữa việc giữ thông tin và giảm độ phức tạp của dữ liệu.

# CHƯƠNG 6: OVERSAMPLING LỚP MINORITY

## 6.1. Mô tả vấn đề mất cân bằng dữ liệu

Trong các bài toán phân loại, dữ liệu mất cân bằng là tình trạng xảy ra khi số lượng mẫu (samples) của một lớp (class) ít hơn đáng kể so với các lớp khác. Trong dự án "Dự đoán xu hướng phim", chúng ta xây dựng biến mục tiêu (target variable) là trend, đại diện cho mức độ thành công của một bộ phim dựa trên lợi nhuận (profit). Biến trend được định nghĩa như sau:

* **trend = 0**: Phim có lợi nhuận thấp hơn trung vị (median) của toàn bộ tập dữ liệu – được xem là **không thành công** (lớp majority).
* **trend = 1**: Phim có lợi nhuận cao hơn hoặc bằng trung vị – được xem là **thành công** (lớp minority).

Sau khi tính toán trung vị của profit và gán nhãn cho biến trend, quá trình phân tích dữ liệu ban đầu cho thấy sự chênh lệch rõ rệt về số lượng mẫu giữa hai lớp. Cụ thể, số lượng phim thuộc lớp trend = 1 (phim thành công) thường ít hơn nhiều so với lớp trend = 0 (phim không thành công). Đây là một đặc điểm phổ biến trong các bài toán thực tế, nơi các sự kiện "hiếm" (như phim thành công về mặt lợi nhuận) có tần suất xuất hiện thấp hơn.

### 6.1.1. Tác động của dữ liệu mất cân bằng

Sự mất cân bằng này gây ra một số vấn đề nghiêm trọng khi huấn luyện mô hình học máy:

1. **Thiên lệch mô hình (Model Bias):**  
   Các thuật toán học máy có xu hướng ưu tiên dự đoán chính xác lớp chiếm đa số (majority class) vì nó đóng góp nhiều hơn vào hàm mất mát (loss function). Kết quả là mô hình có thể bỏ qua hoặc dự đoán kém cho lớp minority (trend = 1), mặc dù đây là lớp quan trọng trong bài toán dự đoán xu hướng phim.
2. **Đánh giá không chính xác:**  
   Nếu chỉ dựa vào chỉ số độ chính xác tổng thể (accuracy), mô hình có thể đạt kết quả cao do dự đoán tốt lớp majority, nhưng lại không phản ánh đúng khả năng nhận diện lớp minority. Ví dụ, nếu 90% dữ liệu thuộc lớp trend = 0, mô hình chỉ cần dự đoán tất cả là trend = 0 để đạt accuracy 90%, dù không nhận diện được phim thành công nào.
3. **Mất thông tin quan trọng:**  
   Trong bối cảnh dự án này, lớp minority (trend = 1) đại diện cho các bộ phim thành công – điều mà các nhà làm phim hoặc nhà đầu tư đặc biệt quan tâm. Nếu mô hình không dự đoán tốt lớp này, giá trị thực tiễn của dự án sẽ bị giảm sút đáng kể.

### 6.1.2.Lý do cần xử lý mất cân bằng

Vì những lý do trên, việc xử lý dữ liệu mất cân bằng trở thành một bước quan trọng trong quy trình tiền xử lý dữ liệu. Mục tiêu là đảm bảo mô hình học máy có thể học được đặc điểm của cả hai lớp một cách công bằng, đặc biệt là cải thiện khả năng dự đoán lớp minority mà không làm mất đi thông tin từ lớp majority.

## 6.2. Phương pháp Oversampling

**Oversampling** là một kỹ thuật xử lý dữ liệu mất cân bằng bằng cách tăng số lượng mẫu của lớp minority để cân bằng với lớp majority. Có hai cách tiếp cận chính trong oversampling:

* **Nhân bản ngẫu nhiên (Random Oversampling):** Sao chép ngẫu nhiên các mẫu hiện có của lớp minority.
* **Tạo mẫu tổng hợp (Synthetic Oversampling):** Sử dụng các thuật toán như SMOTE để tạo ra các mẫu mới dựa trên các mẫu hiện có.

Trong dự án này, chúng ta chọn phương pháp **Random Oversampling** với việc nhân bản ngẫu nhiên các mẫu của lớp minority bằng cách lấy mẫu có hoàn lại (with replacement). Phương pháp này được triển khai trên Apache Spark để xử lý dữ liệu lớn một cách hiệu quả.

### 6.2.1. Lý do chọn Random Oversampling

1. **Đơn giản và dễ triển khai:**  
   Random Oversampling không yêu cầu các thuật toán phức tạp hay tính toán khoảng cách giữa các điểm dữ liệu (như SMOTE). Điều này làm cho nó trở thành lựa chọn phù hợp khi cần xử lý nhanh chóng trên tập dữ liệu lớn.
2. **Không làm mất thông tin:**  
   So với **Undersampling** (giảm số lượng mẫu của lớp majority), oversampling giữ nguyên toàn bộ thông tin từ lớp majority, tránh nguy cơ loại bỏ các mẫu quan trọng.
3. **Tương thích với Spark:**  
   Apache Spark cung cấp các công cụ mạnh mẽ như sample() để thực hiện lấy mẫu ngẫu nhiên trên dữ liệu phân tán, phù hợp với quy mô dữ liệu trong dự án.
4. **Hiệu quả trong trường hợp mất cân bằng vừa phải:**  
   Khi mức độ mất cân bằng không quá nghiêm trọng (ví dụ, tỷ lệ majority:minority là 5:1 thay vì 100:1), Random Oversampling thường mang lại cải thiện đáng kể mà không cần đến các phương pháp phức tạp hơn.

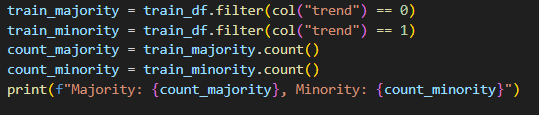
## 6.3. Các bước thực hiện Oversampling

Quá trình oversampling được thực hiện trên **tập huấn luyện** (training set) sau khi đã chia dữ liệu gốc thành tập huấn luyện và tập kiểm tra (test set). Điều này đảm bảo rằng tập kiểm tra vẫn phản ánh phân phối thực tế của dữ liệu, tránh hiện tượng "data leakage" (rò rỉ thông tin từ tập kiểm tra sang tập huấn luyện). Dưới đây là các bước chi tiết:

***Bước 1: Xác định lớp majority và minority***

* Lọc dữ liệu từ tập huấn luyện để tách riêng các mẫu thuộc lớp majority (trend = 0) và lớp minority (trend = 1).
* Tính số lượng mẫu của từng lớp để hiểu mức độ mất cân bằng:
  + count\_majority: Số lượng mẫu của lớp trend = 0.
  + count\_minority: Số lượng mẫu của lớp trend = 1.

**Code ví dụ (sử dụng PySpark):**

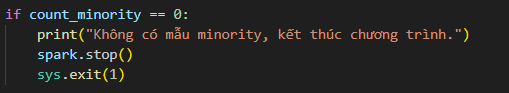


##### Hình 6.1: **Code ví dụ (sử dụng PySpark)**

***Bước 2: Kiểm tra nếu lớp minority có mẫu***

* Trong một số trường hợp hiếm gặp (đặc biệt khi tập dữ liệu nhỏ hoặc quá trình chia dữ liệu không hợp lý), tập huấn luyện có thể không chứa mẫu nào thuộc lớp minority (count\_minority = 0).
* Nếu điều này xảy ra, oversampling không thể thực hiện được. Trong trường hợp này, cần xem xét lại cách chia dữ liệu (ví dụ: sử dụng stratified sampling) hoặc thu thập thêm dữ liệu.

**Code kiểm tra:**



##### HÌnh 6.2 : Code kiểm tra

***Bước 3: Tính tỷ lệ oversampling***

* Để cân bằng số lượng mẫu giữa hai lớp, chúng ta tính tỷ lệ ratio giữa số lượng mẫu của lớp majority và lớp minority:
* Tỷ lệ này sẽ được sử dụng để xác định mức độ nhân bản cần thiết cho lớp minority, sao cho sau khi oversampling, số lượng mẫu của lớp minority xấp xỉ bằng lớp majority.

**Code:**

ratio = count\_majority / count\_minority

print(f"Tỷ lệ oversampling cần thiết: {ratio:.2f}")

##### Hình 6.3: Code Tính tỷ lệ oversampling

**Ví dụ:** Nếu count\_majority = 10,000 và count\_minority = 2,000, thì ratio = 10,000 / 2,000 = 5. Điều này có nghĩa là mỗi mẫu minority sẽ được nhân bản trung bình 5 lần.

