**BỘ GIÁO DỤC ĐÀO TẠO**

**TRƯỜNG ĐẠI HỌC ĐẠI NAM**

**------------------------------**

**A picture containing font, graphics, logo, graphic design

Description automatically generatedNGUYỄN ĐỨC HIỆP**

**MSV: 1571020100**

**NGHIÊN CỨU MÔ HÌNH AI DỰ ĐOÁN ĐIỂM THI ĐÁNH GIÁ NĂNG LỰC DỰA VÀO ĐIỂM HỌC BẠ CỦA HỌC SINH**

**BÁO CÁO ĐỒ ÁN TỐT NGHIỆP**

**NGÀNH CÔNG NGHỆ THÔNG TIN**

**HÀ NỘI – 2025**

**BỘ GIÁO DỤC ĐÀO TẠO**

**TRƯỜNG ĐẠI HỌC ĐẠI NAM**

**------------------------------**

**A picture containing font, graphics, logo, graphic design

Description automatically generatedNGUYỄN ĐỨC HIỆP**

**MSV: 1571020100, KHÓA: 15**

**NGHIÊN CỨU MÔ HÌNH AI DỰ ĐOÁN ĐIỂM THI ĐÁNH GIÁ NĂNG LỰC DỰA VÀO ĐIỂM HỌC BẠ CỦA HỌC SINH**

**BÁO CÁO ĐỒ ÁN TỐT NGHIỆP**

**NGÀNH : CÔNG NGHỆ THÔNG TIN**

**GV HƯỚNG DẪN: ThS. TRẦN THU TRANG**

**HÀ NỘI - 2025**

**LỜI CAM ĐOAN**

Tôi xin cam đoan đề tài “Nghiên cứu mô hình ai dự đoán điểm thi đánh giá năng lực dựa vào điểm học bạ của học sinh” là công trình nghiên cứu của cá nhân tôi. Các số liệu, kết quả nêu trong báo cáo là trung thực và chưa từng được sử dụng trong bất kỳ công trình nào khác.

Tôi hoàn toàn chịu trách nhiệm trước nhà trường về tính trung thực và bản quyền của nội dung đề tài này.

Hà Nội, ngày 22 tháng 05 năm 2025

Sinh viên thực hiện

**(**Ký rõ họ và tên)

**LỜI CẢM ƠN**

Đầu tiên, em xin gửi lời cảm ơn chân thành đến Trường Đại học Đại Nam và Khoa Công nghệ thông tin đã tạo điều kiện thuận lợi, cung cấp môi trường học tập và trang bị những kiến thức nền tảng quý báu cho em trong suốt quá trình học tập tại trường, cũng như tạo cơ hội để em thực hiện đồ án tốt nghiệp này.

Đặc biệt, em xin bày tỏ lòng biết ơn sâu sắc nhất đến giảng viên hướng dẫn ThS. Trần Thu Trang. Cô đã tận tình hướng dẫn, truyền đạt những kiến thức chuyên môn sâu sắc, đưa ra những góp ý xác đáng và đồng hành cùng em trong suốt quá trình nghiên cứu và hoàn thiện báo cáo đồ án tốt nghiệp. Sự hướng dẫn của cô là yếu tố then chốt giúp em hoàn thành đồ án này.

Trong quá trình thực hiện đồ án, em đã có cơ hội đào sâu kiến thức, rèn luyện kỹ năng nghiên cứu độc lập và tích lũy kinh nghiệm thực tế về lĩnh vực AI và xử lý dữ liệu, tạo hành trang quý báu cho con đường sự nghiệp sau này.

Với những kiến thức đã được trang bị và sự hướng dẫn của giảng viên hướng dẫn, em đã hoàn thành báo cáo này. Đây là một bước quan trọng trong hành trình học tập của em, và em hiểu rằng báo cáo có thể vẫn còn những điểm chưa hoàn thiện. Em rất mong nhận được những nhận xét, góp ý chân thành từ giảng viên hướng dẫn và Hội đồng, coi đây là cơ hội quý báu để em học hỏi và nâng cao kiến thức, kỹ năng của mình.

Em xin chân thành cảm ơn!

**LỜI NÓI ĐẦU**

Trong bối cảnh giáo dục ngày càng chú trọng đến việc đánh giá năng lực toàn diện và định hướng nghề nghiệp sớm cho học sinh, các kỳ thi đánh giá năng lực đóng vai trò quan trọng, giúp học sinh và phụ huynh có cái nhìn rõ hơn về khả năng và sự phù hợp với các ngành học, trường đại học khác nhau. Tuy nhiên, kết quả của những kỳ thi này thường mang tính quyết định, tạo ra áp lực không nhỏ cho học sinh trong quá trình chuẩn bị.

Với sự phát triển bùng nổ của Trí tuệ nhân tạo (AI) và khả năng xử lý, phân tích dữ liệu khổng lồ, việc ứng dụng AI vào lĩnh vực giáo dục, đặc biệt là trong việc dự đoán kết quả học tập, mở ra nhiều tiềm năng to lớn. Việc sử dụng dữ liệu điểm học bạ - nguồn dữ liệu sẵn có và phản ánh khá đầy đủ quá trình học tập của học sinh. Việc dự báo khả năng đạt điểm cao trong các kỳ thi đánh giá năng lực là một hướng tiếp cận hiệu quả, có thể mang lại giá trị thiết thực cho học sinh, phụ huynh và cả hệ thống giáo dục.

Xuất phát từ thực tế đó, đồ án tốt nghiệp này của em tập trung nghiên cứu đề tài: "Nghiên cứu mô hình ai dự đoán điểm thi đánh giá năng lực dựa vào điểm học bạ của học sinh". Mục tiêu chính của đồ án là nghiên cứu, xây dựng và thử nghiệm một mô hình AI có khả năng dự đoán tương đối chính xác điểm thi đánh giá năng lực.

Để đạt được mục tiêu này, đồ án tiến hành thu thập, xử lý dữ liệu điểm học bạ và áp dụng, so sánh hiệu quả của các thuật toán học máy khác nhau, bao gồm SVM (Support Vector Machine), Linear Regression, RandomForest và Mạng nơ-ron nhân tạo (ANN - Artificial Neural Network) để xây dựng và đánh giá các mô hình dự đoán.

**DANH MỤC TỪ VIẾT TẮT**

|  |  |
| --- | --- |
| AI | Artificial Intelligence( Trí tuệ nhân tạo) |
| SVM | Support Vector Machine (Máy vector hỗ trợ) |
| SVR | Support Vector Regression (Vector hỗ trợ hồi quy) |
| ML | Machine Learning (học máy) |

**DANH MỤC HÌNH ẢNH**

[Hình 3.1 Dữ liệu tiền xử lí 29](#_Toc199533437)

[Hình 3.2 Dữ liệu sau khi đã xử lí 30](#_Toc199533438)

[Hình 3.3 Xu hướng điểm trung bình của học sinh qua các lớp 10,11,12 31](#_Toc199533439)

[Hình 3.4 Mật độ điểm tổng kết của học sinh theo từng lớp 32](#_Toc199533440)

[Hình 3.5 Xu hướng điểm trung bình của học sinh theo học kì 32](#_Toc199533441)

[Hình 3.6 Phân bổ điểm của học sinh theo từng lớp 33](#_Toc199533442)

[Hình 4.1 Giao diện nhập tải và nhập file excel 39](#_Toc199533419)

[Hình 4.2 Mẫu excel nhập điểm 40](#_Toc199533420)

[Hình 4.3 Giao diện kết quả dự đoán sau khi nhập file excel 40](#_Toc199533421)

[Hình 4.4 Giao diện nhập điểm trực tiếp trên trang web 40](#_Toc199533422)

[Hình 4.5 Giao diện kết quả sau khi nhập điểm trục tiếp trên web 41](#_Toc199533423)

**DANH MỤC BẢNG BIỂU**

[Bảng 1 Bảng tham số điều chỉnh của mô hình SVM 10](#_Toc199531751)

[Bảng 2 Bảng so sánh Grid Search và Random Search 27](#_Toc199531752)

[Bảng 3 So sánh độ lỗi và độ chính xác của từng mô hình 36](#_Toc199531753)

**Mục lục**

[CHƯƠNG 1: GIỚI THIỆU 1](#_Toc199529811)

[1.1 Lý do chọn đề tài 1](#_Toc199529812)

[1.2 Mục tiêu nghiên cứu 2](#_Toc199529813)

[1.3 Lợi ích của đề tài 2](#_Toc199529814)

[1.4 Các thách thức trong quá trình thực hiện đề tài 3](#_Toc199529815)

[CHƯƠNG 2: CỞ SỞ LÝ THUYẾT 5](#_Toc199529816)

[2.1 Giới thiệu cơ bản về học máy 5](#_Toc199529817)

[2.2 Giới thiệu về mô hình dự đoán 6](#_Toc199529818)

[2.2.1 Mô hình SVM (Support Vector Machine) 6](#_Toc199529819)

[2.2.2 Mô hình Linear Regression 11](#_Toc199529820)

[2.2.3 Mô hình Random Forest 15](#_Toc199529821)

[2.2.4 Mô hình ANN (Artificial Neural Network) 20](#_Toc199529822)

[2.3 Các thuật toán tối ưu hóa 26](#_Toc199529823)

[2.3.1 Thuật toán Grid Search 26](#_Toc199529824)

[2.3.2 Thuật toán Random Search 27](#_Toc199529825)

[2.4 So sánh Grid Search và Random Search 28](#_Toc199529826)

[CHƯƠNG 3: PHÂN TÍCH DỮ LIỆU VÀ CHẠY THUẬT TOÁN 29](#_Toc199529827)

[3.1. Phân tích dữ liệu đầu vào 29](#_Toc199529828)

[3.1.1 Thông tin định danh và cơ bản: 29](#_Toc199529829)

[3.1.2 Kết quả học tập chi tiết theo từng năm học 29](#_Toc199529830)

[3.1.3 Điểm liên quan đến kỳ thi HSA 29](#_Toc199529831)

[3.2. Xử lý và chuẩn hóa dữ liệu 30](#_Toc199529832)

[3.2.1 Loại bỏ Cột không cần thiết: 30](#_Toc199529833)

[3.2.2 Xử lý Dữ liệu thiếu (Missing Values): 30](#_Toc199529834)

[3.2.3 Tính toán Feature Trung bình Môn học (Feature Engineering): 30](#_Toc199529835)

[3.2.4 Kết quả dữ liệu sau khi làm sạch 31](#_Toc199529836)

[3.2.5 Trực quan hóa dữ liệu thành biểu đồ 32](#_Toc199529837)

[3.2.6 Cách tiến hành xử lý dữ liệu với 4 mô hình 34](#_Toc199529838)

[3.3 Huấn luyện và đánh giá mô hình 36](#_Toc199529839)

[CHƯƠNG 4: XÂY DỰNG TRANG WEB 39](#_Toc199529840)

[4.1 Thiết kế giao diện web 39](#_Toc199529841)

[4.1.1 Tích hợp giao diện với backend 39](#_Toc199529842)

[4.1.2 Xây dựng trang web nhập điểm và file Excel 40](#_Toc199529843)

[4.2. Tích hợp mô hình và dự đoán 42](#_Toc199529844)

[4.2.1 Triển khai mô hình tối ưu trên backend 42](#_Toc199529845)

[4.3 Xử lý file Excel và nhập dữ liệu 43](#_Toc199529846)

[4.4 Dự báo điểm từ dữ liệu nhập 43](#_Toc199529847)

[KẾT LUẬN 45](#_Toc199529848)

[TÀI LIỆU THAM KHẢO 46](#_Toc199529849)

CHƯƠNG 1: GIỚI THIỆU

* 1. Lý do chọn đề tài

Trong hệ thống giáo dục hiện nay, việc đánh giá năng lực học sinh không chỉ dừng lại ở điểm số các môn học thông thường, mà còn mở rộng sang các kỳ thi đánh giá năng lực. Việc phân loại, công tác tuyển chọn và việc định hướng nghề nghiệp cho học sinh ngày càng cần đến vai trò quan trọng hơn của các kỳ thi này. Những kỳ thi này thường mang tính quyết định rất cao. Tuy nhiên, các kì thi này tiềm ẩn khá nhiều áp lực đối với học sinh và phụ huynh.

