1.1

Trí tuệ nhân tạo (AI) là khả năng thông minh được hiển thị bởi máy tính, mô phỏng hành vi hoặc tư duy con người và có thể được huấn luyện để giải quyết các vấn đề cụ thể. AI là sự kết hợp của các kỹ thuật Học Máy và Học Sâu (Deep Learning). Các loại mô hình Trí Tuệ Nhân Tạo được huấn luyện bằng cách sử dụng lượng lớn dữ liệu và có khả năng đưa ra những quyết định thông minh.

Thống kê và Con số về Trí Tuệ Nhân Tạo:

- Theo Statista, doanh thu từ thị trường phần mềm Trí Tuệ Nhân Tạo (AI) trên toàn thế giới dự kiến sẽ đạt 126 tỷ đô la vào năm 2025.
- Theo Gartner, 37% tổ chức đã triển khai AI dưới mọi hình thức. Tỷ lệ doanh nghiệp sử dụng AI tăng 270% trong vòng bốn năm qua.
- Theo Servion Global Solutions, đến năm 2025, 95% tương tác của khách hàng sẽ được cung cấp bởi AI.
- Một báo cáo gần đây năm 2020 từ Statista tiết lộ rằng thị trường phần mềm AI toàn cầu dự kiến sẽ tăng khoảng 54% hàng năm và dự kiến đạt kích thước dự báo là 22,6 tỷ đô la Mỹ.

Ứng dụng của Trí Tuệ Nhân Tạo (AI):

Trí Tuệ Nhân Tạo (AI) đã trở thành một phần quan trọng và không thể tách rời trong cuộc sống và công việc hàng ngày của chúng ta. Từ những tiến bộ trong học máy và mạng nơ-ron nhân tạo đến khả năng xử lý dữ liệu lớn, AI đã mở ra một loạt các ứng dụng có tiềm năng biến đổi cách chúng ta làm việc, tương tác và sáng tạo. Dưới đây là một số ví dụ về các ứng dụng quan trọng của Trí Tuệ Nhân Tạo:

1. Tư vấn và Hỗ trợ Khách hàng:

Chatbot và hệ thống tự động hóa dựa trên AI được sử dụng rộng rãi để tư vấn và hỗ trợ khách hàng trong nhiều ngành, từ bán lẻ đến dịch vụ tài chính. Chúng có khả năng trả lời câu hỏi, giải quyết vấn đề và cung cấp thông tin 24/7 mà không cần sự can thiệp của con người.

2. Ô tô tự hành:

Công nghệ AI đóng vai trò quan trọng trong phát triển ô tô tự hành. Các hệ thống AI giúp xe ô tô tự nhận diện và đáp ứng đúng với môi trường xung quanh, dự đoán tình huống giao thông và đảm bảo an toàn cho hành khách và người đi đường.

3. Y tế và Chăm sóc sức khỏe:

AI được áp dụng trong chẩn đoán bệnh, dự đoán dịch bệnh và tạo ra kế hoạch điều trị dựa trên dữ liệu lâm sàng và hình ảnh y tế. Hệ thống học máy có khả năng phát hiện triệu chứng sớm của các bệnh nguy hiểm như ung thư và đột quỵ.

4. Tài chính và Giao dịch Tài chính:

AI được sử dụng để dự đoán thị trường tài chính, quản lý rủi ro và phát hiện gian lận trong giao dịch tài chính. Thuật toán AI có khả năng phân tích dữ liệu lớn và dự đoán xu hướng thị trường.

5. Quảng cáo và Tiếp thị:

AI giúp tối ưu hóa chiến dịch quảng cáo trực tuyến, đưa ra gợi ý về nội dung và sản phẩm dựa trên hành vi trực tuyến của người dùng. Hệ thống AI cũng có khả năng phân tích dữ liệu xã hội để đánh giá hiệu quả chiến dịch tiếp thị.

6. Ngôn ngữ tự nhiên và Dịch thuật:

Công nghệ xử lý ngôn ngữ tự nhiên (NLP) trong AI cho phép máy tính hiểu và tạo ra ngôn ngữ như con người. Hệ thống dịch thuật tự động dựa trên AI giúp giao tiếp qua ngôn ngữ trở nên dễ dàng hơn giữa các quốc gia và ngôn ngữ khác nhau.

7. Sản xuất và Quản lý Chuỗi cung ứng:

AI giúp cải thiện hiệu suất sản xuất bằng cách tối ưu hóa lịch trình sản xuất, dự đoán nhu cầu và tối ưu hóa sử dụng tài nguyên. Trong quản lý chuỗi cung ứng, AI giúp dự đoán nhu cầu và lập kế hoạch giao hàng.

8. Nghệ thuật và Sáng tạo:

AI đã tạo ra khả năng mới trong nghệ thuật và sáng tạo. Các tác phẩm nghệ thuật được tạo ra bởi máy tính dựa trên thuật toán AI đã mở ra một thế giới mới trong ngành nghệ thuật số.

9. Công nghệ Xanh và Bảo vệ Môi trường:

AI được sử dụng trong việc quản lý và giám sát nguồn năng lượng, dự đoán biến đổi khí hậu, tối ưu hóa quản lý nước và đẩy mạnh sự phát triển của năng lượng tái tạo.

10. Giáo dục và Học tập trực tuyến:

Công nghệ AI có thể tạo ra trải nghiệm học tập cá nhân hóa, dự đoán cách học tốt nhất cho mỗi học sinh và tạo ra tài liệu học tập tùy chỉnh.

Tổng kết lại, Trí Tuệ Nhân Tạo đã và đang thay đổi cách chúng ta sống, làm việc và tương tác. Từ các ứng dụng trong y tế đến việc cải thiện trải nghiệm khách hàng, AI đang phát triển và mở ra nhiều cơ hội mới trong tương lai. Tuy nhiên, cũng cần xem xét

cẩn thận về các vấn đề liên quan đến quyền riêng tư, đạo đức và tác động xã hội khi triển khai AI.

1.2

Hệ thống thông minh (Intelligent System) là một loại hệ thống máy tính hoặc phần mềm được thiết kế để thực hiện các nhiệm vụ hoặc công việc mà trước đây thường chỉ có con người có khả năng thực hiện. Các hệ thống thông minh được phát triển để mô phỏng hoặc mô hình hóa khả năng trí tuệ của con người trong việc ra quyết định, giải quyết vấn đề và tương tác với môi trường xung quanh mình.

Các hệ thống thông minh đã tồn tại từ lâu và được áp dụng trong nhiều lĩnh vực khác nhau. Chúng thường thuộc phạm vi rộng hơn của Trí Tuệ Nhân Tạo (AI). Các hệ thống thông minh giải quyết các vấn đề phức tạp một cách tự động và hiệu quả hơn trong các môi trường cụ thể. Chúng được hình thành bởi sự hợp tác giữa con người và các công nghệ như Big Data, IoT (Internet of things), mạng di động (3G, 4G, 5G), trí tuệ nhân tạo (AI), robot, phân tích video, thị giác máy tính và thực tế tăng cường, và nhiều công nghệ khác.

1. IGI Global - "Intelligent System" (Link 1)

Định nghĩa: "Hệ thống thông minh là một hệ thống có khả năng tự động hóa thông qua việc sử dụng trí tuệ nhân tạo, khả năng học máy, hoặc các phương pháp khác để thực hiện các nhiệm vụ phức tạp và quyết định thông minh."

2. High-Tech Guide - "What Are Examples of Intelligent Systems?" (Link 2)

Ví dụ: Trong bài viết này, một số ví dụ về hệ thống thông minh bao gồm xe tự hành (self-driving cars), hệ thống trí tuệ nhân tạo dùng trong chẩn đoán bệnh, và hệ thống dự đoán thời tiết sử dụng trí tuệ nhân tạo.

3. IoT For All - "8 Helpful Everyday Examples of Artificial Intelligence" (Link 3)

Ví dụ: Bài viết này liệt kê 8 ví dụ thực tế về trí tuệ nhân tạo, bao gồm ứng dụng hỗ trợ giọng nói (voice assistants) như Amazon Alexa và Google Assistant, hệ thống phát hiện gian lận trong giao dịch tài chính, và hệ thống dự đoán nhu cầu người tiêu dùng dựa trên lịch sử mua sắm.

4. Algotive - "Intelligent Systems: What Are They, How Do They Work, and Why Are They So Important?" (Link 4)

Bài viết này giải thích về hệ thống thông minh và giới thiệu cách chúng hoạt động. Nó cũng đề cập đến sự quan trọng của hệ thống thông minh trong nhiều lĩnh vực khác nhau, từ công nghiệp đến y tế và giao thông.

Đặc điểm quan trọng của hệ thống thông minh bao gồm:

- Khả Năng Học Hỏi (Learning): Hệ thống thông minh có khả năng học hỏi từ dữ liệu và kinh nghiệm. Chúng có thể cải thiện hiệu suất và thích nghi với môi trường bằng cách tự động điều chỉnh và cải thiện mô hình hoạt động.
- Tư Duy Như Con Người: Hệ thống thông minh có khả năng tư duy, nghĩa là chúng có thể xử lý thông tin, tạo ra giải pháp cho các vấn đề phức tạp, và thậm chí có thể đưa ra quyết định dựa trên dữ liệu và kiến thức.
- Tương Tác Ngôn Ngữ Tự Nhiên (Natural Language Interaction): Một số hệ thống thông minh có khả năng tương tác với con người qua ngôn ngữ tự nhiên, chẳng hạn như chatbot hoặc hệ thống trả lời câu hỏi dựa trên giọng nói.
- Phân Tích Dữ Liệu và Trích Xuất Thông Tin: Hệ thống thông minh có khả năng phân tích dữ liệu lớn và trích xuất thông tin quan trọng từ dữ liệu không cấu trúc hoặc cấu trúc.
- Tự Động Hóa Quyết Định và Hành Động: Chúng có thể tự động ra quyết định và thực hiện các hành động mà cần thiết để đạt được mục tiêu cụ thể.