***Bước 4: Thực hiện oversampling***

* Sử dụng phương thức sample() của PySpark để lấy mẫu ngẫu nhiên từ lớp minority:
  + withReplacement=True: Cho phép lấy mẫu có hoàn lại, tức là một mẫu có thể được chọn nhiều lần.
  + fraction=ratio: Xác định tỷ lệ lấy mẫu, đảm bảo số lượng mẫu sau khi oversampling đạt mức mong muốn.
  + seed=42: Đặt giá trị seed để đảm bảo kết quả có thể tái lập (reproducible).

**Code:**

oversampled\_minority = train\_minority.sample(withReplacement=True, fraction=ratio, seed=42)

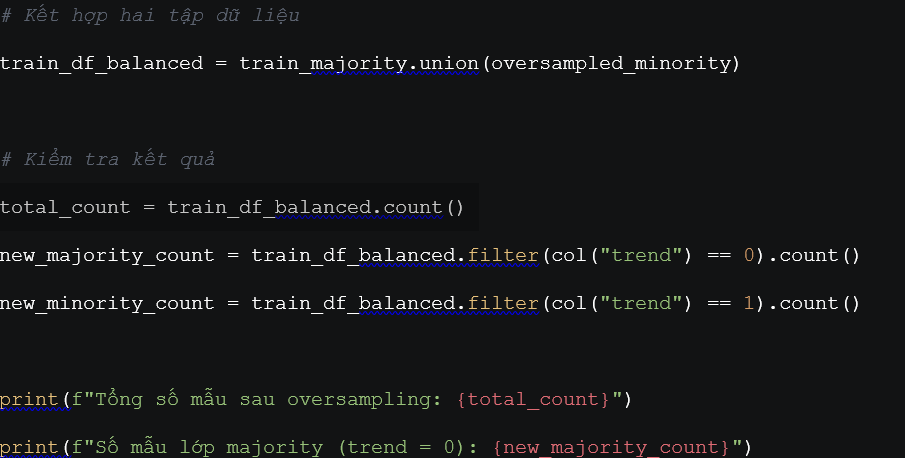
##### Hình 6.4 : Code Thực hiện oversampling

* Sau khi oversampling, số lượng mẫu của lớp minority sẽ tăng lên gần bằng hoặc bằng số lượng mẫu của lớp majority, tùy thuộc vào tính ngẫu nhiên của quá trình lấy mẫu.

***Bước 5: Kết hợp lại tập huấn luyện***

* Gộp DataFrame của lớp majority (train\_majority) và DataFrame của lớp minority đã được oversampling (oversampled\_minority) để tạo ra tập huấn luyện cân bằng mới (train\_df\_balanced).
* Kiểm tra lại số lượng mẫu của từng lớp để đảm bảo quá trình oversampling đã thành công.

**Code:**



##### Hình 6.5 : Code Kết hợp lại tập huấn luyện

## 6.4. Ưu và nhược điểm của Random Oversampling

### 6.4.1.Ưu điểm

1. **Cải thiện độ nhạy (recall) cho lớp minority:**  
   Bằng cách tăng số lượng mẫu của lớp minority, mô hình có nhiều cơ hội học được các đặc trưng của lớp này, từ đó cải thiện khả năng dự đoán các phim thành công.
2. **Giữ nguyên thông tin của lớp majority:**  
   Không giống như undersampling, oversampling không loại bỏ bất kỳ mẫu nào từ lớp majority, đảm bảo rằng toàn bộ thông tin ban đầu được sử dụng trong quá trình huấn luyện.
3. **Dễ triển khai trên dữ liệu lớn:**  
   Với sự hỗ trợ của Spark, việc thực hiện oversampling trên các tập dữ liệu lớn trở nên nhanh chóng và hiệu quả.
4. **Tăng tính ổn định của mô hình:**  
   Khi hai lớp có số lượng mẫu tương đương, mô hình ít bị thiên lệch hơn, dẫn đến kết quả dự đoán ổn định hơn trên cả hai lớp.

### 6.4.2. Nhược điểm

1. **Tăng kích thước dữ liệu:**  
   Việc nhân bản mẫu làm tăng tổng số lượng mẫu trong tập huấn luyện, có thể dẫn đến thời gian huấn luyện mô hình lâu hơn, đặc biệt với các thuật toán phức tạp.
2. **Nguy cơ overfitting:**  
   Vì các mẫu minority được nhân bản từ các mẫu hiện có, mô hình có thể "học thuộc" các mẫu này thay vì tìm hiểu các quy luật tổng quát, dẫn đến hiệu suất kém trên tập kiểm tra.
3. **Không tạo thông tin mới:**  
   Random Oversampling chỉ sao chép các mẫu hiện có mà không tạo ra thông tin mới, không giống như các phương pháp synthetic sampling (như SMOTE) có thể tạo ra các mẫu đa dạng hơn.
4. **Phụ thuộc vào chất lượng dữ liệu ban đầu:**  
   Nếu các mẫu minority ban đầu không đại diện tốt cho toàn bộ lớp (ví dụ: do nhiễu hoặc thiếu đa dạng), việc nhân bản sẽ không cải thiện được chất lượng dữ liệu.

## 6.5. Kết quả sau Oversampling

Sau khi áp dụng oversampling, tập huấn luyện sẽ có phân phối mẫu cân bằng hơn giữa hai lớp. Ví dụ minh họa:

* **Ban đầu:**
  + Lớp majority (trend = 0): 10,000 mẫu
  + Lớp minority (trend = 1): 2,000 mẫu
  + Tỷ lệ: 5:1
* **Sau oversampling với ratio = 5:**
  + Lớp minority được nhân bản lên khoảng 2,000 \* 5 = 10,000 mẫu.
  + Tập huấn luyện mới: 10,000 (majority) + 10,000 (minority) = 20,000 mẫu.
  + Tỷ lệ mới: 1:1

Kết quả này giúp mô hình học máy có cơ hội học tốt hơn từ cả hai lớp, giảm thiểu thiên lệch và cải thiện các chỉ số như precision, recall và F1-score cho lớp minority.

## 6.6. Các phương pháp thay thế để xử lý mất cân bằng

Ngoài Random Oversampling, có nhiều kỹ thuật khác để xử lý dữ liệu mất cân bằng. Dưới đây là một số phương pháp phổ biến và so sánh ngắn gọn:

1. **Undersampling:**
   * **Cách làm:** Giảm ngẫu nhiên số lượng mẫu của lớp majority để cân bằng với lớp minority.
   * **Ưu điểm:** Giảm kích thước dữ liệu, nhanh hơn khi huấn luyện.
   * **Nhược điểm:** Có thể làm mất thông tin quan trọng từ lớp majority.
2. **Synthetic Minority Over-sampling Technique (SMOTE):**
   * **Cách làm:** Tạo ra các mẫu synthetic mới cho lớp minority bằng cách nội suy giữa các mẫu hiện có.
   * **Ưu điểm:** Tăng tính đa dạng của dữ liệu, giảm nguy cơ overfitting so với Random Oversampling.
   * **Nhược điểm:** Phức tạp hơn, khó triển khai trên Spark với dữ liệu lớn.
3. **Sử dụng trọng số lớp (Class Weights):**
   * **Cách làm:** Gán trọng số cao hơn cho lớp minority trong hàm mất mát của mô hình.
   * **Ưu điểm:** Không cần thay đổi dữ liệu, dễ tích hợp vào các thuật toán như Logistic Regression, SVM, hoặc Random Forest.
   * **Nhược điểm:** Không phải lúc nào cũng hiệu quả, phụ thuộc vào thuật toán.
4. **Kết hợp Oversampling và Undersampling:**
   * **Cách làm:** Kết hợp cả hai kỹ thuật để đạt được sự cân bằng mà không làm tăng quá nhiều kích thước dữ liệu hoặc mất quá nhiều thông tin.
   * **Ưu điểm:** Linh hoạt, có thể tùy chỉnh theo bài toán.
   * **Nhược điểm:** Cần thử nghiệm nhiều lần để tìm tỷ lệ tối ưu.

Trong dự án này, Random Oversampling được chọn vì tính đơn giản, hiệu quả với dữ liệu lớn trên Spark, và phù hợp với mức độ mất cân bằng vừa phải của dữ liệu.

