# **PHẦN A : CƠ SỞ LÝ THUYẾT**

1. **NỘI DUNG**

* **Artificial Intelligence là gì ?** 
  + Định nghĩa trí tuệ nhân tạo và các khía cạnh khác nhau của AI, bao gồm khả năng nhận thức, học tập, và hành động của máy móc.
  + AI là khả năng của máy móc để thực hiện các nhiệm vụ mà trước đây chỉ con người mới có thể làm được, chẳng hạn như nhận diện hình ảnh, hiểu ngôn ngữ, và ra quyết định.
* **Lĩnh vực chính :** 
  + Machine Learning (Học máy): Phương pháp giúp máy móc học hỏi từ dữ liệu và cải thiện hiệu suất mà không cần phải lập trình lại.
  + Natural Language Processing (Xử lý ngôn ngữ tự nhiên): Khả năng máy tính hiểu và tương tác với con người bằng ngôn ngữ tự nhiên.
  + Computer Vision (Nhận diện hình ảnh): Kỹ thuật giúp máy tính nhận diện và phân tích hình ảnh từ thế giới thực.
  + Robotics (Robot học): Nghiên cứu và phát triển các robot có khả năng thực hiện các nhiệm vụ vật lý.
* **Ứng dụng :** 
  + Chăm sóc sức khỏe: Phân tích dữ liệu y tế, chẩn đoán bệnh và phát triển các phương pháp điều trị.
  + Giao thông: Tự động hóa phương tiện, tối ưu hóa lộ trình và quản lý giao thông.
  + Tài chính: Dự đoán xu hướng thị trường, phát hiện gian lận và quản lý danh mục đầu tư.
  + Giải trí: Tạo nội dung, cá nhân hóa trải nghiệm người dùng và phân tích dữ liệu người dùng.
* **Các cách giải quyết vấn đề :** 
  + Tìm kiếm :
    - Tìm kiếm là một phương pháp cơ bản để giải quyết vấn đề trong AI, đặc biệt là trong các bài toán mà không thể giải quyết bằng cách tính toán trực tiếp.
    - Mô hình không gian trạng thái là cách để đại diện cho tất cả các khả năng của vấn đề và các trạng thái mà hệ thống có thể đạt được.
    - Tìm kiếm theo chiều sâu (Depth-First Search): Khám phá các nhánh của cây tìm kiếm từ gốc đến các nút con trước khi quay lại.
    - Tìm kiếm theo chiều rộng (Breadth-First Search): Khám phá tất cả các nút ở mức độ hiện tại trước khi chuyển sang mức độ tiếp theo.
    - Tìm kiếm có chiều sâu giới hạn (Depth-Limited Search): Tìm kiếm theo chiều sâu nhưng giới hạn số lượng mức độ mà nó sẽ khám phá.
    - Tìm kiếm theo chi phí (Uniform-Cost Search): Tìm kiếm để đạt được trạng thái với chi phí thấp nhất.
    - *Tìm kiếm A (A* Search):\*\* Tìm kiếm tối ưu bằng cách kết hợp cả chi phí thực tế và dự đoán chi phí còn lại đến mục tiêu.
    - Heuristic Functions: Các hàm heuristic được sử dụng để ước lượng khoảng cách còn lại đến mục tiêu và hướng dẫn quá trình tìm kiếm.
    - Tìm kiếm thông minh (Informed Search): Tìm kiếm sử dụng thông tin bổ sung từ các hàm heuristic để cải thiện hiệu suất.
    - Tìm kiếm các vấn đề động (Dynamic Programming): Giải quyết các vấn đề mà có thể chia nhỏ thành các phần con và lưu trữ kết quả để tránh tính toán lại.
    - Ví dụ về việc sử dụng các kỹ thuật tìm kiếm trong các bài toán như trò chơi cờ, tìm đường đi trong bản đồ, và các bài toán lập kế hoạch.
  + Logic suy luận :
    - Tác nhân logic là những hệ thống sử dụng các phương pháp logic để hiểu và phản ứng với môi trường của chúng. Họ dựa vào các quy tắc logic để suy luận và ra quyết định.
    - Giới thiệu về cách logic có thể được sử dụng để đại diện cho tri thức và quy trình suy luận trong các tác nhân. Logic cung cấp một nền tảng chính xác để diễn đạt các quy tắc và mối quan hệ.
    - Logic Khẳng định (Propositional Logic): Một dạng logic cơ bản trong đó các câu lệnh logic được xây dựng từ các biến đơn giản và các phép toán logic như AND, OR, và NOT.
    - Công thức và Quy tắc: Cách viết các công thức và quy tắc logic để biểu diễn các mối quan hệ và quy tắc trong môi trường của tác nhân.
    - Suy luận Gián tiếp (Inference): Quy trình suy luận từ các công thức logic để tạo ra các kết luận mới. Các phương pháp suy luận gián tiếp như phương pháp kéo dài và phương pháp giả định.
    - Bảng Suy luận: Sử dụng bảng suy luận để xác minh tính đúng đắn của các công thức và suy luận từ các quy tắc.
    - Tác nhân Logic và Hệ thống Thực thi: Các cách mà các tác nhân logic có thể được thiết kế để hoạt động trong môi trường thực tế, bao gồm cả việc triển khai các hệ thống thực thi logic.
    - Khái niệm và Cú pháp: Giới thiệu về logic bậc một (First-Order Logic), nơi mà các công thức logic có thể bao gồm các đối tượng, thuộc tính và mối quan hệ giữa chúng.
    - Quy tắc và Suy luận: Các quy tắc suy luận trong logic bậc một và cách chúng có thể được áp dụng để giải quyết các vấn đề phức tạp hơn.
    - Hệ thống Đại diện Kiến thức: Cách mà các hệ thống logic có thể được sử dụng để đại diện cho tri thức và thông tin trong các tác nhân.
    - Suy luận và Quy tắc Thực thi: Các phương pháp để thực thi suy luận trong các hệ thống logic và cách các quy tắc có thể được áp dụng để đưa ra các quyết định.
  + Lập kế hoạch:
    - **Lập Kế Hoạch và Hành Động:** Lập kế hoạch là quá trình tìm ra một chuỗi hành động để đạt được mục tiêu, trong khi hành động là việc thực hiện các kế hoạch này trong môi trường thực tế.
