TRƯỜNG ĐẠI HỌC – MỎ ĐỊA CHẤT
KHOA CÔNG NGHỆ THÔNG TIN

BÁO CÁO TỔNG KẾT
ĐỀ TÀI NCKH SINH VIÊN

ĐỀ TÀI: Nghiên cứu, ứng dụng học máy xây dựng mô hình phân loại nấm độc và nấm ăn được.

Giáo viên hướng dẫn: Nguyễn Thị Thùy Dương

Hà Nội - 2022

TRƯỜNG ĐẠI HỌC – MỎ ĐỊA CHẤT
KHOA CÔNG NGHỆ THÔNG TIN

BÁO CÁO TỔNG KẾT
ĐỀ TÀI NCKH SINH VIÊN

ĐỀ TÀI : Nghiên cứu, ứng dụng học máy xây dựng mô hình phân loại nấm độc và nấm ăn được.

Thành viên trong nhóm :

Nguyễn Văn Đức	1921050737
Hoàng Nhật Huy	1921050296
Phí Thị Quỳnh	1921050503
Nguyễn Văn Quang	1921050480
Hoàng Thị Ngọc Diễm	1921050121

Hà Nội - 2022

MỤC LỤC

DANH MỤC HÌNH VẼ

CHƯƠNG 1: GIỚI THIỆU ĐỀ TÀI

1.1 Lời mở đầu.

Trí tuệ nhân tạo (AI- artificial intelligence) là nói đến khả năng của máy khi thực hiện các công việc mà con người thường phải xử lý; và khi dáng vẻ ứng xử hoặc kết quả thực hiện của máy là tốt hơn hoặc tương đương với con người thì ta gọi đó là máy thông minh hay máy đó có trí thông minh. Hay nói cách khác, đánh giá sự thông minh của máy không phải dựa trên nguyên lý nó thực hiện nhiệm vụ đó có giống cách con người thực hiện hay không mà dựa trên kết quả hoặc dáng vẻ ứng xử bên ngoài của nó có giống với kết quảhoặc dáng vẻ ứng xử của con người hay không.

Trong những năm gần đây, AI là một cơn sốt lớn, nó như một cuộc cách mạng thứ 4 trên thế giới. Các nhiệm vụ của con người thường xuyên phải thực hiện là: giải bài toán (tìm kiếm, chứng minh, lập luận), học, giao tiếp, thể hiện cảm xúc, thích nghi với môi trường xung quanh… và dựa trên kết quả thực hiện các nhiệm vụ đó để kết luận rằng một ai đó có là thông minh hay không. Môn học Trí tuệ nhân tạo nhằm cung cấp các phương pháp luận để làm ra hệ thống có khả năng thực hiện các nhiệm vụ đó: giải toán, học, giao tiếp, v.v. bất kể cách nó làm có như con người hay không mà là kết quả đạt được hoặc dáng vẻ bên ngoài như con người.

1.2 Giới thiệu chung về đề tài.

Học máy là một bộ phận con của AI. Với đề tài này, Việc xác định nấm ở đây sử dụng bài toán phân phân lớp để dự đoán khả năng phát hiện nấm độc hay nấm không độc. Việc xác định nấm đôc hay nấm không độc liên quan đến môi trường , hình dạng, màu sắc của nấm.

Hiện nay, việc phát hiện nấm độc là việc làm rất cần thiết vì số người ăn và sử dụng phải nấm độc gây nguy hại rất nhiều cho sức khỏe đôi khi còn dẫn đến tử vong.

Thống kê trong 10 năm gần đây, các trung tâm chống độc ở Mỹ đã thống kê 85.556 trường hợp ngộ độc nấm độc trên thế giới. Vì vậy, nhóm em đã tìm hiểu, nghiên cứu về ứng dụng của học máy về khả năng dự đoán của những loại nấm trên thế giới.

Nhóm đã nghiên cứu 4 nội dung gồm:

1. *Tổng quan về học máy*
2. *Tiền xử lý dữ liệu*
3. *Thuật toán KNN cho bài toán phân lớp nhị phân*
4. *Xây dựng mô hình học máy*

Mục đích của đề tài : Dựa trên dữ liệu về màu sắc, kích thước, hình dạng,môi trường để phân biệt được nấm độc và nấm ăn được.

- Mục tiêu của đề tài :

+ Nghiên cứu tổng quan về học máy.

+ Tìm hiểu việc lập trình các thuật toán trong học máy và sử dụng ngôn ngữ lập trình Python các thư viện Pandas tiền xử lý dữ liệu đã có và thư viện Scikit-learn .

+ Nghiên cứu về thuật toán KNN.

+ Xây dựng được mô hình học máy dự đoán nấm độc và nấm ăn đươc.

- Đối tượng nghiên cứu:

+ Bộ dữ liệu: file mushroom.csv gồm có 8125 loại tương tứng với 23 cột là các thuộc tính của nấm.Trong 23 thuộc tníh thì có 22 thuộc tính độc lập và 1 thuộc tính phụ thuộc.

+ Thư viện Pandas: là một thư viện Python cung cấp các cấu trúc dữ liệu nhanh, mạnh mẽ, linh hoạt và mang hàm ý. Tên thư viện được bắt nguồn từ panel

data (bảng dữ liệu). Pandas được thiết kế để làm việc dễ dàng và trực quan với dữ liệu có cấu trúc (dạng bảng, đa chiều, có tiềm năng không đồng nhất) và dữ liệu chuỗi thời gian.

+ Thư viện Matplotlib là một trong những thư viện Python phổ biến nhất được sử dụng để trực quan hóa dữ liệu. Nó là một thư viện đa nền tảng để tạo các đồ thị 2D từ dữ liệu trong các mảng. Matplotlib được viết bằng Python và sử dụng NumPy, phần mở rộng toán học của Python. Nó cung cấp một API hướng đối tượng giúp nhúng các plot trong các ứng dụng và sử dụng bộ công cụ GUI Python như PyQt, WxPythonotTkinter. Ngoài ra có thể được sử dụng trong Python và IPython shell, Jupyter Notebook và các máy chủ web.

+ Thư viện NumPy là thư viện lõi phục vụ cho khoa học máy tính của Python. Nó cung cấp một đối tượng mảng đa chiều hiệu suất cao và các công cụ để làm việc với các mảng này.

+Thư viện học máy SciKit-Learn:....- Phạm vi nghiên cứu: Nghiên cứu cấu trúc, cách thông số trong tập dữ liệu mushroom.csv. Nghiên cứu thư viện học máy SciKit-Learn và thuật toán KNN và xây dựng mô hình học máy phân loại nấm độc và không độc .

Phạm vi nghiên cứu: Dữ liệu từ 1626 loại nấm, tương ứng với 1626 dòng trong file dữ liệu, mỗi dòng có 23 thuộc tính tương ứng với các thông tin khác nhau trong mỗi loại nấm và thư viện học máy scikit learn và đặc biệt là mô hình KNN.

Quan sát dữ liệu ban đầu:

Để quan sát được tập dữ liệu, đầu tiên ta phải gọi các thư viện cần dùng trong bài toán này và đường link dẫn đến file lưu dữ liệu , ở đây, chúng ta cần sử dụng thư viện pandas numpy, seaborn, matplotlib với cú pháp:

```
import pandas as pd (với thư viện pandas)

import numpy as np (với thư viện numpy)

import random as rnd (với thư viện random để lấy kết quả ngẫu nhiên)
```

Các thư viện sau hỗ trợ trực quan hóa dữ liệu:

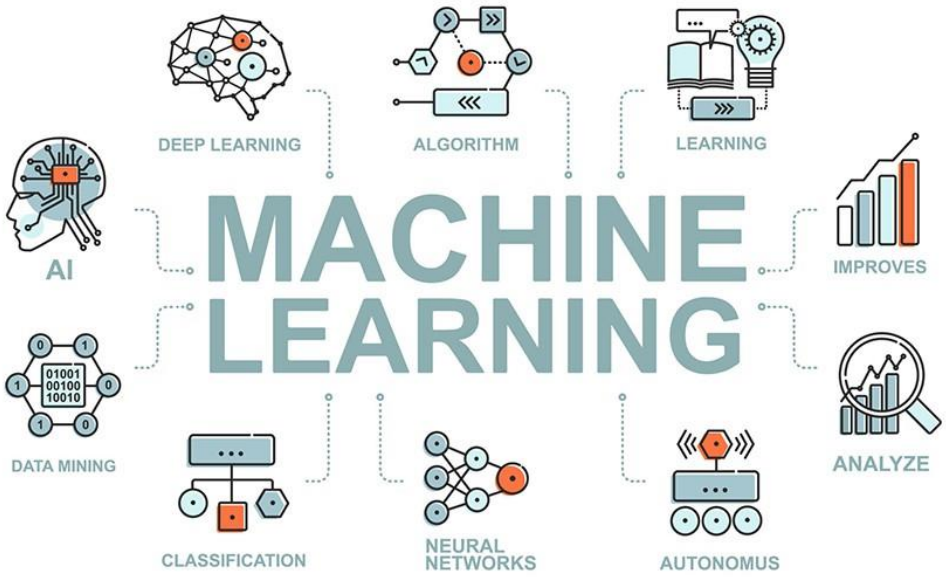
```
import seaborn as sns (với thư viện seaborn cung cấp APIs xây dựng)
```

trên thư viện Matplotlib, cung cấp các lựa chọn cho các kiểu mặc định và màu sắc, định nghĩa các chức năng cấp cao đơn giản cho các kiểu thống kê chung và tích hợp với các chức năng do Pandas DataFrames cung cấp.

CHƯƠNG 2: TỔNG QUAN VỀ HỌC MÁY

2.1. Tổng quan về học máy:

2.1.1. Khái niệm về học máy là gì ?



Hình 1: Hình minh họa tổng quan khái niệm học máy.

Học máy hay máy học trong tiếng Anh là Machine learning, viết tắt: ML.

Học máy (ML) là một công nghệ phát triển từ lĩnh vực trí tuệ nhân tạo. Các thuật toán machine learning là các chương trình máy tính có khả năng học hỏi về cách hoàn thành các nhiệm vụ và cách cải thiện hiệu suất theo thời gian.

Học máy vẫn đòi hỏi sự đánh giá của con người trong việc tìm hiểu dữ liệu cơ sở và lựa chọn các kĩ thuật phù hợp để phân tích dữ liệu. Đồng thời, trước khi sử dụng, dữ liệu phải sạch, không có sai lệch và không có dữ liệu giả.

Các mô hình machine learning yêu cầu lượng dữ liệu đủ lớn để "huấn luyện" và đánh giá mô hình. Trước đây, các thuật toán machine learning thiếu quyền truy cập vào một lượng lớn dữ liệu cần thiết để mô hình hóa các mối quan hệ giữa các

dữ liệu. Sự tăng trưởng trong dữ liệu lớn (big data) đã cung cấp các thuật toán machine learning với đủ dữ liệu để cải thiện độ chính xác của mô hình và dự đoán.

2.1.2. Ứng dụng của học máy.

Machine learning cũng có những giới hạn của nó. Chúng ta không thể nào xây dựng một cỗ máy thông minh để học dữ liệu từ cổ chí kim tới hiện tại. Tuy nhiên, đã có những ứng dụng thực tế mà machine learning làm rất tốt. Sau đây là các lĩnhvực phổ biến mà machine learing góp mặt:

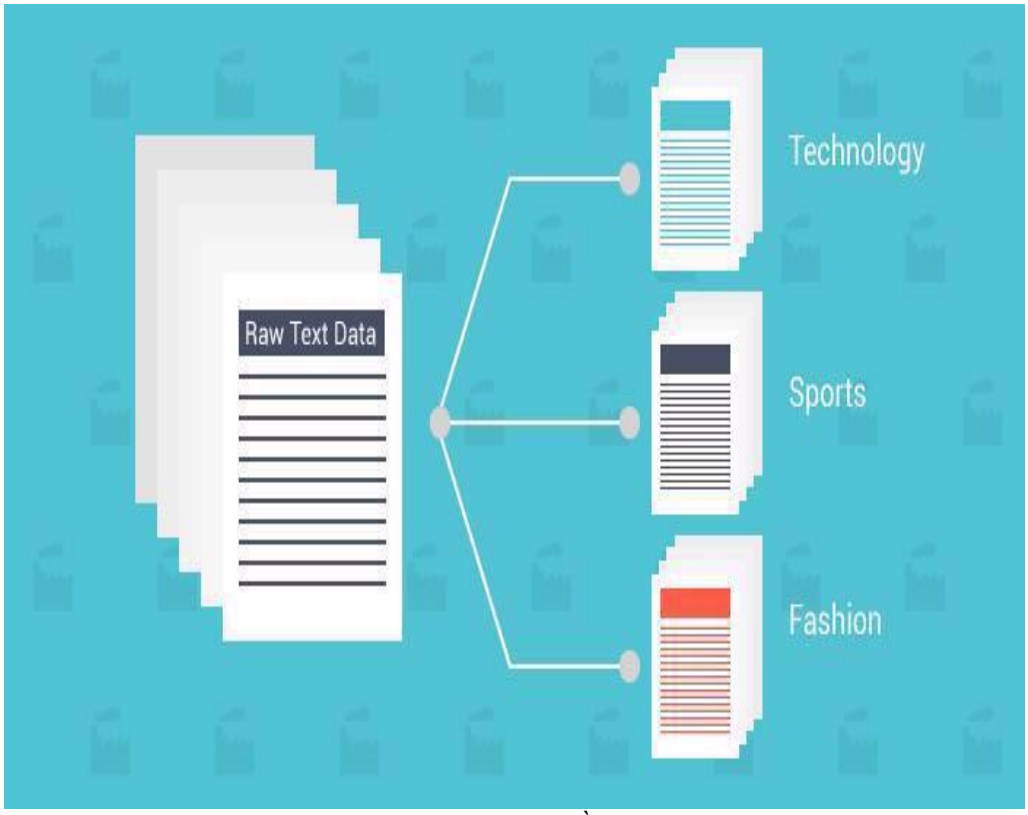
- Xử lý ảnh:

Hình 2: Hình minh họa về xử lý hình ảnh.

