Báo cáo tuần 1:

Phân tích đánh giá tập dữ liệu trực quan :   
A screenshot of a computer

Description automatically generated

Chúng ta quan sát thấy tập dữ liệu có 12 cột: mô tả các cột thuộc tính như sau :

id: định danh duy nhất

gender: "Nam", "Nữ" hoặc "Khác"

age: tuổi của bệnh nhân

hypertension: 0 nếu bệnh nhân không mắc cao huyết áp, 1 nếu bệnh nhân mắc cao huyết áp

heart\_disease: 0 nếu bệnh nhân không mắc bệnh tim, 1 nếu bệnh nhân mắc bệnh tim

ever\_married: "Không" hoặc "Có"

work\_type: "Trẻ em", "Công việc chính phủ", "Chưa từng làm việc", "Tư nhân" hoặc "Tự làm chủ"

Residence\_type: "Nông thôn" hoặc "Thành thị"

avg\_glucose\_level: mức độ glucose trung bình trong máu

bmi: chỉ số khối cơ thể

smoking\_status: "Từng hút thuốc", "Chưa hút bao giờ", "Hút thuốc" hoặc "Không rõ"\*

stroke: 1 nếu bệnh nhân đã mắc đột quỵ hoặc 0 nếu không

\*Lưu ý: " Unknown " trong smoking\_status có nghĩa là thông tin không khả dụng đối với bệnh nhân này.

Đối với cột id là định danh cột này không có giá trị về đánh giá chẩn đoán sớm đột quỵ, nên em sẽ loại bỏ cột này vì thế nên sẽ còn 11 cột trong đó 10 cột là thuộc tính, và 1 cột cuối cùng stroke là phân loại người bệnh có thể mắc đột quỵ sớm hay không?

Chính vì đó nên dữ liệu của em sẽ gồm có các đặc trưng như sau: gender, age, hypertension, heart\_disease, ever\_married, work\_type, Residence\_type, avg\_glucose\_level, bmi, smoking\_status. Và 1 cột phân loại là stroke .Quy trình phân tích dữ liệu và đánh giá => lấy file knn hôm bữa chỉ bài gửi chèn vào chỗ này. Cái này t ko ghi lại.

Báo cáo tuần 2 :

Sau khi em đánh giá được dữ liệu nhận ra rằng dữ liệu chúng ta bị mất cân bằng thế nên em đã xử lý bằng các là tạo dữ liệu ảo và đồng thời cắt bớt dữ liệu để cho cân bằng dữ liệu. sau khi đào tạo mô hình bằng thuật toán knn em nhận ra rằng chỉ số phần trăm chính xác không cao cụ thể xx% (xx) này là gì tui quên rồi. mò file knn lấy ra

Báo cáo tuần 3 :

Em có tham khảo tìm hiểu một số thuật toán đào tạo mô hình phù hợp với dữ liệu mất cân bằng để có thể được sử dụng cải thiện độ chính xác. Thuật toán học máy như Support Vector Machines (SVM), Decision Trees, hoặc các thuật toán ensemble khác như Random Forest, Gradient Boosting, hoặc XGBoost. Các thuật toán phổ biến bao gồm:

1. Random Oversampling and Undersampling: Tăng cường số lượng mẫu ít để cân bằng với mẫu nhiều hoặc giảm bớt số lượng mẫu nhiều để cân bằng với mẫu ít. => phương pháp này em đã dùng trong knn và dùng thấy chỉ được xx% chính xác

Qua quá trình đánh giá em sẽ sử dụng đồng thời 2 phương pháp :  
1. Áp dụng Random Oversampling để tăng số lượng mẫu cho lớp thiểu số lên một mức độ nhất định.

2. Sau đó, sử dụng SMOTE trên dữ liệu đã được oversampling để tạo ra các mẫu tổng hợp, giúp cân bằng dữ liệu một cách tự nhiên hơn.

Đầu tiên em sẽ thêm thư viện :

A black screen with white text

Description automatically generated

Sau đó em quan sát số class 1 và class 0 :  
A screenshot of a computer program

Description automatically generated

Như hình chúng ta thấy class 0 có 4861 mẫu và class 1 là 249. Như thế cho em thấy rằng dữ liệu bị mất cân bằng quá nặng.