    - **Các Vấn Đề Thực Tế:** Các vấn đề lập kế hoạch không chỉ bao gồm việc tìm ra các hành động hiệu quả mà còn phải đối mặt với sự không chắc chắn và các yếu tố thay đổi trong môi trường.
    - **Lập Kế Hoạch Cổ Điển (Classical Planning):** Các phương pháp lập kế hoạch cổ điển sử dụng các mô hình đơn giản hóa để tìm ra các kế hoạch tối ưu. Các phương pháp này có thể bao gồm việc sử dụng các cây tìm kiếm và các thuật toán lập kế hoạch như STRIPS.
    - **Lập Kế Hoạch Động (Dynamic Planning):** Đề cập đến việc lập kế hoạch trong các môi trường động, nơi mà các yếu tố có thể thay đổi trong suốt quá trình thực hiện kế hoạch.
    - **Quản Lý Sự Không Chắc Chắn:** Các phương pháp lập kế hoạch cần phải quản lý sự không chắc chắn trong môi trường thực tế. Điều này có thể bao gồm việc dự đoán các trạng thái tương lai và xử lý các tình huống không lường trước được.
    - **Các Kỹ Thuật Lập Kế Hoạch Dự Đoán:** Sử dụng các mô hình dự đoán để ước lượng các kết quả của các hành động và điều chỉnh kế hoạch dựa trên thông tin mới.
    - **Phản Hồi và Điều Chỉnh:** Cần có khả năng phản hồi và điều chỉnh kế hoạch khi môi trường thay đổi hoặc khi có thông tin mới. Điều này bao gồm việc xử lý các phản hồi từ môi trường và cập nhật kế hoạch để phù hợp với tình huống hiện tại.
    - **Tích Hợp Cảm Biến và Hành Động:** Cách tích hợp thông tin từ các cảm biến để cải thiện quá trình lập kế hoạch và hành động.
    - **Các Ví Dụ Thực Tế:** Cung cấp các ví dụ về lập kế hoạch và hành động trong các tình huống thực tế, chẳng hạn như lập kế hoạch cho robot tự hành, hệ thống điều khiển giao thông, và các ứng dụng khác trong thế giới thực.
  + **Xử lý ngôn ngữ:**
    - **Ngữ âm học (Phonology):** Xử lý các âm thanh ngôn ngữ và cách chúng được biểu diễn.
    - **Hình thái học (Morphology):** Nghiên cứu cấu trúc của từ, như gốc từ, tiền tố, và hậu tố.
    - **Cú pháp học (Syntax):** Xử lý cấu trúc câu và các quy tắc sắp xếp từ trong câu.
    - **Ngữ nghĩa học (Semantics):** Hiểu nghĩa của từ và câu trong ngữ cảnh cụ thể.
    - **Ngữ dụng học (Pragmatics):** Hiểu ý nghĩa dựa trên ngữ cảnh và cách mà ngôn ngữ được sử dụng.
    - **Phương pháp dựa trên quy tắc:** Sử dụng các quy tắc ngôn ngữ cụ thể để phân tích cú pháp và hiểu ngôn ngữ.
    - **Phương pháp dựa trên thống kê:** Sử dụng học máy và dữ liệu lớn để xây dựng các mô hình ngôn ngữ dự đoán.
    - **Phương pháp dựa trên mạng nơ-ron:** Sử dụng các mô hình học sâu như mạng nơ-ron để học các đặc trưng ngôn ngữ phức tạp.
    - **Hệ thống hỏi đáp:** Hỗ trợ tìm kiếm thông tin dựa trên câu hỏi tự nhiên của người dùng.
    - **Dịch máy (Machine Translation):** Tự động dịch ngôn ngữ này sang ngôn ngữ khác.
    - **Phân tích cảm xúc:** Xác định cảm xúc và ý nghĩa ẩn chứa trong văn bản.
    - **Trợ lý ảo và chatbot:** Tương tác với con người thông qua văn bản hoặc giọng nói.
    - **Đa nghĩa:** Từ và câu có thể có nhiều nghĩa khác nhau dựa trên ngữ cảnh.
    - **Sự thay đổi ngôn ngữ:** Ngôn ngữ không ngừng phát triển và thay đổi, làm cho việc xử lý ngôn ngữ trở nên khó khăn hơn.
    - **Hiểu ngữ cảnh:** Máy tính phải hiểu ý nghĩa dựa trên các yếu tố ngữ cảnh rộng hơn, không chỉ từ hay câu riêng lẻ.
  + Nhận diện :
    - **Nhận diện là gì?**: Khả năng của máy móc trong việc cảm nhận và phân tích thông tin từ môi trường thông qua các giác quan nhân tạo như thị giác, thính giác, xúc giác, v.v.
    - **Mục tiêu:** Giúp hệ thống AI cảm nhận môi trường như con người và phản ứng phù hợp.
    - **Cảm biến và hình ảnh:** Cách mà máy tính sử dụng camera và các cảm biến khác để thu thập dữ liệu hình ảnh từ môi trường.
    - **Phân tích hình ảnh:** Sử dụng các kỹ thuật xử lý hình ảnh để nhận dạng đối tượng, phát hiện cạnh, phân đoạn hình ảnh, và theo dõi chuyển động.
    - **Nhận diện khuôn mặt và đối tượng:** Phương pháp nhận diện các đặc điểm khuôn mặt và đối tượng từ hình ảnh và video.
    - **Deep learning trong thị giác máy tính:** Mạng nơ-ron tích chập (CNN) được sử dụng để tự động học các đặc trưng trong hình ảnh và thực hiện các nhiệm vụ nhận dạng phức tạp.
    - **Nhận diện giọng nói (Speech Recognition):** Phương pháp để chuyển đổi âm thanh giọng nói thành văn bản.
    - **Xử lý tín hiệu âm thanh:** Các kỹ thuật phân tích âm thanh như trích xuất đặc trưng, xử lý tín hiệu thời gian thực.
    - **Phân loại âm thanh:** Máy tính có thể phân loại và hiểu các âm thanh từ môi trường như tiếng ồn, giọng nói con người, và âm thanh tự nhiên.
    - **Cảm biến xúc giác:** Sử dụng các cảm biến để giúp robot cảm nhận vật thể thông qua xúc giác.
    - **Phản hồi từ môi trường:** Robot sử dụng dữ liệu từ cảm biến để điều chỉnh hành động của mình trong môi trường thực tế.
    - **Nhiễu loạn:** Dữ liệu cảm nhận từ môi trường thường chứa nhiễu, cần có các kỹ thuật lọc và xử lý.
    - **Đa dạng trong môi trường:** Thế giới thực luôn thay đổi, phức tạp và không hoàn hảo, làm cho việc nhận diện trở nên khó khăn hơn.