Bài toán xử lý ảnh (Image Processing) giải quyết các vấn đề phân tích thông tintừ hình ảnh hay thực hiện một số phép biến đổi. Một số ví dụ là:

- *Gắn thẻ hình ảnh* (Image Tagging): giống như Facebook, một thuật toán tự động phát hiện khuôn mặt của chúng ta và chúng ta bè trên những bức ảnh. Về cơ bản, thuật toán này học từ những bức ảnh mà chúng ta tự gắn thẻ cho mình trước đó.
- *Nhận dạng ký tự* (Optical Character Recognition): là một thuật toán chuyển dữ liệu trên giấy tờ, văn bản thành dữ liệu số hóa. Thuật toán phải học cách nhận biết ảnh chụp của một ký tự là ký tự nào.
- *Ô tô tự lái* (Self-driving cars): một phần cơ chế sử dụng ở đây là xử lý ảnh. Một thuật toán machine learning giúp phát hiện các mép đường, biển báo hay các chướng ngại vật bằng cách xem xét từng khung hình video từ camera.

- Phân tích văn bản:



Hình 3: Hình minh họa về phân tích văn bản.

Phân tích văn bản (Text analysis) là công việc trích xuất hoặc phân lọi thông tin từ văn bản. Các văn bản ở đây có thể là các facebook posts, emails, các đoạn chats, tài liệu, …Một số ví dụ phổ biến là:

- *Lọc spam* (Spam filtering): là một trong những ứng dụng phân loại văn bản được biết và sử dụng nhiều nhất. Ở đây, phân loại văn bản là xác định chủ đề cho một văn bản. Bộ lọc spam sẽ học cách phân loại một email có phải spam không dựa trên nội dung và tiêu đề của email.
- *Phân tích ngữ nghĩa* (Sentiment Analysis): học cách phân loại một ý kiến là tích cực, trung tính hay tiêu cực dựa trên nội dung văn bản của người viết.
- *Khai thác thông tin* (Information Extraction): từ một văn bản, học cách để trích xuất các thông tin hữu ích. Chẳng hạn như trích xuất địa chỉ, tên người, từ khóa, …

- Khai phá dữ liệu:

Hình 4: Hình minh họa về khai phá dữ liệu.

Khai phá dữ liệu (Data mining) là quá trình khám phá ra các thông tin có giá trị hoặc đưa ra các dự đoán từ dữ liệu. Định nghĩa này có vẻ bao quát, nhưng chúng ta hãy nghĩ về việc tìm kiếm thông tin hữu ích từ một bảng dữ liệu rất lớn. Mỗi bản ghi sẽ là một đối tượng cần phải học, và mỗi cột là một đặc trưng. Chúng ta có thể dự đoán giá trị của một cột của bản ghi mới dựa trên các bản ghi đã học. Hoặc là phân nhóm các bản ghi của bản. Sau đây là những ứng dụng của khai phá dữ liệu:

- *Phát hiện bất thường* (Anomaly detection): phát hiện các ngoại lệ, ví dụ như phát hiện gian lận thẻ tín dụng. Chúng ta có thể phát hiện một giao dịch là khả nghi dựa trên các giao dịch thông thường của người dùng đó.
- *Phát hiện các quy luật* (Association rules): ví dụ, trong một siêu thị hay một trang thương mại điện tử. Chúng ta có thể khám phá ra khách hàng thường mua các món hàng nào cùng nhau. Dễ hiểu hơn, khách hàng của chúng ta khi mua món hàng A thường mua kèm món hàng nào? Các thông tin này rất hữu ích cho việc tiếp thị sản phẩm.

- *Gom nhóm*(Grouping): ví dụ như trong các nền tảng SaaS, người dùng được phân nhóm theo hành vi hoặc thông tin hồ sơ của họ.
- *Dự đoán*(Predictions): các cột giá trị (của một bản ghi mới trong database). Ví dụ, chúng ta có thể dự đoán giá của căn hộ dựa trên các dữ liệu về giá các căn hộ chúng ta đã có.

- Trò chơi điện tử & Robot:

Trò chơi điện tử (Video games) và robot(Robotics) là lĩnh vực lớn có sự góp mặt của machine learning. Nếu ta có một nhân vật cần di chuyển và tránh các chướng ngại vật trong game. Machine learning có thể học và giải quyết công việc này thay chúng ta. Một kỹ thuật phổ biến được áp dụng trong trường hợp này là Học tăng cường (Reinforcement learning). Ở đó, máy sẽ học tăng cường với mục tiêu là giải quyết nhiệm vụ trên. Học tăng cường là tiêu cực nếu nó va phải chướngngại vật, là tích cực nếu nó chạm tới đích.

Một thành tựu gần đây nhất là cỗ máy Alpha Go của Google DeepMind đã đánh bại kỳ thủ cờ vậy số 1 thế giới. Trong khi cờ vây là một trò chơi có không gian trạng thái cực kỳ lớn.

Hình 5: Ví dụ cỗ máy Alpha Go của Google DeepMind.

2.1.3. Phân loại học máy :

a/ Phân loại theo phương thức học :

- **Học có giám sát :**

Học có giám sát (supervised learning) là một kỹ thuật của ngành học máy nhằm mục đích xây dựng một hàm f từ dữ tập dữ liệu huấn luyện (Training data). Dữ liệu huấn luyện bao gồm các cặp đối tượng đầu vào và đầu ra mong muốn. Đầu ra của hàm f có thể là một giá trị liên tục hoặc có thể là dự đoán một nhãn phân lớp cho một đối tượng đầu vào.

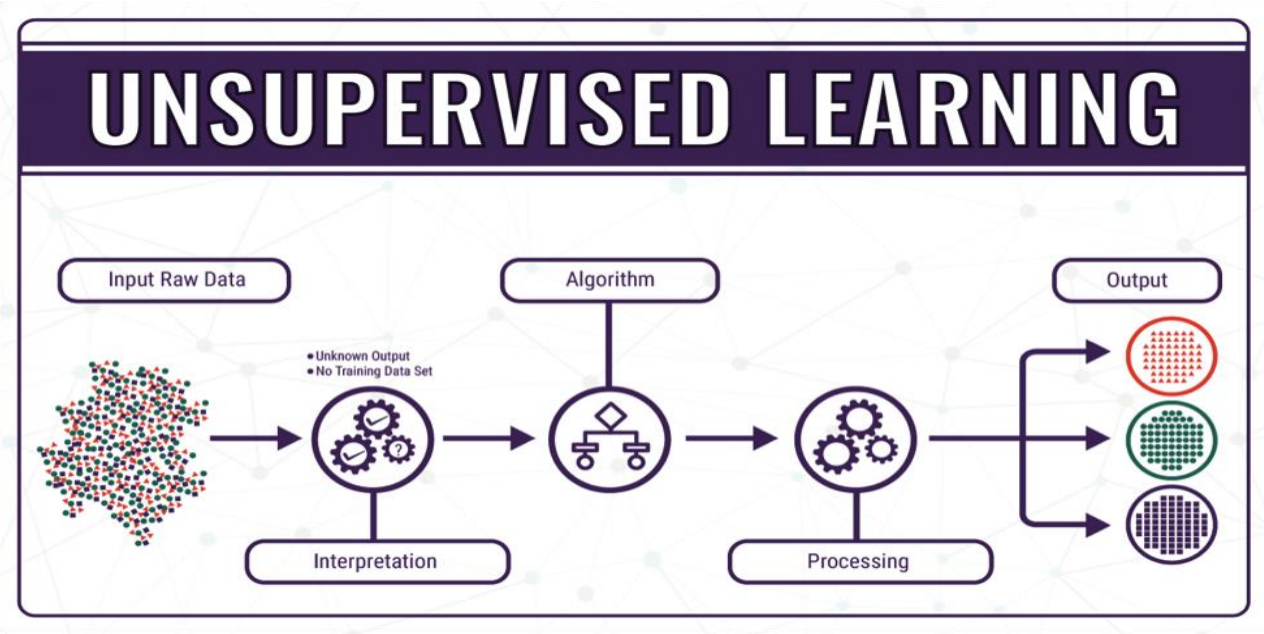
- **Cách giải một bài toán học có giám sát:**

- Bước 1: Xác định loại của các dữ liệu huấn luyện: Trước tiên ta cần phải quyết định xem loại dữ liệu nào sẽ được sử dụng làm dữ liệu huấn luyện. Ta có thể chọn dữ liệu một kí tự viết tay đơn lẻ, toàn bộ một từ viết tay, hay toàn bộ mộtdòng chữ viết tay, …
- Bước 2: Thu thập tập dữ liệu huấn luyện. Khi thu thập tập dữ liệu huấn luyện cần phải đảm bảo được sự đặc trưng cho thực tế sử dụng của hàm

chức năng. Do đó tập các dữ liệu đầu vào và đầu ra tương ứng phải được thu thập từ các chuyên gia hoặc từ việc đo đạc tính toán.

- Bước 3: Xác định việc biểu diễn các đặc trưng đầu vào cho hàm mục tiêu cần tìm. Độ chính xác của mục tiêu phụ thuộc rất lớn vào các đối tượng đầu vào được biểu diễn như thế nào. Đa số các đối tượng đầu vào được chuyển đổi thành một véc tơ đặc trưng chứa các đặc trưng cơ bản của đối tượng đó. Chú ý số lượng các đặc trưng không được lớn quá, để tránh sự bùng nổ tổ hợp tuy nhiên nó phải đủ lớn để đảm bảo dự đoán chính xác đầu ra.
- Bước 4: Xác định cấu trúc của hàm mục tiêu cần tìm và giải thuật học tươngứng. Ví dụ, ta có thể sử dụng mạng nơ-ron nhân tạo, cây quyết định, …
- Bước 5: Hoàn thiện và thiết kế chương trình.Tiến hành chạy giải thuật học với tập dữ liệu huấn luyện thu thập được. Ta có thể điều chỉnh các tham số của giải thuật học bằng cách tối ưu hóa hiệu năng trên một tập con của tập huấn luyện, (gọi là tập kiểm chứng -validation set) của tập huấn luyện hay thông qua kiểm chứng chéo (cross-validation). Sau đó ta tiến hành đo đạc hiệu năng của giải thuật trên một tập dữ liệu kiểm tra độc lập với tập huấn luyện.

- Học không giám sát:



Hình 6: Hình minh họa học không giám sát.

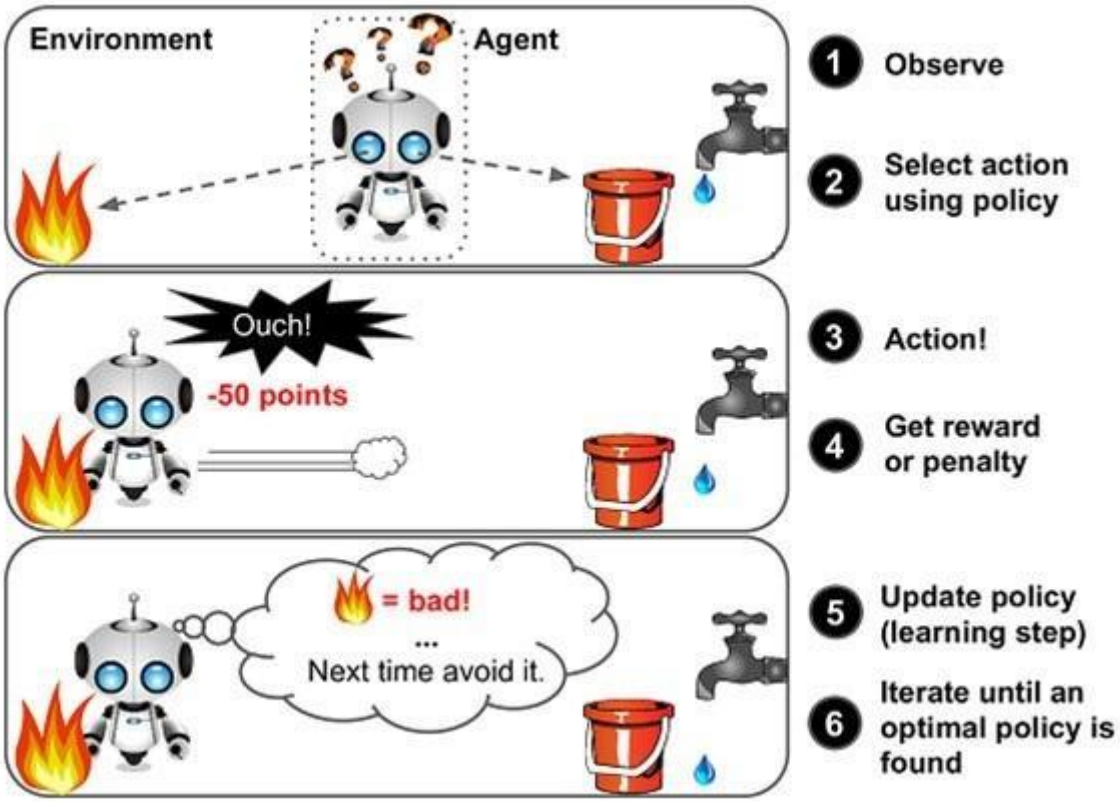
Học không giám sát cũng là một nhánh trong machine learning. Các mẫu dữ liệu trong học không giám sát chỉ cần input (đầu vào) mà không cần label (đầu ra). Nó được sử dụng nhiều trong việc khám phá cấu trúc và mối quan hệ của dữ liệu. Một thuật toán điển hình là bài toán phân cụm (clustering algorithm); Nó học cách để tìm các mẫu dữ liệu tương tự nhau và nhóm vào thành các cụm(cluster). Một số thuật toán phân cụm như K-means học cách phân cụm chỉ học từ tập dữ liệu đầu vào.

- Học tăng cường :

Học tăng cường (Reinforcement Learning-RL) là một trong ba kiểu học máy chính bên cạnh học giám sát (Supervised Learning) và học không giám sát (Unsupervised Learning). Bản chất của RL là trial-and-error, nghĩa là thử đi thử lại và rút ra kinh nghiệm sau mỗi lần thử như vậy. Gần đây, RL đã đạt được những thành tựu đáng kể khi các thuật toán của DeepMind (AlphaGo, AlphaZero,

AlphaStar,...) đã chiến thắng áp đảo các tuyển thủ thế giới trong những trò chơi mà con người đã từng nghĩ rằng máy móc sẽ không bao giờ có thể vượt mặt như cờ vây hay StarCraft.