Tiếp theo em sẽ kiểm tra trước khi dùng oversampling để kiểm tra

A screen shot of a computer program

Description automatically generated

Sau đó em sẽ dùng oversampling : A screenshot of a computer program

Description automatically generated

Thì dữ liệu chúng ta tăng lên 9722 vì nguyên tắc của oversampling là tăng cường dữ liệu bằng cách tăng số lượng mẫu trong tập dữ liệu thiểu số (minority class) để làm cho tập dữ liệu cân bằng hơn. Khi tập dữ liệu mất cân bằng, mô hình học máy có thể có xu hướng học cách phân loại tốt hơn cho lớp chiếm đa số (majority class) và không có đủ dữ liệu để học cách phân loại đúng cho lớp thiểu số. Sau khi áp dụng Random Oversampling, X\_ros và y\_ros sẽ là tập dữ liệu mới có số lượng mẫu đã được tăng lên để cân bằng giữa các lớp. Kết quả của fit\_resample là X\_ros, là tập dữ liệu mới với các đặc trưng, và y\_ros, là tập dữ liệu mới với các nhãn tương ứng. Các thông số như random\_state được sử dụng để đảm bảo tính nhất quán của quá trình oversampling, đồng thời giúp việc xử lý dữ liệu trở nên dễ dàng tái tạo.

1. Synthetic Minority Over-sampling Technique (SMOTE): Tạo dữ liệu ảo cho lớp thiểu số để cân bằng với lớp đa số bằng cách tạo các mẫu mới dựa trên các mẫu hiện có.

Tiếp theo sẽ ứng dụng smote vào cân bằng dữ liệu :   
A screenshot of a computer code

Description automatically generated

Khi chúng ta sử dụng cả 2 phương pháp thì sẽ có nhiều điểm dữ liệu hơn. Và không bị lặp lại bởi các điểm dữ liệu sẵn có trên tập dữ liệu

1. Algorithmic Ensemble Techniques: Sử dụng các thuật toán như Random Forest, Gradient Boosting Machines hoặc XGBoost có khả năng xử lý dữ liệu mất cân bằng tốt hơn các thuật toán đơn lẻ.

Sau đó em sẽ chia data thành tỉ lệ 80% và 20%:

A computer screen shot of a black screen

Description automatically generated

Sau đó em kiểm tra các class 0 và class 1 sau khi sử dụng smote :

A screenshot of a computer program

Description automatically generated

Dựa vào kết quả trên em thấy rằng dữ liệu đã ổn không bị lệch về class nào .

Em sẽ tính toán thủ công tìm ra k hợp lý

Em sẽ khởi tạo điểm X với các thuộc tính mô tả như dưới:

A screenshot of a computer

Description automatically generated

Em sẽ sử dụng X[cols] là các đặc tính , sau đó em sử dung công thức tính eculide:

A screenshot of a computer program

Description automatically generated

Em sẽ được kết quả hỗn hợp như ảnh

Tiếp theo em sẽ sắp xếp và lấy 5 điểm gần nhất

A screenshot of a computer program

Description automatically generated

Dựa vào kết quả em thấy 1<k<=5 thì sẽ mang lại giá trị tốt ở đây em chọn k = 2.

Tiếp theo em sẽ khởi tạo mô hình sử dụng knn :

A screenshot of a computer program

Description automatically generated

Sau đó em sẽ được kết quả như sau :

A screenshot of a computer

Description automatically generated

Từ kết quả này em đánh giá được các chỉ số ở mức tốt cho mô hình knn cụ thể là :   
Precision (Chính xác):

* Precision cho lớp 0: 1.00 (100%)
* Precision cho lớp 1: 0.93 (93%)
* Precision là tỷ lệ của các trường hợp dự đoán đúng (True Positives) so với tổng số trường hợp dự đoán là positive (True Positives + False Positives). Trong trường hợp này, precision cao cho cả hai lớp, đặc biệt là cho lớp 0.

Recall (Recall hoặc Sensitivity - Tỉ lệ nhận biết):

* Recall cho lớp 0: 0.93 (93%)
* Recall cho lớp 1: 1.00 (100%)
* Recall là tỷ lệ của các trường hợp dự đoán đúng (True Positives) so với tổng số trường hợp thực tế là positive (True Positives + False Negatives). Recall cao cho cả hai lớp, đặc biệt là cho lớp 1.

