TRƯỜNG ĐẠI HỌC SÀI GÒN

KHOA CÔNG NGHỆ THÔNG TIN

BÁO CÁO

Khai Phá Dữ Liệu

Đề tài: K-Nearest Neighbors

Giảng viên:	Lê Minh Nhựt Triều	
Thành viên:	Nguyễn Doãn Hiện	*3116410037*
	Phan Anh Trúc	*3116410131*

Hồ Chí Mình, 06/2020

MỤC LỤC

1. LỜI MỞ ĐẦU

1.1.DANH MỤC TỪ VIẾT TẮT

STT	Từ viết tắt	Ý nghĩa
1	AI	Artificial Intelligent
2	ML	Machine Learning
3	SVM	Support Vector Machine
4	KNN	K-Nearest Neighbors
5	IOT	Internet Of Thing
6	DM	Data Mining

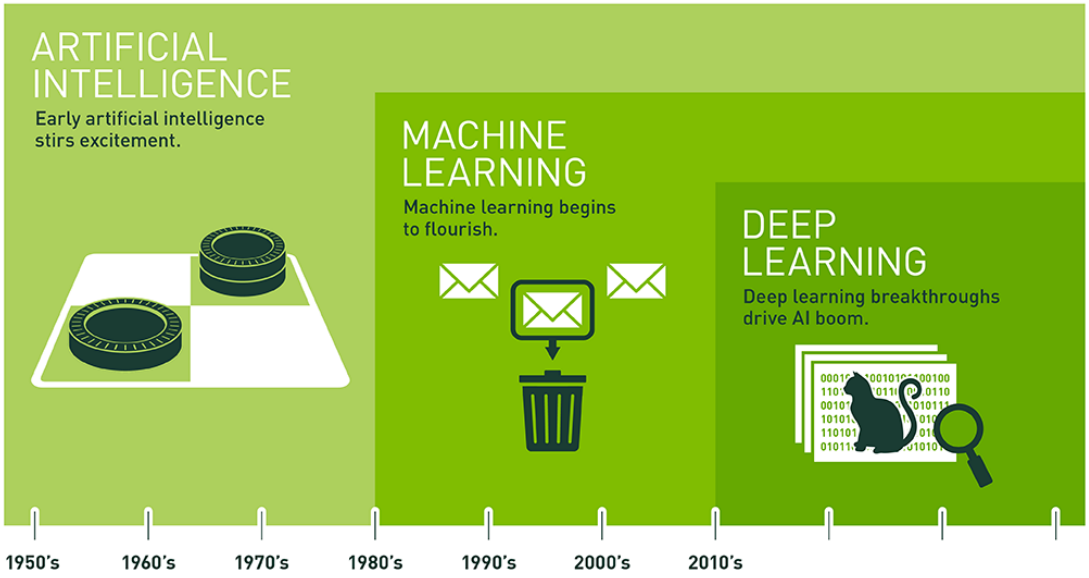
1.2.ĐẶT VẤN ĐỀ

Ngày nay, AI(hay trí tuệ nhân tạo) ngày càng phổ biến hơn trong đời sống của trong ta, số lượng người sử dụng các thiết bị thông minh như smartphone, tablet hay PC, laptop có kết nối Internet để tìm kiếm thông tin, giải trí, trò chuyện, mua sắm,… trên toàn thế giới đang gia tăng với tốc độ tên lửa. Ngoài ra sự xuất hiện của thuật ngữ I.o.T (Internet of Things) miêu tả sự kết nối giữa tất cả các thiết bị với nhau bằng Internet, cho phép trao đổi, truyền tải dữ liệu. I.o.T hỗ trợ con người rất nhiều lĩnh vực không chỉ là vấn đề sinh hoạt trong cuộc sống hàng ngày mà cả công nghiệp, nông nghiệp, bán lẻ đến y tế, xã hội. Các công ty cũng ứng dụng công nghệ I.o.T trong các hoạt động kinh doanh, sản xuất với mục đích tìm kiếm cơ hội gia tăng lợi nhuận, phát hiện sớm các rủi ro. Để ứng dụng công nghệ tiên tiến đó cùng với tốc độ dữ liệu ngày càng lớn dần thì Data Mining – khai phá dữ liệu ngày càng trở nên quan trọng hơn bởi hầu hết mọi lĩnh vực trong đời sống đều liên quan đến dữ liệu.

Data Mining là một tập hợp, một hệ thống các phương pháp tính toán, thuật toán được áp dụng cho các cơ sở dữ liệu lớn và phức tạp nhằm mục đích loại bỏ các chi tiết ngẫu nhiên, chi tiết ngoại lệ, khám phá các mẫu, mô hình, quy luật tiềm ẩn, các thông tin có giá trị trong bộ dữ liệu. Data mining là thành quả công nghệ tiên tiến ngày nay, là quá trình khám phá các kiến thức vô giá bằng cách phân tích khối lượng lớn dữ liệu đồng thời lưu trữ chúng ở nhiều cơ sở dữ liệu khác nhau”

Data mining là quá trình tìm kiếm các chi tiết bất thường (anomalies), các mẫu, mô hình, quy luật của dữ liệu và mối tương quan giữa các tập dữ liệu lớn để dự đoán kết quả, thiết lập các dự báo. Bằng cách áp dụng một loạt các kỹ thuật khác nhau, thông tin có được từ Data mining sẽ hỗ trợ tăng doanh thu, cắt giảm chi phí, cải thiện mối quan hệ khách hàng, giảm rủi ro,..”

Data mining là một quá trình được các công ty sử dụng để biến dữ liệu thô thành những thông tin hữu ích. Bằng cách sử dụng các phần mềm chuyên dụng để tìm kiếm các quy luật, các mẫu, thông tin có giá trị, mối tương quan tiềm ẩn trong khối lượng lớn dữ liệu, công ty có thể tìm hiểu thêm về khách hàng của mình để phát triển các chiến lược tiếp thị hiệu quả hơn, tăng doanh số và giảm chi phí.



Hình 1 Mối quan hệ giữa AI, ML và DL

Xu hướng phát triển công nghệ thông tin ngày càng tăng, song song với nó lượng dữ liệu được sinh ra cũng ngày một lớn. Vì vậy nhu cầu để xử lý dữ liệu cũng lớn hơn. ML đang góp phần giải quyết vấn đề này. Một trong những thuật toán thường dùng trong ML là K Nearest Neighbors.

Ứng dụng thuật toán này được sử dụng rất nhiều và rộng rãi trong các bài toán phân lớp.

1.3.MỤC ĐÍCH NGHIÊN CỨU

- Tìm hiểu về những khái niệm trong ML, DM.
- Nghiên cứu, tìm hiểu thuật toán KNN, ý nghĩa và ví dụ mình họa.
- Đánh giá hiểu quả của thuật toán.

1.4.PHẠM VI VÀ ĐỐI TƯỢNG NGHIÊN CỨU

- Phạm vi nghiên cứu: Thử nghiệm trên Iris flower dataset.
- Đối tượng nghiên cứu: Thuật toán KNN và bộ Iris flower dataset.

1.5.NỘI DUNG THỰC HIỆN

- Tìm hiểu thuật toán KNN.
- Làm quen với bộ dữ liệu Iris.
- Sử dụng bộ dữ liệu vào thử nghiệm đánh giá.

2. CƠ SỞ LÝ THUYẾT

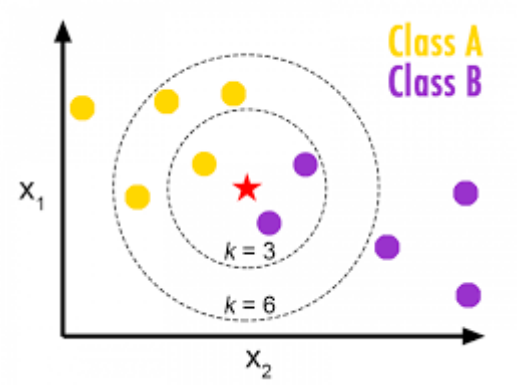
2.1.MACHINE LEARNING

2.1.1. Định nghĩa

- Là một lĩnh vực của trí tuệ nhân tạo liên quan đến việc nghiên cứu và xây dựng các kĩ thuật cho phép các hệ thống học tự động từ dữ liệu để giải quyết các vấn đề cụ thể. Ví dụ các máy có thể học cách phân loại thư điện tử có phải thư rác hay không và tự động sắp xếp các mục tương ứng.
- ML có liên quan đến thống kê vì cả hai lĩnh vực đều nghiên cứu việc phân tích dữ liệu, nhưng khác với thống kê, học máy tập trung vào sự phức tạp của các giải thuật trong việc thực thi tính toán.
- ML hiện nay được áp dụng rộng rãi bao gồm máy truy tìm dữ liệu, máy phân tích thị trường chứng khoáng, nhận dạng tiến nói và chữ viết…

