|  |  |  |
| --- | --- | --- |
|  | BAN CƠ YẾU CHÍNH PHỦ  “HỌC VIỆN KỸ THUẬT MẬT MÃ” | Mẫu 2 |

BÁO CÁO CHUYÊN ĐỀ SỐ 2.2

“Nghiên cứu một số thuật toán học máy tiêu biểu RF, KNN, SVM”

NHIỆM VỤ: “Nghiên cứu và ứng dụng nền tảng học sâu để xây dựng hệ thống phát hiện mã độc trực tuyến”.

Mã số: 06/2022/CB.

Cơ quan chủ trì: Học viện Kỹ thuật Mật mã

Chủ nhiệm: ThS. Lê Đức Thuận

Hà Nội - 2022

|  |  |  |
| --- | --- | --- |
|  | BAN CƠ YẾU CHÍNH PHỦ  “HỌC VIỆN KỸ THUẬT MẬT MÃ” |  |

BÁO CÁO CHUYÊN ĐỀ SỐ 2.2

“Nghiên cứu một số thuật toán học máy tiêu biểu RF, KNN, SVM”

NHIỆM VỤ: “Nghiên cứu và ứng dụng nền tảng học sâu để xây dựng hệ thống phát hiện mã độc trực tuyến”.

Mã số: 06/2022/CB.

Cơ quan chủ trì: Học viện Kỹ thuật Mật mã

Chủ nhiệm: ThS. Lê Đức Thuận

|  |  |
| --- | --- |
| **Người thực hiện chuyên đề** | **Cơ quan chủ trì** |
| *(Họ tên và chữ ký)* | *(Họ tên và chữ ký)* |

Hà Nội - 2022

MỤC LỤC

[MỤC LỤC 3](#_Toc115870760)

[DANH MỤC HÌNH ẢNH 4](#_Toc115870761)

[1. Machine Learning 5](#_Toc115870762)

[1.1. Định nghĩa 5](#_Toc115870763)

[1.2. Một số phương thức của Machine Learning 5](#_Toc115870764)

[1.3. Bài toán phân lớp dữ liệu 6](#_Toc115870765)

[1.3.1. Quá trình phân lớp dữ liệu 6](#_Toc115870766)

[2. K-nearest neighbors (KNN) 8](#_Toc115870767)

[2.1. Sử dụng thuật toán KNN 8](#_Toc115870768)

[2.2. Cách xác định giá trị *k* tối ưu 11](#_Toc115870769)

[2.3. Ưu và nhược điểm của KNN 12](#_Toc115870770)

[2.4. Về ứng dụng của thuật toán KNN phải kể đến như: 12](#_Toc115870771)

[3. Random Forests (RF) 14](#_Toc115870772)

[3.1. Tìm hiểu về Random Forest 14](#_Toc115870773)

[3.2. Cách thức hoạt động của thuật toán RF 15](#_Toc115870774)

[3.3. Ưu nhược điểm của thuật toán 17](#_Toc115870775)

[3.4. Các tính năng quan trọng 18](#_Toc115870776)

[3.5. Ứng dụng 18](#_Toc115870777)

[4. Support Vector Machines (SVM) 20](#_Toc115870778)

[4.1. Các vector hỗ trợ 21](#_Toc115870779)

[4.2. Huấn luyện SVM 25](#_Toc115870780)

# DANH MỤC HÌNH ẢNH

[Hình 2.1. Hoạt động của thuật toán KNN 9](#_Toc115870746)

[Hình 2.2. Thêm điểm dũ liệu mới 10](#_Toc115870747)

[Hình 2.3: Biểu diễn và tính khoảng cách hai điểm 10](#_Toc115870748)

[Hình 2.4: Mô hình hoá hàng xóm gần nhất của điểm 11](#_Toc115870749)

[Hình 3.1. Mô hình hoá Random Forest 15](#_Toc115870750)

[Hình 3.2. Dự đoán kết quả trong RF 15](#_Toc115870751)

[Hình 3.3: Cách thứchoạt động của mô hình Random Forest 16](#_Toc115870752)

[Hình 4.1: Mô phỏng phân lớp dựa trên SVM 21](#_Toc115870753)

[Hình 4.2: Siêu mặt phẳng phân lớp dữ liệu 21](#_Toc115870754)

[Hình 4.3: Vector hỗ trợ siêu mặt phẳng 22](#_Toc115870755)

[Hình 4.4: Biên phân tách 23](#_Toc115870756)

[Hình 4.5: Các giá trị biên trong SVM 24](#_Toc115870757)

[Hình 4.6: Biểu diễn trực quan trên trục đồ thì 24](#_Toc115870758)

[Hình 4.7: Phân tách bằng đồ thị phức tạp 25](#_Toc115870759)

# 1. Machine Learning

## 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 không và tự động sắp xếp vào các thư mục tương ứng.

Machine Learning 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.

