

**TRƯỜNG ĐẠI HỌC MỞ HÀ NỘI**

**KHOA CÔNG NGHỆ THÔNG TIN**

***----------------------***

******

**BÁO CÁO BÀI TẬP LỚN**

**MÔN: NHẬP MÔN KHAI PHÁ DỮ LIỆU VÀ MÁY HỌC**

**ĐỀ TÀI:**

**DỰ ĐOÁN KHẢ NĂNG XẢY RA HỎA HOẠN**

Giảng viên hướng dẫn : TS. Vũ Xuân Hạnh

Nhóm sinh viên thực hiện :Đỗ Quang Hiệp – 2110A02

Nguyễn Thành Đức – 2110A02

Phạm Tuấn Hà – 2110A02

Nguyễn Trường Giang – 2110A02

***Hà Nội - 2024***

**PHÂN CÔNG CÔNG VIỆC VÀ KẾ HOẠCH THỰC HIỆN**

|  |  |  |
| --- | --- | --- |
| **Họ và tên** | **Công việc** | **Chú ý** |
| Đỗ Quang Hiệp | Tổng quan về đề tài, Giảm chiều dữ liệu (Tiền xử lý dữ liệu), Đánh giá mô hình học máy, Hoàn thiện báo cáo. |  |
| Nguyễn Thành Đức | Tổng quan về đề tài, Xây dựng mô hình học máy (NativeBayes và Random Forest), Thu thập dữ liệu (Tiền xử lý dữ liệu), Tích hợp mô hình học máy vào ứng dụng, Hoàn thiện báo cáo. |  |
| Nguyễn Trường Giang | Tổng quan về đề tài, Làm sạch dữ liệu (Tiền xử lý dữ liệu), Đánh giá mô hình học máy, Hoàn thiện báo cáo. |  |
| Phạm Tuấn Hà | Tổng quan về đề tài, Xây dựng mô hình học máy (Cây quyết định), Đánh giá mô hình học máy, Hoàn thiện báo cáo. |  |

**MỤC LỤC**

[I. TỔNG QUAN VỀ ĐỀ TÀI 1](#_Toc164627200)

[1. Lý do chọn đề tài 1](#_Toc164627201)

[2. Tổng quan về Học máy 2](#_Toc164627202)

[3. Chu trình Học máy 2](#_Toc164627203)

[4. Học máy có giám sát 4](#_Toc164627204)

[II. TIỀN XỬ LÝ DỮ LIỆU 4](#_Toc164627205)

[1. Xác định vấn đề 4](#_Toc164627206)

[2. Xác định mục đích 5](#_Toc164627207)

[3. Đặc điểm dữ liệu 5](#_Toc164627208)

[4. Phương pháp thu thập dữ liệu 6](#_Toc164627209)

[5. Tiền xử lý dữ liệu 6](#_Toc164627210)

[5.1. Làm sạch dữ liệu 6](#_Toc164627211)

[5.2. Giảm chiều dữ liệu 7](#_Toc164627212)

[III. XÂY DỰNG MÔ HÌNH HỌC MÁY 9](#_Toc164627213)

[1. Naive Bayes 9](#_Toc164627214)

[1.1. Cơ sở lý thuyết 9](#_Toc164627215)

[1.2. Vì sao chọn Native Bayes cho bài toán 12](#_Toc164627216)

[1.3. Áp dụng cơ sở lý thuyết vào xây dựng mô hình cho bài toán 13](#_Toc164627217)

[2. Decision Tree (Cây quyết định) 13](#_Toc164627218)

[2.1. Cơ sở lý thuyết 13](#_Toc164627219)

[2.2. Vì sao chọn Decision Tree cho bài toán 17](#_Toc164627220)

[2.3. Áp dụng cơ sở lý thuyết vào xây dựng mô hình cho bài toán 18](#_Toc164627221)

[3. Random Forest 18](#_Toc164627222)

[3.1. Cơ sở lý thuyết 18](#_Toc164627223)

[3.2. Vì sao chọn Decision Tree cho bài toán 22](#_Toc164627224)

[3.3. Áp dụng cơ sở lý thuyết vào xây dựng mô hình cho bài toán 22](#_Toc164627225)

[IV. ĐÁNH GIÁ MÔ HÌNH HỌC MÁY 22](#_Toc164627226)

[V. TÍCH HỢP MÔ HÌNH HỌC MÁY VÀO ỨNG DỤNG 27](#_Toc164627227)

[VI. KẾT LUẬN 29](#_Toc164627228)

[TÀI LIỆU THAM KHẢO 30](#_Toc164627229)

1. **TỔNG QUAN VỀ ĐỀ TÀI**
   * + 1. **Lý do chọn đề tài**

Học máy là môn khoa học nhằm phát triển những thuật toán và mô hình thống kê mà các hệ thống máy tính sử dụng để thực hiện các tác vụ dựa vào khuôn mẫu và suy luận mà không cần hướng dẫn cụ thể. Các hệ thống máy tính sử dụng thuật toán máy học để xử lý khối lượng lớn dữ liệu trong quá khứ và xác định các khuôn mẫu dữ liệu. Việc này cho phép chúng dự đoán kết quả chính xác hơn từ cùng một tập dữ liệu đầu vào cho trước.

Việc chọn đề tài dự đoán phát hiện khả năng xảy ra hỏa hoạn là một quyết định mang tính chiến lược và cần thiết vì nhiều lý do. Trước hết, việc phát hiện khả năng xảy ra hỏa hoạn sớm có thể giúp ngăn chặn sự lan rộng của đám cháy, giảm thiểu thiệt hại về người và tài sản. Điều này đặc biệt quan trọng trong các môi trường như nhà máy, nhà kho, hoặc các tòa nhà có nhiều người sinh sống hoặc làm việc.

Thứ hai, dự đoán phát hiện khả năng xảy ra hỏa hoạn cũng giúp tăng cường sự an toàn và bảo vệ môi trường. Bằng cách sử dụng công nghệ và thuật toán hiện đại, ta có thể xác định các dấu hiệu sớm của hỏa hoạn trước khi nó lan rộng và gây ra các vấn đề nghiêm trọng.

Thêm vào đó, việc nghiên cứu và phát triển các hệ thống dự đoán khả năng xảy ra hỏa hoạn cũng đóng vai trò quan trọng trong việc nâng cao hiệu quả hoạt động của lực lượng cứu hỏa và cứu thương. Thông qua việc cung cấp thông tin chính xác và kịp thời, các đội cứu hỏa có thể tổ chức và triển khai kế hoạch phản ứng một cách hiệu quả, giảm thiểu thời gian phản ứng và tăng cơ hội cứu sống người bị nạn.

Từ những lý do ở trên, chúng em quyết định lựa chọn đề tài ***“Dự đoán khả năng xảy ra hỏa hoạn*** ***dựa trên phương pháp học máy có giám sát”.***