2.1.2. Một số phương thức của Machine Learning

- **Lazy learning algorithm:** Là một thuật toán đơn giản chỉ sinh ra kết quả khi truy vấn được thực thi. KNN là một ví dụ điển hình của thuật toán này mà ta sẽ tìm hiểu.
- **None-parametric learning algorthm:** Trong ML thuật toán phân lớp có thể có tham số(parametric) hoặc không có tham số(non-parametric). Thuật toán non-parametric sử dụng một tham số linh hoạt và giá trị của số thường tăng khi bộ data lớn.
- **Học có giám sát:** Thuật toán dự đoán đầu ra của một dữ liệu mới (new input) dựa trên các cặp (input, outcome) đã biết từ trước. Cặp dữ liệu này còn được gọi là (data, label), tức (dữ liệu, nhãn). Supervised learning là nhóm phổ biến nhất trong các thuật toán Machine Learning
    - Học có giám sát được chia thành 2 loại chính:
        - **Classification (phân lớp):** Là quá trình phân lớp một đối tượng dữ liệu vào một hay nhiều lớp đã cho trước nhờ một mô hình phân lớp (model). Mô hình này được xây dựng dựa trên một tập dữ liệu được xây dựng trước đó gắn nhãn (hay còn gọi là tập huấn luyện). Quá trình phân lớp là quá trình gán nhãn cho đối tượng dữ liệu.



Hình 2 Ví dụ về mô hình phân lớp

Có nhiều bài toán phân lớp như phân lớp nhị phân, phân lớp đa phân, phân lớp đa trị. Trong đó phân lớp nhị phân là một loại phân lớp đặc biệt của phân lớp đa lớp

Ứng dụng của bài toán phân lớp được sử dụng rất nhiều và rộng rãi như nhận dạng khuôn mặt, nhận dạng chữ viết, nhận dạng giọng nói, phát hiện thư rác…

  - **Regression (hồi quy):** Nếu không được chia thành các nhóm mà là một giá trị thực cụ thể. Đầu ra của một điểm dữ liệu sẽ bằng chính đầu ra của điểm dữ liệu đã biết.

- **Học không giám sát:** Là một kĩ thuật của máy học nhằm tìm ra một mô hình hay cấu trúc bị ẩn bởi tập dữ liệu không được gán nhãn cho trước. UL khác với SL là không thể xác định được output từ tập dữ liệu huấn luyện được. Tùy thuộc vào tập huấn luyện kết quả output sẽ khác nhau. Trái ngược với SL, tập dữ liệu huấn luyện của UL không do con người gán nhãn, máy tính sẽ phải tự học hoàn toàn. Có thể nói, học không giám sát thì giá trị đầu ra sẽ phụ thuộc vào thuật toán UL. Ứng dụng lớn phổ biến của học không giám sát là bài toán phân cụm.
- **Học bán giám sát:** Các bài toán khi có một số lượng lớn dữ liệu nhưng chỉ có một phần trong chúng được dán nhãn. Những bài toán này nằm giữa phương thức học giám sát và không giám sát.

2.2.BÀI TOÁN PHÂN LỚP DỮ LIỆU

2.2.1. Định nghĩa

Bài toán phân lớp (classification) và bài toán gom cụm (cluster) là hai bài toán lớn trong lĩnh vực Machine Learnig (ML). Bài toán phân lớp là quá trình phân lớp một đối tượng dữ liệu vào một hay nhiều lớp đã cho trước nhờ một mô hình phân lớp (model). Mô hình này được xây dựng dựa trên một tập dữ liệu được xây dựng trước đó có gán nhãn (hay còn gọi là tập huấn luyện). Quá trình phân lớp là quá trình gán nhãn cho đối tượng dữ liệu.

Như vậy, nhiệm vụ của bài toán phân lớp là cần tìm một mô hình phần lớp để khi có dữ liệu mới thì có thể xác định được dữ liệu đó thuộc vào phân lớp nào.

Có nhiều bài toán phân lớp dữ liệu như phân lớp nhị phân (binary), phân lớp đa lớp (multiclass), phân lớp đa trị.

Bài toán phân lớp nhị phân là bài toán gắn nhãn dữ liệu cho đối tượng vào một trong hai lớp khác nhau dựa vào việc dữ liệu đó có hay không có các đặc trưng (feature) của bộ phân lớp

Bài toán phân lớp đa lớp là quá trình phân lớp dữ liệu với số lượng lớp lớn hơn hai. Như vậy với từng dữ liệu chúng ta phải xem xét và phân lớp chúng vào những lớp khác nhau chứ không phải là hai lớp như bài toán phân lớp nhị phân. Và thực chất bài toán phân lớp nhị phân là một bài toán đặt biệt của phân lớp đa lớp.

Ứng dụng của bài toán này được sử dụng rất nhiều và rộng rãi trong thực tế ví dụ như bài toán nhận dạng khuôn mặt, nhận diện giọng nói, phát hiện email spam…

Hình 3 Ví dụ bài toán phân lớp detect email spam

2.2.2. Quá trình phân lớp dữ liệu

Để xây dựng được mô hình phân lớp và đánh giá hiệu quả của mô hình cần phải thực hiện quá trình sau đây:

- **Bước 1: Chuẩn bị tập dữ liệu huấn luyện (dataset) và rút trích đặc trưng (feature extraction)**

Công đoạn này được xem là công đoạn quan trọng trong các bài toán về ML. vì đây là input cho việc học để tìm ra mô hình của bài toán. Chúng ta phải biết cần chọn ra những đặc trưng tốt nhất của dữ liệu, lược bỏ những đặc trưng không tốt của dữ liệu, gây nhiễu. Ước lượng số chiều của dữ liệu bao nhiêu là tốt hay nói cách khác là chọn bao nhiêu feature. Nếu số chiều quá lớn gây khó khăn cho việc tính toán thì phải giảm số chiều của dữ liệu nhưng vẫn giữ được độ chính xác của dữ liệu.

Ở bước này chúng ta cũng chuẩn bị bộ dữ liệu để test trên mô hình. Thông thường sẽ sử dụng cross-validation (kiểm tra chéo) để chia tập dataset thành 2 phần, một phần phục vụ cho training và phần còn lại phục vụ cho mục đích testing trên mô hình. Có hai cách thường sử dụng trong cross-validation là splitting và k-fold.

- **Bước 2: Xây dựng mô hình phân lớp (classifier model)**

Mục đích của mô hình huấn luyện là tìm ra hàm F(x) và thông qua hàm f tìm được chúng ta gán nhãn cho dữ liệu. Bước này thường được gọi là học hay training.

Trong đó: x là các feature hay input đầu vào của dữ liệu, y là nhãn các lớp hay output đầu ra

Thông thường để xây dựng mô hình phân lớp cho bài toán này chúng ta sử dụng các thuật toán học giám sát như **KNN**, **NN**, **SVM**, **Decision Tree**, **Navie Bayers**.

- **Bước 3: Kiểm tra dữ liệu với mô hình (make prediction)**

Sau khi tìm được mô hình phân lớp ở bước hai, thì bước này chúng ta sẽ đưa vào các dữ liệu mới để kiểm tra trên mô hình phân lớp.

- **Bước 4: Đánh giá mô hình phân lớp và chọn ra mô hình tốt nhất**

Bước cuối cùng chúng ta sẽ đánh giá mô hình bằng cách đánh giá mức độ lỗi của dữ liệu testing và dữ liệu training thông qua mô hình tìm được. Nếu không đạt được kết quả mong muốn của chúng ta thì phải thay đổi các tham số của thuật toán học để tìm ra các mô hình tốt hơn và kiểm tra, đánh giá lại mô hình phân lớp, và cuối cùng chọn ra mô hình phân lớp tốt nhất cho bài toán của chúng ta.

3. THUẬT TOÁN K-NEAREST NEIGHTBORS

3.1.ĐỊNH NGHĨA

Thuật toán KNN là thuật toán có mục đích phân loại lớp cho một mẫu mới (Query point) dựa trên các thuộc tính và lớp của các mẫu sẵn có (Training Data), các mẫu này được nằm trong một hệ gọi là không gian mẫu.

Một đối tượng được phân lớp dựa vào K láng giềng của nó. K là số nguyên dương được xác định trước khi thực hiện thuật toán. Người ta thường dùng khoảng cách Euclidean để tính toán khoảng cách giữa các đối tượng.

K-nearest neighbors (KNN) là một trong những thuật toán học có giám sát đơn giản nhất trong Machine Learning. Ý tưởng của KNN là tìm ra output của dữ liệu dựa trên thông tin của những dữ liệu training gần nó nhất.