F1-score:

* F1-score cho lớp 0: 0.96 (96%)
* F1-score cho lớp 1: 0.97 (97%)
* F1-score là sự kết hợp giữa precision và recall. Nó là một số đo tổng hợp của hiệu suất của mô hình. F1-score càng cao, mô hình càng tốt.

Support:

* Support cho lớp 0: 975
* Support cho lớp 1: 970
* Support là số lượng thực tế của mỗi lớp trong tập kiểm tra.

Accuracy (Độ chính xác):

* Accuracy: 96.50%
* Accuracy là tỷ lệ của các dự đoán đúng (True Positives + True Negatives) so với tổng số mẫu.

Ma trận nhầm lẫn(confusion\_matrix):

True Negative (TN): 907

False Positive (FP): 68

False Negative (FN): 0

True Positive (TP): 970

Ma trận nhầm lẫn cung cấp cái nhìn cụ thể về cách mô hình phân loại từng loại mẫu.

Tổng quan, em đánh giá mô hình của bạn có hiệu suất tốt với độ chính xác lên đến 96.50%. Hiệu suất cao của cả hai lớp được thể hiện qua các chỉ số metric precision, recall, và F1-score.

Ngoài knn em có thử 1 số mô hình khác để xem hiệu quả giữa các mô hình với tập dữ liệu của em.

Mô hình random forest rừng ngẫu nhiên :

A computer screen shot of a program

Description automatically generated

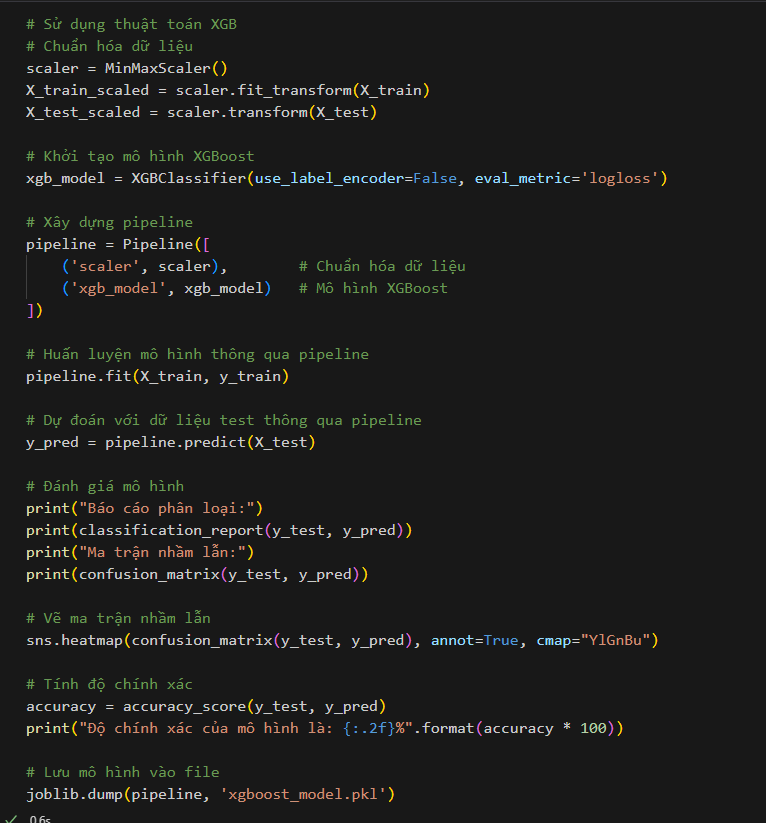
Em có được kết quả khá bất ngờ :

A screenshot of a computer

Description automatically generated

Với kết quả trên thì mô hình đúng 99,5% với tất cả các chỉ số ở lớp 0 và lớp 1 đều hoàn hảo. có thể nói random forest là thuật toán học máy đối với các trường hợp dữ liệu mất cân bằng tốt nhất khi so với các thuật toán khác. Chúng ta có thể sử dụng modal này để phân loại.