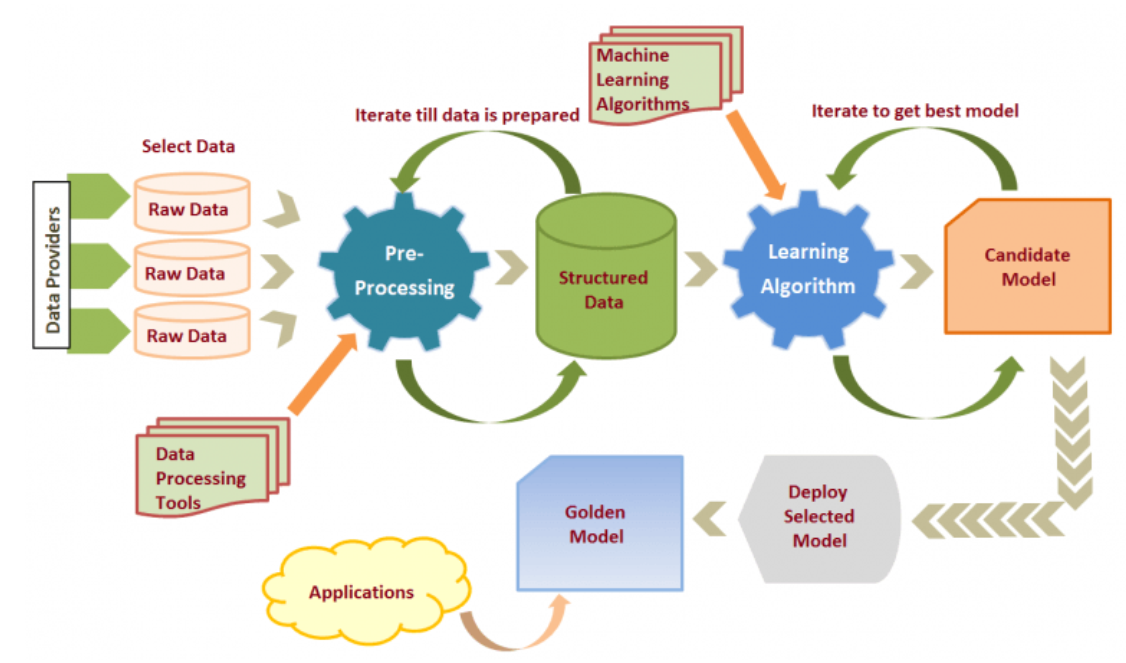
* + - 1. **Tổng quan về Học máy**

Học máy (machine learning) 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 những vấn đề cụ thể.

Một cách tổng quát, trong cuốn sách Machine Learning của tác giả Tom Mitchell xuất bản năm 1997, học máy được định nghĩa như sau: “A computer program is said to learn to perform a task T from experience E, if its performance at task T, as measured by a performance metric P, improves with experience E over time” (một chương trình máy tính được cho là học để thực hiện một nhiệm vụ T từ kinh nghiệm E, nếu hiệu suất thực hiện công việc T của nó được đo bởi chỉ số hiệu suất P và được cải thiện bởi kinh nghiệm E theo thời gian).

Như vậy, học máy nói đến một chương trình giúp cho máy tính có thể giải một bài toán cụ thể bằng những “kinh nghiệm” mà nó đã được “học” từ dữ liệu.

* + - 1. **Chu trình Học máy**



Quy trình học máy

* **Giai đoạn 1:** Tiền xử lý dữ liệu

Giai đoạn này, tập trung ở việc thu thập dữ liệu, sau đó tiến hành làm sạch dữ liệu(điền các giá trị còn thiếu, xác định các ngoại lệ như dữ liệu bị trùng lặp, dữ liệu không nhất quán và làm mịn dữ liệu nhiễu), tích hợp và chuyển đổi dữ liệu(tích hợp nhiều dataset, chuẩn hóa và tổng hợp dữ liệu), giảm dữ liệu(giảm bớt về các thuộc tính của dữ liệu, giảm bớt kích thước dữ liệu – nhưng vẫn đảm bảo thu được các kết quả khai phá dữ liệu tương đương) và rời rạc hóa.

* **Giai đoạn 2:** Xử lý dữ liệu

Trong quá trình này, chúng ta cũng cần phải thực hiện phân chia dữ liệu thành các tập khác nhau bao gồm: 1 tập huấn luyện (training set), tập tối ưu (validation set) và tập kiểm thử (test set) với mục đích đo đạc, chọn ra mô hình với thông số phù hợp và kiểm tra độ chính xác của mô hình trong thực tế.

* **Giai đoạn 3:** Xây dựng mô hình và điều chỉnh tham số

Mục đích của bước này là xây dựng các mô hình cho việc huấn luyện. Các tham số của các mô hình sẽ được điều chỉnh phù hợp dựa trên kết quả học của tập huấn luyện và tập tối ưu.

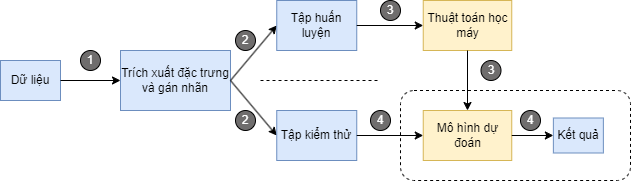
* **Giai đoạn 4:** Kiểm thử trên tập thử nghiệm

Sau đó, các mô hình tốt nhất đại diện cho mỗi thuật toán sẽ được đưa vào đánh giá hiệu quả trên tập kiểm thử, sử dụng các phân tích, đánh giá để lựa chọn mô hình tốt nhất (Golden Model).

* **Giai đoạn 5:** Áp dụng mô hình học máy vào ứng dung

Cuối cùng, sau khi chọn được mô hình, chúng ta sẽ áp dụng mô hình học máy vào ứng dụng để người sử dụng có thể nhập các thông tin.

* + - 1. **Học máy có giám sát**



Mô hình học máy có giám sát

Học có giám sát là trường hợp dữ liệu huấn luyện được cho một cách tường minh dưới dạng đầu vào và đầu ra của hàm đích, ví dụ, cho trước tập các mẫu cùng nhãn phân loại tương ứng.

Hàm y = f(x) từ tập huấn luyện cho trước {x1, x2,…, xn; y1, y2,…, yn} sao cho yi ≅ f(xi) với mọi i

Học máy khi đó được gọi là có giám sát để thể hiện việc thuật toán nhận được chỉ dẫn trực tiếp về lời giải cho từng trường hợp.

Việc học máy có giám sát có thể được nhóm lại thành các vấn đề về phân loại và hồi quy.

* Phân loại (Classification): Việc phân loại diễn ra khi biến đầu ra là một thể loại nào đó, chẳng hạn như “đỏ” hoặc “xanh” hoặc “có hỏa hoạn” và “không có hỏa hoạn”.
* Hồi quy (Regression): Việc hồi quy xảy ra là khi biến đầu ra là một giá trị thực, chẳng hạn như “đô la” hay “trọng lượng”.

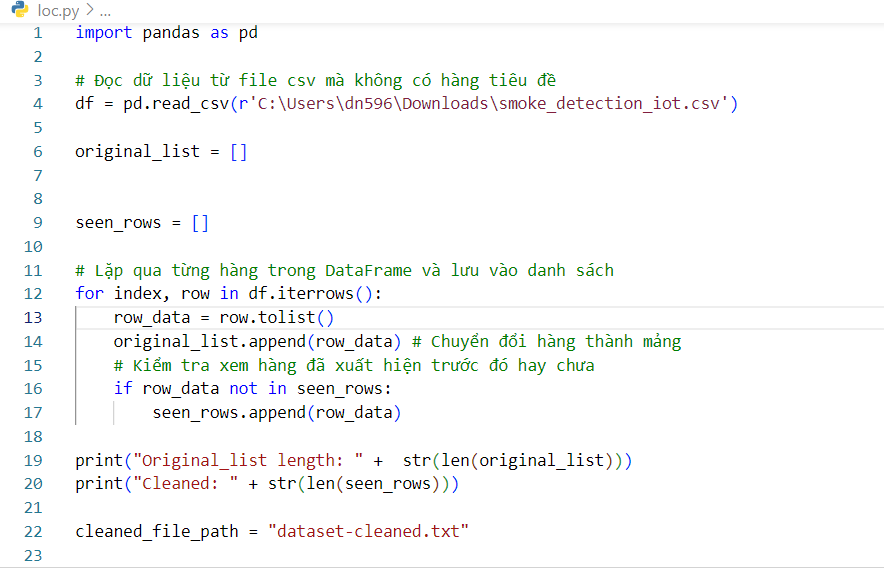
1. **TIỀN XỬ LÝ DỮ LIỆU**
   * + 1. **Xác định vấn đề**