KNN (K-Nearest Neighbors) là một trong những thuật toán học có giám sát đơn giản nhất được sử dụng nhiều trong khai phá dữ liệu và học máy. Ý tưởng của thuật toán này là nó không học một điều gì từ tập dữ liệu học (nên KNN được xếp vào loại lazy learning), mọi tính toán được thực hiện khi nó cần dự đoán nhãn của dữ liệu mới.

Lớp (nhãn) của một đối tượng dữ liệu mới có thể dự đoán từ các lớp (nhãn) của k hàng xóm gần nó nhất.

Là một trong những phương pháp phân lớp dựa trên thể hiện(Instance-based).

- Lưu trữ các mẫu/đối tượng huấn luyện và chỉ xử lý khi có yêu cầu phân lớp mẫu/đối tượng mới.
- Đưa mẫu/đối tượng vào lớp mà gần với chúng nhất.

Thuật toán k-NN: xác định lớp cho mẫu mới E

- Tính khoảng cách giữa mẫu E và tất cả các mẫu trong tập huấn luyện (Euclide)
- Chọn k mẫu gần nhất với mẫu E trong tập huấn luyện
- Gán mẫu E vào lớp có số mẫu chiếm đa số trong k mẫu láng giềng đó (hoặc mẫu E nhận giá trị trung bình của k mẫu)

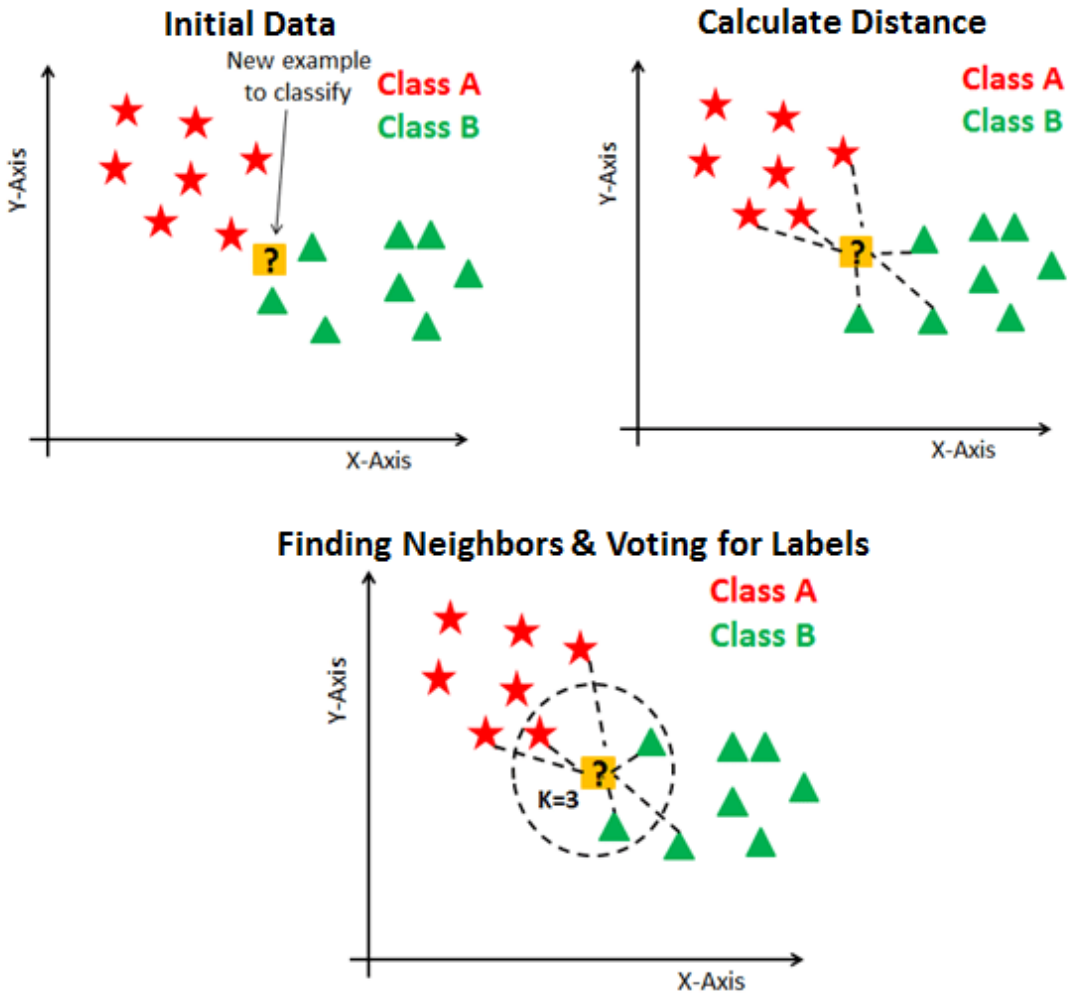
3.2.QUY TRÌNH LÀM VIỆC CỦA THUẬT TOÁN KNN

Các mẫu được mô tả bằng n-chiều thuộc tính số. Mỗi mẫu đại diện cho một điểm trong chiều không gian n-chiều. Theo cách này tất cả các mẫu được lưu trẫ trong một mô hình không gian n-chiều

Các bước thực hiện thuật toán KNN được mô tả như sau:

- **Bước 1:** Xác định tham số K (số láng giềng gần nhất)
- **Bước 2:** Tính khoảng cách đối tượng cần phân lớp với tất cả đối tượng trong training data.
- **Bước 3:** Sắp xếp khoảng cách theo thữ tự tăng dần và xác định K láng giềng gần nhất với đối tượng cần phân lớp.
- **Bước 4:** Lấy tất cả các lớp của K láng giềng gần nhất.
- **Bước 5:** Dựa vào phần lớn lớp của K để xác đin lớp cho đối tượng cần phân lớp.

3.2.1. Ví dụ minh họa



Hình 4 Ví dụ minh họa thuật toán KNN

Ta dễ dàng nhìn thấy có 2 loại: Class A (hình sao đỏ) và Class B(hình tam giác xanh). Và dấu '?' màu vàng(điểm mình muốn biết thuộc lớp nào.

Phương pháp đơn giản nhất để kiểm tra là tìm xem điểm gần chấm vàng thuộc màu nào nào (hình sao đỏ hay hình tam giác xanh). Từ hình trên ta dễ dạng nhận thấy điểm gần chấm vàng nhất là hình tam giác màu xanh, do đó nó sẽ được phân vào lớp tam giác màu xanh (K=3).

Có một vấn đề trong phương pháp trên, xung quanh chấm vàng xuất hiện rất nhiều chấm đỏ nên việc xét điểm gần nhất là chưa khả thi. Trường hợp K=4 ta sẽ thấy xung quanh chấm vàng có 2 đỏ và 2 xanh, đây là trường hợp có điểm bằng nhau, với trường hợp này

KNN sẽ xử lý bằng cách so sánh tổng khoảng cách của các hình gần nhất với điểm ta đang xét.

Do xuất hiện trường hợp có điểm bằng nhau, vì vậy người ta thường chọn k là số lẻ. Đó cũng là ý tưởng của KNN.

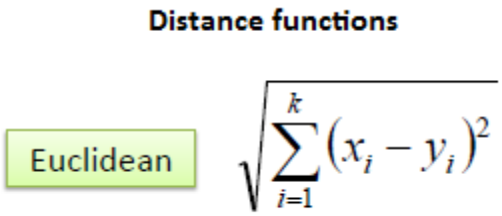
3.2.2. Ưu, nhược điểm của thuật toán

- **Ưu điểm:**
  - Dễ sử dụng và cài đặt.
  - Việc dự đoán kết quả của dữ liệu mới dễ dàng.
  - Độ phức tạp tính toán nhỏ.
- **Nhược điểm:**
  - Cần thời gian lưu training set, khi dữ liệu training và test tăng lên nhiều sẽ mất nhiều thời gian tính toán.
  - KNN rất nhạy cảm với dữ liệu khi K nhỏ

3.3.KHOẢNG CÁCH TRONG KHÔNG GIAN VECTOR

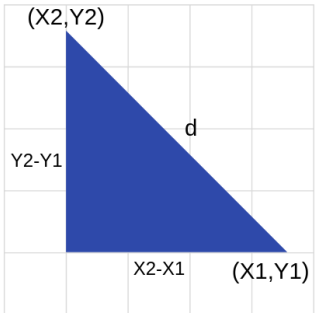
Trong không giam một chiều, việc đo khoản cách giữa hai điểm đã rất quen thuộc: lấy trị tuyết đối của hiệu giữa hai giá trị đó. Trong không gian hai chiều, tức mặt phẳng, chúng ta thường dùng khoản cách Euclid để đo khoản cách giữa hai điểm.