Ví dụ về hệ thống thông minh:

Một ví dụ cụ thể về hệ thống thông minh là "hệ thống giao thông thông minh" (Intelligent Transportation System - ITS). Hệ thống này kết hợp nhiều công nghệ để quản lý và tối ưu hóa giao thông trong các đô thị và các tuyến đường. Dưới đây là một số thành phần và ứng dụng cụ thể của hệ thống giao thông thông minh:

1. Giao thông thông minh: Các cảm biến và hệ thống giám sát theo dõi lưu lượng giao thông, tình trạng đường, và tốc độ di chuyển của phương tiện.

2. Điều khiển tín hiệu giao thông: Hệ thống điều khiển tín hiệu giao thông tự động điều chỉnh tín hiệu dựa trên thông tin thời gian thực về lưu lượng giao thông để giảm ùn tắc và cải thiện luồng giao thông.

3. Hướng dẫn thông tin: Các bảng hiển thị tại các điểm trọng yếu cung cấp thông tin về tình hình giao thông, lưu lượng, tuyến đường thay thế và thời gian dự kiến đến đích.

4. Hệ thống phát hiện tai nạn: Các cảm biến và camera giúp phát hiện tai nạn và sự cố trên đường, thông báo đến người quản lý giao thông và cơ quan cứu hộ.

5. Điều phối giao thông thông minh: Hệ thống sử dụng dữ liệu về tình hình giao thông để đề xuất các tuyến đường tối ưu cho phương tiện, giúp giảm thời gian di chuyển và nồng độ ùn tắc.

6. Phân tích dữ liệu: Dữ liệu từ các cảm biến và hệ thống giám sát được phân tích để dự đoán tình hình giao thông, định vị điểm nút giao thông có nguy cơ ùn tắc và đưa ra biện pháp phòng ngừa.

7. Ứng dụng di động thông minh: Các ứng dụng di động cung cấp thông tin giao thông thời gian thực, tuyến đường tối ưu, và thông báo về các tình huống không mong muốn.

Hệ thống giao thông thông minh giúp cải thiện hiệu suất giao thông, giảm ùn tắc, tăng an toàn và tiết kiệm thời gian cho người tham gia giao thông. Đây chỉ là một ví dụ trong số nhiều ứng dụng của hệ thống thông minh trong cuộc sống hàng ngày.

1.3

Hệ thống thông minh (AI - Artificial Intelligence) đã và đang thay đổi cách chúng ta làm việc và tương tác với thế giới xung quanh. Các ứng dụng của AI đã lan rộng sang nhiều lĩnh vực khác nhau và sử dụng nhiều kỹ thuật khác nhau để đáp ứng nhu cầu đa dạng của xã hội và công nghiệp.

1. Lĩnh Vực Sử Dụng Của Hệ Thống Thông Minh

a. Y Tế và Dinh Dưỡng

- Chuẩn Đoán Bệnh: AI được sử dụng để phân tích hình ảnh y tế như chụp X-quang và MRI để phát hiện các bệnh lý, giúp bác sĩ xác định chuẩn đoán một cách nhanh chóng và chính xác.
- Quản Lý Dinh Dưỡng: Ứng dụng thông minh giúp người dùng theo dõi và quản lý chế độ ăn uống, cung cấp lời khuyên về dinh dưỡng dựa trên dữ liệu cá nhân và mục tiêu sức khỏe.

b. Ô Tô Tự Hành

- Xe Tự Lái: Hệ thống thông minh trên xe ô tô tự hành sử dụng các cảm biến và học máy để tự động lái xe, giúp giảm tai nạn giao thông và tạo ra giao thông thông thoáng hơn.
- Dịch Vụ Giao Hàng Tự Động: Các dịch vụ giao hàng dựa trên xe tự hành sử dụng AI để tối ưu hóa tuyến đường và quản lý việc giao hàng.

c. Tài Chính

- Giao Dịch Tài Chính Tự Động: Các thuật toán AI được sử dụng để dự đoán và thực hiện giao dịch tài chính một cách nhanh chóng và tự động.
- Quản Lý Rủi Ro: AI được sử dụng để phân tích dữ liệu tài chính và đưa ra các dự đoán về rủi ro đầu tư.

d. Giáo Dục

- Học Trực Tuyến: AI được sử dụng trong các nền tảng học trực tuyến để cá nhân hóa quá trình học tập cho từng học sinh và đề xuất nội dung học tập phù hợp.
- Tự Động Chấm Bài: AI có thể tự động chấm bài thi trắc nghiệm và cung cấp kết quả ngay sau khi hoàn thành.

e. Công Nghiệp

Tự Động Hóa Nhà Máy: AI được sử dụng để điều khiển và tối ưu hóa các quy trình sản xuất trong nhà máy, giúp tiết kiệm thời gian và nguồn lực.

Dự Đoán Bảo Trì: AI có thể dự đoán khi nào máy móc cần bảo trì để tránh sự cố và giảm thiểu thời gian dừng máy.

2. Các Kỹ Thuật Trong AI

a. Học Máy (Machine Learning)

- Học Giám Sát: Dựa vào dữ liệu huấn luyện có nhãn, học máy giúp xây dựng mô hình dự đoán và phân loại, chẳng hạn như học máy hồi quy và cây quyết định.
- Học Không Giám Sát: Học máy cũng có khả năng tự động phát hiện mẫu trong dữ liệu mà không cần dữ liệu huấn luyện, chẳng hạn như gom cụm và phân loại không giám sát.

b. Deep Learning

- Mạng Nơ-ron Nhân Tạo (Artificial Neural Networks - ANN): Deep learning sử dụng các mô hình mạng nơ-ron với nhiều tầng ẩn để học các biểu diễn phức tạp từ dữ liệu, chẳng hạn như mạng nơ-ron tích chập (CNN) cho xử lý hình ảnh và mạng nơ-ron hồi quy (RNN) cho xử lý chuỗi thời gian.

c. Xử Lý Ngôn Ngữ Tự Nhiên (Natural Language Processing - NLP)

- Phân Tích Tự Động Văn Bản: NLP giúp máy tính hiểu và xử lý ngôn ngữ tự nhiên, dựa trên đó có thể thực hiện các nhiệm vụ như dịch máy, tóm tắt văn bản, và phân loại tin tức.

d. Thị Giác Máy Tính (Computer Vision)

- Nhận Dạng Hình Ảnh: AI sử dụng thị giác máy tính để nhận dạng và phân loại đối tượng trong hình ảnh, chẳng hạn như trong ứng dụng xe tự hành và nhận dạng khuôn mặt.

e. Tối Ưu Hóa

- Tối Ưu Hóa Mô Hình: Các kỹ thuật tối ưu hóa giúp cải thiện hiệu suất của mô hình AI, tối ưu hóa hàm mục tiêu và tinh chỉnh các tham số.

f. Học Tăng Cường (Reinforcement Learning)

- Học Từ Phản Hồi: Học tăng cường sử dụng hệ thống thích nghi với môi trường thông qua phản hồi để đạt được mục tiêu, chẳng hạn như trong việc huấn luyện các hệ thống điều khiển tự động.

g. Tự động hóa và IoT (Internet of Things):

- Hệ thống thông minh có thể tích hợp với các thiết bị IoT để quản lý và kiểm soát thông tin từ môi trường thực tế. Ví dụ, trong nhà thông minh, hệ thống thông minh có thể điều khiển ánh sáng, nhiệt độ, bảo mật, và nhiều yếu tố khác dựa trên dữ liệu từ các thiết bị IoT.

h. Quản lý Rủi ro và An ninh:

- Trong ngành bảo hiểm, hệ thống thông minh có thể được sử dụng để đánh giá rủi ro và đưa ra quyết định về việc đề xuất chính sách bảo hiểm. Hệ thống cũng có thể giúp kiểm soát an ninh thông tin bằng cách phát hiện và ngăn chặn các hành vi đe dọa.

i. Quản lý Năng lượng:

- Trong lĩnh vực năng lượng, hệ thống thông minh có thể được sử dụng để tối ưu hóa sử dụng năng lượng trong các hệ thống đô thị và công nghiệp. Chúng có thể dự đoán tiêu thụ năng lượng và điều chỉnh sản xuất năng lượng tái tạo để đáp ứng nhu cầu.

j. Hệ thống Tự động Hoá Công nghiệp (Industrial Automation):

- Trong sản xuất và công nghiệp, hệ thống thông minh được sử dụng để tự động hóa quy trình sản xuất và kiểm soát máy móc. Chúng có thể tối ưu hóa lịch trình sản xuất, dự đoán hỏng hóc máy móc, và giảm thiểu lỗi nhân viên.

k. Giáo dục:

- Hệ thống thông minh có thể cung cấp hỗ trợ giảng dạy và học tập dựa trên dữ liệu về tiến bộ học tập của học sinh. Chúng có thể tạo ra nội dung giảng dạy tùy chỉnh và đề xuất tài liệu học tập dựa trên nhu cầu cá nhân.