1. **TÌM HIỂU VỀ THUẬT TOÁN**

* **Thuật toán tìm kiếm theo chiều sâu có hạn (DLS)** 
  + Khái niệm :
    - DFS khám phá các nhánh của cây tìm kiếm bằng cách đi sâu vào từng nhánh một trước khi quay lại và khám phá các nhánh khác.
    - DFS có thể bị mắc kẹt trong các nhánh vô tận hoặc trong các vòng lặp nếu không có cơ chế dừng.
  + Cách hoạt động :
    - DLS đặt ra một mức giới hạn về độ sâu, gọi là depth limit. Khi độ sâu của một nhánh vượt quá giới hạn này, thuật toán sẽ dừng tìm kiếm ở nhánh đó và quay lại để kiểm tra các nhánh khác.
    - Bằng cách này, DLS có thể tránh việc đi sâu vô tận trong các cây tìm kiếm có các nhánh vô hạn hoặc vòng lặp.
  + Các bước thực hiện:
    - Bước 1: Bắt đầu từ nút gốc (root).
    - Bước 2: Tiếp tục đi sâu vào các nhánh con của nút hiện tại, nhưng chỉ đến khi đạt đến giới hạn độ sâu đã được định trước.
    - Bước 3: Nếu đạt đến độ sâu giới hạn mà không tìm thấy nút mục tiêu, thuật toán sẽ dừng tìm kiếm ở nhánh đó và quay lại để kiểm tra nhánh khác.
    - Bước 4: Lặp lại quá trình này cho đến khi tìm thấy nút mục tiêu hoặc kiểm tra hết các nhánh trong cây tìm kiếm.

A computer screen with white text

Description automatically generated

*Hình ảnh thuật toán*

* + Ưu nhược điểm của thuật toán ;
    - Tránh vòng lặp vô hạn: Do có giới hạn độ sâu, DLS không bị mắc kẹt trong các vòng lặp vô hạn.
    - Sử dụng ít bộ nhớ: Giống như DFS, DLS chỉ lưu trạng thái của các nút đang được thăm, do đó không yêu cầu quá nhiều bộ nhớ như các thuật toán tìm kiếm theo chiều rộng (BFS).
    - Không đảm bảo tìm được lời giải tối ưu: Nếu nút mục tiêu nằm ngoài phạm vi của giới hạn độ sâu, thuật toán sẽ không tìm thấy lời giải.
    - Khó chọn giới hạn đúng: Việc chọn đúng giới hạn độ sâu là một vấn đề khó khăn. Nếu giới hạn quá nhỏ, thuật toán có thể không tìm thấy mục tiêu. Nếu giới hạn quá lớn, thuật toán có thể tốn nhiều thời gian để tìm kiếm.
  + Ứng dụng thuật toán:
    - DLS thường được sử dụng trong các bài toán mà không gian tìm kiếm có thể rất lớn hoặc không giới hạn, như tìm kiếm trong các cây trò chơi, lập kế hoạch robot, hoặc các hệ thống với trạng thái phức tạp.
    - Thuật toán này cũng là một thành phần quan trọng trong tìm kiếm lặp sâu tăng dần (Iterative Deepening Search - IDS), một thuật toán tìm kiếm hiệu quả kết hợp ưu điểm của DLS và BFS.
* **Thuật toán tìm kiếm sâu dần (IDS)**
  + Khái niệm:
    - Tìm kiếm sâu dần (IDS) là một thuật toán kết hợp giữa tìm kiếm theo chiều rộng (Breadth-First Search - BFS) và tìm kiếm theo chiều sâu có hạn (Depth-Limited Search - DLS). Nó có mục đích tìm kiếm toàn bộ không gian trạng thái một cách hiệu quả bằng cách thực hiện các tìm kiếm theo chiều sâu lặp đi lặp lại, nhưng giới hạn độ sâu của tìm kiếm tăng dần theo mỗi lần lặp
  + Ý tưởng của thuật toán
    - IDS thực hiện các tìm kiếm theo chiều sâu có hạn (DLS) với giới hạn độ sâu ban đầu là 0, sau đó tăng dần giới hạn độ sâu theo mỗi lần lặp.
    - Ở mỗi lần lặp, thuật toán tìm kiếm đến độ sâu hiện tại và nếu không tìm thấy giải pháp, nó sẽ tăng giới hạn độ sâu thêm 1 và thực hiện lại tìm kiếm.
    - IDS thừa hưởng ưu điểm tiết kiệm bộ nhớ của DFS và đảm bảo tìm kiếm được lời giải tối ưu như BFS, nhưng không yêu cầu lưu trữ nhiều trạng thái trong bộ nhớ.
  + Cách hoạt động:
    - Bước 1: Bắt đầu từ độ sâu giới hạn (depth limit) bằng 0.
    - Bước 2: Thực hiện tìm kiếm theo chiều sâu (DFS) với giới hạn độ sâu hiện tại.
    - Bước 3: Nếu tìm thấy lời giải, dừng lại và trả về kết quả.
    - Bước 4: Nếu không tìm thấy, tăng giới hạn độ sâu lên 1 và lặp lại quá trình tìm kiếm.
    - Bước 5: Lặp lại quá trình cho đến khi tìm thấy lời giải.

A computer screen with white text

Description automatically generated

*Hình ảnh thuật toán*

* + Ưu nhược điểm của thuật toán:
    - Tối ưu Bộ Nhớ: IDS sử dụng ít bộ nhớ hơn so với BFS vì nó không cần lưu trữ toàn bộ cây tìm kiếm trong bộ nhớ. IDS chỉ lưu trạng thái của các nút đang được thăm trong nhánh tìm kiếm hiện tại.
    - Tìm được Giải pháp Tối Ưu: Giống như BFS, IDS tìm ra giải pháp tối ưu nếu các bước chuyển đều có chi phí bằng nhau (do nó kiểm tra tất cả các nút ở mức độ sâu thấp hơn trước khi tăng độ sâu).
    - Tránh Vòng Lặp Vô Tận: IDS không mắc kẹt trong các vòng lặp vô hạn, vì nó tăng dần độ sâu và không đi sâu vô tận như DFS.
    - Tính Toán Lặp Lại: IDS lặp lại nhiều lần các nút ở độ sâu thấp hơn khi nó tăng dần giới hạn độ sâu. Ví dụ, nếu giới hạn tăng từ 0 đến 5, IDS sẽ thăm lại các nút ở độ sâu 1, 2, và 3 nhiều lần. Tuy nhiên, chi phí của sự lặp lại này thường không quá lớn do số lượng nút tăng theo cấp số mũ khi độ sâu tăng.
  + Ứng dụng:
    - IDS thường được sử dụng trong các bài toán tìm kiếm với không gian lớn, nơi mà cả DFS và BFS đều gặp các hạn chế về bộ nhớ hoặc thời gian.
    - Nó thường được áp dụng trong các bài toán tìm kiếm giải pháp cho các trò chơi, các bài toán đồ thị, hoặc trong lập kế hoạch cho robot.