Công thức Euclid:



Hình 5 Công thức Euclidean

Một cách trực quan ta có hình vẽ sau



Hình 6 Công thức Euclidean dưới dạng hình vẽ

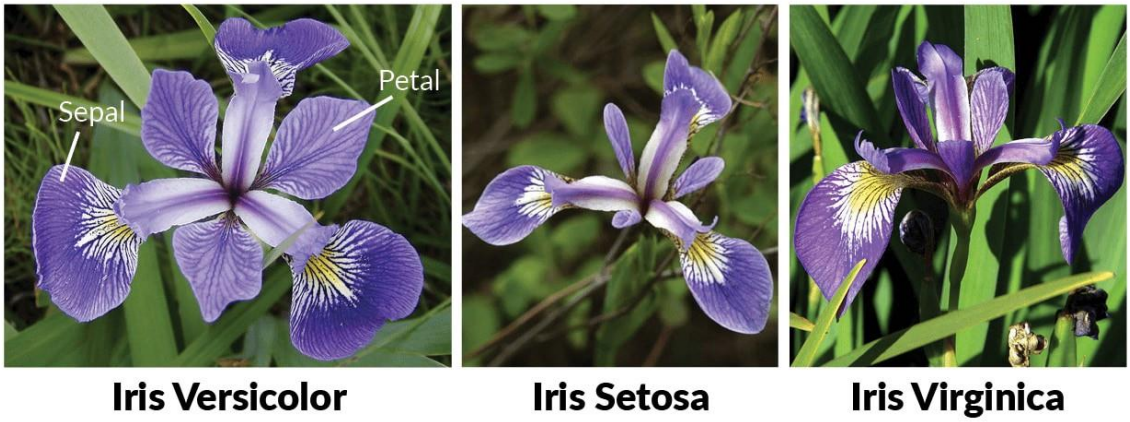
4. THỬ NGHIỆM

4.1.BỘ DỮ LIỆU IRIS FLOWER DATASET

4.1.1. Giới thiệu

Tập dữ liệu hoa Iris hoặc tập dữ liệu Iris của Fisher là một tập dữ liệu đa biến được giới thiệu bởi nhà thống kê, nhà ưu sinh học và nhà sinh vật học người Anh Ronald Fisher trong bài báo năm 1936 của ông Việc sử dụng nhiều phép đo trong các vấn đề phân loại như một ví dụ về phân tích phân biệt tuyến tính. Đôi khi nó được gọi là tập dữ liệu Iris của Anderson vì Edgar Anderson đã thu thập dữ liệu để định lượng sự biến đổi hình thái của hoa Iris của ba loài liên quan. Hai trong số ba loài được thu thập ở bán đảo.

Bộ dữ liệu bao gồm 50 mẫu từ mỗi ba loại hoa Iris (Iris setosa, Iris virginica và Iris verscolor). Bốn đặc điểm được đo từ mỗi mẫu: chiều dài và chiều rộng của đài hòa, chiều dài và chiều rộng của cánh hoa tính bằng centimet. Dựa trên sự kết hợp của bốn tính năng này, Fisher đã phát triển một mô hình phân biệt tuyến tính để phân biệt các loài với nhau.



Hình 7 Hình ảnh minh họa về Iris flower dataset

4.1.2. Định nghĩa bài toán

Với 1 bộ dữ liệu đồ sộ về hoa Iris đã có về các cá thể hoa thuộc các loài Iris cụ thể, ta không cần phải phân tích về gen hay các phân tích về sinh học phức tạp để nhận biết ra một cá thể hoa đang xét thuộc loài hoa Iris nào, bằng các dữ liệu đã có sẵn và dựa vào các thuộc tính của cá thể, ta có thể đưa ra nhận biết nhanh chóng cá thể đó thuộc loài hoa Iris nào bằng các thuộc tính toán số học trên máy tính (Machine Learning).

4.1.3. Thu thập và tiền xử lý số liệu

Bộ dữ liệu sau khi được rút gọn bao gồm 5 thuộc tính:

- 4 Thuộc tính kiểu số: chiều dài đài hoa, chiều rộng đài hoa, chiều dài cánh hoa, chiều rộng cánh hoa (đơn vị cm).
- 1 Thuộc tính còn lại là tên của loài hoa Iris (có 3 loài tất cả: Iris Setosa, Iris Versicolour, Iris Virginica).

Theo tổng kết thống kê:

	Giá trị nhỏ nhất	Giá trị lớn nhất	Giá trị trung bình
Chiều dài đài hoa	4.3	7.9	5.84
Chiều rộng đài hoa	2.0	4.4	3.05
Chiều dài cánh hoa	1.0	6.9	3.76
Chiều rộng cánh hoa	0.1	2.5	1.20

Tỷ lệ phân chia cho mỗi loài trong 3 loài Iris trên là 33%

4.1.3.1. Làm sạch dữ liệu (Data Cleaning)

- Thiếu giá trị: Khi xảy ra sự thiếu thông tin ở 1 thuộc tính nào đó trong 1 bản ghi của bộ dữ liệu thu thập, khi mà tính đảm bảo số bản ghi chia đều cho 3 loài trên đã có (33,3%) và số lượng bản ghi thiếu là ít ta có thể áp dụng phương pháp loại bỏ. Mặt khác khi cần phải điền vào các giá trị thiếu bằng các giá trị trung bình đã được thống kê ở trên
- Nhiễu dữ liệu: Khi xuất hiện 1 giá trị bất ngờ nào đó, đột nhiên vượt qua các giá trị biên đã được thống kê, ta có thể sửa lại giá trị đó thành các giá trị ở vùng biên theo bảng thống kê ở trên.

4.1.3.2. Chọn lọc dữ liệu (Data Selection)

- Tích hợp và dư thừa dữ liệu: do dữ liệu trên thu thập từ 1 nguồn duy nhất nên việc tích hợp là không cần thiết, các thuộc tính của dữ liệu là độc lập nhau, không có mối quan hệ tương quan nào, các thuộc tính đã được rút gọn chọc lọc nên không cần việc phải phân tích dư thừa dữ liệu.

4.1.3.3. Biến đổi dữ liệu (Data Transformation)

- Biến đổi dữ liệu: ta sử dụng phương pháp chuẩn hóa dữ liệu về khoảng giá trị [0,1], phần này sẽ được nói rõ hơn ở phần sau tùy vào phương pháp khai phá dữ liệu được chọn.

4.1.3.4. Rút gọn dữ liệu (Data Reduction)

- Như đã nói từ đầu, bộ dữ liệu đã được loại bỏ đi các thuộc tính mang tính chất mô tả, chỉ giữ lại 1 thuộc tính mô tả duy nhất là tên loài Iris cụ thể. Các thuộc tính mang chỉ số học còn lại vẫn đảm bảo được tính phân loại.

4.1.4. Khai phá dữ liệu (Data Mining)

Dựa vào phát biểu và mục đích của bài toán, dễ dàng nhận thấy đây là 1 bài toán phân lớp: với 1 bộ dữ liệu đã cho và đã được gán nhãn các lớp cho trước(thể hiện thông qua thuộc tính loài Iris cụ thể), ta cần gán cấc mẫu mới(chưa biết thuộc nhãn lớp nào) vào các lớp với độ chính xác cao nhất có thể.

Bộ dữ liệu được chọn có số thuộc tính không lớn, để đơn giản và dễ sử dụng đề phân lớp ta lựa chọn dùng phân lớp theo thuật toán "K-Người hàng xóm gần nhất" (K-Nearest Neighbors Algorithmn).