Machine Learning có hiện nay được áp dựng rộng rãi bao gồm máy truy tìm dữ liệu, máy phân tích thị trường chứng khoán, nhận dạng tiếng nói và chữ viết, v.v.

## 1.2. Một số phương thức của Machine Learning

* 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). Supervised learning là nhóm phổ biến nhất trong các thuật toán của Machine Learning.

Học có giám sát (Supervised learning) được chia thành hai 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 có gán nhãn (hay còn gọi là tập huấn luyện). Quá trình phân lớp là quá trình gán nhãn cho đối tượng dữ liệu.

Có nhiều bài toán phân lớp như phân lớp nhị phân, phân lớp đa lớp, 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 (Unsupervised learning): 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 trước 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.

## 1.3. Bài toán phân lớp dữ liệu

### 1.3.1. 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 và rút trích đặc trưng.

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 nhiều 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 hai phần, một phần phục vụ cho traning 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

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

*F(x) = y*

Trong đó: x là các feature hay input đầu vào của dữ liệu

y là nhãn dán lớp hay output đầu ra

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

* **Bước 3:** Kiểm tra dữ liệu với mô hình

Sau khi tìm được mô hình phân lớp ở bước 2, 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.

# 

# 2. K-nearest neighbors (KNN)

K-nearest neighbor là một trong những thuật toán học máy có giám sát 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. KNN có thể áp dụng được vào cả hai loại của bài toán phân loại (classification) và hồi quy (regression).

Với KNN, trong bài toán Classification, nhãn (label) của một dữ liệu mới được suy ra từ K điểm dữ liệu gần nhất trong tập dữ liệu tập huấn (training set). Nhãn của một tập dữ liệu kiểm thử (test set) có thể được quyết định bằng bầu chọn theo số phiếu (major voting) giữa các điểm gần nhất đó rồi suy ra nhãn.

Trong bài toán Regression, 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.

Nói 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 tập dữ liệu tập huấn 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.

## 2.1. Sử dụng thuật toán KNN

Giả sử có hai danh mục, tức là Danh mục A và Danh mục B, và chúng ta có một điểm dữ liệu mới x1, vậy điểm dữ liệu này sẽ nằm trong danh mục nào trong số các danh mục này. Để giải quyết dạng bài toán này, chúng ta cần một thuật toán KNN. Với sự trợ giúp của KNN, chúng ta có thể dễ dàng xác định danh mục hoặc lớp của một tập dữ liệu cụ thể. Sơ đồ:



Hình 2.1. Hoạt động của thuật toán KNN

Hoạt động KNN có thể được giải thích dựa trên thuật toán dưới đây:

* Bước 1: Chọn số K = số láng giềng gần nhất
* Bước 2: Tính khoảng cách Euclide của K số láng giềng
* Bước 3: Lấy K láng giềng gần nhất theo khoảng cách Euclide được tính toán
* Bước 4: Trong số K lân cận này, đếm số điểm dữ liệu trong mỗi loại
* Bước 5: Gán các điểm dữ liệu mới cho danh mục đó mà số lượng hàng xóm là tối đa

Giả sử có một điểm dữ liệu mới và cần đặt nó vào danh mục bắt buộc như trong hình dưới đây:



Hình 2.2. Thêm điểm dũ liệu mới

Đầu tiên, ta sẽ chọn số lượng hàng xóm, ví dụ chọn k = 5.

Tiếp theo, ta tính toán khoảng cách Euclide giữa các điểm dữ liệu. Khoảng cách Euclide là khoảng cách giữa hai điểm mà chúng ta đã học trong hình học, được tính bằng công thức:



Hình 2.3: Biểu diễn và tính khoảng cách hai điểm

Bằng cách tính toán khoảng cách Euclide, chúng ta có được những người hàng xóm gần nhất trong loại A và hai người hàng xóm gần nhất trong loại B như hình dưới đây:



Hình 2.4: Mô hình hoá hàng xóm gần nhất của điểm

Ta thấy, 3 láng giềng gần nhất thuộc loại A, do đó điểm dữ liệu mới này phải thuộc loại A.

## 2.2. Cách xác định giá trị *k* tối ưu

Một số điểm cần nhớ khi chọn giá trị của k:

* k = 1: Mô hình quá cụ thể và không khái quát hóa tốt. Nó cũng có xu hướng nhạt cảm với các dữ liệu bất thường (nhiễu) trong tập dữ liệu. Mô hình đạt được độ chính xác cao trên dữ liệu đào tạo nhưng sẽ là một công cụ dự đoán kém đối với các điểm dữ liệu mới chưa được gặp trước đây. Do đó, chúng ta có thể sẽ tạo ra một mô hình quá khớp (overfit).
* K = 100: Ngược lại với k = 1, mô hình quá tổng quát và không phải là một công cụ dự báo tốt trên cả tập dữ liệu đào tạo và tập dữ liệu kiểm tra. Tình trạng này được gọi là mô hình chưa khớp (underfit).