Dữ liệu điểm học bạ là một nguồn thông tin vốn sẵn có của học sinh THPT trong 3 năm học, nó phản ánh sự nỗ lực cũng như quá trình học tập của học sinh một cách tương đối toàn diện trong suốt nhiều năm liền. Nguồn dữ liệu này cần được phân tích một cách hiệu quả và khai thác một cách triệt để. Việc này là vô cùng cần thiết để đưa ra được những dự báo sớm về khả năng đạt được kết quả tốt trong các kỳ thi đánh giá năng lực. Điều này không chỉ giúp cho học sinh có được cái nhìn khách quan hơn về năng lực của chính mình và điều chỉnh được phương pháp học tập, ôn luyện, mà nó còn hỗ trợ phụ huynh, giáo viên đưa ra được những lời khuyên và định hướng phù hợp hơn.

Với sự phát triển mạnh mẽ của Trí tuệ nhân tạo (AI) cùng các kỹ thuật Học máy (Machine Learning), việc xây dựng những mô hình dự đoán dựa trên dữ liệu sẵn có đã mang lại hiệu quả cao và trở nên khả thi trong nhiều lĩnh vực, kể cả giáo dục (EdTech). Các đặc trưng giữa điểm học bạ cùng với điểm thi đánh giá năng lực đã được AI ứng dụng vào việc phân tích. Nhờ đó có thể tạo ra 1 công cụ để có thể trợ giúp cho học sinh có thể tham khảo điểm của mình.

Xuất phát từ nhận thức về tầm quan trọng của việc dự báo kết quả học tập, tiềm năng ứng dụng lớn lao của AI, cùng với sự quan tâm sâu sắc của em đến lĩnh vực xử lý dữ liệu và học máy trong giáo dục, em đã quyết định lựa chọn đề tài "Nghiên cứu mô hình AI dự đoán điểm thi đánh giá năng lực dựa vào điểm học bạ của học sinh" để làm đề tài đồ án tốt nghiệp của chính mình. Đề tài này mang tính ứng dụng thực tiễn rất cao, đồng thời là cơ hội tốt để em có thể đào sâu kiến thức đã học về AI và Machine Learning.

* 1. Mục tiêu nghiên cứu

Mục tiêu chung của đề tài là nghiên cứu, xây dựng và đánh giá hiệu quả của các mô hình Trí tuệ nhân tạo trong việc dự đoán điểm thi đánh giá năng lực dựa trên dữ liệu điểm học bạ của học sinh.

Các mục tiêu cụ thể bao gồm:

1. Thu thập và xử lý dữ liệu điểm học bạ của học sinh cùng với kết quả tương ứng từ một/một số kỳ thi đánh giá năng lực cụ thể.
2. Nghiên cứu các thuật toán học máy phù hợp cho bài toán dự đoán (ví dụ: SVR, Linear Regression, RandomForest, ANN).
3. Xây dựng các mô hình dự đoán điểm thi đánh giá năng lực sử dụng các thuật toán đã nghiên cứu trên tập dữ liệu thu thập được.
4. Đánh giá, so sánh hiệu quả (độ chính xác, sai số,...) của các mô hình đã xây dựng dựa trên các tiêu chí đánh giá phù hợp.
5. Đưa ra kết luận về mô hình tiềm năng nhất cho bài toán dự đoán này và các yếu tố ảnh hưởng.
6. Xây dựng một giao diện web cho phép người dùng nhập dữ liệu điểm học bạ và sử dụng mô hình dự đoán tốt nhất đã nghiên cứu để hiển thị kết quả dự đoán điểm thi đánh giá năng lực, nhằm minh họa tính ứng dụng của đề tài.
   1. Lợi ích của đề tài

Kết quả nghiên cứu của đề tài này được kỳ vọng sẽ mang lại những lợi ích thiết thực cho các bên liên quan:

* Đối với học sinh: Cung cấp một công cụ tham khảo giúp các em hiểu rõ hơn về khả năng của mình dựa trên kết quả học tập đã có, từ đó đưa ra kế hoạch ôn tập hiệu quả và lựa chọn ngành học, trường học phù hợp với năng lực thực tế, giảm bớt sự lo lắng trước kỳ thi.
* Đối với phụ huynh: Giúp phụ huynh có thêm cơ sở để đồng hành, tư vấn và hỗ trợ con em trong việc học tập và định hướng tương lai.
* Đối với nhà trường/giáo viên: Cung cấp thông tin tham khảo để hỗ trợ công tác tư vấn hướng nghiệp cho học sinh, có thể phát hiện sớm những học sinh có nguy cơ hoặc tiềm năng để có biện pháp hỗ trợ kịp thời.
* Đối với cộng đồng nghiên cứu: Góp phần vào kho tàng kiến thức ứng dụng AI trong lĩnh vực giáo dục, mở ra hướng nghiên cứu và phát triển các công cụ EdTech thông minh.
  1. Các thách thức trong quá trình thực hiện đề tài

Trong quá trình nghiên cứu và triển khai đề tài này, em nhận thấy có thể sẽ đối mặt với một số thách thức chính:

Thách thức về dữ liệu:

* Thu thập và xử lý: Việc thu thập được tập dữ liệu đủ lớn, đa dạng và đáng tin cậy (điểm học bạ và điểm thi đánh giá năng lực tương ứng) có thể gặp khó khăn do vấn đề quyền riêng tư và tính sẵn có của dữ liệu.
* Chất lượng dữ liệu: Dữ liệu điểm học bạ có thể không đồng nhất về thang điểm, cách đánh giá giữa các trường hoặc các năm học khác nhau, đòi hỏi quá trình tiền xử lý phức tạp.
* Chọn lọc đặc trưng: Xác định các môn học hoặc loại điểm học bạ nào có ảnh hưởng lớn nhất đến điểm thi đánh giá năng lực là một bài toán cần nghiên cứu.

Thách thức về mô hình:

* Lựa chọn mô hình và tối ưu: Lựa chọn được mô hình (trong số SVR, Linear Regression, RandomForest, ANN) phù hợp nhất với đặc điểm dữ liệu và tối ưu hóa các tham số của mô hình là một quá trình thử nghiệm tốn thời gian.
* Độ chính xác giới hạn: Điểm thi đánh giá năng lực phụ thuộc vào nhiều yếu tố (tâm lý phòng thi, sức khỏe,...), do đó, việc dự đoán chỉ dựa vào điểm học bạ sẽ có giới hạn về độ chính xác, khó có thể đạt được kết quả hoàn hảo.
* Tính giải thích: Một số mô hình (như ANN) có thể mang lại độ chính xác cao nhưng khó giải thích được tại sao lại đưa ra dự đoán đó, làm giảm tính minh bạch và tin cậy đối với người dùng cuối.

Thách thức về ứng dụng:

* Tính nhạy cảm: Việc đưa ra dự đoán kết quả thi là vấn đề nhạy cảm, cần cân nhắc kỹ lưỡng cách truyền đạt thông tin để tránh gây áp lực tiêu cực hoặc làm học sinh chủ quan.
* Khả năng tổng quát: Mô hình được xây dựng trên một tập dữ liệu cụ thể có thể không hoạt động tốt trên dữ liệu từ các nguồn khác (trường khác, năm khác) nếu không được đào tạo hoặc hiệu chỉnh phù hợp.

CHƯƠNG 2: CỞ SỞ LÝ THUYẾT

## **2.1 Giới thiệu cơ bản về học máy**

**2.1.1 Giới thiệu về học máy**

Trong kỷ nguyên của nền công nghiệp 4.0, Trí tuệ nhân tạo (AI – Artificial Intelligence) nổi lên như một trong những công nghệ cốt lõi định hình lại mọi lĩnh vực trong đời sống và sản xuất. Nếu cuộc cách mạng công nghiệp lần thứ nhất gắn liền với động cơ hơi nước, lần thứ hai là năng lượng điện, lần thứ ba là công nghệ thông tin, thì cuộc cách mạng thứ tư – đang diễn ra – chính là thời đại của dữ liệu lớn, AI và tự động hóa thông minh.

Trong đó, Machine Learning (ML – Học máy) là một nhánh quan trọng của AI, cho phép máy tính học từ dữ liệu thay vì phải lập trình cứng từng hành vi. Nói cách khác, ML giúp máy "tự học" từ kinh nghiệm, cải thiện hiệu suất dần theo thời gian mà không cần con người viết sẵn tất cả luật lệ.

Chúng ta đang chứng kiến hàng loạt ứng dụng học máy trong cuộc sống thường nhật: xe tự hành của Google và Tesla, hệ thống nhận diện khuôn mặt trên Facebook, trợ lý ảo Siri của Apple, gợi ý mua hàng của Amazon, gợi ý phim của Netflix, hay AlphaGo – chương trình chơi cờ vây đánh bại kỳ thủ chuyên nghiệp. Mỗi ứng dụng trên đều sử dụng các mô hình học máy khác nhau như SVM (Support Vector Machine), Random Forest, Linear Regression, hay Mạng nơ-ron nhân tạo (ANN) – những thuật toán cốt lõi giúp máy học từ dữ liệu và đưa ra quyết định.

Song song với sự phát triển của học máy, khi sức mạnh tính toán và dữ liệu tăng vượt bậc, một lĩnh vực mới ra đời gọi là Deep Learning (Học sâu) – mô hình học máy sử dụng nhiều lớp mạng nơ-ron phức tạp. Deep Learning đã thúc đẩy AI tiến một bước dài: giúp máy tính nhận diện hàng ngàn vật thể trong ảnh, tạo phụ đề tự động cho video, giả lập giọng nói con người, và thậm chí... sáng tác nhạc hay thơ. Các công nghệ như ChatGPT, DALL·E, hay công cụ dịch tự động hiện nay chính là kết quả của sự bùng nổ từ học sâu.

## **2.2 Giới thiệu về mô hình dự đoán**

### **2.2.1 Mô hình SVM (Support Vector Machine)**

**2.2.1.1 Khái niệm và nguyên lý hoạt động**

Khái niệm:

Support Vector Machine (SVM), hay Máy Vector Hỗ trợ, là một thuật toán học có giám sát mạnh mẽ, được sử dụng chủ yếu cho các bài toán phân loại (classification) nhưng cũng có thể áp dụng cho các bài toán hồi quy (regression). SVM trở nên phổ biến từ những năm 1990 và đặc biệt hiệu quả đối với các tập dữ liệu nhỏ hoặc phức tạp, thường yêu cầu ít tinh chỉnh. Mục tiêu cốt lõi của SVM là tìm ra một "siêu phẳng" (hyperplane) tối ưu trong không gian đặc trưng nhiều chiều để phân tách các lớp dữ liệu một cách tốt nhất. Mô hình này được ứng dụng trong nhiều lĩnh vực như đánh giá rủi ro tín dụng, nhận dạng mẫu, phân tích hình ảnh, xử lý văn bản, phát hiện thư rác, nhận dạng đối tượng, phân tích tình cảm và tin sinh học.

Nguyên lý hoạt động:

Nguyên lý hoạt động của SVM xoay quanh việc xác định một ranh giới quyết định sao cho khoảng cách đến các điểm dữ liệu gần nhất của mỗi lớp là lớn nhất. Các thành phần chính bao gồm:

* Siêu phẳng (Hyperplane): Trong không gian đặc trưng, siêu phẳng đóng vai trò là ranh giới quyết định phân chia các điểm dữ liệu thuộc các lớp khác nhau. Nếu không gian là 2 chiều, siêu phẳng là một đường thẳng; nếu là 3 chiều, nó là một mặt phẳng; và trong không gian nhiều chiều hơn, nó là một siêu phẳng.SVM tìm kiếm siêu phẳng "tốt nhất" để phân chia dữ liệu một cách tối ưu.
* Tối đa hóa lề (Margin Maximization): Đây là nguyên tắc trung tâm của SVM. Thuật toán cố gắng tìm ra siêu phẳng có khoảng cách (gọi là "lề" - margin) lớn nhất tới các điểm dữ liệu gần nhất của tất cả các lớp. Việc tối đa hóa lề này giúp tạo ra một "khoảng trống" rộng nhất có thể giữa các lớp, dẫn đến khả năng tổng quát hóa tốt hơn trên dữ liệu chưa biết và giảm nguy cơ overfitting, đồng thời tăng độ tin cậy của dự đoán. Siêu phẳng tối ưu nằm chính giữa lề này. Điều này khác biệt với các phương pháp như Hồi quy Logistic, vốn xem xét tất cả các điểm dữ liệu khi xác định ranh giới.
* Vector hỗ trợ (Support Vectors): Đây là những điểm dữ liệu nằm gần siêu phẳng nhất và đóng vai trò quan trọng trong việc xác định lề và hướng của siêu phẳng. Nếu các vector hỗ trợ này bị loại bỏ, vị trí của siêu phẳng phân chia sẽ thay đổi.Hàm quyết định của SVM chỉ phụ thuộc vào các vector hỗ trợ này, thay vì toàn bộ tập dữ liệu.