4.1.5. Bộ dataset

Chiều dài đài hoa	Chiều rộng đài hoa	Chiều dài cánh hoa	Chiều rộng cánh hoa	Loại
5.1	3.5	1.4	0.2	setosa
4.9	3.0	1.4	0.2	setosa
4.7	3.2	1.3	0.2	setosa
4.6	3.1	1.5	0.2	setosa
5.0	3.6	1.4	0.2	setosa
5.4	3.9	1.7	0.4	setosa
4.6	3.4	1.4	0.3	setosa
5.0	3.4	1.5	0.2	setosa
4.4	2.9	1.4	0.2	setosa
4.9	3.1	1.5	0.1	setosa
5.4	3.7	1.5	0.2	setosa
4.8	3.4	1.6	0.2	setosa
4.8	3.0	1.4	0.1	setosa
4.3	3.0	1.1	0.1	setosa
5.8	4.0	1.2	0.2	setosa
5.7	4.4	1.5	0.4	setosa
5.4	3.9	1.3	0.4	setosa
5.1	3.5	1.4	0.3	setosa
5.7	3.8	1.7	0.3	setosa
5.1	3.8	1.5	0.3	setosa
5.4	3.4	1.7	0.2	setosa
5.1	3.7	1.5	0.4	setosa

4.6	3.6	1.0	0.2	setosa
5.1	3.3	1.7	0.5	setosa
4.8	3.4	1.9	0.2	setosa
5.0	3.0	1.6	0.2	setosa
5.0	3.4	1.6	0.4	setosa
5.2	3.5	1.5	0.2	setosa
5.2	3.4	1.4	0.2	setosa
4.7	3.2	1.6	0.2	setosa
4.8	3.1	1.6	0.2	setosa
5.4	3.4	1.5	0.4	setosa
5.2	4.1	1.5	0.1	setosa
5.5	4.2	1.4	0.2	setosa
4.9	3.1	1.5	0.1	setosa
5.0	3.2	1.2	0.2	setosa
5.5	3.5	1.3	0.2	setosa
4.9	3.1	1.5	0.1	setosa
4.4	3.0	1.3	0.2	setosa
5.1	3.4	1.5	0.2	setosa
5.0	3.5	1.3	0.3	setosa
4.5	2.3	1.3	0.3	setosa
4.4	3.2	1.3	0.2	setosa
5.0	3.5	1.6	0.6	setosa
5.1	3.8	1.9	0.4	setosa

4.8	3.0	1.4	0.3	setosa
5.1	3.8	1.6	0.2	setosa
4.6	3.2	1.4	0.2	setosa
5.3	3.7	1.5	0.2	setosa
5.0	3.3	1.4	0.2	setosa
7.0	3.2	4.7	1.4	versicolor
6.4	3.2	4.5	1.5	versicolor
6.9	3.1	4.9	1.5	versicolor
5.5	2.3	4.0	1.3	versicolor
6.5	2.8	4.6	1.5	versicolor
5.7	2.8	4.5	1.3	versicolor
6.3	3.3	4.7	1.6	versicolor
4.9	2.4	3.3	1.0	versicolor
6.6	2.9	4.6	1.3	versicolor
5.2	2.7	3.9	1.4	versicolor
5.0	2.0	3.5	1.0	versicolor
5.9	3.0	4.2	1.5	versicolor
6.0	2.2	4.0	1.0	versicolor
6.1	2.9	4.7	1.4	versicolor
5.6	2.9	3.6	1.3	versicolor
6.7	3.1	4.4	1.4	versicolor
5.6	3.0	4.5	1.5	versicolor
5.8	2.7	4.1	1.0	versicolor

6.2	2.2	4.5	1.5	versicolor
5.6	2.5	3.9	1.1	versicolor
5.9	3.2	4.8	1.8	versicolor
6.1	2.8	4.0	1.3	versicolor
6.3	2.5	4.9	1.5	versicolor
6.1	2.8	4.7	1.2	versicolor
6.4	2.9	4.3	1.3	versicolor
6.6	3.0	4.4	1.4	versicolor
6.8	2.8	4.8	1.4	versicolor
6.7	3.0	5.0	1.7	versicolor
6.0	2.9	4.5	1.5	versicolor
5.7	2.6	3.5	1.0	versicolor
5.5	2.4	3.8	1.1	versicolor
5.5	2.4	3.7	1.0	versicolor
5.8	2.7	3.9	1.2	versicolor
6.0	2.7	5.1	1.6	versicolor
5.4	3.0	4.5	1.5	versicolor
6.0	3.4	4.5	1.6	versicolor
6.7	3.1	4.7	1.5	versicolor
6.3	2.3	4.4	1.3	versicolor
5.6	3.0	4.1	1.3	versicolor
5.5	2.5	4.0	1.3	versicolor
5.5	2.6	4.4	1.2	versicolor

6.1	3.0	4.6	1.4	versicolor
5.8	2.6	4.0	1.2	versicolor
5.0	2.3	3.3	1.0	versicolor
5.6	2.7	4.2	1.3	versicolor
5.7	3.0	4.2	1.2	versicolor
5.7	2.9	4.2	1.3	versicolor
6.2	2.9	4.3	1.3	versicolor
5.1	2.5	3.0	1.1	versicolor
5.7	2.8	4.1	1.3	versicolor
6.3	3.3	6.0	2.5	virginica
5.8	2.7	5.1	1.9	virginica
7.1	3.0	5.9	2.1	virginica
6.3	2.9	5.6	1.8	virginica
6.5	3.0	5.8	2.2	virginica
7.6	3.0	6.6	2.1	virginica
4.9	2.5	4.5	1.7	virginica
7.3	2.9	6.3	1.8	virginica
6.7	2.5	5.8	1.8	virginica
7.2	3.6	6.1	2.5	virginica
6.5	3.2	5.1	2.0	virginica
6.4	2.7	5.3	1.9	virginica
6.8	3.0	5.5	2.1	virginica
5.7	2.5	5.0	2.0	virginica

5.8	2.8	5.1	2.4	virginica
6.4	3.2	5.3	2.3	virginica
6.5	3.0	5.5	1.8	virginica
7.7	3.8	6.7	2.2	virginica
7.7	2.6	6.9	2.3	virginica
6.0	2.2	5.0	1.5	virginica
6.9	3.2	5.7	2.3	virginica
5.6	2.8	4.9	2.0	virginica
7.7	2.8	6.7	2.0	virginica
6.3	2.7	4.9	1.8	virginica
6.7	3.3	5.7	2.1	virginica
7.2	3.2	6.0	1.8	virginica
6.2	2.8	4.8	1.8	virginica
6.1	3.0	4.9	1.8	virginica
6.4	2.8	5.6	2.1	virginica
7.2	3.0	5.8	1.6	virginica
7.4	2.8	6.1	1.9	virginica
7.9	3.8	6.4	2.0	virginica
6.4	2.8	5.6	2.2	virginica
6.3	2.8	5.1	1.5	virginica
6.1	2.6	5.6	1.4	virginica
7.7	3.0	6.1	2.3	virginica
6.3	3.4	5.6	2.4	virginica

6.4	3.1	5.5	1.8	virginica
6.0	3.0	4.8	1.8	virginica
6.9	3.1	5.4	2.1	virginica
6.7	3.1	5.6	2.4	virginica
6.9	3.1	5.1	2.3	virginica
5.8	2.7	5.1	1.9	virginica
6.8	3.2	5.9	2.3	virginica
6.7	3.3	5.7	2.5	virginica
6.7	3.0	5.2	2.3	virginica
6.3	2.5	5.0	1.9	virginica
6.5	3.0	5.2	2.0	virginica
6.2	3.4	5.4	2.3	virginica
5.9	3.0	5.1	1.8	virginica

4.1.6. Áp dụng thuật toán K-NN vào bài toán phân lớp hoa Iris

Trích dẫn từ bộ dữ liệu hoa Iris, ta lập ra 1 bảng dữ liệu nhỏ và tiến hành thực hiện theo thuật toán K-NN để minh họa:

Chiều dài đài hoa	**Chiều rộng đài hoa**	**Chiều dài cánh hoa**	**Chiều rộng cánh hoa**	**Tên loài**
5.1	3.5	1.4	0.2	**Setosa**
4.7	3.2	1.3	0.2	**Setosa**
7.0	3.2	4.7	1.4	**Versicolor**
6.3	3.3	6.0	2.5	**Virginica**
5.9	3.0	4.2	1.5	**Versicolor**
5.1	3.8	1.6	0.2	**???**

Ta tiến hành thường hóa dữ liệu, bảng dữ liệu sẽ được biến đổi thành bảng sau:

Chiều dài đài hoa	Chiều rộng đài hoa	Chiều dài cánh hoa	Chiều rộng cánh hoa	Tên loài
5.1/7.0=0.73	3.5/3.8=0.92	1.4/6.0=0.23	0.2/2.5=0.08	**Setosa**
4.7/7.0=0.67	3.2/3.8=0.84	1.3/6.0=0.22	0.2/2.5=0.08	**Setosa**
7.0/7.0=1	3.2/3.8=0.84	4.7/6.0=0.78	1.4/2.5=0.56	**Versicolor**
6.3/7.0=0.9	3.3/3.8=0.87	6.0/6.0=1	2.5/2.5=1	**Virginica**
5.9/7.0=0.84	3.0/3.8=0.79	4.2/6.0=0.7	1.5/2.5=0.6	**Versicolor**
5.1/7.0=0.73	3.8/3.8=1	1.6/6.0=0.27	0.2/2.5=0.08	**???**

Ta tính toán khoảng cách **Euclide** được bằng khoảng cách tương ứng:

Tên loài	Khoảng cách
Setosa	0.008
Iris-setosa	0.178
Versicolor	0.767
Virginica	1.194
Versicolor	0.715

- **Chọn K = 3**

Ta chọn K hàng xóm gần nhất là 3 thì 3 khoảng cách nhỏ nhất lần lượt là: 0.008, 0.178, 0.715 tương ứng với tên loài: Setosa, Setosa và Versicolor

Trong 3 hàng xóm gần nhất này có 2 hàng xóm là Setosa và 1 hàng xóm là Versicolor. Vậy ta kết luận loài cần phân lớp thuộc loài Setosa.

