Ứng dụng thuật toán máy học vào dự đoán Tình trạng sức khỏe thai nhi

Nguyễn Minh Duy
21520208
KHCL2021.1
Trường ĐH CNTT, ĐHQG-HCM
21520208@gm.uit.edu.vn

Huỳnh Võ Ngọc Thanh
21520449
KHCL2021.1
Trường ĐH CNTT, ĐHQG-HCM
21520449@gm.uit.edu.vn

Nguyễn Trần Hoài Bão
21520618
KHCL2021.1
Trường ĐH CNTT, ĐHQG-HCM
21520618@gm.uit.edu.vn

***Tóm tắt nội dung*—Giảm tử vong ở trẻ sơ sinh là một trong số Mục tiêu Phát triển Bền vững của Liên Hiệp Quốc. Việc đánh giá và dự đoán tình trạng sức khỏe của thai nhi là một công việc quan trọng giúp phát hiện sớm tình trạng bất thường của thai nhi từ đó giúp các chuyên gia y tế có những biện pháp can thiệp kịp thời cho cả mẹ và bé. So với phương pháp phân loại Logistic Regression truyền thống chỉ phù hợp cho đa số các bài toán phân loại nhị phân thì ba phương pháp Support Vector Machines (SVM), K-Nearest Neighbors (KNN) và Random Forest (RF) cho hiệu quả cao hơn trong bài toán phân loại đa lớp. Trong bài báo này, chúng ta cùng nhau phân tích và đánh giá ba thuật toán phân loại trên trong tập dữ liệu Fetal Health mà nhóm đã thu thập. Kết quả cho thấy cả ba mô hình đều mang lại tiến bộ trong việc dự đoán sức khỏe thai nhi. Nhưng qua đa số các lần huấn luyện, mô hình SVM cho độ chính xác cao hơn hai mô hình còn lại do ưu điểm vượt trội của nó là khả năng tạo ra các siêu phẳng phức tạp để phân loại các lớp dữ liệu phi tuyến.**

Từ khóa: máy học, học có giám sát, phân loại, điện tâm đồ, Support Vector Machines (SVM), K-Nearest Neighbors (KNN), Random Forest (RF).

I. GIỚI THIỆU - INTRODUCTION

Mô hình Phân loại sức khỏe thai nhi - Fetal Health Classification Model (FHCM) tập trung vào việc dự đoán tình trạng sức khỏe của thai nhi dựa trên một tập hợp các bản ghi với các đặc trưng liên quan đến tình trạng sức khỏe của thai nhi bằng các thuật toán học có giám sát.

Học có giám sát là một mô hình máy học sử dụng dữ liệu đã được gán nhãn, nghĩa là mỗi điểm dữ liệu chứa các đặc trưng (biến độc lập) và một nhãn tương ứng. Mục tiêu của các thuật toán học có giám sát là học một hàm ánh xạ từ vector đặc trưng (đầu vào) đến nhãn (đầu ra) dựa trên các dữ liệu có sẵn. Hai nhóm bài toán cơ bản trong học có giám sát là phân loại (classification) và hồi quy (regression), trong đó biến đầu ra của bài toán phân loại có các giá trị rời rạc trong khi biến đầu ra của bài toán hồi quy có các giá trị liên tục.

Phân loại sức khỏe thai nhi được sử dụng để phát hiện sớm các vấn đề sức khỏe như dị tật bẩm sinh, bất thường trong phát triển của thai nhi, từ đó có thể quyết định phương pháp sinh phù hợp. Bên cạnh đó, việc dự đoán tình trạng của thai nhi cũng giảm bớt chi phí điều trị các biến chứng và vấn đề sức khỏe phát sinh lâu dài, giúp tiết kiệm tài chính cho gia đình, giảm gánh nặng cho xã hội.

Trong phạm vi bài báo này, nhóm sẽ tìm hiểu cách hoạt động, phân tích và đánh giá ba mô hình học máy học: Support Vector Machines (SVM), K-Nearest Neighbors (KNN) và Random Forest (RF), đồng thời sử dụng bộ dữ liệu Fetal Health mà nhóm thu thập để cài đặt và rút trích những ưu nhược điểm của các mô hình. Để đánh giá mô hình nào tốt, nhóm đã tính các độ đo F1, Accuracy, Precision và Recall trên từng mô hình và điều chỉnh các tham số sao cho thuật toán có kết quả tốt nhất, sau đó so sánh các chỉ số này với nhau.

Input: các chỉ số dùng để dự đoán sức khỏe của thai nhi, gồm: nhịp tim cơ bản của thai nhi (baseline value), gia tốc (accelerations), số cử động của thai nhi trên giây (fetal movement),...

Output: dự đoán sức khỏe thai nhi thuộc lớp bình thường (normal), nghi ngờ (suspect) hay mắc bệnh (pathological).

II. BỘ DỮ LIỆU - DATASET

A. Giới thiệu bộ dữ liệu:

Bộ dữ liệu mà nhóm sử dụng được thu thập từ website Kaggle download dataset

Bộ dữ liệu này được trích xuất từ bài báo: Ayres de Campos et al. (2000) SisPorto 2.0 - một chương trình phân tích điện tâm đồ tự động. J Matern Fetal Med 5:311-318. Link bài báo: Ayres de Campos et al. (2000) SisPorto 2.0 A Program for Automated Analysis of Cardiotocograms. J Matern Fetal Med 5:311-318

Bộ dữ liệu chứa dữ liệu từ điện tâm đồ (CTG). CTG mô tả kết quả siêu âm, các bài đọc cho thấy nhịp tim của thai nhi, các cơn co tử cung trong số các chỉ số khác về sức khỏe của thai nhi. Chúng rất cần thiết và đơn giản trong việc cung cấp dữ liệu về sức khỏe hiện tại của thai nhi cũng như có thể là dấu hiệu đầu tiên cho thấy tình trạng suy thai.

B. Mô tả bộ dữ liệu:

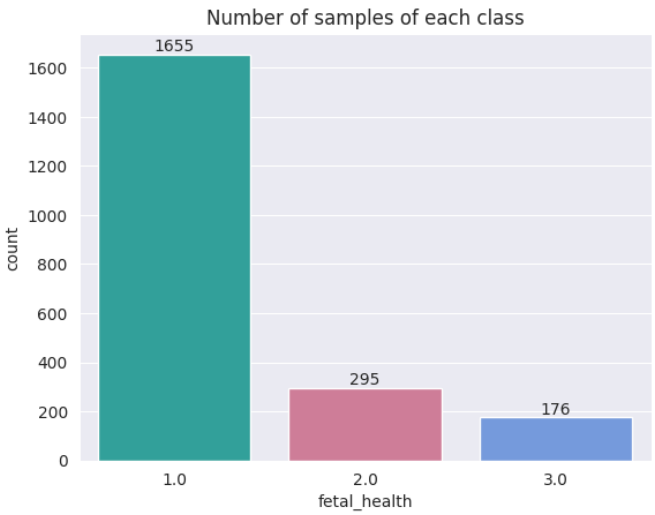
Bộ dữ liệu nhóm thu thập có 2126 đối tượng và 22 đặc trưng gồm các số liệu liên quan đến việc dự đoán sức khỏe của thai nhi và được trình bày dưới dạng bảng thống kê (file csv). Ở đây, nhóm sẽ nói rõ hơn về các đặc trưng để chúng ta có cái nhìn tổng quan hơn về bộ dữ liệu:

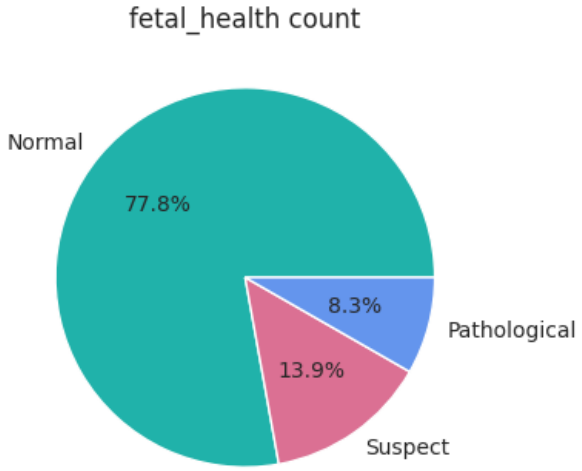
- baseline value: chỉ số nhịp tim cơ bản của thai nhi (nhịp/phút)
- accelerations: gia tốc trong FHR (tương quan nghịch với tỷ lệ tử vong của thai nhi hoặc không đáng kể)
- fetal_movement: số lần chuyển động của thai nhi trong một giây
- uterine_contractions: số lần co tử cung trong một giây
- light_decelerations (LDs): số lần giảm nhẹ nhịp tim trong một giây
- severe_decelerations (SDs): số lần giảm nghiêm trọng nhịp tim trong một giây
- prolongued_decelerations (PDs): số lần giảm kéo dài trong một giây
- abnormal_short_term_variability: sự biến thiên ngắn hạn bất thường
- mean_value_of_short_term_variability: giá trị trung bình của biến thiên ngắn hạn
- percentage_of_time_with_abnormal_long_term _variability: tỷ lệ thời gian biến thiên dài hạn bất thường
- mean_value_of_long_term_variability: giá trị trung bình của biến thiên dài hạn
- histogram_width: độ rộng của biểu đồ
- histogram_min: giá trị tối thiểu của biểu đồ
- histogram_max: giá trị tối đa của biểu đồ
- histogram_number_of_peaks: số đỉnh trong biểu đồ
- histogram_number_of_zeroes: số lượng giá trị 0 trong biểu đồ
- histogram_mode: mode của biểu đồ
- histogram_mean: giá trị trung bình của biểu đồ
- histogram_median: trung vị của biểu đồ
- histogram_variance: phương sai của biểu đồ
- histogram_tendency: xu hướng của biểu đồ
- fetal_health (cột mục tiêu): tình trạng thai nhi [1: Normal (Bình thường), 2: Suspect (Nghi ngờ), 3: Pathological (Bệnh lý)]

Bộ dữ liệu có 22 thuộc tính tất cả đều ở dạng số nên không cần thiết phải thực hiện encode, tuy nhiên nhóm có đề xuất chuyển các giá trị ở cột mục tiêu từ 1,2,3 thành 0,1,2 giúp các thuật toán có thể hiểu được mối quan hệ thứ tự và khoảng cách giữa các giá trị phân loại khi thực hiện tính toán.

```
RangeIndex: 2126 entries, 0 to 2125
Data columns (total 22 columns):
 #   Column                                                  Non-Null Count  Dtype
---  ------                                                  --------------  -----
 0   baseline value                                          2126 non-null   float64
 1   accelerations                                           2126 non-null   float64
 2   fetal_movement                                          2126 non-null   float64
 3   uterine_contractions                                    2126 non-null   float64
 4   light_decelerations                                     2126 non-null   float64
 5   severe_decelerations                                    2126 non-null   float64
 6   prolongued_decelerations                                2126 non-null   float64
 7   abnormal_short_term_variability                         2126 non-null   float64
 8   mean_value_of_short_term_variability                    2126 non-null   float64
 9   percentage_of_time_with_abnormal_long_term_variability  2126 non-null   float64
 10  mean_value_of_long_term_variability                     2126 non-null   float64
 11  histogram_width                                         2126 non-null   float64
 12  histogram_min                                           2126 non-null   float64
 13  histogram_max                                           2126 non-null   float64
 14  histogram_number_of_peaks                               2126 non-null   float64
 15  histogram_number_of_zeroes                              2126 non-null   float64
 16  histogram_mode                                          2126 non-null   float64
 17  histogram_mean                                          2126 non-null   float64
 18  histogram_median                                        2126 non-null   float64
 19  histogram_variance                                      2126 non-null   float64
 20  histogram_tendency                                      2126 non-null   float64
 21  fetal_health                                            2126 non-null   float64
dtypes: float64(22)
memory usage: 365.5 KB
```

Qua quan sát, đây là bộ dữ liệu bị mất cân bằng, tỉ lệ của ba lớp trong cột mục tiêu có sự chênh lệch rõ ràng, tỷ lệ phân bố của nhãn 1 (Normal) quá cao so với hai nhãn còn lại.



