|  |  |
| --- | --- |
| **bỘ GIÁO DỤC VÀ ĐÀO TẠO** | **BỘ NÔNG NGHIỆP VÀ PTNT** |

TRƯỜNG ĐẠI HỌC THỦY LỢI



nGUYỄN dUY TÂN

Dự đoán xếp hạng mì ramen

bằng mô hình học máy

ĐỒ ÁN TỐT NGHIỆP

HÀ NỘI, NĂM 2024

|  |  |
| --- | --- |
| **bỘ GIÁO DỤC VÀ ĐÀO TẠO** | **BỘ NÔNG NGHIỆP VÀ PTNT** |

TRƯỜNG ĐẠI HỌC THỦY LỢI

NGUYỄN DUY TÂN

Dự đoán xếp hạng mì ramen

bằng mô hình học máy

|  |  |
| --- | --- |
| Ngành : | Hệ thống thông tin |
| Mã số: | 7480104 |

|  |  |
| --- | --- |
| NGƯỜI HƯỚNG DẪN | **TS. Tạ Quang Chiểu** |
|  |  |

HÀ NỘI, NĂM 2024

**GÁY BÌA ĐỒ ÁN TỐT NGHIỆP, KHÓA LUẬN TỐT NGHIỆP**

**NGUYỄN DUY TÂN ĐỒ ÁN TỐT NGHIỆP HÀ NỘI, NĂM 2024**

LỜI CAM ĐOAN

Tác giả xin cam đoan đây là Đồ án tốt nghiệp của bản thân. Tất cả các kết quả trong Đồ án tốt nghiệp này là trung thực, và không sao chép từ bất kỳ một nguồn nào và dưới bất kỳ hình thức nào. Việc tham khảo các nguồn tài liệu đã được thực hiện trích dẫn và ghi nguồn tài liệu tham khảo đúng quy định.

|  |  |
| --- | --- |
|  | **Tác giả ĐATN**    **Nguyễn Duy Tân** |

LỜI CÁM ƠN

Trước hết, em muốn bày tỏ lòng biết ơn chân thành đến đội ngũ giảng viên tại Khoa Công nghệ thông tin của trường Đại học Thủy Lợi. Sự chỉ dẫn và hỗ trợ nhiệt tình từ phía thầy cô trong suốt quãng thời gian học đã đóng góp quan trọng, giúp em tích lũy đủ kiến thức và kỹ năng cần thiết để hoàn thành khóa học cũng như đồ án tốt nghiệp của mình.

Em muốn gửi lời biết ơn chân thành đến tất cả những người đã đồng hành và hỗ trợ em trong suốt quá trình học tập. Đặc biệt, em muốn gửi lời cảm ơn sâu sắc đến Thầy TS. Tạ Quang Chiểu với sự tận tâm và hướng dẫn sát sao đã giúp em vượt qua những khó khăn và hoàn thành đồ án tốt nghiệp một cách thành công. Em rất trân trọng và biết ơn sự hỗ trợ nhiệt tình từ Thầy.

Tiếp theo, em muốn gửi lời biết ơn sâu sắc đến tất cả bạn bè, anh chị em đã luôn đồng hành và hỗ trợ em trong suốt quãng thời gian học tập và nghiên cứu. Sự chia sẻ kiến thức và kinh nghiệm, cùng những ý kiến đóng góp quý báu từ mọi người, thực sự là nguồn động viên không thể thiếu giúp em vượt qua mọi thách thức.

Mặc dù em đã cố gắng tập trung để hoàn thành đồ án tốt nghiệp theo khả năng của mình, nhưng vẫn tồn tại nhiều thiếu sót. Em chân thành nhận được sự đồng cảm và những đóng góp xây dựng từ phía các thầy cô và bạn bè, giúp em có cơ hội hiểu biết sâu hơn và phát triển từ những trải nghiệm học tập này.

Em xin chân thành cảm ơn!

MỤC LỤC

[DANH MỤC CÁC HÌNH ẢNH v](#_Toc169651974)

[DANH MỤC BẢNG BIỂU vi](#_Toc169651975)

[DANH MỤC CÁC TỪ VIẾT TẮT VÀ GIẢI THÍCH CÁC THUẬT NGỮ vii](#_Toc169651976)

[CHƯƠNG 1 GIỚI THIỆU BÀI TOÁN, GIỚI THIỆU TỔNG QUAN 1](#_Toc169651977)

[1.1 Lý do chọn đề tài 1](#_Toc169651978)

[1.2 Mục tiêu nghiên cứu 1](#_Toc169651979)

[1.3 Đối tượng và phạm vi nghiên cứu 2](#_Toc169651980)

[CHƯƠNG 2: CÁC KỸ THUẬT DÙNG TRONG BÀI TOÁN 3](#_Toc169651981)

[2.1 Cơ sở lý thuyết chung của học máy 3](#_Toc169651982)

[2.1.1 Học có giám sát (Supervised Learning) 4](#_Toc169651983)

[2.1.2 Học không giám sát (Unsupervised Learning) 5](#_Toc169651984)

[2.1.3 Học sâu (Deep learning) 6](#_Toc169651985)

[2.2 Kỹ thuật dùng trong bài toán 7](#_Toc169651986)

[2.2.1 Hồi quy Logistic 7](#_Toc169651987)

[2.2.2 DecisionTreeClassifier 8](#_Toc169651988)

[2.2.4 RandomForestClassifier 10](#_Toc169651989)

[2.3 Đánh giá mô hình 11](#_Toc169651990)

[2.3.1. Accuracy 11](#_Toc169651991)

[2.3.2. Precision 11](#_Toc169651992)

[2.3.3. Recall 12](#_Toc169651993)

[2.3.4. F1-score 12](#_Toc169651994)

[CHƯƠNG 3: ỨNG DỤNG PHƯƠNG PHÁP VÀ XÂY DỰNG MÔ HÌNH 13](#_Toc169651995)

[3.1 Mô tả bài toán 13](#_Toc169651996)

[3.2 Phân tích dữ liệu 13](#_Toc169651997)

[3.2.1 Tiền xử lý dữ liệu 14](#_Toc169651998)

[3.2.2 Phân tích dữ liệu 22](#_Toc169651999)

[3.3 Công cụ sử dụng 24](#_Toc169652000)

[3.4 Xây dựng mô hình 25](#_Toc169652001)

[**CHƯƠNG 4: KẾT LUẬN** 32](#_Toc169652002)

DANH MỤC CÁC HÌNH ẢNH

[Hình 2.1 Sơ đồ lý thuyết học máy 4](#_Toc169652426)

[Hình 2.2 Sơ đồ hoạt động hồi quy logistic 7](#_Toc169652427)

[Hình 2.3 Minh họa về thuật toán DecisionTreeClassifier 9](#_Toc169652428)

[Hình 2.4 Minh họa về cách hoạt động của thuật toán RandomForestClassifier 10](#_Toc169652429)

[Hình 3.1 Dữ liệu ban đầu 13](#_Toc169652430)

[Hình 3.2 Kiểm tra giá trị null trong từng cột 14](#_Toc169652431)

[Hình 3.3 Dữ liệu mới sau khi loại bỏ null và 2 cột Review # và T 14](#_Toc169652432)