Không có cách cụ thể nào để xác định giá trị tốt nhất cho “k”, vì vậy chúng ta cần thử một số giá trị để tìm ra giá trị tốt nhất trong số đó. [Scikit-learning](https://vncoder.vn/bai-hoc/machine-learning-thu-vien-scikit-learn-382) cung cấp chức năng GridSearchCV cho phép ta dễ dàng kiểm tra nhiều giá trị k khác nhau.

## 2.3. Ưu và nhược điểm của KNN

* **Ưu điểm:**
* Đơn giản và dễ giải thích
* Độ phức tạp tính toán của quá trình training bằng 0
* Không dựa trên bất kỳ giả định nào, vì thế nó có thể được sử dụng trong các bài toán phi tuyến tính
* Việc dự đoán kết quả của dữ liệu mới rất đơn giản
* Hoạt động tốt trong trường hợp phân loại với nhiều lớp
* Sử dụng được trong cả phân loại và hồi quy
* **Nhược điểm:**
* Tốn bộ nhớ
* Nhạy cảm với các dữ liệu bất thường (nhiễu khi k nhỏ)
* Mọi tính toán đều nằm ở khâu test. Trong đó việc tính khoảng cách tới từng điểm dữ liệu trong traning 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.

## ****2.4. Về ứng dụng của thuật toán KNN phải kể đến như:****

* **Trong y tế:** xác định bệnh lý của người bệnh mới dựa trên dữ liệu lịch sử của các bệnh nhân có cùng bệnh lý có cùng các đặc điểm đã được chữa khỏi trước đây, hay xác định loại thuốc phù hợp giống ví dụ chúng tôi trình bày ở trên.
* **Trong lĩnh vực ngân hàng:** xác định khả năng khách hàng chậm trả các khoản vay hoặc rủi ro tín dụng do nợ xấu dựa trên phân tích Credit score; xác định xem liệu các giao dịch có hành vi phạm tội, lừa đảo hay không.
* **Trong giáo dục:** phân loại các học sinh theo hoàn cảnh, học lực để xem xem cần hỗ trợ gì cho những học sinh ví dụ như hoàn cảnh sống khó khăn nhưng học lực lại tốt.
* **Trong thương mại điện tử:** phân loại khách hàng theo sở thích cụ thể để hỗ trợ personalized marketing hay xây dựng hệ thống khuyến nghị, dựa trên dữ liệu từ website, social media.
* **Trong kinh tế nói chung:** giúp dự báo các sự kiện kinh tế trong tương lai, dự báo tình hình thời tiết trong nông nghiệp, xác định xu hướng thị trường chứng khoán để lên kế hoạch đầu tư thích hợp.

# 

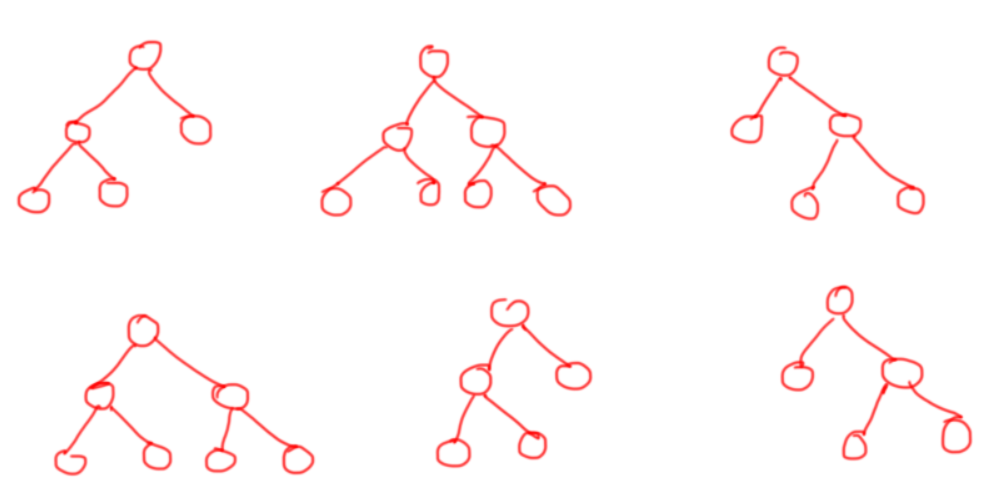
# 3. Random Forests (RF)

## 3.1. Tìm hiểu về Random Forest

Random Forests là thuật toán học có giám sát (supervised learning). Nó có thể được sử dụng cho cả bài toán phân lớp và hồi quy. Nó cũng là thuật toán linh hoạt và dễ sử dụng nhất. Một khu rừng bao gồm cây cối. Người ta nói rằng càng có nhiều cây thì rừng càng mạnh. Random forests tạo ra [cây quyết định](https://machinelearningcoban.com/2018/01/14/id3/) trên các mẫu dữ liệu được chọn ngẫu nhiên, được dự đoán từ mỗi cây và chọn giải pháp tốt nhất bằng cách bỏ phiếu. Nó cũng cung cấp một chỉ báo khá tốt về tầm quan trọng của tính năng. Random forests có nhiều ứng dụng, chẳng hạn như công cụ đề xuất, phân loại hình ảnh và lựa chọn tính năng. Nó có thể được sử dụng để phân loại các ứng viên cho vay trung thành, xác định hoạt động gian lận và dự đoán các bệnh. Nó nằm ở cơ sở của thuật toán Boruta, chọn các tính năng quan trọng trong tập dữ liệu.