1. **ĐỘ PHỨC TẠP CỦA THUẬT TOÁN**

* **Thuật toán tìm kiếm chiều sâu (DFS - Depth-First Search)**
  + Khái niệm:
    - Thuật toán DFS là một phương pháp duyệt hoặc tìm kiếm trên cây hoặc đồ thị bằng cách đi sâu vào các nhánh con trước khi quay lại các nhánh ngang cấp. DFS sử dụng cấu trúc dữ liệu ngăn xếp (stack) hoặc có thể được triển khai dễ dàng với đệ quy (recursive).
  + Độ phức tạp thời gian O(V + E) với :
    - V là số đỉnh (vertices).
    - E là số cạnh (edges).
    - Vì DFS thăm mỗi đỉnh và mỗi cạnh đúng một lần, nên độ phức tạp là tuyến tính.
    - DFS duyệt qua tất cả các đỉnh và các cạnh trong đồ thị. Với mỗi đỉnh, DFS sẽ thăm tất cả các cạnh của nó ít nhất một lần, do đó, thời gian thực thi phụ thuộc vào tổng số đỉnh và cạnh trong đồ thị.
  + Độ phức tạp không gian O(V) với:
    - O(V), trong đó V là số lượng đỉnh.
    - DFS sử dụng một ngăn xếp (stack) để lưu các đỉnh cần thăm. Trong trường hợp xấu nhất, số lượng đỉnh lưu trong ngăn xếp có thể lên tới số đỉnh của đồ thị, dẫn đến độ phức tạp không gian tuyến tính theo số đỉnh.
  + Đánh giá :
    - DFS thích hợp với các bài toán yêu cầu tìm kiếm một đường đi bất kỳ từ đỉnh bắt đầu đến đỉnh mục tiêu, đặc biệt trong trường hợp không cần tìm đường đi ngắn nhất.
    - DFS sử dụng ít bộ nhớ, nhưng trong đồ thị vô hạn hoặc có vòng lặp, thuật toán có thể đi vào vòng lặp vô hạn nếu không có biện pháp giới hạn.
  + Cách thuật toán thực hiện :
    - Đánh dấu đỉnh đó là đã được thăm.
    - Nếu đã đến đỉnh mục tiêu, dừng thuật toán.
    - Nếu không, tiếp tục thăm các đỉnh con của đỉnh mới này.
    - Tiếp tục thăm các đỉnh còn lại trên nhánh cha cho đến khi đã thăm hết tất cả các đỉnh.
* **Thuật toán tìm kiếm chiều rộng (BFS - Breadth-First Search)** 
  + Khái niệm :
    - BFS là một thuật toán tìm kiếm duyệt đồ thị hoặc cây theo cấp độ (layer) từ nông đến sâu. BFS sử dụng cấu trúc dữ liệu hàng đợi (queue) để quản lý các đỉnh sẽ được thăm tiếp theo.
  + Độ phức tạp thời gian O(V + E) với :
    - V là số đỉnh (vertices).
    - E là số cạnh (edges).
    - BFS duyệt qua tất cả các đỉnh và cạnh trong đồ thị một lần, do đó độ phức tạp thời gian phụ thuộc vào tổng số đỉnh và cạnh
  + Độ phức tạo không gian O(V):
    - O(V), trong đó V là số lượng đỉnh.
    - BFS sử dụng hàng đợi (queue) để lưu các đỉnh cần thăm. Trong trường hợp xấu nhất, tất cả các đỉnh trong đồ thị có thể nằm trong hàng đợi, dẫn đến độ phức tạp không gian tuyến tính theo số lượng đỉnh
  + Đánh giá:
    - BFS đảm bảo tìm được đường đi ngắn nhất trong đồ thị không trọng số, và thích hợp cho các bài toán cần tối ưu hóa đường đi.
    - Nhược điểm chính của BFS là nó tiêu tốn nhiều bộ nhớ hơn DFS, đặc biệt khi đồ thị có nhiều nhánh.
  + Cách thuật toán hoạt động :
    - Đánh dấu đỉnh này là đã được thăm và đưa nó vào hàng đợi.
    - Lấy đỉnh đầu tiên trong hàng đợi ra để thăm.
    - Thêm tất cả các đỉnh kề của nó mà chưa được thăm vào hàng đợi.
    - Đánh dấu các đỉnh đó là đã được thăm.
    - Dừng lại khi đến đỉnh mục tiêu hoặc hàng đợi rỗng.
* **Thuật toán A\* (A-star)**
  + Khái niệm :
    - A\* là một thuật toán tìm kiếm dựa trên heuristic, nghĩa là nó sử dụng một hàm đánh giá để ước tính chi phí đến đích. A\* sử dụng cả chi phí thực sự đã đi từ điểm bắt đầu đến một đỉnh nào đó g(n) và một hàm ước lượng chi phí từ đỉnh đó đến đích h(n) (hàm heuristic).
  + Độ phức tạp thời gian O(b ^ d) với:
    - b là hệ số nhánh (branching factor).
    - d là độ sâu của lời giải.
    - Độ phức tạp có thể xấu đi nếu hàm heuristic không tốt, khiến thuật toán phải thăm quá nhiều đỉnh.
    - Trong trường hợp tốt hơn, nếu hàm heuristic h(n) được thiết kế tốt và đủ chính xác (hàm "đơn điệu" hoặc "admissible" – không bao giờ vượt quá chi phí thực sự), thời gian thực thi có thể giảm đi đáng kể.
  + Độ phức tạp không gian O(b^d),
    - O(b^d), trong đó b là hệ số nhánh và d là độ sâu.
    - A\* sử dụng nhiều bộ nhớ để lưu các đỉnh đã thăm và các đỉnh cần thăm, nên có thể tiêu tốn nhiều bộ nhớ khi không gian tìm kiếm lớn.
  + Đánh giá:
    - A\* là một trong những thuật toán tìm kiếm hiệu quả nhất cho các bài toán tìm kiếm đường đi ngắn nhất, đặc biệt khi hàm heuristic được thiết kế tốt.
    - Tuy nhiên, A\* có thể tiêu tốn nhiều bộ nhớ và thời gian khi không gian tìm kiếm lớn và hàm heuristic không đủ chính xác.
  + Cách thuật toán hoạt động :
    - Khởi tạo một tập mở (open set) chứa đỉnh gốc, với giá trị g(n) là 0 và h(n) được tính theo hàm heuristic.
    - Lấy đỉnh có giá trị f(n) nhỏ nhất từ tập mở.
    - Nếu đỉnh này là đỉnh mục tiêu, dừng lại và trả về kết quả.
    - Nếu không, chuyển đỉnh này sang tập đóng (closed set).
    - Thêm các đỉnh kề của đỉnh hiện tại vào tập mở và tính giá trị f(n) cho từng đỉnh.
    - Dừng lại khi tìm thấy đỉnh mục tiêu hoặc không còn đỉnh nào để thăm.