[Hình 3.4 Dữ liệu sau khi phân tích cột IsSpicy 16](#_Toc169652433)

[Hình 3.5 Dữ liệu sau khi trích xuất token từ thuộc tính Variety 19](#_Toc169652434)

[Hình 3.6 Các biến giả được tạo ra từ cột Brand 20](#_Toc169652435)

[Hình 3.7 Các biến giả được tạo ra từ cột Style 20](#_Toc169652436)

[Hình 3.8 Các biến giả được tạo ra từ cột Country 21](#_Toc169652437)

[Hình 3.9 Dữ liệu cuối cùng thu được 22](#_Toc169652438)

DANH MỤC BẢNG BIỂU

[Bảng 2.1 Cách định dạng lề giấy **Error! Bookmark not defined.**](file:///D:\Box\My%20Box%20Files\Quan%20Ly%20NCS\Huong%20dan%20trinh%20bay%20LATS_updated%2001-2013\Huong%20dan%20trinh%20bay%20LATS.docx#_Toc405384698)

[Bảng 2.2 Tóm tắt các kiểu định dạng (style) cho các đề mục **Error! Bookmark not defined.**](file:///D:\Box\My%20Box%20Files\Quan%20Ly%20NCS\Huong%20dan%20trinh%20bay%20LATS_updated%2001-2013\Huong%20dan%20trinh%20bay%20LATS.docx#_Toc405384699)

DANH MỤC CÁC TỪ VIẾT TẮT VÀ GIẢI THÍCH CÁC THUẬT NGỮ

|  |  |
| --- | --- |
| **Từ viết tắt** | **Viết đầy đủ** |
| **ĐATN** | Đồ án tốt nghiệp |
| **AI** | Artificial Intelligence |

# GIỚI THIỆU BÀI TOÁN, GIỚI THIỆU TỔNG QUAN

## Lý do chọn đề tài

Mì ramen hay được biết tới với cái tên khác là mì ăn liền, đã trở thành một phần không thể thiếu trong ẩm thực hàng ngày của nhiều người trên khắp thế giới. Tuy nhiên, mì ramen hay loại mì ăn liền được yêu thích vì sự thuận tiện, giá cả phải chăng và hương vị đa dạng. Để lựa chọn giữa các thương hiệu mì khác nhau thường là việc khó khăn cho người tiêu dùng.

Từ đây đặt ra một câu hỏi quan trọng: Điều gì tạo nên sự khác biệt của một loại mì ramen so với các loại mì khác? Liệu có thể xác định các yếu tố cụ thể nào làm cho một thương hiệu mì ramen nổi bật hơn và có hương vị tốt hơn so với các thương hiệu khác không?

Trong đồ án này, em tiến hành nghiên cứu các kỹ thuật học máy để dự đoán xem một loại mì ramen đó sẽ được xem là "tốt" hay "không tốt" dựa trên các yếu tố như loại sản phẩm, kiểu mì, thương hiệu và quốc gia sản xuất. Vì vậy em thực hiện ĐATN về đề tài: “Dự đoán xếp hạng mì ramen bằng mô hình học máy”.

## Mục tiêu nghiên cứu

Ứng dụng xếp hạng mì ramen cung cấp thông tin hữu ích cho người tiêu dùng, Điều này giúp người tiêu dùng có cái nhìn tổng quan về hương vị của sản phẩm và dễ dàng tìm kiếm các loại mì ramen phù hợp với sở thích của họ.

Trong lĩnh vực kinh doanh thực phẩm, chủ các nhà hàng có thể dựa vào kết quả nghiên cứu để tìm ra loại mì phù hợp cho tình hình kinh tế của nhà hàng mà vẫn tạo nên trải nghiệm tốt nhất cho người dùng

Trong lĩnh vực chăm sóc khách hàng, nguồn dữ liệu đánh giá đến từ chính những người tiêu dùng từ đó phân tích và dự đoán sở thích cá nhân của từng người dùng. Điều này giúp cho nhà sản xuất hiểu rõ hơn về khẩu vị và sở thích cá nhân của mỗi người tiêu dùng để cải thiện các loại mì ramen phù hợp với người tiêu dùng.

Trong lĩnh vực quảng cáo, dữ liệu thu thập được có thể sử dụng để phân tích sở thích và nhu cầu của khách hàng từ đó tạo ra các chiến dịch quảng cáo hiệu quả hơn.

## Đối tượng và phạm vi nghiên cứu

Đối tượng nghiên cứu: Là xếp hạng của các loại mì ramen, tìm ra loại mì có star cao

Phạm vi nghiên cứu: Các loại mì ramen của các nhà sản xuất trên khắp thế giới

CHƯƠNG 2: CÁC KỸ THUẬT DÙNG TRONG BÀI TOÁN

## 2.1 Cơ sở lý thuyết chung của học máy

Trí tuệ nhân tạo (AI) là một lĩnh vực của khoa học máy tính tập trung vào việc tạo ra các hệ thống máy tính có khả năng thực hiện các nhiệm vụ đòi hỏi trí thông minh của con người. AI bao gồm nhiều lĩnh vực con như học máy (machine learning), xử lý ngôn ngữ tự nhiên (natural language processing), thị giác máy tính (computer vision), và robot học (robotics). Mục tiêu chính của AI là phát triển các hệ thống có khả năng tư duy, học hỏi, và đưa ra quyết định một cách tự động.

AI có thể được phân thành hai loại chính: AI yếu (narrow AI) và AI mạnh (general AI). AI yếu được thiết kế để thực hiện một nhiệm vụ cụ thể, chẳng hạn như nhận diện khuôn mặt hoặc dự đoán xu hướng thị trường. AI mạnh, tuy vẫn còn trong giai đoạn nghiên cứu, được kỳ vọng có khả năng thực hiện bất kỳ nhiệm vụ trí tuệ nào mà con người có thể làm. Khi nhắc tới AI thì không thể bỏ qua khái niệm cực kỳ nổi bật đó là học máy Học máy, hay tên gọi tiếng Anh là machine learning, là một nhánh con quan trọng của AI, tập trung vào việc phát triển các thuật toán và mô hình cho phép máy tính học hỏi từ dữ liệu. Thay vì được lập trình rõ ràng cho từng nhiệm vụ, các hệ thống học máy tự động cải thiện hiệu suất thông qua việc học từ kinh nghiệm và dữ liệu. Học máy là một phần của AI, đóng vai trò quan trọng trong việc hiện thực hóa các hệ thống AI thông minh. Trong nhiều ứng dụng AI hiện đại, học máy được sử dụng để tạo ra các mô hình có khả năng dự đoán, phân loại, và ra quyết định tự động. Nhờ đó [1] [2], AI có thể phân tích và hiểu dữ liệu phức tạp, từ đó thực hiện các nhiệm vụ mà trước đây chỉ con người mới có thể làm được.

Trong lĩnh vực học máy, có ba phương pháp chính được sử dụng rộng rãi: học có giám sát (Supervised learning), học không giám sát (Unsupervised learning), và học tăng cường (Reinforcement learning). Mỗi phương pháp đều có những đặc điểm và ứng dụng riêng, giúp giải quyết các vấn đề khác nhau.