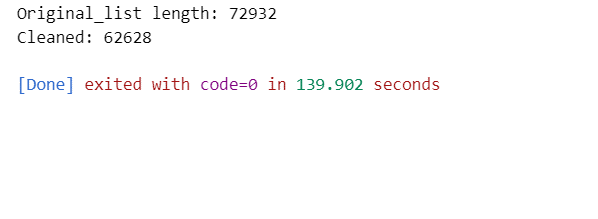
* Giảm thiểu áp lực cho hệ thống phòng cháy chữa cháy.
* Giúp người sử dụng đánh giá được tình hình hiện tại của nơi phát hiện xảy ra hỏa hoạn.
* Tích hợp với các hệ thống phòng ngừa hoản hoạn khác.
* Góp phần vào việc kiểm soát và phòng ngừa hỏa hoạn.
  + - 1. **Xác định mục đích**
* Xác định mục đích:
  + - Phát triển một mô hình dự đoán có thể xác định với độ chính xác cao khoảng 85% - 90% khả năng cháy dựa trên nhiệt độ, đổ ẩm, khí CO2, v.v
    - Giảm tỷ lệ phán đoán sai về dưới 10%.
      1. **Đặc điểm dữ liệu**
* Các thông tin mà dữ liệu cần có:
  + - Nhiệt độ không khí (độ C)
    - Độ ẩm (%)
    - Các hợp chất hữu cơ dễ bay hơi (ppb)
    - Nồng độ khí CO2 (ppm)
    - Hidro phân tử nguyên chất
    - Khí ethanol nguyên chất
    - Áp suất không khí (hPa)
    - Kích thước hạt vật chất, bao gồm:
    - PM1.0 với kích thước hạt vật chất < 1,0 µm.
    - PM2.5 với kích thước hạt vật chất nằm trong khoảng 1,0 µm – 2.5 µm.
    - Nồng độ số của các hạt vật chất, bao gồm:
    - NC0.5: Nồng độ số của các hạt vật chất có kích thước < 0,5 µm.
    - NC1.0: Nồng độ số của các hạt vật chất có kích thước từ 0,5 µm đến dưới 1,0 µm.
    - NC2.5: Nồng độ số của các hạt vật chất có kích thước từ 1,0 µm đến dưới 2,5 µm.
    - Bộ đếm mẫu
    - Kết quả phát hiện xảy ra hỏa hoạn
      * Yes (xảy ra hỏa hoạn)
      * No (không xảy ra hỏa hoạn)
* Để xây dựng mô hình đạt được hiệu quả tốt nhất, hoàn thành các mục đích đề ra, khối lượng của dữ liệu phải khoảng từ 5000 – 10000 các bản ghi dữ liệu.
* Độ chính xác và tin cậy chính xác, được đảm bởi các nguồn uy tín như Kaggle.
  + - 1. **Phương pháp thu thập dữ liệu**

Phương pháp thu thập dữ liệu là từ nguồn hiện có, cụ thể được ghi trong báo cáo như sau:

Smoke Detection Dataset, thu thập dữ liệu từ Kaggle: <https://www.kaggle.com/datasets/deepcontractor/smoke-detection-dataset/data>

* + - 1. **Tiền xử lý dữ liệu**
  1. **Làm sạch dữ liệu**
* Lọc các bản ghi trùng lặp
  + Lặp qua từng hàng trong DataFrame df bằng cách sử dụng phương thức iterrows().
  + Đối với mỗi hàng, nó chuyển đổi hàng đó thành một danh sách Python bằng cách sử dụng row.tolist().
  + Sau đó, nó kiểm tra xem hàng hiện tại đã xuất hiện trước đó trong seen\_rows hay chưa. Nếu chưa, nó thêm vào seen\_rows để đảm bảo rằng mỗi hàng chỉ xuất hiện một lần trong seen\_rows.
  + Cuối cùng, nó in ra độ dài của original\_list để kiểm tra số lượng hàng ban đầu và số lượng hàng không trùng lặp.

Kết quả:



Ban đầu, tập dữ liệu có 72932 bản ghi. Sau khi thực hiện việc lọc dữ liệu trùng, tập dữ liệu còn lại 62628 bản ghi.

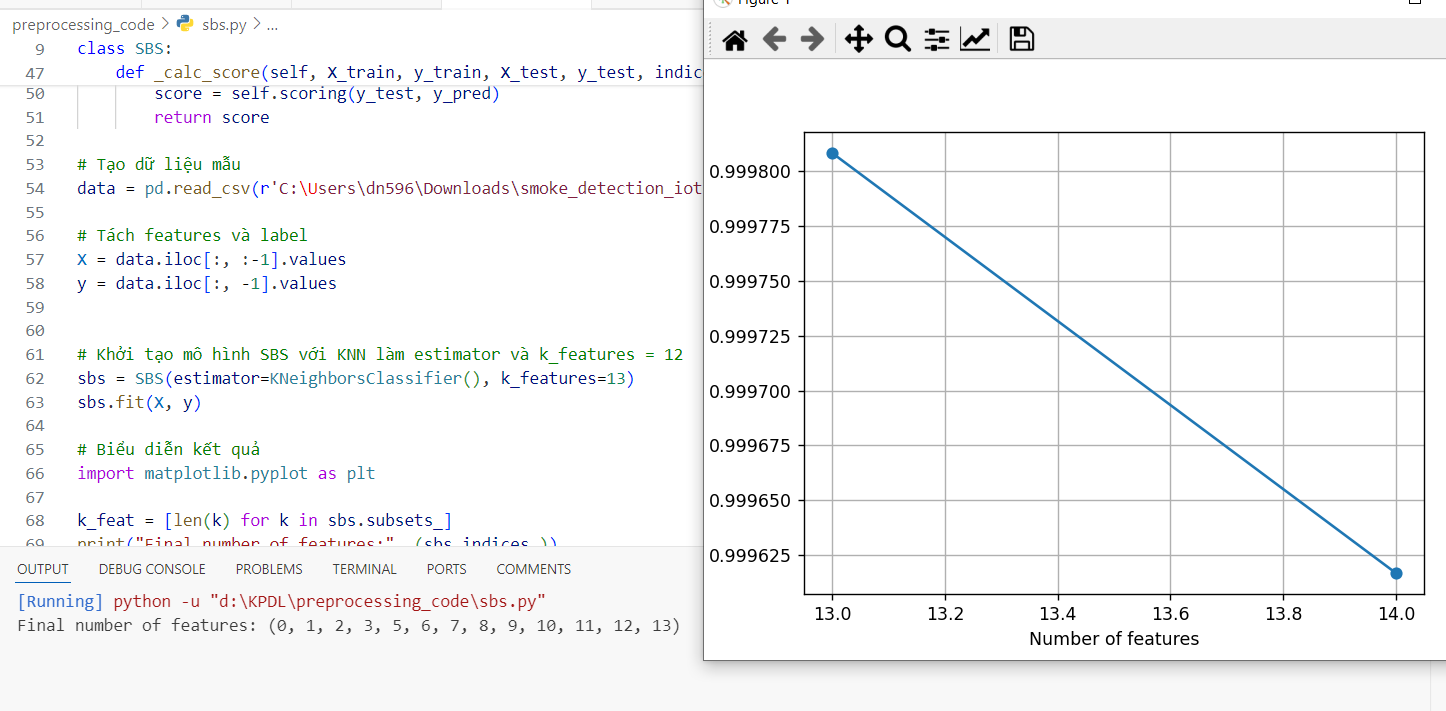
* 1. **Giảm chiều dữ liệu**
* SBS (Sequential Backward Selection) là một phương pháp tìm tập con tốt nhất của các đặc trưng trong quá trình huấn luyện mô hình học máy.
  + - * Phương pháp SBS bắt đầu với toàn bộ tập đặc trưng và loại bỏ lần lượt các đặc trưng cho đến khi đạt được độ chính xác tốt nhất của mô hình.
      * **Đầu vào:** Tập tất cả các đặc trưng Y = {y1; y2;…}
      * **Đầu ra:** Xk = {xj | j = 1,2,….k; xj Y}, k = (0, 1, 2, …, d) SBS trả về một tập hợp con các đặc trưng; số đặc trưng k (k < d) và phải được chỉ định trước
      * **Khởi tạo:** X0 = Y; k = d. Khởi tạo thuật toán cùng với việc đưa tất cả các đặc trưng từ Y vào X
      * **Bước 1 (bao gồm):** x- = argmaxJ (Xk - x), x Xk

Xk-1 = Xk – x-

k = k - 1

Trong bước này, bỏ X- khỏi Xk, X- là đặc trưng đem lại hiệu suất phân loại cao nhất nếu nó được bỏ ra khỏi Xk. Lặp lại cho đến khi tiêu chí dừng được xác lập.