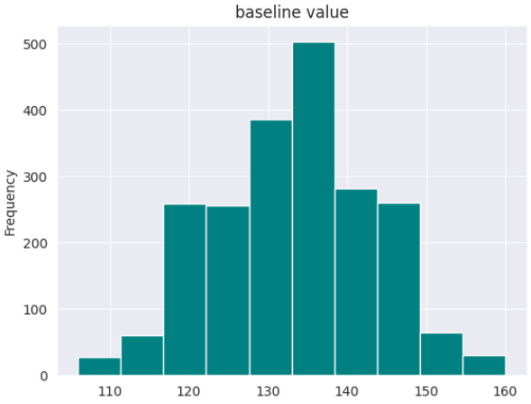


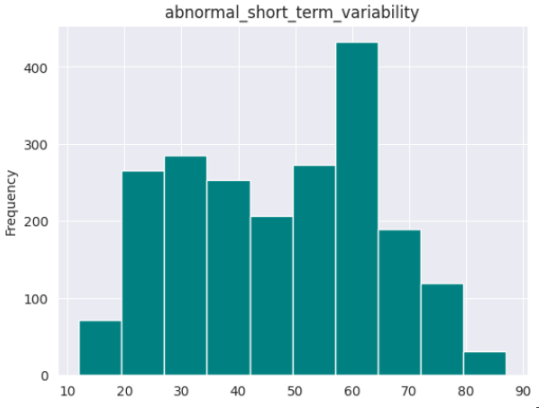
Nhóm sử dụng mối tương quan để xem thuộc tính nào ảnh hưởng đến thuộc tính mục tiêu và ảnh hưởng như thế nào:

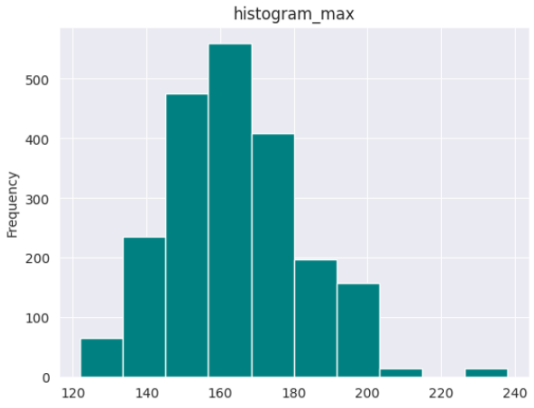
	fetal_health
fetal_health	1.000000
prolongued_decelerations	0.484859
abnormal_short_term_variability	0.471191
percentage_of_time_with_abnormal_long_term_variability	0.426146
histogram_variance	0.206630
baseline value	0.148151
severe_decelerations	0.131934
fetal_movement	0.088010
histogram_min	0.063175
light_decelerations	0.058870
histogram_number_of_zeroes	-0.016682
histogram_number_of_peaks	-0.023666
histogram_max	-0.045265
histogram_width	-0.068789
mean_value_of_short_term_variability	-0.103382
histogram_tendency	-0.131976
uterine_contractions	-0.204894
histogram_median	-0.205033
mean_value_of_long_term_variability	-0.226797
histogram_mean	-0.226985
histogram_mode	-0.250412
accelerations	-0.364066

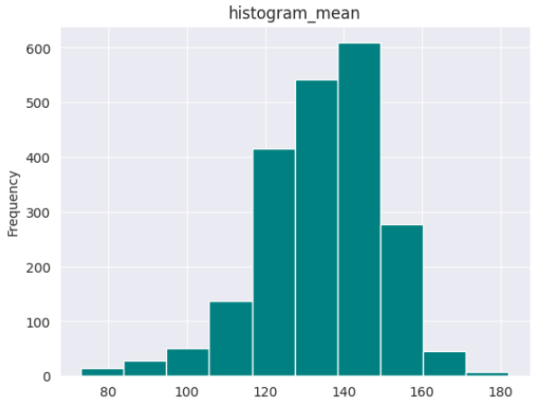
Ta thấy, thuộc tính mục tiêu có: mối tương quan nghịch (mạnh) với accelerations, nghĩa là có ít trường hợp xảy ra bệnh khi có gia tốc nhịp tim thai cao và ngược lại, mối tương quan thuận (nhẹ) với baseline value, nghĩa là có một cơ hội nhỏ rằng chỉ số nhịp tim tăng có thể chẩn đoán là có bệnh, mối tương quan nghịch (nhẹ) với uterine_contractions, việc tử cung ít co bóp thì có thể thai nhi đang mang bệnh, mối tương quan thuận (mạnh) với prolongued_decelerations, giảm đột ngột kéo dài có khả năng cao liên quan đến trường hợp có bệnh. Mối tương quan thuận (mạnh) với abnormal_short_term_variability, sự biến thiên ngắn hạn bất thường xảy ra càng nhiều thì khả năng thai nhi bị bệnh càng cao, mối tương quan thuận (mạnh) với percentage_of_time_with_abnormal_long_term _variability, tỷ lệ thời gian có biến thiên dài hạn bất thường cao với các trường hợp có bệnh.

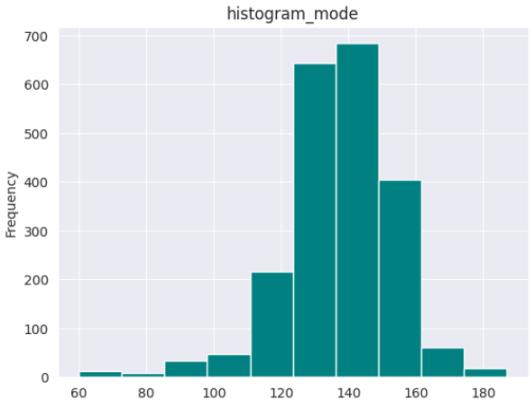
Các đặc trưng được phân phối theo phân phối chuẩn: baseline_value, abnormal_short_term_variability, histogram_max, histogram_mode, histogram_mean, histogram_median. Những đặc trưng khác có thể bị lệch và cần được điều chỉnh.

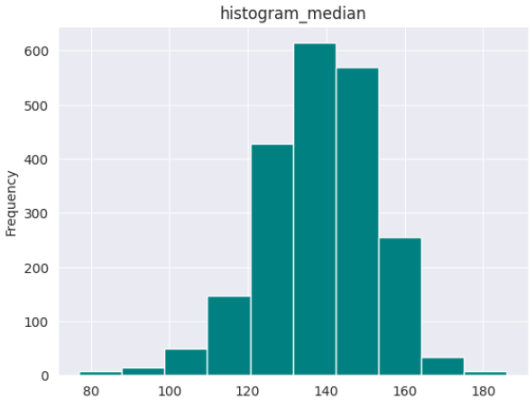




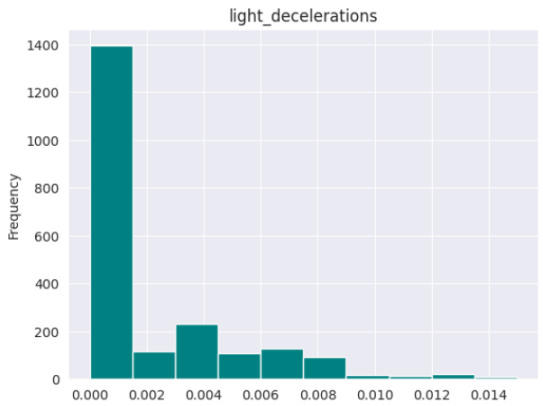


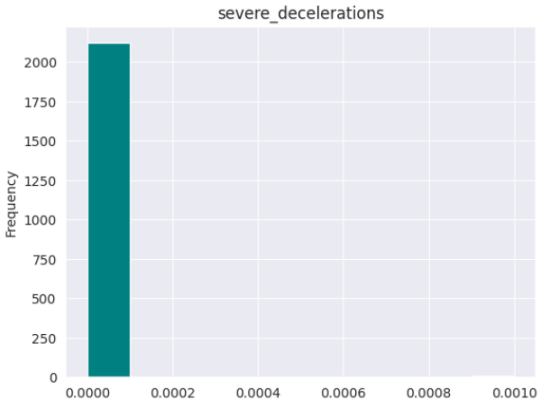


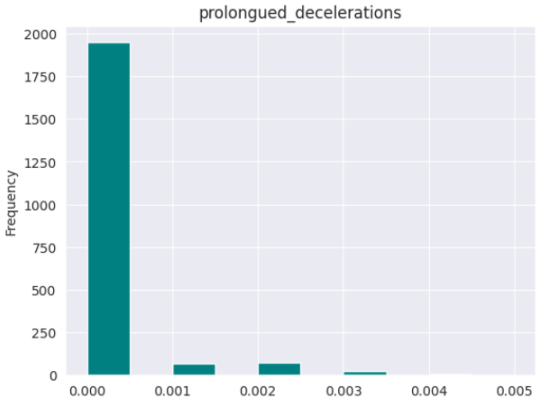


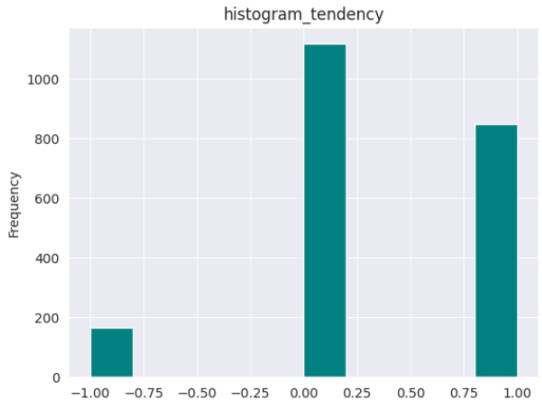


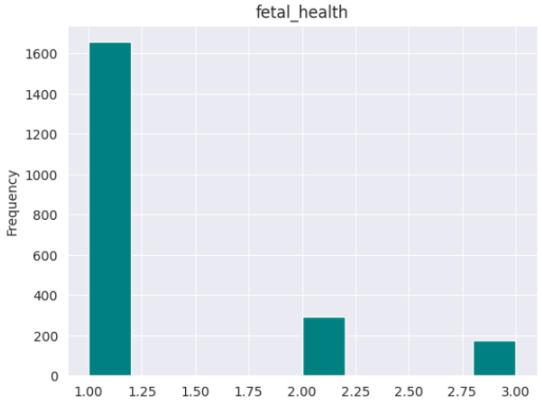
Các biểu đồ: light_decelerations, severe_decelerations, prolongued_decelerations cho thấy có rất nhiều biến có giá trị bằng không. Chúng cần được xử lý trước khi đưa vào mô hình huấn luyện.