- **Chọn K = 4**

Trong trường hợp ta chọn K hàng xóm gần nhất là 4 thì trong 4 hàng xóm đấy có 2 hàng xóm là Setosa và 2 hàng xóm là Versicolor

Do số hàng xóm ở đây bằng nhau (đều bằng 2). Ta tiến hành thêm 1 bước nữa là đi tính trung bình khoảng cách của mỗi hàng xóm rồi so sánh các khoảng cách trung bình đó với nhau rồi chọn ra khoảng cách nhỏ nhất và lấy loài hoa đại diện cho khoảng cách được chọn gán nhãn cho cá thể đang xét.

Như ví dụ trên ta đi tính khoảng cách trung bình nên ta có bảng:

Tên loài	Khoảng cách trung bình
Setosa	0.093

Versicolor	0.741

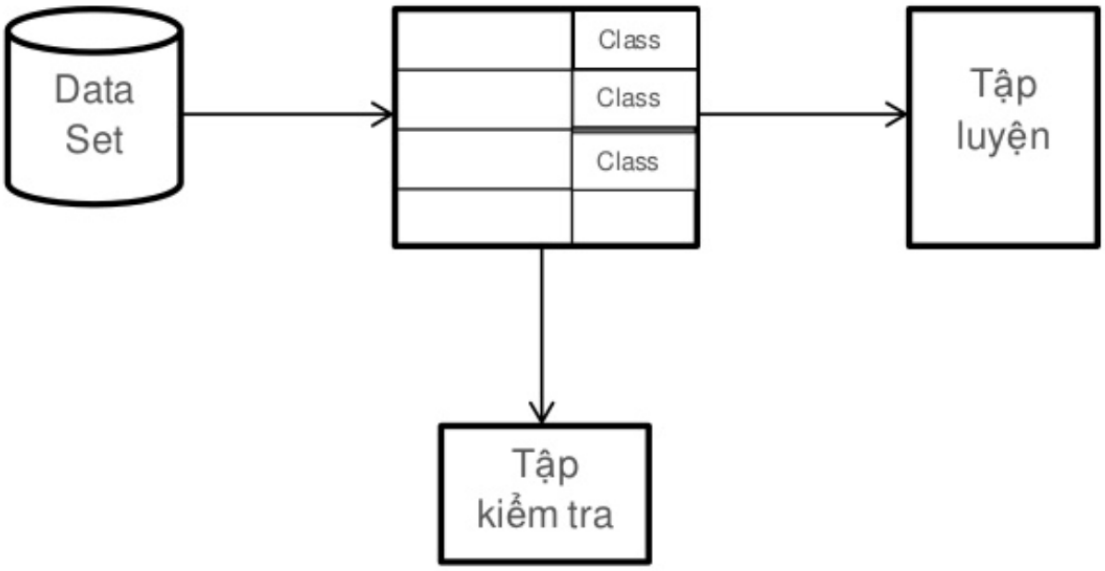
Ta chọn khoảng cách trung bình nhỏ nhất là 0.093 tương ứng với loài hoa Setosa. Ta kết luận cá thể hoa đang xét thuộc loài Setosa.

4.1.7. Đánh giá độ chính xác của mô hình và chương trình chạy

4.1.7.1. Cách thức đánh giá độ chính xác của mô hình

Ước lượng đọ chính xác của bộ phân lớp là quan trọng bởi nó cho phép dự đoán được độ chính xác của các kết quả phân lớp những dữ liệu trong tương lai. Phương pháp được thực hiện qua hai bước sau:

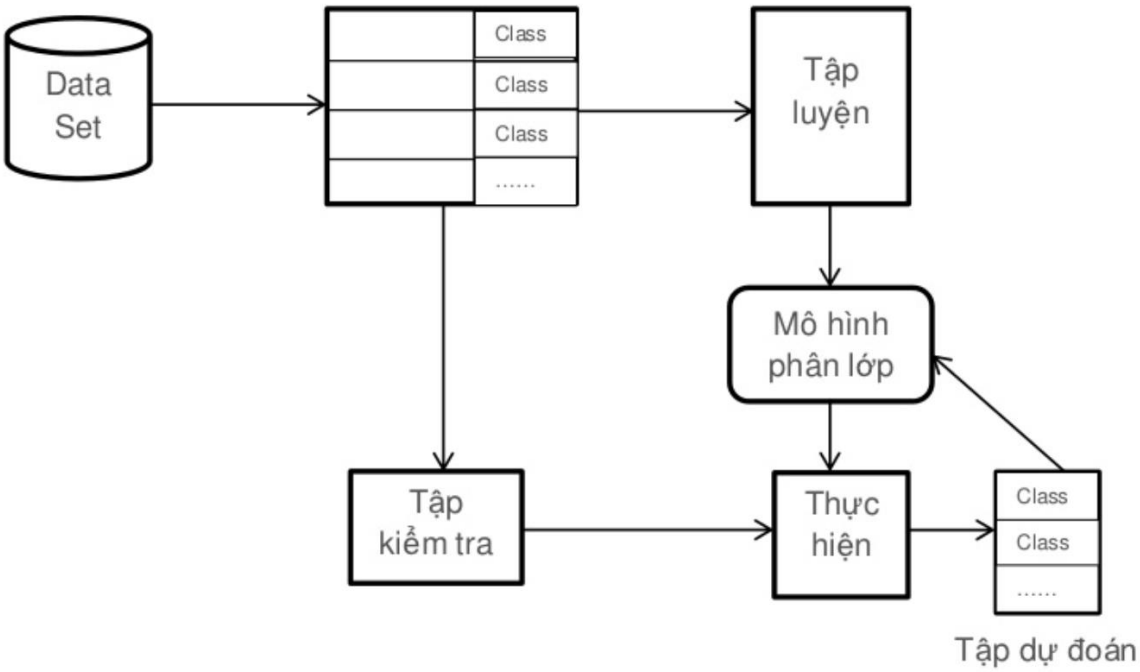
- **Bước 1:** Phân dữ liệu một cách ngẫu nhiên vào hai tập là tập luyện và tập kiểm tra.



Một cách tổng qua, tập luyện lớn hơn thì việc phân lớp tốt hơn, tập kiểm tra lớn hơn sẽ đúng hơn về việc đánh giá lỗi. Thông thường, người ta thường chọn tập huấn luyên bằng 4/5 và tập kiểm tra bằng 1/5.

Trong trương hợp só phần tử của các lớp trong tập dữ liệu gốc là không đều, ví dụ như về y học thì lớp khỏe mạnh chiếm 90% còn lớp bệnh tật chiếm 10%, như vậy nếu vẫn chọn ra ngẫu nhiên tập kiểm tra thì sẽ không đánh giá đúng tỷ lệ lỗi. Để khắc phục, ta chọn ngẫu nhiên ở mỗi lớp ra số phần tử cho tập kiểm tra bằng 1/5 số phần tử mỗi lớp.

- **Bước 2:** Đánh giá mô hình thông qua tập kiểm tra



Tỷ lệ lỗi:

$$Error\ rate = \frac{Số\ mẫu\ phân\ sai}{Tổng\ số\ mẫu\ kiểm\ tra}\ X\ 100\%$$

Ta đánh giá sự chính xác của mô hình thông qua tỷ lệ lỗi. Tỷ lệ lỗi càng nhỏ, mô hình có độ chính xác càng cao, có tính phù hợp tốt.