Học có giám sát là phương pháp phổ biến nhất. Trong phương pháp này, mô hình được huấn luyện trên một tập dữ liệu đã gắn nhãn, tức là mỗi dữ liệu đầu vào (input) đi kèm với một nhãn đầu ra (output). Mục tiêu là học cách ánh xạ từ đầu vào đến đầu ra dựa trên các cặp (data, label).

Học không giám sát không yêu cầu dữ liệu phải được gắn nhãn. Thay vào đó, nó khám phá các cấu trúc tiềm ẩn trong dữ liệu. Học không giám sát thường được sử dụng cho các bài toán phân cụm (clustering) và giảm số chiều (dimensionality reduction).

Học tăng cường là phương pháp trong đó một tác nhân (agent) học cách hành động trong một môi trường để tối đa hóa phần thưởng tích lũy. Mô hình này học qua việc tương tác với môi trường và nhận phản hồi dưới dạng phần thưởng hoặc hình phạt.

Trí tuệ nhân tạo (AI) và học máy (Machine Learning) đang mở ra những khả năng mới và thay đổi cách chúng ta tiếp cận các vấn đề phức tạp. Học máy, như một nhánh quan trọng của AI, đang ngày càng chứng tỏ vai trò không thể thiếu trong việc phân tích và khai thác dữ liệu. Từ việc hỗ trợ chẩn đoán y tế đến tối ưu hóa kinh doanh, học máy đang giúp cải thiện hiệu suất và hiệu quả trong nhiều lĩnh vực. Tuy nhiên, để khai thác tối đa tiềm năng của học máy, cần có sự đầu tư liên tục vào nghiên cứu, phát triển và quản lý dữ liệu một cách hiệu quả. Học máy không chỉ là công cụ hỗ trợ mà còn là động lực thúc đẩy sự đổi mới và phát triển trong tương lai.[1]

Hình 2.1A diagram of a graph

Description automatically generated with medium confidence Sơ đồ lý thuyết học máy

### 2.1.1 Học có giám sát (Supervised Learning)

Học có giám sát là phương pháp thông dụng nhất trong học máy, ở học có giám sát bộ dữ liệu được gắn nhãn đưa vào mô hình có sẵn, sau khi huấn luyện ta thu được các kết quả đầu ra. Trong học có giám sát, mỗi mẫu dữ liệu đầu vào được đại diện bằng x và mục tiêu dự đoán được đại diện bằng y. Mục tiêu y thường là một nhãn hoặc giá trị mong muốn để dự đoán.

Phương pháp học có giám sát thường được phân ra hai loại chính là Classification (Phân loại) và Regression (Hồi quy).

Phân loại (Classification): Trong bài toán phân loại, đó là quá trình gắn nhãn cho dữ liệu và từng phân lớp hoặc các nhóm khác nhau dự vào các đặc điểm của nó. Mục tiêu là xây dựng một mô hình học máy có khả năng dự đoán nhãn (lớp) của các dữ liệu mới dựa trên các đặc trưng đầu vào. Ví dụ, trong bài toán xác định xem một email có phải là spam hay không, thuật toán phân loại sẽ được sử dụng để dự đoán nhãn "spam" hoặc "non-spam" cho từng email.

Hồi quy (Regression): Trong học máy, hồi quy là một loại mô hình dự đoán giá trị của một biến liên tục dựa trên các biến độc lập. Nó được sử dụng để mô hình hóa và dự đoán mối quan hệ giữa các biến. Mục tiêu là xây dựng một mô hình học máy có khả năng dự đoán giá trị số cho các dữ liệu mới dựa trên các đặc trưng đầu vào. Ví dụ, trong bài toán dự đoán giá của một căn nhà dựa trên diện tích, số phòng ngủ và khoảng cách đến trung tâm thành phố, thuật toán hồi quy sẽ được sử dụng để dự đoán giá trị số là giá của căn nhà.[3]

### 2.1.2 Học không giám sát (Unsupervised Learning)

Học không giám sát (unsupervised learning) là một lĩnh vực quan trọng trong học máy, nơi mà mô hình được huấn luyện trên một tập dữ liệu không có nhãn. Điều này có nghĩa là các đầu ra mong muốn không được cung cấp trong quá trình huấn luyện, buộc mô hình phải tự tìm kiếm và xác định các mẫu và cấu trúc tiềm ẩn trong dữ liệu. Học không giám sát đóng vai trò then chốt trong việc khai thác thông tin từ dữ liệu phi cấu trúc và đã được ứng dụng rộng rãi trong nhiều lĩnh vực khác nhau như xử lý ngôn ngữ tự nhiên, phân tích thị trường, và sinh học.

Cụ thể, hai loại chính trong học không giám sát là:

Clustering (phân cụm): Phân cụm là quá trình chia dữ liệu thành các nhóm sao cho các điểm dữ liệu trong cùng một nhóm có nhiều điểm chung với nhau. Đây là một kỹ thuật mạnh mẽ để khám phá các cấu trúc ẩn trong dữ liệu. Ví dụ, trong phân tích thị trường, phân cụm có thể được sử dụng để phân loại khách hàng dựa trên hành vi mua sắm của họ, từ đó hỗ trợ việc phát triển chiến lược tiếp thị hiệu quả.

Không những thế phân cụm còn hỗ trợ trong việc giảm chiều dữ liệu và phát hiện các ngoại lệ. Chẳng hạn, trong sinh học, phân cụm có thể giúp xác định các nhóm gene có chức năng tương tự, hoặc trong bảo mật mạng, nó có thể được sử dụng để phát hiện các hoạt động bất thường.

Association (Luật kết hợp): Luật kết hợp tập trung vào việc tìm kiếm các quy luật hoặc mối liên hệ giữa các biến trong tập dữ liệu. Thuật toán Apriori và FP-Growth là những công cụ phổ biến trong lĩnh vực này. Ví dụ: Từ trong giỏ hàng mua sắm của khách hàng, nơi các nhà bán lẻ có thể phát hiện ra những mặt hàng nào thường được mua cùng nhau, từ đó tối ưu hóa việc sắp xếp hàng hóa và các chương trình khuyến mãi. Ngoài ra, khai thác luật kết hợp còn có ứng dụng trong nhiều lĩnh vực khác như phân tích văn bản, nơi nó có thể giúp tìm ra các từ thường xuất hiện cùng nhau trong các tài liệu, hoặc trong y học, để xác định mối liên hệ giữa các triệu chứng và bệnh tật.

### 2.1.3 Học sâu (Deep learning)

* Khái niệm

Học sâu (Deep Learning) là một lĩnh vực then chốt trong học máy, được phát triển dựa trên mô hình và hoạt động của mạng nơ-ron nhân tạo (Artificial Neural Networks). Ý tưởng cốt lõi của học sâu là thiết lập các mạng nơ-ron với nhiều tầng ẩn (hidden layers) để thu nhận và xử lý thông tin từ dữ liệu đầu vào một cách hiệu quả.

Trong học sâu các mạng nơ-ron được thiết kế với kiến trúc phức tạp, cho phép việc học hỏi dữ liệu thông qua nhiều lớp nơ-ron được kết nối liên tiếp. Mỗi lớp nơ-ron trong mạng có khả năng nhận diện các đặc trưng trừu tượng từ dữ liệu đầu vào, bắt đầu từ những đặc trưng cơ bản đến những đặc trưng phức tạp hơn.