Dữ liệu của đặc trưng 'histogram_tendency' thực ra là các biến phân loại tương tự các biến mục tiêu 'fetal_health'

III. VẤN ĐỀ LIÊN QUAN - RELATED WORK

Các thuật toán Máy học có thể giúp các chuyên gia y tế đưa ra quyết định sớm trong các tình huống phức tạp như chuẩn đoán, giảm tỷ lệ các biến chứng trong quá trình sinh. Nhóm sử dụng ba phương pháp tiêu chuẩn để phân loại là SVM, KNN và RF. Cả ba mô hình đều mang lại hiệu suất cao hơn so với mô hình Logistic Regression trong việc dự đoán sức khỏe thai nhi đối với dữ liệu mới, giảm overfitting, khả năng xử lý các tập dữ liệu lớn mà không cần quá nhiều thời gian huấn luyện. Tuy nhiên mô hình SVM có phần tốt hơn hai mô hình còn lại do khả năng tạo ra các siêu phẳng phức tạp để phân loại các lớp dữ liệu phi tuyến. Xếp sau là mô hình Random Forest, do mô hình này kết hợp dự đoán từ nhiều cây khác nhau giúp giảm overfitting tốt với các dữ liệu lớn, nhưng với dữ liệu nhỏ thì overfitting vẫn có thể xảy ra. Mô hình KNN có thể được áp dụng cho dữ liệu không thể mô tả dưới dạng vector đặc trưng nếu có một độ đo tương đồng có sẵn. Trong đa số các lần huấn luyện, thuật toán KNN cho độ chính xác chưa bằng hai mô hình trên một phần do

cơ chế "biểu quyết đa số" có thể làm việc dự đoán dữ liệu mới bị sai lệch.

Tín hiệu CTG được sử dụng để đánh giá trực tiếp nhịp tim của bệnh nhân, cung cấp kết quả chính xác và cập nhật cho các chuyên gia y tế. CTG có trách nhiệm đo nhịp tim của thai nhi, đo tần suất chuyển động của thai nhi và sự co bóp của tử cung, do đó nó đóng vai trò quan trọng trong việc đánh giá thai nhi trước khi sinh và trong quá trình sinh.

Mô hình SVM được biểu diễn trong không gian vector, phương pháp này tìm ra một siêu phẳng tốt nhất có thể chia các điểm dữ liệu trên không gian thành hai lớp riêng biệt tương ứng. Chất lượng của siêu phẳng này được quyết định bởi khoảng cách (gọi là biên) của điểm dữ liệu gần nhất thuộc mỗi lớp đến siêu phẳng. Tuy nhiên trên thực tế, có rất nhiều trường hợp dữ liệu phân tách phi tuyến tính, vì thế việc lựa chọn kernel phù hợp vẫn là thách thức của thuật toán SVM và Nguyễn Minh Duy sẽ phân tích chi tiết hơn về thuật toán này ở phần IV.

Tiếp theo, mô hình KNN sẽ được Huỳnh Võ Ngọc Thanh nghiên cứu cụ thể trong phần IV. Thuật toán này không học ngay từ tập dữ liệu huấn luyện mà thay vào đó, nó chủ yếu dựa vào bộ nhớ để lưu trữ tất cả dữ liệu huấn luyện và khi nhận được dữ liệu mới nó phân loại dữ liệu đó vào một hạng mục tương tự với dữ liệu sẵn có. Một điểm đặc biệt của thuật toán KNN là nó nhạy cảm với cấu trúc cục bộ của dữ liệu và việc tìm ra tham số k nào thật sự phù hợp với từng bộ dữ liệu là một vấn đề khi sử dụng thuật toán này.

Mô hình phân loại cuối cùng mà nhóm sử dụng là mô hình RF do Nguyễn Trần Hoài Bão trình bày ở phần IV. Đây là mô hình dựa trên việc kết hợp nhiều cây quyết định (Decision Tree) thành một "rừng" (forest) để đưa ra dự đoán. Random Forest đưa dữ liệu cần phân loại qua mỗi cây quyết định trong mô hình. Mỗi cây quyết định này đưa ra dự đoán riêng và cuối cùng kết quả dự đoán được xác định dựa trên quy tắc "phiếu bầu đa số" (majority voting) cho bài toán phân loại hoặc tính trung bình cho bài toán hồi quy. Cần xác định số lượng cây quyết định phù hợp để cân bằng giữa hiệu suất và tốc độ tính toán.

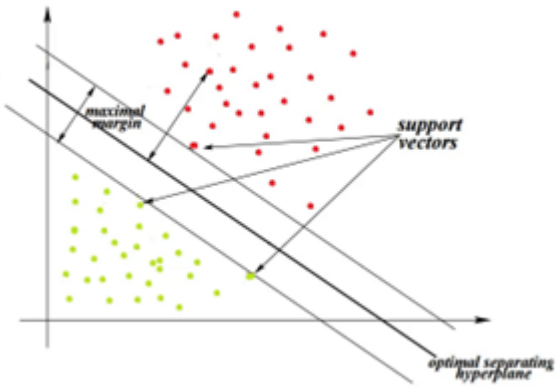
IV. CÁC PHƯƠNG PHÁP - OUR METHORDS

A. Support Vector Machines

1) Giới thiệu sơ lược về SVM:

Thuật toán Support Vector Machines (SVM) là một trong số những thuật toán máy học phân loại phổ biến và được sử dụng nhiều nhất trước khi mạng nơ ron nhân tạo phát triển. Là thuật toán học có giám sát phát triển vào những năm 1990 được sử dụng cho các bài toán phân loại và hồi quy.

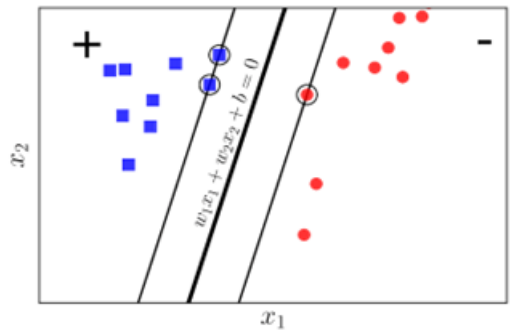
Mục tiêu của SVM là xây dựng một siêu mặt phẳng trong không gian đa chiều (N chiều), hiệu quả trong phân chia các điểm dữ liệu thuộc các lớp tương ứng khác nhau. Siêu phẳng này được tối ưu dựa trên một số điểm dữ liệu gọi là vector hỗ trợ (support vectors), nằm gần nhất với ranh giới giữa các lớp. SVM cố gắng tối đa hóa khoảng cách giữa siêu phẳng và các vector hỗ trợ, được gọi là biên (margin) nhằm đảm bảo tính tổng quát và khả năng phân loại tốt trên dữ liệu mới.



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

2) Xây dựng bài toán tối ưu cho SVM:

Giả sử rằng các cặp dữ liệu của training set là $(x_1, y_1), (x_2, y_2), \ldots, (x_N, y_N)$ với vector $x_i \in R_d$ thể hiện đầu vào của một điểm dữ liệu và y_i là nhãn của điểm dữ liệu đó. d là số chiều của dữ liệu và N là số điểm dữ liệu. Giả sử rằng nhãn của mỗi điểm dữ liệu được xác định bởi $y_i = 1$ (class 1) hoặc $y_i = -1$ (class 2).



MỤC TIÊU: tìm ra một siêu phẳng *(hyperplane)* trong không gian d chiều làm đường biên phân chia sao cho độ rộng **Margin** của chúng là **lớn nhất** vì khi phân chia theo đường biên này thì các nhóm là tách biệt nhất.

Giả sử rằng các điểm vuông xanh thuộc class 1, các điểm tròn đỏ thuộc class -1 và siêu phẳng d : $w^Tx + b = 0$ là mặt phân chia giữa hai classes.

Khoảng cách từ điểm $x_0(x_1, x_2, \ldots, x_n)$ đến mặt phân chia là:

$$D(x_0, d) = \frac{|w^Tx_0 + b|}{||w||_2}$$

Xét bỏ dấu giá trị tuyệt đối:

$$|w^Tx_i + b| = y_i(w^Tx_i + b) \text{ với } y_i = \{-1, 1\}$$

- Xét trường hợp $y_i = -1 < 0$ thì điểm Z_i nằm ở mặt âm (class 1) và có $w^Tx_i + b \leq 0$ do đó $y_i(w^Tx_i + b) \geq 0$ (Thỏa mãn dấu trị tuyệt đối).
- Xét trường hợp $y_i = 1 > 0$ thì điểm Z_i nằm ở mặt dương (class 2) và có $w^Tx_i + b \geq 0$ do đó $y_i(w^Tx_i + b) \geq 0$ (Thoả mãn dấu trị tuyệt đối).