Học tăng tường hay học củng cố là bài toán giúp cho một hệ thống tự động xác định hành vi dựa trên hoàn cảnh để đạt được lợi ích cao nhất. Hiện tại, reinforcement learning chủ yếu được áp dụng vào Lý Thuyết Trò Chơi (Game Theory), các thuật toán cần xác định nước đi tiếp theo để đạt được điểm số cao nhất. Hình 8 là một ví dụ đơn giản sử dụng học tăng cường.



Hình 7: Hình minh họa học máy tăng cường.

AlphaGo - một phần mềm chơi cờ vây trên máy tính được xây dựng bởi Google DeepMind hay chương trình dạy máy tính chơi game Mario là những ứng dụng sử dụng học tăng cường.

Cờ vậy được xem là trò chơi có độ phức tạp cực kỳ cao với tổng số nước đi là xấp xỉ 1076110761, so với cờ vua là 1012010120, vì vậy thuật toán phải chọn ra một nước đi tối ưu trong số hàng tỉ tỉ lựa chọn. Về cơ bản, AlphaGo bao gồm các thuật toán thuộc cả Supervised learning và Reinforcement learning. Trong phần Supervised learning, dữ liệu từ các ván cờ do con người chơi với nhau được đưa vào để huấn luyện. Tuy nhiên, mục tiêu cuối cùng của AlphaGo không phải là chơi như con người mà phải thắng được con người. Vì vậy, sau khi học xong các ván cờ của con người, AlphaGo tự chơi với chính nó thông qua hàng triệu ván cờ để tìm ra các nước đi mới tối ưu hơn. Thuật toán trong phần tự chơi này được xếp vào loại Reinforcement learning.

Đơn giản hơn cờ vây, tại một thời điểm cụ thể, người chơi game Mario chỉ cần bấm một số lượng nhỏ các nút (di chuyển, nhảy, bắn đạn) hoặc không cần bấm nút nào ứng với một chướng ngại vật cố định ở một vị trí cố định. Khi đó thuật toán trong ứng dụng dạy máy tính chơi game Mario sẽ nhận đầu vào là sơ đồ của màn hình tại thời điểm hiện hành, nhiệm vụ của thuật toán là tìm ra tổ hợp phím nên được bấm ứng với đầu vào đó. Việc huấn luyện này được dựa trên điểm số cho việc di chuyển được bao xa với thời gian bao lâu trong game, càng xa và càng nhanh thì điểm thưởng đạt được càng cao, tất nhiên điểm thưởng này không phải là điểm của trò chơi mà là điểm do chính người lập trình tạo ra. Thông qua huấn luyện, thuật toán sẽ tìm ra một cách tối ưu để tối đa số điểm trên, qua đó đạt được mục đích cuối cùng là cứu công chúa.

Có nhiều cách khác nhau để thuật toán có thể mô hình hóa một vấn đề dựa trên sự tương tác của nó với dữ liệu đầu vào. Phân loại hoặc cách tổ chức thuật toán học máy này rất hữu ích vì nó buộc chúng ta phải suy nghĩ về vai trò của dữ liệu đầu vào và quy trình chuẩn bị mô hình và chọn một thuật toán phù hợp nhất cho vấn đề của chúng ta để có kết quả tốt nhất.

b/ Phân loại theo chức năng :

- Các thuật toán hồi quy (Regression Algorithms)

Hồi quy là quá trình tìm mối quan hệ phụ thuộc của một biến (được gọi là biến phụ thuộc hay biến được giải thích, biến được dự báo, biến được hồi quy, biến phản ứng, biến nội sinh) vào một hoặc nhiều biến khác (được gọi là biến độc lập, biến giải thích, biến dự báo, biến hồi quy, biến tác nhân hay biến kiểm soát, biến ngoại sinh) nhằm mục đích ước lượng hoặc tiên đoán giá trị kỳ vọng của biến phụ thuộc khi biết trước giá trị của biến độc lập. Hình 6 tượng trưng cho ý tưởng của các thuật toán hồi quy.

Ví dụ như, dự đoán rằng nếu tăng lãi suất tiền gửi thì sẽ huy động được lượng tiền gửi nhiều hơn, khi đó ngân hàng A cần biết mối quan hệ giữa lượng tiền gửi và lãi suất tiền gửi, cụ thể hơn họ muốn biết khi tăng lãi suất thêm 0.1% thì lượng tiền gửi sẽ tăng trung bình là bao nhiêu.

- Thuật toán dựa trên mẫu (Instance-based Algorithms)

Mô hình học tập dựa trên mẫu hay thực thể là bài toán ra quyết định dựa vào các trường hợp hoặc các mẫu dữ liệu huấn luyện được coi là quan trọng hay bắt buộc đối với mô hình.

Nhóm thuật toán này thường xây dựng cơ sở dữ liệu về dữ liệu mẫu và so sánh dữ liệu mới với cơ sở dữ liệu bằng cách sử dụng thước đo tương tự để tìm kết quả phù hợp nhất và đưa ra dự đoán. Khi đó trọng tâm được đặt vào đại diện của các thể hiện .

- Thuật toán chuẩn hoá (Regularization Algorithms)

Các thuật toán chuẩn hoá ra đời từ sự mở rộng các phương pháp đã có (điển hình là các phương pháp hồi quy) bằng cách xử phạt các mô hình dựa trên mức độ phức tạp của chúng. Việc ưu tiên các mô hình đơn giản hơn cũng tốt hơn trong việc khái quát hóa.

- Thuật toán cây quyết định (Decision Tree Algorithms)

Đây là phương pháp xây dựng mô hình ra quyết định dựa trên các giá trị thực của những thuộc tính trong dữ liệu. Sự quyết định được rẽ nhánh trong cấu trúc cây cho đến khi quyết định dự đoán được đưa ra cho một mẫu nhất định. Phương pháp này được sử dụng trong việc huấn luyện dữ liệu cho bài toán phân lớp và hồi quy. Vì sự nhanh chóng, chính xác nên phương pháp này rất được ưa chuộng trong ML.

- Thuật toán phân cụm (Clustering Algorithms)

Tất cả các phương pháp đều sử dụng các cấu trúc vốn có trong dữ liệu để tổ chức tốt nhất dữ liệu thành các nhóm có mức độ phổ biến tối đa dựa vào trọng tâm (centroid) và thứ bậc (hierarchal) .

- Các thuật toán luật kết hợp (Association Rule Learning Algorithms)

Đây là những thuật toán sẽ rút trích ra các quy tắc giải thích tốt nhất mối quan hệ giữa các biến trong dữ liệu. Các quy tắc này có thể giúp khám phá ra các tính chất quan trọng và hữu ích trong các tập dữ liệu lớn và cao chiều trong thương mại cùng các lĩnh vực khác.

- Thuật toán mạng nơron nhân tạo (Artificial Neural Network Algorithms)

Mạng nơron nhân tạo là các mô hình được lấy cảm hứng từ cấu trúc và chức năng của mạng lưới thần kinh sinh học. Nhóm thuật toán này có thể được sử dụng cho bài toán phân lớp và hồi quy với rất nhiều biến thể khác nhau cho hầu hết các vấn đề. Tuy nhiên, trong bài viết này mình chỉ trình bày các thuật toán cổ điển và phổ biến nhất.

- Thuật toán học sâu (Deep Learning Algorithms)

Thực chất Deep Learning là một bản cập nhật hiện đại cho Artificial Neural Networks nhằm khai thác khả năng tính toán của máy tính, tuy nhiên vì sự phát triển lớn mạnh của chúng nên mình tách ra thành một nhóm riêng.

Deep Learning quan tâm đến việc xây dựng các mạng thần kinh lớn hơn, phức tạp hơn nhiều, và làm sao để khai thác hiệu quả các bộ dữ liệu lớn chứa rất ít dữ liệu đã được gán nhãn.

- Nhóm thuật toán Giảm chiều dữ liệu (Dimensionality Reduction Algorithms)

Giống như các phương pháp phân cụm, giảm không gian tìm kiếm và khai thác cấu trúc vốn có trong dữ liệu nhưng theo cách không giám sát hoặc để tóm tắt hay mô tả dữ liệu sử dụng ít thông tin hơn là mục tiêu của nhóm phương pháp này.

Điều này có thể hữu ích để trực quan hóa dữ liệu hoặc đơn giản hóa dữ liệu mà sau đó có thể được sử dụng trong phương pháp học có giám sát. Nhiều trong số các phương pháp này có thể được điều chỉnh để sử dụng trong phân lớp và hồi quy.

- Thuật toán tập hợp (Ensemble Algorithms)

Ensemble methods là những phương pháp kết hợp các mô hình yếu hơn được huấn luyện độc lập và phần dự đoán của chúng sẽ được kết hợp theo một cách nào đó để đưa ra dự đoán tổng thể.

Nhóm thuật toán này khá mạnh và được nghiên cứu nhiều, đặc biệt là về cách để kết hợp các mô hình với nhau.

2.2.Tìm hiểu về học máy có giám sát.

2.2.1. Khái niệm về học có giám sát.

Học có giám sát hay còn gọi là học có thầy là thuật toán dự đoán nhãn (label)/đầu ra (output) của một dữ liệu mới dựa trên tập dữ liệu huấn luyện mà trong đó mỗi mẫu dữ liệu đều đã được gán nhãn như minh hoạ ở Hình 1. Khi đó, thông qua một quá trình huấn luyện, một mô hình sẽ được xây dựng để cho ra các dự đoán và khi các dự đoán bị sai thì mô hình này sẽ được tinh chỉnh lại. Việc huấn luyện sẽ tiếp tục cho đến khi mô hình đạt được mức độ chính xác mong muốn trên dữ liệu huấn luyện. Điều này cũng giống như khi chúng ta đi học trên lớp, ta biết câu trả lời chính xác từ giáo viên (tập dữ liệu có nhãn) và từ đó ta sẽ sửa chữa nếu làm sai. Học có giám sát là nhóm phổ biến nhất trong các thuật toán ML.

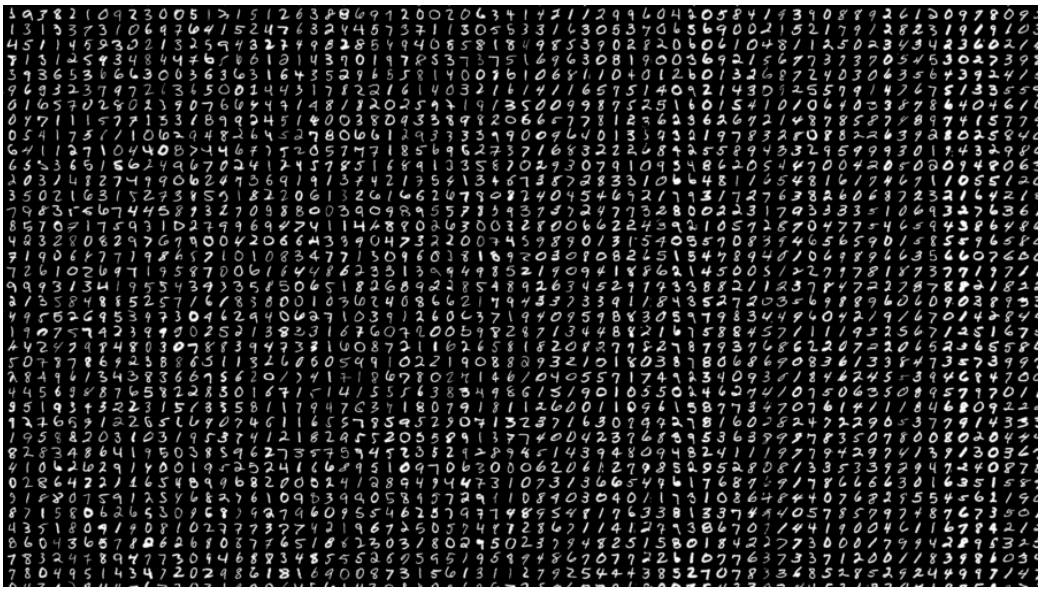
Một cách toán học, học có giám sát là khi chúng ra có một tập hợp biến đầu vào $ X={x_1,x_2,…,x_N} $ và một tập hợp nhãn tương ứng $ Y={y_1,y_2,…,y_N} $, trong đó $ x_i$, $y_i $ là các vector. Các cặp dữ liệu biết trước $(x_i, y_i) \in X \times Y $ được gọi là tập dữ liệu huấn luyện (training data). Từ tập dữ liệu huấn luyện này, chúng ta cần tạo ra một hàm số ánh xạ mỗi phần tử từ tập X sang một phần tử (xấp xỉ) tương ứng của tập Y:

y_i \approx f(x_i), \forall i=1, 2, …, N$yi≈f(xi),∀i=1,2,…,N$

Mục đích là xấp xỉ hàm số f thật tốt để khi có một dữ liệu x mới, chúng ta có thể tính được nhãn tương ứng của nó $y=f(x)$.

Ví dụ: Trong nhận dạng chữ số viết tay, ta có ảnh của hàng nghìn trường hợp ứng với mỗi chữ số được viết bởi nhiều người khác nhau. Ta đưa các bức ảnh này vào một thuật toán học và chỉ cho nó biết “mỗi bức ảnh tương ứng với chữ số nào”. Sau khi thuật toán tạo ra một mô hình, tức là một hàm số nhận đầu vào là một bức

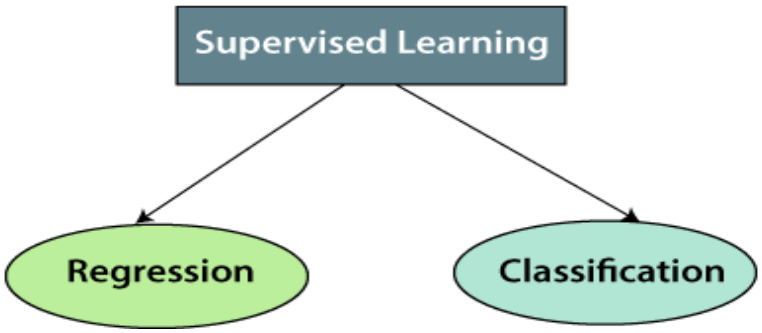
ảnh và cho ra kết quả là một chữ số. Khi nhận được một bức ảnh mới mà mô hình "chưa từng gặp qua" và nó sẽ dự đoán xem bức ảnh đó tương ứng với chữ số nào.