Các mô hình học sâu như Convolutional Neural Networks (CNNs) và Recurrent Neural Networks (RNNs) đã chứng minh được hiệu quả cao trong việc giải quyết các bài toán nhận dạng hình ảnh, nhận dạng giọng nói, và xử lý ngôn ngữ tự nhiên với độ phức tạp và yêu cầu độ chính xác cao. Khả năng tự động học từ dữ liệu lớn mà không cần sự can thiệp quá nhiều từ con người đã mở ra nhiều ứng dụng mới, từ xe tự lái đến chẩn đoán y khoa. Tương lai của học sâu hứa hẹn sẽ tiếp tục mang lại những đột phá mới, góp phần nâng cao hiệu quả và độ chính xác trong nhiều lĩnh vực khác nhau.

***2.2 Kỹ thuật dùng trong bài toán***

### 2.2.1 Hồi quy Logistic

* Khái niệm

Hồi quy logistic là một kỹ thuật phân tích dữ liệu sử dụng toán học để tìm ra mối quan hệ giữa hai yếu tố dữ liệu. Sau đó, nó sử dụng mối quan hệ này để dự đoán giá trị của một trong những yếu tố đó dựa trên yếu tố còn lại. Dự đoán này thường có một số kết quả hữu hạn, như có hoặc không.

Ta có phương trình tuyến tính của mô hình:

Trong đó:

A diagram of a graph

Description automatically generated

Hình 2.2 Sơ đồ hoạt động hồi quy logistic

* Hàm mất mát

Hàm mất mát (loss function) là một hàm toán học được sử dụng để đo lường mức độ sai lệch giữa giá trị dự đoán và giá trị thực tế trong một bài toán học máy. Nhiệm vụ của nó là đưa ra một đánh giá chất lượng của mô hình dự đoán.

Trong phương trình trên:

* L(w) là hàm mất mát, là đại lượng cần tối thiểu hóa
* w là tham số của mô hình hồi quy
* n là số lượng mẫu dữ liệu trong tập huấn luyện.
* 𝑦𝑖yi là giá trị thực tế của đầu ra thứ i.
* 𝑝𝑖pi là xác suất dự đoán thuộc lớp 1 của mô hình cho đầu vào thứ i.

Giá trị của hàm số L(w) càng bé thì mô hình hồi quy càng tốt. Để tối ưu hóa cho hàm số w thì ta có thể sử dụng thuật toán gradient descent. Thuật toán này hoạt động bằng cách bắt đầu với một giá trị ban đầu của tham số w và sau đó liên tục cập nhật tham số này cho đến khi hàm mất mát đạt đến một giá trị nhỏ nhất.

### 2.2.2 DecisionTreeClassifier

* **Khái niệm:**

DecisionTreeClassifier là một thuật toán học có giám sát được sử dụng trong học máy để phân loại dữ liệu. Thuật toán này hoạt động bằng cách phân chia tập dữ liệu thành các nhóm con nhỏ hơn dựa trên các thuộc tính của dữ liệu. Quy trình này được lặp lại đệ quy cho đến khi đạt được các nhóm con thuần nhất hoặc đáp ứng một số điều kiện dừng nhất định. Kết quả cuối cùng là một cấu trúc cây quyết định, trong đó mỗi nút biểu diễn một thuộc tính của dữ liệu, mỗi nhánh biểu diễn một kết quả có thể có của thuộc tính đó, và mỗi lá cây biểu diễn một nhãn hoặc giá trị quyết định.



Hình 2.3 Minh họa về thuật toán DecisionTreeClassifier

* **Ưu điểm:**
* **Dễ hiểu và trực quan:** Cây quyết định dễ dàng giải thích và trực quan hóa, giúp người dùng dễ dàng hiểu được cách thức hoạt động của mô hình.
* **Không cần giả định về phân phối dữ liệu:** Thuật toán không dựa vào các giả định về phân phối của các biến đầu vào.
* **Có thể xử lý cả dữ liệu số và dữ liệu phân loại:** DecisionTreeClassifier có thể làm việc với cả các thuộc tính dạng số và dạng phân loại mà không cần chuyển đổi.
* **Nhược điểm:**
* **Dễ bị overfitting:** Cây quyết định có thể dễ dàng bị overfitting nếu không được kiểm soát độ sâu hoặc không có các kỹ thuật làm giảm độ phức tạp của cây.
* **Không ổn định với dữ liệu biến đổi:** Một thay đổi nhỏ trong dữ liệu có thể dẫn đến sự thay đổi lớn trong cấu trúc của cây quyết định, làm cho mô hình không ổn định.
* **Thiên vị các thuộc tính có nhiều mức:** Cây quyết định có xu hướng ưa thích các thuộc tính có nhiều mức phân chia hơn, điều này có thể dẫn đến các cây không tối ưu.
* **Hiệu suất kém với dữ liệu lớn:** Khi làm việc với các tập dữ liệu rất lớn hoặc các thuộc tính có nhiều giá trị, cây quyết định có thể trở nên rất phức tạp và kém hiệu quả.

### 2.2.4 RandomForestClassifier

Đúng như tên gọi của thuật toán, Random là ngẫu nhiên, Forest là rừng, ở thuật toán này nó kết hợp nhiều cây quyết định (decision trees) thành một "rừng" (forest) và dựa trên nguyên tắc số đông để đưa ra dự đoán cuối cùng. Mỗi cây quyết định trong rừng được huấn luyện trên một mẫu ngẫu nhiên của dữ liệu, và cây đó đưa ra một dự đoán sau đó chọn dự đoán xuất hiện nhiều nhất trong số các dự đoán của các cây để đưa ra kết quả cuối cùng.

A diagram of a tree

Description automatically generated

Hình 2.4 Minh họa về cách hoạt động của thuật toán RandomForestClassifier

Ưu Điểm:

* **Hiệu suất cao và độ chính xác:** Thường có độ chính xác cao hơn so với một cây quyết định đơn lẻ.
* **Khả năng chống overfitting:** Do kết hợp nhiều cây quyết định với nhau, RandomForest ít bị ảnh hưởng bởi overfitting.
* **Xử lý dữ liệu mất cân bằng:** Có thể xử lý tốt các tập dữ liệu có số lượng lớp không cân bằng.
* **Xử lý dữ liệu đa dạng:** Có khả năng làm việc với cả dữ liệu phân loại và dữ liệu liên tục.
* **Tính năng quan trọng:** Cung cấp thông tin về tầm quan trọng của các đặc trưng nào ảnh hưởng nhiều nhất đến kết quả dự đoán.

Nhược Điểm:

* **Tính toán và tài nguyên:** Yêu cầu nhiều tài nguyên tính toán và bộ nhớ.
* **Diễn giải kết quả phức tạp:** Kết quả của RandomForest khó diễn giải hơn so với các thuật toán đơn giản như cây quyết định hoặc hồi quy tuyến tính.
* **Khả năng dự đoán không tốt trên dữ liệu mới:** Mặc dù giảm thiểu được overfitting, nhưng đôi khi RandomForest có thể không dự đoán tốt trên dữ liệu hoàn toàn mới nếu mẫu huấn luyện không đủ lớn.