* + - * **Dừng:** k = p thêm các đặc trưng vào tập con Xk cho đến khi số đặc trưng k bằng p được chỉ định trước hoặc đạt một hiệu suất nào đó.
* Áp dụng phương pháp SBS vào giảm chiều dữ liệu vào dataset mới, ta thấy k giảm đi còn 13 cột dữ liệu.



1. **XÂY DỰNG MÔ HÌNH HỌC MÁY**
   * + 1. **Naive Bayes**
   1. **Cơ sở lý thuyết**

* Phân loại Naive Bayes là một trong các phương pháp học máy được áp dụng phổ biến nhất trong thực tế, dựa trên định lý Bayes về lý thuyết xác suất để đưa ra các phan đoán cũng như phân loại dữ liệu dựa trên các dữ liệu được quan sát và thống kê.
* Định lý Bayes:

Trong đó:

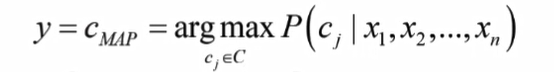
P(A): Xác suất xảy ra của A

P(B): Xác suất xảy ra của B

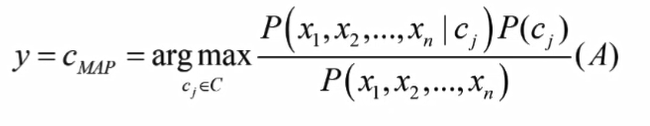
P(B|A): Xác suất của B khi biết A (xác suất có điều kiện)

P(A|B): Xác suất của A khi biết B (xác suất có điều kiện)

* Việc phân loại sẽ dựa trên các giá trị xác suất của các khả năng xảy ra các giả thuyết (phân loại)
* Mặc dù được đặt giả sử về sự độc lập có điều kiện của các thuộc tính đối với các phân lớp, nhưng phương pháp phân loại Naive Bayes vẫn thu được các kết quả phân loại tốt trong nhiều lĩnh vực ứng dụng thực tế.
* Trong giai đoạn huấn luyện, dữ liệu huấn luyện được cung cấp dưới dạng các mẫu <xi, yi>. Sau khi huấn luyện xong, bộ phân loại cần dự đoán nhãn cho mẫu mới x.
* Theo lý thuyết học Bayes, nhãn phân loại được xác định bằng cách tính xác suất điều kiện của nhãn khi quan sát thấy tổ hợp giá trị thuộc tính {x1, x2, x3, …, xn}. Thuộc tính được chọn, ký hiệu CMAP là thuộc tính có xác suất điều kiện cao nhất (MAP là viết tắt của maximum a posterior), tức là



* Sử dụng quy tắc Bayes, biểu thức trên được viết lại như sau



* Tại công thức A nhận thấy giá trị của P{x1, x2, x3, …, xn} không phụ thuộc vào giá trị của c, do đó không làm ảnh hưởng đến CMAP nên có thể viết lại như sau:



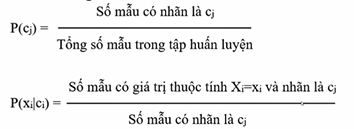
* Giá trị P(cj) được tính bằng tần suất quan sát thấy nhãn cj trên tập huấn luyện. Việc tính P{x1, x2, x3, …, xn|cj) khó khăn hơn nhiều. Vấn đề là số tổ hợp giá trị của n thuộc tính cùng với nhãn phân loại là rất lớn khi n lớn. Để tính xác suất này được chính xác, mỗi tổ hợp giá trị thuộc tính phải xuất hiện cùng nhãn phân loại đủ nhiều, trong khi số mẫu huấn luyện thường không đủ lớn.
* Để giải quyết vấn đề trên, giả sử các thuộc tính là độc lập về xác suất với nhau khi biết nhãn phân loại cj
* Với giả thiết về tính độc lập xác suất có điều kiện, có thể viết như sau:



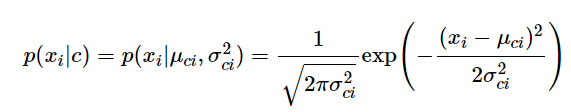
tức là xác suất đồng thời quan sát thấy các thuộc tính bằng tích xác suất điều kiện của từng thuộc tính riêng lẻ. Thay vào biểu thức ở trên, ta được bộ phân loại Bayes đơn giản (có đầu ra ký hiệu là cNB) như sau:



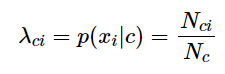
* Quá trình huấn luyện hay học Bayes đơn giản là quá trình tính các xác suất P(cj) và các xác suất điều kiện P(xi|cj) bằng cách đếm trên tập dữ liệu huấn luyện. Các xác suất P(cj) và các xác suất điều kiện P(xi|cj) được tính trên tập dữ liệu huấn luyên theo công thức sau:



* + 1. **Gaussian Naive Bayes**
* Mô hình này được sử dụng chủ yếu trong loại dữ liệu mà các thành phần là các biến liên tục.
* Với mỗi chiều dữ liệu i và một class c, xi tuân theo một phân phối chuẩn có kỳ vọng μci và phương sai :



* + 1. **Multinomial Naive Bayes**
* Mô hình này chủ yếu được sử dụng trong phân loại văn bản.
* Lúc này, mỗi văn bản được biểu diễn bởi một vector có độ dài d chính là số từ trong từ điển. Giá trị của thành phần thứ i trong mỗi vector chính là số lần từ thứ i xuất hiện trong văn bản đó.
* Khi đó, p(xi|c) tỉ lệ với tần suất từ thứ i (hay feature thứ i cho trường hợp tổng quát) xuất hiện trong các văn bản của class c. Giá trị này có thể được tính bằng cách:



Trong đó:

* + - Nci là tổng số lần từ thứ i xuất hiện trong các văn bản của class c, nó được tính là tổng của tất cả các thành phần thứ i của các feature vectors ứng với class c.
    - Nclà tổng số từ (kể cả lặp) xuất hiện trong class c. Nói cách khác, nó bằng tổng độ dài của toàn bộ các văn bản thuộc vào class c
    1. **Bernoulli Naive Bayes**
* Mô hình này được áp dụng cho các loại dữ liệu mà mỗi thành phần là một giá trị binary - bẳng 0 hoặc 1.
* Công thức tính:



với p(i|c) có thể được hiểu là xác suất từ thứ i xuất hiện trong các văn bản của class c.

* 1. **Vì sao chọn Native Bayes cho bài toán**

Naive Bayes là một thuật toán hiệu quả và có thể xử lý lượng dữ liệu lớn một cách nhanh chóng. Điều này rất quan trọng khi đề tài này có số lượng bản ghi rất lớn (cụ thể là hơn 60000 bản ghi).