Việc SVM chỉ dựa vào các vector hỗ trợ mang lại một lợi thế đáng kể. Các mô hình như Hồi quy Logistic sử dụng tất cả các điểm dữ liệu để xác định ranh giới quyết định. Ngược lại, SVM tập trung vào các điểm dữ liệu quan trọng nhất - những điểm nằm gần ranh giới nhất. Vị trí của các điểm dữ liệu nằm xa ranh giới không ảnh hưởng đến vị trí của siêu phẳng tối ưu. Do đó, mô hình cuối cùng chỉ cần lưu trữ hoặc xem xét các vector hỗ trợ này. Tính "thưa" (sparsity) này làm cho SVM hiệu quả về mặt tính toán và bộ nhớ trong quá trình dự đoán, đồng thời có thể mạnh mẽ trong không gian nhiều chiều, nơi nhiều đặc trưng có thể không liên quan đến việc phân tách gần ranh giới. Điều này đặc biệt thuận lợi khi số lượng vector hỗ trợ nhỏ so với tổng kích thước tập dữ liệu.

**2.2.1.2 Support Vector Regression (SVR) – hồi quy vector hỗ trợ**

SVR là một biến thể của SVM, một thuật toán học có giám sát phổ biến, ban đầu được thiết kế để phân loại dữ liệu. Tuy nhiên, SVR được điều chỉnh để giải bài toán hồi quy, tức là thuật toán sẽ dự đoán các giá trị của dữ liệu liên tục.  
Mục tiêu của SVM (và SVR) là tìm ra một siêu phẳng (hyperplane) tối ưu, giúp phân chia hoặc mô tả dữ liệu đầu ra một cách tốt nhất trong không gian nhiều chiều.

**2.1.1.3 Kernel – Hàm hạt nhân**

SVR sử dụng kernel để xử lý dữ liệu không tuyến tính. Kernel là hàm đo độ tương đồng giữa các điểm dữ liệu. Có hai loại chính:

* Kernel tuyến tính: sử dụng tích vô hướng đơn giản giữa hai vector.
* Kernel phi tuyến (như RBF, polynomial): giúp ánh xạ dữ liệu sang không gian đặc trưng cao hơn để phát hiện các mẫu phức tạp hơn.

Việc chọn loại kernel phù hợp là rất quan trọng, phụ thuộc vào cấu trúc và độ phức tạp của dữ liệu.

**2.1.1.4 Siêu tham số quan trọng**

SVR có một số siêu tham số (hyperparameters) cần được tinh chỉnh:

* + C (Regularization parameter): Điều chỉnh độ cân bằng giữa độ chính xác trên tập huấn luyện và khả năng tổng quát.
    - C lớn: ít sai số hơn nhưng dễ overfit.
    - C nhỏ: linh hoạt hơn, cho phép một số lỗi nhỏ.
  + ε (epsilon): Xác định “ống sai số” xung quanh siêu phẳng mà trong đó mô hình không bị phạt vì sai số.
    - Epsilon càng lớn: mô hình càng dễ bỏ qua sai số nhỏ.

**2.1.1.5 Đánh giá mô hình**

Để đánh giá hiệu quả của SVR, dữ liệu thường được chia thành:

* Tập huấn luyện (training set): để huấn luyện mô hình.
* Tập kiểm tra (test set): để đánh giá độ chính xác dự đoán.

Một số thước đo phổ biến để đánh giá:

* MSE (Mean Squared Error): Sai số bình phương trung bình.
* MAE (Mean Absolute Error): Sai số tuyệt đối trung bình.

Giá trị càng thấp càng tốt, cho thấy mô hình dự đoán gần với giá trị thực tế.

**2.2.1.6 Các tham số điều chỉnh**

|  |  |  |  |
| --- | --- | --- | --- |
| Tham số | Mô tả | Lựa chọn/Giá trị tiêu biểu | Tác động (Giá trị cao/thấp) |
| C | Kiểm soát sự đánh đổi giữa tối đa hóa lề và tối thiểu hóa lỗi huấn luyện | Giá trị dương (vd: 0.1, 1, 10, 100) | Cao: Lề hẹp hơn, phức tạp hơn, thiên vị thấp, phương sai cao, nguy cơ overfitting cao. Thấp: Lề rộng hơn, đơn giản hơn, thiên vị cao, phương sai thấp, thúc đẩy tổng quát hóa. |
| kernel | Xác định hàm biến đổi dữ liệu để xử lý tính phi tuyến | 'linear', 'poly', 'rbf', 'sigmoid' | 'linear': Ranh giới tuyến tính. 'rbf', 'poly', 'sigmoid': Ranh giới phi tuyến, độ phức tạp phụ thuộc vào các tham số khác. |
| gamma | Xác định phạm vi ảnh hưởng của một điểm dữ liệu (cho kernel phi tuyến) | Giá trị dương (vd: 'scale', 'auto', 0.01, 0.1, 1) | Cao: Ảnh hưởng cục bộ, ranh giới phức tạp/khúc khuỷu, nguy cơ overfitting cao. Thấp: Ảnh hưởng lan tỏa, ranh giới mượt mà hơn, nguy cơ underfitting cao. |
| degree | Bậc của đa thức (chỉ cho kernel 'poly') | Số nguyên dương (vd: 2, 3, 4) | Cao: Ranh giới phức tạp hơn, nguy cơ overfitting cao. Thấp: Ranh giới đơn giản hơn. |

Bảng 1 Bảng tham số điều chỉnh của mô hình SVM

**2.2.1.3 Ưu và nhược điểm**

Ưu điểm:

* Hiệu quả trong không gian nhiều chiều: SVM hoạt động tốt ngay cả khi số lượng chiều (đặc trưng) lớn hơn số lượng mẫu.
* Hiệu quả về bộ nhớ: Do chỉ sử dụng các vector hỗ trợ trong hàm quyết định, SVM tiết kiệm bộ nhớ.
* Xử lý dữ liệu phi tuyến: Rất hiệu quả đối với dữ liệu phức tạp, phi tuyến tính nhờ vào việc sử dụng các kernel.
* Khả năng chống overfitting: Việc tối đa hóa lề giúp mô hình tổng quát hóa tốt, giảm nguy cơ overfitting, đặc biệt khi được tinh chỉnh đúng cách.
* Hiệu quả với tập dữ liệu nhỏ: Hoạt động tốt trên các tập dữ liệu có kích thước nhỏ.
* Ít nhạy cảm với ngoại lệ (outliers): So với hồi quy logistic, SVM ít bị ảnh hưởng bởi các điểm dữ liệu ngoại lệ do tập trung chủ yếu vào các vector hỗ trợ.

Nhược điểm:

* Chi phí tính toán cao với dữ liệu lớn: Quá trình huấn luyện SVM có thể tốn kém về mặt tính toán và chậm đối với các tập dữ liệu rất lớn. Độ phức tạp thời gian huấn luyện có thể cao.
* Khó tinh chỉnh siêu tham số: Việc chọn kernel và các tham số (C, gamma) phù hợp có thể khó khăn và tốn thời gian, đòi hỏi nhiều thử nghiệm.
* Hiệu suất kém với dữ liệu nhiễu: Nếu dữ liệu có nhiều nhiễu hoặc các lớp chồng chéo lên nhau, SVM có thể gặp khó khăn trong việc tìm ra sự phân tách tốt.
* Thiếu đầu ra xác suất trực tiếp: SVM không cung cấp ước lượng xác suất một cách tự nhiên như Hồi quy Logistic. (Mặc dù có thể hiệu chỉnh để có đầu ra xác suất).
* Khó diễn giải: Mô hình SVM, đặc biệt là khi sử dụng kernel phi tuyến, thường được coi là "hộp đen", khó diễn giải và hiểu lý do đằng sau một dự đoán cụ thể.
* Vấn đề về khả năng mở rộng: Gặp khó khăn về thời gian huấn luyện và có thể cả bộ nhớ (lưu trữ ma trận kernel hoặc vector hỗ trợ) khi làm việc với các tập dữ liệu cực lớn.Việc song song hóa thuật toán khớp SVM cốt lõi có thể là một thách thức.

### **2.2.2 Mô hình Linear Regression**

**2.2.2.1 Khái niệm và nguyên lý hoạt động**

Khái niệm:

Linear Regression là một thuật toán học có giám sát (supervised learning) trong [Machine Learning](https://trituenhantao.io/tu-dien-thuat-ngu/machine-learning/), nó là một phương pháp thống kê dùng để ước lượng mối quan hệ giữa các biến độc lập (input features) và biến phụ thuộc (output target). Linear Regression giả định rằng sự tương quan giữa các biến là tuyến tính, từ đó tìm ra hàm tuyến tính tốt nhất để biểu diễn mối quan hệ này. Thuật toán này dự báo giá trị của biến output từ các giá trị của các biến đầu vào.

Nguyên lý hoạt động:

Nguyên lý hoạt động của Hồi quy tuyến tính (Linear Regression) dựa trên ý tưởng tìm kiếm mối quan hệ tuyến tính giữa một biến phụ thuộc (biến mà chúng ta muốn dự đoán) và một hoặc nhiều biến độc lập (các biến dùng để dự đoán).

Dưới đây là các nguyên lý cốt lõi:

Mô hình hóa mối quan hệ tuyến tính: Linear Regression giả định rằng có một mối quan hệ tuyến tính giữa biến phụ thuộc (ký hiệu là y) và các biến độc lập (ký hiệu là x1​,x2​,...,xn​). Mối quan hệ này được biểu diễn bằng một phương trình tuyến tính.

Đối với hồi quy tuyến tính đơn giản (Simple Linear Regression - một biến độc lập), phương trình có dạng:

* + y: biến phụ thuộc
  + x: biến độc lập
  + a: hệ số chặn (intercept) - giá trị dự đoán của y khi x bằng 0
  + b: hệ số góc (slope) - cho biết sự thay đổi trung bình của y khi x thay đổi một đơn vị
  + ϵ: sai số ngẫu nhiên (error term) - phần không giải thích được bởi mô hình.

Đối với Hồi quy tuyến tính bội (Multiple Linear Regression - nhiều biến độc lập), phương trình tổng quát có dạng:

* + y: biến phụ thuộc
  + x1​,x2​,...,xn​: các biến độc lập
  + a: hệ số chặn
  + b1​,b2​,...,bn​: các hệ số hồi quy tương ứng với từng biến độc lập, cho biết sự thay đổi trung bình của y khi biến độc lập tương ứng thay đổi một đơn vị (giữ các biến độc lập khác không đổi).
  + ϵ: sai số ngẫu nhiên.

Tìm đường (hoặc mặt phẳng) phù hợp nhất: Mục tiêu chính của Linear Regression là tìm ra các giá trị tốt nhất cho các hệ số (a,b,b1​,b2​,...) sao cho đường thẳng (trong trường hợp đơn giản) hoặc siêu phẳng (trong trường hợp đa biến) được tạo bởi phương trình tuyến tính "phù hợp nhất" với dữ liệu quan sát. Sự phù hợp này được đánh giá dựa trên việc giảm thiểu sự khác biệt giữa giá trị thực tế của biến phụ thuộc (y) và giá trị dự đoán bởi mô hình (

Nguyên lý Bình phương tối thiểu thông thường (Ordinary Least Squares - OLS): Đây là phương pháp phổ biến nhất để ước lượng các hệ số trong Linear Regression. Nguyên lý OLS hoạt động bằng cách giảm thiểu tổng bình phương của các sai số (residuals). Sai số là sự khác biệt giữa giá trị y quan sát được và giá trị y^​ được dự đoán bởi mô hình cho cùng một tập hợp các giá trị của biến độc lập. Công thức của tổng bình phương sai số (Sum of Squared Errors - SSE) là: , trong đó m là số lượng điểm dữ liệu. OLS tìm các hệ số a,b,... sao cho giá trị của SSE là nhỏ nhất.