## *2.3 Đánh giá mô hình*

2.3.1. Accuracy

Đây là một độ đo của bài toán phân lớp đơn giản nhất. Tính độ chính xác bằng cách sử dụng số kết quả dự đoán chính xác chia cho toàn bộ số mẫu dự đoán. Có nhiều phương pháp để tìm ra độ chính xác của một model. Trong đó thường được sử dụng nhất là phương pháp Accuracy

Khi giá trị Accuracy cao cho thấy mô hình có khả năng phân loại chính xác các mẫu dữ liệu và ngược lại khi giá trị Accuracy thấpta thấy mô hình đang phân loại sai nhiều mẫu dữ liệu.

Nhược điểm của Accuracy là chỉ cung cấp một tỉ lệ phần trăm chính xác tổng thể của mô hình trên toàn bộ tập dữ liệu mà không chỉ ra được độ chính xác của mỗi mẫu riêng lẻ. Điều này có thể dẫn tới việc không phản ánh đúng về hiệu suất của mô hình trên các trường hợp cụ thể, đặc biệt khi tập dữ liệu mất cân bằng. Độ chính xác cũng không thể đo lường được sự chênh lệch của mô hình giữa các lớp dữ liệu, khiến cho một mô hình có độ chính xác cao tổng thể vẫn có thể dự đoán kém trên các lớp thiểu số.

2.3.2. Precision

Precision là phương pháp đánh giá mô hình thể hiện tỷ lệ các dự đoán **đúng** trong số các dự đoán được phân loại là **positive**. Nói cách khác, nó cho ta biết mức độ tin cậy của các kết quả positive mà mô hình đưa ra.

Công thức tính Precision:

Precision = TP / (TP + FP)

* TP (True Positive): Tổng số trường hợp dự báo khớp Positive.
* TN (True Negative): Tổng số trường hợp dự báo khớp Negative.
* FP (False Positive): Tổng số trường hợp dự báo các quan sát thuộc nhãn Negative thành Positive.

2.3.3. Recall

Cũng tương tự với Precision, chỉ khác ở mẫu số công thức, Recall là phương pháp đo lường tỷ lệ dự báo chính xác các trường hợp positive trên toàn bộ mẫu thuộc nhóm positive

Công thức tính Recall:

Recall = TP / (TP + FN)

* FN (False Negative): Tổng số trường hợp dự báo các quan sát thuộc nhãn Positive thành Negative.

2.3.4. F1-score

Từ mối quan hệ của Precision và Recall, F1-score chính là trung bình của 2 phương pháp trên, giúp đánh giá tổng hiệu suất của mô hình. Nó được sử dụng khi ta quan tâm đến **cả hai** việc dự đoán true positive và tránh dự đoán False Negative.

F1-score **không phải lúc nào cũng là lựa chọn tốt nhất.** Trong một số trường hợp, ta có thể quan trọng hơn một chỉ số so với chỉ số kia, ví dụ như Recall cao hơn Precision. Khi đó, ta nên sử dụng chỉ số **phù hợp** với mục tiêu cụ thể của bài toán. [3]

# CHƯƠNG 3: ỨNG DỤNG PHƯƠNG PHÁP VÀ XÂY DỰNG MÔ HÌNH

## *3.1 Mô tả bài toán*

Chương trình xếp hạng mì ramen là một phần mềm được thiết kế trong mục đích xác định chất lượng dựa trên những đặc điểm cơ bản của một loại mì như mùi vị, cách đóng gói,… Từ đó ta có thể lựa chọn ra loại mì có rating cao mà mình phù hợp với sở thích và khả năng tài chính của mình.

## *3.2 Phân tích dữ liệu*

Dữ liệu bao gồm 7 cột thuộc tính bao gồm

1. Reviews #: Review ID(Integer)
2. Brand: To which brand the ramen is associated with (Object)
3. Style: Style of Ramen['Cup','Box','Tray','Bowl','Pack'] (Categorical)
4. Variety: Variety of the ramen associated with the brand(object)
5. Country: Name of the Country Sold in(object)
6. Ratings: Rated on scale of 1-5(object)
7. Top Ten: In which and rank the ramen was in Top ten(object)

A screenshot of a computer

Description automatically generated

Hình 3.15 Dữ liệu ban đầu

Nhiệm vụ là xếp hạng từng loại mì xem đó là mì có rating cao hay thấp, . Bộ dữ liệu tổng cộng 4650 bản ghi và được tổng hợp bởi trang [THE RAMEN RATER](https://www.theramenrater.com/)

3.2.1 Tiền xử lý dữ liệu

Dữ liệu ban đầu gồm có 4650 bản ghi và gồm 7 cột

3.2.1.1 Kiểm tra dữ liệu

Sử dụng câu lệnh df.isnull().sum() để kiểm tra xem dữ liệu có bị thiếu hay không?

**A screenshot of a computer

Description automatically generated**

Hình 3.26 Kiểm tra giá trị null trong từng cột

Xóa giá trị null có trong cột Stars và xóa luôn cột không cần thiết là cột Review #(chỉ là cột mang thứ tự)

A screenshot of a menu

Description automatically generated

Hình 3.37 Dữ liệu mới sau khi loại bỏ null và 2 cột Review # và T

**Tiến hành làm sạch dữ liệu trong cột Stars**

Kiểm tra dữ liệu có trong cột Stars

A black screen with white text

Description automatically generated

Nhận thấy không chỉ có riêng các giá trị là số mà còn một số giá trị không xác định

Chuyển dữ liệu trong cột về kiểu numeric và loại bỏ các giá trị không xác định

df['Stars']=pd.to\_numeric(df['Stars'], errors='coerce')

df = df.dropna(subset=['Stars'])

df.to\_csv('ramen\_rating\_2023\_clean.csv', index=None)

Sau quá trình xử lý trên, toàn bộ dữ liệu đã được làm sạch, loại bỏ cột không cần thiết xuất ra file dữ liệu csv mới và sẵn sàng cho quá trình tiếp theo.

3.2.1.2 Bổ sung các trường dữ liệu

Việc bổ sung các trường dữ liệu thiếu được thực hiện bằng cách trích xuất thông tin từ những cột dữ liệu đã có sẵn thay vì chỉ sử dụng những dữ liệu đã có hiện tại để tạo mô hình. Ví dụ từ cột Variety ta có thể tìm ra được đó là loại mì cay hay không dựa vào tên của từng loại mì. Thêm nữa, ta sẽ tìm những giá trị nổi trội ở các trường dữ liệu Brand, Country để chọn ra thêm nhiều cột dữ liệu khác để tăng độ lớn của dữ liệu.