Hình 8:Ảnh minh hoạ cho tập dữ liệu chữ số viết tay – MNIST.

Đối với những ai sử dụng mạng xã hội Facebook thì khá quen thuộc với tính năng phát hiện khuôn mặt trong một bức ảnh, bản chất của thuật toán dò tìm các khuôn mặt này là một thuật toán học có giám sát với tập huấn luyện là vô số ảnh đã được gán nhãn là mặt người hay không phải mặt người.

Các thuật toán học có giám sát còn được phân ra thành hai loại chính là phân lớp (Classification) và hồi quy (Regression).



Hình 9: Hình ảnh phân loại của học máy có giám sát.

2.2.2. Phân loại :

Classification (phân loại): Một bài toán được gọi là classification nếu các label của input data được chia thành một số hữu hạn nhóm. Ví dụ: Gmail xác định xem một email có phải là spam hay không; các hãng tín dụng xác định xem một khách hàng có khả năng thanh toán nợ hay không. Ba ví dụ phía trên được chia vào loại này.

Regression (hồi quy): Nếu label không được chia thành các nhóm mà là một giá trị thực cụ thể. Ví dụ: một căn nhà rộng x m2x m2, có yy phòng ngủ và cách trung tâm thành phố z kmz km sẽ có giá là bao nhiêu?

Gần đây Microsoft có một ứng dụng dự đoán giới tính và tuổi dựa trên khuôn mặt. Phần dự đoán giới tính có thể coi là thuật toán Classification, phần dự đoán tuổi có thể coi là thuật toán Regression. Chú ý rằng phần dự đoán tuổi cũng có thể coi là Classification nếu ta coi tuổi là một số nguyên dương không lớn hơn 150, chúng ta sẽ có 150 class (lớp) khác nhau.

CHƯƠNG 3 : TIỀN XỬ LÍ DỮ LIỆU

Tiền xử lý dữ liệu là một bước rất quan trọng trong việc giải quyết bất kỳ vấn đề nào trong lĩnh vực Học Máy. Hầu hết các bộ dữ liệu được sử dụng trong các vấn đề liên quan đến Học Máy cần được xử lý, làm sạch và biến đổi trước khi một thuật toán Học Máy có thể được huấn luyện trên những bộ dữ liệu này. Các kỹ thuật tiền xử lý dữ liệu phổ biến hiện nay bao gồm: xử lý dữ liệu bị khuyết (missing data), mã hóa các biến nhóm (encoding categorical variables), chuẩn hóa dữ liệu (standardizing data), co giãn dữ liệu (scaling data),… Những kỹ thuật này tương đối dễ hiểu nhưng sẽ có nhiều vấn đề phát sinh khi chúng ta áp dụng vào các dữ liệu thực tế. Bởi lẽ các bộ dữ liệu ứng với các bài toán trong thực tế rất khác nhau và mỗi bài toàn thì đối mặt với những thách thức khác nhau về mặt dữ liệu.

3.1 Phương pháp tiền xử lí dữ liệu

3.1.1. Làm sạch dữ liệu

Trong thực tế, dữ liệu có thể không được thu thập trực tiếp bởi con người vì các lý do xoay quanh vấn đề về chi phí, cơ sở hạ tầng, con người. Do đó, dữ liệu có thể bị thiếu bởi một sai sót của máy móc, hoặc thực tế nó không tồn tại tại một thời điểm nhất định trong khi thu thập dữ liệu. Qua nghiên cứu, người ta xác định được rằng các dữ liệu bị khuyết bởi các nguyên nhân sau đây:

- Khuyết ngẫn nhiên (Missing at Random – MAR): Khuyết ngẫn nhiên nghĩa là xu hướng giá trị của một đặc trưng bị khuyết không liên quan đến tính chất của đặc trưng đó nhưng liên quan đến một vài đặc trưng không bị khuyết khác. Nói cách khác, sự khuyết dữ liệu của đặc trưng này có điều kiện hoặc phụ thuộc vào một hoặc một vài các đặc trưng khác.

- Khuyết hoàn toàn ngẫu nhiên (Missing Completely at Random – MCAR): Khuyết hoàn toàn ngẫu nhiên nghĩa là xu hướng bị khuyết của một đặc trưng là hoàn toàn ngẫu nhiên. Không có mối quan hệ nào giữa đặc trưng bị khuyết với các giá trị giả định hoặc các ràng buộc trên các đặc trưng khác. Ở đây, tập dữ liệu bị khuyết chỉ là một tập con ngẫu nhiên của bộ dữ liệu.

- Khuyết không ngẫu nhiên (Missing not at Random – MNAR): Khuyết không ngẫu nhiên xảy ra khi một điểm dữ liệu bị khuyết phụ thuộc cả vào các giá

trị giả định (ví dụ như những người giàu có thường không tiết lộ mức thu nhập của họ khi bạn khảo sát) và các giá trị của các đặc trưng khác (ví dụ như khi bạn muốn khảo sát tuổi của một người, mà người đó là con gái thì thường là bạn sẽ không nhận được câu trả lời từ họ).

Trong hai trường hợp đầu tiên, xóa đi các điểm dữ liệu bị thiếu dựa vào số lần xuất hiện của chúng là chấp nhận được. Nhưng trong trường hợp thứ ba, việc xóa đi các quan sát bị khuyết giá trị có thể khiến cho mô hình bị ảnh hưởng. Do đó, chúng ta cần rất lưu tâm trước khi xóa đi những điểm dữ liệu này.

Ngoài ra, chúng ta cũng cần hiểu rằng cách bỏ qua (không xử lý gì) những điểm dữ liệu bị khuyết này là cách không nên áp dụng, nó có thể rất tai hại nếu như bạn không xử lý đúng đắn khi phân tích dữ liệu của bạn. Bởi lẽ dữ liệu bị khuyết sẽ khiến bạn đưa ra những kết luận sai lầm về bộ dữ liệu của bạn khi bạn nhìn vào các giá trị sai về tổng, trung bình hoặc phân phối của bộ dữ liệu.

3.1.1.1/ Xử lý missing data

Missing data được định nghĩa là giá trị dữ liệu của một phần tử trong dataframe bị khuyết thiếu Missing data xảy ra rất nhiều trong thực tế. Việc xuất hiện missing data sẽ làm ảnh hưởng đến kết quả quá trình phân tích dữ liệu

Dữ liệu bị thiếu(missing data) là dữ liệu không có giá trị, thường được bỏ trống hoặc kí hiệu là NA (non-available).

Ví dụ: Chúng ta có tập dữ liệu về chiều cao cân nặng của các học sinh trong lớp học, nhưng do học sinh A không nhớ chiều cao của mình, nên ô chiều cao này bị bỏ trống, đó là missing data

Để xử lý dữ liệu missing, ta cần phải hiểu về tập dữ liệu. Việc lựa chọn phương pháp xử lý phụ thuộc vào từng bài toán cụ thể.

Có nhiều cách để xử lí missing value:

Cách 1: Xóa thuộc tính chứa giá trị thiếu. Phương pháp này khá tệ, vì có thể trên cột này chỉ chứa 1 hay vài giá trị bị thiếu, nếu bỏ hẳn cột thì khá lãng phí vì nó còn nó còn chứa nhiều thông tin hữu ích khác.

Cách 2: Thay thế các missing data bằng các giá trị khác

- Thay thế bằng một giá trị cố định.
- Thay thế bằng giá trị liền trước.
- Thay thế bằng giá trị liền sau.
- Thay thế giá trị bằng nội suy.

3.1.1.2/ Dữ liệu bị nhiễu (noisy data)

Nhiễu là lỗi ngẫu nhiên, nó có phương sai khác không so với dữ liệu chuẩn. Ta nên phân biệt giữa ngoại lai (outlier) và dữ liệu sai (wrong data). Ngoại lai là những dữ liệu nằm ngoài giá trị của thuộc tính còn dữ liệu sai là dữ liệu không đúng sự thật. Ví dụ: chiều cao thuộc mức cao là 170-180 cm, nhưng xuất hiện giá trị 182cm đó là giá trị ngoại lai. Dữ liệu sai khi người đó thực tế đo được chiều cao là 165cm, nhưng tập dữ liệu lại được nhập vào là 170cm.

Cách xử lí:

- Phân khoảng (binning): Sắp xếp dữ liệu và phân chia vào các khoảng (bin) có tần số xuất hiện như nhau. Sau đó, mỗi khoảng dữ liệu có thể được biểu diễn bằng trung bình trung bị, hoặc các giới hạn… của các giá trị trong khoảng đó.
- Hồi quy: sử dụng một hàm hồi quy để phân lớp dữ liệu. Ví dụ có 2 lớp ta có thể dùng Linear regression, nhiều hơn 2 lớp ta dùng Multiple linear regression.
- Phân tích ngoại lai: Ngoại lai có thể được phát hiện bằng việc gom cụm. Những ngoại lai gần giống nhau ta cho vào một cụm.

3.1.2. Tích hợp dữ liệu (data integration)

Tích hợp dữ liệu là quá trình gộp dữ liệu từ nhiều nguồn khác nhau vào một kho dữ liệu có sẵn cho quá trình khai phá dữ liệu. Những vấn đề xảy ra trong quá trình tích hợp dữ liệu: Nhận diện thực thể, dư thừa dữ liệu, mâu thuẫn dữ liệu.

3.1.3. Giảm chiều dữ liệu/ tinh gọn dữ liệu (data reduction)

Giảm chiều dữ liệu (tiếng Anh: dimensionality reduction, hay dimension reduction), là sự biến đổi dữ liệu từ không gian chiều-cao thành không gian chiều-thấp để biểu diễn ở dạng chiều-thấp đồng thời giữ lại một số thuộc tính có ý nghĩa của dữ liệu gốc, có ý tưởng là gần với chiều nội tại (intrinsic dimension).

Phân tích dữ liệu trong không gian chiều-cao có thể khó khăn vì nhiều lý do; dữ liệu thô thường có tính thưa thớt (sparse matrix) là một hậu quả của lời nguyền chiều, và do đó việc phân tích thường khó tính toán; hơn nữa các thuật toán có thể mất rất nhiều thời gian để xử lý dữ liệu. Giảm chiều dữ liệu là phổ biến trong các lĩnh vực có số lượng quan sát lớn và/hoặc số lượng biến lớn, chẳng hạn như xử lý tín hiệu, nhận dạng tiếng nói, thông tin học thần kinh (tin học thần kinh, neuroinformatics), và tin sinh học.

Các phương pháp giảm chiều dữ liệu thông thường được chia thành cách tiếp cận tuyến tính và phi tuyến tính.Các cách tiếp cận cũng được chia thành chọn đặc tính (feature selection) và trích chọn đặc trưng (feature extraction).Giảm chiều dữ liệu có thể được sử dụng cho giảm nhiễu (noise reduction), trực quan hóa dữ liệu (data visualization), phân tích cụm, hoặc là một bước trung gian để tạo điều kiện thuận lợi cho các phân tích khác.

3.1.4. Biến đổi dữ liệu phù hợp với bài toán (data transformation)

Chuyển đổi dữ liệu là quá trình sửa đổi, tính toán, phân tách và kết hợp dữ liệu thô thành các mô hình dữ liệu sẵn sàng phân tích. Mô hình dữ liệu là những đại diện của thực tế có thể dễ dàng chuyển thành chỉ số, báo cáo và trang tổng quan để giúp người dùng hoàn thành các mục tiêu cụ thể.

Đặc biệt, các doanh nghiệp cần KPI và các thước đo khác để định lượng và hiểu được họ đang làm gì và như thế nào.

Chuyển đổi chuẩn bị dữ liệu cho một loạt các trường hợp sử dụng, bao gồm:

Analytics – Phân tích để hỗ trợ các quyết định bắt đầu với các chỉ số. Đôi khi, các chỉ số có thể được tính toán từ một nguồn duy nhất và chỉ cần một lượng biến đổi nhỏ. Những lần khác, cách duy nhất để tính toán số liệu là kết hợp dữ liệu từ nhiều nguồn và sau đó tổng hợp lại.

Máy học – Máy học là nhận dạng mẫu tự động. Các ứng dụng kinh doanh của máy học bao gồm dự báo doanh thu và lợi nhuận, mô hình dự đoán để hỗ trợ các quyết định chính, hệ thống giới thiệu sản phẩm cho khách hàng và tất cả các loại tự động hóa quy trình kinh doanh.

Tuân thủ quy định – Lưu trữ không cần thiết thông tin nhận dạng cá nhân (PII) khiến dữ liệu dễ bị lỗi bởi một loạt các vi phạm dữ liệu độc hại và ngẫu nhiên. Vi phạm dữ liệu làm tổn hại đến quyền riêng tư của dữ liệu và tạo ra các vấn đề nghiêm trọng cho cả bạn và khách hàng.

3.2. Thực hiện phân tích và tiền xử lý dữ liệu :

Bộ dữ liệu: file mushroom.csv gồm có 8125 loại tương tứng với 23 cột là các thuộc tính của nấm.Trong 23 thuộc tníh thì có 22 thuộc tính độc lập và 1 thuộc tính phụ thuộc.ộ dữ liệu này bao gồm các mô tả về các mẫu giả định tương ứng với 23 loài nấm mang trong họ Nấm Agaricus và Lepiota được rút ra từ Hướng dẫn thực địa của Hiệp hội Audubon về Nấm Bắc Mỹ (1981). Mỗi loài được xác định là chắc chắn ăn được, chắc chắn có độc, hoặc không rõ có thể ăn được và không được khuyến cáo. Lớp thứ hai này được kết hợp với lớp độc.