* 1. **Áp dụng cơ sở lý thuyết vào xây dựng mô hình cho bài toán**
* Đọc dữ liệu từ file CSV

data=pd.read\_csv(r'D:\KPDL\smoke\_detection\_training.csv')

* Phân chia dữ liệu thành features và target

X = data.drop('Fire Alarm', axis=1)

y = data['Fire Alarm']

* Chia dữ liệu thành tập training và tập test (theo tỷ lệ 8:2)

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)

* Tạo mô hình Naïve Bayes

naïve\_bayes\_model = GaussianNB()

* Huấn luyện mô hình Naïve Bayes

naïve\_bayes\_model.fit(X\_train, y\_train)

* Lưu mô hình vào file

joblib.dump(naïve\_bayes\_model,‘naïve\_bayes\_model\_joblib’)

1. **Decision Tree (Cây quyết định)**
   1. **Cơ sở lý thuyết**

* Mô hình cây quyết định là một mô hình được sử dụng khá phổ biến và hiệu quả trong cả hai lớp bài toán Classification và Regression của học có giám sát. Khác với những thuật toán khác trong học có giám sát, mô hình cây quyết định không tồn tại phương trình dự báo. Mọi việc chúng ta cần thực hiện đó là tìm ra một cây quyết định dự báo tốt trên tập train và sử dụng cây quyết định này dự báo trên tập kiểm tra.
* Mỗi nhánh của cây giống như nhánh if, else nên có thể hiểu cây quyết định là tập hợp các luật nếu thì.
* Cây quyết định nhận đầu vào là một bộ giá trị thuộc tính mô tả một đối tượng hay một tình huống nào đó và trả về một giá trị rời rạc.
* Mỗi bộ thuộc tính đầu vào được gọi là một mẫu hay một ví dụ, đầu ra gọi là loại hay nhãn phân loại. Thuộc tính đầu vào còn được gọi là đặc trưng và có thể nhận giá trị rời rạc hoặc liên tục.
* Để cho đơn giản, trước tiên sẽ xem xét thuộc tính rời rạc, sau đó sẽ mở rộng cho trường hợp thuộc tính nhận giá trị liên tục.
* Thuật toán:
  + - Input: tập dữ liệu huấn luyện
    - Output: Cây quyết định
      * Khởi đầu: nút hiện thời là nút gốc chứa toàn bộ tập dữ liệu huấn luyện
      * Tại nút hiện thời n, lựa chọn thuộc tính:

Chưa được sử dụng (tức là nút nằm trên đường đi từ gốc tới nút hiện thời)

Cho phép phân chia tập dữ liệu hiện thời thành các tập con tốt nhất

* + - * Với mỗi giá trị thuộc tính được chọn:

Thêm một nút con bên dưới

Chia mẫu ở nút hiện thời về các nút con theo giá trị thuộc tính được chọn

* + - * Lặp (đệ quy) với mỗi nút con cho đến khi:

Tất cả các thuộc tính đã được sử dụng ở các nút phía trên.

Tất cả ví dụ tại nút hiện thời có cùng nhãn phân loại nhãn của nút được lấy theo đa số nhãn của ví dụ tại nút hiện thời.

* + 1. **Decision trong bài toán Classification**
* Biểu diễn dữ liệu:
  + - Mỗi đối tượng quan sát được thể hiện bởi một vector n chiều, mỗi chiều đại diện cho một thuộc tính đối tượng.
    - Mỗi đỉnh bên trong cây biểu diễn một thuộc tính để kiểm tra dữ liệu đến về sau.
    - Mỗi nhánh xuất phát từ một đỉnh tương ứng với mỗi một giá trị trong miền giá trị của thuộc tính đó.
    - Mỗi lá lưu nhãn lớp.
    - Cho dữ liệu cần phán đoán duyệt qua cây đấy, dùng các thuộc tính của nó duyệt từ gốc đến lá, lấy nhãn ở lá để phán đoán cho dữ liệu mới.
    1. **Một số thông số quan trọng của Decision Tree**
* Thuật toán xây dựng cây ID3 sử dụng entropy làm mức đo độ đồng nhất của tập dữ liệu. Trên cơ sở entropy, thuật toán tính độ tăng thông tin như mức tăng độ đồng nhất, từ đây xác định thuộc tính tốt nhất tại mỗi nút.
* Trong trường hợp chỉ có hai nhãn phân loại, ký hiệu là + và -, entropy H(S) của tập dữ liệu S được tính như sau:

H(S) = - p+log2p+ - p–log2p-

Trong đó p+ và p- là xác suất quan sát thấy nhãn phân loại + và -, tính bằng tần suất quan sát thấy + và – trong dữ liệu.

* Trong bảng huấn luyện trên, ta thấy có 9 nhãn “Yes” và 5 nhãn “No”, do đó:

+

* Có thể nhận thấy, trong trường hợp nhãn nhị phân, entropy đạt giá trị tối đa bằng 1 khi xác suất hai nhãn bằng nhau và bằng 0.5, entropy đạt giá trị nhỏ nhất bằng 0 khi xác suất một nhãn là 1 và nhãn còn lại là 0. Nói chung, entropy càng gần 0 thì tập đối tượng càng đồng nhất.
* Trong TH tổng quát với C nhãn phân loại có xác suất lần lượt là p1, p2, …, pc entropy được tính như sau:
* Giá trị cực đại của entropy khi đó sẽ bằng log2C khi các nhãn có xác suất như nhau và giá trị nhỏ nhất của entropy bằng 0 khi tất cả đối tượng có chung một nhãn. Tương tự trường hợp nhãn nhị phân, giá trị entropy càng nhỏ thì tập dữ liệu càng đồng nhất.
* Độ tăng thông tin (Information Gain), ký hiệu IG, là chỉ số đánh giá độ tốt của thuộc tính trong việc phân chia tập dữ liệu thành những tập con đồng nhất. IG được tính dựa trên entropy theo công thức sau:

Trong đó:

* + - S là tập dữ liệu ở nút hiện tại
    - A là thuộc tính values (A) là tập các giá trị của thuộc tính A
    - Sv là tập các mẫu có giá trị thuộc tính A bằng v
    - |S| và |Sv| là lực lượng của các tập hợp tương ứng
    1. **Cách Decision Tree hoạt động**

Mô hình Decision Tree (cây quyết định) hoạt động bằng cách xây dựng một cây quyết định từ dữ liệu huấn luyện để dự đoán nhãn hoặc giá trị mục tiêu cho dữ liệu mới. Dưới đây là cách Decision Tree hoạt động chi tiết:

* Chọn biến phân chia: Quá trình bắt đầu từ nút gốc của cây, thuật toán chọn một biến đặc trưng và một giá trị phân chia sao cho phân chia dữ liệu thành các nhóm có độ thuần khiết tốt nhất có thể. Độ thuần khiết được đo lường bằng các phép đo như Gini impurity hoặc entropy.
* Tạo nút mới trên cây: Sau khi biến phân chia được chọn, một nút mới được tạo trên cây và dữ liệu được phân loại vào các nhánh con tương ứng dựa trên giá trị của biến phân chia.
* Lặp lại quá trình cho các nhánh con: Quá trình này được lặp lại đệ quy cho mỗi nhánh con, tạo ra các nút mới và phân chia dữ liệu trong từng nhánh con đến khi một điều kiện dừng được đạt đến. Điều kiện dừng có thể là đạt đến một độ sâu tối đa cho cây, đạt đến một độ thuần khiết nhất định, hoặc các điều kiện khác.
* Dừng xây dựng cây: Quá trình xây dựng cây kết thúc khi một điều kiện dừng được đạt đến. Tại các nút lá cuối cùng của cây, dữ liệu được phân loại vào các nhãn hoặc dự đoán giá trị cuối cùng.
* Dự đoán cho dữ liệu mới: Khi cây đã được xây dựng, nó có thể được sử dụng để dự đoán nhãn hoặc giá trị cho các mẫu dữ liệu mới. Mẫu dữ liệu mới được đưa vào cây và đi theo đường dẫn từ nút gốc đến nút lá, và nhãn hoặc giá trị cuối cùng của nút lá đó sẽ là dự đoán cho mẫu dữ liệu mới.
  1. **Vì sao chọn Decision Tree cho bài toán**

Cây quyết định tạo ra một mô hình dễ hiểu, có thể diễn giải được bằng cách sử dụng các quy tắc và câu lệnh điều kiện đơn giản. Mô hình cây quyết định thường dễ kiểm tra với dữ liệu mới, và thậm chí còn cho phép kiểm tra tại thời điểm xây dựng cây.