Ta tiến hành trích xuất thông tin từ cột Variety với các thuộc tính sau:

* IsSpicy: có vị cay
* HasChicken: có gà
* HasBeef: có bò
* HasSeafoods: có hải sản

Bước đầu tiên, ta kết hợp các phần tử có trong danh sách Variety bằng câu lệnh

variety\_text = " ".join(variety)

đồng thời chuyển chuỗi thu được về dạng lower(chữ thường)

variety\_text\_lower = variety\_text.lower()

Tiếp theo lọc ra những ký tự đặc biệt để loại bỏ chúng

punctuation\_list = list(string.punctuation)

variety\_lower\_clean = "".join([char for char in variety\_text\_lower if char not in punctuation\_list])

Tách chuỗi trên thành các token đơn lẻ

variety\_token = nltk.tokenize.word\_tokenize(variety\_lower\_clean)

Sau khi phân tích các token đơn lẻ trên thì dựa vào các từ có liên quan đến “cay” có trong tên các loại mì theo ngôn ngữ của các quốc gia khác nhau như "spicy", "curry", "hot", "tom", "chilli", "chili", "kari", "pedas", "laksa". Với những mì có vị cay thì giá trị của mì đó sẽ là 1, và ngược lại nếu không có giá trị sẽ là 0.

is\_spicy = []

spicy\_flavor = ["spicy", "curry", "hot", "tom", "chilli", "chili", "kari", "pedas", "laksa"]

for variety in df["Variety"].str.lower():

    if any(flavor in variety for flavor in spicy\_flavor):

        is\_spicy.append(1)

    else:

        is\_spicy.append(0)

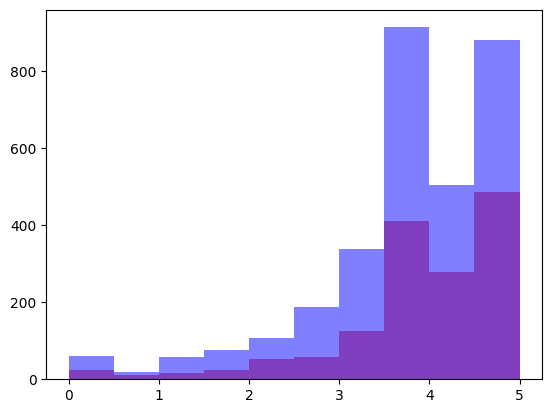
df["IsSpicy"] = is\_spicy

A screenshot of a menu

Description automatically generated

Hình 3.48 Dữ liệu sau khi phân tích cột IsSpicy

A diagram of a chart

Description automatically generated with medium confidence

Mức rating mà mì có vị cay “IsSpicy” cao hơn rõ rệt so với mì không cay

Tương tự như vậy, tiếp tục dựa vào các token để thêm các trường dữ liệu khác. Ở đây ta sử dụng hàm collections.Counter để xem tần xuất xuất hiện của từng token. Từ đó có thể dựa vào các token có tần xuất xuất hiện lớn để trích xuất ra các thuộc tính quan trọng khác.

variety\_counters = collections.Counter(variety\_token)

variety\_counters

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| noodles | noodle | ramen | flavor | instant | flavour | chicken | spicy | soup | beef | |
| 1047 | 976 | 897 | 657 | 637 | 580 | 533 | 532 | 427 | 359 | |
| cup | with | sauce | hot | curry | rice | artificial | shrimp | mi | tom | seafood |
| 294 | 289 | 278 | 246 | 224 | 217 | 181 | 177 | 172 | 169 | 163 |

*Bảng một số token có tần xuất xuất hiện lớn*

Nhận thấy ‘chicken’, ‘beef’, ‘seafood’ có tần xuất xuất hiện lớn nên ta sẽ tìm thêm các từ đồng nghĩa hoặc các từ mang ý nghĩa tương tự để trích ra thuộc tính quan trọng. Ở phần này với số lượng token lớn và hầu hết đều là tiếng nước ngoài nên em đã sử dụng chatgpt để lọc ra những token mang ý nghĩa tương tự với thuộc tính cần chọn lọc. Lợi ích khi sử dụng chatgpt trong phần chọn lọc này là với sự đa dạng về ngôn ngữ, chatgpt có thể tìm ra được các token có ý nghĩa tương tự trong thời gian ngắn mà vẫn vô cùng chính xác, có thể lọc được các token bị viết sai chính tả.

Tạo thêm thuộc tính ‘HasChicken’ với các token như "artificial", "chicken", "ayam", "pollo", "gallina"

has\_chicken = []

chicken\_flavor = ["artificial", "chicken", "ayam", "pollo", "gallina"]

for variety in df["Variety"].str.lower():

    if any(flavor in variety for flavor in chicken\_flavor):

        has\_chicken.append(1)

    else:

        has\_chicken.append(0)

df["HasChicken"] = has\_chicken

Tạo thêm thuộc tính ‘HasBeef với các token như "beef", "cow", "meat", "steak", "rib", "barbecue", "brisket","sirloin","tenderloin", "ribeye", "shank"

beef\_flavor = ["beef", "cow", "meat", "steak", "rib", "barbecue", "brisket","sirloin","tenderloin", "ribeye", "shank"]

for variety in df["Variety"].str.lower():

    if any(flavor in variety for flavor in beef\_flavor):

        has\_beef.append(1)

    else:

        has\_beef.append(0)

df["HasBeef"] = has\_beef

Tạo thêm thuộc tính ‘HasSeafoods’ với các token như "seafood", "fish", "crab", "shrimp", "prawn", "tom", "lobster", "scallop", "clam", "abalone", "oyster", "squid", "seaweed", "sushi", "ebi", "caviar"

has\_seafoods = []

seafoods\_flavor = ["seafood", "fish", "crab", "shrimp", "prawn", "tom", "lobster", "scallop", "clam", "abalone", "oyster", "squid", "seaweed", "sushi", "ebi", "caviar"]

for variety in df["Variety"].str.lower():

    if any(flavor in variety for flavor in seafoods\_flavor):

        has\_seafoods.append(1)

    else:

        has\_seafoods.append(0)

df["HasSeafoods"] = has\_seafoods

df

Dữ liệu sau khi trích xuất token từ thuộc tính Variety như sau:

A screenshot of a computer

Description automatically generated

Hình 3.59 Dữ liệu sau khi trích xuất token từ thuộc tính Variety

*3.4.1.2 Tạo biến giả*

Từ cột Brand chứa thông tin về các thương hiệu khác nhau. Mục tiêu hiện tại là tạo ra các cột mới phân loại các brand thành “TopBrand” và “Other” sau đó tạo các biến giả cho 2 loại này. Các biến giả được tạo cho các danh mục này để sử dụng trong phân tích hoặc mô hình hóa, với một danh mục bị loại bỏ để tránh Multicollinearity. Multicollinearity là hiện tượng xảy ra khi các biến độc lập trong một mô hình hồi quy tuyến tính có mối tương quan cao với nhau. Hiện tượng này có thể dẫn đến việc các khoảng tin cậy rộng hơn, làm giảm độ tin cậy của các ước tính về ảnh hưởng của các biến độc lập trong mô hình. Việc phát hiện và xử lý multicollinearity là rất quan trọng để đảm bảo mô hình hoạt động chính xác và hiệu quả. [5]

Phân loại các Brand hiện tại thành TopBrand và Other. Tạo các biến giả cho 2 loại này với các giá trị là 0 và 1

#create new column with name TopBrand

top\_brand = df['Brand'].value\_counts()[:22]

new\_brand = []

for brand in df['Brand']:

    if brand in top\_brand:

        new\_brand.append(brand)

    else:

        new\_brand.append("Other")

df['TopBrand'] = new\_brand

brand\_dummy = pd.get\_dummies(df['TopBrand'], prefix="from")

brand\_dummy = brand\_dummy.astype(int)