## 6.7. Đánh giá hiệu quả của Oversampling

Để đánh giá xem oversampling có thực sự cải thiện hiệu suất mô hình hay không, chúng ta cần so sánh hiệu suất trước và sau khi áp dụng oversampling trên tập kiểm tra. Các chỉ số quan trọng cần xem xét bao gồm:

* **Precision:** Tỷ lệ dự đoán đúng trong số các mẫu được dự đoán là trend = 1.
* **Recall:** Tỷ lệ các mẫu trend = 1 thực tế được dự đoán đúng.
* **F1-score:** Trung bình hài hòa giữa precision và recall, đặc biệt quan trọng với dữ liệu mất cân bằng.
* **ROC-AUC:** Đo lường khả năng phân biệt giữa hai lớp.

Nếu sau oversampling, các chỉ số này tăng lên (đặc biệt là recall và F1-score cho lớp minority) mà không làm giảm đáng kể hiệu suất trên lớp majority, có thể kết luận rằng phương pháp này đã thành công.

## 6.8. Kết luận

Quá trình oversampling lớp minority trong dự án "Dự đoán xu hướng phim" là một bước quan trọng để xử lý vấn đề mất cân bằng dữ liệu. Bằng cách sử dụng Random Oversampling trên Spark, chúng ta đã tăng số lượng mẫu của lớp trend = 1 để cân bằng với lớp trend = 0, từ đó cải thiện khả năng dự đoán của mô hình, đặc biệt đối với các phim thành công. Mặc dù phương pháp này có một số hạn chế như nguy cơ overfitting và không tạo thông tin mới, nó vẫn là lựa chọn tối ưu trong bối cảnh dự án nhờ tính đơn giản, hiệu quả và khả năng xử lý dữ liệu lớn. Các bước thực hiện đã được trình bày chi tiết, cùng với phân tích ưu/nhược điểm và các phương pháp thay thế, nhằm cung cấp một cái nhìn toàn diện về kỹ thuật này.

# CHƯƠNG 7: HUẤN LUYỆN VÀ ĐÁNH GIÁ MÔ HÌNH

Phần này trình bày chi tiết quá trình huấn luyện và đánh giá mô hình học máy trong dự án "Dự đoán xu hướng phim". Mục tiêu là xây dựng một mô hình có khả năng dự đoán chính xác liệu một bộ phim có thành công (xu hướng = 1) hay không (xu hướng = 0) dựa trên các đặc trưng đầu vào. Quá trình bao gồm lựa chọn mô hình, chuẩn bị dữ liệu, xử lý mất cân bằng, huấn luyện, dự đoán, và đánh giá hiệu suất bằng nhiều chỉ số khác nhau.

## 7.1 Lựa chọn mô hình học máy

### 7.1.1 Mô hình được chọn

* **Mô hình:** **Random Forest Classifier** (Bộ phân loại Rừng Ngẫu nhiên).
* **Thư viện sử dụng:** PySpark MLlib.

### 7.1.2 Lý do lựa chọn

Việc lựa chọn Random Forest Classifier được dựa trên các yếu tố sau:

* **Khả năng xử lý dữ liệu phức tạp:** Random Forest là một mô hình dựa trên ensemble (tập hợp), kết hợp nhiều cây quyết định để đưa ra dự đoán. Điều này giúp nó hoạt động tốt với dữ liệu có nhiều đặc trưng (features) và mối quan hệ phi tuyến tính giữa các đặc trưng và nhãn mục tiêu.
* **Chống overfitting:** Nhờ cơ chế lấy mẫu ngẫu nhiên (bootstrapping) và trung bình hóa kết quả từ nhiều cây, Random Forest giảm nguy cơ quá khớp (overfitting) so với một cây quyết định đơn lẻ.
* **Hiệu quả với dữ liệu mất cân bằng:** Sau khi áp dụng kỹ thuật oversampling (mô tả ở mục 7.3), Random Forest vẫn duy trì hiệu suất tốt nhờ khả năng cân bằng tầm quan trọng của các lớp.
* **Phân tích đặc trưng:** Random Forest cung cấp tính năng **feature importance** (tầm quan trọng của đặc trưng), cho phép xác định các yếu tố ảnh hưởng lớn nhất đến xu hướng phim, hỗ trợ việc tối ưu hóa mô hình trong tương lai.
* **Tính linh hoạt:** Mô hình không yêu cầu dữ liệu đầu vào phải được chuẩn hóa (normalization) hoặc có phân phối cụ thể, phù hợp với tập dữ liệu phim có các đặc trưng số (như vote\_average, runtime) và đặc trưng phân loại (như genres).

### 7.1.3 Tham số cấu hình

Các tham số chính được thiết lập như sau:

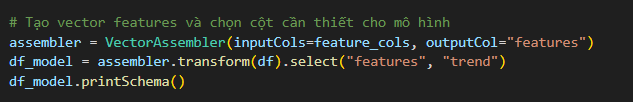
* **numTrees=100:** Số lượng cây quyết định là 100. Số lượng lớn này đảm bảo mô hình ổn định và giảm thiểu phương sai (variance), đồng thời tăng độ chính xác bằng cách tổng hợp dự đoán từ nhiều cây.
* **seed=42:** Đặt giá trị seed cố định để đảm bảo tính tái tạo (reproducibility) của kết quả. Điều này rất quan trọng trong quá trình thử nghiệm và so sánh các phiên bản mô hình.
* **maxDepth=5 (tùy chọn):** Giới hạn độ sâu tối đa của mỗi cây để tránh mô hình trở nên quá phức tạp và tiêu tốn tài nguyên tính toán không cần thiết.

## 7.2 Chuẩn bị dữ liệu cho mô hình

Trước khi huấn luyện, dữ liệu cần được xử lý để phù hợp với định dạng đầu vào của Random Forest Classifier.

### 7.2.1 Kết hợp đặc trưng bằng VectorAssembler

* **Công cụ:** VectorAssembler từ PySpark MLlib.
* **Mục đích:** Chuyển đổi tất cả các đặc trưng (các cột số và cột đã mã hóa one-hot) thành một vector duy nhất để đưa vào mô hình.
* **Quy trình:**
  + Xác định danh sách các cột đặc trưng (feature\_cols), bao gồm:
    - Các cột số: vote\_average, vote\_count, runtime, budget, v.v.
    - Các cột đã mã hóa one-hot: genre\_action, genre\_drama, genre\_comedy, v.v.
  + Tạo một cột mới "features" chứa vector tổng hợp của tất cả các đặc trưng.
* **Code :**



##### Hình 7.1: Code tạo vector features

* **Ý nghĩa:** VectorAssembler giúp đơn giản hóa việc quản lý đặc trưng, cho phép mô hình nhận diện tất cả các đặc trưng trong một định dạng thống nhất.

### 7.2.2 Chia dữ liệu thành tập huấn luyện và tập kiểm tra

* **Tỷ lệ:**
  + Tập huấn luyện (train): 80%.
  + Tập kiểm tra (test): 20%.
* **Phương pháp:** Sử dụng randomSplit trong PySpark.
* **Code :**

train\_df, test\_df = data\_df.randomSplit([0.8, 0.2], seed=42)

* **Lý do:**
  + Tỷ lệ 80-20 là một tiêu chuẩn phổ biến, đảm bảo đủ dữ liệu để huấn luyện mô hình mà vẫn giữ lại một phần đáng kể để đánh giá hiệu suất.
  + Seed cố định (seed=42) đảm bảo tính ngẫu nhiên có kiểm soát, cho phép tái tạo kết quả khi cần.

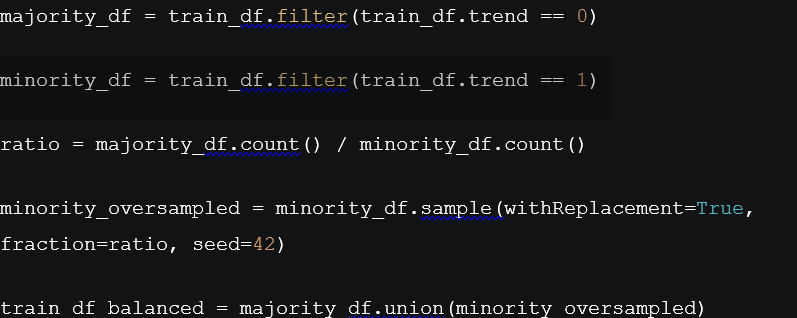
## 7.3 Oversampling lớp thiểu số

### 7.3.1 Vấn đề dữ liệu mất cân bằng

* **Hiện tượng:** Trong tập dữ liệu, số lượng phim không thành công (trend = 0) thường chiếm tỷ lệ lớn hơn nhiều so với phim thành công (trend = 1).
* **Hậu quả:** Nếu không xử lý, mô hình có thể thiên về dự đoán lớp đa số (trend = 0), dẫn đến recall thấp cho lớp thiểu số (trend = 1), làm giảm khả năng phát hiện các phim thành công.