Tóm lại, nguyên lý hoạt động của Linear Regression là xây dựng một mô hình tuyến tính để biểu diễn mối quan hệ giữa các biến và sử dụng phương pháp Bình phương tối thiểu (OLS) để tìm ra các hệ số tối ưu cho mô hình đó, nhằm giảm thiểu sự khác biệt giữa giá trị thực tế và giá trị dự đoán.

**2.2.2.3 Ưu và nhược điểm**

Ưu điểm:

* Đơn giản: Dễ hiểu, dễ triển khai và diễn giải.
* Khả năng diễn giải cao: Các hệ số hồi quy cung cấp cái nhìn sâu sắc rõ ràng về độ lớn và hướng của mối quan hệ giữa từng biến độc lập và biến phụ thuộc.
* Hiệu quả tính toán: Huấn luyện mô hình nhanh chóng, đặc biệt so với các mô hình phức tạp hơn.Có thể mở rộng tốt cho các tập dữ liệu lớn nếu các giả định được đáp ứng.
* Nền tảng: Là cơ sở cho nhiều kỹ thuật thống kê và học máy tiên tiến khác.

Nhược điểm:

* Nhạy cảm với các giả định: Hiệu suất và tính hợp lệ của mô hình phụ thuộc mạnh mẽ vào các giả định (tuyến tính, độc lập, chuẩn, phương sai không đổi), vốn thường bị vi phạm trong thực tế.
* Hạn chế về tính tuyến tính: Chỉ có thể mô hình hóa hiệu quả các mối quan hệ tuyến tính. Nếu mối quan hệ cơ bản là phi tuyến, mô hình sẽ hoạt động kém. (Hồi quy đa thức có thể giải quyết phần nào nhưng làm tăng độ phức tạp).
* Nhạy cảm với ngoại lệ (Outliers): Phương pháp OLS rất nhạy cảm với các giá trị ngoại lệ do nó tối thiểu hóa tổng bình phương sai số, khiến các lỗi lớn có ảnh hưởng không cân xứng.
* Dễ bị Overfitting/Underfitting: Có thể bị overfitting nếu có quá nhiều biến dự báo so với số lượng mẫu, hoặc bị underfitting nếu mô hình quá đơn giản so với độ phức tạp của dữ liệu. Thường cần các kỹ thuật chính quy hóa (Regularization) như Ridge hoặc Lasso.
* Vấn đề đa cộng tuyến (Multicollinearity): Trong hồi quy bội, nếu các biến độc lập có tương quan cao với nhau, các ước lượng hệ số sẽ trở nên không ổn định và khó diễn giải chính xác.

### **2.2.3 Mô hình Random Forest**

**2.2.3.1 Khái niệm và cách hoạt động**

Khái niệm:

Random Forest (RF), hay Rừng ngẫu nhiên, là một thuật toán học máy có giám sát thuộc nhóm phương pháp học tập kết hợp (ensemble learning). Nó xây dựng một tập hợp gồm nhiều cây quyết định (decision trees) trong quá trình huấn luyện và tổng hợp kết quả dự đoán từ các cây riêng lẻ để đưa ra dự đoán cuối cùng. Random Forest có thể được sử dụng cho cả bài toán phân loại và hồi quy. Thuật toán này dựa trên nguyên tắc "trí tuệ đám đông" (wisdom of crowds), nơi quyết định tập thể thường tốt hơn quyết định cá nhân. Random Forest được đề xuất bởi Leo Breiman vào năm 2001.

Nguyên lý hoạt động

Bước 1: Tạo các tập dữ liệu huấn luyện con (Bootstrap Sampling):

* Random Forest không sử dụng toàn bộ tập dữ liệu huấn luyện ban đầu để xây dựng mỗi cây quyết định. Thay vào đó, nó tạo ra nhiều tập dữ liệu con bằng kỹ thuật bootstrap sampling.
* Bootstrap sampling là quá trình lấy mẫu ngẫu nhiên có thay thế từ tập dữ liệu gốc. Điều này có nghĩa là một mẫu dữ liệu cụ thể có thể xuất hiện nhiều lần trong cùng một tập dữ liệu con, hoặc không xuất hiện lần nào.
* Mỗi cây trong rừng sẽ được huấn luyện trên một tập dữ liệu con được tạo ra bằng phương pháp bootstrap này. Do được lấy mẫu có thay thế, các tập dữ liệu con này sẽ khác nhau, tạo ra sự đa dạng về dữ liệu cho mỗi cây.

Bước 2: Xây dựng các cây quyết định với lựa chọn đặc trưng ngẫu nhiên:

* Đối với mỗi tập dữ liệu huấn luyện con đã được tạo ở bước 1, Random Forest sẽ xây dựng một cây quyết định.
* Điểm khác biệt quan trọng so với cây quyết định thông thường là tại mỗi nút khi cây đang phân tách (splitting), Random Forest không xem xét tất cả các đặc trưng có sẵn để tìm ra đặc trưng tốt nhất để phân tách. Thay vào đó, nó chỉ xem xét một tập con ngẫu nhiên các đặc trưng được chọn từ tập hợp đặc trưng đầy đủ của dữ liệu gốc.
* Kích thước của tập con đặc trưng này thường được xác định trước (ví dụ: căn bậc hai của tổng số đặc trưng đối với bài toán phân loại, hoặc một phần ba tổng số đặc trưng đối với bài toán hồi quy).
* Trong số các đặc trưng được chọn ngẫu nhiên ở nút hiện tại, thuật toán sẽ tìm ra đặc trưng và điểm phân tách (split point) tốt nhất dựa trên tiêu chí phân loại (ví dụ: Gini impurity hoặc entropy) hoặc hồi quy (ví dụ: sai số bình phương).
* Quá trình xây dựng cây này tiếp tục cho đến khi đạt được điều kiện dừng (ví dụ: độ sâu tối đa của cây, số lượng mẫu tối thiểu trong một lá).

Bước 3: Tổng hợp kết quả (Aggregation):

* Sau khi xây dựng xong một số lượng lớn các cây quyết định (đã được huấn luyện trên các tập dữ liệu con bootstrap và sử dụng lựa chọn đặc trưng ngẫu nhiên tại mỗi nút), Random Forest sẽ tổng hợp kết quả từ tất cả các cây để đưa ra dự đoán cuối cùng.
* Đối với bài toán phân loại, dự đoán cuối cùng thường là lớp được bầu chọn nhiều nhất bởi các cây riêng lẻ (majority voting).
* Đối với bài toán hồi quy, dự đoán cuối cùng thường là giá trị trung bình của dự đoán từ tất cả các cây.

**2.2.3.2 Cách xây dựng rừng quyết định**

1. Xác định số lượng cây (N) cần xây dựng: Đây là một siêu tham số (hyperparameter) quan trọng cần được xác định trước. Số lượng cây càng lớn thì mô hình càng mạnh mẽ và ổn định hơn, nhưng cũng tốn kém chi phí tính toán hơn.
2. Lặp lại N lần để xây dựng từng cây quyết định độc lập: Đối với mỗi lần lặp (từ 1 đến N), thực hiện các bước sau để xây dựng một cây quyết định:

a. Tạo tập dữ liệu huấn luyện con bằng Bootstrap Sampling: Từ tập dữ liệu huấn luyện gốc có M mẫu, tạo ra một tập dữ liệu huấn luyện con mới có cùng kích thước M bằng cách lấy mẫu ngẫu nhiên có thay thế. Điều này có nghĩa là một mẫu dữ liệu cụ thể từ tập gốc có thể được chọn nhiều lần hoặc không được chọn lần nào trong tập con này.

b. Xây dựng cây quyết định trên tập dữ liệu con với Lựa chọn Đặc trưng Ngẫu nhiên:

\* Bắt đầu xây dựng một cây quyết định từ nút gốc (root node) sử dụng tập dữ liệu con đã tạo ở bước 2a.

\* Tại mỗi nút trong quá trình phát triển cây, thay vì xem xét tất cả D đặc trưng có sẵn trong tập dữ liệu gốc, chỉ chọn ngẫu nhiên một tập con gồm d đặc trưng (với d<D). Kích thước d này là một siêu tham số khác, thường được đặt là ​ cho bài toán phân loại hoặc D/3 cho bài toán hồi quy, nhưng có thể điều chỉnh.

\* Trong số d đặc trưng đã chọn ngẫu nhiên này, tìm đặc trưng và điểm phân tách tốt nhất để chia dữ liệu tại nút đó dựa trên tiêu chí tối ưu hóa (ví dụ: giảm Gini impurity hoặc entropy cho phân loại, giảm sai số bình phương cho hồi quy).

\* Lặp lại quá trình phân tách này tại các nút con cho đến khi đạt được điều kiện dừng được xác định trước (ví dụ: độ sâu tối đa của cây, số lượng mẫu tối thiểu trong một lá, không thể phân tách thêm). Mỗi cây thường được xây dựng cho đến khi đạt độ sâu tối đa hoặc gần như vậy (pruning thường không được áp dụng mạnh mẽ cho từng cây riêng lẻ trong Random Forest).

1. Tổng hợp kết quả từ tất cả các cây: Sau khi N cây quyết định đã được xây dựng hoàn chỉnh, Random Forest sẵn sàng đưa ra dự đoán.
   * Đối với bài toán phân loại: Khi có một mẫu dữ liệu mới cần dự đoán, đưa mẫu này qua tất cả N cây quyết định. Mỗi cây sẽ đưa ra một dự đoán về lớp của mẫu đó. Kết quả cuối cùng là lớp được dự đoán bởi đa số các cây (major voting).
   * Đối với bài toán hồi quy: Khi có một mẫu dữ liệu mới cần dự đoán, đưa mẫu này qua tất cả N cây quyết định. Mỗi cây sẽ đưa ra một giá trị dự đoán. Kết quả cuối cùng là giá trị trung bình của các dự đoán từ tất cả các cây.

Tóm lược các yếu tố chính trong việc xây dựng Random Forest:

* Ensemble Learning: Kết hợp sức mạnh của nhiều mô hình yếu (cây quyết định).
* Bagging (Bootstrap Aggregating): Tạo sự đa dạng về dữ liệu huấn luyện cho mỗi cây bằng cách lấy mẫu ngẫu nhiên có thay thế.
* Random Subspace Method (Lựa chọn đặc trưng ngẫu nhiên): Tạo sự đa dạng trong không gian đặc trưng được mỗi cây xem xét tại mỗi bước phân tách.
* Cây quyết định không được tỉa cành (hoặc tỉa ít): Các cây riêng lẻ thường được phát triển sâu để có phương sai cao, sau đó phương sai này được giảm đi thông qua quá trình tổng hợp của rừng.
* Tổng hợp kết quả: Sử dụng bỏ phiếu đa số (phân loại) hoặc trung bình (hồi quy) để đưa ra dự đoán cuối cùng.

Quá trình này đảm bảo rằng các cây trong rừng không giống hệt nhau và mỗi cây có xu hướng chuyên biệt vào các khía cạnh khác nhau của dữ liệu và đặc trưng. Khi kết hợp lại, chúng tạo nên một mô hình mạnh mẽ, ít bị overfitting và thường cho kết quả dự đoán tốt trên dữ liệu mới.

**2.2.3.3 Ưu và nhược điểm**

Ưu điểm:

* Độ chính xác cao: Thường đạt được độ chính xác dự đoán cao trên nhiều loại bài toán.
* Chống overfitting hiệu quả: Bản chất kết hợp và tính ngẫu nhiên kép giúp giảm đáng kể hiện tượng overfitting so với cây quyết định đơn lẻ.
* Xử lý tốt tính phi tuyến: Kế thừa khả năng xử lý các mối quan hệ phi tuyến từ cây quyết định.
* Xử lý dữ liệu lớn và nhiều chiều: Hoạt động hiệu quả với các tập dữ liệu có số lượng mẫu (N) lớn và số lượng đặc trưng (p) lớn.
* Đo lường tầm quan trọng của đặc trưng: Cung cấp các cơ chế tích hợp để ước tính mức độ quan trọng của từng đặc trưng đầu vào đối với kết quả dự đoán.
* Xử lý dữ liệu bị thiếu: Có khả năng xử lý dữ liệu bị thiếu ở một mức độ nhất định (nhờ feature bagging).
* Ít yêu cầu tiền xử lý: Thường yêu cầu ít chuẩn hóa hoặc co giãn dữ liệu (scaling/normalization) hơn so với các mô hình dựa trên khoảng cách như SVM hoặc các phương pháp nhạy cảm với thang đo đặc trưng.
* Đánh giá nội bộ: Lỗi OOB cung cấp một thước đo đánh giá hiệu suất tổng quát hóa tích hợp sẵn mà không cần tập kiểm định riêng.
* Linh hoạt: Áp dụng được cho cả bài toán phân loại và hồi quy.