Hệ thống thông minh đã xâm nhập vào hầu hết mọi khía cạnh của cuộc sống và công nghiệp. Chúng sử dụng một loạt các kỹ thuật AI, từ học máy đến thị giác máy tính và xử lý ngôn ngữ tự nhiên, để tạo ra các ứng dụng thông minh và tự động hóa quá trình ra

quyết định. Sự kết hợp giữa khả năng tính toán mạnh mẽ và sáng tạo của AI đã thúc đẩy cuộc cách mạng công nghiệp và cuộc sống của chúng ta.

1.4

Trí tuệ nhân tạo có thể được chia thành các loại dựa trên khả năng và chức năng.

a) Có ba loại trí tuệ nhân tạo dựa trên khả năng:

1. Trí Tuệ Nhân Tạo Hạn Hẹp (Artificial Narrow Intelligence - ANI):

Còn được gọi là Trí Tuệ Yếu, ANI là giai đoạn của Trí Tuệ Nhân Tạo liên quan đến các máy có thể thực hiện chỉ một tập hợp hẹp và cụ thể các nhiệm vụ. Ở giai đoạn này, máy không có khả năng tư duy, chỉ thực hiện một tập hợp các chức năng đã được định nghĩa trước.

Các ví dụ về Trí Tuệ Yếu bao gồm Siri, Alexa, xe tự lái, Alpha-Go, con người giả tưởng Sophia và nhiều hệ thống dựa trên AI khác. Hầu hết tất cả các hệ thống dựa trên AI được xây dựng cho đến thời điểm này thuộc loại Trí Tuệ Yếu.

2. Trí Tuệ Nhân Tạo Tổng Quát (Artificial General Intelligence - AGI):

Còn được gọi là Trí Tuệ Mạnh, AGI là giai đoạn trong sự tiến hóa của Trí Tuệ Nhân Tạo trong đó máy sẽ có khả năng tư duy và ra quyết định giống như con người.

Hiện tại không có ví dụ cụ thể về Trí Tuệ Mạnh, tuy nhiên, tin rằng chúng ta sẽ sớm có thể tạo ra những máy thông minh như con người.

Trí Tuệ Mạnh được xem là mối đe dọa đối với sự tồn tại của con người bởi nhiều nhà khoa học, bao gồm cả Stephen Hawking, người tuyên bố:

"Việc phát triển Trí Tuệ Nhân Tạo đầy đủ có thể đánh dấu sự kết thúc của loài người... Nó sẽ phát triển một cách độc lập và tự chế biến với tốc độ ngày càng tăng. Con người, bị giới hạn bởi tiến hóa sinh học chậm rãi, không thể cạnh tranh và sẽ bị thay thế."

3. Trí Tuệ Nhân Tạo Siêu Cấp (Artificial Super Intelligence – ASI):

Trí Tuệ Nhân Tạo Siêu Cấp là giai đoạn trong Trí Tuệ Nhân Tạo khi khả năng của máy tính sẽ vượt qua con người. ASI hiện đang là tình huống giả thuyết như được mô tả trong các bộ phim và sách khoa học viễn tưởng, nơi máy tính đã chiếm đoạt thế giới.

Một số tin tưởng rằng máy tính không còn xa để đạt được giai đoạn này, dựa trên tốc độ phát triển hiện tại của chúng.

"Tiến trình phát triển trí tuệ nhân tạo đầy đủ (tôi không đề cập đến Trí Tuệ Hạn Hẹp) cực kỳ nhanh chóng. Trừ khi bạn tiếp xúc trực tiếp với các nhóm như Deepmind, bạn sẽ

không biết tốc độ này - nó đang phát triển với tốc độ gần như là hàm mũ. Nguy cơ xảy ra điều gì đó nguy hiểm thực sự trong khoảng năm tới. Nhiều nhất là 10 năm." - Elon Musk đã trích dẫn.

b) Dựa trên chức năng, chúng ta có bốn loại trí tuệ nhân tạo:

1. Hệ thống Reactive Machines AI (Hệ thống AI Hoạt Động):

Loại AI này bao gồm các máy hoạt động dựa trên dữ liệu hiện tại, chỉ xem xét tình hình hiện tại. Các máy AI hoạt động không thể suy luận từ dữ liệu để đánh giá hành động tương lai của chúng. Chúng có thể thực hiện một phạm vi hẹp của các nhiệm vụ đã được định nghĩa trước.

Một ví dụ về Reactive AI là chương trình cờ vua nổi tiếng của IBM đã đánh bại nhà vô địch thế giới, Garry Kasparov.

2. Hệ thống Limited Memory AI (Hệ thống AI Bộ Nhớ Hạn Chế):

Như tên gợi ý, AI Bộ Nhớ Hạn Chế có thể ra quyết định thông minh và cải thiện bằng cách nghiên cứu dữ liệu quá khứ từ bộ nhớ của nó. Loại AI này có một bộ nhớ ngắn hạn hoặc tạm thời có thể được sử dụng để lưu trữ các trải nghiệm quá khứ và do đó đánh giá hành động tương lai.

Các xe tự lái là AI Bộ Nhớ Hạn Chế, sử dụng dữ liệu thu thập trong quá khứ gần để đưa ra quyết định ngay lập tức. Ví dụ, các xe tự lái sử dụng cảm biến để nhận biết dân thường băng qua đường, đường dốc, tín hiệu giao thông và vân vân để đưa ra quyết định lái xe tốt hơn. Điều này giúp ngăn ngừa các tai nạn trong tương lai.

3. Hệ thống Theory Of Mind AI (Hệ thống AI Lý Thuyết Về Tâm):

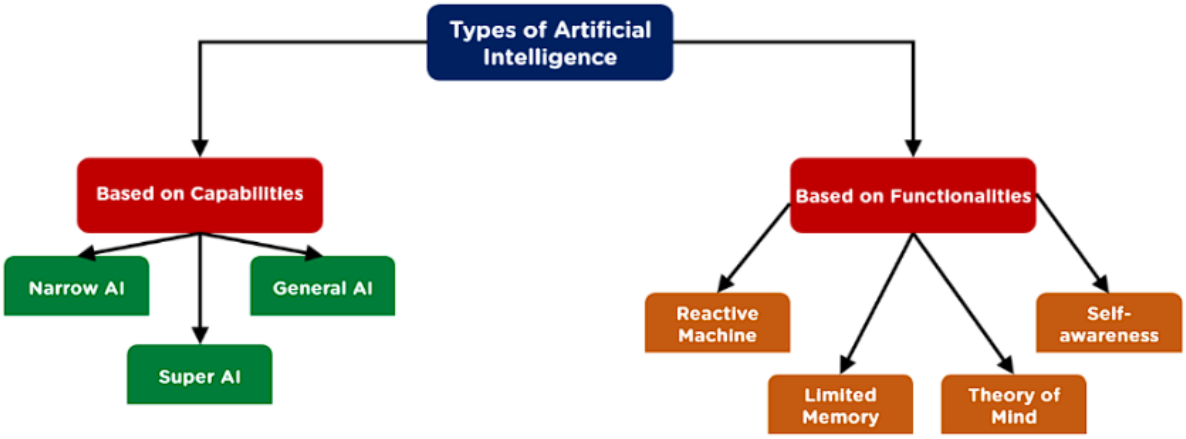
Loại Trí Tuệ Nhân Tạo này là một dạng tiến xa hơn. Loại máy này được cho là sẽ đóng một vai trò quan trọng trong tâm lý học. Loại AI này sẽ tập trung chủ yếu vào trí tuệ cảm xúc để có thể hiểu rõ hơn về niềm tin và suy nghĩ của con người.

Loại Theory Of Mind AI này chưa được phát triển hoàn toàn, nhưng nghiên cứu mạnh mẽ đang diễn ra trong lĩnh vực này.

4. Hệ thống Self-Aware AI (Hệ thống AI Tự Nhận Thức):

Chúng ta chỉ cầu nguyện rằng chúng ta không đạt được trạng thái của AI, trong đó máy tính có ý thức riêng và trở nên tự nhận thức. Loại Trí Tuệ Nhân Tạo này còn hơi xa vời dựa trên tình hình hiện tại. Tuy nhiên, trong tương lai, việc đạt được một giai đoạn siêu trí tuệ có thể là khả thi.

Những loại trí tuệ nhân tạo này đa dạng về khả năng và ứng dụng, và chúng đang tiếp tục phát triển để đáp ứng nhu cầu của thế giới hiện đại.