A screenshot of a computer

Description automatically generated

Hình 3.610 Các biến giả được tạo ra từ cột Brand

Tiếp tục tạo biến giả với cột Style khi chia các giá trị có trong cột Style thành “TopStyle” và “Other”. Ở đây ta sẽ lựa chọn 4 Style có tần xuất đứng đầu là TopStyle. Các cột biến giả chứa các giá trị số nguyên (0 hoặc 1) thay vì giá trị kiểu float mặc định.

top\_style = df['Style'].value\_counts()[:4]

new\_style = []

for style in df['Style']:

    if style in top\_style:

        new\_style.append(style)

    else:

        new\_style.append("Other")

df['TopStyle'] = new\_style

style\_dummy = pd.get\_dummies(df['TopStyle'], prefix="With")

style\_dummy = style\_dummy.astype(int)

A screenshot of a table

Description automatically generated

Hình 3.711 Các biến giả được tạo ra từ cột Style

Tiếp tục tạo biến giả với cột Style khi chia các giá trị có trong cột Style thành “TopCountry” và “Other”. Ở đây ta sẽ lựa chọn 11 Country có tần xuất cao nhất là TopCountry.

top\_country = df['Country'].value\_counts()[:11]

new\_country = []

for country in df['Country']:

    if country in top\_country:

        new\_country.append(country)

    else:

        new\_country.append("Other")

df['TopCountry'] = new\_country

country\_dummy = pd.get\_dummies(df['TopCountry'], prefix="In")

country\_dummy = country\_dummy.astype(int)

A screenshot of a computer

Description automatically generated

Hình 3.812 Các biến giả được tạo ra từ cột Country

Tập dữ liệu sau khi được xử lý thêm các cột biến giả vừa tạo và xóa đi những cột không cần thiết còn lại, ta thu được bộ dữ liệu bao gồm 4632 dữ liệu với 44 cột

df\_temp = df.drop(['Brand', 'Variety', 'Style', 'Country', 'TopBrand', 'TopStyle', 'TopCountry'], axis=1)

df\_final = pd.concat([df\_temp, style\_dummy,brand\_dummy, country\_dummy ], axis=1)

A screenshot of a computer

Description automatically generated

Hình 3.913 Dữ liệu cuối cùng thu được

### 3.2.2 Phân tích dữ liệu

**Vẽ biểu đồ trực quan dữ liệu**

Phân tích mối liên hệ của Styles với Stars (Ramen Ratings)

A graph with numbers and lines

Description automatically generated

Dựa vào biểu đồ, 2 giá trị Can và Bar có quá ít dữ liệu để hiển thị, ngược lại rating của Box và Pack lại rất cao. Với Box phần lớn giá trị rating trải từ 4 đến 5 trong đó trung vị rất cao, tuy nhiên cũng có những loại mì ngoại lệ trong khoảng từ 0 đến 2 rating. Với giá trị Pack phần lớn giá trị rating nằm trong khoảng 3.5 đến 4.75 với trung vị 4.

Phân tích trường dữ liệu Country

Country là cột thể hiện nguồn gốc của các loại mì có trong dữ liệu.



Biểu đồ thể hiện số lượng mì có ở các quốc gia, số loại mì trên thế giới chủ yếu đến từ 4 quốc gia Japan, US, South Korea, Taiwan. Đáng chú ý số lượng lớn đến từ các quốc gia châu Á, thể hiện sự đa dạng trong ẩm thực châu Á so với thế giới.



Mì đến từ các quốc gia châu Á Taiwan, Malaysia, Indonesia, Singapore là những loại mì được ưa chuộng và có rating cao chủ yếu từ 3.5 đến 5.

Biểu đồ boxplot

## *3.3 Công cụ sử dụng*

**Môi trường thực nghiệm:**

* Ngôn ngữ: Python
* Nền tảng phát triển: Jupyter Notebook
* Công cụ: Visual Studio Code
* Thiết bị: Window 11, Ram 16GB, Intel(R) Core (TM) i5-11400H

**Thư viện sử dụng chính:**

**Pandas và Numpy:**

Pandas là một thư viện Python cung cấp các cấu trúc dữ liệu và công cụ phân tích dữ liệu mạnh mẽ, đặc biệt là cho việc thao tác và phân tích dữ liệu dạng bảng (DataFrame). Numpy là một thư viện toán học cung cấp thao tác với các mảng (arrays) và ma trận (matrices) cùng với các hàm toán học để vận hành trên chúng.

**Sklearn:**

Sklearn Là một thư viện học máy trong Python cung cấp các công cụ học máy hiệu quả và dễ sử dụng để phân tích dữ liệu và xây dựng mô hình dự đoán.

**Matplotlib và Seaborn:**

Cả Matplotlib và Seaborn đều là những thư viện dùng để trực quan hóa dữ liệu trong Python. Matplotlib cung cấp các biểu đồ cơ bản, trong khi Seaborn xây dựng trên Matplotlib, cung cấp các giao diện đơn giản hơn và các kiểu biểu đồ phong phú hơn, giúp dễ dàng tạo ra các biểu đồ trực quan hóa dữ liệu phức tạp và đẹp mắt hơn.

**Nltk:**

Nltk là một thư viện xử lý ngôn ngữ tự nhiên của python cung cấp nhiều kho dữ liệu văn bản khác nhau cùng nhiều chức năng để xử lý văn bản phù hợp với nhiều mục đích sử dụng từ đó nâng cao hiệu quả, chất lượng của các mô hình thuật toán.[3]

**Streamlit:**

Streamlit là một framework web được viết bằng Python giúp người dùng tạo ứng dụng web một cách nhanh chóng và dễ dàng. Người dùng chỉ cần viết code Python và sử dụng các thành phần có sẵn của Streamlit để xây dựng giao diện người dùng và xử lý dữ liệu.

Một số widget của Streamlit trong Python

|  |  |
| --- | --- |
| Widget | Mô tả |
| st.button() | Tạo một nút nhấn. |
| st.checkbox() | Tạo một ô checkbox. |
| st.selectbox() | Tạo một hộp dropdown cho việc chọn từ một danh sách các lựa chọn. |
| st.multiselect() | Tạo một hộp dropdown cho phép chọn nhiều lựa chọn từ một danh sách. |
| st.text\_input() | Tạo một ô nhập liệu văn bản. |
| st.number\_input() | Tạo một ô nhập liệu cho các giá trị số. |
| st.date\_input() và st.time\_input() | Tạo ô nhập liệu cho ngày và thời gian. |
| st.file\_uploader() | Tạo một khung tải tệp lên. |
| st.spinner() | Hiển thị một spinner khi ứng dụng đang xử lý. |
| MessageBox | Nó được sử dụng để hiển thị hộp thông báo trong các ứng dụng desktop. |
| st.progress() | Hiển thị thanh tiến trình. |
| st.text\_area() | Tạo một ô nhập liệu lớn cho văn bản đa dòng. |

*Bảng 2.1 Bảng tổng hợp widget của Streamlit* [3]

## *3.4 Xây dựng mô hình*

**Huấn luyện mô hình**

Chuẩn bị dữ liệu: Tiến hành split data để bắt đầu huấn luyện

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, stratify=y, random\_state=42)