$\Rightarrow$ Để tồn tại khoảng cách giữa 1 điểm đến với siêu phẳng thì sẽ luôn tồn tại ràng buộc: (ĐIỀU KIỆN TỒN TẠI MARGIN) (1)

$y_i(w^Tx_i + b) \geq k$ (Chọn k = 1 để thuận tiện tính toán vì k không đổi)

Để tính độ dài của Margin xét 2 vector X^+, X^- của 2 điểm gần nhau thuộc 2 lớp khác nhau:

$$(X^+ - X^-).\widehat{w^T}$$

$$\Leftrightarrow (X^+ - X^-).\frac{w^T}{||w||_2}$$

$$\Leftrightarrow \frac{(X^+.w^T - X^-.w^T}{||w||_2}$$

Mà:

$$\Leftrightarrow X^+.w^T + b = 1$$

$$\Leftrightarrow X^-.w^T + b = -1$$

$$\Rightarrow Margin = \frac{2}{||w||_2}$$

Theo như mục tiêu của bài toán ta sẽ tìm trọng số w và b sao cho Margin chúng ta tìm được là lớn nhất.

$$w, b = argmax(\frac{2}{||w||_2})$$

$$\Leftrightarrow w, b = argmin(\frac{1}{2}\frac{2}{||w||_2^2}) \quad (2) \text{ (Thêm } \tfrac{1}{2} \text{ để tiện tính toán về sau)}$$

Từ (1) và (2) ta có:

$$(3) \begin{cases} w, b = argmin(\frac{1}{2}\frac{2}{||w||_2^2}) \\ y_i(w^Tx_i + b) \geq 1 \end{cases}$$

Trong bài toán (3), hàm mục tiêu là một norm, nên là một hàm lồi. Các hàm bất đẳng thức ràng buộc là các hàm tuyến tính theo w và b, nên chúng cũng là các hàm lồi. Vậy bài toán tối ưu (3) có hàm mục tiêu là lồi, và các hàm ràng buộc cũng là lồi, nên nó là một bài toán lồi. Hơn nữa, nó là một Quadratic Programming. Thậm chí, hàm mục tiêu là strictly convex vì $||w||_2 = w^TIw$ và I là ma trận đơn vị - là một ma trận xác định dương. Từ đây có thể suy ra nghiệm cho SVM là duy nhất.

Đến đây thì bài toán này có thể giải được bằng các công cụ hỗ trợ tìm nghiệm cho Quadratic Programing, ví dụ CVXOPT.

3) Giới thiệu về Kernel SVM:

Ý tưởng cơ bản của Kernel SVM và các phương pháp kernel nói chung là tìm một phép biến đổi sao cho dữ liệu ban đầu là không phân biệt tuyến tính được biến sang không gian mới. Ở không gian mới này, dữ liệu trở nên phân biệt tuyến tính.

Nói một cách ngắn gọn, Kernel SVM là việc đi tìm một hàm ánh xạ biến đổi dữ liệu x từ không gian feature ban đầu thành dữ liệu trong một không gian mới bằng hàm số $\phi(x)$. Kernel là một hàm đo đạc sự tương đồng giữa hai điểm dữ liệu trong không gian ban đầu. Nó tính toán độ tương đồng dựa trên các đặc trưng của các điểm dữ liệu và sử dụng kết quả để xác định khoảng cách giữa chúng

trong không gian mới. Kernel cho phép chúng ta ánh xạ dữ liệu vào không gian mới, nơi các điểm dữ liệu có thể được phân loại tuyến tính bằng một siêu mặt phẳng.

Không phải hàm $k()$ bất kỳ nào cũng được sử dụng. Các hàm kerrnel cần có các tính chất:

- Đối xứng: $k(x, z) = k(z, x)$
- Thỏa mãn điều kiện Mercer:
  $$\sum_{n=1}^{N}\sum_{m=1}^{N} k(x_m, x_n)c_n c_m \geq 0,$$
  $$\forall c_i \in R, i = 1, 2, .., N_0$$

Từ những ràng buộc trên ta suy ra bài toán tối ưu phải là lồi và hàm mục tiêu là một hàm lồi (một quadratic form).

4) Ưu điểm, nhược điểm:

Ưu điểm:

- Xử lý dữ liệu phi tuyến: Kernel SVM cho phép phân loại các dữ liệu phi tuyến. Nhờ việc ánh xạ dữ liệu vào không gian cao hơn, SVM có khả năng tạo ra các đường ranh giới phức tạp hơn để phân loại các lớp dữ liệu không tuyến tính.
- Tính linh hoạt: Kernel SVM linh hoạt trong việc áp dụng các kernel khác nhau. Có nhiều loại kernel để lựa chọn, bao gồm kernel đa thức, kernel Gaussian, kernel tuyến tính và nhiều kernel khác. Điều này cho phép SVM thích ứng với đa dạng các bài toán phân loại và mô hình dữ liệu.
- Hiệu suất tốt trên tập dữ liệu lớn: SVM xử lý tốt trên các tập dữ liệu có kích thước lớn. Với việc sử dụng các vector hỗ trợ, SVM chỉ giữ lại một số điểm dữ liệu quan trọng trong quá trình huấn luyện, giúp tiết kiệm bộ nhớ và tính toán hiệu quả.

Nhược điểm:

- Độ phức tạp tính toán: SVM có độ phức tạp tính toán cao, đặc biệt là khi sử dụng các kernel phức tạp. Việc tính toán trong không gian cao hơn có thể tạo ra một số thách thức về hiệu suất và tốn nhiều thời gian huấn luyện.
- Phụ thuộc vào lựa chọn Kernel: Hiệu suất của Kernel SVM phụ thuộc rất nhiều vào lựa chọn kernel phù hợp. Việc chọn sai kernel có thể dẫn đến kết quả phân loại không tốt hoặc overfitting.
- Đối với dữ liệu có số chiều cao: Khi số lượng thuộc tính của dữ liệu lớn hơn số lượng mẫu, SVM có thể không hoạt động hiệu quả và cho kết quả không tốt.

Tóm lại, Kernel SVM là một công cụ mạnh mẽ để phân loại dữ liệu phi tuyến tính. Tuy nhiên, việc lựa chọn kernel phù hợp và xử lý dữ liệu có số chiều cao vẫn là thách thức đối với SVM.

B. K-Nearest Neighbor

1) Thuật toán K-Nearest Neighbor:

Thuật toán K-Nearest Neighbor (KNN) là một phương pháp học có giám sát và được sử dụng để phân loại và hồi quy. Đối với KNN, không có dạng hàm ánh xạ được xác định trước. Đây cũng được gọi là một thuật toán học lười vì nó không học ngay từ tập dữ liệu huấn luyện mà thay vào đó, nó chủ yếu dựa vào bộ nhớ để lưu trữ tất cả dữ liệu huấn luyện và khi nhận được dữ liệu mới nó phân loại dữ liệu đó vào một hạng mục tương tự với dữ liệu sẵn có.

Trong giai đoạn phân loại, k là hằng số do người dùng xác định. Nhiệm vụ của thuật toán KNN là phân loại một đối tượng theo đa số phiếu bầu của các 'hàng xóm' xung quanh nó, tức là đối tượng sẽ được gán vào lớp phổ biến nhất trong số k 'hàng xóm' gần nhất với nó. Nếu k = 1, thì đối tượng được gán đơn giản vào lớp gần nhất.

Các 'hàng xóm' được lấy từ một tập hợp các đối tượng đã biết lớp. Đây có thể coi là tập huấn luyện cho thuật toán, mặc dù không yêu cầu bước huấn luyện rõ ràng. Một điểm đặc biệt của thuật toán KNN là nó nhạy cảm với cấu trúc cục bộ của dữ liệu. Các ví dụ huấn luyện là các vectơ trong một không gian đặc trưng nhiều chiều, mỗi vectơ có một nhãn lớp. Giai đoạn đào tạo của thuật toán chỉ bao gồm lưu trữ các vectơ đặc trưng và nhãn lớp của các mẫu đào tạo. Mã giả để thực hiện thuật toán KNN:

1. Tải dữ liệu.
2. Chuẩn bị dữ liệu, xử lý giá trị thiếu và giảm chiều dữ liệu (nếu cần thiết).
3. Tìm giá trị tối ưu cho k:
   - Tính khoảng cách (q, x_i) với i=1,2,3,...,n. Trong đó, q là điểm dữ liệu mới, xi là

điểm dữ liệu huấn luyện, khoảng cách được tính dựa trên phương pháp khoảng cách đã chọn.

- Sắp xếp các khoảng cách này theo thứ tự tăng dần và đi kèm với dữ liệu huấn luyện tương ứng.
- Từ danh sách đã sắp xếp, chọn ra các k 'hàng xóm' có khoảng cách nhỏ nhất.
- Tìm lớp xuất hiện nhiều nhất trong k đã chọn, đây là lớp cần dự đoán.

2) Chọn siêu tham số k:

Nếu giá trị k quá nhỏ thì nhiễu sẽ phụ thuộc vào kết quả, mô hình có thể bị overfitting. Giá trị k quá lớn sẽ phá hủy nguyên tắc đằng sau KNN. Việc lựa chọn tốt nhất của k phần lớn sẽ phụ thuộc vào dữ liệu đầu vào vì dữ liệu có nhiều ngoại lệ hoặc nhiễu sẽ có khả năng hoạt động tốt hơn với giá trị k cao hơn. Nhìn chung, nên có một số lẻ cho k để tránh ràng buộc trong phân loại và có thể sử dụng chiến thuật xác thực chéo (cross validation) để giúp chọn ra k tối ưu cho tập dữ liệu. Về vấn đề này, bài báo xin đề cập thêm ở phần điều chỉnh tham số.

3) Khoảng cách:

Ta có một tập dữ liệu huấn luyện D gồm các mẫu huấn luyện $x_i(i \in n)$, với $n = |D|$. Mỗi dữ liệu huấn luyện được gắn nhãn với lớp $y_j \in Y$. Mục tiêu của ta là phân loại một dữ liệu không xác định q. Các dữ liệu được mô tả bằng một tập các đặc trưng F. Đối với mỗi x_i thuộc D, ta có thể tính khoảng cách giữa q và x_i như sau:

- Khoảng cách *Euclidean*: khoảng cách đường thẳng thông thường giữa hai điểm trong không gian Euclidean. Khoảng cách giữa dữ liệu mới và dữ liệu huấn luyện được tính:
$$d(q, x_i) = \sqrt{\sum_{f \in F} (q - x_i)^2}$$
- Khoảng cách *Manhattan*: là một phép đo khoảng cách khác phổ biến, nó đo giá trị tuyệt đối giữa hai điểm, được sử dụng khi các kiểu dữ liệu không đồng nhất.
$$d(q, x_i) = \sum_{f \in F} |q - x_i|$$
- Khoảng cách *Minkowski*: là một phép đo khoảng cách tổng quát của khoảng cách Euclidean và khoảng cách Manhattan. Tham số p trong công thức dưới đây cho phép tạo ra các phép đo khoảng cách khác:
$$d(q, x_i) = (\sum_{f \in F} |q - x_i|)^{\frac{1}{p}}$$

4) Tương đồng Cosin:

Tương đồng Cosin dùng để đo độ tương đồng giữa hai vectơ hoặc điểm dữ liệu trong không gian đa chiều. Hai vectơ cùng hướng có độ tương đồng Cosin là 1, hai vectơ vuông góc nhau có độ tương đồng Cosin là 0, và hai vectơ đối nhau theo đường kính có độ tương đồng Cosin là âm 1. Tương đồng Cosin thường được sử dụng trong không gian dương, nơi kết quả được giới hạn trong khoảng [0,1]. Tương đồng Cosin hoạt động với dữ liệu vectơ đặc trưng. Độ tương đồng Cosin giữa điểm truy vấn q và dữ liệu đầu vào x được tính như sau:

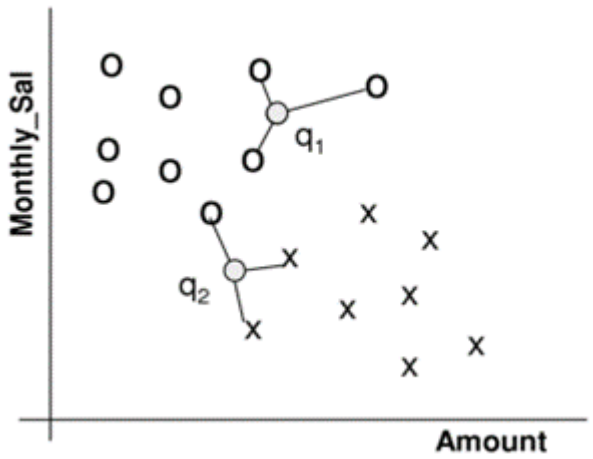
$$sim = cos(q, x_i) = \frac{\sum_{f \in F} (q.x_i)}{\sqrt{\sum_{f \in F} q^2} \sqrt{\sum_{f \in F} (x_i)^2}}$$

Tuy nhiên, tương đồng Cosin có thể không tận dụng độ lớn của vector mà chỉ tính theo hướng. Điều này vô tình làm mất mát thông tin so sánh.