Các loại dữ liệu thuộc tính ban đầu :

Attribute Information: (classes: edible=e, poisonous=p)

cap-shape:	bell=b, conical=c, convex=x, flat=f, knobbed=k, sunken=s
cap-surface:	fibrous=f, grooves=g, scaly=y, smooth=s
cap-color:	brown=n, buff=b, cinnamon=c, gray=g, green=r, pink=p, purple=u, red=e, white=w, yellow=y
bruises:	bruises=t, no=f
odor:	almond=a, anise=l, creosote=c, fishy=y, foul=f, musty=m, none=n, pungent=p, spicy=s
gill-attachment:	attached=a, descending=d, free=f, notched=n
gill-spacing:	close=c, crowded=w, distant=d
gill-size:	broad=b, narrow=n
gill-color:	black=k, brown=n, buff=b, chocolate=h, gray=g, green=r, orange=o, pink=p, purple=u, red=e, white=w, yellow=y
stalk-shape:	enlarging=e, tapering=t
stalk-root:	bulbous=b, club=c, cup=u, equal=e, rhizomorphs=z, rooted=r, missing=?
stalk-surface-above-ring:	fibrous=f, scaly=y, silky=k, smooth=s
stalk-surface-below-ring:	fibrous=f, scaly=y, silky=k, smooth=s
stalk-color-above-ring:	brown=n, buff=b, cinnamon=c, gray=g, orange=o, pink=p, red=e, white=w, yellow=y
stalk-color-below-ring:	brown=n, buff=b, cinnamon=c, gray=g, orange=o, pink=p, red=e, white=w, yellow=y
veil-type:	partial=p,universal=u
veil-color:	brown=n, orange=o, white=w, yellow=y

ring-number:	none=n, one=o, two=t
ring-type:	cobwebby=c, evanescent=e, flaring=f, large=l, none=n, pendant=p, sheathing=s, zone=z
spore-print-color:	black=k, brown=n, buff=b, chocolate=h, green=r, orange=o, purple=u, white=w, yellow=y
population:	abundant=a, clustered=c, numerous=n, scattered=s, several=v, solitary=y
habitat:	grasses=g, leaves=l, meadows=m, paths=p, urban=u, waste=w, woods=d

Dữ liêu khi được dịch sang tiếng Việt cho dễ đọc hiểu các thuộc tính:

1. hình dạng nắp:	chuông = b, hình nón = c, lồi = x, phẳng = f, núm = k, trũng = s
2. bề mặt nắp:	dạng sợi = f, rãnh = g, vảy = y, nhẵn = s
3. màu nắp:	nâu = n, buff = b, quế = c, xám = g, xanh lá cây = r, hồng = p, tím = u, đỏ = e, trắng = w, vàng = y
4. bầm tím :	bầm = t, không = f
5. mùi:	hạnh = a, hồi = l, creosote = c, tanh = y, hôi = f, mốc = m, không = n, hăng = p, cay = s
6. mang- đính kèm:	đính kèm = a, giảm dần = d, tự do = f, khía = n
7. khoảng cách mang:	gần = c, đông đúc = w, xa = d
8. kích thước mang:	rộng = b, hẹp = n
9. màu mang:	đen = k, nâu = n, da bò = b, socola = h, xám = g, xanh lục = r, cam = o, hồng = p, tím = u, đỏ = e, trắng = w, vàng = y
10. cuống-hình:	to ra = e, thon = t

11. cuống-rễ: củ = b, chùy = c, cốc = u, bằng = e, hình rễ = z, rễ = r, khuyết =?
12. cuống-bề mặt-trên-vòng: sợi = f, vảy = y, mượt = k, mịn = s
13. cuống-bề mặt-dưới-vòng: sợi = f, vảy = y, mượt = k, mịn = s
14. cuống-màu-trên-vòng: nâu = n, buff = b, quế = c, xám = g, cam = o, hồng = p, đỏ = e, trắng = w, vàng = y
15. màu cuống -vòng dưới: nâu = n, buff = b, quế = c, xám = g, cam = o, hồng = p, đỏ = e, trắng = w, vàng = y
16. loại màn: một phần = p, phổ quát = u
17. màu màn che: nâu = n, da cam = o, trắng = w, vàng = y
18. số vòng: không = n, một = o, hai = t
19. kiểu vòng: cobwebby = c, evanescent = e, loe = f, lớn = l, không có = n, mặt dây chuyền = p, vỏ bọc = s, vùng = z
20. bào tử-in-màu: black = k, brown = n, buff = b, chocolate = h, green = r, cam = o, tím = u, trắng = w, vàng = y
21. dân số: phong phú = a, nhóm = c, nhiều = n, rải rác = s, vài = v, đơn độc = y
22. môi trường sống: cỏ = g, lá = l, đồng cỏ = m, lối đi = p, thành thị = u, chất thải = w, rừng = d
23. các lớp: ăn được = e, độc = p

Khai báo thư viện sử dụng:

Khai báo thư viện Pandas: import pandas as pd

Pandas có 2 cấu trúc dữ liệu cơ bản là:

+ Series(1 chiều): Series có nhiều thuộc tính như *index, array, values, dtype*, v.v.

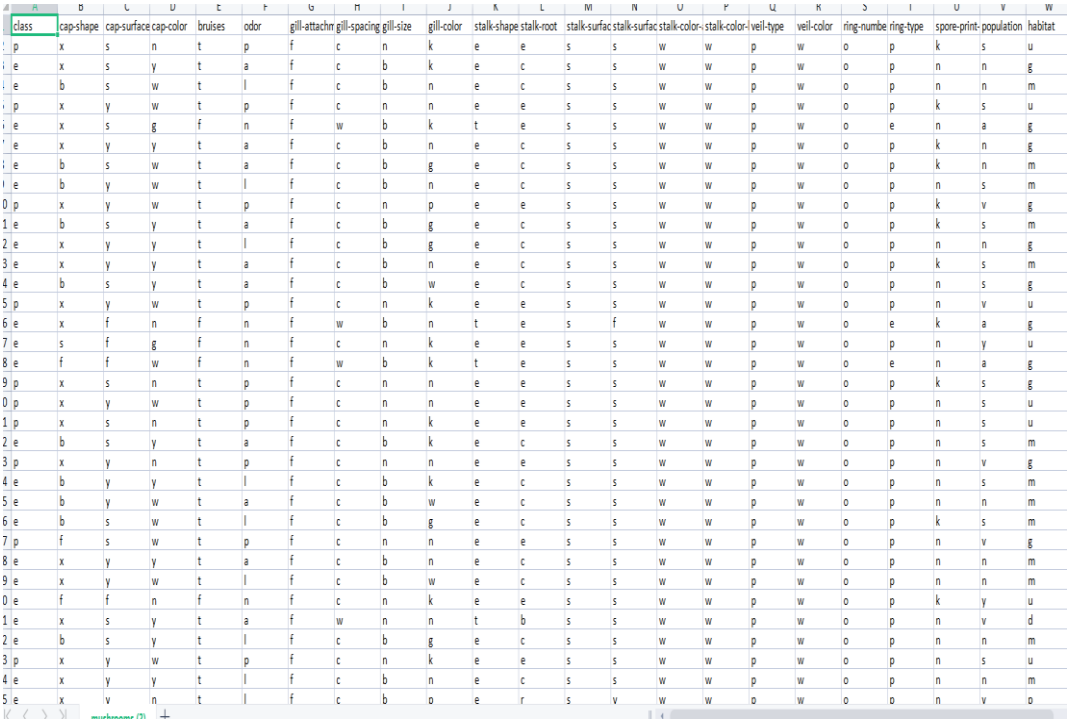
+ DataFrame(2 chiều): Dataframe là cấu trúc dữ liệu được gắn nhãn hai chiều với các cột và hàng như bảng tính (spreadsheet) hoặc bảng (table).

Thực hiện loại bỏ một số thuộc tính dư thừa:

Xóa bỏ mốt số thuộc tính dư thừa trong pandas ta có câu lệnh:

```
f.drop(columns=['B', 'C']) (Xóa các cột có tên là B và C)
```

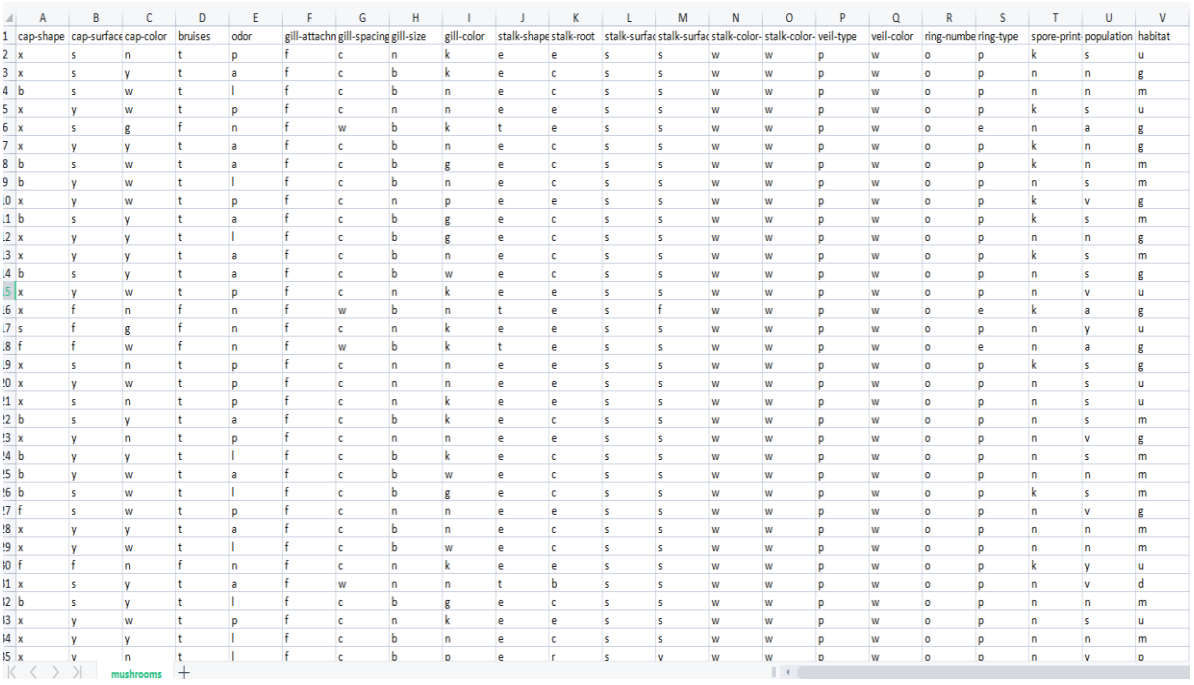
Quan sát dữ liệu sau khi bị loại bỏ một số thuộc tính:



class	cap-shape	cap-surface	cap-color	bruises	odor	gill-attachm	gill-spacing	gill-size	gill-color	stalk-shape	stalk-root	stalk-surfac	stalk-surfac	stalk-color-	stalk-color-	veil-type	veil-color	ring-numbe	ring-type	spore-print-	population	habitat
p	x	s	n	t	p	f	c	n	k	e	e	s	s	w	w	p	w	o	p	k	s	u
e	x	s	y	t	a	f	c	b	k	e	c	s	s	w	w	p	w	o	p	n	n	g
e	b	s	w	t	l	f	c	b	n	e	c	s	s	w	w	p	w	o	p	n	n	m
p	x	y	w	t	p	f	c	n	n	e	e	s	s	w	w	p	w	o	p	k	s	u
e	x	s	g	f	n	f	w	b	k	t	e	s	s	w	w	p	w	o	e	n	a	g
e	x	y	y	t	a	f	c	b	n	e	c	s	s	w	w	p	w	o	p	k	n	g
e	b	s	w	t	a	f	c	b	g	e	c	s	s	w	w	p	w	o	p	k	n	m
e	b	y	w	t	l	f	c	b	n	e	c	s	s	w	w	p	w	o	p	n	s	m
p	x	y	w	t	p	f	c	n	p	e	e	s	s	w	w	p	w	o	p	k	v	g
e	b	s	y	t	a	f	c	b	g	e	c	s	s	w	w	p	w	o	p	k	s	m
e	x	y	y	t	l	f	c	b	g	e	c	s	s	w	w	p	w	o	p	n	n	g
e	x	y	y	t	a	f	c	b	n	e	c	s	s	w	w	p	w	o	p	k	s	m
e	b	s	y	t	a	f	c	b	w	e	c	s	s	w	w	p	w	o	p	n	s	g
p	x	y	w	t	p	f	c	n	k	e	e	s	s	w	w	p	w	o	p	n	v	u
e	x	f	n	f	n	f	w	b	n	t	e	s	f	w	w	p	w	o	e	k	a	g
e	s	f	g	f	n	f	c	n	k	e	e	s	s	w	w	p	w	o	p	n	y	u
e	f	f	w	f	n	f	w	b	k	t	e	s	s	w	w	p	w	o	e	n	a	g
p	x	s	n	t	p	f	c	n	n	e	e	s	s	w	w	p	w	o	p	k	s	g
p	x	y	w	t	p	f	c	n	n	e	e	s	s	w	w	p	w	o	p	n	s	u
p	x	s	n	t	p	f	c	n	k	e	e	s	s	w	w	p	w	o	p	n	s	u
e	b	s	y	t	a	f	c	b	k	e	c	s	s	w	w	p	w	o	p	n	s	m
p	x	y	n	t	p	f	c	n	n	e	e	s	s	w	w	p	w	o	p	n	v	g
e	b	y	y	t	l	f	c	b	k	e	c	s	s	w	w	p	w	o	p	n	s	m
e	b	y	w	t	a	f	c	b	w	e	c	s	s	w	w	p	w	o	p	n	n	m
e	b	s	w	t	l	f	c	b	g	e	c	s	s	w	w	p	w	o	p	k	s	m
p	f	s	w	t	p	f	c	n	n	e	e	s	s	w	w	p	w	o	p	n	v	g
e	x	y	y	t	a	f	c	b	n	e	c	s	s	w	w	p	w	o	p	n	n	m
e	x	y	w	t	l	f	c	b	w	e	c	s	s	w	w	p	w	o	p	n	n	m
e	f	f	n	f	n	f	c	n	k	e	e	s	s	w	w	p	w	o	p	k	y	u
e	x	s	y	t	a	f	w	n	n	t	b	s	s	w	w	p	w	o	p	n	v	d
e	b	s	y	t	l	f	c	b	g	e	c	s	s	w	w	p	w	o	p	n	n	m
p	x	y	w	t	p	f	c	n	k	e	e	s	s	w	w	p	w	o	p	n	s	u
e	x	y	y	t	l	f	c	b	n	e	c	s	s	w	w	p	w	o	p	n	n	m
e	x	v	n	t	l	f	c	b	o	e	r	s	v	w	w	p	w	o	o	n	v	o

Hình 10: Hình ảnh trước khi bị loại bỏ thuộc tính.