1.5

Các ứng dụng của Hệ thống thông minh qua figture 7:

- Intelligent Autonomous System: Đây là loại hệ thống thông minh có khả năng hoạt động tự động và thông minh trong môi trường thực tế hoặc ảo, có khả năng tương tác và đưa ra quyết định mà không cần sự can thiệp của con người.
- Autonomous in a Real Environment: Đây là loại hệ thống thông minh có khả năng hoạt động tự động trong môi trường thực tế mà không cần sự can thiệp của con người. Dưới đây là một số ví dụ cụ thể về các ứng dụng của hệ thống này:
  - Robotic System: Một robot dược sử dụng trong môi trường công nghiệp để thực hiện các nhiệm vụ sản xuất tự động mà không cần sự can thiệp của con người.
    - Autonomous Robot: Các robot tự động có thể thực hiện các tác vụ trong môi trường thực tế mà không cần sự can thiệp của con người. Ví dụ, robot hút bụi thông minh có khả năng tự động dọn dẹp sàn nhà mà không cần người điều khiển.
    - Collaborative Robot: Các cobot (collaborative robot) là những người bạn làm việc cùng với con người trong môi trường sản xuất. Chúng có thể hỗ trợ trong việc lắp ráp, kiểm tra chất lượng, hoặc vận chuyển hàng hóa.
    - Self-Driving Vehicle: Một phương tiện di chuyển tự động mà không cần người lái. Đây thường là xe ô tô hoặc xe tải có khả năng tự định hướng, đo đạc môi trường và thực hiện các tác vụ lái xe mà không cần sự can thiệp của người lái.

- Autonomous Automation System: Hệ thống tự động hóa tự động có khả năng thực hiện các nhiệm vụ trong môi trường thực tế mà không cần sự can thiệp của con người. Dưới đây là một số ví dụ cụ thể về các ứng dụng của hệ thống này:
  - Home Automation: Hệ thống tự động hóa trong gia đình có thể điều khiển ánh sáng, nhiệt độ, an ninh, và thiết bị gia đình khác dựa trên lịch trình hoặc yêu cầu của người dùng. Ví dụ, bạn có thể điều khiển đèn, máy lạnh, và cửa ra vào thông qua điện thoại di động.
  - Intelligent Industrial Automation: Trong môi trường sản xuất công nghiệp, hệ thống tự động hóa thông minh có thể quản lý quy trình sản xuất, theo dõi thiết bị và dự đoán hỏng hóc, từ đó tối ưu hóa hiệu suất và tăng năng suất.

- Autonomous in a Virtual Environment: Đây là loại hệ thống thông minh có khả năng hoạt động tự động trong môi trường ảo hoặc giả lập mà không cần sự can thiệp của con người
  - Autonomous with Restricted User Interaction: Đây là loại hệ thống thông minh có khả năng tương tác với người dùng trong môi trường ảo hoặc giả lập, nhưng chỉ có thể thực hiện một số tương tác cố định. Dưới đây là một số ví dụ cụ thể về các ứng dụng của hệ thống này:
    - Software Robot: Các ứng dụng phần mềm tự động có thể thực hiện các tác vụ như xử lý dữ liệu, gửi thông báo tự động, hoặc tạo báo cáo dựa trên quy tắc đã được cài đặt trước.
    - Virtual Character: Nhân vật ảo có khả năng tương tác với người dùng trong môi trường ảo, ví dụ: trong trò chơi hoặc trong môi trường giả lập đào tạo.
    - Intelligent Tutoring System: Hệ thống hỗ trợ giảng dạy thông minh có thể cung cấp hướng dẫn tương tác và cá nhân hóa cho người học.
  - Autonomous with Open Natural Language: Đây là loại hệ thống thông minh có khả năng hiểu và đáp ứng các yêu cầu từ người dùng bằng ngôn ngữ tự nhiên. Dưới đây là một số ví dụ cụ thể về các ứng dụng của hệ thống này:
    - Personal Assistant: Trợ lý cá nhân có khả năng tương tác qua ngôn ngữ tự nhiên, giúp người dùng quản lý lịch trình, tạo hẹn, tra cứu thông tin, và thực hiện các tác vụ khác dựa trên yêu cầu của họ.
    - Transactional Chatbot: Trò chuyện tự động có khả năng thực hiện các giao dịch cơ bản như đặt hàng, đặt lịch hẹn, hoặc thanh toán dựa trên tương tác qua ngôn ngữ tự nhiên.

➢ Intelligent Advisor System: Đây là loại hệ thống thông minh có khả năng cung cấp lời khuyên, hướng dẫn và thông tin dựa trên thông tin và ngữ cảnh mà không cần sự can thiệp của con người.

- Advisor with Open Natural Language: Đây là loại hệ thống thông minh có khả năng hiểu và trả lời các yêu cầu của người dùng bằng ngôn ngữ tự nhiên. Dưới đây là một số ví dụ cụ thể về các ứng dụng của hệ thống này:
  - Question Answering System: Hệ thống trả lời câu hỏi có khả năng hiểu và đáp ứng các yêu cầu thông qua ngôn ngữ tự nhiên. Ví dụ, hệ thống này có thể trả lời các câu hỏi về lịch sử, khoa học, văn hóa, và nhiều lĩnh vực khác.
- Advisor using Data Measured from Environment: Đây là loại hệ thống thông minh có khả năng cung cấp lời khuyên và thông tin dựa trên dữ liệu được thu thập từ môi trường hoặc ngữ cảnh. Dưới đây là một số ví dụ cụ thể về các ứng dụng của hệ thống này:
  - Management Support System: Hệ thống hỗ trợ quản lý sử dụng dữ liệu từ môi trường kinh doanh để cung cấp thông tin và phân tích hỗ trợ quyết định cho quản lý. Ví dụ, hệ thống này có thể phân tích dữ liệu doanh số bán hàng để đưa ra chiến lược kinh doanh.
  - Recommender System: Hệ thống đề xuất sản phẩm, dịch vụ, hoặc nội dung dựa trên dữ liệu về sở thích và hành vi của người dùng. Ví dụ, dựa trên lịch sử mua sắm, hệ thống có thể đề xuất các sản phẩm liên quan hoặc phù hợp với sở thích của người dùng.
- Advisor with Restricted Interaction: Đây là loại hệ thống thông minh có khả năng cung cấp lời khuyên hoặc thông tin theo yêu cầu cụ thể của người dùng, nhưng có giới hạn tương tác và phản hồi. Dưới đây là một số ví dụ cụ thể về các ứng dụng của hệ thống này:
  - First Generation Expert System: Hệ thống thông minh sử dụng luật logic và kiến thức đã lưu trữ trước đó để đưa ra lời khuyên hoặc quyết định trong một lĩnh vực cụ thể. Ví dụ, một hệ thống chẩn đoán bệnh dựa trên tri thức y học có thể đưa ra lời khuyên về bệnh dựa trên triệu chứng mà bệnh nhân cung cấp.

1.6

1. Numpy:

NumPy (Numerical Python) là một thư viện quan trọng trong ngôn ngữ lập trình Python dành cho tính toán khoa học và toán học. NumPy cung cấp một cấu trúc dữ liệu mảng nhiều chiều (ndarray) mạnh mẽ và các công cụ để thực hiện các phép toán số học và logic trên mảng này.

Đặc trưng của Numpy:

- Mảng nhiều chiều (ndarray): NumPy cung cấp cấu trúc dữ liệu mảng nhiều chiều (ndarray) cho phép lưu trữ và xử lý dữ liệu trong các chiều khác nhau.

- Phép toán toán học: NumPy hỗ trợ phép toán số học như cộng, trừ, nhân, chia trên mảng, cũng như các phép toán khác như lũy thừa, căn bậc hai, logarit.
- Indexing và Slicing: NumPy cho phép truy cập và thay đổi các phần tử trong mảng thông qua cú pháp indexing và slicing.
- Phép toán logic và so sánh: NumPy hỗ trợ phép toán logic như AND, OR, NOT trên các mảng boolean, cũng như phép so sánh giá trị giữa các phần tử mảng.
- Broadcasting: NumPy cho phép thực hiện các phép toán trên các mảng có hình dáng khác nhau mà không cần phải thực hiện việc mở rộng kích thước mảng thủ công.

Mục đích của Numpy: NumPy được sử dụng rộng rãi trong tính toán khoa học, xử lý dữ liệu, phân tích dữ liệu và nhiều lĩnh vực khác. Dưới đây là một số mục đích chính của NumPy:

- Tính toán khoa học: NumPy cung cấp các công cụ để thực hiện tính toán phức tạp trong các lĩnh vực như toán học, vật lý, hóa học và kỹ thuật.
- Xử lý dữ liệu: NumPy giúp xử lý dữ liệu số lượng lớn dễ dàng hơn, bao gồm đọc dữ liệu từ các nguồn khác nhau và thực hiện các phép toán trên dữ liệu này.
- Trực quan hóa dữ liệu: NumPy có thể cộng tác với các thư viện khác như Matplotlib để tạo biểu đồ và đồ thị trực quan hóa dữ liệu.

Ví dụ:

```python
import numpy as np

# Tạo mảng numpy
arr = np.array([1, 2, 3, 4, 5])

# Thực hiện phép toán số học trên mảng
result = arr * 2

# Tính trung bình cộng của mảng
mean = np.mean(arr)

# Tạo mảng 2 chiều
matrix = np.array([[1, 2, 3], [4, 5, 6]])

# Tích hai ma trận
product = np.dot(matrix, matrix.T)
```

2. Pandas:

Pandas là một thư viện quan trọng trong ngôn ngữ lập trình Python, được sử dụng rộng rãi cho xử lý và phân tích dữ liệu có cấu trúc. Pandas cung cấp các cấu trúc dữ liệu mạnh mẽ như Series và DataFrame để làm việc với dữ liệu dưới dạng bảng và chuỗi thời gian.