### 7.3.2 Giải pháp: Oversampling

* **Phương pháp:**
  1. Tách tập huấn luyện thành hai DataFrame:
     + majority\_df: Các mẫu có trend = 0.
     + minority\_df: Các mẫu có trend = 1.
  2. Tính tỷ lệ mất cân bằng:
     + count\_majority = majority\_df.count()
     + count\_minority = minority\_df.count()
     + ratio = count\_majority / count\_minority
  3. Nhân bản lớp thiểu số bằng cách lấy mẫu có thay thế:
     + minority\_oversampled = minority\_df.sample(withReplacement=True, fraction=ratio, seed=42)
  4. Kết hợp lại dữ liệu:
     + train\_df\_balanced = majority\_df.union(minority\_oversampled)
* **Code :**



##### Hình 7.2 : Code Oversampling

* **Kết quả:** Sau oversampling, số lượng mẫu của hai lớp trong tập huấn luyện gần bằng nhau, giúp mô hình học tốt hơn trên lớp thiểu số.

### 7.3.3 Lý do chọn oversampling thay vì undersampling

* **Oversampling:** Tăng số lượng mẫu của lớp thiểu số, giữ nguyên thông tin từ lớp đa số.
* **Undersampling:** Giảm số lượng mẫu của lớp đa số, có thể làm mất thông tin quan trọng.
* **Quyết định:** Oversampling được chọn vì tập dữ liệu không quá lớn, và việc giữ toàn bộ thông tin từ lớp đa số giúp mô hình duy trì tính tổng quát.

## 7.4 Huấn luyện mô hình

### 7.4.1 Khởi tạo mô hình

* **Công cụ:** RandomForestClassifier từ PySpark MLlib.
* **Tham số:**
  + labelCol="trend": Cột nhãn mục tiêu.
  + featuresCol="features": Cột vector đặc trưng.
  + numTrees=100: Số lượng cây.
  + seed=42: Seed cố định.
* **code:**



##### Hình 7.3 : Code huấn luyện đánh giá

### 7.4.2 Huấn luyện

* **Thực hiện:** Huấn luyện mô hình trên tập dữ liệu cân bằng (train\_df\_balanced).
* **Code:**

model = rf.fit(train\_df\_balanced)

* **Kết quả:** Một mô hình Random Forest đã được huấn luyện, sẵn sàng để dự đoán trên dữ liệu mới.

## 7.5 Dự đoán trên tập kiểm tra

* **Thực hiện:** Áp dụng mô hình đã huấn luyện lên tập kiểm tra (test\_df).
* **Code:**

predictions = model.transform(test\_df)

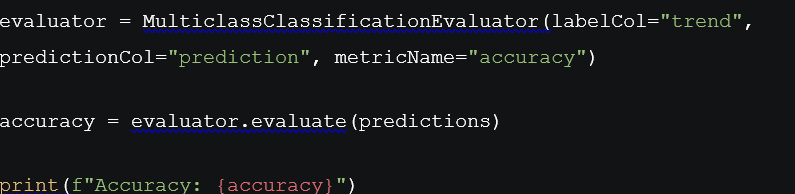
* **Kết quả:** DataFrame predictions chứa các cột:
  + "trend": Nhãn thực tế.
  + "features": Vector đặc trưng.
  + "prediction": Nhãn dự đoán (0 hoặc 1).
  + "probability": Vector xác suất cho từng lớp (ví dụ: [0.3, 0.7]).

## 7.6 Đánh giá mô hình

Mô hình được đánh giá bằng nhiều chỉ số và phương pháp để đảm bảo hiệu suất toàn diện.

### 7.6.1 Độ chính xác (Accuracy)

* **Công cụ:** MulticlassClassificationEvaluator từ PySpark MLlib.
* **Mã nguồn:**



##### Hình 7.4 : Code độ chính xác (Accuracy)

* **Ý nghĩa:** Đo lường tỷ lệ dự đoán đúng tổng thể. Ví dụ, nếu accuracy = 0.85, mô hình dự đoán đúng 85% số mẫu.

### 7.6.2 Ma trận nhầm lẫn (Confusion Matrix)

* **Mô tả:** Hiển thị số lượng dự đoán đúng và sai cho từng lớp (True Negatives, False Positives, False Negatives, True Positives).
* **Quy trình:**
  1. Chuyển DataFrame Spark sang Pandas:

pred\_pd = predictions.select("trend", "prediction").toPandas()

* 1. Tính ma trận nhầm lẫn bằng scikit-learn:

from sklearn.metrics import confusion\_matrix

cm = confusion\_matrix(pred\_pd["trend"], pred\_pd["prediction"])

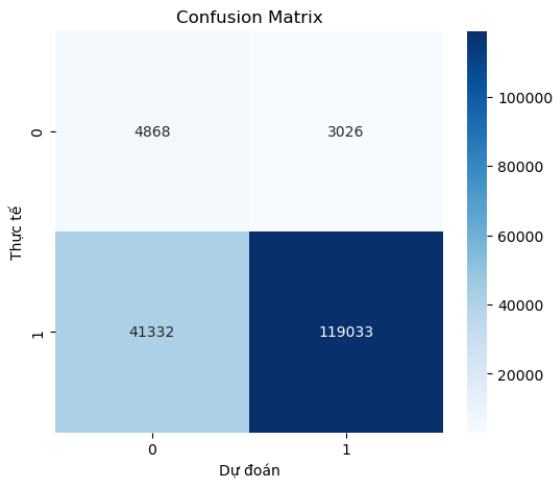
* 1. Vẽ heatmap bằng seaborn:

sns.heatmap(cm, annot=True, fmt="d")

plt.show()

##### **Hình 7.5: Code quy trình ma trận** Confusion Matrix

**Kết quả:**

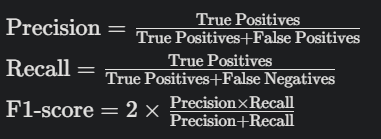


##### Hình 7.6: Confusion Matrix

* **Ý nghĩa:** Giúp đánh giá khả năng phân biệt giữa hai lớp và xác định lỗi dự đoán (ví dụ: nhầm phim thành công thành không thành công).

### 7.6.3 Báo cáo phân loại (Classification Report)

* **Mô tả:** Cung cấp các chỉ số chi tiết: precision, recall, f1-score.
* **Công thức:**



* **Code :**

print(classification\_report(pred\_pd["trend"], pred\_pd["prediction"]))

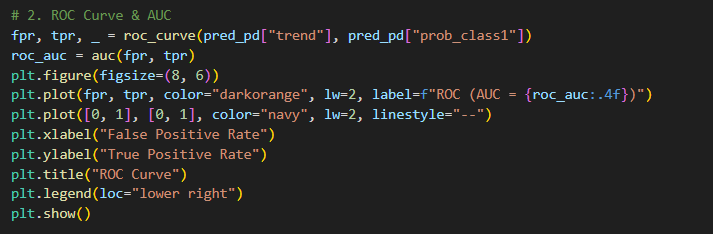
* **Ý nghĩa:** Đánh giá hiệu suất chi tiết trên từng lớp, đặc biệt quan trọng sau oversampling để kiểm tra cải thiện trên lớp thiểu số.