* 1. **Áp dụng cơ sở lý thuyết vào xây dựng mô hình cho bài toán**
* Đọc dữ liệu từ file CSV

data=pd.read\_csv(r'D:\KPDL\smoke\_detection\_training.csv')

* Phân chia dữ liệu thành features và target

X = data.drop('Fire Alarm', axis=1)

y = data['Fire Alarm']

* Chia dữ liệu thành tập training và tập test (theo tỷ lệ 8:2)

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)

* Tạo mô hình ID3

model = DecisionTreeClassifier(criterion=’entropy’]

* Huấn luyện mô hình

model.fit(X\_train, y\_train)

* In cây quyết định

Tree\_rules=export\_text(model, feature\_names=list(X.colomns))

Print(tree\_rules)

* Lưu mô hình vào file

joblib.dump(model,‘decision\_tree\_model\_joblib’)

1. **Random Forest**
   1. **Cơ sở lý thuyết**

* Random Forest là một phương pháp thống kê mô hình hóa bằng máy (machine learning statistic) dùng để phục vụ các mục đích phân loại, tính hồi quy và các nhiệm vụ khác bằng cách xây dựng nhiều cây quyết định (Decision tree). Một cây quyết định là một cách đơn giản để biểu diễn một giao thức (Protocol). Nói cách khác, cây quyết định biểu diễn một kế hoạch, trả lời câu hỏi phải làm gì trong một hoàn cảnh nhất định. Mỗi Node của cây sẽ là các thuộc tính, và các nhánh là giá trị lựa chọn của thuộc tính đó. Bằng cách đi theo các giá trị thuộc tính trên cây, cây quyết định sẽ cho ta biết giá trị dự đoán. Nhóm thuật toán cây quyết định có một điểm mạnh đól à có thể sử dụng cho cả bài toán Phân loại (Classification) và Hồi quy (Regression).Random Forest có khả năng tìm ra thuộc tính nào quan trọng hơn so với những thuộc tính khác.
* Thuật toán:
* Bước 1: Từ tập dữ liệu huấn luyện D, ta tạo dữ liệu ngẫu nhiên (mẫu bootstrap).
* Bước 2: Sử dụng các tập con dữ liệu lấy mẫu ngẫu nhiên D1 , D2 ,…, Dk xây dựng nên các cây T1 , T2 ,…,Tk.
* Bước 3: Kết hợp các cây: sử dụng chiến lược bình chọn theo số đông với bài toán phân loại hoặc lấy trung bình các giá trị dự đoán từ các cây với bài toán hồi quy.
  + 1. **Một số ưu điểm của Random Forest**

(1) Độ chính xác của RF tương đối cao.

(2) Thuật toán giải quyết tốt các bài toán có nhiều dữ liệu nhiễu.

(3) Thuật toán chạy nhanh hơn so với bagging.

(4) Có những sự ước lượng nội tại như độ chính xác của mô hình dự đoán hoặc độ mạnh và liên quan giữa các thuộc tính.

(5) Dễ dàng thực hiện song song.

(6) Tuy nhiên để đạt được các tính chất mạnh trên, thời gian thực thi của thuật toán khá lâu và phải sử dụng nhiều tài nguyên của hệ thống.

* + 1. **Một số thông số quan trọng của Random Forest**
    - n\_estimators: Đây là số lượng cây quyết định trong Random Forest. Điều này ảnh hưởng đến tính đa dạng và ổn định của mô hình. Thông thường, một số lượng cây lớn hơn sẽ làm cho mô hình tốt hơn, nhưng cũng tăng chi phí tính toán.
    - criterion: Đây là hàm để đo lường chất lượng của mỗi phép chia. Đối với bài toán phân loại, hai lựa chọn phổ biến là "gini" (hệ số Gini) và "entropy" (thu nhập thông tin). Đối với bài toán hồi quy, thường sử dụng "mse" (mean squared error) hoặc "mae" (mean absolute error).
    - max\_depth: Đây là độ sâu tối đa của mỗi cây quyết định trong Random Forest. Điều này ảnh hưởng đến khả năng tập trung và tính tổng quát của mô hình. Một max\_depth lớn có thể dẫn đến overfitting.
    - min\_samples\_split: Số lượng mẫu tối thiểu cần thiết để phân chia một nút. Giá trị nhỏ hơn có thể dẫn đến overfitting, trong khi giá trị lớn hơn có thể làm mất thông tin.
    - min\_samples\_leaf: Số lượng mẫu tối thiểu được yêu cầu để tạo thành một lá trong cây. Giống như min\_samples\_split, giá trị nhỏ hơn có thể dẫn đến overfitting, trong khi giá trị lớn hơn có thể tạo ra các lá ít thông tin hoặc tập trung quá mức.
    - max\_features: Số lượng đặc trưng được lựa chọn ngẫu nhiên để xem xét cho việc phân chia tại mỗi nút. Việc giảm số lượng đặc trưng có thể giúp giảm overfitting.
    - bootstrap: Xác định xem liệu mẫu bootstrap (lấy mẫu có thay thế từ dữ liệu huấn luyện) có được sử dụng không.
    1. **Cách Random Forest hoạt động**

Random Forest hoạt động bằng cách kết hợp nhiều cây quyết định (decision trees) để tạo ra một mô hình dự đoán mạnh mẽ và ổn định. Dưới đây là quá trình hoạt động cụ thể của Random Forest:

* + - * Tạo dữ liệu con (Bootstrap Sampling):

Random Forest bắt đầu bằng việc lấy mẫu dữ liệu từ tập dữ liệu huấn luyện bằng cách sử dụng phương pháp lấy mẫu có thay thế, được gọi là Bootstrap Sampling. Điều này có nghĩa là một số mẫu có thể xuất hiện nhiều lần trong dữ liệu con, trong khi một số mẫu khác có thể không xuất hiện.

* + - * Xây dựng cây quyết định (Decision Trees):

Sau khi có dữ liệu con, một cây quyết định được xây dựng trên mỗi dữ liệu con này. Quá trình xây dựng cây quyết định này có thể được kiểm soát bằng cách sử dụng các tham số như độ sâu tối đa của cây, số lượng mẫu tối thiểu cho mỗi nút lá, hoặc số lượng đặc trưng được xem xét cho mỗi phép chia.

* + - * Phân loại hoặc dự đoán:

Khi cần dự đoán trên một dữ liệu mới, Random Forest tính toán dự đoán bằng cách chạy dữ liệu mới qua tất cả các cây quyết định trong mô hình. Trong bài toán phân loại, kết quả cuối cùng thường được xác định bằng phương pháp bầu cử (voting) giữa các cây, trong đó lớp được dự đoán là lớp được bầu chọn nhiều nhất. Trong bài toán hồi quy, kết quả cuối cùng thường là trung bình của các dự đoán từ tất cả các cây.

* 1. **Vì sao chọn Random Forest cho bài toán**

Random Forest thường có hiệu suất tốt trên nhiều loại dữ liệu, bao gồm cả dữ liệu có nhiễu và dữ liệu không cân bằng.

Do kết hợp nhiều cây quyết định độc lập nhau, mô hình Random Forest thường ổn định hơn so với một cây quyết định đơn lẻ. Điều này giúp giảm thiểu hiện tượng overfitting..