Đặc trưng của pandas:

- DataFrame và Series: pandas cung cấp hai cấu trúc chính là DataFrame (dùng cho dữ liệu dạng bảng) và Series (dùng cho chuỗi dữ liệu một chiều) để quản lý và xử lý dữ liệu.
- Xử lý dữ liệu: pandas cho phép tải dữ liệu từ nhiều nguồn khác nhau, xử lý dữ liệu bị thiếu, thực hiện các phép toán trên dữ liệu và thực hiện các biến đổi dữ liệu.
- Indexing và Slicing: pandas hỗ trợ các phương pháp indexing và slicing để truy cập và thay đổi dữ liệu trong DataFrame và Series.
- Grouping và Aggregation: pandas cung cấp chức năng groupby để nhóm dữ liệu theo các thuộc tính và thực hiện các phép tính tổng hợp trên các nhóm này.
- Thao tác với dữ liệu chuỗi thời gian: pandas hỗ trợ xử lý và phân tích dữ liệu chuỗi thời gian một cách hiệu quả.

Mục đích của pandas: pandas được sử dụng rộng rãi trong việc xử lý dữ liệu, phân tích dữ liệu và chuỗi thời gian.

- Xử lý dữ liệu: pandas giúp tải, lưu trữ và xử lý dữ liệu dễ dàng hơn, đồng thời cung cấp các công cụ để xử lý các vấn đề phát sinh từ dữ liệu bị thiếu hoặc không đồng nhất.
- Phân tích dữ liệu: pandas cho phép thực hiện các phân tích thống kê, tính toán tổng hợp và vẽ biểu đồ để hiểu rõ hơn về dữ liệu.
- Xử lý dữ liệu chuỗi thời gian: pandas hỗ trợ xử lý và phân tích dữ liệu chuỗi thời gian một cách thuận tiện, bao gồm tính toán theo thời gian và vẽ biểu đồ chuỗi thời gian.

Ví dụ:

```python
import pandas as pd

# Đọc dữ liệu từ file CSV
data = pd.read_csv('data.csv')

# Hiển thị 5 dòng đầu tiên của DataFrame
print(data.head())

# Lọc dữ liệu theo điều kiện
filtered_data = data[data['age'] > 25]

# Tính tổng các giá trị trong cột
total = data['salary'].sum()

# Tạo DataFrame mới từ dict
new_data = pd.DataFrame({'name': ['Alice', 'Bob'], 'age': [25, 30]})
```

3. Matplotlib:

Matplotlib là một thư viện trực quan hóa dữ liệu mạnh mẽ và linh hoạt trong ngôn ngữ lập trình Python. Matplotlib cung cấp các công cụ để tạo ra biểu đồ, đồ thị và hình ảnh trực quan hóa dữ liệu.

Đặc trưng của Matplotlib:

- Loạt biểu đồ: Matplotlib hỗ trợ nhiều loại biểu đồ như đường, cột, hình tròn, đám mây điểm và nhiều loại khác để hiển thị dữ liệu theo cách trực quan.
- Tùy chỉnh: Matplotlib cho phép tùy chỉnh nhiều khía cạnh của biểu đồ như màu sắc, định dạng văn bản, phông chữ và vị trí của các thành phần trên biểu đồ.
- Lưu trữ hình ảnh: Matplotlib cho phép lưu trữ hình ảnh được tạo ra dưới nhiều định dạng như PNG, JPG, PDF để sử dụng sau này hoặc chia sẻ.

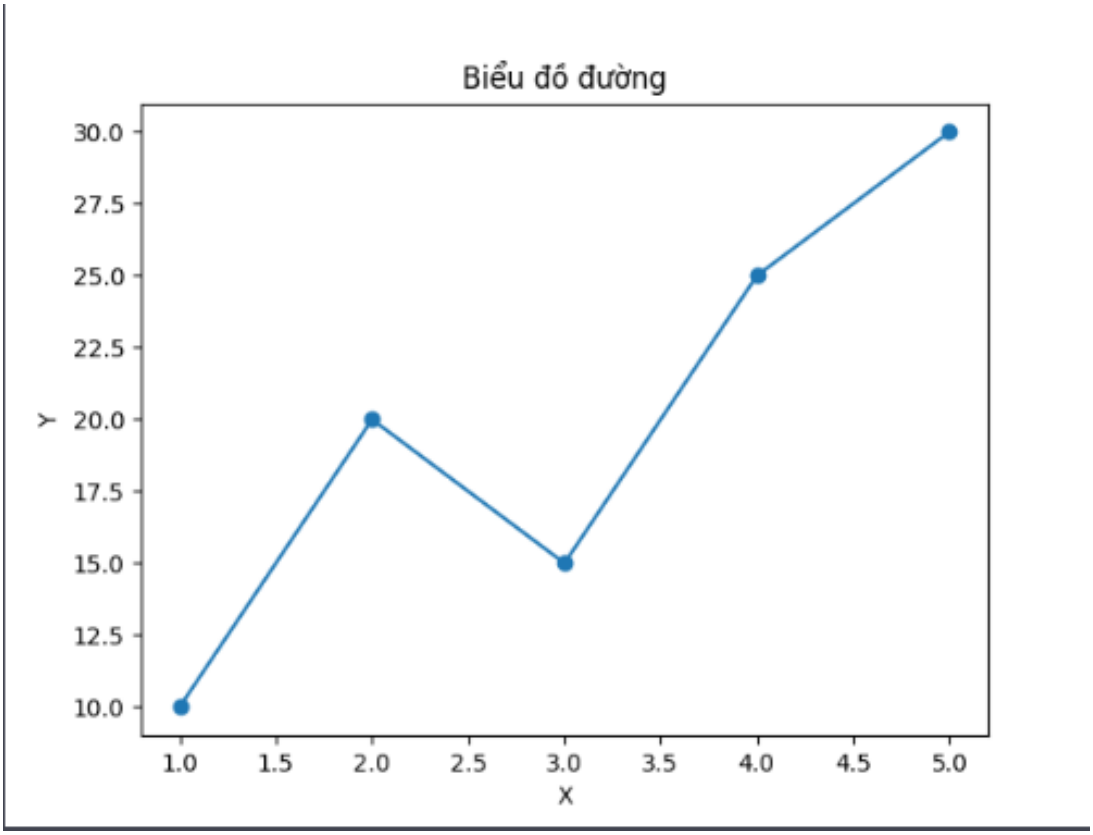
Mục đích của Matplotlib: Matplotlib được sử dụng rộng rãi để trực quan hóa dữ liệu và kết quả phân tích một cách trực quan và hiểu rõ hơn.

- Hiển thị dữ liệu: Matplotlib giúp hiển thị dữ liệu dưới dạng biểu đồ hoặc đồ thị để giúp hiểu rõ hơn về mẫu, xu hướng và tương quan trong dữ liệu.

- Trực quan hóa phân tích: Matplotlib cho phép trực quan hóa kết quả phân tích dưới dạng biểu đồ, giúp làm cho thông tin trở nên dễ hiểu và trình bày một cách thể hiện.
- Trình bày báo cáo: Matplotlib giúp tạo ra các hình ảnh trực quan hóa để chèn vào báo cáo, bài thuyết trình hoặc tài liệu học tập.

Ví dụ:

```
import matplotlib.pyplot as plt

# Dữ liệu
x = [1, 2, 3, 4, 5]
y = [10, 20, 15, 25, 30]

# Vẽ biểu đồ đường
plt.plot(x, y, marker='o')

# Đặt tiêu đề và nhãn trục
plt.title('Biểu đồ đường')
plt.xlabel('X')
plt.ylabel('Y')

# Hiển thị biểu đồ
plt.show()
```



4. Scikit-Learn:

Scikit-Learn là một thư viện mạnh mẽ cho học máy và khai phá dữ liệu trong ngôn ngữ lập trình Python. Sklearn cung cấp các công cụ cho việc xây dựng, đào tạo và đánh giá các mô hình học máy khác nhau.

Đặc trưng của scikit-learn:

- Các thuật toán học máy: Sklearn bao gồm nhiều thuật toán học máy như hồi quy tuyến tính, phân loại, gom cụm, phân tích thành phần chính và nhiều thuật toán khác.
- Tiền xử lý dữ liệu: Sklearn cung cấp các công cụ cho việc tiền xử lý dữ liệu như chuẩn hóa, mã hóa và xử lý dữ liệu bị thiếu.
- Tối ưu hóa mô hình: Sklearn giúp tối ưu hóa các tham số mô hình để đạt được hiệu suất tốt nhất trên dữ liệu.
- Chẩn đoán mô hình: Sklearn cung cấp các công cụ cho việc đánh giá hiệu suất mô hình qua việc tính toán các độ đo như chính xác, F1-score, RMSE, và nhiều độ đo khác.

Mục đích của scikit-learn: sử dụng để xây dựng và đào tạo các mô hình học máy trên dữ liệu thực tế và giúp thực hiện các tác vụ khác nhau như phân loại, dự đoán và gom cụm.

- Xây dựng mô hình học máy: Sklearn cho phép bạn xây dựng các mô hình học máy từ các thuật toán tiêu chuẩn.
- Phân loại và dự đoán: Sklearn có thể được sử dụng để phân loại các mẫu vào các lớp khác nhau và dự đoán kết quả trên dữ liệu mới.
- Gom cụm: Sklearn hỗ trợ gom cụm dữ liệu thành các nhóm dựa trên các đặc trưng chung.