5) Trọng số theo khoảng cách:

Hình dưới mô tả một bộ phân loại với k = 3 trên bài toán hai lớp trong không gian đặc trưng hai chiều. Trong ví dụ này, việc phân loại cho q_1 rất đơn giản vì cả ba 'hàng xóm' gần nhất với nó đều thuộc lớp O nên nó được phân loại là O. Trường hợp đối với q_2 phức tạp hơn vì nó có hai 'hàng xóm' thuộc lớp X và một 'hàng xóm' thuộc lớp O. Điều này có thể được giải quyết bằng *bỏ phiếu theo đa số* hoặc bằng *bỏ phiếu có trọng số theo khoảng cách*.

Tuy nhiên, khi phân loại *biểu quyết theo đa số* các điểm dữ liệu của một lớp phổ biến hơn có xu hướng thống trị dự đoán của dữ liệu mới do số lượng lớn của chúng. Để khắc phục vấn đề này, *phân loại theo trọng số* có tính đến khoảng cách từ điểm kiểm tra đến mỗi k 'hàng xóm' gần nhất của nó.



Để gán trọng số lớn hơn cho các 'hàng xóm' gần hơn trong việc quyết định lớp của điểm truy vấn. Một kỹ thuật khá tổng quát là bỏ phiếu có *trọng số theo khoảng cách*, trong đó các 'hàng xóm' được phép bỏ phiếu cho lớp của điểm truy vấn với trọng số là nghịch đảo khoảng cách của chúng đến điểm truy vấn:

$$Vote(y_j) = \sum_{c=1}^{k} \frac{1}{d(q, x_c)^p} 1(y_j, y_c)$$

Số phiếu được gán cho lớp y_j bởi 'hàng xóm' x_c là một chia cho khoảng cách đến 'hàng xóm' đó, tức là $1(y_j, y_c)$ trả về 1 nếu nhãn khớp và 0 nếu ngược lại.

Để những điểm gần nhất trong k 'hàng xóm' có ảnh hưởng lớn hơn đến kết quả của điểm truy vấn, bài báo giới thiệu một tập hợp các trọng số W, được xác định bởi sự gần nhau tương đối của mỗi 'hàng xóm' đối với điểm truy vấn:

$$W(q, x_c) = \frac{exp(-d(q, x_c))}{\sum_{c=1}^{k} exp(-d(q, x_c))}$$

6) Tỷ lệ lỗi:

Đối với phân loại KNN đa lớp , Cover và Hart (1967) chứng minh tỷ lệ lỗi giới hạn trên là:

$$R_B \leq R_{KNN} \leq R_B \, (2 - \frac{MR_B}{M-1})$$

Trong đó: R_B là tỷ lệ lỗi Bayes (tỷ lệ lỗi tối thiểu có thể đạt được dựa trên phân phối dữ liệu), R_{KNN} là tỷ lệ lỗi KNN, M là số lớp.

C_n^{KNN} biểu thị k bộ phân loại 'hàng xóm' gần nhất dựa trên tập huấn luyện có kích thước n. Rủi ro vượt mức (the excess risk) được định nghĩa là:

$$R(C_n^{KNN}) - (C^{Bayes})$$
$$= \{b_1 \frac{1}{k} + b_2 . (\frac{k}{n})^{(4/d)}\}\{1 + o(1)\}.$$

Trong đó: b_1, b_2 là hằng số, d là khoảng cách đến 'hàng xóm'.

7) Ưu điểm, nhược điểm:

Ưu điểm:

- KNN có thể được áp dụng cho dữ liệu mà không thể mô tả dưới dạng vector đặc trưng nếu có một độ đo tương đồng có sẵn. Do đó, KNN có thể được sử dụng trong những tình huống mà các thuật toán máy học khác không thể áp dụng.
- KNN có một số kỹ thuật giảm nhiễu có thể cải thiện độ chính xác của bộ phân loại.
- Trong một số trường hợp, các cơ chế như Kd-Trees hoặc Ball Trees có thể cải thiện thời gian truy xuất mà không làm giảm độ chính xác.

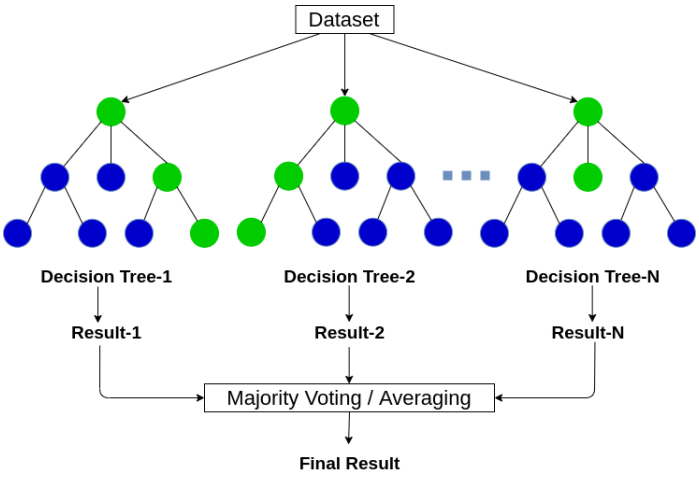
Nhược điểm:

- Vì phân loại "biểu quyết theo đa số" cơ bản xảy ra khi phân phối lớp bị sai lệch, các đối tượng của một lớp phổ biến hơn có xu hướng thống trị dự đoán của đối tượng mới.
- KNN rất nhạy cảm với các đặc trưng không liên quan hoặc trùng lặp vì tất cả các đặc trưng đóng góp vào độ tương đồng. Điều này có thể được cải thiện bằng việc lựa chọn đặc trưng hoặc đánh trọng số cho đặc trưng một cách cẩn thận.

C. RandomForest

1) Giới thiệu về Random Forest:

Thuật toán Random Forest là một phương pháp học có giám sát mạnh mẽ, được sử dụng cho cả phân loại (classification) và hồi quy (regression). Nó là một phương pháp linh hoạt và dễ sử dụng. Random Forest bao gồm một tập hợp các cây quyết định (decision trees), và càng có nhiều cây thì "rừng" càng mạnh, càng chính xác nhưng thời gian huấn luyện là một trở ngại.



Mô hình Random Forest được huấn luyện dựa trên sự phối hợp giữa *kết hợp (ensembling)* và quá trình *lấy mẫu tái lặp (boostrapping)*. Cụ thể thuật toán này tạo ra nhiều cây quyết định mà mỗi cây quyết định được huấn luyện dựa trên nhiều mẫu con khác nhau và kết quả dự báo là bầu cử (voting) từ toàn bộ những cây quyết định. Như vậy một kết quả dự báo được tổng hợp từ nhiều mô hình nên kết quả của chúng sẽ không bị chệch. Đồng thời kết hợp kết quả dự báo từ nhiều mô hình sẽ có phương sai nhỏ hơn so với chỉ một mô hình. Điều này giúp cho mô hình khắc phục được hiện tượng quá khớp (overfitting).

Thuật toán Random Forest cung cấp một chỉ báo quan trọng về đặc trưng, giúp đánh giá mức độ quan trọng của từng đặc trưng đối với việc phân loại hoặc dự đoán. Đối với bài toán phân loại: Random Forest có khả năng phân loại dữ liệu dựa trên các đặc trưng đã được học từ tập huấn luyện. Nó có thể xác định lớp/phân nhóm chính xác của các mẫu dữ liệu mới dựa trên quyết định của nhiều cây quyết định.

2) Kết hợp và lấy mẫu tái lặp

Mô hình kết hợp (ensemble model) là một phương pháp trong học máy nhằm kết hợp các dự đoán từ nhiều mô hình con khác nhau để đưa ra dự đoán cuối cùng. Mục tiêu của mô hình kết hợp là tận dụng sự đa dạng và sự khác biệt giữa các mô hình con để cải thiện tính ổn định và độ chính xác của dự đoán.

Có nhiều phương pháp mô hình kết hợp như Bagging, Boosting và Stacking. Trong đó, Bagging là một phương pháp quan trọng, và nó liên quan mật thiết đến *lấy mẫu tái lập (boostrapping)*.

Các mô hình con trong mô hình kết hợp (ví dụ: trong Bagging) được huấn luyện trên các tập dữ liệu con này. Do mỗi tập dữ liệu con được tạo bằng cách lấy mẫu tái lập, các mô hình con sẽ có sự đa dạng trong việc nhìn nhận dữ liệu. Sau đó, các dự đoán từ các mô hình con được kết hợp lại để tạo ra dự đoán cuối cùng.

Giả định dữ liệu huấn luyện mô hình là một tập $D = \{(x_1, y_1), (x_2, y_2), ...$
$, (x_N, y_N)\}$ bao gồm N quan sát. Thuật toán Random Forest sẽ sử dụng phương pháp lấy mẫu tái lập để tạo thành B tập dữ liệu con. Quá trình lấy mẫu tái lập này còn gọi là *bỏ túi* (bagging). Tức là chúng ta sẽ thực hiện M lượt nhặt các mẫu từ tổng thể và bỏ vào túi để tạo thành tập $B_i = \{(x_1^{(i)}, y_1^{(i)}), (x_2^{(i)}, y_2^{(i)}), ..., (x_M^{(i)}, y_M^{(i)})\}$

Tập B_i cho phép các phần tử được lặp lại. Như vậy sẽ tồn tại những quan sát thuộc D nhưng không thuộc B_i. Đây là những quan sát chưa được bỏ vào túi và chúng ta gọi chúng là *nằm ngoài túi (out of bag)*.