1. **TÌM HIỂU MÔ HÌNH HỌC MÁY**

* **Khái niệm :** 
  + Mô hình máy học (machine learning model) là một hệ thống được huấn luyện để học từ dữ liệu và đưa ra dự đoán hoặc quyết định dựa trên dữ liệu đó. Các mô hình máy học thường được sử dụng để giải quyết các bài toán liên quan đến phân loại, hồi quy, cụm, và các vấn đề học tập không giám sát khác. Việc xây dựng một mô hình máy học bao gồm các bước chính như: thu thập dữ liệu, tiền xử lý dữ liệu, chọn mô hình, huấn luyện mô hình, và đánh giá mô hình
* **Phân loại:** 
  + Học Có Giám Sát (Supervised Learning) :
    - Trong học có giám sát, dữ liệu được cung cấp dưới dạng các cặp (input, output), trong đó:
    - Input: Là dữ liệu đầu vào (các biến độc lập).
    - Output: Là nhãn hoặc giá trị đích (các biến phụ thuộc).
    - Mô hình sẽ học cách ánh xạ từ input sang output dựa trên dữ liệu huấn luyện. Sau khi được huấn luyện, mô hình có thể dự đoán output cho những input mới.
    - Ví dụ về các bài toán học có giám sát:
    - Phân loại (Classification): Dự đoán nhãn của một đối tượng. Ví dụ: phân loại email là spam hay không.
    - Hồi quy (Regression): Dự đoán một giá trị liên tục. Ví dụ: dự đoán giá nhà dựa trên diện tích, số phòng, v.v.
    - Các thuật toán học có giám sát phổ biến:
      * Hồi quy tuyến tính (Linear Regression)
      * Hồi quy logistic (Logistic Regression)
      * Cây quyết định (Decision Tree)
      * Máy vector hỗ trợ (Support Vector Machine - SVM)
      * Mạng nơ-ron nhân tạo (Artificial Neural Networks)
  + Học Không Giám Sát (Unsupervised Learning)
    - Trong học không giám sát, dữ liệu chỉ bao gồm các input mà không có output tương ứng. Mục tiêu của mô hình là tìm ra các mẫu hoặc cấu trúc ẩn trong dữ liệu mà không có sự hướng dẫn từ nhãn.
    - Ví dụ về các bài toán học không giám sát:
      * Phân cụm (Clustering): Tìm ra các nhóm đối tượng tương đồng trong dữ liệu. Ví dụ: phân cụm khách hàng thành các nhóm dựa trên hành vi mua sắm.
      * Giảm chiều dữ liệu (Dimensionality Reduction): Giảm số lượng đặc trưng trong dữ liệu để dễ dàng trực quan hóa hoặc giảm độ phức tạp. Ví dụ: sử dụng PCA để giảm chiều dữ liệu ảnh.
    - Các thuật toán học không giám sát phổ biến:
      * K-Means Clustering
      * Hierarchical Clustering (Phân cụm phân cấp)
      * Principal Component Analysis (PCA)
      * Autoencoders (Mạng tự mã hóa)
  + Học Bán Giám Sát (Semi-supervised Learning)
    - Học bán giám sát là sự kết hợp giữa học có giám sát và học không giám sát. Dữ liệu huấn luyện chỉ có một phần nhỏ được gán nhãn, còn lại là dữ liệu không gán nhãn. Mô hình sẽ học từ cả hai loại dữ liệu để cải thiện độ chính xác.
    - Ứng dụng : Học bán giám sát thường được sử dụng trong những tình huống mà việc gán nhãn dữ liệu tốn kém hoặc khó khăn, ví dụ như nhận diện hình ảnh, phân loại video.
  + Học Tăng Cường (Reinforcement Learning):
    - Học tăng cường là một loại hình học máy dựa trên việc ra quyết định trong một môi trường động. Một agent (tác nhân) sẽ tương tác với môi trường và học cách hành động thông qua quá trình thử và sai để tối ưu hóa phần thưởng dài hạn.
    - Ví dụ về các ứng dụng học tăng cường:
      * Robot tự hành học cách điều hướng.
      * Các chương trình AI như AlphaGo, chơi cờ vua hay các trò chơi điện tử.
    - Các thuật toán học tăng cường phổ biến:
      * Q-Learning
      * Deep Q-Networks (DQN)
      * Policy Gradient Methods
* **Quy trinh xây dựng:**
  + **Thu thập dữ liệu**: Thu thập dữ liệu đủ lớn và chất lượng cao để mô hình có thể học được. Dữ liệu có thể bao gồm các dạng như: hình ảnh, văn bản, số liệu.
  + **Tiền xử lý dữ liệu**: Trước khi đưa vào mô hình, dữ liệu cần được làm sạch và chuẩn hóa. Điều này bao gồm xử lý các giá trị thiếu, chuyển đổi các biến, hoặc thực hiện chuẩn hóa các đặc trưng.
  + **Chọn mô hình**: Dựa trên bài toán cụ thể (phân loại, hồi quy, cụm, v.v.), lựa chọn một mô hình phù hợp để áp dụng.
  + **Huấn luyện mô hình**: Sử dụng dữ liệu huấn luyện để huấn luyện mô hình. Quá trình này bao gồm tối ưu hóa các tham số của mô hình dựa trên các thuật toán học máy.
  + **Đánh giá mô hình**: Sử dụng dữ liệu kiểm thử để đánh giá hiệu suất của mô hình. Các chỉ số đánh giá có thể bao gồm độ chính xác, độ nhạy, độ đặc hiệu, và F1-score.
  + **Tối ưu hóa mô hình**: Điều chỉnh các tham số của mô hình hoặc lựa chọn mô hình khác tốt hơn để cải thiện hiệu suất.
  + **Triển khai mô hình**: Sau khi mô hình đã đạt hiệu suất mong muốn, nó có thể được triển khai trong thực tế để dự đoán hoặc ra quyết định.
* **Một số mô hình máy học phổ biến** :
  + Hồi Quy Tuyến Tính (Linear Regression)
    - Mô hình dự đoán giá trị của một biến phụ thuộc dựa trên một hoặc nhiều biến độc lập.
    - Thường được sử dụng trong các bài toán dự báo như dự báo giá nhà, dự đoán doanh thu.
  + Cây Quyết Định (Decision Tree)
    - Là một mô hình học có giám sát, dùng để giải quyết cả bài toán phân loại và hồi quy. Mô hình xây dựng một cấu trúc giống như cây để ra quyết định.
    - Mỗi nút trong cây đại diện cho một quyết định dựa trên một đặc trưng của dữ liệu.
  + Mạng Nơ-ron Nhân Tạo (Artificial Neural Networks - ANN)
    - Mô hình lấy cảm hứng từ cách hoạt động của bộ não con người, gồm nhiều lớp nơ-ron (neuron) liên kết với nhau.