Về mặt kỹ thuật, nó là một phương pháp tổng hợp (dựa trên cách tiếp cận phân chia và chinh phục) của các cây quyết định được tạo ra trên một tập dữ liệu được chia ngẫu nhiên. Bộ sưu tập phân loại cây quyết định này còn được gọi là rừng. Cây quyết định riêng lẻ được tạo ra bằng cách sử dụng chỉ báo chọn thuộc tính như tăng thông tin, tỷ lệ tăng và chỉ số Gini cho từng thuộc tính. Mỗi cây phụ thuộc vào một mẫu ngẫu nhiên độc lập. Trong bài toán phân loại, mỗi phiếu bầu chọn và lớp phổ biến nhất được chọn là kết quả cuối cùng. Trong trường hợp hồi quy, mức trung bình của tất cả các kết quả đầu ra của cây được coi là kết quả cuối cùng. Nó đơn giản và mạnh mẽ hơn so với các thuật toán phân loại phi tuyến tính khác.

Ở bước huấn luyện thì ta sẽ xây dựng nhiều cây quyết định, các cây quyết định có thể khác nhau.



Hình 3.1. Mô hình hoá Random Forest

A picture containing diagram

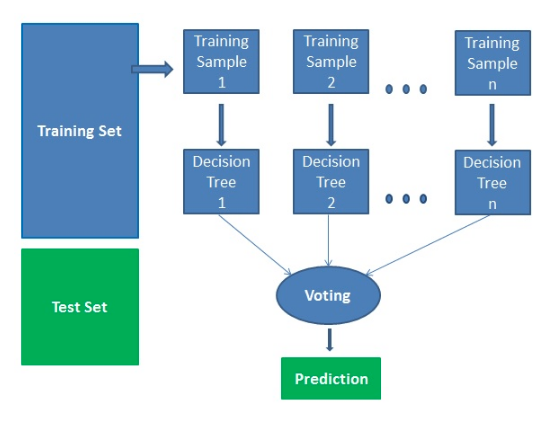
Description automatically generatedSau đó ở bước dự đoán, với một dữ liệu mới, thì ở mỗi cây quyết định ta sẽ đi từ trên xuống theo các node điều kiện để được các dự đoán, sau đó kết quả cuối cùng được tổng hợp từ kết quả của các cây quyết định.

Hình 3.2. Dự đoán kết quả trong RF

Ví dụ như trên, thuật toán Random Forest có 6 cây quyết định, 5 cây dự đoán 1 và 1 cây dự đoán 0, do đó cho ra dự đoán cuối cùng là 1.

## 3.2. Cách thức hoạt động của thuật toán RF

Random Forest hoạt động theo bốn bước:



Hình 3.3: Cách thứchoạt động của mô hình Random Forest

1. Chọn các mẫu ngẫu nhiên từ tập dữ liệu đã cho.
2. Thiết lập cây quyết định cho từng mẫu và nhận kết quả dự đoán từ mỗi quyết định cây.
3. Hãy bỏ phiếu cho mỗi kết quả dự đoán.
4. Chọn kết quả được dự đoán nhiều nhất là dự đoán cuối cùng.

Do quá trình xây dựng mỗi cây quyết định đều có yếu tố ngẫu nhiên (random) nên kết quả là các cây quyết định trong thuật toán Random Forest có thể khác nhau.

Thuật toán Random Forest sẽ bao gồm nhiều cây quyết định, mỗi cây được xây dựng dùng thuật toán Decision Tree trên tập dữ liệu khác nhau và dùng tập thuộc tính khác nhau. Sau đó kết quả dự đoán của thuật toán Random Forest sẽ được tổng hợp từ các cây quyết định.

Khi dùng thuật toán Random Forest, ta hay để ý các thuộc tính như: số lượng cây quyết định sẽ xây dựng, số lượng thuộc tính dùng để xây dựng cây. Ngoài ra, vẫn có các thuộc tính của thuật toán Decision Tree để xây dựng cây như độ sâu tối đa, số phần tử tối thiểu trong 1 node để có thể tách.