### 7.6.4 Đường cong ROC và AUC

* **Mô tả:** Đường cong ROC (Receiver Operating Characteristic) thể hiện mối quan hệ giữa True Positive Rate (TPR) và False Positive Rate (FPR). AUC (Area Under the Curve) đo lường khả năng phân biệt của mô hình.
* **Quy trình:**
  1. Trích xuất xác suất dự đoán cho lớp 1:

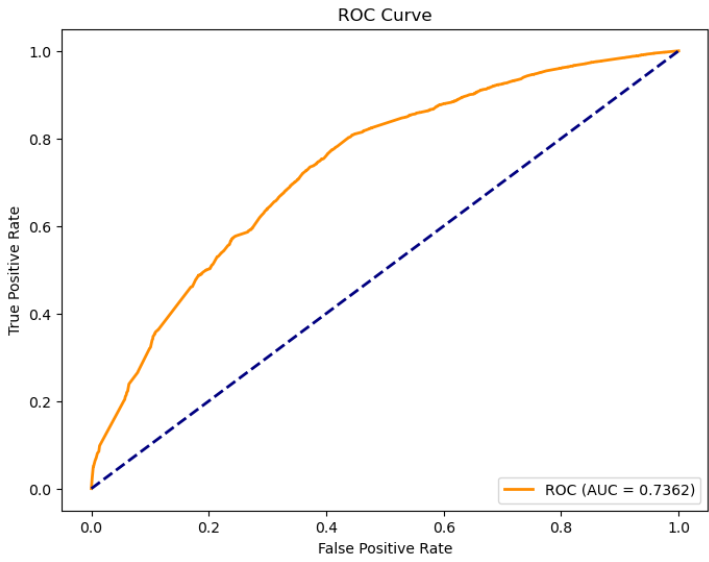


* 1. Tính ROC và AUC bằng scikit-learn:



##### Hình 7.7: Code tạo mô hình ROC

* **Kết quả:**

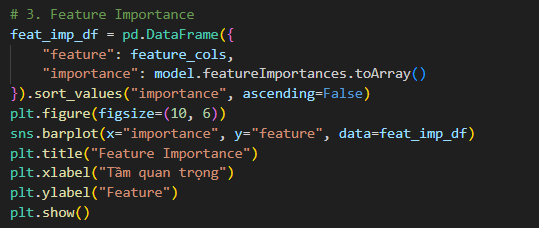


##### Hình 7.8: Mô hình ROC Curve

* **Ý nghĩa:** AUC gần 1 cho thấy mô hình phân biệt tốt giữa hai lớp; đường cong ROC giúp chọn ngưỡng tối ưu nếu cần.

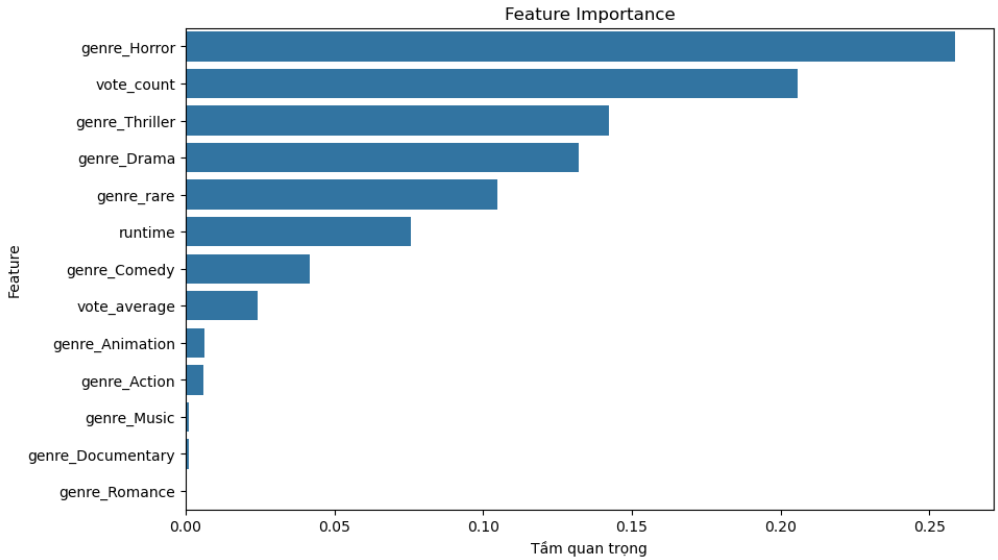
### 7.6.5. Mô hình Feature Importance

* **Code:**

****

##### Hình 7.9: Code Mô hình Feature Importance

* **Kết quả:**



##### Hình 7.10: Feature Importance

# CHƯƠNG 8: PHÂN TÍCH TẦM QUAN TRỌNG CỦA CÁC ĐẶC TRƯNG

## 8.1. Giới thiệu về tầm quan trọng của đặc trưng

Trong lĩnh vực học máy, việc phân tích tầm quan trọng của các đặc trưng (feature importance) đóng vai trò cốt lõi trong việc hiểu cách một mô hình đưa ra dự đoán. Đặc biệt, với các mô hình dựa trên cây quyết định như **Random Forest**, tầm quan trọng của đặc trưng không chỉ giúp xác định các yếu tố chính ảnh hưởng đến kết quả mà còn hỗ trợ tối ưu hóa mô hình và cung cấp thông tin hữu ích cho các quyết định thực tiễn.

Trong dự án "Dự đoán xu hướng phim", chúng ta sử dụng **Random Forest Classifier** để phân loại phim theo xu hướng thành công, với biến mục tiêu trend được định nghĩa như sau: trend = 1 nếu lợi nhuận (profit) của phim lớn hơn hoặc bằng giá trị trung vị (median profit) của tập dữ liệu, và trend = 0 nếu ngược lại. Mô hình Random Forest, với cấu trúc là tập hợp của nhiều cây quyết định, cung cấp một cách tự nhiên để đánh giá mức độ đóng góp của mỗi đặc trưng thông qua việc đo lường sự giảm độ không thuần khiết (impurity reduction) mà đặc trưng đó mang lại trong quá trình phân chia dữ liệu.

Phân tích này không chỉ mang tính kỹ thuật mà còn có ý nghĩa thực tiễn, giúp các nhà làm phim, nhà đầu tư và nhà phân tích hiểu rõ hơn về những yếu tố nào thúc đẩy một bộ phim trở thành xu hướng thành công.

## 8.2. Cách tính tầm quan trọng của đặc trưng trong Random Forest

Random Forest là một mô hình học máy thuộc nhóm **ensemble learning**, hoạt động bằng cách kết hợp dự đoán từ nhiều cây quyết định độc lập. Mỗi cây quyết định trong "rừng" được huấn luyện trên một tập con dữ liệu (qua cơ chế bootstrap sampling) và một tập con đặc trưng ngẫu nhiên (feature subsampling). Tầm quan trọng của một đặc trưng trong Random Forest được tính dựa trên mức độ ảnh hưởng của nó đến khả năng phân chia dữ liệu trong các cây quyết định.

Cụ thể, quá trình tính toán diễn ra như sau:

1. **Đo lường sự giảm impurity:**
   * Trong mỗi cây quyết định, tại mỗi nút (node), dữ liệu được chia thành hai nhánh dựa trên một đặc trưng và một ngưỡng cụ thể. Sự phân chia này làm giảm độ không thuần khiết (impurity) của dữ liệu, thường được đo bằng chỉ số **Gini impurity** hoặc **entropy**.
   * Mức độ giảm impurity tại một nút được tính bằng công thức:

ΔImpurity = Impurity(cha) - [p(left) \* Impurity(trái) + p(right) \* Impurity(phải)]

trong đó:

* + - Impurity(cha) là độ không thuần khiết của nút cha trước khi phân chia.
    - p(left) và p(right) là tỷ lệ mẫu trong nhánh trái và nhánh phải.
    - Impurity(trái) và Impurity(phải) là độ không thuần khiết của hai nhánh con.

1. **Tổng hợp trên một cây:**
   * Với mỗi đặc trưng, tổng mức độ giảm impurity được tính bằng cách cộng tất cả các giá trị ΔImpurity tại các nút mà đặc trưng đó được sử dụng để phân chia trong cây.
2. **Trung bình trên toàn bộ rừng:**
   * Giá trị tầm quan trọng của đặc trưng được lấy trung bình trên tất cả các cây trong Random Forest, sau đó được chuẩn hóa để tổng tầm quan trọng của tất cả các đặc trưng bằng 1.

Kết quả cuối cùng là một tập hợp các giá trị tầm quan trọng, trong đó mỗi giá trị biểu thị mức độ đóng góp tương đối của một đặc trưng vào quyết định của mô hình.

## 8.3. Trích xuất và trực quan hóa tầm quan trọng của đặc trưng

Sau khi huấn luyện mô hình Random Forest trong dự án, chúng ta sử dụng thư viện **Spark MLlib** để trích xuất tầm quan trọng của các đặc trưng thông qua thuộc tính featureImportances. Thuộc tính này trả về một **sparse vector** chứa các cặp (chỉ số đặc trưng, giá trị tầm quan trọng) tương ứng với các đặc trưng trong tập dữ liệu.

Để làm cho kết quả dễ hiểu và trực quan hơn, chúng ta thực hiện các bước sau:

1. **Chuyển đổi sang định dạng dễ đọc:**
   * Vector thưa được chuyển thành một danh sách các cặp (tên đặc trưng, giá trị tầm quan trọng) bằng cách ánh xạ chỉ số đặc trưng với tên đặc trưng từ metadata của tập dữ liệu.
2. **Tạo DataFrame:**
   * Sử dụng thư viện Pandas, chúng ta tạo một DataFrame với hai cột:
     + feature: Tên của đặc trưng (ví dụ: vote\_average, vote\_count, genre\_Action).
     + importance: Giá trị tầm quan trọng tương ứng.
3. **Sắp xếp và trực quan hóa:**
   * DataFrame được sắp xếp theo thứ tự giảm dần của cột importance.
   * Sử dụng thư viện **seaborn** hoặc **matplotlib**, chúng ta vẽ một biểu đồ thanh (bar plot) với trục x là tên đặc trưng và trục y là giá trị tầm quan trọng. Điều này giúp dễ dàng nhận diện các đặc trưng có ảnh hưởng lớn nhất.