    - Thường được sử dụng cho các bài toán phức tạp như nhận diện hình ảnh, xử lý ngôn ngữ tự nhiên.
  + Máy Học Tập Hỗ Trợ (Support Vector Machine - SVM)
    - SVM là một thuật toán phân loại mạnh mẽ, được sử dụng để phân biệt các đối tượng thuộc các lớp khác nhau bằng cách tìm một siêu phẳng tối ưu để chia tách dữ liệu.
* **Ưu điểm :**
  + Tự động hóa việc ra quyết định:
    - Mô hình máy học có khả năng tự động hóa các quy trình phức tạp, ra quyết định mà không cần sự can thiệp thủ công. Điều này giúp giảm chi phí lao động và tăng tốc quá trình xử lý.
  + Khả năng xử lý lượng dữ liệu lớn:
    - Máy học có khả năng xử lý và phân tích lượng dữ liệu khổng lồ mà con người không thể làm một cách thủ công. Điều này rất hữu ích trong các lĩnh vực như tài chính, y tế và thương mại điện tử.
  + Tăng độ chính xác và hiệu suất:
    - Mô hình máy học có thể cải thiện độ chính xác của dự đoán hoặc phân loại nhờ vào khả năng học hỏi từ dữ liệu. Các mô hình phức tạp như mạng nơ-ron (neural networks) có thể học các mẫu rất phức tạp trong dữ liệu.
  + Khả năng học hỏi liên tục:
    - Khi được cung cấp thêm dữ liệu mới, các mô hình máy học có thể tự động cải thiện hiệu suất của chúng, giúp mô hình thích nghi tốt hơn với các thay đổi trong môi trường thực tế.
  + Giảm thiểu sai sót của con người:
    - Máy học giúp giảm thiểu sai sót do con người gây ra, đặc biệt trong các công việc yêu cầu tính chính xác cao như y tế, tài chính, hoặc sản xuất.
* **Nhược điểm:**
  + Yêu cầu lượng dữ liệu lớn:
    - Các mô hình máy học, đặc biệt là các mô hình phức tạp, yêu cầu lượng dữ liệu lớn để huấn luyện và đạt hiệu quả cao. Dữ liệu ít hoặc không đầy đủ có thể dẫn đến các kết quả không chính xác.
  + Khả năng quá khớp (Overfitting):
    - Mô hình có thể học quá mức từ dữ liệu huấn luyện, khiến nó hoạt động tốt trên dữ liệu huấn luyện nhưng kém hiệu quả trên dữ liệu mới. Điều này là một vấn đề phổ biến trong máy học nếu không kiểm soát tốt.
  + Khả năng giải thích hạn chế:
    - Nhiều mô hình, đặc biệt là các mô hình phức tạp như mạng nơ-ron sâu (deep learning), khó giải thích. Điều này gây ra thách thức trong các lĩnh vực như y tế, tài chính, nơi mà các quyết định cần phải rõ ràng và có thể được giải thích.
  + Chi phí tính toán cao:
    - Các mô hình lớn, phức tạp thường yêu cầu nhiều tài nguyên tính toán, bao gồm thời gian và bộ nhớ. Việc huấn luyện các mô hình này có thể tốn kém về cả chi phí phần cứng và năng lượng.
  + Vấn đề về tính đạo đức và quyền riêng tư:
    - Máy học sử dụng dữ liệu cá nhân có thể gây ra các lo ngại về quyền riêng tư. Ví dụ, trong nhận diện khuôn mặt hoặc phân tích dữ liệu y tế, việc lạm dụng dữ liệu có thể gây ra các vấn đề pháp lý và đạo đức.
* **Ứng dụng :**
  + Y tế (Healthcare):
    - Máy học được sử dụng để chẩn đoán bệnh tật, dự đoán kết quả điều trị và phát hiện sớm các bệnh như ung thư. Các thuật toán học sâu (deep learning) có thể phân tích hình ảnh y tế, như X-quang, MRI, để nhận diện các bất thường.
    - Ví dụ: Phần mềm IBM Watson Health sử dụng máy học để giúp các bác sĩ đưa ra các quyết định điều trị dựa trên dữ liệu y khoa.
  + Tài chính (Finance):
    - Trong tài chính, máy học giúp phát hiện các giao dịch gian lận, dự đoán biến động thị trường và tối ưu hóa danh mục đầu tư. Các ngân hàng và tổ chức tài chính sử dụng các mô hình để quản lý rủi ro tín dụng và tự động phát hiện các giao dịch bất thường.
    - Ví dụ: JP Morgan Chase sử dụng máy học để phát hiện giao dịch gian lận và dự đoán các khoản nợ khó đòi.
  + Thương mại điện tử (E-commerce):
    - Máy học được sử dụng trong các hệ thống đề xuất sản phẩm, phân tích hành vi khách hàng và tối ưu hóa giá cả. Các trang web như Amazon và Netflix sử dụng thuật toán máy học để đề xuất các sản phẩm hoặc phim phù hợp với sở thích của người dùng.
    - Ví dụ: Amazon sử dụng các mô hình học máy để đề xuất sản phẩm dựa trên hành vi mua sắm của người dùng.
  + Giao thông và Tự động hóa (Transportation and Automation):
    - Xe tự hành (autonomous vehicles) là một trong những ứng dụng nổi bật của máy học. Các thuật toán học sâu giúp xe có khả năng nhận diện đối tượng, điều hướng và đưa ra quyết định trong thời gian thực.
    - Ví dụ: Tesla và Waymo đang phát triển xe tự lái sử dụng các thuật toán học sâu để điều hướng trong môi trường thực tế.
  + Trí tuệ nhân tạo và Chatbots:
    - Máy học được sử dụng trong các chatbot và trợ lý ảo để tương tác với con người một cách thông minh. Các trợ lý ảo như Siri của Apple và Alexa của Amazon sử dụng học sâu để hiểu và trả lời các câu hỏi của người dùng.
    - Ví dụ: Google Assistant sử dụng các mô hình máy học để cung cấp thông tin, đặt lịch và trả lời các câu hỏi của người dùng dựa trên dữ liệu lịch sử và ngữ cảnh.
  + Xử lý ngôn ngữ tự nhiên (Natural Language Processing - NLP):
    - Máy học giúp xử lý và phân tích văn bản, dịch ngôn ngữ, và phân tích cảm xúc. Nó được sử dụng để xây dựng các hệ thống dịch thuật tự động, phân loại văn bản và hỗ trợ phân tích cảm xúc trong các bình luận trên mạng xã hội.
    - Ví dụ: Google Translate sử dụng học máy để cải thiện chất lượng dịch thuật giữa các ngôn ngữ khác nhau.