4.2.CÀI ĐẶT

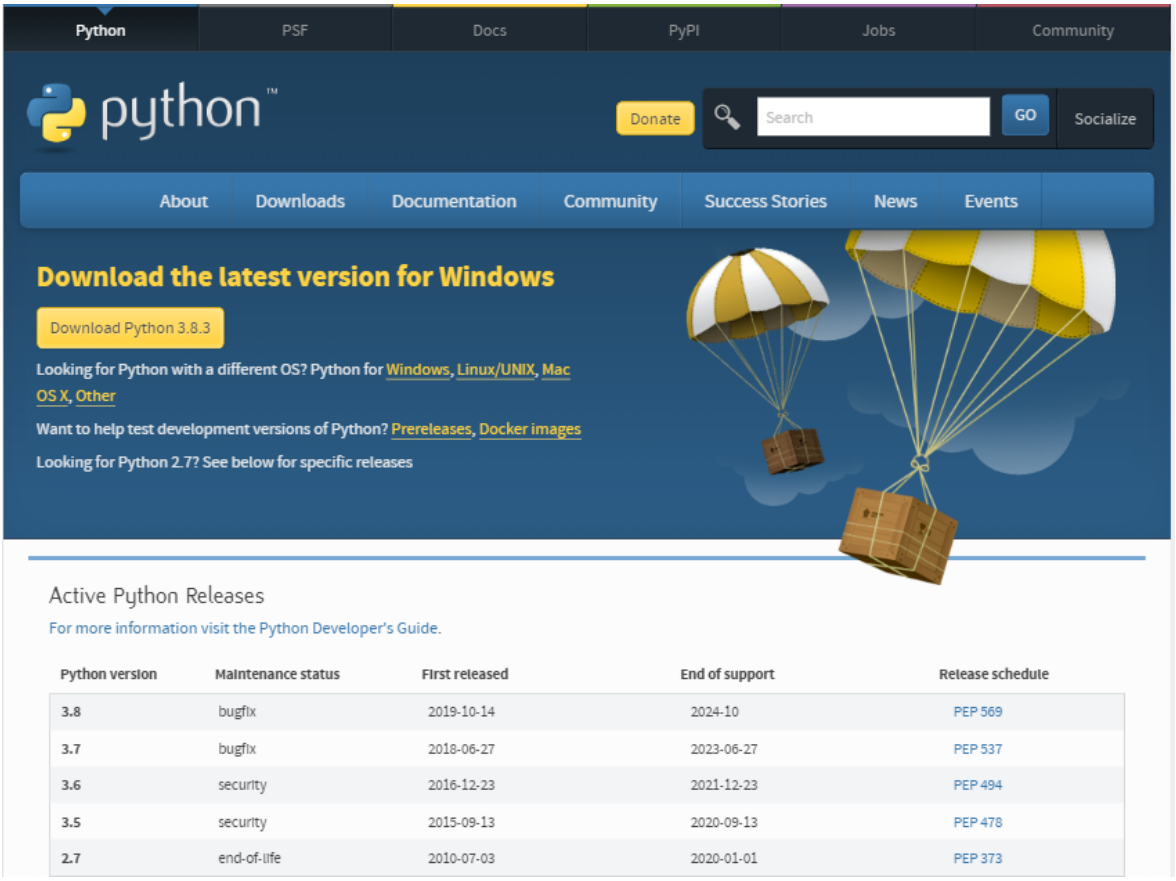
4.2.1. Cài đặt môi trường phát triển

4.2.1.1. Python

Python là một ngôn ngữ lập trình mạnh mẽ, đặc biệt là trong việc xử lý tính toán khoa học, đây là một ngôn ngữ tuyệt vời bởi cú pháp ngắn gọn và logic.

Tải Python tại đây: https://www.python.org/downloads/

Phiên bản python hiện tại là 3.8.3

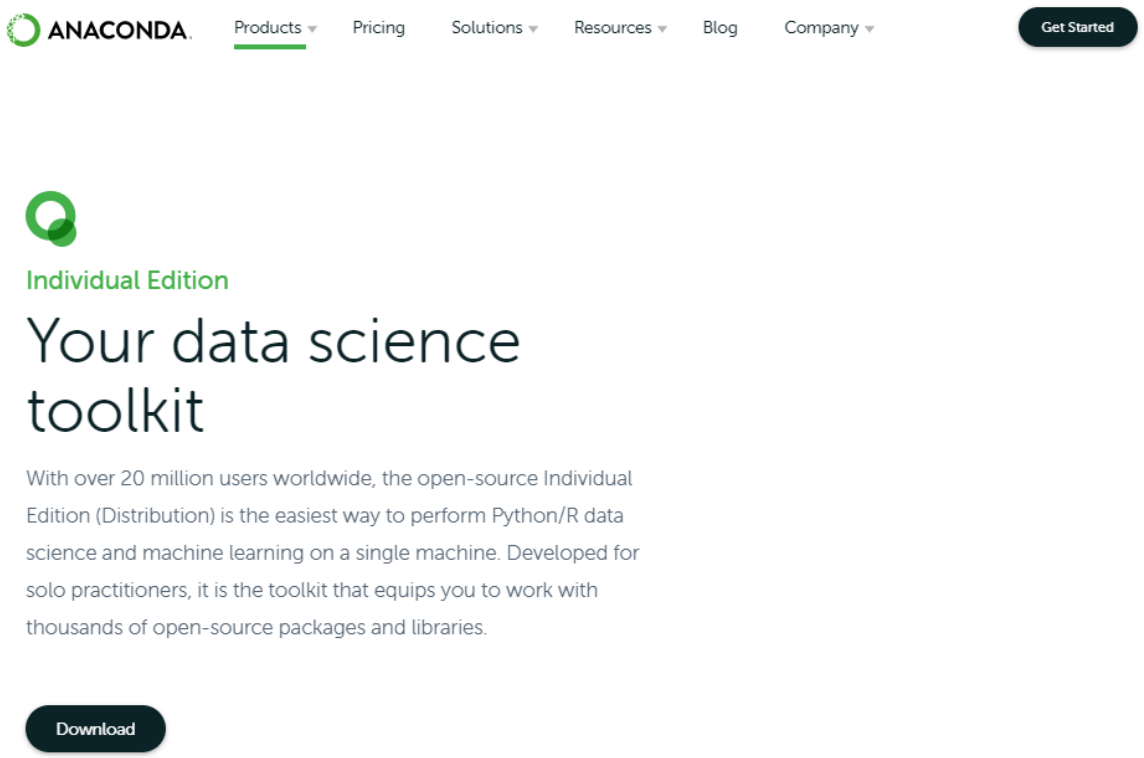


4.2.1.2. Anaconda

Được dịch từ tiếng Anh-Anaconda là một bản phân phối miễn phí và nguồn mở của các ngôn ngữ lập trình Python và R cho máy tính khoa học.

Trong Anaconda, ta sẽ quan tâm đến một công cụ hữu ích là **Jupyter Notebook.** Đây như là một trình soạn thảo và biên dịch code online, ta có thể thực thi câu lệnh và ghi chú cho từng câu lệnh một cách trực quan.

Tải Anaconda tại đây: https://docs.conda.io/projects/conda/en/latest/user-guide/install/download.html



4.2.1.3. Jupyter Notebook

Sau khi cài xong anaconda, ta có thể dụng lệnh sau để cài notebook

```
conda install -c conda-forge notebook
```

Sau khi cài xong ta tiến hành mở Jupyter Notebook bằng các bước sau:

- **Bước 1:** Khởi chạy môi trường conda (anaconda đã cài ở trên)