## 3.3. Ưu nhược điểm của thuật toán

* **Ưu điểm:**
* Trong thuật toán Decision Tree, khi xây dựng cây quyết định nếu để độ sâu tùy ý thì cây sẽ phân loại đúng hết các dữ liệu trong tập training dẫn đến mô hình có thể dự đoán tệ trên tập validation/test, khi đó mô hình bị overfitting, hay nói cách khác là mô hình có [high variance](https://viblo.asia/p/the-bias-variance-decomposition-eW65Gm3YZDO).
* Thuật toán Random Forest gồm nhiều cây quyết định, mỗi cây quyết định đều có những yếu tố ngẫu nhiên:
  + Lấy ngẫu nhiên dữ liệu để xây dựng cây quyết định.
  + Lấy ngẫu nhiên các thuộc tính để xây dựng cây quyết định.
* Do mỗi cây quyết định trong thuật toán Random Forest không dùng tất cả dữ liệu training, cũng như không dùng tất cả các thuộc tính của dữ liệu để xây dựng cây nên mỗi cây có thể sẽ dự đoán không tốt, khi đó mỗi mô hình cây quyết định không bị overfitting mà có thế bị underfitting, hay nói cách khác là mô hình có high bias. Tuy nhiên, kết quả cuối cùng của thuật toán Random Forest lại tổng hợp từ nhiều cây quyết định, thế nên thông tin từ các cây sẽ bổ sung thông tin cho nhau, dẫn đến mô hình có low bias và low variance, hay mô hình có kết quả dự đoán tốt.
* **Nhược điểm:**
* Random forest chậm tạo dự đoán bởi vì nó có nhiều cây quyết định. Bất cứ khi nào nó đưa ra dự đoán, tất cả các cây trong rừng phải đưa ra dự đoán cho cùng một đầu vào cho trước và sau đó thực hiện bỏ phiếu trên đó. Toàn bộ quá trình này tốn thời gian.
* Mô hình khó hiểu hơn so với cây quyết định, nơi bạn có thể dễ dàng đưa ra quyết định bằng cách đi theo đường dẫn trong cây.

## 3.4. Các tính năng quan trọng

Random forests cũng cung cấp một chỉ số lựa chọn tính năng tốt. Scikit-learn cung cấp thêm một biến với mô hình, cho thấy tầm quan trọng hoặc đóng góp tương đối của từng tính năng trong dự đoán. Nó tự động tính toán điểm liên quan của từng tính năng trong giai đoạn đào tạo. Sau đó, nó cân đối mức độ liên quan xuống sao cho tổng của tất cả các điểm là 1.

Điểm số này sẽ giúp bạn chọn các tính năng quan trọng nhất và thả các tính năng quan trọng nhất để xây dựng mô hình.

Random forests sử dụng tầm quan trọng của gini hoặc giảm tạp chất trung bình (MDI) để tính toán tầm quan trọng của từng tính năng. Gini tầm quan trọng còn được gọi là tổng giảm trong tạp chất nút. Đây là mức độ phù hợp hoặc độ chính xác của mô hình giảm khi bạn thả biến. Độ lớn càng lớn thì biến số càng có ý nghĩa. Ở đây, giảm trung bình là một tham số quan trọng cho việc lựa chọn biến. Chỉ số Gini có thể mô tả sức mạnh giải thích tổng thể của các biến. Random Forests và cây quyết định Random Forests là một tập hợp của nhiều cây quyết định. Cây quyết định sâu có thể bị ảnh hưởng quá mức, nhưng Random forests ngăn cản việc lấp đầy bằng cách tạo cây trên các tập con ngẫu nhiên. Cây quyết định nhanh hơn tính toán. Random forests khó giải thích, trong khi cây quyết định có thể diễn giải dễ dàng và có thể chuyển đổi thành quy tắc.

## 3.5. Ứng dụng

Thuật toán rừng ngẫu nhiên được sử dụng trong rất nhiều lĩnh vực khác nhau, như ngân hàng, thị trường chứng khoán, y học và thương mại điện tử.

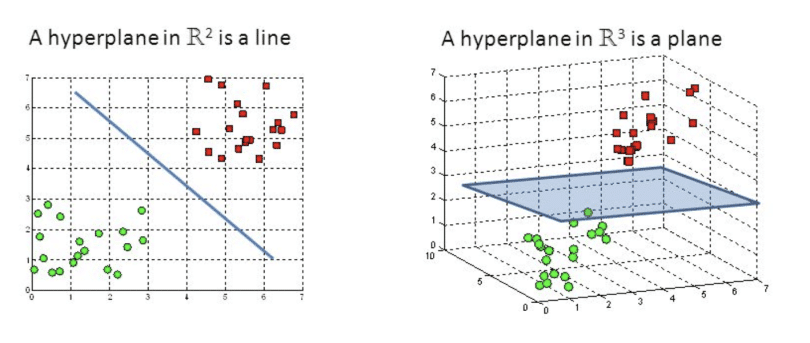
* Ví dụ, trong tài chính, nó được sử dụng để phát hiện khách hàng có nhiều khả năng trả nợ đúng hạn hoặc sử dụng dịch vụ của ngân hàng thường xuyên hơn. Trong tên miền này, nó cũng được sử dụng để phát hiện những kẻ lừa đảo ra ngoài để lừa đảo ngân hàng. Trong giao dịch, thuật toán có thể được sử dụng để xác định hành vi trong tương lai của một cổ phiếu.
* Trong chăm sóc sức khỏe, nó được sử dụng để xác định sự kết hợp chính xác của các thành phần trong y học và phân tích lịch sử y tế của bệnh nhân để xác định bệnh.
* Rừng ngẫu nhiên được sử dụng trong thương mại điện tử để xác định xem khách hàng có thực sự thích sản phẩm hay không.