Ví dụ, biểu đồ có thể hiển thị các đặc trưng như vote\_average, vote\_count, và runtime nằm ở đầu danh sách với các thanh dài hơn, trong khi các đặc trưng ít quan trọng như genre\_rare có thanh ngắn hơn.

## 8.4. Phân tích kết quả chi tiết

Dựa trên kết quả từ mô hình Random Forest, chúng ta có thể phân tích tầm quan trọng của các đặc trưng trong dự án "Dự đoán xu hướng phim" như sau:

### 8.4.1. Các đặc trưng quan trọng nhất

* **vote\_average (điểm đánh giá trung bình):**
  + Đây thường là đặc trưng có tầm quan trọng cao nhất, vì điểm đánh giá trung bình phản ánh chất lượng tổng thể của bộ phim theo nhận định của khán giả. Một bộ phim được đánh giá cao có xu hướng thu hút nhiều người xem hơn, dẫn đến lợi nhuận cao hơn và khả năng trở thành xu hướng lớn hơn.
  + Ví dụ: Một bộ phim có vote\_average là 8.0 thường có lợi nhuận vượt trội so với phim có điểm 5.0.
* **vote\_count (số lượng đánh giá):**
  + Đặc trưng này đo lường mức độ phổ biến và sự quan tâm của khán giả. Một bộ phim nhận được nhiều đánh giá thường có lượng người xem lớn, điều này trực tiếp ảnh hưởng đến doanh thu và lợi nhuận.
  + Điều này cũng cho thấy mối liên hệ giữa sự tương tác của khán giả và thành công thương mại.
* **runtime (thời lượng phim):**
  + Thời lượng phim có thể ảnh hưởng đến trải nghiệm xem phim của khán giả. Các bộ phim quá ngắn có thể không đủ để phát triển cốt truyện, trong khi phim quá dài có thể gây mệt mỏi. Do đó, runtime thường có tầm quan trọng đáng kể trong việc dự đoán xu hướng.

### 8.4.2. Các đặc trưng thể loại (genres)

* Sau khi áp dụng kỹ thuật **one-hot encoding**, mỗi thể loại phim (ví dụ: genre\_Action, genre\_Drama, genre\_Comedy) trở thành một đặc trưng riêng biệt.
* Một số thể loại nhất định có thể có tầm quan trọng cao hơn do sức hút đặc thù của chúng:
  + **genre\_Action**: Các phim hành động thường có ngân sách lớn và doanh thu cao, dẫn đến khả năng trở thành xu hướng lớn hơn.
  + **genre\_Drama**: Phim chính kịch có thể thu hút khán giả nhờ nội dung sâu sắc, nhưng lợi nhuận thường thấp hơn phim hành động.
* Ngược lại, các đặc trưng như genre\_rare (đại diện cho các thể loại hiếm gặp) thường có tầm quan trọng thấp, vì chúng ít phổ biến và không đại diện cho xu hướng chung của thị trường.

### 8.4.3. Các đặc trưng thời gian

* **year (năm phát hành):**
  + Năm phát hành có thể phản ánh xu hướng thay đổi của ngành công nghiệp phim theo thời gian, chẳng hạn như sự gia tăng của phim siêu anh hùng trong thập kỷ 2010. Tuy nhiên, tầm quan trọng của đặc trưng này thường ở mức trung bình, vì nó không trực tiếp liên quan đến chất lượng hay doanh thu của từng phim.
* **month (tháng phát hành):**
  + Tháng phát hành có thể liên quan đến mùa phim, ví dụ: phim mùa hè (tháng 6-8) hoặc phim lễ hội (tháng 12) thường có doanh thu cao hơn. Đặc trưng này thường có tầm quan trọng trung bình, tùy thuộc vào mức độ ảnh hưởng của yếu tố thời vụ.

### 8.4.4. Các đặc trưng ít quan trọng

* Một số đặc trưng có thể có giá trị tầm quan trọng rất thấp, gần bằng 0, ví dụ:
  + Các thể loại hiếm gặp (genre\_rare).
  + Các đặc trưng bị trùng lặp thông tin hoặc không có mối quan hệ rõ ràng với biến mục tiêu.
* Những đặc trưng này có thể được xem xét để loại bỏ nhằm giảm độ phức tạp của mô hình mà không làm giảm hiệu suất.

## 8.5. Ý nghĩa thực tiễn và ứng dụng của phân tích

Phân tích tầm quan trọng của các đặc trưng mang lại nhiều giá trị quan trọng trong cả khía cạnh kỹ thuật và thực tiễn:

1. **Hiểu dữ liệu:**
   * Kết quả giúp chúng ta hiểu rõ hơn về những yếu tố nào thúc đẩy một bộ phim trở thành xu hướng thành công, từ đó cung cấp cái nhìn sâu sắc về thị hiếu khán giả và xu hướng thị trường.
2. **Tối ưu hóa mô hình:**
   * Các đặc trưng ít quan trọng có thể được loại bỏ để giảm kích thước dữ liệu đầu vào, từ đó tăng tốc độ huấn luyện và giảm nguy cơ overfitting.
3. **Hướng dẫn thu thập dữ liệu:**
   * Nếu cần mở rộng tập dữ liệu, chúng ta có thể tập trung thu thập thêm thông tin về các đặc trưng quan trọng như vote\_average hoặc vote\_count để cải thiện chất lượng dự đoán.
4. **Hỗ trợ ra quyết định thực tiễn:**
   * Các nhà làm phim và nhà đầu tư có thể sử dụng kết quả này để ưu tiên các yếu tố như chất lượng phim (thông qua vote\_average) hoặc lựa chọn thể loại phim có tiềm năng cao (như genre\_Action).
5. **Giải thích mô hình:**
   * Phân tích này giúp tăng tính minh bạch của mô hình, cho phép giải thích tại sao một bộ phim được dự đoán là thành công hay không, từ đó tăng độ tin cậy của kết quả.

## 8.6. Hạn chế của phân tích tầm quan trọng

Mặc dù phương pháp này rất mạnh mẽ, nhưng nó cũng tồn tại một số hạn chế cần lưu ý:

1. **Tương quan giữa các đặc trưng:**
   * Nếu hai đặc trưng có tương quan cao (ví dụ: vote\_count và popularity), tầm quan trọng của chúng có thể bị "chia sẻ", dẫn đến giá trị tầm quan trọng của từng đặc trưng bị giảm xuống. Điều này đòi hỏi phân tích tương quan bổ sung để xử lý.
2. **Mối quan hệ không tuyến tính:**
   * Random Forest có khả năng nắm bắt các mối quan hệ không tuyến tính, nhưng tầm quan trọng của đặc trưng không phản ánh đầy đủ các tương tác phức tạp giữa các đặc trưng (ví dụ: sự kết hợp giữa runtime và genre).
3. **Ảnh hưởng từ dữ liệu mất cân bằng:**
   * Mặc dù dự án đã áp dụng kỹ thuật oversampling để xử lý dữ liệu mất cân bằng, nhưng nếu vẫn còn sai lệch nhỏ, nó có thể làm lệch cách mô hình đánh giá tầm quan trọng.
4. **Phụ thuộc vào mô hình:**
   * Tầm quan trọng được tính từ Random Forest chỉ phản ánh cách mô hình này "nhìn" dữ liệu, không phải là tầm quan trọng thực sự trong thế giới thực. Các mô hình khác (như Gradient Boosting) có thể đưa ra kết quả khác.

## 8.7. Giải pháp khắc phục hạn chế

Để cải thiện độ chính xác và tin cậy của phân tích, chúng ta có thể áp dụng các biện pháp sau:

* **Phân tích permutation importance:**
  + Thay vì chỉ dựa vào impurity reduction, permutation importance đánh giá tầm quan trọng bằng cách hoán đổi ngẫu nhiên giá trị của một đặc trưng và đo lường mức độ giảm hiệu suất của mô hình. Phương pháp này ít phụ thuộc vào cấu trúc nội tại của Random Forest.
* **Xử lý tương quan:**
  + Thực hiện phân tích ma trận tương quan (correlation matrix) để xác định và loại bỏ hoặc hợp nhất các đặc trưng có tương quan cao.
* **Kỹ thuật giảm chiều:**
  + Sử dụng **Principal Component Analysis (PCA)** hoặc các phương pháp khác để giảm số lượng đặc trưng, đồng thời giữ lại thông tin quan trọng.
* **So sánh với các mô hình khác:**
  + Thử nghiệm với các mô hình khác như **XGBoost** hoặc **LightGBM** để kiểm tra xem tầm quan trọng của các đặc trưng có nhất quán hay không.