Nhược điểm:

* Phức tạp và khó diễn giải: Mô hình là một "hộp đen"; khó diễn giải hơn nhiều so với một cây quyết định đơn lẻ. Việc hiểu tại sao một dự đoán cụ thể được đưa ra là rất khó khăn.
* Chi phí tính toán: Có thể chậm trong quá trình huấn luyện, đặc biệt với số lượng cây lớn, cây sâu hoặc tập dữ liệu lớn. Yêu cầu nhiều tài nguyên bộ nhớ hơn.
* Tốc độ dự đoán: Quá trình dự đoán có thể chậm hơn so với các mô hình đơn giản hơn nếu rừng lớn. Không lý tưởng cho các ứng dụng thời gian thực đòi hỏi dự đoán cực nhanh.
* Có thể không tối ưu cho dữ liệu thưa: So với các mô hình tuyến tính, Random Forest có thể không hoạt động tốt trên các tập dữ liệu rất thưa (sparse data).
* Vẫn cần tinh chỉnh: Mặc dù thường ít nhạy cảm hơn SVM, việc tinh chỉnh siêu tham số vẫn có thể cần thiết để đạt hiệu suất tối ưu.

### **2.2.4 Mô hình ANN (Artificial Neural Network)**

**2.2.4.1 Khái niệm và nguyên lý hoạt động**

Khái niệm:

Mạng nơ-ron nhân tạo (Artificial Neural Network - ANN) là một mô hình tính toán được lấy cảm hứng từ cấu trúc và chức năng của mạng lưới nơ-ron sinh học trong não người. ANN bao gồm các đơn vị xử lý được kết nối với nhau gọi là nơ-ron (neurons) hoặc nút (nodes), được tổ chức thành các lớp (layers). ANN là một lĩnh vực con của Trí tuệ Nhân tạo (AI) và là nền tảng cơ bản của Học sâu (Deep Learning). Chúng được sử dụng để giải quyết các bài toán phức tạp như nhận dạng mẫu, xử lý ngôn ngữ tự nhiên (NLP), thị giác máy tính và dự đoán.

Cấu trúc:

Một mạng nơ-ron nhân tạo điển hình có kiến trúc phân lớp:

* Lớp đầu vào (Input Layer): Nhận dữ liệu thô đầu vào. Mỗi nơ-ron trong lớp này tương ứng với một đặc trưng (feature) của dữ liệu đầu vào.
* Lớp ẩn (Hidden Layers): Nằm giữa lớp đầu vào và lớp đầu ra. Có thể có một hoặc nhiều lớp ẩn. Các lớp này thực hiện phần lớn các phép tính toán trung gian, trích xuất đặc trưng và biến đổi thông tin. Mạng có nhiều lớp ẩn được gọi là mạng nơ-ron sâu (deep neural networks).
* Lớp đầu ra (Output Layer): Lớp cuối cùng của mạng, tạo ra kết quả dự đoán cuối cùng của mô hình. Số lượng nơ-ron và hàm kích hoạt của lớp này phụ thuộc vào nhiệm vụ cụ thể (ví dụ: một nơ-ron với hàm sigmoid cho phân loại nhị phân, nhiều nơ-ron với hàm softmax cho phân loại đa lớp, một nơ-ron tuyến tính cho hồi quy).

Các thành phần cốt lõi:

* Nơ-ron (Neurons/Nodes/Units): Đơn vị tính toán cơ bản của mạng.
* Kết nối (Connections): Liên kết giữa các nơ-ron, truyền tín hiệu thông tin.
* Trọng số (Weights): Các tham số liên kết với các kết nối, biểu thị độ mạnh của tín hiệu. Các trọng số này được học trong quá trình huấn luyện.
* Thiên vị (Biases): Các tham số bổ sung liên kết với mỗi nơ-ron, cho phép dịch chuyển hàm kích hoạt. Cũng được học trong quá trình huấn luyện.
* Hàm kích hoạt (Activation Functions): Áp dụng cho đầu ra được tính toán của mỗi nơ-ron (tổng trọng số của đầu vào cộng với thiên vị). Chúng đưa tính phi tuyến vào mạng, điều này rất cần thiết để học các mẫu phức tạp. Các hàm phổ biến bao gồm Sigmoid, Tanh, ReLU (Rectified Linear Unit), Softmax.

Nguyên lý hoạt động (Luồng thông tin và Học tập):

* Lan truyền tiến (Forward Propagation): Dữ liệu đầu vào được đưa vào lớp đầu vào và lan truyền qua các lớp ẩn đến lớp đầu ra. Tại mỗi nơ-ron (trừ lớp đầu vào), nó nhận tín hiệu từ các nơ-ron ở lớp trước, tính toán tổng trọng số của các tín hiệu đầu vào, cộng thêm giá trị thiên vị, sau đó áp dụng hàm kích hoạt cho kết quả này và truyền tín hiệu đi tới lớp tiếp theo. Quá trình này tạo ra dự đoán ở lớp đầu ra.
* Lan truyền ngược (Backpropagation): Đây là thuật toán học cốt lõi của hầu hết các ANN. Sau khi có dự đoán từ lan truyền tiến, mạng tính toán lỗi (sự khác biệt giữa giá trị dự đoán và giá trị thực tế). Lỗi này sau đó được lan truyền ngược lại qua mạng, từ lớp đầu ra về lớp đầu vào. Trong quá trình lan truyền ngược, gradient của hàm lỗi theo từng trọng số và thiên vị được tính toán. Các trọng số và thiên vị này sau đó được cập nhật theo hướng làm giảm lỗi, thường sử dụng một thuật toán tối ưu hóa như Gradient Descent hoặc các biến thể tiên tiến hơn như Adam, RMSprop. Quá trình lan truyền tiến và lan truyền ngược được lặp lại nhiều lần (qua nhiều epochs) trên tập dữ liệu huấn luyện để mô hình học được cách thực hiện nhiệm vụ.

Các loại ANN: Có nhiều kiến trúc ANN khác nhau được thiết kế cho các loại dữ liệu và nhiệm vụ cụ thể:

* Mạng nơ-ron truyền thẳng (FeedForward Neural Networks - FFNN / Multi-Layer Perceptrons - MLP): Thông tin chỉ di chuyển theo một hướng, từ đầu vào đến đầu ra, không có chu trình. Đây là loại mạng phổ biến nhất.
* Mạng nơ-ron tích chập (Convolutional Neural Networks - CNN): Được thiết kế đặc biệt cho dữ liệu có cấu trúc dạng lưới (ví dụ: hình ảnh). Sử dụng các lớp tích chập để tự động học các đặc trưng không gian phân cấp.
* Mạng nơ-ron hồi quy (Recurrent Neural Networks - RNN): Được thiết kế cho dữ liệu tuần tự (ví dụ: văn bản, chuỗi thời gian). Có các kết nối tạo thành chu trình, cho phép thông tin từ các bước thời gian trước đó được lưu giữ và ảnh hưởng đến các bước sau (tạo ra "bộ nhớ").

Một khía cạnh lý thuyết quan trọng của ANN là khả năng xấp xỉ hàm phổ quát (Universal Approximation Theorem). Định lý này phát biểu rằng một mạng nơ-ron truyền thẳng với ít nhất một lớp ẩn và một hàm kích hoạt phi tuyến phù hợp có thể xấp xỉ bất kỳ hàm liên tục nào với độ chính xác tùy ý. Điều này có nghĩa là ANN có khả năng học các mối quan hệ cực kỳ phức tạp và phi tuyến trực tiếp từ dữ liệu.26 Các hiện tượng trong thế giới thực thường rất phức tạp và phi tuyến. Trong khi các mô hình truyền thống như Hồi quy tuyến tính giả định các dạng hàm cụ thể (ví dụ: tuyến tính), ANN, thông qua cấu trúc phân lớp, kết nối có trọng số và các hàm kích hoạt phi tuyến, có thể xây dựng các ánh xạ phức tạp từ đầu vào đến đầu ra. Thuật toán lan truyền ngược cho phép mạng tự động điều chỉnh trọng số và thiên vị để học ánh xạ này từ các ví dụ huấn luyện mà không cần giả định trước về dạng của mối quan hệ. Khả năng lý thuyết này làm cho ANN trở nên cực kỳ mạnh mẽ và linh hoạt cho nhiều nhiệm vụ mà quá trình tạo dữ liệu cơ bản là không xác định hoặc phức tạp. Tuy nhiên, sức mạnh này đi kèm với những thách thức: chúng thường đòi hỏi lượng lớn dữ liệu, tài nguyên tính toán đáng kể để huấn luyện, tinh chỉnh cẩn thận và dễ bị overfitting nếu không được chính quy hóa đúng cách. Bản chất "hộp đen" của chúng cũng gây khó khăn cho việc diễn giải.

**2.2.4.2 Cách xác định tham số**

Việc xây dựng và huấn luyện ANN đòi hỏi phải xác định nhiều tham số và siêu tham số. Cần phân biệt giữa các tham số của mô hình (trọng số và thiên vị), được học từ dữ liệu trong quá trình huấn luyện, và các siêu tham số, được thiết lập trước khi quá trình huấn luyện bắt đầu và kiểm soát kiến trúc cũng như quá trình học. Các framework như TensorFlow và Keras giúp đơn giản hóa việc xây dựng, huấn luyện và tinh chỉnh ANN. Các tham số quan trọng cần xác định bao gồm:

* + Kiến trúc mạng (Network Architecture):
    - * Số lớp ẩn (Number of Hidden Layers): Quyết định độ "sâu" của mạng. Mạng sâu hơn có thể học các đặc trưng phân cấp phức tạp hơn, nhưng khó huấn luyện hơn (dễ gặp vấn đề vanishing/exploding gradients) và tăng nguy cơ overfitting.
      * Số nơ-ron mỗi lớp ẩn (Number of Neurons per Hidden Layer): Quyết định "độ rộng" của mỗi lớp. Nhiều nơ-ron hơn làm tăng năng lực của mô hình (khả năng biểu diễn) nhưng cũng tăng chi phí tính toán và nguy cơ overfitting. Thường được xác định bằng thực nghiệm hoặc dựa trên các quy tắc kinh nghiệm.
  + Hàm kích hoạt (Activation Functions): Lựa chọn hàm kích hoạt cho các nơ-ron ở lớp ẩn và lớp đầu ra. ReLU là lựa chọn phổ biến cho các lớp ẩn do hiệu quả tính toán và khả năng giảm thiểu vấn đề vanishing gradient. Sigmoid hoặc Softmax thường dùng cho lớp đầu ra trong bài toán phân loại. Hàm tuyến tính (không kích hoạt) thường dùng cho lớp đầu ra trong bài toán hồi quy.
  + Thuật toán tối ưu hóa (Optimizer): Thuật toán được sử dụng để cập nhật trọng số và thiên vị nhằm giảm thiểu hàm mất mát (ví dụ: SGD, Adam, RMSprop). Adam thường là một lựa chọn mặc định tốt và hiệu quả trong nhiều trường hợp.
  + Tốc độ học (Learning Rate): Kích thước bước cập nhật trọng số trong mỗi lần lặp của thuật toán tối ưu hóa. Đây là một siêu tham số cực kỳ quan trọng. Nếu quá cao, quá trình học có thể phân kỳ hoặc dao động mạnh quanh điểm tối ưu. Nếu quá thấp, quá trình học sẽ rất chậm. Thường yêu cầu tinh chỉnh cẩn thận. Các thuật toán tối ưu hóa thích ứng như Adam, RMSprop có khả năng tự điều chỉnh tốc độ học ở một mức độ nào đó.
  + Số Epochs: Số lần toàn bộ tập dữ liệu huấn luyện được đưa qua mạng để huấn luyện. Quá ít epochs có thể dẫn đến underfitting (mô hình chưa học đủ). Quá nhiều epochs có thể dẫn đến overfitting (mô hình học cả nhiễu trong dữ liệu huấn luyện). Số epochs tối ưu thường được xác định bằng kỹ thuật "early stopping", tức là dừng huấn luyện khi hiệu suất trên tập kiểm định (validation set) bắt đầu giảm.
  + Kích thước lô (Batch Size): Số lượng mẫu huấn luyện được xử lý trong một lần cập nhật trọng số. Kích thước lô ảnh hưởng đến tốc độ huấn luyện, việc sử dụng bộ nhớ và độ ổn định/nhiễu của gradient. Kích thước lô nhỏ hơn có thể dẫn đến gradient nhiễu hơn nhưng đôi khi giúp mô hình thoát khỏi các điểm tối ưu cục bộ và tổng quát hóa tốt hơn. Kích thước lô lớn hơn cung cấp ước lượng gradient ổn định hơn và có thể tăng tốc độ huấn luyện (đặc biệt trên GPU) nhưng có thể dẫn đến khả năng tổng quát hóa kém hơn và dễ bị mắc kẹt ở các điểm cực tiểu "sắc" (sharp minima). Các giá trị phổ biến thường là lũy thừa của 2 (ví dụ: 32, 64, 128).
  + Kỹ thuật chính quy hóa (Regularization Techniques): Mặc dù không hoàn toàn là siêu tham số theo nghĩa truyền thống, việc lựa chọn và cấu hình các kỹ thuật chính quy hóa là rất quan trọng để ngăn chặn overfitting. Các phương pháp phổ biến bao gồm chính quy hóa L1/L2 (thêm hình phạt vào hàm mất mát dựa trên độ lớn của trọng số) và Dropout (tạm thời vô hiệu hóa một tỷ lệ ngẫu nhiên các nơ-ron trong quá trình huấn luyện).