# 4. Support Vector Machines (SVM)

Trong các nghiên cứu thời gian gần đây, phương pháp phân lớp dùng tập phân lớp vector hỗ trợ đang được nghiên cứu và áp dụng tương đối mạnh mẽ trong lĩnh vực phân lớp và nhận dạng. SVM là phương pháp được ra đời từ lý thuyết học thống kê do Vapnik và Chervonenkis nghiên cứu và phát triển. Đây là phương pháp có nhiều tiềm năng phát triển về trong thực tiễn cũng như trong các nghiên cứu lý thuyết. Từ kết quả của những nghiên cứu gần đây cho thấy rằng SVM là phương pháp có khả năng phân lớp khá tốt không chỉ đối với bài toán phân lớp dữ liệu văn bản mà còn trong nhiều ứng dụng như nhận dạng văn bản viết tay, phát hiện mặt người khung hình… Khả năng phân lớp của phương pháp SVM được đánh giá là khá cao so với các phương pháp khác.

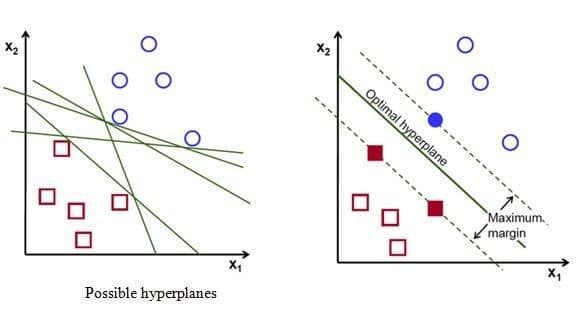
SVM sử dụng thuật toán học với mục tiêu tìm ra một mặt siêu phẳng làm nhỏ nhất có thể độ phân lớp sai cho một đối tượng dữ liệu mới. Độ phân lớp sai của một mặt siêu phẳng được đặc trưng bởi khoảng cách từ điểm gần nhất tới siêu phẳng đấy. Đặc trưng thiết yếu thể hiện khả năng của mô hình phân lớp chính là khả năng phân lớp những dữ liệu mới sau quá trình huấn luyện. Phương pháp huấn luyện sẽ được đánh giá là tốt nếu hiệu suất tổng quát hoá của mô hình phân lớp cao và ngược lại phương pháp huấn luyện sẽ được đánh giá là chưa tốt nếu hiệu suất tổng quát hóa của mô hình là thấp. Hiệu suất tổng quát hoá ở đây phụ thuộc vào hai yếu tố là năng lực của máy học và sai số huấn luyện. Trong các tham số này thì sai số huấn luyện được hiểu là tỷ lệ sai lỗi của quá trình phân lớp trên tập dữ liệu huấn luyện. Còn yếu tố thứ hai năng lực của máy học được xác định bằng kích thước VC (Vapnik-Chervonenkis). Đây được coi là một khái niệm quan trọng đối với một mô hình phân lớp. Kích thước VC được tính bởi số điểm cực đại mô hình phân lớp có thể tách trong không gian đối tượng cần phân loại.

Siêu phẳng tạo ra một biên giới phân chia 2 lớp của dữ liệu.



Hình 4.1: Mô phỏng phân lớp dựa trên SVM

Để phân chia 2 lớp dữ liệu, rõ ràng là có rất nhiều siêu phẳng có thể làm được điều này. Mặc dù vậy, mục tiêu của chúng ta là tìm ra siêu phẳng có lề rộng nhất, tức là có khoảng cách tới các điểm của 2 lớp là lớn nhất, ví dụ như hình dưới đây:

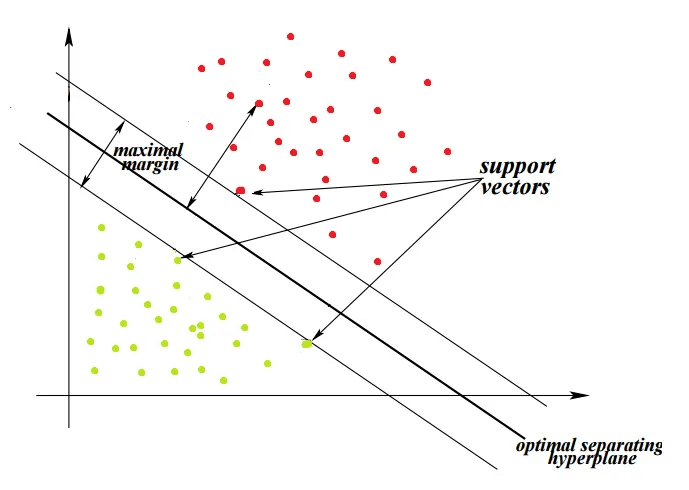


Hình 4.2: Siêu mặt phẳng phân lớp dữ liệu

Lưu ý: Số chiều của siêu phẳng phụ thuộc vào số đặc trưng.

## 4.1. Các vector hỗ trợ

Một điểm trong không gian vector có thể được coi là một vector từ gốc tọa độ tới điểm đó. Các điểm dữ liệu nằm trên hoặc gần nhất với siêu phẳng được gọi là vector hỗ trợ, chúng ảnh hưởng đến vị trí và hướng của siêu phẳng. Các vector này được sử dụng để tối ưu hóa lề và nếu xóa các điểm này, vị trí của siêu phẳng sẽ thay đổi. Một điểm lưu ý nữa đó là các vector hỗ trợ phải cách đều siêu phẳng.



Hình 4.3: Vector hỗ trợ siêu mặt phẳng

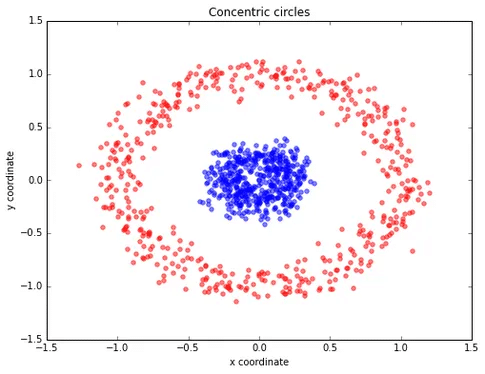
SVM chỉ có thể hoạt độgn trên dữ liệu có thể phân chia tuyến tính.

Nếu dữ liệu không thể phân chia tuyến tính thì sao?

**Ví dụ:** Nhìn vào hình ảnh bên dưới, dữ liệu được phân tách phi tuyến tính, rõ ràng ta không thể vẽ một đường thẳng để phân loại các điểm dữ liệu đỏ và xanh. Để giải quyết vấn đề này, có 2 giải pháp:

1. Lề mềm (Soft margin)

2. Thủ thuật Kernel (Kernel tricks)



Hình 4.4: Biên phân tách

Thuật toán này cho phép SVM mắc một số lỗi nhất định và giữ cho lề càng rộng càng tốt để các điểm khác vẫn có thể được phân loại chính xác. Nói một cách khác, nó cân bằng giữa việc phân loại sai và tối đa hóa lề.

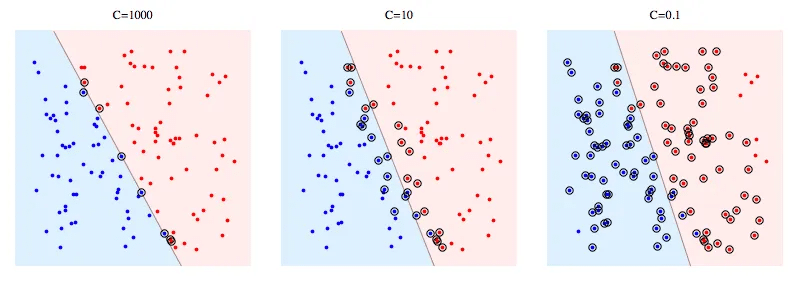
Có 2 kiểu phân loại sai có thể xảy ra:

1. Dữ liệu nằm ở đúng bên nhưng phạm vào lề

2. Dữ liệu nằm ở sai bên

Mức độ chấp nhận lỗi

Mức độ chấp nhận lỗi là một siêu tham số quan trọng trong SVM. Khi lập trình với sklearn, mức độ chấp nhận lỗi được coi như một tham số phạt (C). Hình dưới thể hiện SVM với [các giá trị C khác nhau](https://machinelearningcoban.com/2017/04/13/softmarginsmv/#-anh-huong-cua-\c\-len-nghiem).



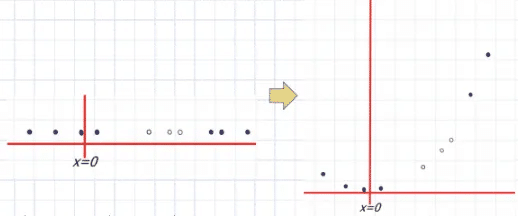
Hình 4.5: Các giá trị biên trong SVM

C càng lớn có nghĩa là SVM càng bị phạt nặng khi thực hiện phân loại sai. Do đó, lề càng hẹp và càng ít vector hỗ trợ được sử dụng.