Với mỗi tập dữ liệu B_i chúng ta xây dựng một mô hình cây quyết định và trả về kết quả dự báo là $\hat{y}_j^{(i)} = f_i(x_j)$. Trong đó $\hat{y}_j^{(i)}$ là dự báo của quan sát thứ j từ mô hình thứ (i), x_j là giá trị véc tơ đầu vào, $f_i(.)$ là hàm dự báo của mô hình thứ i. Mô hình dự báo từ cây quyết định là giá trị trung bình hoặc bầu cử của B cây quyết định.

- *Đối với mô hình dự báo*: Chúng ta tính giá trị trung bình của các dự báo từ mô hình con.

$$\hat{y}_j = \frac{1}{B}\sum_{i=1}^{B}\hat{y}_j^{(i)}$$

- *Đối với mô hình phân loại*: Chúng ta thực hiện bầu cử từ các mô hình con để chọn ra nhãn dự báo có tần suất lớn nhất.

$$\hat{y}_j = argmax_c \sum_{i=1}^{B} p(\hat{y}_j^{(i)} = c)$$

Như vậy phương sai của mô hình trong trường hợp đối với bài toán dự báo:

$$\sigma_{\hat{y}}^2 = Var(\frac{1}{B}\sum_{i=1}^{B}\hat{y}^{(i)}) =$$

$$= \frac{1}{B^2}[\sum_{i=1}^{B} Var(\hat{y}^{(i)}) + 2\sum_{1\le m<n\le B} cov(y^{(m)}, y^{(n)})]$$

Do kết quả của mô hình con A không chịu ảnh hưởng hoặc phụ thuộc vào mô hình con B nên ta có thể giả định kết quả dự báo từ các mô hình là hoàn toàn độc lập nhau. Tức là ta có $cov(y^{(m)}, y^{(n)}) = 0, \forall 1 \leq m < n \leq B$. Đồng thời giả định chất lượng các mô hình là đồng đều, được thể hiện qua phương sai dự báo là đồng nhất $Var(\hat{y}^{(i)}) = \sigma^2, \forall i = \overline{1, B}$. Từ đó suy ra:

$$\sigma_{\hat{y}}^2 = \frac{1}{B^2}[\sum_{i=1}^{B} Var(\hat{y}^{(i)})]$$
$$= \frac{1}{B^2} B\sigma^2 = \frac{1}{B}\sigma^2$$

Như vậy nếu sử dụng dự báo là trung bình kết hợp từ nhiều mô hình cây quyết định thì phương sai có thể giảm B lần so với chỉ sử dụng một mô hình duy nhất. Trong một mô hình Random Forest, số lượng các cây quyết định là rất lớn. Do đó phương sai dự báo từ mô hình có thể giảm gấp nhiều lần và tạo ra một dự báo ổn định hơn.

3) Ưu điểm, nhược điểm:

Ưu điểm:

- Random Forest được huấn luyện trên nhiều tập dữ liệu con khác nhau, bao gồm cả tập loại bỏ outliers. Điều này giúp mô hình ít bị nhạy cảm với dữ liệu outliers hơn.
- Với cây quyết định đơn lẻ, có thể dễ dàng gặp tình trạng overfitting khi không giới hạn độ sâu của cây. Tuy nhiên, Random Forest sử dụng nhiều cây độc lập, làm giảm khả năng overfitting của từng cây riêng lẻ.
- Với dữ liệu nhiều chiều, Random Forest tỏ ra mạnh mẽ hơn. Mỗi cây trong Random Forest được huấn luyện trên các bộ dữ liệu con khác nhau, tạo ra sự đa dạng trong quyết định của mô hình và giúp tổng quát hóa trên dữ liệu nhiều chiều.
- Kết quả dự đoán của mô hình Random Forest thường có độ chính xác cao. Bằng cách kết hợp các dự đoán từ nhiều cây khác nhau, mô hình Random Forest tận dụng được sự đa dạng và tổng quát hóa, dẫn đến kết quả dự đoán chính xác và đáng tin cậy.

Nhược điểm:

- Một mô hình Random Forest có thể bao gồm hàng trăm cây quyết định, điều này làm cho việc hiểu và diễn giải mô hình trở nên khó khăn. Không dễ dàng xác định quy tắc hoặc mức độ quan trọng của từng đặc trưng.
- Random Forest có thể mất thời gian đáng kể để huấn luyện, đặc biệt là khi có một số lượng cây lớn và tập dữ liệu lớn.
- Random Forest có khả năng xử lý overfitting tốt với các tập dữ liệu lớn, nhưng khi sử dụng tập dữ liệu nhỏ, có nguy cơ mô hình trở nên quá phức tạp và overfitting có thể xảy ra.
- Một Random Forest với số lượng cây lớn và đặc trưng nhiều có thể yêu cầu một lượng tài nguyên lớn để huấn luyện và dự đoán. Điều này có thể gây khó khăn khi triển khai mô hình trên các hệ thống có tài nguyên hạn chế.

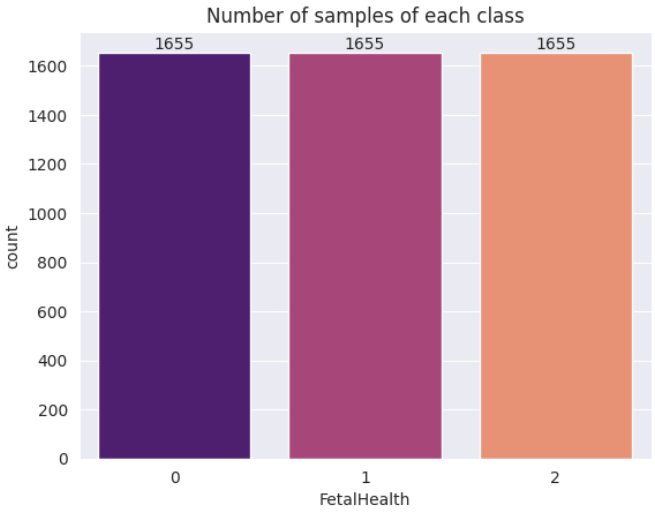
Tóm lại, Random Forest là một thuật toán mạnh mẽ trong phân loại sức khỏe thai nhi, với độ chính xác cao, khả năng xử lý tập dữ liệu lớn, và khả năng xử lý cả đặc trưng liên tục và không liên tục. Ngoài ra, nó giảm thiểu overfitting và đảm bảo tính ổn định. Tuy nhiên, việc hiểu và diễn giải mô hình có thể khó khăn, và mô hình yêu cầu tài nguyên cao khi có số lượng cây lớn. Tuy vậy, Random Forest vẫn là một lựa chọn hợp lý và mạnh mẽ trong bài toán này.

V. THỰC NGHIỆM

A. Xử lý dữ liệu

Để ba mô hình đạt được hiệu quả cao nhất, nhóm có đề xuất chuyển các giá trị ở cột mục tiêu từ 1,2,3 thành 0,1,2 để các thuật toán có thể hiểu được mối quan hệ thứ tự và khoảng cách giữa các giá trị phân loại bằng cách dùng LabelEncoder.

Qua trực quan hóa dữ liệu, ta nhận thấy đây là bộ dữ liệu mất cân bằng, tỷ lệ phân bố của nhãn 1 (Normal) quá cao so với hai nhãn còn lại, do đó nhóm đề xuất phương pháp Oversampling để cân bằng dữ liệu của ba lớp. Sau khi LabelEncoder và cân bằng dữ liệu ta có biểu đồ mô tả phân bố dữ liệu ở ba lớp 0 (Normal), 1 (Suspect), 2 (Pathological) như sau:



Đối với SVM, xét cùng bộ tham số [C: 10, gamma: 0.1, kernel: rbf], thì mô hình lúc cân bằng dữ liệu cho độ chính xác (accuracy) cao hơn lúc chưa cân bằng dữ liệu.

Tương tự, đối với KNN, xét cùng bộ tham số [algorithm: auto, leaf_size: 20, n_neighbors: 5, p: 1, weights: distance], thì mô hình lúc cân bằng dữ liệu cũng cho độ chính xác (accuracy) vượt trội hơn lúc chưa cân bằng dữ liệu.

Đối với RF, xét cùng bộ tham số [criterion: entropy, max_depth: 9, max_features: auto, n_estimators: 250], thì mô hình lúc cân bằng dữ liệu cho độ chính xác (accuracy) cao hơn lúc chưa cân bằng dữ liệu. Cụ thể như sau:

	SVM	KNN	RF
Non-Oversampling	92.2	91	95.5
Oversampling	97.1	97	97.3

B. Cross Validation

Cross validation là một cải tiến của validation với lượng dữ liệu trong tập validation là nhỏ nhưng chất lượng mô hình được đánh giá trên nhiều tập validation khác nhau. Một cách thường được sử dụng là chia tập training thành k tập con không có phần tử chung, với kích thước gần bằng nhau. Tại mỗi lần Kiểm thử, một trong số k tập con được lấy ra làm tập kiểm thử. Mô hình sẽ được xây dựng dựa vào tập hợp của k-1 tập con còn lại. Mô hình cuối được xác định dựa trên trung bình của các train error và validation error. Cách làm này còn có tên gọi là k-fold cross validation. Khi k bằng với số lượng phần tử trong tập training ban đầu, tức mỗi tập con có đúng 1 phần tử, ta gọi kỹ thuật này là leave-one-out.

Trong bài báo này, nhóm sẽ thực hiện huấn luyện và tinh chỉnh các mô hình đề xuất với Kfold Cross Validation được tích hợp sẵn trong GridSearch với CV = 5.

C. Điều chỉnh siêu tham số

Do bộ dữ liệu không quá lớn, các siêu tham số có thể được tìm thấy bằng cách thử tất cả các kết hợp và xem thông số nào hoạt động tốt nhất. Ý tưởng chính đằng sau nó là tạo một lưới các siêu tham số và chỉ cần thử tất cả các kết hợp của chúng. Phương pháp này được gọi là Gridsearch, và thư viện Sklearn có một công cụ là GridSearchCV hỗ trợ cho ta tìm được bộ siêu tham số tốt nhất cho mô hình theo ý tưởng như trên.