Việc tìm ra tổ hợp siêu tham số tối ưu là rất quan trọng đối với hiệu suất của ANN và thường đòi hỏi quá trình thử nghiệm đáng kể. Các kỹ thuật như Grid Search, Random Search hoặc các phương pháp tiên tiến hơn như Tối ưu hóa Bayes (Bayesian Optimization) thường được sử dụng để tự động hóa quá trình tìm kiếm này.

**2.2.4.3 Ưu và nhược điểm**

Ưu điểm:

* Sức mạnh và Linh hoạt: Có khả năng mô hình hóa các mối quan hệ phi tuyến tính, phức tạp cao mà các mô hình khác có thể bỏ lỡ. Là các bộ xấp xỉ hàm phổ quát.
* Học đặc trưng tự động: Các mạng sâu (đặc biệt là CNN, RNN) có thể tự động học các đặc trưng phân cấp hữu ích từ dữ liệu thô, giảm bớt nhu cầu về kỹ thuật đặc trưng thủ công tốn công sức.
* Hiệu suất hàng đầu: Đạt được hiệu suất tiên tiến (state-of-the-art) trong nhiều lĩnh vực đòi hỏi cao như thị giác máy tính, xử lý ngôn ngữ tự nhiên, nhận dạng giọng nói.
* Khả năng thích ứng: Có thể điều chỉnh cho phù hợp với nhiều loại dữ liệu khác nhau (hình ảnh, văn bản, chuỗi thời gian, dữ liệu dạng bảng) thông qua các kiến trúc mạng khác nhau (CNN, RNN, MLP, Transformer,...).
* Khả năng mở rộng (với phần cứng): Các framework hiện đại như TensorFlow, PyTorch cho phép tận dụng phần cứng chuyên dụng (GPU, TPU) để huấn luyện các mô hình lớn trên các tập dữ liệu khổng lồ.

Nhược điểm:

* Phức tạp: Có thể khó thiết kế, tinh chỉnh và hiểu được cơ chế hoạt động bên trong.
* Bản chất "Hộp đen": Rất khó diễn giải tại sao mạng đưa ra một quyết định cụ thể, gây khó khăn trong các ứng dụng đòi hỏi tính minh bạch.
* Đòi hỏi nhiều dữ liệu: Thường yêu cầu một lượng lớn dữ liệu huấn luyện có gán nhãn để đạt được hiệu suất tốt và tránh overfitting.
* Tốn kém tài nguyên tính toán: Quá trình huấn luyện có thể đòi hỏi thời gian dài và tài nguyên tính toán đáng kể (CPU, GPU, bộ nhớ).
* Nhạy cảm với siêu tham số: Hiệu suất của mạng rất nhạy cảm với việc lựa chọn kiến trúc mạng và các siêu tham số huấn luyện. Quá trình tinh chỉnh có thể tốn nhiều thời gian và công sức.
* Nguy cơ Overfitting cao: Do tính linh hoạt và năng lực biểu diễn lớn, ANN rất dễ bị overfitting nếu không được kiểm soát cẩn thận bằng các kỹ thuật chính quy hóa phù hợp.
* Yêu cầu chuyên môn: Việc triển khai và tối ưu hóa ANN hiệu quả thường đòi hỏi kiến thức và kinh nghiệm chuyên sâu.

## **2.3 Các thuật toán tối ưu hóa**

### **2.3.1 Thuật toán Grid Search**

Định nghĩa và Nguyên lý: Grid Search (Tìm kiếm Lưới) là một kỹ thuật tìm kiếm toàn diện và có hệ thống trên một tập hợp các giá trị siêu tham số rời rạc do người dùng xác định trước. Nó tạo ra một "lưới" từ tất cả các tổ hợp giá trị có thể có và đánh giá hiệu năng mô hình (thường qua đánh giá chéo) cho từng tổ hợp. Mục tiêu là tìm ra tổ hợp tốt nhất trong lưới.

Ưu điểm:

* Toàn diện: Đảm bảo tìm thấy tổ hợp tốt nhất trong phạm vi lưới đã xác định.
* Đơn giản: Dễ hiểu và triển khai.
* Song song hóa: Dễ dàng chạy song song các đánh giá.

Nhược điểm:

* Chi phí Tính toán Cao: Cực kỳ tốn kém khi không gian tìm kiếm lớn (nhiều siêu tham số hoặc nhiều giá trị), chịu ảnh hưởng của "lời nguyền về số chiều".
* Kém hiệu quả: Lãng phí tài nguyên nếu chỉ một vài siêu tham số quan trọng.
* Khám phá Hạn chế: Chỉ kiểm tra các điểm rời rạc, có thể bỏ lỡ giá trị tốt hơn ở giữa.

### **2.3.2 Thuật toán Random Search**

Định nghĩa và Nguyên lý: Random Search (Tìm kiếm Ngẫu nhiên) lấy mẫu ngẫu nhiên một số lượng cố định (n\_iter) các cấu hình siêu tham số từ các phân phối xác suất hoặc phạm vi giá trị được xác định trước. Thay vì thử mọi tổ hợp, nó chọn ngẫu nhiên n\_iter bộ tham số, đánh giá chúng (thường qua đánh giá chéo) và trả về bộ tốt nhất đã tìm thấy.

Ưu điểm:

* Hiệu quả Tính toán: Thường nhanh hơn đáng kể so với Grid Search, đặc biệt ở không gian nhiều chiều. Chi phí được kiểm soát bởi n\_iter.
* Hiệu quả ở không gian nhiều chiều: Thường tìm thấy kết quả tốt hoặc tương đương Grid Search với ít lần đánh giá hơn, đặc biệt khi chỉ một vài tham số quan trọng.
* Khám phá Rộng hơn: Có thể khám phá các giá trị liên tục hiệu quả hơn.
* Song song hóa: Dễ dàng chạy song song.

Nhược điểm:

* Không Đảm bảo Tối ưu: Do tính ngẫu nhiên, không đảm bảo tìm thấy giải pháp tốt nhất tuyệt đối.
* Biến động: Kết quả có thể thay đổi giữa các lần chạy (trừ khi dùng random seed).
* Bao phủ Không đều: Có thể bỏ lỡ các vùng tối ưu hẹp nếu n\_iter quá nhỏ.

## **2.4 So sánh Grid Search và Random Search**

|  |  |  |
| --- | --- | --- |
| **Đặc điểm** | **Grid Search** | **Random Search** |
| **Chiến lược tìm kiếm** | |  | | --- | | Tìm kiếm toàn bộ tất cả các tổ hợp có thể |  |  | | --- | |  | | |  | | --- | | Lấy mẫu ngẫu nhiên các tổ hợp siêu tham số |  |  | | --- | |  | |
| **Chi phí tính toán** | |  | | --- | | Tốn kém, đặc biệt với không gian siêu tham số lớn |  |  | | --- | |  | | |  | | --- | | Ít tốn kém hơn, có thể giới hạn số lần lặp |  |  | | --- | |  | |
| **Hiệu quả** | |  | | --- | | Có thể không hiệu quả với không gian lớn |  |  | | --- | |  | | |  | | --- | | Hiệu quả hơn đối với không gian siêu tham số lớn |  |  | | --- | |  | |
| **Đảm bảo tối ưu** | Có (trong lưới) | Không (xác suất) |
| **Tìm giải pháp tốt** | Phụ thuộc lưới | Thường tìm thấy giải pháp tốt/gần tối ưu nhanh |

Bảng 2 Bảng so sánh Grid Search và Random Search

Nghiên cứu cho thấy Random Search thường hiệu quả hơn, tìm được kết quả tương đương hoặc tốt hơn Grid Search với chi phí tính toán thấp hơn đáng kể, đặc biệt khi không gian siêu tham số lớn và chỉ một vài tham số thực sự quan trọng.

Lựa chọn:

* Grid Search: Thích hợp khi không gian siêu tham số nhỏ, các tham số mang tính rời rạc, có hiểu biết tốt từ trước về phạm vi giá trị hợp lý, và có đủ tài nguyên tính toán để thực hiện tìm kiếm toàn diện.
* Random Search: Ưu tiên sử dụng khi không gian siêu tham số lớn, bao gồm nhiều tham số liên tục, ngân sách tính toán hạn chế, hoặc khi nghi ngờ rằng chỉ một số ít tham số có ảnh hưởng đáng kể đến hiệu suất mô hình.

CHƯƠNG 3: PHÂN TÍCH DỮ LIỆU VÀ CHẠY THUẬT TOÁN

## **3.1. Phân tích dữ liệu đầu vào**

Dữ liệu bao gồm các loại thông tin sau:

### **3.1.1 Thông tin định danh và cơ bản:**

* Mã ĐTN: Mã định danh của thí sinh/học sinh.
* Giới tính: Giới tính của học sinh (Nam/Nữ).

### **3.1.2 Kết quả học tập chi tiết theo từng năm học**

* Điểm tổng kết: Điểm trung bình kết thúc các học kỳ (HK I, HK II) và cả năm (CN) cho từng năm lớp 10, 11, 12.
* Học lực: Đánh giá học lực (Giỏi, Khá, Trung bình, Yếu) theo từng học kỳ và cả năm cho từng năm lớp 10, 11, 12.
* Hạnh kiểm: Đánh giá hạnh kiểm (Tốt, Khá, Trung bình, Yếu) theo từng học kỳ và cả năm cho từng năm lớp 10, 11, 12.
* Điểm các môn học cụ thể: Điểm theo từng học kỳ và cả năm cho các môn: Toán, Văn, Vật lí, Hóa học, Sinh học, Lịch sử, Địa lí, GDCD, Ngoại ngữ.
* Môn ngoại ngữ: Thông tin về môn ngoại ngữ mà học sinh học (ví dụ: N1).

### **3.1.3 Điểm liên quan đến kỳ thi HSA**

* d1, d2, d3: Đây có thể là điểm hoặc kết quả dự đoán cho các tổ hợp môn/bài thi cụ thể của kỳ thi HSA. Dựa trên tên file kết quả, d1 là điểm thi môn Toán, d2 là điểm thi môn Văn, và d3 là điểm thi môn Khoa học.
* diem: Cột này là tổng điểm thi thực tế của 3 đầu điểm d1, d2, d3.

Tóm lại, tập dữ liệu cung cấp một cái nhìn toàn diện về quá trình học tập 3 năm THPT của học sinh, bao gồm cả các yếu tố định tính (học lực, hạnh kiểm) và định lượng (điểm số chi tiết các môn và điểm tổng kết), cùng với các cột điểm được cho là liên quan đến kỳ thi HSA. Đây là cơ sở dữ liệu phù hợp để xây dựng và đánh giá các mô hình dự đoán kết quả HSA dựa trên quá trình học tập của học sinh.