# **PHẦN B: ỨNG DỤNG, THỰC HÀNH**

## **Chương 1 : Phát biểu và phân tích bài toán**

### **Phát biểu bài toán**

* Dự đoán liệu một sinh viên có tốt nghiệp đúng hạn hay không, dựa trên các thông tin như độ tuổi (age), điểm GPA (gpa), tỷ lệ tham gia học tập (attendance), và hỗ trợ tài chính (financial\_aid). Bài toán là một bài toán phân loại nhị phân, với nhãn đầu ra là on\_time\_graduation (1: tốt nghiệp đúng hạn, 0: không tốt nghiệp đúng hạn).
* Mục tiêu là xây dựng một mô hình để dự đoán xác suất tốt nghiệp đúng hạn của sinh viên, sau đó hiển thị kết quả dự đoán thông qua giao diện đồ họa (GUI) do thư viện Tkinter cung cấp.

### **Phân tích bài toán**

* Sử dụng thuật toán **Logistic Regression**:
  + Logistic Regression là một thuật toán học có giám sát, được sử dụng để giải quyết các bài toán phân loại nhị phân. Nó sử dụng một hàm logistic (hay sigmoid) để dự đoán xác suất của lớp nhãn đầu ra.
  + Cụ thể, hàm logistic trả về xác suất của một biến là 1 (tức là tốt nghiệp đúng hạn), và nếu xác suất này lớn hơn 0.5, mô hình dự đoán sinh viên sẽ tốt nghiệp đúng hạn, ngược lại thì không.
* Các bước thực hiện:
  + Chuẩn hóa dữ liệu: Biến đầu vào cần được chuẩn hóa (chuẩn hóa mean-variance) để đảm bảo các biến đầu vào có cùng đơn vị đo lường.
  + Huấn luyện mô hình: Mô hình hồi quy logistic được huấn luyện dựa trên các dữ liệu đã chuẩn hóa.
  + Dự đoán: Sau khi huấn luyện, mô hình sẽ dự đoán kết quả cho toàn bộ sinh viên trong tập dữ liệu.
  + Hiển thị kết quả: Kết quả dự đoán được hiển thị qua một giao diện đồ họa sử dụng Tkinter.

### **Xây dựng cấu trúc dữ liệu và các biểu diễn trạng thái**

* Input Data: Dữ liệu đầu vào là một DataFrame từ file Excel, chứa các thông tin như tuổi, GPA, tỷ lệ tham gia học tập và hỗ trợ tài chính của sinh viên.
* Feature Matrix (X): Dữ liệu đầu vào X bao gồm các cột:
  + age: Tuổi của sinh viên.
  + gpa: Điểm GPA.
  + attendance: Tỷ lệ tham gia học tập của sinh viên.
  + financial\_aid: Có nhận hỗ trợ tài chính hay không.
* Target (y): Biến mục tiêu on\_time\_graduation, xác định sinh viên có tốt nghiệp đúng hạn hay không.
* Scaled Data (X\_scaled): Sau khi chuẩn hóa các biến đầu vào, dữ liệu trở thành một ma trận chuẩn hóa.
* Predicted Graduation: Kết quả dự đoán từ mô hình Logistic Regression, được lưu vào DataFrame dưới dạng một cột predicted\_graduation.

### **Cấu trúc mã nguồn**

* Khối lệnh import sử dụng các thư viện
  + pandas: Đọc và xử lý dữ liệu từ file Excel.
  + StandardScaler: Chuẩn hóa dữ liệu đầu vào (mean-variance scaling).
  + LogisticRegression: Mô hình Logistic Regression từ thư viện scikit-learn.
  + Tkinter: Thư viện giao diện đồ họa (GUI).
  + ttk: Module mở rộng của Tkinter, dùng để tạo các widget như bảng Treeview.

A screen shot of a computer

Description automatically generated

*Hình ảnh khối lệnh import thư viện*

* Đọc dữ liệu
  + Đọc dữ liệu sinh viên từ file Excel students.xlsx vào một DataFrame df.

A black background with white text

Description automatically generated

*Hình ảnh khối lệnh đọc dữ liệu*

* Tách biến đầu vào (X) và biến mục tiêu (y):
  + X: Ma trận đặc trưng chứa các cột age, gpa, attendance, financial\_aid.
  + y: Biến mục tiêu on\_time\_graduation là cột dùng để xác định nhãn (tốt nghiệp đúng hạn hay không).

A black background with white text

Description automatically generated

*Hình ảnh khối lệnh tách biến*

* Chuẩn hóa dữ liệu:
  + Sử dụng StandardScaler để chuẩn hóa các biến đầu vào. X\_scaled là dữ liệu sau khi chuẩn hóa, mỗi cột có mean = 0 và standard deviation = 1.

A black background with white text

Description automatically generated

*Hình ảnh khối lệnh chuẩn hóa dữ liệu*

* Khởi tạo mô hình và huấn luyện mô hình :
  + Khởi tạo mô hình: Tạo một mô hình Logistic Regression.
  + Huấn luyện mô hình: Huấn luyện mô hình dựa trên dữ liệu chuẩn hóa X\_scaled và biến mục tiêu y.

A black background with white text

Description automatically generated

*Hình ảnh khối lệnh khởi tạo và huấn luyện mô hình*

* Dự đoán:
  + Sử dụng mô hình đã huấn luyện để dự đoán kết quả y\_pred (tốt nghiệp đúng hạn hoặc không) cho tất cả sinh viên trong tập dữ liệu.

A black background with white text

Description automatically generated

*Hình ảnh khối lệnh dự đoán*

* Xử lý kết quả :
  + Thêm kết quả dự đoán vào DataFrame : kết quả dự đoán predicted\_graduation và kết quả thực tế actual\_graduation được thêm vào DataFrame df.