Sau khi bị loại bỏ 1 thuộc tính phụ thuộc ta đã có bản dữ liệu mới:



cap-shape	cap-surface	cap-color	bruises	odor	gill-attachm	gill-spacing	gill-size	gill-color	stalk-shape	stalk-root	stalk-surfac	stalk-surfac	stalk-color-	stalk-color-	veil-type	veil-color	ring-numbe	ring-type	spore-print	population	habitat
x	s	n	t	p	f	c	n	k	e	e	s	s	w	w	p	w	o	p	k	s	u
x	s	y	t	a	f	c	b	k	e	c	s	s	w	w	p	w	o	p	n	n	g
b	s	w	t	l	f	c	b	n	e	c	s	s	w	w	p	w	o	p	n	n	m
x	y	w	t	p	f	c	n	n	e	e	s	s	w	w	p	w	o	p	k	s	u
x	s	g	f	n	f	w	b	k	t	e	s	s	w	w	p	w	o	e	n	a	g
x	y	y	t	a	f	c	b	n	e	c	s	s	w	w	p	w	o	p	k	n	g
b	s	w	t	a	f	c	b	g	e	c	s	s	w	w	p	w	o	p	k	n	m
b	y	w	t	l	f	c	b	n	e	c	s	s	w	w	p	w	o	p	n	s	m
x	y	w	t	p	f	c	n	p	e	e	s	s	w	w	p	w	o	p	k	v	g
b	s	y	t	a	f	c	b	g	e	c	s	s	w	w	p	w	o	p	k	s	m
x	y	y	t	l	f	c	b	g	e	c	s	s	w	w	p	w	o	p	n	n	g
x	y	y	t	a	f	c	b	n	e	c	s	s	w	w	p	w	o	p	k	s	m
b	s	y	t	a	f	c	b	w	e	c	s	s	w	w	p	w	o	p	n	s	g
x	y	w	t	p	f	c	n	k	e	e	s	s	w	w	p	w	o	p	n	v	u
x	f	n	f	n	f	w	b	n	t	e	s	f	w	w	p	w	o	e	k	a	g
s	f	g	f	n	f	c	n	k	e	e	s	s	w	w	p	w	o	p	n	y	u
f	f	w	f	n	f	w	b	k	t	e	s	s	w	w	p	w	o	e	n	a	g
x	s	n	t	p	f	c	n	n	e	e	s	s	w	w	p	w	o	p	k	s	g
x	y	w	t	p	f	c	n	n	e	e	s	s	w	w	p	w	o	p	n	s	u
x	s	n	t	p	f	c	n	k	e	e	s	s	w	w	p	w	o	p	n	s	u
b	s	y	t	a	f	c	b	k	e	c	s	s	w	w	p	w	o	p	n	s	m
x	y	n	t	p	f	c	n	n	e	e	s	s	w	w	p	w	o	p	n	v	g
b	y	y	t	l	f	c	b	k	e	c	s	s	w	w	p	w	o	p	n	s	m
b	y	w	t	a	f	c	b	w	e	c	s	s	w	w	p	w	o	p	n	s	m
b	s	w	t	l	f	c	b	g	e	c	s	s	w	w	p	w	o	p	k	s	m
f	s	w	t	p	f	c	n	n	e	e	s	s	w	w	p	w	o	p	n	v	g
x	y	y	t	a	f	c	b	n	e	c	s	s	w	w	p	w	o	p	n	n	m
x	y	w	t	l	f	c	b	w	e	c	s	s	w	w	p	w	o	p	n	n	m
f	f	n	f	n	f	c	n	k	e	e	s	s	w	w	p	w	o	p	k	y	u
x	s	y	t	a	f	w	n	n	t	b	s	s	w	w	p	w	o	p	n	v	d
b	s	y	t	l	f	c	b	g	e	c	s	s	w	w	p	w	o	p	n	n	m
x	y	w	t	p	f	c	n	k	e	e	s	s	w	w	p	w	o	p	n	s	u
x	y	y	t	l	f	c	b	n	e	c	s	s	w	w	p	w	o	p	n	n	m
x	v	n	t	l	f	c	b	o	e	r	s	v	w	w	p	w	o	o	n	v	o

Hình 11: Dữ liệu sau khi bị loại bỏ 1 thuộc tính.

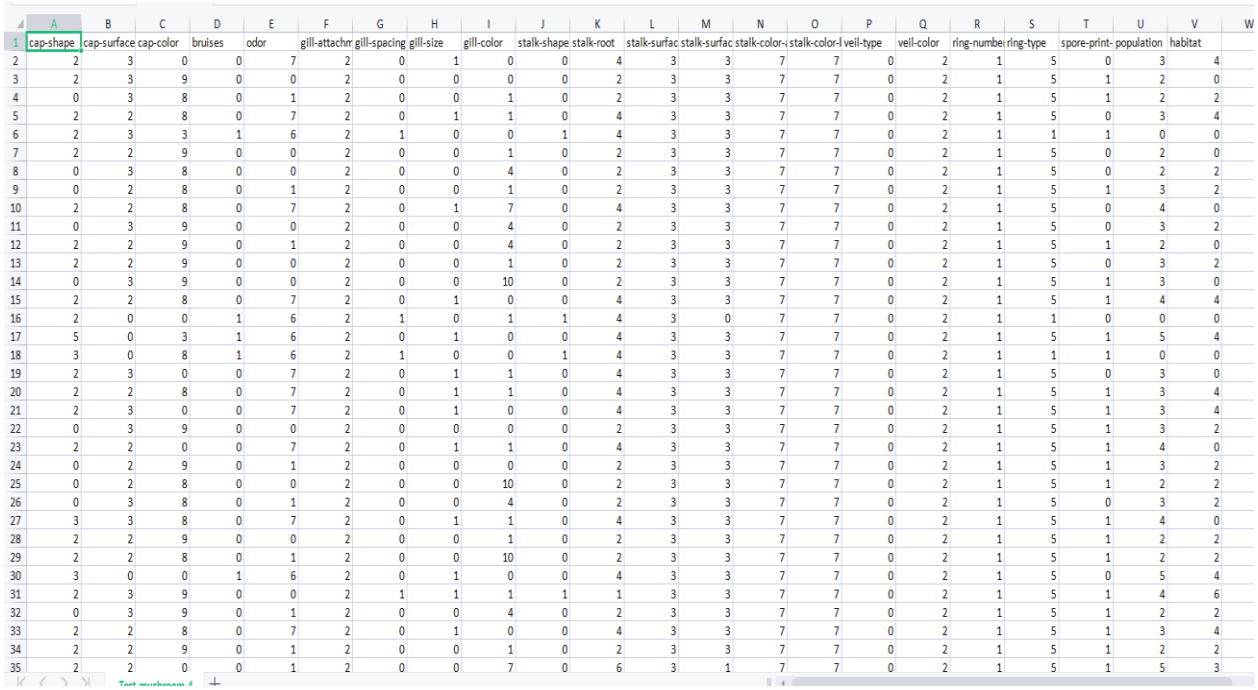
Chuyển đổi một số thuộc tính về dạng số nguyên:

- Với thuộc tính class ta đặt:	p là 1 với e là 0
- Với thuộc tính cap-shape đặt:	'b'=0, 'c'=1, 'x'=2, 'f'=3, 'k'=4, 's'=5
- Với thuộc tính cap-surfac:	'f'=0, 'g'=1, 'y'=2, 's'=3
- Với thuộc tính cap-color :	'n'=0, 'b'=1, 'c'=2, 'g'=3, 'r'=4, 'p'=5, 'u'=6, 'e'=7, 'w'=8, 'y'=9
- Với thuộc tínhbruises:	't'=0, 'f'=1
- Với thuộc tính odor:	'a'=0, 'l'=1, 'c'=2, 'y'=3, 'f'=4, 'm'=5, 'n'=6, 'p'=7, 's'=8
- Với thuộc tính gill-attachment:	'a'=0, 'd'=1, 'f'=2, 'n'=3
- Với thuộc tính gill-spacing:	'c'= 0, 'w'= 1, 'd'= 2
- Với thuộc tính gill-size:	'b'=0, 'n'=1
-Với thuộc tính gill-color:	‘k’=0, 'n'=1, 'b'=2, 'h'=3, 'g'=4, 'r'=5,' o'=6, 'p'=7, 'u'=8, 'e'=9, 'w'=10, 'y'=11
- Với thuộc tính lớp stalk-shape :	'e'=0, 't'=1
- Với thuộc tính stalk-root :	'?'=0, 'b'=1, 'c'=2, 'u'=3, 'e'=4, 'z'=5, 'r'=6
- Với thuộc tính stalk-surface-above-ring :	'f'=0,'y'=1, 'k'=2,'s'=3
- Với lớp stalk-surface-below-ring :	'f'=0,'y'=1, 'k'=2,'s'=3
- Với thuộc tính stalk-color-above-ring :	'n'=0, 'b'=1, 'c'=2, 'g'=3, 'o'=4, 'p'=5, 'e'=6, 'w'=7, 'y'=8
- Với thuộc tính stalk-color-below-ring :	'n'=0, 'b'=1, 'c'=2, 'g'=3, 'o'=4, 'p'=5, 'e'=6, 'w'=7, 'y'=8
- Với thuộc tính veil-type :	'p'=0, 'u'=1
- Với thuộc tính veil-color :	'n'=0, 'o'=1, 'w'=2, 'y'=3
- Với thuộc tính ring-number :	'n'=0, 'o'=1, 't'=2
- Với thuộc tính ring-type :	'c'=0, 'e'=1, 'f'=2, 'l'=3, 'n'=4, 'p'=5, 's'=6, 'z'=7

- Với thuộc tính spore-print-color : 'k'=0, 'n'=1, 'b'=2, 'h'=3, 'r'=4, 'o'=5, 'u'=6, 'w'=7, 'y'=8
- Với thuộc tính population : 'a'=0, 'c'=1, 'n'=2, 's'=3, 'v'=4, 'y'=5
- Với thuộc tính habitat : 'g'=0, 'l'=1, 'm'=2, 'p'=3, 'u'=4, 'w'=5, 'd'=6.

```
import pandas as pd
path = 'mushrooms.csv'
data_df = pd.read_csv(path)
data_df['class'] = data_df['class'].map({'p':1, 'e':0}).astype(int)
data_df['cap-shape'] = data_df['cap-shape'].map({'b':0, 'c':1, 'x':2,'f':3,'k':4,'s':5}).astype(int)
data_df['cap-surface'] = data_df['cap-surface'].map({'f':0, 'g':1,'y':2,'s':3}).astype(int)
data_df['cap-color'] = data_df['cap-color'].map({'n':0, 'b':1,'c':2,'g':3,'r':4,'p':5,'u':6,'e':7,'w':8,'y':9}).astype(int)
data_df['bruises'] = data_df['bruises'].map({'t':0, 'f':1}).astype(int)
data_df['odor'] = data_df['odor'].map({'a':0, 'l':1,'c':2,'y':3,'f':4,'m':5,'n':6,'p':7,'s':8}).astype(int)
data_df['gill-attachment'] = data_df['gill-attachment'].map({'a':0, 'd':1,'f':2,'n':3}).astype(int)
data_df['gill-spacing'] (variable) data_df: DataFrame {'c':0, 'w':1,'d':2}).astype(int)
data_df['gill-size'] = data_df['gill-size'].map({'b':0, 'n':1}).astype(int)
data_df['gill-color'] = data_df['gill-color'].map({'k':0, 'n':1,'b':2,'h':3,'g':4,'r':5,'o':6,'p':7,'u':8,'e':9,'w':10,'y':11}).astype(int)
data_df['stalk-shape'] = data_df['stalk-shape'].map({'e':0,'t':1}).astype(int)
data_df['stalk-root'] = data_df['stalk-root'].map({'?':0,'b':1, 'c':2,'u':3,'e':4,'z':5,'r':6}).astype(int)
data_df['stalk-surface-above-ring'] = data_df['stalk-surface-above-ring'].map({'f':0,'y':1, 'k':2,'s':3}).astype(int)
data_df['stalk-surface-below-ring'] = data_df['stalk-surface-below-ring'].map({'f':0,'y':1, 'k':2,'s':3}).astype(int)
data_df['stalk-color-above-ring'] = data_df['stalk-color-above-ring'].map({'n':0,'b':1, 'c':2,'g':3,'o':4,'p':5,'e':6,'w':7,'y':8}).astype(int)
data_df['stalk-color-below-ring'] = data_df['stalk-color-below-ring'].map({'n':0,'b':1, 'c':2,'g':3,'o':4,'p':5,'e':6,'w':7,'y':8}).astype(int)
data_df['veil-type'] = data_df['veil-type'].map({'p':0, 'u':1}).astype(int)
data_df['veil-color'] = data_df['veil-color'].map({'n':0, 'o':1,'w':2,'y':3}).astype(int)
data_df['ring-number'] = data_df['ring-number'].map({'n':0, 'o':1,'t':2}).astype(int)
data_df['ring-type'] = data_df['ring-type'].map({'c':0, 'e':1,'f':2,'l':3,'n':4,'p':5,'s':6,'z':7}).astype(int)
data_df['spore-print-color'] = data_df['spore-print-color'].map({'k':0, 'n':1,'b':2,'h':3,'r':4,'o':5,'u':6,'w':7,'y':8}).astype(int)
data_df['population'] = data_df['population'].map({'a':0, 'c':1,'n':2,'s':3,'v':4,'y':5}).astype(int)
data_df['habitat'] = data_df['habitat'].map({'g':0, 'l':1,'m':2,'p':3,'u':4,'w':5,'d':6}).astype(int)
print(data_df)
```

Hình 12: Ảnh code chuyển đổi dữ liệu từ chữ thành số.