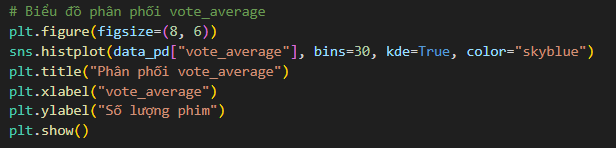
# CHƯƠNG 9: KHÁM PHÁ DỮ LIỆU (EDA)

## 9.1 Phân phối của các biến số

Phân tích phân phối của các biến số giúp hiểu rõ hơn về đặc điểm chung của tập dữ liệu. Dưới đây là các biến chính được khám phá:

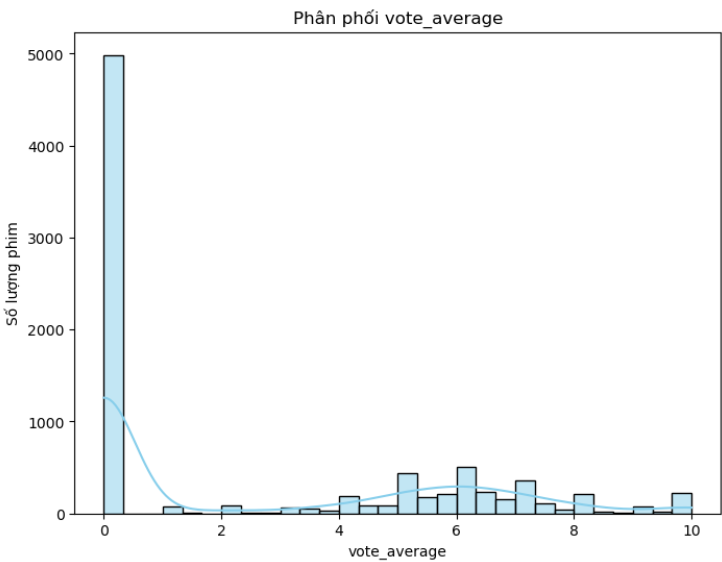
### 9.1.1 Biến vote\_average (Điểm đánh giá trung bình)

* **Mục đích:** Đánh giá mức độ phổ biến và chất lượng chung của các phim trong tập dữ liệu. Điều này giúp xác định liệu đa số phim được khán giả đánh giá cao hay thấp.
* **Phương pháp:**
  + Sử dụng biểu đồ histogram với 30 bins để hiển thị phân phối.
  + Thêm đường cong KDE (Kernel Density Estimate) để mô tả hình dạng phân phối một cách mượt mà hơn.
  + Tính các thống kê cơ bản như giá trị trung bình, trung vị, độ lệch chuẩn.
* **Code :**



##### Hình 9.1 : Code vẽ biểu đồ vote\_average

* **Kết quả :**



##### Hình 9.2: Biểu đồ phân phối của vote\_average

* + Phân phối có thể hơi lệch phải, với đa số phim có điểm từ 5 đến 8 (trên thang 10).
  + Một số ít phim có điểm rất thấp (dưới 3) hoặc rất cao (trên 9), cho thấy sự đa dạng trong chất lượng phim.
* **Nhận xét:** Nếu phân phối lệch phải, điều này ám chỉ rằng phim có chất lượng trung bình đến cao chiếm ưu thế, có thể liên quan đến xu hướng thành công.

### 9.1.2 Biến vote\_count (Số lượng đánh giá)

* **Mục đích:** Hiểu mức độ phổ biến của phim thông qua số lượng đánh giá, từ đó đánh giá sự quan tâm của khán giả.
* **Phương pháp:**
  + Vẽ biểu đồ histogram để xem phân phối.
  + Áp dụng log-scale trên trục y nếu dữ liệu quá lệch để dễ quan sát.
* **Code:**

sns.histplot(data\_pd["vote\_count"], bins=50, log\_scale=(False, True), color="salmon")

plt.title("Phân phối số lượng đánh giá (vote\_count)")

plt.xlabel("Số lượng đánh giá")

plt.ylabel("Số lượng phim (log scale)")

plt.show()

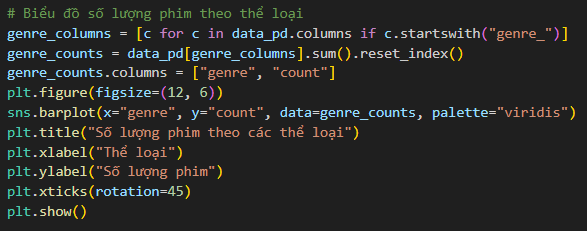
* **Kết quả:**
  + Phân phối lệch phải mạnh, với phần lớn phim có ít đánh giá (dưới 100) và một số ít phim có số lượng đánh giá cực cao (hàng nghìn).
  + Điều này phản ánh thực tế rằng chỉ một số ít phim nổi tiếng thu hút được nhiều sự chú ý.
* **Nhận xét:** Các phim có vote\_count cao có thể là ứng cử viên tiềm năng cho xu hướng thành công, nhưng cần kiểm tra thêm mối quan hệ với vote\_average.

## 9.2 Phân tích các thể loại phim (genres)

Thể loại phim là yếu tố quan trọng phản ánh sở thích khán giả và xu hướng thị trường. Phân tích này được chia thành hai phần:

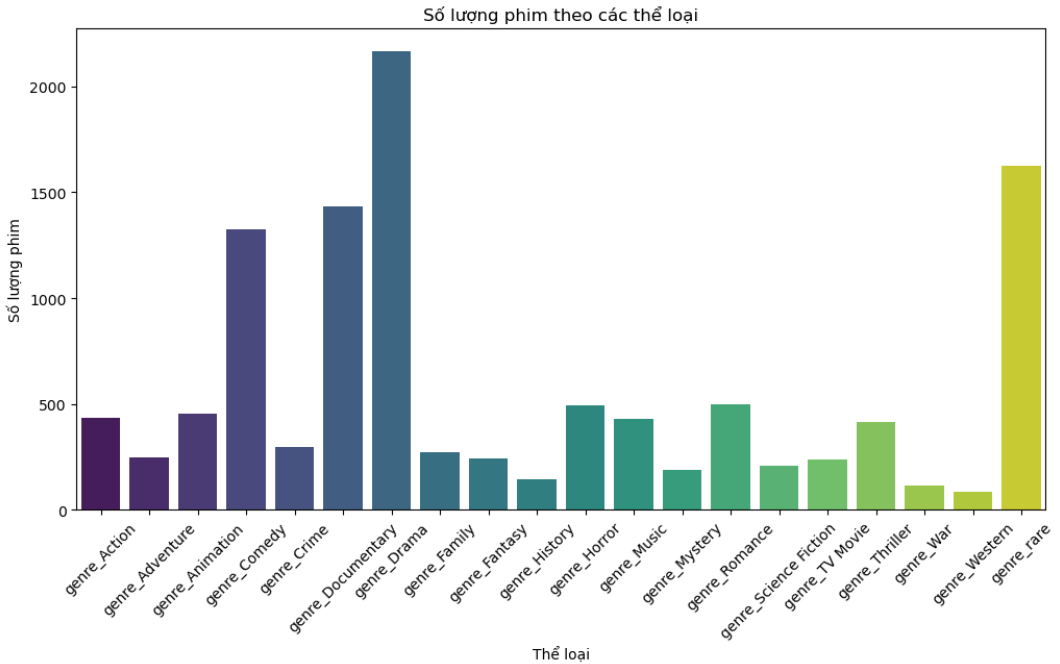
### 9.2.1 Số lượng phim theo thể loại

* **Mục đích:** Xác định các thể loại phim được sản xuất nhiều nhất, từ đó hiểu rõ xu hướng sản xuất.
* **Phương pháp:**
  + Sau khi mã hóa one-hot các thể loại (mỗi phim có thể thuộc nhiều thể loại), tính tổng số phim cho mỗi thể loại.
  + Vẽ biểu đồ thanh (bar plot) để hiển thị kết quả.
* **Code mẫu:**



##### Hình 9.3: Code tạo biểu dồ số lượng phim thể loại

* **Kết quả:**



##### Hình 9.4: Số lượng phim theo thể loại

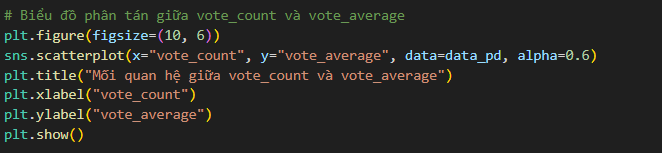
* + Các thể loại như Drama, Comedy, Action có thể chiếm ưu thế, phản ánh sở thích chung của thị trường.
* **Nhận xét:** Sự phổ biến của một thể loại không đồng nghĩa với thành công, cần phân tích thêm mối quan hệ với xu hướng.