A black background with colorful text

Description automatically generated

*Hình ảnh khối lệnh xử lý kết quả*

* Tạo giao diện kết quả:
  + Tạo cửa sổ giao diện chính bằng Tkinter với kích thước , tiêu đề .

A black background with white text

Description automatically generated

*Hình ảnh khối lệnh tạo giao diện kết quả*

* Tạo bảng Treeview:
  + Sử dụng ttk.Treeview để tạo bảng hiển thị dữ liệu sinh viên và kết quả dự đoán, bao gồm các cột: Student, Age, GPA, Attendance, Predicted, Actual, Gender, Major.

A screen shot of a computer code

Description automatically generated

*Hình ảnh khối lệnh tạo bảng Treeview*

* Thêm dữ liệu vào bảng kết quả:
  + Lặp qua từng hàng trong DataFrame df, thêm dữ liệu về sinh viên vào bảng Treeview. Kết quả dự đoán (predicted\_graduation) và kết quả thực tế (actual\_graduation) được hiển thị dưới dạng "Tốt nghiệp đúng hạn" hoặc "Không tốt nghiệp đúng hạn".

A computer screen with colorful text

Description automatically generated

*Hình ảnh khối lệnh thêm dữ liệu vào bảng kết quả*

* Trả kết quả :
  + Khởi chạy giao diện đồ họa Tkinter, cho phép tương tác với người dùng.

A screen shot of a computer program

Description automatically generated

A screenshot of a computer screen

Description automatically generated

*Hình ảnh khối lệnh chạy kết quả*

* Tóm tắt:
  + Bài toán là dự đoán liệu một sinh viên có tốt nghiệp đúng hạn dựa trên các thông tin cá nhân và học tập.
  + Mô hình Logistic Regression được sử dụng để phân loại sinh viên, dựa trên dữ liệu đầu vào đã được chuẩn hóa.
  + Kết quả dự đoán được hiển thị qua một giao diện đồ họa sử dụng Tkinter, cho phép người dùng quan sát trực tiếp dự đoán của mô hình và so sánh với kết quả thực tế.

## **Chương 2: Kết quả thực nghiệm chương trình**

### **Giao diện của chương trình**

* Giao diện màn hình chính

A screenshot of a computer

Description automatically generated

* Giao diện màn hình kết quá dự đoán

A screenshot of a computer

Description automatically generated

### **Mô phỏng 2 thực nghiệm**

* **Thực nghiệm 1 :**
  + Bước 1 : Chạy chương trình

A screenshot of a computer

Description automatically generated

* + Bước 2 : Chọn file excel dữ liệu sinh viên students.xlsx

A screenshot of a computer

Description automatically generated

* + Bước 3 : Kết quả trả về danh sách dự đoán

A screenshot of a computer

Description automatically generated

* **Thực nghiệm 2 :**
  + Bước 1 : Chạy chương trình

A screenshot of a computer

Description automatically generated

* + Bước 2 : Chọn file excel dữ liệu sinh viên liststudents.xlxs

A screenshot of a computer

Description automatically generated

* + Bước 3 : Kết quả trả về danh sách dự đoán

A screenshot of a computer

Description automatically generated

## **Chương 3: Kết luận và đánh giá kết quả thực nghiệm**

### **Đánh giá**

### Thuật toán Logistic Regression được sử dụng để giải quyết bài toán phân loại nhị phân. Nó hoạt động dựa trên việc tìm mối quan hệ giữa các biến đầu vào và biến mục tiêu thông qua hàm logistic (hay sigmoid)

* Độ chính xác: Đánh giá mức độ chính xác của mô hình dự đoán.
* Khả năng phân loại: Xem xét khả năng phân loại chính xác các sinh viên có tốt nghiệp đúng hạn hay không.
* Dữ liệu đầu vào: Kiểm tra tính đầy đủ và chất lượng của dữ liệu đầu vào để đảm bảo mô hình học được từ dữ liệu chính xác.
* Hiệu suất mô hình: So sánh kết quả dự đoán với thực tế để đánh giá hiệu suất của mô hình. Điều này có thể bao gồm phân tích ma trận nhầm lẫn và các chỉ số như Precision, Recall, và F1 Score.
* Ưu điểm:
  + Đơn giản và dễ hiểu.
  + Thực hiện nhanh chóng.
  + Có thể cung cấp xác suất cho mỗi lớp.
* Nhược điểm:
  + Không thể xử lý tốt các quan hệ phi tuyến tính giữa các biến.
  + Không mạnh khi có nhiều biến đầu vào không liên quan hoặc có mối liên hệ phức tạp.

Mô hình Logistic Regression có thể là một giải pháp tốt cho bài toán phân loại nhị phân như dự đoán khả năng tốt nghiệp đúng hạn của sinh viên. Tuy nhiên, cần phải đánh giá kết quả mô hình một cách toàn diện và có thể cần xem xét các mô hình khác nếu kết quả không đạt yêu cầu. Các yếu tố như chất lượng dữ liệu, các biến đầu vào, và các chỉ số đánh giá sẽ ảnh hưởng đến hiệu quả của mô hình.

### **Dự kiến phát triển**

* Cải thiện mô hình dự đoán:
  + Khám Phá và Tiền Xử Lý Dữ Liệu
  + Phát hiện giá trị thiếu: Kiểm tra và xử lý các giá trị thiếu trong dữ liệu.
  + Khám phá dữ liệu: Phân tích thống kê dữ liệu và các mối liên hệ giữa các biến.
  + Tính năng bổ sung: Tạo thêm các đặc trưng từ dữ liệu hiện có, chẳng hạn như tương tác giữa các biến.
* Sử Dụng Các Mô Hình Phức Tạp Hơn
  + Neural Networks: Nếu dữ liệu đủ lớn và phong phú, có thể thử các mạng nơ-ron sâu (Deep Learning).
  + Đánh giá chéo (Cross-validation): Đảm bảo rằng mô hình được đánh giá chính xác trên các tập dữ liệu khác nhau để tránh overfitting.
* **Tăng Cường Giao Diện Người Dùng**
  + Thêm Tính Năng Tải Dữ Liệu
  + Cho phép người dùng chọn và tải nhiều loại tệp khác nhau, chẳng hạn như CSV.
  + Thêm kiểm tra lỗi để đảm bảo tệp người dùng chọn là hợp lệ và chứa các dữ liệu cần thiết.
  + Cung cấp đồ thị và biểu đồ (ví dụ: Ma trận nhầm lẫn, ROC Curve) để người dùng có cái nhìn trực quan về hiệu suất mô hình.
  + Cung Cấp Phản Hồi Người Dùng
  + Thêm thông báo lỗi và hướng dẫn cho người dùng khi dữ liệu