*Huấn luyện mô hình*

Thực hiện huấn luyện mô hình với thuật toán LogisticRegression:

#LogisticRegression

logr = LogisticRegression()

logr.fit(X\_train, y\_train)

logr\_pred = logr.predict(X\_test)

Đoạn code trên sử dụng thư viện thuật toán LogisticRegression để huấn luyện mô hình

Thực hiện huấn luyện mô hình với thuật toán DecisionTreeClassifier:

#decisiontreeclassifier

tree = DecisionTreeClassifier()

tree.fit(X\_train, y\_train)

tree\_pred = tree.predict(X\_test)

Đoạn code trên sử dụng thư viện thuật toán DecisionTreeClassifier để huấn luyện mô hình

Thực hiện huấn luyện mô hình với thuật toán RandomForestClassifier:

#RandomForestClassifier

forest = RandomForestClassifier()

forest.fit(X\_train, y\_train)

forest\_pred = forest.predict(X\_test)

Đoạn code trên sử dụng thư viện thuật toán RandomForestClassifier để huấn luyện mô hình

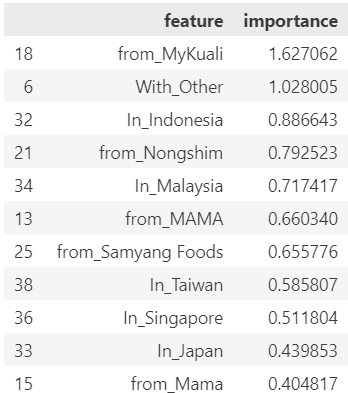
*Kiểm tra các hệ số thể hiện mức độ ảnh hưởng của mỗi đặc trưng đến dự đoán của mô hình*

LogisticRegression:

logr\_feature = pd.DataFrame({"feature": X.columns,

                             "importance": logr.coef\_[0]})

logr\_feature.sort\_values('importance', ascending=False)



Với thuật toán này, cột from\_MyKuali lại có tính quan trọng nhất trong bộ dữ liệu, ảnh hưởng tới kết quả xử lý của mô hình.

DecisionTreeClassifier:

tree\_feature = pd.DataFrame({"feature": X.columns,

                             "importance": tree.feature\_importances\_})

tree\_feature.sort\_values('importance', ascending=False)

A screenshot of a computer

Description automatically generated

Với thuật toán này, cột In\_Other lại có tính quan trọng nhất trong bộ dữ liệu, ảnh hưởng tới kết quả xử lý của mô hình.

RandomForestClassifier:

forest\_feature = pd.DataFrame({"feature": X.columns,

                             "importance": forest.feature\_importances\_})

forest\_feature.sort\_values('importance', ascending=False)

A screenshot of a table

Description automatically generated

Với thuật toán này, cột IsSpicy lại có tính quan trọng nhất trong bộ dữ liệu, ảnh hưởng tới kết quả xử lý của mô hình.

*Đánh giá chất lượng mô hình*

LogisticRegression

A screenshot of a computer

Description automatically generated

Với mô hình này, độ chính xác của thuật toán còn thấp, chỉ đạt mức 67% kết quả dự đoán đúng. Với bài toán có độ cân bằng thấp, các dữ liệu đầu vào chênh lệch nhau rất nhiều thì việc độ chính xác đạt 67% là ở mức tạm chấp nhận được và cần cải thiện thêm rất nhiều để tăng độ chính xác lên cao hơn.

DecisionTreeClassifier

A screenshot of a computer

Description automatically generated

Với mô hình này, độ chính xác của thuật toán còn thấp, chỉ đạt mức 64% và 59% kết quả dự đoán đúng. Như nhận xét về bộ dữ liệu ở trên thì việc độ chính xác đạt 64% và 59% là ở mức tạm chấp nhận được tuy nhiên cần cải thiện thêm để tăng độ chính xác lên cao hơn.

RandomForestClassifier

A screenshot of a computer

Description automatically generated

Với mô hình này, độ chính xác của thuật toán cũng ở mức thấp, chỉ đạt mức 66% và 59% kết quả dự đoán đúng. Mức độ chính xác này không khác biệt nhiều so với 2 mô hình phía trên.

3.5 Xây dựng giao diện ứng dụng

Trước hết ta sẽ cần lưu lại model với thư viện pickle, các model sẽ được lưu dưới dạng tệp tin “.pkl”.

Về giao diện, ta sẽ sử dụng thư viện Streamlit của python. Cụ thể, ta sẽ xây dựng giao diện mà người dùng có thể nhập vào các thông tin như IsSpicy, HasChicken,….

Sau đó dữ liệu nhập vào sẽ được chuyển thành một mảng rồi đưa vào mô hình thực hiện dự đoán. Cuối cùng sẽ hiển thị lên kết quả.

Giao diện chính sẽ có phần lựa chọn mô hình sẽ sử dụng và phần nhập dữ liệu

A screenshot of a computer

Description automatically generated

**CHƯƠNG 4: KẾT LUẬN**

**Kết quả đạt được:**

***Về mặt lý thuyết:***

Tìm hiểu được lý thuyết Machine Learning là gì. Nắm bắt được chi tiết lý thuyết một số kĩ thuật học máy.

Tìm hiểu được ba mô hình LogisticRegression, DecisionTreeClassifier, RandomForestClassifier, từ đó hiểu được mô hình có thể áp dụng cho các bài toán nhất định.

***Về mặt ứng dụng:***

Ứng dụng là bài nghiên cứu, nó cũng giúp bản thân biết được những mô hình những thư viện của Python cần dùng trong những bài toán thực tế.

***Những khó khăn mắc phải:***

Trong quá trình cài đặt môi trường gặp một số khó khăn về phiên bản, cấu hình.

Các thư viện cần cài đặt đã lỗi thời và không còn được hỗ trợ.

***Hạn chế của bài:***

Độ chính xác còn thấp, dữ liệu chưa đủ nên chưa thử nghiệm được cao.

***Định hướng phát triển:***

Thử nghiệm mô hình trên nhiều tập dữ liệu khác nhau, tăng số lượng dữ liệu cũng như số vòng lặp huấn luyện cải thiện độ chính xác. Thực hiện so sánh với các mô hình khác xem mô hình nào có tính ổn định và tối ưu.

Em xin chân thành cảm ơn !

|  |  |
| --- | --- |
| [1] | A. Amazon, "What’s the Difference Between Supervised and Unsupervised Learning?," [Online]. Available: https://aws.amazon.com/compare/the-difference-between-machine-learning-supervised-and-unsupervised/?nc1=h\_ls. |
| [2] | B. Vinbigdata, "Supervised Learning và Unsupervised Learning: Khác biệt là gì?," [Online]. Available: https://blog.vinbigdata.org/supervised-learning-va-unsupervised-learning-khac-biet-la-gi/. |
| [3] | "Streamlit Documentation," [Online]. Available: https://docs.streamlit.io/develop/api-reference/widgets. |
| [4] | Trituenhantao.io, "Cây Quyết Định (Decision Tree)," [Online]. Available: https://trituenhantao.io/kien-thuc/decision-tree/. |
| [5] | T. Nguyễn, "Random Forest algorithm," [Online]. Available: https://machinelearningcoban.com/tabml\_book/ch\_model/random\_forest.html. |