* 1. **Áp dụng cơ sở lý thuyết vào xây dựng mô hình cho bài toán**
* Đọc dữ liệu từ file CSV

data=pd.read\_csv(r'D:\KPDL\smoke\_detection\_training.csv')

* Phân chia dữ liệu thành features và target

X = data.drop('Fire Alarm', axis=1)

y = data['Fire Alarm']

* Chia dữ liệu thành tập training và tập test (theo tỷ lệ 8:2)

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)

* Tạo mô hình

model=RandomForestClassifier(n\_estimators=100, random\_state=42)]

* Huấn luyện mô hình

model.fit(X\_train, y\_train)

* Lưu mô hình vào file

joblib.dump(model,‘random\_forest\_model\_joblib’)

1. **ĐÁNH GIÁ MÔ HÌNH HỌC MÁY**

Đánh giá mô hình thông qua ma trận nhầm lẫn (Confusing Matrix).

* Ma trận nhầm lẫn (confusion matrix) là công cụ trực quan và hiệu quả để đánh giá hiệu suất của mô hình phân loại. Nó thể hiện rõ ràng mức độ chính xác của mô hình trong việc dự đoán các lớp (classes) khác nhau.
* Phương pháp đánh giá:
  + - Tính toán các thước đo
      * Accuracy: tỉ lệ giữa số điểm dữ liệu được dự đoán đúng và tổng số điểm dữ liệu.
      * Precision: tỉ lệ số điểm true positive trong số những điểm được phân loại là positive.
      * Recall: tỉ lệ số điểm true positive trong số những điểm thực sự là positive.
      * F1 Score: Điểm cân bằng giữa precision và recall.
    - Phân tích các thước đo trong ma trận: Một mô hình tốt khi cả Precision và Recall đều cao có nghĩa là ít phân loại nhầm và tỷ lệ bỏ sót thấp.
* Công thức tính:
  + - Accuracy = (TP+TN)/(TP+TN+FP+FN)
    - Precision = TP/(TP+ FP)
    - Recall = TP/(TP+ FN)
    - Accuracy = 2\* (Precision\*Recall )/(Precision+Recall)

Trong đó:

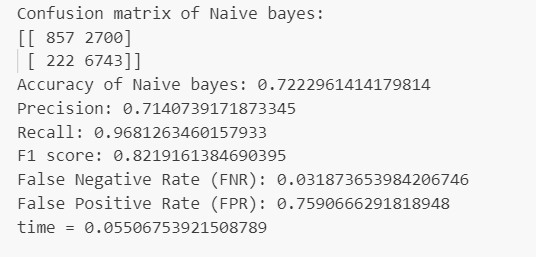
TP ( True Positive): Số lượng các trường hợp mà mô hình dự đoán đúng là positive (dương tính) và thực tế cũng là positive.

TN ( True Negative): Số lượng các trường hợp mà mô hình dự đoán đúng là negative và thực tế cũng là negative.

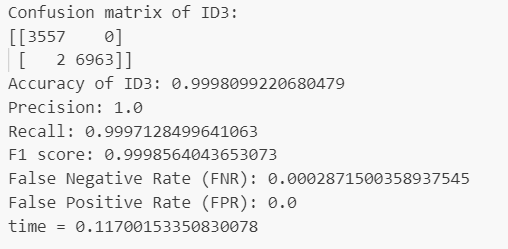
FP ( False Positive): Số lượng các trường hợp mà mô hình dự đoán là positive nhưng thực tế lại là negative (âm tính).

FN ( False Negative): Số lượng các trường hợp mà mô hình dự đoán là negative nhưng thực tế lại là positive.

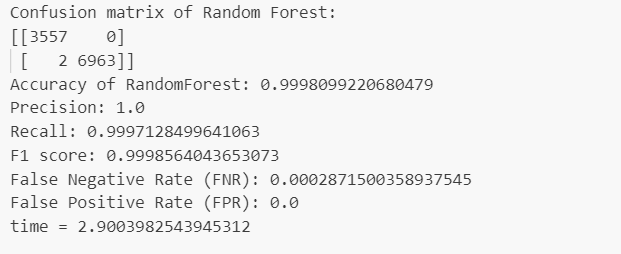
* Đánh giá mô hình Naïve Bayes:



* + Nhận xét:
    - Mô hình có Accuracy thấp (72,23%)
    - Mô hình có Precision thấp (71,41%)
    - Mô hình có Recall rất cao (96,81%)
    - Mô hình có: F1 Score thấp (75,91%)
    - Mô hình có: thời gian hoàn thành rất ngắn (0.055s)
      * Mô hình có hiệu suất kém
* Đánh giá mô hình Decision Trees:



* + Nhận xét:
    - Mô hình có Accuracy rất cao (99,98%)
    - Mô hình có Precision tuyệt đối (100%)
    - Mô hình có Recall rất cao (99,97%)
    - Mô hình có: F1 Score rất cao (99,98%)
    - Mô hình có: thời gian hoàn thành rất ngắn (0.117s)
      * Mô hình có hiệu suất rất tốt
* Đánh giá mô hình Random Forest



* + Nhận xét:
    - Mô hình có Accuracy rất cao (99,98%)
    - Mô hình có Precision tuyệt đối (100%)
    - Mô hình có Recall rất cao (99,97%)
    - Mô hình có: F1 Score rất cao (99,98%)
    - Mô hình có: thời gian hoàn thành ngắn (2.9s)
      * Mô hình có hiệu suất rất tốt
* So sánh 3 mô hình

|  |  |  |  |
| --- | --- | --- | --- |
|  | Decision Tree | Naïve Bayes | Random Forest |
| Accuracy | 99,98% | 72,23% | 99,98% |
| Precision | 100% | 71,41% | 100% |
| Recall | 99,97% | 96,81% | 99,97% |
| F1 – Score | 99,98% | 75,91% | 99,98% |
| Time | 0.117s | 0.055s | 2.9s |

* + - Nhận xét

Decision Tree và Random Forest có độ chính xác cao hơn so với Naive Bayes (99,98% so với 72,23%). Điều này cho thấy Decision Tree và Random Forest có khả năng phân loại chính xác hơn trên tập dữ liệu kiểm tra.

Hai mô hình Decision Tree và Random Forest đều có Precision cao tuyệt đối (100%), trong khi, Naïve Bayes có Precision khá thấp (72,23%). Điều này cho thấy Decision Tree và Random Forest có tỉ lệ dự đoán đúng positive cao hơn Naïve Bayes.

Cả ba mô hình đều có Recall rất cao, tuy nhiên, Decision Tree và Random Forest có Recall ngang nhau và cao hơn Naive Bayes (99,97% so với 96,81%). Điều này chỉ ra rằng Decision Tree và Random Forest có khả năng phát hiện các trường hợp positive tốt hơn Naive Bayes trên tập dữ liệu kiểm tra.

Hai mô hình Decision Tree và Random Forest đều có F1 Score rất cao (99,98%), trong khi đó, Naïve Bayes có F1 Score khá thấp (75,91%). F1 Score kết hợp cả Precision và Recall, do đó, mô hình Decision Tree và Random Forest có sự cân bằng giữa việc dự đoán chính xác positive và phát hiện được nhiều positive hơn trong khi Naive Bayes có sự chênh lệch giữa Precision và Recall.

* + - * Mô hình Decision Tree và Random Forest có hiệu suất tốt hơn cho bài toán đặt ra.
* So sánh 2 mô hình Decision Tree và Random Forest về thời gian huấn luyện, ta thấy mô hình Deciosion Tree có thời gian huấn luyện ngắn hơn Random Forest.
  + - * Vì vậy, mô hình Decision Tree có hiệu suất tốt hơn cho bài toán đặt ra.