Ví dụ:

```python
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error

# Dữ liệu huấn luyện
X = [[1], [2], [3], [4], [5]]
y = [2, 4, 6, 8, 10]

# Chia dữ liệu thành tập huấn luyện và kiểm tra
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Tạo mô hình học máy
model = LinearRegression()

# Huấn luyện mô hình trên dữ liệu huấn luyện
model.fit(X_train, y_train)

# Dự đoán kết quả trên tập kiểm tra
y_pred = model.predict(X_test)

# Đánh giá hiệu suất của mô hình
mse = mean_squared_error(y_test, y_pred)
```

1.7

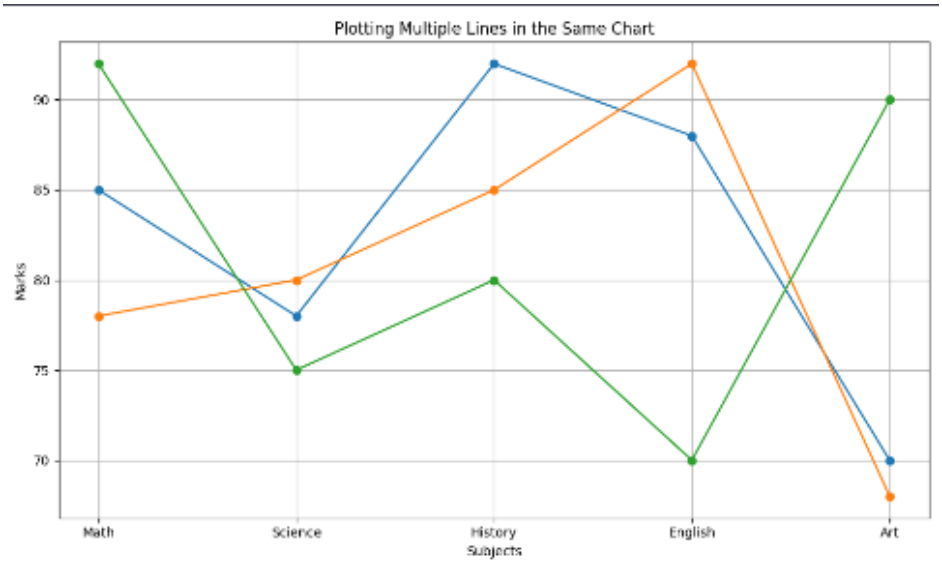
```python
import numpy as np

# Given data
names = np.array(['Ann', 'Joe', 'Mark'])
heights = np.array([1.5, 1.78, 1.6])
weights = np.array([65, 46, 59])

# Calculate BMI for each individual
bmi_values = weights / (heights ** 2)

# Define BMI categories
underweight_threshold = 18.5
overweight_threshold = 25

# Classify individuals based on BMI
bmi_categories = []
for bmi in bmi_values:
    if bmi < underweight_threshold:
        bmi_categories.append('Underweight')
    elif bmi > overweight_threshold:
        bmi_categories.append('Overweight')
    else:
        bmi_categories.append('Normal weight')

# Display results for each individual
for i in range(len(names)):
    print(f"{names[i]} - BMI: {bmi_values[i]:.2f} - Category: {bmi_categories[i]}")
```

```
C:\Users\PhucQuach\AppData\Local\Microsoft\Wind
Ann - BMI: 28.89 - Category: Overweight
Joe - BMI: 14.52 - Category: Underweight
Mark - BMI: 23.05 - Category: Normal weight
```

1.8

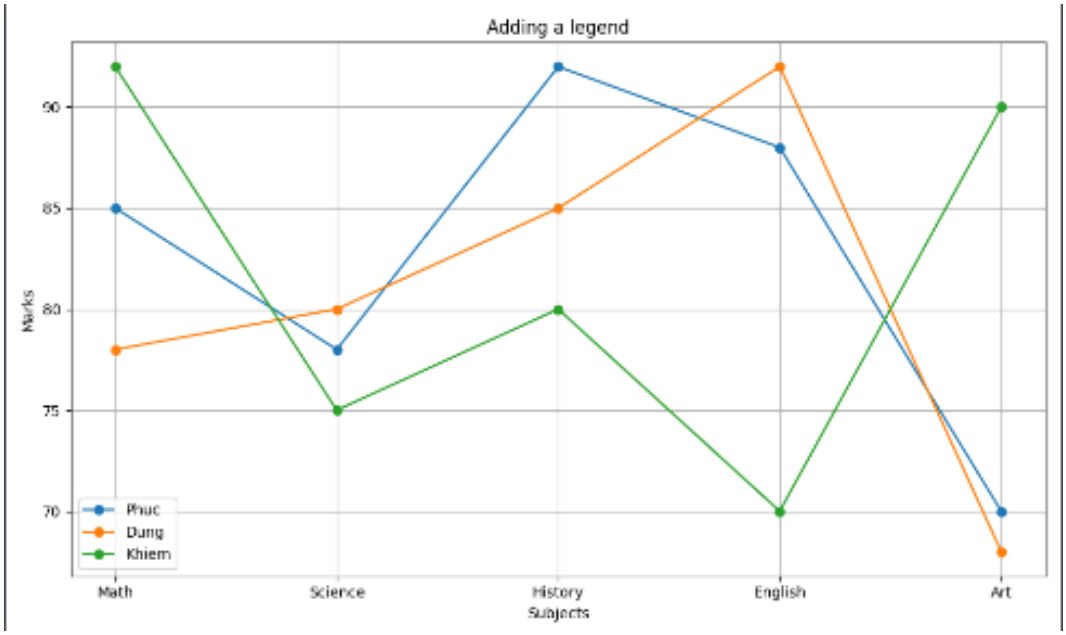
Plotting Multiple Lines in the Same Chart:

```python
import matplotlib.pyplot as plt
import numpy as np

# Sample student data
student_data = [
    {'student_name': 'Phuc', 'Math': 85, 'Science': 78, 'History': 92, 'English': 88, 'Art': 70},
    {'student_name': 'Dung', 'Math': 78, 'Science': 80, 'History': 85, 'English': 92, 'Art': 68},
    {'student_name': 'Khiem', 'Math': 92, 'Science': 75, 'History': 80, 'English': 70, 'Art': 90}
]

# Extract subjects and marks
subjects = ['Math', 'Science', 'History', 'English', 'Art']
marks = np.array([[student[subject] for subject in subjects] for student in student_data])
student_names = [student['student_name'] for student in student_data]

# Plotting Multiple Lines in the Same Chart
plt.figure(figsize=(10, 6))
for i in range(len(student_data)):
    plt.plot(subjects, marks[i, :], marker='o', label=student_names[i])

plt.xlabel('Subjects')
plt.ylabel('Marks')
plt.title('Plotting Multiple Lines in the Same Chart')
plt.grid(True)
plt.tight_layout()

# Show the plot
plt.show()
```



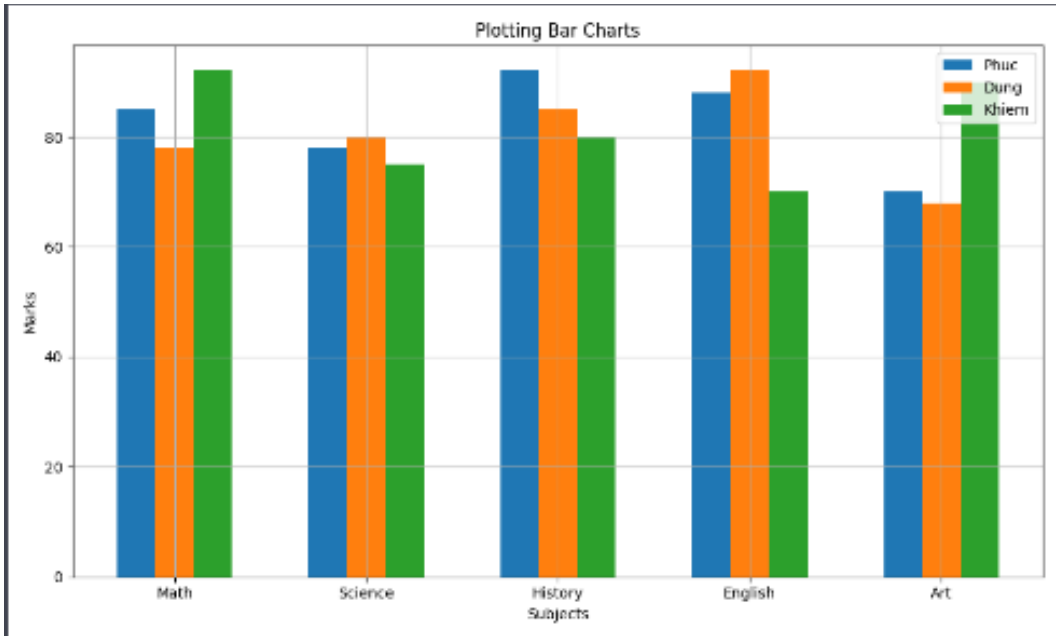
Adding a Legend:

```
# Adding a legend
plt.figure(figsize=(10, 6))
for i in range(len(student_data)):
    plt.plot(subjects, marks[i, :], marker='o', label=student_names[i])

plt.xlabel('Subjects')
plt.ylabel('Marks')
plt.title('Adding a legend')
plt.grid(True)
plt.tight_layout()
plt.legend()

# Show the plot after adding a legend
plt.show()
```