Tùy vào mỗi bộ dữ liệu sẽ phù hợp với một bộ tham số nhất định. Việc chọn ra siêu tham số nào để huấn luyện mô hình sẽ ảnh hưởng trực tiếp đến kết quả của mô hình:

1) Mô hình Support Vector Machine:

Các siêu tham số được sử dụng trong mô hình:

a) Kernel

Quá trình xây dựng siêu phẳng phân cách trong SVM được thực hiện qua các phép biến đổi đại số thông qua các Kernel như Linear, Polynomial, Radial Basic Function, Sigmoid. Siêu tham số này sẽ quyết định sử dụng phép biến đổi nào để phân tách dữ liệu cho phù hợp với bộ dataset.

b) Regularization

Tham số Regularization (được nhắc đến trong thư viên sklearn là tham số C) điều chỉnh việc có nên bỏ qua các điểm dữ liệu bất thường trong quá trình tối ưu mô hình SVM. Nếu tham số này có giá trị lớn, quá trình tối ưu sẽ chọn một siêu phẳng sao cho siêu phẳng này phân cách tất cả các điểm dữ liệu một cách tốt nhất, từ đó khoảng cách giữa siêu phẳng tới các điểm dữ liệu của 2 lớp sẽ có giá trị nhỏ (small-margin). Ngược lại, khi tham số này có giá trị nhỏ, siêu phẳng sẽ được xây dựng sao cho khoảng cách với các điểm dữ liệu của 2 lớp có giá trị lớn (large-margin), kể cả khi siêu phẳng này sẽ phân loại sai nhiều điểm dữ liệu hơn.

c) Gamma

Tham số gamma xác định việc sử dụng bao nhiêu

điểm dữ liệu cho việc xây dựng siêu phẳng phân cách. Với giá trị gamma nhỏ, các điểm dữ liệu nằm xa đường phân cách sẽ được sử dụng trong việc tính toán đường phân cách. Ngược lại, với giá trị gamma lớn, chỉ những điểm nằm gần đường phân cách mới được sử dụng để tính toán.

d) Đánh giá qua các dạng Kernel

Quá trình đánh giá riêng biệt model qua các dạng Kernel của SVM với các tham số mặc định (C = 100, gamma = auto).

	Accuracy	Precision	Recall	F1_score
Linear	0.89	0.89	0.89	0.89
Polinomial	0.95	0.96	0.95	0.95
Radial Basic Function	0.96	0.96	0.96	0.96
Sigmoid	0.71	0.72	0.71	0.71

Nhận xét: Từ bảng đánh giá trên ta có thể thấy được với bộ dataset này thì dữ liệu được phân bổ phức tạp theo hướng Polynomial trong không gian, nên khi train model với Kernel Polynomial hoặc Radial Basic Function sẽ đạt được kết quả tốt hơn. Tuy nhiên để đạt được hiệu quả cao nhất thì chúng ta sẽ train model với Kernal Radial Basic Function.

e) Đánh giá qua bộ tham số tốt nhất được tìm ra từ GirdSearchCV

SVM có một số tham số và việc tìm kiếm siêu tham số tối ưu là một nhiệm vụ rất khó giải quyết. Nhưng nó có thể được tìm thấy bằng cách thử tất cả các kết hợp và xem thông số nào hoạt động tốt nhất. Ý tưởng chính đằng sau nó là tạo một lưới các siêu tham số và chỉ cần thử tất cả các kết hợp của chúng. Do đó, phương pháp này được gọi là Gridsearch, và thư viện Sklearn có một công cụ là GridSearchCV hỗ trợ cho chúng ta có thể tìm được bô siêu tham số tốt nhất cho mô hình theo ý tưởng như trên.

GridSearchCV lấy một từ điển mô tả các tham số có thể được thử trên một mô hình để huấn luyện nó. Lưới tham số được định nghĩa như một từ điển, trong đó các khóa là các tham số và các giá trị là cài đặt cần kiểm tra.

Ở bộ dataset này thực hiện Turning tham số theo các giá trị như sau:

+ 'C' : [0.01, 0.1, 1, 10, 100].
+ 'gamma': [100, 10, 1, 0.1, 0.01].
+ 'kernel': ['linear', 'rbf', 'poly', 'sigmoid'].

Bộ tham số tốt nhất sau khi thực hiện Turning:

+ 'C': 100.
+ 'gamma': 10.
+ 'kernel': 'rbf'.

	Precision	Recall	F1_score
0	0.98	1.00	0.99
1	1.00	0.98	0.99
2	1.00	1.00	1.00

Accuracy: 0.994

2) Mô hình K-Nearest Neighbor:

Các siêu tham số được sử dụng trong mô hình:

a) n_neighbors

Đây chính là siêu tham số k, siêu tham số này sẽ quyết định chọn bao nhiêu điểm dữ liệu gần với điểm cần dự đoán để dự đoán nhãn cho nó. Không có giá trị k khuyên dùng cho tất cả bài toán vì nó phụ thuộc rất nhiều vào dữ liệu.

b) weights

Siêu tham số này có thể nhận một trong hai giá trị hàm cung cấp sẵn: "uniform" hoặc "distance", hoặc người dùng có thể tự cung cấp hàm để gán trọng số. "uniform" nghĩa là mô hình sẽ xem k điểm gần điểm đang xét có độ quan trọng là như nhau, hay có thể hiểu là trọng số cho các điểm đó đều là 1. "distance" thì mô hình sẽ gán trọng số khác nhau tùy thuộc vào khoảng cách giữa điểm đang xét và k điểm kia.

c) algorithm

Siêu tham số này quy định cách mô hình giải bài toán tìm k điểm gần nhất với dữ liệu là tất cả các khoảng cách đã được tính trước. Bài báo đề cập đến bốn giá trị sau: "brute", "ball_tree", "kd_tree", "auto". Mặc định là "auto". Khi tham số là "brute", nghĩa là mô hình dùng cách thủ công. Mô hình sẽ lần lượt so sánh tất cả các giá trị khoảng cách với nhau để tìm ra các điểm gần nhất. Với "ball_tree", mô hình sẽ lưu trữ các điểm có nhiều chiều vào cấu trúc ball_tree, một cấu trúc cây nhị phân nhưng nó bao gồm các "cluster"(cụ thể hơn là vòng tròn 2D hoặc hình cầu 3D) bao bọc lấy nhau. Với cấu trúc dữ liệu này, việc tìm ra k điểm gần nhất sẽ nhanh chóng hơn rất nhiều khi bộ dữ liệu lớn dần. "kd_tree" cũng tương tự như ball_tree, là một cấu trúc cây nhị phân dùng để lưu trữ các điểm dữ liệu nhiều chiều, nhưng sẽ chia đôi các điểm dữ liệu theo các giá trị trung bình do người dùng lựa chọn (mean, mode, median) theo từng trục lần lượt nhau.

Cuối cùng, “auto” sẽ giúp mô hình tự lựa chọn 1 trong 3 thuật toán trên để sử dụng tùy thuộc vào dataset truyền vào hàm. Tất cả thuật toán đều có ưu nhược điểm riêng: “brute” sẽ cho ra kết quả rất tốt đối với bộ dữ liệu nhỏ, nhưng khi kích thước bộ dữ liệu lớn dần thì 2 cấu trúc cây sẽ thể hiện tốt hơn.

d) leaf_size

Quy định kích thước của những node lá trong cấu trúc cây. Tham số này sẽ ảnh hưởng đến tốc độ hình thành nên cây cũng như lượng bộ nhớ cần thiết cho cấu trúc cây.

e) p

Siêu tham số này mang giá trị là số mũ trong độ đo Minkowski. Vì đối với độ đo này, với các giá trị số mũ khác nhau thì công thức sẽ khác nhau. Nếu p = 1, công thức sẽ tương đương khoảng cách Manhattan, p=2 là khoảng cách Euclidean, với giá trị p khác thì sẽ áp vào công thức gốc của Minkowski.

Bộ dữ liệu của nhóm không quá lớn, các tham số có thể được điều chỉnh cách thử tất cả các kết hợp và xem thông số nào hoạt động tốt nhất thì lấy. Do đó, nhóm đề xuất phương pháp được gọi là Gridsearch, và thư viện Sklearn có một công cụ là GridSearchCV hỗ trợ tìm được bô siêu tham số tốt nhất cho mô hình theo ý tưởng trên. Các tham số trong lưới GridSearch:

+ ’n_neighbors’ : [3,5,7,9].
+ ’leaf_size’: [10, 20, 30, 40].
+ ’p’: [1,2].
+ ’weights’: [’uniform’,’distance’].
+ ’algorithm’: [’auto’,’brute’,’ball_tree’,’kd_tree’].

GridSearchCV sẽ thực hiện huấn luyện và đánh giá mô hình với nhiều tham số khác nhau để chọn ra tham số tốt nhất cho mô hình. Kết quả thực nghiệm thu được bộ tham số:

+ ’n_neighbors’: 3.
+ ’leaf_size’: 10.
+ ’p’: 2.
+ ’weights’: ’distance’.
+ ’algorithm’: ’auto’.

Với bộ tham số trên, mô hình KNN đạt hiệu quả cao với:

	Precision	Recall	F1_score
0	0.99	0.94	0.96
1	0.95	0.99	0.97
2	1.00	1.00	1.00

Accuracy: 0.977

3) Mô hình Random Forest:

a) n_estimators

Đây là siêu tham số quan trọng nhất trong mô hình Random Forest. Nó xác định số lượng cây quyết định được sử dụng trong mô hình. Tăng giá trị n_estimators có thể làm tăng độ chính xác của mô hình nhưng đồng thời cũng làm tăng thời gian huấn luyện. Việc chọn giá trị tối ưu cho n_estimators thường chỉ quá trình tìm kiếm để tìm ra giá trị mà mô hình đạt được độ chính xác tốt nhất trên tập dữ liệu kiểm tra.

b) max_depth

Là siêu tham số quyết định độ sâu tối đa của các cây quyết định trong mô hình Random Forest. Khi max_depth được đặt là một giá trị cụ thể, cây quyết định sẽ không được phép phân chia thêm sau khi đạt đến độ sâu tối đa này. Điều này giúp kiểm soát tình trạng overfitting. Nếu giá trị max_depth quá nhỏ, mô hình có thể bị underfitting, và ngược lại, nếu giá trị quá lớn, mô hình có thể bị overfitting.

c) min_samples_split và min_samples_leaf

Đây là siêu tham số quyết định số lượng mẫu tối thiểu yêu cầu để tiếp tục chia một nút (split) hoặc tạo một lá (leaf) trong cây quyết định trong Random Forest. min_samples_split xác định số lượng mẫu tối thiểu để tiếp tục phân chia một nút và min_samples_leaf xác định số lượng mẫu tối thiểu để tạo thành một lá. Điều chỉnh các giá trị này giúp kiểm soát độ phức tạp và overfitting của mô hình. Tăng giá trị của hai siêu tham số có thể làm giảm khả năng xảy ra hiện tượng overfitting của mô hình.

d) max_features

Là siêu tham số xác định số lượng đặc trưng được xem xét trong quá trình tìm kiếm phân chia tốt nhất khi xây dựng các cây quyết định trong Random Forest. Giá trị mặc định là "auto" hoặc "sqrt" (căn bậc 2 của số lượng đặc trưng). Tuy nhiên, bạn có thể điều chỉnh giá trị max_features thành "log2", "None" (tất cả các đặc trưng), hoặc một giá trị cụ thể khác như 0.5 (một nửa số lượng đặc trưng). Trong bài báo là "auto". Việc điều chỉnh giá trị này có thể

ảnh hưởng đến độ chính xác và đa dạng của các cây quyết định trong mô hình Random Forest.

e) criterion

Đây là siêu tham số xác định phương pháp đánh giá chất lượng phân chia trong quá trình xây dựng các cây quyết định trong RandomForest. Trong bài báo này, nhóm sử dụng GridSearchCV để chọn ra criterion tốt nhất trong hai tham số "gini" và "entropy". "Gini" sử dụng chỉ số Gini để đo lường độ tinh khiết của phân chia, trong khi "entropy" sử dụng độ mất thông tin (entropy) để đo lường. Cả hai phương pháp đều hữu ích và thường cho kết quả tương tự. Việc chọn criterion phụ thuộc vào vấn đề cụ thể và thường được chọn mặc định là "gini".