1. **TÍCH HỢP MÔ HÌNH HỌC MÁY VÀO ỨNG DỤNG**

* Thuật toán

import tkinter as tk

import joblib

def submit\_form():

# Lấy dữ liệu từ các ô nhập

data = []

for entry in entry\_fields:

data.append(float(entry.get()))

model = joblib.load(r'D:\KPDL\test\_joblib\random\_forest\_model.joblib')

# Dự đoán trên tập dữ liệu

y\_pred = model.predict([data])

if y\_pred[0] == 1:

result\_label.config(text="Xảy ra cháy", font=("Arial", 20), fg="red")

else:

result\_label.config(text="Không xảy ra cháy", font=("Arial", 20), fg="blue")

def reset\_form():

# Xóa nội dung đã nhập trong các ô nhập

for entry in entry\_fields:

entry.delete(0, tk.END)

# Reset nhãn kết quả

result\_label.config(text="Kết quả dự đoán:", font=("Arial", 20))

# Tạo cửa sổ

root = tk.Tk()

root.title("Form Nhập Dữ Liệu")

# Tạo Frame cho phần tiêu đề

title\_frame = tk.Frame(root)

title\_frame.grid(row=0, column=0, columnspan=2, pady=10)

# Thêm tiêu đề cho form

form\_title = tk.Label(title\_frame, text="Form Test", font=("Arial", 20, "bold"), fg="red")

form\_title.pack()

# Tạo Frame cho phần nhập liệu

input\_frame = tk.Frame(root)

input\_frame.grid(row=1, column=0, columnspan=2)

# Tạo danh sách các tên của từng ô nhập

entry\_names = ["Nhiệt độ (độ C)", "Độ ẩm (%)", "Các hợp chất hữu cơ dễ bay hơi (ppb)", "Nồng độ CO2 tương đương (ppm)", "Hydro phân tử nguyên chất; chưa được bù đắp (Sai số, nhiệt độ, vv.)", "Khí ethanol nguyên chất",

"Áp suất không khí (hPa)", "PM1.0 (Kích thước hạt vật chất < 1,0 µm)", "PM2.5 (Kích thước hạt vật chất trong [1,0 µm - 2,5 µm] )", "NC0.5 (Nồng độ số của các hạt có kích thước nhỏ hơn 0.5 µm)",

"NC1.0 (Nồng độ số của các hạt có kích thước từ 0.5 µm đến dưới 1.0 µm)", "NC2.5 (Nồng độ số của các hạt có kích thước từ 1.0 µm đến dưới 2.5 µm)", "Bộ đếm mẫu"]

# Tạo danh sách các ô nhập

entry\_fields = []

for i in range(13):

label = tk.Label(input\_frame, text=f"{entry\_names[i]}:", font=("Arial", 12), fg = "blue") # Chỉnh cỡ chữ ở đây

label.grid(row=i, column=0, padx=10, pady=5, sticky="e") # Sử dụng sticky để căn chỉnh về phía đông

entry = tk.Entry(input\_frame, font=("Arial", 12)) # Chỉnh cỡ chữ ở đây

entry.grid(row=i, column=1, padx=10, pady=5, sticky="w") # Sử dụng sticky để căn chỉnh về phía tây

entry\_fields.append(entry)

# Tạo nút "Submit" để gửi dữ liệu

submit\_button = tk.Button(root, text="Submit", command=submit\_form, font=("Arial", 12)) # Chỉnh cỡ chữ ở đây

submit\_button.grid(row=2, column=0, padx=10, pady=10)

# Tạo nút "Reset" để xóa dữ liệu đã nhập

reset\_button = tk.Button(root, text="Reset", command=reset\_form, font=("Arial", 12)) # Chỉnh cỡ chữ ở đây

reset\_button.grid(row=2, column=1, padx=10, pady=10)

# Tạo Frame cho phần kết quả

result\_frame = tk.Frame(root)

result\_frame.grid(row=3, column=0, columnspan=2)

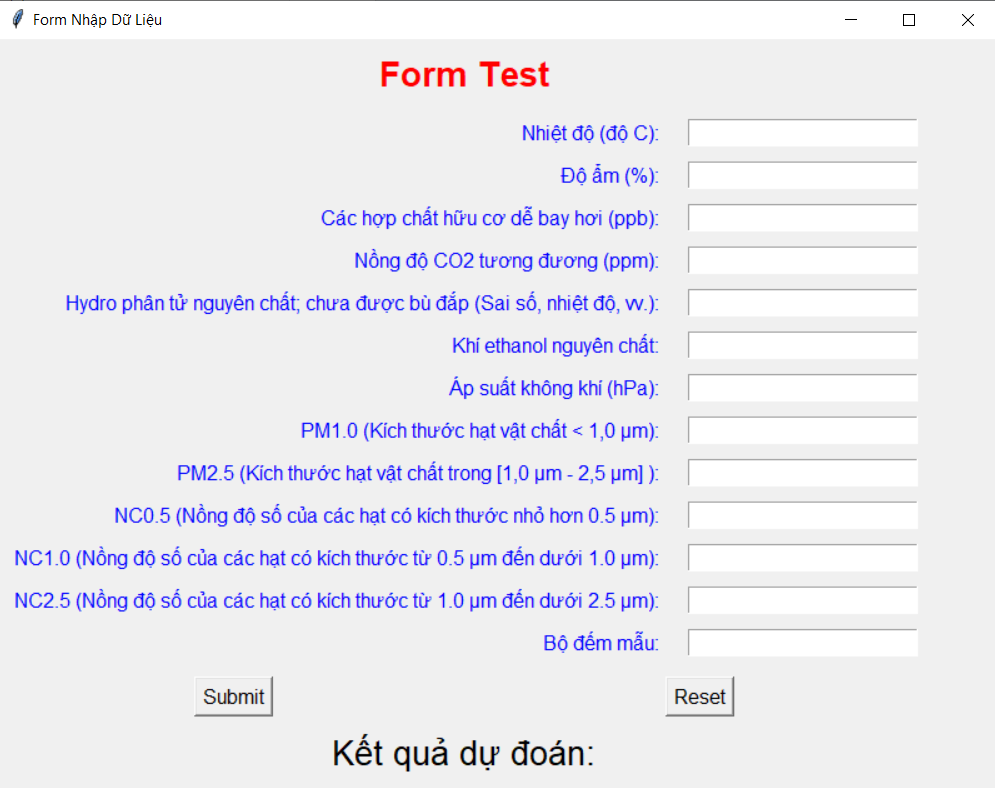
result\_label = tk.Label(result\_frame, text="Kết quả dự đoán:", font=("Arial", 20)) # Chỉnh cỡ chữ ở đây

result\_label.pack()

root.geometry("800x600") # Thay đổi kích thước cửa sổ

root.mainloop()

* Kết quả



1. **KẾT LUẬN**

Trong quá trình làm bài tập lớn, nhóm đã thực hiện các công việc sau:

* Nghiên cứu về các phương pháp học máy có giám sát.
* Áp dụng các giai đoạn của chu trình học máy vào bài toán thực tế
* Thực hiện đánh giá, so sánh 3 mô hình học để lựa chọn mô hình học máy tốt nhất.
* Tích hợp được mô hình học máy vào ứng dụng để giải quyết các vấn đề trong việc phòng ngừa hỏa hoạn.

**TÀI LIỆU THAM KHẢO**

[1] Bài giảng môn Nhập môn Khai phá dữ liệu và Máy học, TS Vũ Xuân Hạnh, Khoa Công nghệ thông tin, Trường Đại Học Mở Hà Nội

[2] Machine Learning cơ bản, Vũ Hữu Tiệp. Available: <https://machinelearningcoban.com/categories/> .

[3] Khái niệm về phương pháp random forest trong cuộc cách mạng machine learning và định hướng ứng dụng trong lĩnh vực viễn thám. Available: <https://jgac.vn/journal/article/view/344/331>

[4] <https://scikit-learn.org/stable/modules/generated/sklearn.impute.SimpleImputer.html>

[5] Thuật toán phân loại Naive Bayes và ứng dụng. Available: <https://whitehat.vn/threads/thuat-toan-phan-loai-naive-bayes-va-ung-dung.13775/>

[6] Bài giảng môn Nhập môn Khai phá dữ liệu và Máy học, Nguyễn Nhật Quang, Đại Học Bách Khoa Hà Nội. Available: <https://drive.google.com/file/d/1_Gsz68GAulRJUzaGIgq93B70PVHYSuPC/view>