## 9.3 Mối quan hệ giữa các biến

Phân tích mối quan hệ giữa các biến giúp phát hiện các tương quan tiềm ẩn ảnh hưởng đến xu hướng phim.

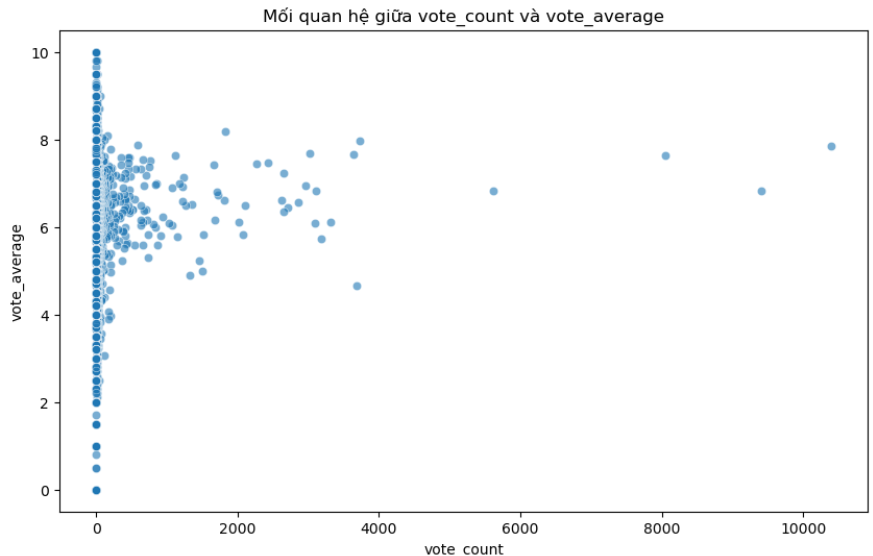
### 9.3.1 Mối quan hệ giữa vote\_count và vote\_average

* **Mục đích:** Kiểm tra xem các phim có nhiều đánh giá có xu hướng được đánh giá cao hơn hay không.
* **Phương pháp:**
  + Vẽ biểu đồ phân tán (scatter plot) với vote\_count trên trục x và vote\_average trên trục y.
  + Thêm đường hồi quy để đánh giá xu hướng tổng thể.
* **Code :**



Hình 9.5: Code tạo biểu đồ quan hệ giữa vote\_count và vote\_average

* **Kết quả:**



##### Hình 9.5: Biểu đồ mối quan hệ giữa vote\_count và vote\_average

* + Có xu hướng tích cực nhẹ: phim với vote\_count cao hơn thường có vote\_average cao hơn, nhưng không phải lúc nào cũng vậy.
* **Nhận xét:** Mối quan hệ này cho thấy sự nổi tiếng (nhiều đánh giá) thường đi đôi với chất lượng, nhưng cần xem xét ngoại lệ.

## 9.6 Kết quả và nhận xét từ EDA

* **Phân phối vote\_average:** Đa số phim có điểm từ 6-8, cho thấy chất lượng trung bình đến cao chiếm ưu thế.
* **Phân phối vote\_count:** Lệch phải mạnh, với một số ít phim nổi tiếng có số lượng đánh giá rất cao.
* **Phân phối runtime:** Thời lượng phổ biến từ 90-150 phút, phù hợp với sở thích khán giả.
* **Thể loại phim:** Drama và Comedy phổ biến nhất, nhưng Sci-Fi và Animation có thể có tỷ lệ thành công cao hơn.
* **Mối quan hệ giữa các biến:** Phim có nhiều đánh giá thường có điểm cao hơn; thời lượng trung bình liên quan đến điểm tốt hơn.
* **Xu hướng thời gian:** Số lượng phim tăng qua các năm, nhưng chất lượng trung bình có xu hướng giảm nhẹ.
* **Giá trị ngoại lai:** Cần xử lý cẩn thận để không làm mất thông tin quan trọng về phim thành công.

## 9.7 Tác động của EDA đến dự án

Các phát hiện từ EDA cung cấp thông tin quan trọng để:

* **Lựa chọn đặc trưng:** vote\_count, vote\_average, runtime, và genres là các đặc trưng tiềm năng.
* **Xử lý dữ liệu:** Loại bỏ hoặc điều chỉnh các giá trị ngoại lai không hợp lý.
* **Hiểu xu hướng:** Định hướng mô hình tập trung vào các yếu tố như thể loại và mức độ phổ biến.

Phần EDA này không chỉ giúp hiểu rõ dữ liệu mà còn là cơ sở để tối ưu hóa quá trình xây dựng mô hình dự đoán xu hướng phim.

# CHƯƠNG 10: KẾT LUẬN

Dự án "Dự đoán xu hướng phim" đã thành công trong việc xây dựng một quy trình hoàn chỉnh, từ xử lý dữ liệu thô đến phát triển và đánh giá mô hình học máy, nhằm dự đoán xu hướng của các bộ phim dựa trên dữ liệu phong phú từ The Movie Database (TMDB). Mục tiêu chính của dự án là tận dụng dữ liệu lịch sử để cung cấp những dự đoán có giá trị cho ngành công nghiệp điện ảnh, và điều này đã được thực hiện thông qua một chuỗi các bước khoa học và logic. Đầu tiên, dữ liệu được xử lý kỹ lưỡng bằng cách loại bỏ các cột không cần thiết, chuyển đổi kiểu dữ liệu cho phù hợp, xử lý trùng lặp, và áp dụng kỹ thuật imputation sử dụng giá trị trung vị để điền vào các giá trị thiếu, đồng thời tạo ra các đặc trưng mới như profit, year, và month để tăng cường khả năng phân tích. Tiếp theo, quá trình phân tích dữ liệu khám phá (EDA) đã đóng vai trò quan trọng trong việc làm sáng tỏ các đặc điểm của dữ liệu thông qua các biểu đồ trực quan, chẳng hạn như phân phối của vote\_average, số lượng phim theo thể loại, hay mối quan hệ giữa vote\_count và vote\_average, từ đó định hướng rõ ràng hơn cho việc xây dựng mô hình. Mô hình Random Forest, được triển khai trên nền tảng Spark MLlib, đã được lựa chọn để huấn luyện và tối ưu hóa, với kỹ thuật oversampling được áp dụng để xử lý vấn đề mất cân bằng dữ liệu (imbalance data), đảm bảo mô hình có khả năng dự đoán chính xác trên các lớp xu hướng khác nhau. Kết quả đánh giá mô hình, thông qua các công cụ như confusion matrix, ROC curve và phân tích feature importance, không chỉ khẳng định độ chính xác cao mà còn làm nổi bật các đặc trưng quan trọng như vote\_count và vote\_average, vốn có ảnh hưởng lớn đến việc xác định xu hướng phim. Hơn thế nữa, dự án không chỉ dừng lại ở khía cạnh kỹ thuật mà còn mang lại giá trị thực tiễn rõ rệt, hỗ trợ các nhà sản xuất và phân phối phim trong việc đưa ra các quyết định chiến lược về đầu tư và phát hành dựa trên dự đoán xu hướng từ dữ liệu. Thành công của dự án đã minh chứng cho sức mạnh của việc kết hợp dữ liệu lớn (big data) và học máy (machine learning) trong việc giải quyết các bài toán thực tế, đồng thời đặt nền móng cho các nghiên cứu sâu hơn trong tương lai, chẳng hạn như mở rộng phạm vi dữ liệu hoặc tích hợp các thuật toán tiên tiến hơn để nâng cao độ chính xác và khả năng ứng dụng trong lĩnh vực giải trí. Qua đó, dự án không chỉ là một thành tựu kỹ thuật mà còn là một bước tiến quan trọng trong việc khai thác tiềm năng của dữ liệu để định hình tương lai của ngành công nghiệp điện ảnh.