A screenshot of a computer

AI-generated content may be incorrect.

Hình 3.1 Dữ liệu tiền xử lí

## **3.2. Xử lý và chuẩn hóa dữ liệu**

### **3.2.1 Loại bỏ Cột không cần thiết:**

* Các cột định danh và thông tin không dùng cho mô hình bị loại bỏ: 'Mã ĐTN', 'Giới tính', '10.Môn ngoại ngữ', '11.Môn ngoại ngữ'.
* Các cột điểm chi tiết theo học kỳ (HK I, HK II), học lực, hạnh kiểm cũng bị loại bỏ để chỉ giữ lại điểm tổng kết cả năm (CN) của các môn học.

### **3.2.2 Xử lý Dữ liệu thiếu (Missing Values):**

* Kiểm tra và phát hiện một số giá trị thiếu (1-2 giá trị mỗi cột) trong các cột điểm của lớp 10, 11, 12.
* Các hàng chứa giá trị thiếu này bị loại bỏ hoàn toàn khỏi DataFrame bằng df.dropna(inplace=True).

### **3.2.3 Tính toán Feature Trung bình Môn học (Feature Engineering):**

* Điểm Trung bình 3 năm: Tính điểm trung bình 3 năm (lớp 10, 11, 12) cho các môn học chính: Toán, Văn, Vật lí, Hóa học, Sinh học, Lịch sử, Địa lí, GDCD, Ngoại ngữ.
  + Công thức: Sử dụng công thức tính trung bình có trọng số, ưu tiên điểm lớp 12:

​.

* + Các cột điểm lớp 10, 11, 12 ban đầu của từng môn được thay thế bằng cột điểm trung bình mới (ví dụ: 'Toán\_TB').
* Điểm Trung bình Khối Khoa học:
  + Tính điểm trung bình Khoa học Xã hội (KHXH\_TB) từ trung bình của 'Lịch sử\_TB' và 'Địa lí\_TB'.
  + Tính điểm trung bình Khoa học Tự nhiên (KHTN\_TB) từ trung bình của 'Vật lí\_TB', 'Hóa học\_TB', và 'Sinh học\_TB'.
  + Các cột điểm môn thành phần và 'GDCD\_TB' bị loại bỏ.
* Điểm Trung bình Khoa học Tổng hợp: Tính điểm KH\_TB bằng trung bình cộng của KHXH\_TB và KHTN\_TB. Sau đó loại bỏ các cột KHXH\_TB, KHTN\_TB và Ngoại ngữ\_TB.

A screenshot of a spreadsheet

AI-generated content may be incorrect.

Hình 3.2 Dữ liệu sau khi đã xử lí

Sau khi đã xử lí dữ liệu và tính toán theo công thức thì em đã thu được 1 file chưa điểm trung bình môn toán (Toán\_TB), môn văn (Văn\_TB), môn khoa học (KH\_TB) của từng học sinh trong dữ liệu đã cho. Và label của điểm d1 (Điểm thi toán), d2 (Điểm thi văn), d3 (Điểm thi khoa học) và diem (Tổng điểm của 3 phần thi.

**3.2.4 Kết quả dữ liệu sau khi làm sạch**Dữ liệu cuối cùng bao gồm các cột:

* Features: ‘Toán\_TB’, ‘Văn\_TB’, ‘KH\_TB’ và labels: ‘d1’,’d2’,’d3’,’diem
* DataFrame đã làm sạch được lưu vào file HSA\_HD\_Final\_Cleaned.csv.

### **3.2.5 Trực quan hóa dữ liệu thành biểu đồ**

Tiếp theo em sẽ sử dụng thư viện matplotlib và thư viên seaborn để trực quan hóa dữ liệu đã cho.

A screenshot of a computer

AI-generated content may be incorrect.

Hình 3.3 Xu hướng điểm trung bình của học sinh qua các lớp 10,11,12

A screenshot of a computer

AI-generated content may be incorrect.

Hình 3.4 Mật độ điểm tổng kết của học sinh theo từng lớp

A screenshot of a graph

AI-generated content may be incorrect.

Hình 3.5 Xu hướng điểm trung bình của học sinh theo học kì

A screenshot of a graph

AI-generated content may be incorrect.

Hình 3.6 Phân bổ điểm của học sinh theo từng lớp

### **3.2.6 Cách tiến hành xử lý dữ liệu với 4 mô hình**

**3.2.6.1 Mô hình Hồi quy Tuyến tính (Linear Regression)**

Quy trình huấn luyện mô hình Linear Regression được xây dựng dưới dạng pipeline gồm hai bước: chuẩn hóa đặc trưng bằng StandardScaler và huấn luyện mô hình LinearRegression. Tập dữ liệu đầu vào được chia theo đặc trưng phù hợp: chỉ sử dụng cột Toán\_TB cho d1, Văn\_TB cho d2, KH\_TB cho d3, và toàn bộ 3 cột cho tổng điểm.

Tối ưu hóa mô hình được thực hiện bằng GridSearchCV với các siêu tham số:

* fit\_intercept: [True, False] – Cho biết có tính hệ số chặn hay không.
* positive: [True, False] – Ép hệ số hồi quy phải dương.

Các tham số tốt nhất sau khi tìm kiếm:

* Model d1 (Toán): fit\_intercept=True, positive=True
* Model d2 (Văn): fit\_intercept=True, positive=True
* Model d3 (KHTN): fit\_intercept=True, positive=True
* Model Tổng điểm: fit\_intercept=True, positive=False

**3.2.6.2 Mô hình Máy Vector Hỗ trợ Hồi quy (SVR)**

Với SVR, pipeline huấn luyện gồm bước chuẩn hóa dữ liệu đầu vào và mô hình SVR. Quá trình tinh chỉnh sử dụng GridSearchCV trên các tham số:

* C: [1, 10, 100] – Tham số điều chuẩn, điều khiển độ phạt lỗi.
* gamma: ['scale', 0.1, 1] – Xác định phạm vi ảnh hưởng của điểm dữ liệu.
* kernel: ['rbf'] – Kernel phi tuyến phổ biến.

Các tham số tốt nhất được chọn bởi hệ thống:

* Model d1 (Toán): C=10, gamma=0.1, kernel='rbf'
* Model d2 (Văn): C=1, gamma=0.1, kernel='rbf'
* Model d3 (KHTN): C=10, gamma=0.1, kernel='rbf'
* Model Tổng điểm: C=10, gamma=0.1, kernel='rbf'

**3.2.6.3 Mô hình Rừng Ngẫu nhiên Hồi quy (Random Forest)**

Random Forest được huấn luyện qua pipeline chuẩn hóa kết hợp với mô hình RandomForestRegressor. Việc tinh chỉnh được thực hiện với GridSearchCV trên không gian tham số mở rộng:

* n\_estimators: [100, 200, 300, 500] – Số lượng cây trong rừng.
* max\_depth: [10, 20, 30, None] – Độ sâu tối đa của mỗi cây.
* min\_samples\_split: [2, 5, 10, 20] – Số mẫu tối thiểu để tách nút.
* min\_samples\_leaf: [1, 2, 4, 8] – Số mẫu tối thiểu tại một nút lá.
* max\_features: ['sqrt', 'log2', 1.0] – Số đặc trưng xem xét khi chia nhánh.

Tham số tốt nhất tìm được cho từng mô hình:

* Model d1 (Toán): max\_depth=20, max\_features=’sqrt’, min\_samples\_leaf=8, min\_samples\_split=20, n\_estimators=100
* Model d2 (Văn): max\_depth=10, max\_features=’sqrt’, min\_samples\_leaf=8, min\_samples\_split=20, n\_estimators=200
* Model d3 (KHTN): max\_depth=10, max\_features=’sqrt’, min\_samples\_leaf=8, min\_samples\_split=2, n\_estimators=300
* Model Tổng điểm: max\_depth=10, max\_features=’sqrt’, min\_samples\_leaf=8, min\_samples\_split=2, n\_estimators=500

**3.2.6.4 Mô hình Mạng Neural Nhân tạo (ANN)**

Mô hình ANN được xây dựng bằng Keras, không sử dụng GridSearch mà thiết kế kiến trúc thủ công dựa trên kinh nghiệm và thực hành phổ biến. Dữ liệu đầu vào được chuẩn hóa bằng StandardScaler. Cấu trúc mạng bao gồm:

* Lớp Dense (128 units, relu) – học đặc trưng cơ bản.
* Lớp Dense (64 units, relu) – học đặc trưng trung gian.
* Lớp Dropout (tỷ lệ 0.3) – chống overfitting.
* Lớp Dense (32 units, relu) – học đặc trưng sâu.
* Lớp Dense (1 unit) – đầu ra hồi quy.

Cấu hình huấn luyện:

* Optimizer: Adam, learning rate: 0.001
* Loss function: mse
* Epochs: 500, Batch size: 16
* Validation split: 0.1, EarlyStopping: patience=30

Một mô hình riêng biệt được huấn luyện cho từng mục tiêu: d1 (Toán), d2 (Văn), d3 (KHTN) và tổng điểm. Việc huấn luyện dừng sớm nếu không cải thiện trên tập kiểm tra, đảm bảo chất lượng mô hình được chọn.

## **3.3 Huấn luyện và đánh giá mô hình**

Sau khi hoàn tất bước tiền xử lý dữ liệu, nhóm tiến hành huấn luyện các mô hình học máy nhằm dự đoán điểm từng môn học cũng như tổng điểm của học sinh. Dữ liệu được chia thành hai tập huấn luyện (80%) và kiểm thử (20%) để đảm bảo đánh giá mô hình một cách khách quan. Trong số các mô hình được triển khai, mô hình Hồi quy véctơ hỗ trợ (Support Vector Regression – SVR) cho thấy hiệu quả vượt trội. SVM (hay SVR) được huấn luyện theo pipeline gồm chuẩn hóa dữ liệu bằng StandardScaler và tối ưu siêu tham số bằng GridSearchCV, sử dụng kernel RBF để mô hình hóa các quan hệ phi tuyến. Kết quả kiểm thử cho thấy SVM (hay SVR) đạt MAE trung bình khoảng 3.39–3.94 điểm và RMSE từ 4.24–4.77 điểm cho từng môn học, đồng thời đạt MAE chỉ khoảng 7.96 và RMSE 9.82 cho tổng điểm, với độ chính xác lên đến 89.19%. Những con số này cho thấy SVM(hay SVR) không chỉ có khả năng khái quát tốt mà còn ổn định và phù hợp với bài toán dự đoán điểm học sinh trong bối cảnh dữ liệu thực tế có độ nhiễu và phân bố không đồng đều.

|  |  |  |  |
| --- | --- | --- | --- |
| Mô hình | MAE | RMSE | Độ chính xác |
| Linear Regression | | | 84.36% |
| - Model d1 (Toán) | 4.45 | 5.49 |
| - Model d2 (Văn) | 3.29 | 4.20 |
| - Model d3 (KH) | 4.03 | 5.06 |
| - Tổng điểm | 8.47 | 10.43 |
| SVM | | | 89.19% |
| - Model d1 (Toán) | 3.94 | 4.77 |
| - Model d2 (Văn) | 3.39 | 4.24 |
| - Model d3 (KH) | 3.73 | 4.59 |
| - Tổng điểm | 7.96 | 9.82 |
| Random Forest | | | 84.32% |
| - Model d1 (Toán) | 4.03 | 4.92 |
| - Model d2 (Văn) | 3.58 | 4.40 |
| - Model d3 (KH) | 4.07 | 5.04 |
| - Tổng điểm | 8.61 | 10.45 |
| ANN | | | 88.83% |
| - Model d1 (Toán) | 4.25 | 5.19 |
| - Model d2 (Văn) | 3.26 | 4.09 |
| - Model d3 (KH) | 3.78 | 4.73 |
| - Tổng điểm | 8.17 | 10 |

Bảng 3 So sánh độ lỗi và độ chính xác của từng mô hình

Giải thích các chỉ số:

* MAE (Mean Absolute Error): Thể hiện sai số trung bình tuyệt đối của mô hình (đơn vị điểm).
* RMSE (Root Mean Squared Error): Cũng thể hiện sai số nhưng nhấn mạnh vào các lỗi lớn hơn (do bình phương). Các mô hình đều có RMSE lớn hơn MAE một chút, cho thấy có một số điểm dự đoán lệch nhiều so với thực tế.
* Độ chính xác: Tỷ lệ dự đoán đúng của mô hình.