	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W
1	cap-shape	cap-surface	cap-color	bruises	odor	gill-attachm	gill-spacing	gill-size	gill-color	stalk-shape	stalk-root	stalk-surfac	stalk-surfac	stalk-color-s	stalk-color-l	veil-type	veil-color	ring-numbe	ring-type	spore-print-	population	habitat	
2	2	3	0	0	7	2	0	1	0	0	4	3	3	7	7	0	2	1	5	0	3	4	
3	2	3	9	0	0	2	0	0	0	0	2	3	3	7	7	0	2	1	5	1	2	0	
4	0	3	8	0	1	2	0	0	1	0	2	3	3	7	7	0	2	1	5	1	2	2	
5	2	2	8	0	7	2	0	1	1	0	4	3	3	7	7	0	2	1	5	0	3	4	
6	2	3	3	1	6	2	1	0	0	1	4	3	3	7	7	0	2	1	1	1	0	0	
7	2	2	9	0	0	2	0	0	1	0	2	3	3	7	7	0	2	1	5	0	2	0	
8	0	3	8	0	0	2	0	0	4	0	2	3	3	7	7	0	2	1	5	0	2	2	
9	0	2	8	0	1	2	0	0	1	0	2	3	3	7	7	0	2	1	5	1	3	2	
10	2	2	8	0	7	2	0	1	7	0	4	3	3	7	7	0	2	1	5	0	4	0	
11	0	3	9	0	0	2	0	0	4	0	2	3	3	7	7	0	2	1	5	0	3	2	
12	2	2	9	0	1	2	0	0	4	0	2	3	3	7	7	0	2	1	5	1	2	0	
13	2	2	9	0	0	2	0	0	1	0	2	3	3	7	7	0	2	1	5	0	3	2	
14	0	3	9	0	0	2	0	0	10	0	2	3	3	7	7	0	2	1	5	1	3	0	
15	2	2	8	0	7	2	0	1	0	0	4	3	3	7	7	0	2	1	5	1	4	4	
16	2	0	0	1	6	2	1	0	1	1	4	3	0	7	7	0	2	1	1	0	0	0	
17	5	0	3	1	6	2	0	1	0	0	4	3	3	7	7	0	2	1	5	1	5	4	
18	3	0	8	1	6	2	1	0	0	1	4	3	3	7	7	0	2	1	1	1	0	0	
19	2	3	0	0	7	2	0	1	1	0	4	3	3	7	7	0	2	1	5	0	3	0	
20	2	2	8	0	7	2	0	1	1	0	4	3	3	7	7	0	2	1	5	1	3	4	
21	2	3	0	0	7	2	0	1	0	0	4	3	3	7	7	0	2	1	5	1	3	4	
22	0	3	9	0	0	2	0	0	0	0	2	3	3	7	7	0	2	1	5	1	3	2	
23	2	2	0	0	7	2	0	1	1	0	4	3	3	7	7	0	2	1	5	1	4	0	
24	0	2	9	0	1	2	0	0	0	0	2	3	3	7	7	0	2	1	5	1	3	2	
25	0	2	8	0	0	2	0	0	10	0	2	3	3	7	7	0	2	1	5	1	2	2	
26	0	3	8	0	1	2	0	0	4	0	2	3	3	7	7	0	2	1	5	0	3	2	
27	3	3	8	0	7	2	0	1	1	0	4	3	3	7	7	0	2	1	5	1	4	0	
28	2	2	9	0	0	2	0	0	1	0	2	3	3	7	7	0	2	1	5	1	2	2	
29	2	2	8	0	1	2	0	0	10	0	2	3	3	7	7	0	2	1	5	1	2	2	
30	3	0	0	1	6	2	0	1	0	0	4	3	3	7	7	0	2	1	5	0	5	4	
31	2	3	9	0	0	2	1	1	1	1	1	3	3	7	7	0	2	1	5	1	4	6	
32	0	3	9	0	1	2	0	0	4	0	2	3	3	7	7	0	2	1	5	1	2	2	
33	2	2	8	0	7	2	0	1	0	0	4	3	3	7	7	0	2	1	5	1	3	4	
34	2	2	9	0	1	2	0	0	1	0	2	3	3	7	7	0	2	1	5	1	2	2	
35	2	2	0	0	1	2	0	0	7	0	6	3	1	7	7	0	2	1	5	1	5	3	

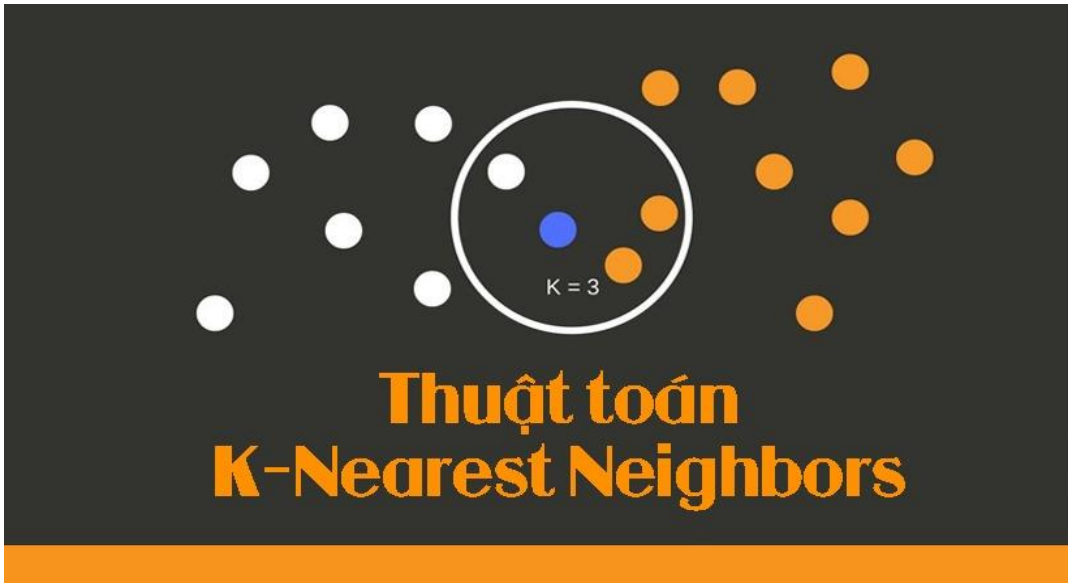
Hình 13:File dữ liệu đã được chuyển từ chữ về số.

- Xử lí các giá trị missing theo hướng nội suy:
  - Thuật toán nội suy tuyến tính (linear)
  - Hướng nội suy tiến lên

CHƯƠNG 4: THUẬT TOÁN KNN CHO BÀI TOÁN LỚP NHỊ PHÂN

4.1. Nghiên cứu về thuật toán KNN

4.1.1. Khái niệm về thuật toán



Hình 14: Thuật toán KNN.

K-nearest neighbor được giải quyết trong bài toán học có giám sát.

K-nearest neighbor là một trong những thuật toán supervised-learning đơn giản nhất (mà hiệu quả trong một vài trường hợp) trong Machine Learning. Khi training, thuật toán này không học một điều gì từ dữ liệu training (đây cũng là lý do thuật toán này được xếp vào loại lazy learning), mọi tính toán được thực hiện khi nó cần dự đoán kết quả của dữ liệu mới. K-nearest neighbor có thể áp dụng được vào cả hai loại của bài toán Supervised learning là Classification và Regression. KNN còn được gọi là một thuật toán Instance-based hay Memory-based learning.

Có một vài khái niệm tương ứng người-máy như sau:

Với KNN, trong bài toán Classification, label của một điểm dữ liệu mới (hay kết quả của câu hỏi trong bài thi) được suy ra trực tiếp từ K điểm dữ liệu gần nhất trong training set. Label của một test data có thể được quyết định bằng major voting (bầu chọn theo số phiếu) giữa các điểm gần nhất, hoặc nó có thể được suy ra bằng cách đánh trọng số khác

nhau cho mỗi trong các điểm gần nhất đó rồi suy ra label. Chi tiết sẽ được nêu trong phần tiếp theo.

Trong bài toán Regresssion, đầu ra của một điểm dữ liệu sẽ bằng chính đầu ra của điểm dữ liệu đã biết gần nhất (trong trường hợp K=1), hoặc là trung bình có trọng số của đầu ra của những điểm gần nhất, hoặc bằng một mối quan hệ dựa trên khoảng cách tới các điểm gần nhất đó.

Một cách ngắn gọn, KNN là thuật toán đi tìm đầu ra của một điểm dữ liệu mới bằng cách chỉ dựa trên thông tin của K điểm dữ liệu trong training set gần nó nhất (K-lân cận), không quan tâm đến việc có một vài điểm dữ liệu trong những điểm gần nhất này là nhiễu. Hình dưới đây là một ví dụ về KNN trong classification với K = 1.

4.1.2. Khoảng cách không gian vectơ

Trong không gian một chiều, khoảng cách giữa hai điểm là trị tuyệt đối giữa hiệu giá trị của hai điểm đó. Trong không gian nhiều chiều, khoảng cách giữa hai điểm có thể được định nghĩa bằng nhiều hàm số khác nhau, trong đó độ dài đường thẳng nối hai điểm chỉ là một trường hợp đặc biệt trong đó. Nhiều thông tin bổ ích (cho Machine Learning) có thể được tìm thấy tại Norms (chuẩn) của vector trong tab Math.

Trong không gian một chiều, việc đo khoảng cách giưa hai điểm đa rất quen thuộc: lấy trị tuyệt đối của hiệu giưa hai giá trị đó. Trong không gian hai chiều, tức mặt phẳng, chúng ta thường dùng khoảng cách Euclid để đo khoảng cách giưa hai điểm.

Việc đo khoảng cách giưa hai điểm dư liệu nhiều chiều, tức hai vector, là rất cần thiết trong Machine Learning. Chúng ta cần đánh giá xem điểm nào là điểm gần nhất của một điểm khác; chúng ta cũng cần đánh giá xem độ chính xác của việc ước lương; và trong rất nhiều ví dụ khác nữa.

Và đó chính là lý do mà khái niệm norm ra đời. Có nhiều loai norm khác nhau mà các ban sẽ thấy ở dưới đây:

Để xác định khoảng cách giưa hai vector y và z, người ta thường áp dụng một hàm số lên vector hiệu x = y−z. Một hàm số đươc dùng để đo các vector cần có một vài tính chất đặc biệt.

4.1.3. Phân tích toán học

- Tách train và test sets

Tách file mushroom.csv thành 2 file file train chiếm 80% và file test chiếm 20%

- Phương pháp đánh giá (evaluation method)

Để đánh giá độ chính xác của thuật toán KNN classifier này, chúng ta xem xem có bao nhiêu điểm trong test data được dự đoán đúng. Lấy số lượng này chia cho tổng số lượng trong tập test data sẽ ra độ chính xác. Scikit-learn cung cấp hàm số accuracy_score để thực hiện công việc này.

- Đánh trọng số cho các điểm lân cận

Cách đánh trọng số phải thoải mãn điều kiện là một điểm càng gần điểm test data thì phải được đánh trọng số càng cao (tin tưởng hơn). Cách đơn giản nhất là lấy nghịch đảo của khoảng cách này. (Trong trường hợp test data trùng với 1 điểm dữ liệu trong training data, tức khoảng cách bằng 0, ta lấy luôn label của điểm training data).

4.1.4. Ưu và nhược điểm của KNN

- Ưu điểm :

+ Thuật toán đơn giản, dễ dàng triển khai.

+ Độ phức tạp tính toán nhỏ.

+ Việc dự đoán kết quả của dữ liệu mới rất đơn giản.

- Nhược điểm:

+ KNN rất nhạy cảm với nhiễu khi K nhỏ.

+ Như đã nói, KNN là một thuật toán mà mọi tính toán đều nằm ở khâu test. Trong đó việc tính khoảng cách tới từng điểm dữ liệu trong training set sẽ tốn rất nhiều thời gian, đặc biệt là với các cơ sở dữ liệu có số chiều lớn và có nhiều điểm dữ liệu. Với K càng lớn thì độ phức tạp cũng sẽ tăng lên. Ngoài ra, việc lưu toàn bộ dữ liệu trong bộ nhớ cũng ảnh hưởng tới hiệu năng của KNN.

CHƯƠNG 5: XÂY DỰNG MÔ HÌNH HỌC MÁY

5.1. Nghiên cứu thư viện Scikit-Learn

Scikit-learning còn được biết đến với các từ đồng nghĩa như scikits.learn (đã biết trước đây) hoặc sklearn. Nó là một thư viện học máy mã nguồn mở miễn phí được sử dụng cho ngôn ngữ lập trình Python. Thư viện được phát triển bởi David Cournapeau như một dự án Google Summer Code vào năm 2007. Dự án sau đó được Matthieu Brucher tham gia vào năm 2010. Thư viện lần đầu tiên ra mắt công chúng vào tháng 2 năm 2010 và chỉ trong hai năm, tức là vào tháng 11 năm 2012, thư viện đã trở thành một trong những thư viện phổ biến nhất về học máy trên Github. Các tính năng chính của thư viện Scikit-learning bao gồm thuật toán phân loại, hồi quy và phân cụm (hỗ trợ máy vectơ, rừng ngẫu nhiên, tăng độ dốc, k-means và DBSCAN). Sklearn được thiết kế để xử lý các thư viện số và khoa học của Python như NumPy và SciPy.

5.1.1. Khái niệm về thư viện Scikit-Learn

Hình 15: Thư viện Scikit-learn.

Scikit-learn (Sklearn) là thư viện mạnh mẽ nhất dành cho các thuật toán học máy được viết trên ngôn ngữ Python. Thư viện cung cấp một tập các công cụ xử lý các bài toán machine learning và statistical modeling gồm: classification, regression, clustering, và dimensionality reduction.

Thư viện được cấp phép bản quyền chuẩn FreeBSD và chạy được trên nhiều nền tảng Linux. Scikit-learn được sử dụng như một tài liệu để học tập.