Plotting Bar Charts:

```
# Plotting Bar Charts
plt.figure(figsize=(10, 6))
x = np.arange(len(subjects))  # x-coordinates for bars
width = 0.2  # width of the bars

for i in range(len(student_data)):
    plt.bar(x + i * width, marks[i, :], width=width, label=student_names[i])

plt.xlabel('Subjects')
plt.ylabel('Marks')
plt.title('Plotting Bar Charts')
plt.xticks(x + (len(student_data) - 1) * width / 2, subjects)  # Place x-ticks in the center of groups
plt.legend()
plt.grid(True)
plt.tight_layout()

# Show the plot
plt.show()
```

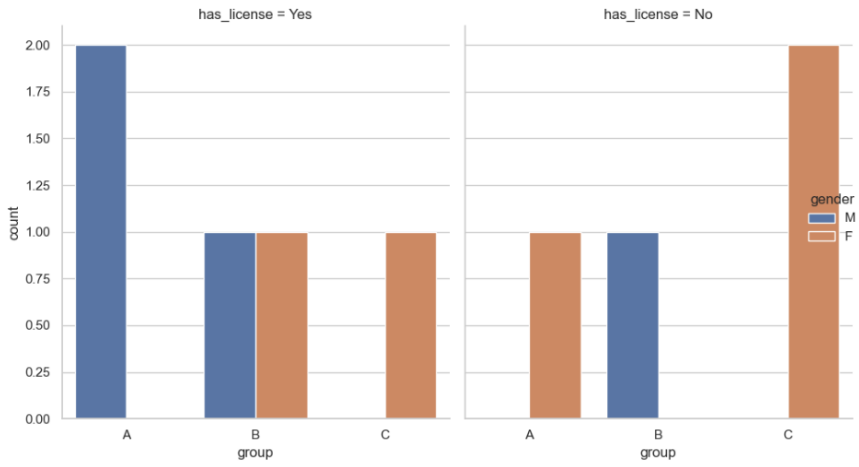


1.9

driver_data.csv

```
group,gender,has_license
A,M,Yes
A,F,No
A,M,Yes
B,F,Yes
B,M,No
B,M,Yes
C,F,No
C,F,Yes
C,F,No
```

```python
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt

# Read the CSV file
data = pd.read_csv('driver_data.csv')

# Create a categorical plot using Seaborn
plt.figure(figsize=(8, 6))
sns.set(style="whitegrid")
sns.catplot(data=data, x="group", hue="gender", col="has_license", kind="count", height=6, aspect=0.8)
plt.suptitle("Proportion of Men and Women with Driver's License in Each Group", y=1.02)
plt.tight_layout()

# Display the plot
plt.show()
```



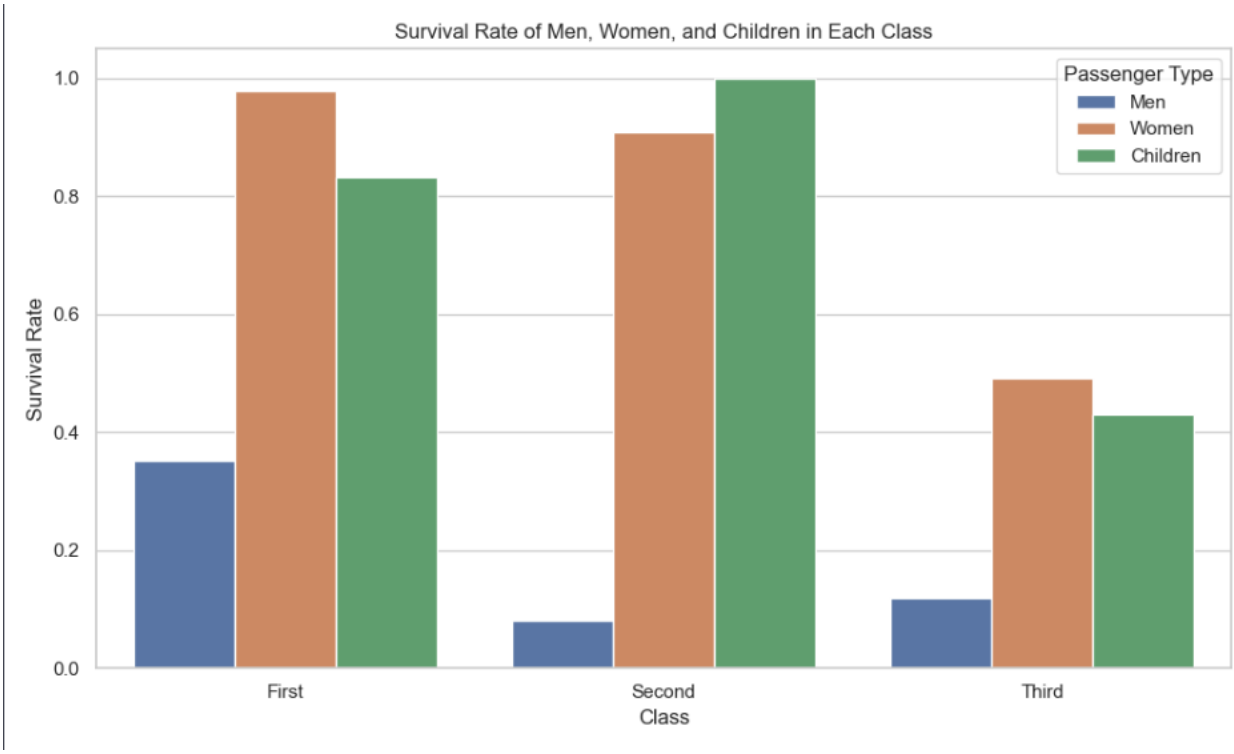
1.10

Titanic.csv

```
survived,pclass,sex,age,sibsp,parch,fare,embarked,class,who,adult_male,deck,embark_town,alive,alone
0,3,male,22.0,1,0,7.25,S,Third,man,True,,Southampton,no,False
1,1,female,38.0,1,0,71.2833,C,First,woman,False,C,Cherbourg,yes,False
1,3,female,26.0,0,0,7.925,S,Third,woman,False,,Southampton,yes,True
1,1,female,35.0,1,0,53.1,S,First,woman,False,C,Southampton,yes,False
0,3,male,35.0,0,0,8.05,S,Third,man,True,,Southampton,no,True
0,3,male,,0,0,8.4583,Q,Third,man,True,,Queenstown,no,True
0,1,male,54.0,0,0,51.8625,S,First,man,True,E,Southampton,no,True
0,3,male,2.0,3,1,21.075,S,Third,child,False,,Southampton,no,False
1,3,female,27.0,0,2,11.1333,S,Third,woman,False,,Southampton,yes,False
1,2,female,14.0,1,0,30.0708,C,Second,child,False,,Cherbourg,yes,False
1,3,female,4.0,1,1,16.7,S,Third,child,False,G,Southampton,yes,False
1,1,female,58.0,0,0,26.55,S,First,woman,False,C,Southampton,yes,True
0,3,male,20.0,0,0,8.05,S,Third,man,True,,Southampton,no,True
0,3,male,39.0,1,5,31.275,S,Third,man,True,,Southampton,no,False
0,3,female,14.0,0,0,7.8542,S,Third,child,False,,Southampton,no,True
1,2,female,55.0,0,0,16.0,S,Second,woman,False,,Southampton,yes,True
0,3,male,2.0,4,1,29.125,Q,Third,child,False,,Queenstown,no,False
1,2,male,,0,0,13.0,S,Second,man,True,,Southampton,yes,True
0,3,female,31.0,1,0,18.0,S,Third,woman,False,,Southampton,no,False
1,3,female,,0,0,7.225,C,Third,woman,False,,Cherbourg,yes,True
0,2,male,35.0,0,0,26.0,S,Second,man,True,,Southampton,no,True
1,2,male,34.0,0,0,13.0,S,Second,man,True,D,Southampton,yes,True
1,3,female,15.0,0,0,8.0292,Q,Third,child,False,,Queenstown,yes,True
1,1,male,28.0,0,0,35.5,S,First,man,True,A,Southampton,yes,True
0,3,female,8.0,3,1,21.075,S,Third,child,False,,Southampton,no,False
1,3,female,38.0,1,5,31.3875,S,Third,woman,False,,Southampton,yes,False
0,3,male,,0,0,7.225,C,Third,man,True,,Cherbourg,no,True
0,1,male,19.0,3,2,263.0,S,First,man,True,C,Southampton,no,False
1,3,female,,0,0,7.8792,Q,Third,woman,False,,Queenstown,yes,True
```

```python
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt

# Read the Titanic dataset CSV file
titanic_data = pd.read_csv('titanic.csv')


# Define a function to categorize passengers as men, women, or children
def categorize_passenger(row):
    if row['who'] == 'man':
        return 'Men'
    elif row['who'] == 'woman':
        return 'Women'
    else:
        return 'Children'
```

```python
# Apply the categorize_passenger function to create a new 'passenger_type' column
titanic_data['passenger_type'] = titanic_data.apply(categorize_passenger, axis=1)

# Calculate survival rate for each combination of passenger type and class
survival_rate_data = titanic_data.groupby(['class', 'passenger_type'])['survived'].mean().reset_index()

# Create a bar plot to visualize the survival rate of men, women, and children in each class
plt.figure(figsize=(10, 6))
sns.set(style="whitegrid")
sns.barplot(data=survival_rate_data, x='class', y='survived', hue='passenger_type',
            hue_order=['Men', 'Women', 'Children'])
plt.title("Survival Rate of Men, Women, and Children in Each Class")
plt.xlabel("Class")
plt.ylabel("Survival Rate")
plt.legend(title="Passenger Type")
plt.tight_layout()

# Display the plot
plt.show()
```