Nhận xét:

* Mô hình d2 (Văn học) có sai số thấp nhất trong hầu hết các mô hình.
* Mô hình d1 (Toán học) có sai số cao nhất trong hầu hết các mô hình. Điều này có thể do sự chênh lệch điểm Toán cao giữa điểm trên học bạ và điểm thi thực tế của học sinh hoặc biên độ đề thi lớn.
* Mô hình d3 (Khoa học) có sai số ở mức trung bình, nằm ở giữa mô hình d1 và d2
* Mô hình tổng điểm có MAE và RMSE ở mức chấp nhận được (thường <= 10).
* Mô hình SVM(hay SVR) có độ chính xác chung cao nhất (89.19%).

CHƯƠNG 4: XÂY DỰNG TRANG WEB

## **4.1 Thiết kế giao diện web**

### **4.1.1 Tích hợp giao diện với backend**

Ứng dụng được xây dựng theo mô hình client-server, trong đó giao diện người dùng (frontend) chạy trên trình duyệt của người dùng và backend chạy trên máy chủ sử dụng framework Flask.

Quá trình tiếp xử lí thông tin như sau:

1. Giao diện gửi yêu cầu: Khi người dùng tương tác với giao diện (ví dụ: điền thông tin vào form và nhấn nút, tải lên file), trình duyệt sẽ gửi một yêu cầu HTTP (có thể là GET hoặc POST) đến địa chỉ của ứng dụng backend. Yêu cầu này mang theo thông tin mà người dùng đã nhập hoặc file đã tải lên.

2. Backend xử lý yêu cầu: Framework Flask ở backend sẽ tiếp nhận yêu cầu này. Dựa vào địa chỉ URL được yêu cầu (ví dụ: /predict, /upload) và phương thức HTTP (GET, POST), Flask sẽ chuyển yêu cầu đến hàm xử lý tương ứng trong mã nguồn app.py.

3. Xử lý logic nghiệp vụ: Bên trong hàm xử lý, backend sẽ:

* Trích xuất dữ liệu được gửi từ frontend (ví dụ: các giá trị từ form, nội dung file).
* Thực hiện các tác vụ cần thiết (ví dụ: gọi mô hình dự đoán, đọc dữ liệu từ file, ghi dữ liệu vào file).
* Truy cập và xử lý dữ liệu (ví dụ: sử dụng các model dự đoán, lưu kết quả vào file CSV).

4. Backend gửi phản hồi: Sau khi xử lý xong, backend tạo ra phản hồi HTTP và gửi trả lại cho trình duyệt của người dùng. Phản hồi này có thể là:

* Một trang HTML mới được tạo động (sử dụng template engine như Jinja2) hiển thị kết quả dự đoán, bảng dữ liệu, hoặc thông báo.
* Dữ liệu dạng JSON (ví dụ: cho các yêu cầu báo lỗi) để giao diện người dùng xử lý bằng JavaScript.

5. Giao diện hiển thị phản hồi: Trình duyệt nhận phản hồi từ backend và hiển thị nội dung tương ứng cho người dùng (ví dụ: hiển thị kết quả trên trang web, hiển thị thông báo thành công/thất bại).

Như vậy, việc tích hợp được thực hiện chủ yếu thông qua việc trao đổi các yêu cầu và phản hồi HTTP giữa trình duyệt (frontend) và máy chủ (backend) theo các địa chỉ và quy tắc đã được định nghĩa trong mã nguồn Flask. Backend đóng vai trò xử lý logic và dữ liệu, còn frontend tập trung vào việc thu thập thông tin đầu vào và hiển thị kết quả cho người dùng.

**4.1.2 Xây dựng trang web nhập điểm và file Excel**

**4.1.2.1 Giao diện chính của trang web**

A screenshot of a computer

AI-generated content may be incorrect.

Hình 4.1 Giao diện nhập tải và nhập file excel

A screenshot of a computer

AI-generated content may be incorrect.

Hình 4.2 Mẫu excel nhập điểm

A screenshot of a computer

AI-generated content may be incorrect.

Hình 4.3 Giao diện kết quả dự đoán sau khi nhập file excel

A screenshot of a computer

AI-generated content may be incorrect.

Hình 4. 4 Giao diện nhập điểm trực tiếp trên trang web

A screenshot of a computer

AI-generated content may be incorrect.

Hình 4.5 Giao diện kết quả sau khi nhập điểm trục tiếp trên web

## **4.2. Tích hợp mô hình và dự đoán**

### **4.2.1 Triển khai mô hình tối ưu trên backend**

Lựa chọn Mô hình Dự đoán Tối ưu

* Trong quá trình phát triển hệ thống, em đã tiến hành huấn luyện và đánh giá một số mô hình học máy tiềm năng cho bài toán dự đoán điểm HSA, bao gồm Mạng Nơ-ron Nhân tạo (Artificial Neural Network – ANN), Hồi quy Tuyến tính (Linear Regression), Máy hỗ trợ Vector (SVM) và Rừng ngẫu nhiên (Random Forest). Các mô hình này được so sánh dựa trên các chỉ số hiệu suất quan trọng như độ chính xác dự đoán và mức độ sai số trên tập dữ liệu kiểm thử. Sau khi phân tích kỹ lưỡng kết quả đánh giá, mô hình Máy hỗ trợ Vector (SVM) đã cho thấy hiệu năng vượt trội nhất, đạt được độ chính xác cao hơn và có sai số dự đoán thấp hơn đáng kể so với các mô hình còn lại. Dựa trên kết quả thực nghiệm này, em đã quyết định lựa chọn mô hình SVM là mô hình chính thức để triển khai trong ứng dụng.
* Mô hình SVM đã được lựa chọn sau quá trình đánh giá hiệu năng đã được tích hợp trực tiếp vào phần backend của ứng dụng, được xây dựng sử dụng framework Flask. Việc tích hợp này cho phép mô hình SVM sẵn sàng nhận các yêu cầu dự đoán từ giao diện người dùng. Khi người dùng nhập thông tin đầu vào và gửi yêu cầu dự đoán, phần backend sẽ gọi đến mô hình SVM đã được nạp để thực hiện dự đoán dựa trên dữ liệu đầu vào đó. Kết quả dự đoán từ mô hình SVM sau đó sẽ được xử lý bởi backend và gửi trả lại giao diện người dùng để hiển thị. Mô hình SVM được cấu hình làm mô hình dự đoán mặc định trong ứng dụng để đảm bảo các yêu cầu dự đoán đơn lẻ được xử lý bởi mô hình có hiệu năng tốt nhất.

## **4.3 Xử lý file Excel và nhập dữ liệu**

Để thuận tiện cho người dùng trong việc nhập liệu, hệ thống cung cấp hai phương thức nhập dữ liệu:

* Nhập liệu thủ công: Người dùng có thể trực tiếp điền điểm trung bình từng môn (Toán, Ngữ văn, Khoa học) vào form nhập liệu trên website.
* Tải lên file Excel/CSV: Người dùng có thể tải lên tệp chứa điểm học bạ theo định dạng mẫu có sẵn.

Trong đó, việc xử lý file Excel được thực hiện thông qua thư viện pandas. Sau khi file được tải lên qua giao diện người dùng, hệ thống tự động đọc và kiểm tra từng dòng dữ liệu, bỏ qua dòng tiêu đề đầu tiên và chỉ lấy 3 cột điểm chính (Toán, Văn, Khoa học). Mỗi dòng được chuyển đổi thành các giá trị số thực (float) và kiểm tra điều kiện hợp lệ (trong khoảng 0–10 điểm). Nếu dữ liệu hợp lệ, hệ thống sẽ đưa vào quy trình dự đoán điểm thi.

## **4.4 Dự báo điểm từ dữ liệu nhập**

Sau khi người dùng nhập điểm hoặc tải lên file dữ liệu, hệ thống tiến hành dự báo điểm thi HSA (kỳ thi đánh giá năng lực) theo ba lĩnh vực:

* D1 – Tư duy định lượng (Toán)
* D2 – Tư duy định tính (Văn)
* D3 – Tư duy khoa học (KHTN/KHXH)
* Tổng điểm HSA = D1 + D2 + D3

Quy trình dự đoán gồm các bước:

1. Tiền xử lý dữ liệu: Kiểm tra điểm số trong khoảng 0–10, đảm bảo dữ liệu không bị thiếu hoặc sai định dạng.
2. Dự đoán điểm thi: Mỗi tổ hợp điểm được truyền vào mô hình và trả về 4 kết quả: điểm D1, D2, D3 và tổng điểm.
3. Hiển thị kết quả: Dữ liệu đầu vào và kết quả dự đoán được trình bày rõ ràng trên giao diện người dùng.

KẾT LUẬN

Qua quá trình thực hiện đồ án tốt nghiệp với đề tài "Nghiên cứu mô hình AI dự đoán điểm thi đánh giá năng lực dựa vào điểm học bạ của học sinh", em đã có cơ hội áp dụng những kiến thức đã học vào một bài toán thực tiễn có ý nghĩa trong lĩnh vực giáo dục. Đồ án đã hoàn thành các mục tiêu chính như: thu thập và xử lý dữ liệu học bạ, xây dựng và so sánh các mô hình học máy như SVM, Linear Regression, Random Forest để dự đoán điểm thi HSA, đồng thời triển khai thành công một hệ thống web có giao diện thân thiện, hỗ trợ cả nhập liệu thủ công và tải file Excel. Hệ thống cho phép người dùng xem kết quả dự đoán từng môn học, tổng điểm, đồng thời phản hồi nhận xét để cải thiện mô hình.

Trong quá trình nghiên cứu, em nhận thấy mô hình SVM cho kết quả ổn định nhất trên tập dữ liệu hiện có, vì vậy được chọn làm mô hình mặc định của hệ thống. Việc triển khai ứng dụng web bằng Flask cũng giúp em hiểu rõ hơn về cách tích hợp mô hình AI vào sản phẩm thực tế. Bên cạnh đó, đồ án cũng góp phần thể hiện tiềm năng của AI trong việc hỗ trợ học sinh và giáo viên trong việc định hướng học tập, giúp giảm bớt áp lực từ các kỳ thi quan trọng.

Tuy nhiên, em cũng nhận thấy đồ án vẫn còn một số hạn chế như dữ liệu huấn luyện còn ít. Trong tương lai, em mong muốn có thể tiếp tục mở rộng hệ thống, cải thiện độ chính xác và tính cá nhân hóa của mô hình dự đoán, đồng thời tích hợp chức năng theo dõi tiến độ học tập để hệ thống trở nên toàn diện hơn.

Thông qua đề tài này, em đã tích lũy thêm được nhiều kinh nghiệm về xử lý dữ liệu, xây dựng mô hình AI, triển khai ứng dụng thực tế và đặc biệt là tư duy phân tích, giải quyết vấn đề một cách hệ thống. Em tin rằng đây sẽ là nền tảng quan trọng cho con đường học tập và phát triển nghề nghiệp của em trong tương lai.

TÀI LIỆU THAM KHẢO

Tiếng Việt:

[1] Vũ Hữu Tiệp (2018). Học máy. Nhà xuất bản Khoa học và Kỹ thuật

Tiếng Anh:

[1] GeeksforGeeks. (2023). Random Forest Algorithm – Explained with Examples.

[2] Microsoft Azure. (2022). What is an Artificial Neural Network?.

[3] IBM. (2023). What is Support Vector Machine (SVM)?

[4] Towards Data Science. (2022). Linear Regression in Machine Learning – A Complete Guide.

Danh mục website tham khảo:

[1]<https://viblo.asia/p/linear-regression-hoi-quy-tuyen-tinh-trong-machine-learning-4P856akRlY3>

[2]<https://viblo.asia/p/gioi-thieu-ve-support-vector-machine-svm-6J3ZgPVElmB>

[3]<https://viettelidc.com.vn/tin-tuc/cam-nang-ai-artificial-neural-network-la-gi-cau-truc-cach-hoat-dong-va-ung-dung-cua-mo-hinh-nay>