[Thủ thuật Kernel](https://machinelearningcoban.com/2017/04/22/kernelsmv/)

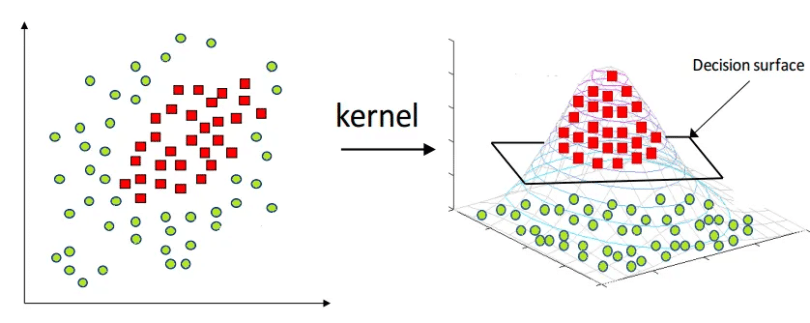
Một kernel là một hàm ánh xạ dữ liệu từ không gian ít nhiều hơn sang không gian nhiều chiều hơn, tự đó ta tìm được siêu phẳng phân tách dữ liệu. Một cách trực quan, kỹ thuật này giống như việc bạn gập tờ giấy lại để có thể dùng kéo cắt một lỗ tròn trên nó.

Biểu diễn trực quan của thủ thuật kernel:



Hình 4.6: Biểu diễn trực quan trên trục đồ thì

Ví dụ



Hình 4.7: Phân tách bằng đồ thị phức tạp

Các kiểu Kernel:

1. Tuyến tính

2. Đa thức

3. RBF

4. Sigmoid

## 4.2. Huấn luyện SVM

Để huấn luyện máy hỗ trợ vector thực chất chúng ta sẽ tiến hành giải bài toán quy hoạch toàn phương Support Vector Machine. Để giải bài toán này chúng ta có thể dùng các phương pháp số. Cụ thể các phương pháp này cần một ma trận kích thước bằng bình phương của số lượng mẫu dùng trong việc training. Điều này trong thực tế đôi khi không khả thi vì kích thước của tập dữ liệu dùng để training thường rất lớn (thậm chí có thể đến hàng chục nghìn mẫu huấn luyện). Nhằm giải quyết vấn đề nêu trên người ta đã phát triển nhiều thuật toán khác nhau dựa trên việc phân rã tập training thành những nhóm dữ liệu. Lúc này bài toán quy hoạch toàn phương sẽ được giải với kích thước nhỏ hơn. Sau đó, những thuật toán này sẽ kiểm tra các điều kiện Karush KuhnTucker để tìm ra được phương án tối ưu nhất. Một vài phương pháp huấn luyện dựa vào tính chất: Nếu trong tập training của bài toán quy hoạch toàn phương Support Vector Machine con (bài toán nhỏ) có ít nhất một mẫu vi phạm vào điều kiện Karush KuhnTucker, thì bài toán này sau khi được giải, hàm mục tiêu sẽ tăng. Một loạt các bài toán quy hoạch toàn phương Support Vector Machine con với ít một mẫu nào đó vi phạm các điều kiện Karush KuhnTucker được đảm bảo sẽ sự hội tụ đến phương án tối ưu.

**Ưu điểm của SVM đó là:**

* Đây là thuật toán hoạt động hiệu quả với không gian cao chiều (high dimensional spaces).
* Thuật toán tiêu tốn ít bộ nhớ vì chỉ sử dụng các điểm trong tập hỗ trợ để dự báo trong hàm quyết định.
* Chúng ta có thể tạo ra nhiều hàm quyết định từ những hàm kernel khác nhau. Thậm chí sử dụng đúng kernel có thể giúp cải thiện thuật toán lên đáng kể.

Chính vì tính hiệu quả mà SVM thường được áp dụng nhiều trong các tác vụ phân loại và dự báo, cũng như được nhiều công ty ứng dụng và triển khai trên môi trường production. Chúng ta có thể liệt kê một số ứng dụng của thuật toán SVM đó là:

* Mô hình chuẩn đoán bệnh. Dựa vào biến mục tiêu là những chỉ số xét nghiệm lâm sàng, thuật toán đưa ra dự báo về một số bệnh như tiểu đường, suy thận, máu nhiễm mỡ,…
* Trước khi thuật toán CNN và Deep Learning bùng nổ thì SVM là lớp mô hình cực kì phổ biến trong phân loại ảnh.
* Mô hình phân loại tin tức. Xác định chủ đề của một đoạn văn bản, phân loại cảm xúc văn bản, phân loại thư rác.
* Mô hình phát hiện gian lận.