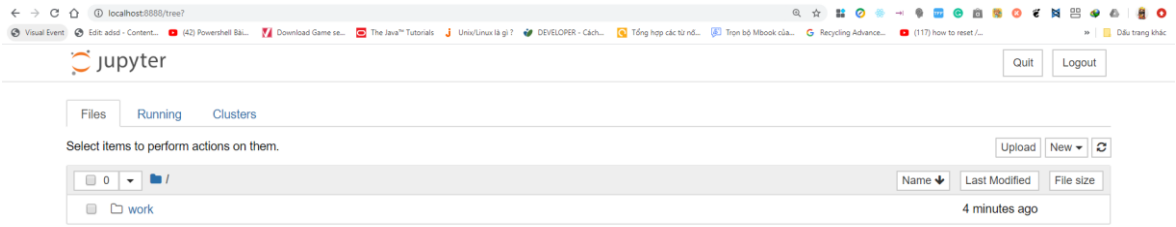
```
conda active root
```

- **Bước 2:** Chạy jupyter notebook

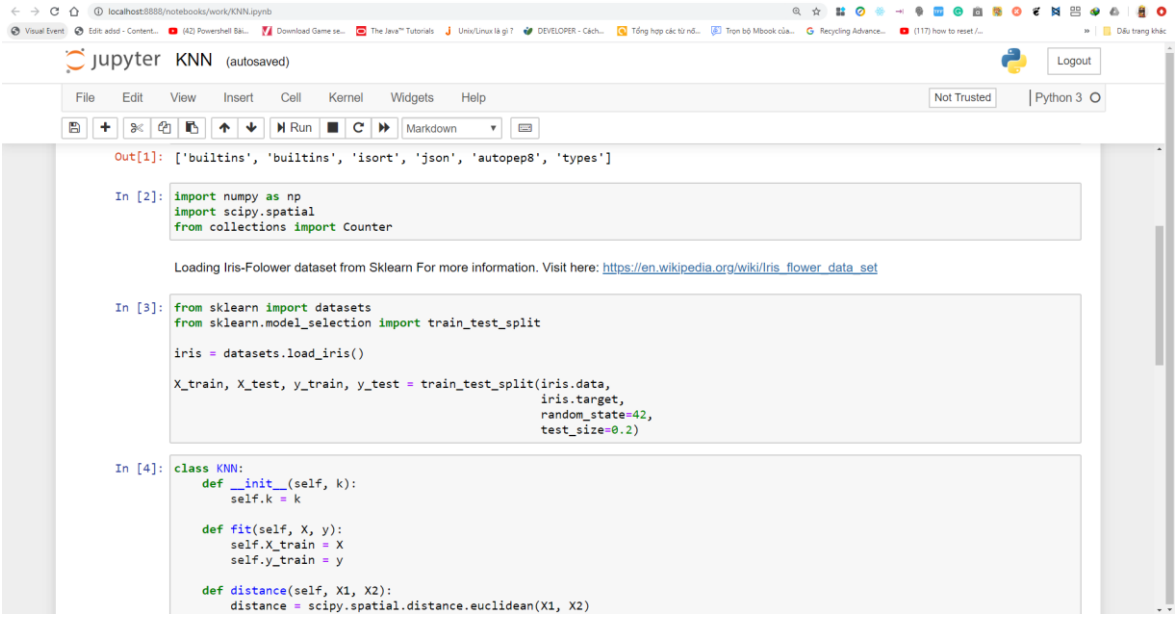
```
jupyter notebook
```

Sau khi chạy xong câu lệnh trên, một server web sẽ mở ra thường lắng nghe ở http://localhost:8888/.

Đây là giao diện khi ta truy cập



Bước tiếp theo là ta sẽ tiến hành mở file **.ipynb** (Đây sẽ là file trình bày thuật toán).



Ở môi trường notebook này, a có thể thêm chú thích, chạy từng dòng lệnh một cách dễ dàng. Bước tiếp theo ta sẽ tìm hiểu cách triển khai thuật toán.

4.2.2. Phát triển thuật toán

Trước khi bắt đầu, ta cần phải import những thư viện hỗ trợ xử lý bộ dữ liệu dataset. Ta sẽ import những thư viện sau:

Numpy: Thư viện hỗ trợ tính toán

Sklearn: là một thư viện máy học phần mềm miễn phí cho ngôn ngữ lập trình Python

```
import numpy as np
import scipy.spatial
from collections import Counter
from sklearn import datasets
from sklearn.model_selection import train_test_split
```

Sau đó ta tiến hành import bộ dữ liệu Iris. Ta có thể truy xuất bộ dataset Iris thông qua thư viện sklearn.

Sau khi import xong bộ iris dataset. Ta tiến hành chia thành 2 phần. Phân 1(80%) và phần 2(20%) để huấn luyện và kiểm thử.

```
iris = datasets.load_iris()
X_train, X_test, y_train, y_test = train_test_split(
    iris.data,
    iris.target,
    (random_state = 42),
    (test_size = 0.2)
  ))
```

Bước tiếp theo sẽ là viết một hàm để tính toán chính

Hàm khởi tạo __init__(k), sẽ nhận vào tham số K, K là số lượng hàng xóm gần nhất dùng để tìm ra K phần tử và chọn ra kết quả sẽ là lớp chiếm đa số trong K phần tử tìm được.

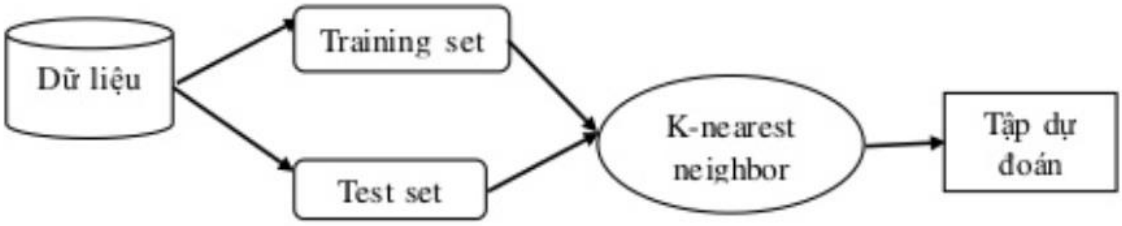
Hàm fit() sẽ là hàm nhận dataset để training.

Hàm distance() dùng để tính khoảng cách. Ở đây ta có thể sử dụng hàm tính khoảng cách thông qua thư viện.

Hàm score() sẽ là hàm tính độ chính xác của thuật toán.

```
class KNN:
    def __init__(self, k):
        self.k = k

    def fit(self, X, y):
        self.X_train = X
        self.y_train = y

    def distance(self, X1, X2):
        distance = scipy.spatial.distance.euclidean(X1, X2)

    def predict(self, X_test):
        final_output = []
        for i in range(len(X_test)):
            d = []
            votes = []
            for j in range(len(X_train)):
                dist =
scipy.spatial.distance.euclidean(X_train[j], X_test[i])
                d.append([dist, j])
            d.sort()
            d = d[0:self.k]
            for d, j in d:
                votes.append(y_train[j])
            ans = Counter(votes).most_common(1)[0][0]
            final_output.append(ans)

        return final_output

    def score(self, X_test, y_test):
        predictions = self.predict(X_test)
        return (predictions == y_test).sum() / len(y_test)
```

4.3.THỬ NGHIỆM



Hình 8 Mô hình bài toán

Sau khi viết xong hàm xử lý tính toán, ta sẽ tiến hành kiểm tra bộ dữ liệu. Ở ví dụ này với K = 3. Ta sẽ kiểm thử kết quả của bộ test (20%) mà ta đã chia ra ở trên. Sau đó xuất ra kết quả.

```
clf = KNN(3)
clf.fit(X_train, y_train)
prediction = clf.predict(X_test)

for i in prediction:
    print(i, end= ' ')
```

```
1 0 2 1 1 0 1 2 1 1 2 0 0 0 0 1 2 1 1 2 0 2 0 2 2 2 2 2 0 0
```

Đây là kết quả: Tương ứng với các giá trị sau.

0: setosa

1: versicolor

2: virginica

Sau đó, ta sẽ kiểm tra kết quả test trên với kết quả test đã biết trước mà ta đã chia ở trên.

```
prediction == y_test
```

So khớp kết quả.

```
array([True,True,True,True,True,True,True,True,True,True,Tr
ue,True,True,True,True,True,True,True,True,True,True,True,T
rue,True,True,True,True,True,True,True,])
```

Sau đó tính toán độ chính xác với kết quả

```
clf.score(X_test, y_test)
```

Với kết quả.

```
1.0
```

Bộ test trên là bộ test được tách ra từ tập huấn luyên Iris gồm 150 bộ. Ta dùng 120 bộ để train(80%) và 30 bộ để test(20%). Và kết luận được độ chính xác của bộ dữ liệu.

Phần tiếp theo ta sẽ thử nghiệm kết quả trên một bộ ngẫu nhiên.

```
tests=[[5,5,5,5],[6,5,2,4]]
prediction = clf.predict(tests)
```

Và kết quả tương ứng với [5,5,5,5]➔Virginica và [6,5,2,4] ➔Setosa

```
2 0
```

5. TỔNG KẾT

5.1.NHẬN XÉT

Thuật toán KNN là một thuật toán đơn giản, có giám sát có thể được sử dụng để giải quyết bài toán phân loại và hồi quy. Cách triển khai rất đơn giản và dễ hiểu nhưng có một hạn chế là sẽ trở nên rất chậm khi dữ liệu ngày càng lớn bởi phải tính toán cho mỗi dự liệu lớn đó.

KNN hoạt động dựa vào tính khoảng cách đến các bộ training trong data, chọn ra một số K và chọn ra những loại xuất hiện nhiều nhất.

Qua quá trình thực hiện nghiên cứu, em đã nắm được những kiến thức về thuật toán K-Nearest Neighbors và tổng quan về Machine Learning.

5.2.KẾT LUẬN

Những kết quả đạt được:

- Sự hiểu biết về thuật toán KNN cơ bản tương đối tốt
- Làm quen với Iris flower dataset
- Từ những gì làm được, từ đó hiểu biết thêm về AI, ứng dụng của ML vào đời sống công nghệ hiện đại.
- Làm quen ngôn ngữ lập trình python.

Những hạn chế:

- Thuật toán phụ thuộc nhiều vào hệ số K.
- Bộ dữ liệu đơn giản

6. TÀI LIỆU THAM KHẢO

https://machinelearningcoban.com/2017/01/08/knn/

https://en.wikipedia.org/wiki/Iris_flower_data_set

https://nguyenvanhieu.vn/tai-lieu-machine-learning-co-ban/