Để cài đặt scikit-learn trước tiên phải cài thư viện SciPy (Scientific Python). Những thành phần gồm:

- Numpy: Gói thư viện xử lý dãy số và ma trận nhiều chiều
- SciPy: Gói các hàm tính toán logic khoa học
- Matplotlib: Biểu diễn dữ liệu dưới dạng đồ thị 2 chiều, 3 chiều
- IPython: Notebook dùng để tương tác trực quan với Python
- SymPy: Gói thư viện các kí tự toán học
- Pandas: Xử lý, phân tích dữ liệu dưới dạng bảng

Những thư viện mở rộng của SciPy thường được đặt tên dạng SciKits. Như thư viện này là gói các lớp, hàm sử dụng trong thuật toán học máy thì được đặt tên là scikit-learn.

5.1.2. Tại sao nên sử dụng scikit-learn

- Hỗ trợ hầu hết các thuật toán của machine learning một cách đơn giản, hiệu quả mà chúng ta không cần phải mất công ngồi cài đặt lại.

- Có tài liệu hướng dẫn sử dụng

- Độ tin cậy cao do scikit-learn được xây dựng bởi các chuyên gia hàng đầu

- Có nguồn dữ liệu phong phú: iris, digit, …

Cách sử dụng thư viện scikit-learn

Trước tiên máy bạn phải được cài đặt Python rồi. Sau đó bạn có thể vào trực tiếp trang chủ của scikit-learn để xem cách cài đặt nó. Đơn giản nhất là sử dụng Anaconda

Như đã nói, sklearn được xây dựng trên NumPy và SciPy nên để sử dụng sklearn, chúng ta bắt buộc phải có 2 packages này. Tất nhiên ta nên cài đặt matplotlib, một package không thể thiếu trong scientific plotting. Cuối cùng, chúng ta cần cài thêm package pandas, phục vụ cho data wrangling và analysis.

Nếu đã có Python thì có thể cài đặt các thư viện trên sử dụng **pip**:

```
pip install numpy scipy matplotlib scikit-learn pandas
```

5.2 Xây dựng mô hình học máy:

Việc tạo ra một Mô hình ML đầy đủ nghĩa là xây dựng một mô hình mà từ dữ liệu **đầu vào**, tạo ra được thông tin chính xác ở **đầu ra**. Bạn có thể tưởng tượng rằng mô hình đó như một chiếc **hộp đen**, đầu vào được nạp và đầu ra sẽ cho kết quả - nhưng quá trình giữa hai đầu này lại rất phức tạp.

5.2.1 Gọi package sklearn :

Để sử dụng thuật toán KNN (**K-nearest neighbors)** chúng ta thực hiện import thư viện này vào chương trình như câu lệnh dưới đây:

```
#Gọi package sklearn
from sklearn.neighbors import KNeighborsClassifier
```

5.2.2 Phân tách tập huấn luyện (train) , tập kiểm thử (test data):

Để đánh giá một mô hình phân tích , khai phá dữ liệu thì việc đầu tiên và thông thường nhất mà mọi nhà nghiên cứu đều thực hiện là chia tập dữ liệu thành các phần phục vụ cho huấn luyện mô hình (trainning data) và kiểm thử mô hình (testing data). Xây dựng model thông qua việc lựa chọn thauajt toán , xác định

biến dữ liệu hay điều chỉnh tham số sao cho phù hợp với dữ liệu trainning. Sau khi hoàn thành mô hình cơ bản chúng ta sẽ tiến hành sử dụng tập test hay cong gọi là " unseen data" để thử nghiệm.

Như vậy, với bài toán này , nhóm thực hiện đánh giá mô hình Classification gồm 2 phần : phân chia bộ dữ liệu để huấn luyện và kiểm thử mô hình; sử dụng chỉ số để đánh giá độ hiệu quả.

Trong phần này, nhóm thực phân chia bộ dữ liệu để huấn luyện và kiểm thử mô hình:

- **Tách lấy 80% tập dữ liệu tập train:**

Cú pháp:

train_finish=dataframe_df[1:6000]

train_finish

```
In [4]: train_finish=dataframe_df[1:6000]
        train_finish
```

Out[4]:

	class	cap-shape	cap-surface	cap-color	bruises	odor	gill-attachment	gill-spacing	gill-size	gill-color	...	stalk-surface-below-ring	stalk-color-above-ring	stalk-color-below-ring	veil-type	veil-color	ring-number	ring-type	spore-print-color	populatio
1	0	2	3	9	0	0	2	0	0	0	...	3	7	7	0	2	1	5	1	
2	0	0	3	8	0	1	2	0	0	1	...	3	7	7	0	2	1	5	1	
3	1	2	2	8	0	7	2	0	1	1	...	3	7	7	0	2	1	5	0	
4	0	2	3	3	1	6	2	1	0	0	...	3	7	7	0	2	1	1	1	
5	0	2	2	9	0	0	2	0	0	1	...	3	7	7	0	2	1	5	0	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
5995	1	2	2	0	1	4	2	0	1	2	...	3	7	7	0	2	1	1	7	
5996	0	4	2	1	0	6	2	0	0	9	...	3	7	7	0	2	2	1	7	
5997	0	2	2	8	1	6	2	0	1	7	...	0	7	7	0	2	1	2	3	
5998	1	3	3	1	0	4	2	0	0	7	...	0	7	7	0	2	1	5	3	
5999	1	3	2	9	1	4	2	0	0	4	...	2	0	5	0	2	1	3	3	

5999 rows × 23 columns

- **Tách lấy 20% tập dữ liệu test:**

Cú pháp:

test=dataframe_df[6000:]

test_finish=test.drop("ket qua",axis=1)

test_finish

```
In [5]: test=dataframe_df[6000:]
        test_finish=test.drop("class",axis=1)
        test_finish
```

Out[5]:

	cap-shape	cap-surface	cap-color	bruises	odor	gill-attachment	gill-spacing	gill-size	gill-color	stalk-shape	...	stalk-surface-below-ring	stalk-color-above-ring	stalk-color-below-ring	veil-type	veil-color	ring-number	ring-type	spore-print-color	populati
6000	2	2	7	1	4	2	0	1	2	1	...	3	5	5	0	2	1	1	7	
6001	3	3	7	1	8	2	0	1	2	1	...	3	5	5	0	2	1	1	7	
6002	2	2	7	1	4	2	0	1	2	1	...	3	7	5	0	2	1	1	7	
6003	2	3	0	1	4	2	0	1	2	1	...	2	7	7	0	2	1	1	7	
6004	3	2	0	1	4	2	0	1	2	1	...	3	7	7	0	2	1	1	7	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
8119	4	3	0	1	6	0	0	0	11	0	...	3	4	4	0	1	1	5	2	
8120	2	3	0	1	6	0	0	0	11	0	...	3	4	4	0	0	1	5	2	
8121	3	3	0	1	6	0	0	0	1	0	...	3	4	4	0	1	1	5	2	
8122	4	2	0	1	3	2	0	1	2	1	...	2	7	7	0	2	1	1	7	
8123	2	3	0	1	6	0	0	0	11	0	...	3	4	4	0	1	1	5	5	

2124 rows × 22 columns

- **Thực hiện tách dữ liệu ra thánh các thuộc tính input(X_Train| thuộc tính outcome (Y_train):**

Cú pháp:

X_train = train_finish.drop("ket qua", axis=1)

Y_train = train_finish["ket qua"]

- **Thực hiện tách dữ liệu để lấy các thuộc tính input(X_Test):**

Cú pháp:

X_test = test_finish

print("Thông tin tập X_train: ", X_train.shape)

print("Thông tin tập Y_train: ", Y_train.shape)

print("Thông tin tập X_test: ", X_test.shape)

```
In [6]: #Tách dữ liệu ra thành các thuộc tính input (X_Train) | thuộc tính outcome (Y_train)
        X_train = train_finish.drop("class", axis=1)
        Y_train = train_finish["class"]
        #Tách dữ liệu để lấy các thuộc tính input (X_Test)
        X_test = test_finish

In [7]: print("Thông tin tập X_train: ", X_train.shape)
        print("Thông tin tập Y_train: ", Y_train.shape)
        print("Thông tin tập X_test: ", X_test.shape)
        #print(type(X_train))

        Thông tin tập X_train:  (5999, 22)
        Thông tin tập Y_train:  (5999,)
        Thông tin tập X_test:  (2124, 22)
```

- **Thực hiện lấy dữ liệu của 10 bệnh nhân đầu tiên của tập X_train và Y_train**:

Cú pháp:

print(X_train[:10])
print('-'*40)
print(Y_train[:10])

```
In [8]: print(X_train[:10])
        print('-'*40)
        print(Y_train[:10])
```

```
    cap-shape  cap-surface  cap-color  bruises  odor  gill-attachment  \
1           2            3          9        0     0                2
2           0            3          8        0     1                2
3           2            2          8        0     7                2
4           2            3          3        1     6                2
5           2            2          9        0     0                2
6           0            3          8        0     0                2
7           0            2          8        0     1                2
8           2            2          8        0     7                2
9           0            3          9        0     0                2
10          2            2          9        0     1                2

    gill-spacing  gill-size  gill-color  stalk-shape  ...  \
1              0          0           0            0  ...
2              0          0           1            0  ...
3              0          1           1            0  ...
4              1          0           0            1  ...
5              0          0           1            0  ...
6              0          0           4            0  ...
7              0          0           1            0  ...
8              0          1           7            0  ...
9              0          0           4            0  ...
10             0          0           4            0  ...

    population  habitat
1            2        0
2            2        2
3            3        4
4            0        0
5            2        0
6            2        2
7            3        2
8            4        0
9            3        2
10           2        0

[10 rows x 22 columns]
----------------------------------------
1     0
2     0
3     1
4     0
5     0
6     0
7     0
8     1
9     0
10    0
Name: class, dtype: int64

    stalk-surface-below-ring  stalk-color-above-ring  stalk-color-below-ring  \
1                          3                       7                       7
2                          3                       7                       7
3                          3                       7                       7
4                          3                       7                       7
5                          3                       7                       7
6                          3                       7                       7
7                          3                       7                       7
8                          3                       7                       7
9                          3                       7                       7
10                         3                       7                       7

    veil-type  veil-color  ring-number  ring-type  spore-print-color  \
1           0           2            1          5                  1
2           0           2            1          5                  1
3           0           2            1          5                  0
4           0           2            1          1                  1
5           0           2            1          5                  0
6           0           2            1          5                  0
7           0           2            1          5                  1
8           0           2            1          5                  0
9           0           2            1          5                  0
10          0           2            1          5                  1
```

5.2.3 *Khởi tạo biến KNN*

Ở bài toán này chúng ta sẽ đi khởi tạo một biến có tên knn sử dụng thuật toán phân lớp KNN, với láng giềng gần nhất là n = 3 bằng câu lệnh sau :

```
In [17]: # Gọi package sklearn
         # Khởi tạo biến knn sử dụng thuật toán phân lớp KNN, với n = 3 (láng giềng gần nhất)
         knn = KNeighborsClassifier(algorithm='ball_tree',n_neighbors = 3)

Out[17]: KNeighborsClassifier(algorithm='ball_tree', n_neighbors=3)
```

5.2.4 *Huấn luyện mô hình với tập dữ liệu train*

Đưa tập dữ liệu X_train (dữ liệu đầu vào) và Y_train (Kết quả) vào huấn luyện mô hình bằng câu lệnh :

```
In [ ]: #Huấn luyện mô hình với tập dữ liệu train:
        knn.fit(X_train, Y_train)
```

5.2.5 *Dự đoán kết quả với tập Test*

Dùng Tập X_test để dự đoán kết quả trên tập dữ liệu Test, xác định tỉ lệ dự đoán đúng và tỉ lệ dự đoán sai bằng câu lệnh :

```
In [19]: Y_pred_knn = knn.predict(X_test)
         print(Y_pred_knn)
```

5.2.6 *Đánh giá chính xác của mô hình trên tập dữ liệu train*

Đánh giá độ chính xác của mô hình là một là một bước giúp chúng ta biết được độ chính xác của tập dữ liệu huấn luyện đối với mô hình học máy và để xem độ chính xác của mô hình đã đạt yêu cầu chưa, hay sẽ phải sửa chữa cải tiến tập dữ liệu huấn luyện. Để đánh giá độ chính xác ta thực hiện câu lệnh sau :

```
In [12]: acc_knn = round(knn.score(X_train, Y_train) * 100, 2)
         print(knn.score(X_train, Y_train))
         print("Độ chính xác của mô hình KNN trên tập Train: ", acc_knn, "%")
         print("Các tham số của Model KNN: \n",knn)

         1.0
         Độ chính xác của mô hình KNN trên tập Train:  100.0 %
         Các tham số của Model KNN:
          KNeighborsClassifier(algorithm='ball_tree', n_neighbors=3)
```

CHƯƠNG 6: KẾT LUẬN

Sau một thời gian nghiên cứu về "ứng dụng của học máy để xác định nấm độc hay không độc" với nội dung:

1. Giới thiệu đề tài
2. Tổng quan về học máy
3. Tiền xử lý dữ liệu
4. Thuật toán KNN cho bài toán phân lớp nhị phân
5. Xây dựng mô hình học máy

Nhóm đã thực hiện được mục tiêu đã đưa ra : Sử dụng ngôn ngữ lập trình Python và các thư viện Pandas tiền xử lý dữ liệu đã có và thư viện Scikit-learn và xây dựng được mô hình học máy với thuật toán KNN (K-Nearest Neighbor) xác định nấm độc hay không độc . Đánh giá kết quả thu được của mô hình học máy so với kết quả thực tế.

Sau khi hoàn thành bài nghiên cứu này, nhóm đã hiểu rõ hơn về thư viện SK-learn dành cho học máy,các mô hình có thể áp dụng cho một bài toán thực tế, từ đó , có thể hiểu rõ hơn quy trình để làm một đề tài nghiên cứu về khoa học dữ liệu.

Qua đây, các thành viên trong nhóm có thêm được kỹ năng làm việc nhóm, kiên trì giải quyết một vấn đề và hiểu biết thêm về vấn đề liên quan đến cây nấm. Trong thời gian tới, nhóm sẽ tiếp tục nghiên cứu và bổ sung về phần học máy này một cách tối ưu hơn nữa. Từ đây,nhóm sẽ có thêm nhiều kiến thức thực tế trong quá trình học tập trong nhà trường, bắt kịp xu thế của xã hội và cách mạng CN 4.0.