Các tham số trong lưới GridSearch:

+ 'n_estimators' : [100, 150, 200, 250].
+ 'max_depth': [7, 8, 9, 10].
+ 'min_samples_split': [2, 5, 10].
+ 'min_samples_leaf': [1, 2, 4].
+ 'max_features': ['auto'].
+ 'criterion': ['gini', 'entropy'].

Sử dụng công cụ GridSearchCV để chọn ra bộ tham số tối ưu cho tập dữ liệu. Kết quả thu về mô hình Random Forest đạt kết quả đánh giá cao nhất với bộ tham số:

+ 'n_estimators' : 250.
+ 'max_depth': 10.
+ 'min_samples_split': 2.
+ 'min_samples_leaf': 1.
+ 'max_features': 'auto'.
+ 'criterion': 'entropy'.

Độ chính xác được đánh giá bằng các thang đo ứng với các giá trị tham số:

	Precision	Recall	F1_score
0	1.00	0.97	0.98
1	0.96	1.00	0.98
2	1.00	0.98	0.99

Accuracy: 0.983

D. Các thang đo đánh giá

1) Thang đo Accuracy:

Khi xây dựng mô hình phân loại chúng ta muốn biết một cách khái quát tỷ lệ các trường hợp được dự báo đúng trên tổng số các trường hợp là bao nhiêu. Tỷ lệ đó được gọi là độ chính xác (accuracy). Độ chính xác giúp ta đánh giá hiệu quả dự báo của mô hình trên một bộ dữ liệu. Độ chính xác càng cao thì mô hình của chúng ta càng chuẩn xác.

Công thức:

$$Accuracy = \frac{TP + TN}{totalsample}$$

True Positive (TP): số lượng điểm của lớp positive được phân loại đúng là positive.

True Negative (TN): số lượng điểm của lớp negative được phân loại đúng là negative.

False Positive (FP): số lượng điểm của lớp negative bị phân loại nhầm thành positive.

False Negative (FN): số lượng điểm của lớp positive bị phân loại nhầm thành negative.

2) Thang đo Precision:

Bằng trực giác, ta có thể đưa bài toán phân loại đa lớp về bài toán phân lớp nhị phân bằng cách xem xét từng lớp. Với mỗi lớp, ta coi dữ liệu thuộc lớp có label là positive, tất cả các dữ liệu còn lại có label là negative. Với mỗi lớp, ta sẽ nhận được một cặp giá trị Precision và Recall. Với các bài toán có nhiều lớp dữ liệu như hiện tại, hai phép đánh giá dựa trên Precision và Recall nên được sử dụng là micro-average và macro-average.

Macro-average precision là trung bình cộng của các precision theo lớp, tương tự với Macro-average recall.

Precision đo lường tỷ lệ của các trường hợp dự báo positive mà thực sự là positive, hay nói cách khác, số lượng dự đoán đúng positive so với tổng số các trường hợp dự đoán positive.

Công thức:

$$Precision = \frac{TP}{TP + FP}$$

3) Thang đo Recall:

Độ nhạy Recall là một thước đo quan trọng để đánh giá hiệu suất của mô hình phân loại, đặc biệt khi quan tâm đến nhóm positive (thực tế là positive). Recall đo lường tỷ lệ của các trường hợp positive thực tế mà mô hình đã dự đoán đúng, tức là số lượng trường hợp positive dự đoán đúng so với tổng số các trường hợp positive thực tế.

Công thức:

$$Recall = \frac{TP}{TP + FN}$$

4) Thang đo F1-score:

F1 score là một thước đo tổng hòa giữa precision và recall, thường được sử dụng để đánh giá hiệu suất tổng thể của mô hình phân loại. F1 score tính toán một giá trị đo lường sự cân bằng giữa việc đạt được độ chính xác cao (precision) và việc đảm bảo mức độ phủ sóng cao (recall).

Công thức:

$$\frac{2}{F1} = \frac{1}{Precision} + \frac{1}{Recall}$$

F1 score có giá trị nằm trong nửa khoảng (0,1]. F1 càng cao thì bộ phân lớp càng tốt. Khi cả recall và precision đều bằng 1 thì F1 = 1.

E. Đánh giá tổng quan:

Sau quá trình thực nghiệm huấn luyện nhiều trường hợp của mô hình để tìm ra bộ tham số tốt nhất cho mô hình ở mỗi phương pháp, nhóm đã tổng hợp được bản kết quả thực nghiệm độ chính xác (Accuracy) với mô hình tốt nhất ở mỗi phương pháp như sau:

	SVM	KNN	RF
Accuracy	0.994	0.977	0.983

Qua đó cho thấy mô hình SVM đạt hiệu quả cao nhất trong phạm vi bài báo này với độ chính xác 99.4%. Lí do là vì Kernel SVM cho phép tạo ra các đường ranh giới phức tạp hơn để phân loại các lớp dữ liệu không tuyến tính. Một phần do kỹ năng xử lý dữ liệu của nhóm trước khi huấn luyện mô hình (cân bằng dữ liệu, mã hóa, chuẩn hóa dữ liệu,...). Nhược điểm của SVM, phụ thuộc vào lựa chọn kernel. Việc chọn sai kernel có thể dẫn đến kết quả phân loại không tốt hoặc bị overfitting.

Theo sau là mô hình RF đạt độ chính xác 98.3%. Random Forest được huấn luyện trên nhiều tập dữ liệu con khác nhau, bao gồm cả tập loại bỏ outliers. Điều này giúp mô hình ít bị nhạy cảm với dữ liệu outliers hơn. Một Random Forest có nhiều đặc trưng có thể yêu cầu một lượng tài nguyên lớn để huấn luyện và dự đoán. Điều này có thể gây khó khăn khi triển khai mô hình trên các hệ thống có tài nguyên hạn chế.

Mô hình KNN chưa đạt hiệu quả bằng hai mô hình trên nhưng vẫn tốt với 97.7%. Có nhiều nguyên nhân nhưng nguyên nhân phổ biến do mô hình K-Nearest Neighbors là mô hình khá đơn giản, phân loại theo "biểu quyết theo đa số" có thể làm kết quả bị sai lệch nhưng không đáng kể, do các đối tượng của một lớp phổ biến hơn có xu hướng thống trị dự đoán của đối tượng mới. Trong một vài trường hợp, các cơ chế Kd-Trees hoặc Ball Trees có thể cải thiện thời gian truy xuất mà không làm giảm độ chính xác.

VI. Kết luận

Dựa vào các số liệu liên quan đến sức khỏe của thai nhi, và sử dụng các phương pháp máy học tiên tiến hiện nay, chúng tôi đã thành công xây dựng hệ thống cơ bản trong việc hỗ trợ dự đoán sức khỏe của thai nhi. Điều này chứng tỏ sức mạnh vượt trội của các phương pháp máy học trong lĩnh vực y học. Mặc dù các thực nghiệm hiện tại của nhóm chỉ tập trung chủ yếu trong phạm vi đồ án môn học, chúng tôi cam kết sẽ tiếp tục nghiên cứu và cải tiến hệ thống này. Chúng tôi sẽ tìm hiểu thêm về các mô hình mới và phương pháp cải tiến, cũng như tìm kiếm những nguồn dữ liệu uy tín khác và áp dụng các bước xử lý dữ liệu tiên tiến hơn. Chúng tôi luôn hi vọng một ngày nào đó sẽ hoàn thiện và áp dụng thành công mô hình Fetal Health Classification vào thực tiễn để giúp đỡ cho mọi người.

Nhóm xin chân thành cảm ơn thầy Nguyễn Vinh Tiệp đã cho những lời hướng dẫn giúp hoàn thiện bài báo này. Cảm ơn các thành viên đã luôn không ngừng cố gắng, nỗ lực từ khâu chọn đề tài, nghiên cứu cho đến thành quả cuối cùng. Chúc mừng vì đã hoàn thành đề tài với mô hình tốt hơn cả mong đợi.

- **Nguyễn Minh Duy**: Mô hình Support Vector Machines
- **Huỳnh Võ Ngọc Thanh**: Mô hình K-nearest neigbors
- **Nguyễn Trần Hoài Bão**: Mô hình Random Forest

VII. Tài liệu tham khảo

1) https://laodong.vn/xa-hoi/chuyen-gia-phan-tich-nguyen-nhan-ti-le-tu-vong-\me-va-be-o-viet-nam-con-cao-1124097.ldo

2) https://scikit-learn.org/stable/modules/generated/sklearn.svm.SVC.html

3) https://www.kaggle.com/code/prashant111/svm-classifier-tutorial#:~:text=SVMs%20are%20one%20of%20the,non%2Dprobabilistic%20binary%20linear%20classifier.

4) https://ieeexplore.ieee.org/document/9618748

5) https://machinelearningcoban.com/2017/04/09/smv/

6) https://www.mdpi.com/2673-7426/3/2/19

7) https://www.ibm.com/topics/knn#:~:text=The%20k%2Dnearest%20neighbors%20algorithm%2C%20also%20known%20as%20KNN%20or,of%20an%20individual%20data%20point.

8) https://www.researchgate.net/publication/2948052_KNN_Model-Based_Approach_in_Classification

9) https://paperswithcode.com/paper/k-nearest-neighbour-classifiers-2nd-edition

10) https://ieeexplore.ieee.org/document/9618748

11) https://www.kdnuggets.com/2020/11/most-popular-distance-metrics-knn.html

12) https://medium.com/@luigi.fiori.lf0303/distance-metrics-and-k-nearest-neighbor-knn-\1b840969c0f4

13) https://link.springer.com/chapter/10.1007/978-3-319-07773-4_19

14) https://en.wikipedia.org/wiki/Random_forest

15) https://towardsdatascience.com/the-mathematics-of-decision-trees-random-forest\-and-feature-importance-in-scikit-learn-and-\spark-f2861df67e3#:~:text=The%20sum%20of%20the%20feature's,T%20%3D%20total%20number%20of%20trees

16) https://machinelearningcoban.com/tabml_book/ch_model/random_forest.html

17) https://www.kaggle.com/code/radhikapotey/random-forest-fetal-health-classification/output

18) https://www.youtube.com/watch?v=v6VJ2RO66Ag