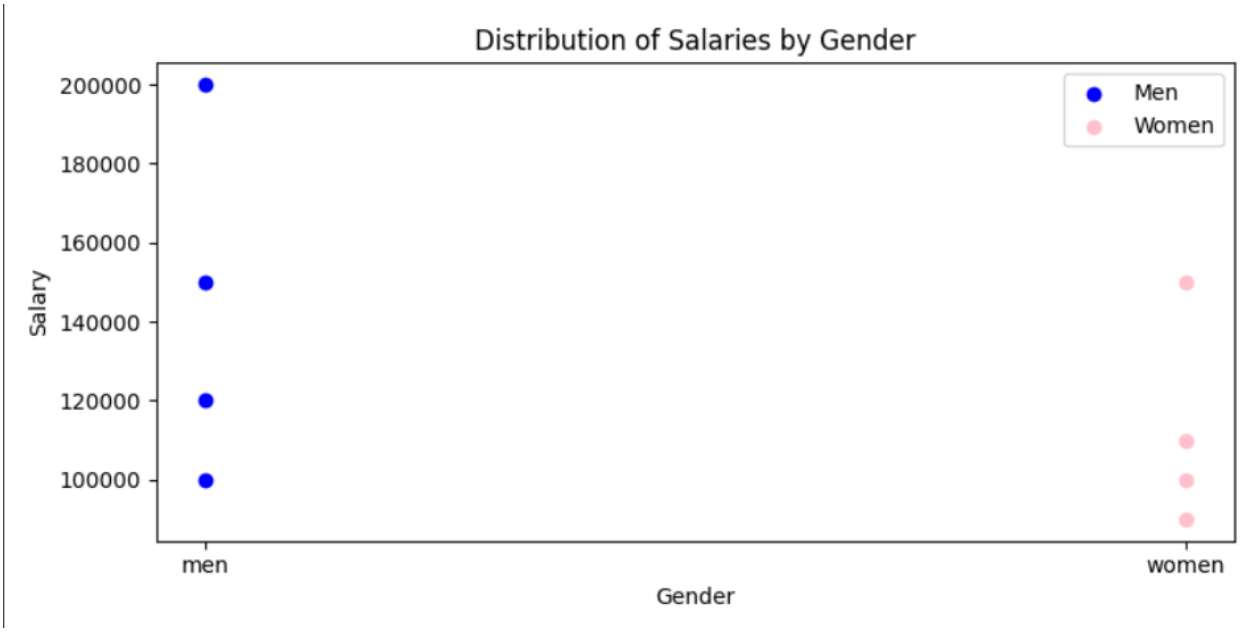


1.11

Salary.csv

	C1	C2
1	gender	salary
2	men	100000
3	men	120000
4	women	100000
5	men	200000
6	women	90000
7	women	150000
8	men	150000
9	women	110000

```python
import pandas as pd
import matplotlib.pyplot as plt

# Read the salary data from the CSV file
data = pd.read_csv('salary.csv')

# Separate data for men and women
men_data = data[data['gender'] == 'men']
women_data = data[data['gender'] == 'women']

# Create a scatter plot to show the distribution of salaries by gender
plt.figure(figsize=(8, 4))
plt.scatter(men_data['gender'], men_data['salary'], label='Men', color='blue')
plt.scatter(women_data['gender'], women_data['salary'], label='Women', color='pink')

# Add labels and title
plt.xlabel('Gender')
plt.ylabel('Salary')
plt.title('Distribution of Salaries by Gender')
plt.legend()

# Show the plot
plt.tight_layout()
plt.show()
```



1.12

```python
import numpy as np
from sklearn.linear_model import LinearRegression

# Original data
data = np.array([(50, 2.5), (60, 3), (65, 3.5), (70, 3.8), (75, 4), (80, 4.5), (85, 5)])

# Separate the data into two arrays: area and price
areas = data[:, 0].reshape(-1, 1)
prices = data[:, 1]

# Create a linear regression model
model = LinearRegression()

# Fit the model with the training data
model.fit(areas, prices)

# Predict prices for the given areas
new_areas = np.array([55, 68, 76, 90]).reshape(-1, 1)
predicted_prices = model.predict(new_areas)

# Print the prediction results
for area, price in zip(new_areas, predicted_prices):
    print(f"Area: {area} m2, Predicted Price: {price:.2f} billion")
```

```
C:\Users\PhucQuach\AppData\Local\Microsoft\Windo
Area: [55] m2, Predicted Price: 2.75 billion
Area: [68] m2, Predicted Price: 3.67 billion
Area: [76] m2, Predicted Price: 4.23 billion
Area: [90] m2, Predicted Price: 5.21 billion
```

1.13

```python
import numpy as np
from sklearn.linear_model import LinearRegression

# Sample data: (height in cm, weight in kg)
data = np.array([
    [150, 45],
    [160, 50],
    [165, 55],
    [170, 60],
    [175, 65],
    [180, 70]
])
heights = data[:, 0].reshape(-1, 1)
weights = data[:, 1]

model = LinearRegression()
model.fit(heights, weights)

new_heights = np.array([[155], [168], [172]])
predicted_weights = model.predict(new_heights)

for height, weight in zip(new_heights, predicted_weights):
    print(f"Height: {height[0]} cm, Predicted Weight: {weight:.2f} kg")
```

```
C:\Users\PhucQuach\AppData\Local\Microsoft\Win
Height: 155 cm, Predicted Weight: 47.50 kg
Height: 168 cm, Predicted Weight: 58.64 kg
Height: 172 cm, Predicted Weight: 62.07 kg
```

1.14

```python
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
import matplotlib.pyplot as plt

# Load the Boston Housing Dataset
df = pd.read_csv(
    filepath_or_buffer="http://lib.stat.cmu.edu/datasets/boston",
    delim_whitespace=True,
    skiprows=21,
    header=None,
)
```

```python
# Define column names
columns = [
    'CRIM',
    'ZN',
    'INDUS',
    'CHAS',
    'NOX',
    'RM',
    'AGE',
    'DIS',
    'RAD',
    'TAX',
    'PTRATIO',
    'B',
    'LSTAT',
    'MEDV',
]

# Flatten all the values into a single long list and remove the nulls
values_w_nulls = df.values.flatten()
all_values = values_w_nulls[~np.isnan(values_w_nulls)]

# Reshape the values to have 14 columns and make a new DataFrame out of them
df = pd.DataFrame(
    data=all_values.reshape(-1, len(columns)),
    columns=columns,
)
```

```
# Select features and target
X = df.drop('MEDV', axis=1)
y = df['MEDV']

# Split the dataset into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# Create a linear regression model
model = LinearRegression()
model.fit(X_train, y_train)

# Make predictions on the test data
y_pred = model.predict(X_test)

# Print actual price and predicted price
for actual_price, predicted_price in zip(y_test, y_pred):
    print(f"Actual Price: {actual_price:.2f}, Predicted Price: {predicted_price:.2f}")

# Visualize predicted vs. actual prices
plt.scatter(y_test, y_pred)
plt.xlabel("Actual Price")
plt.ylabel("Predicted Price")
plt.title("Actual vs. Predicted Prices")
plt.show()
```

```
C:\Users\PhucQuach\AppData\Local\Microsoft\WindowsApps\p
Actual Price: 23.60, Predicted Price: 29.00
Actual Price: 32.40, Predicted Price: 36.03
Actual Price: 13.60, Predicted Price: 14.82
Actual Price: 22.80, Predicted Price: 25.03
Actual Price: 16.10, Predicted Price: 18.77
Actual Price: 20.00, Predicted Price: 23.25
Actual Price: 17.80, Predicted Price: 17.66
Actual Price: 14.00, Predicted Price: 14.34
Actual Price: 19.60, Predicted Price: 23.01
Actual Price: 16.80, Predicted Price: 20.63
Actual Price: 21.50, Predicted Price: 24.91
Actual Price: 18.90, Predicted Price: 18.64
Actual Price: 7.00, Predicted Price: -6.09
Actual Price: 21.20, Predicted Price: 21.76
Actual Price: 18.50, Predicted Price: 19.24
Actual Price: 29.80, Predicted Price: 26.19
Actual Price: 18.80, Predicted Price: 20.65
Actual Price: 10.20, Predicted Price: 5.79
Actual Price: 50.00, Predicted Price: 40.50
Actual Price: 14.10, Predicted Price: 17.61
Actual Price: 25.20, Predicted Price: 27.25
Actual Price: 29.10, Predicted Price: 30.07
Actual Price: 12.70, Predicted Price: 11.34
Actual Price: 22.40, Predicted Price: 24.16
Actual Price: 14.20, Predicted Price: 17.86
Actual Price: 13.80, Predicted Price: 15.84
Actual Price: 20.30, Predicted Price: 22.78
```