Thuật toán XGBoost :



Em có kết quả thu được là :

A screenshot of a computer

Description automatically generated

Từ kết quả như hình trên em thấy được rằng mô hình có độ chính xác khá cao cụ thể là 97% với các chỉ số tốt. đối với lớp 1 (bị đột quỵ) phân loại được chính xác hoàn toàn 970 mẫu. đối với trường hợp không bị đột quỵ thì sẽ bị phân loại nhầm sang bị đột quỵ 43 mẫu. modal ta có thể phải tinh chỉnh lại rồi mới sử dụng được.

Thuật toán gradient boost:

A screen shot of a computer program

Description automatically generated

Em có kết quả :

A screenshot of a computer

Description automatically generated

Dường như thuật toán này không mạnh về lĩnh vực học máy. Quan sát thấy được các chỉ số khá kém. ở cả 2 class 0 và 1. Nên modal này không thể sử dụng.

Cây quyết định :

A screen shot of a computer program

Description automatically generated

Em có được kết quả :

A screenshot of a computer

Description automatically generated

Nhìn thoáng qua em thấy được rằng: mô hình này ở mức tốt. chỉ sau random forest. Ở các chỉ số đều tốt với các chỉ số presion, recall, f1 và support đều ổn. và có được tỉ lệ chính xác cao cụ thể là 98%. Qua 2 mô hình em thấy rằng các dữ liệu dạng phân loại thì em sử dụng Các cây phân loại tốt hơn các thuật toán học máy khác

A computer screen shot of a program code

Description automatically generated

Em có kết quả như sau :

A screenshot of a computer

Description automatically generated

Khi nhìn vào kết quả em thấy mô hình đưa ra chỉ số đánh giá tệ dường như mô hình không thể sử dụng được vì các chỉ số thấp và phân loại sai rất nhiều.

Sau khi em đã tìm hiểu và được học, sử dụng một trong những phương pháp trên để xử lý vấn đề mất cân bằng về tập dữ liệu của em, việc đánh giá mô hình trở nên quan trọng hơn bao giờ hết. Em cần kiểm tra kỹ thuật đánh giá mô hình bằng nhiều thuật toán, bao gồm chia bộ dữ liệu thành tập huấn luyện và tập kiểm tra, đánh giá bằng các độ đo chính xác, độ phủ, độ chính xác dự đoán, và F1-score.

Nếu chỉ số chính xác của mô hình chỉ đạt khoảng 96.5% bằng thuật toán KNN thì không thể đảm bảo thế nên em phải cân nhắc điều chỉnh thêm các tham số của mô hình và sử dụng thuật toán phù hợp, tinh chỉnh dữ liệu đầu vào hoặc thử nghiệm với các phương pháp mô hình hóa khác nhau để cải thiện hiệu suất.

Báo cáo tuần 4 :   
Em sẽ nói về thuật toán Ensemble là một phương pháp mà nhiều mô hình học máy được kết hợp lại để tạo ra một mô hình mạnh hơn kế thừa từ thuật toán KNN đã đào tạo từ tuần 2, bằng cách kết hợp dự đoán từ nhiều mô hình nhỏ hơn. Các thuật toán Ensemble rất hiệu quả trong việc xử lý dữ liệu mất cân bằng bởi vì chúng có khả năng xử lý các tình huống phức tạp hơn và tạo ra dự đoán chính xác hơn.

Dưới đây là em đã tổng hợp một số phương pháp Ensemble phổ biến mà em có thể sử dụng để xử lý dữ liệu mất cân bằng trong đề tài của nhóm em:

1. Random Forest (RF): RF là một thuật toán học máy phổ biến được xây dựng dựa trên ý tưởng của "rừng quyết định". Nó tạo ra nhiều cây quyết định và sau đó kết hợp các dự đoán từ các cây này để tạo ra dự đoán cuối cùng. RF thường hoạt động tốt trên dữ liệu mất cân bằng vì nó có khả năng xử lý nhiễu và không quan trọng việc cân bằng dữ liệu.
2. Gradient Boosting Machines (GBM): GBM là một thuật toán Ensemble mạnh mẽ khác, nó tạo ra một loạt các mô hình yếu và kết hợp chúng để tạo ra một mô hình mạnh mẽ hơn. GBM cũng có khả năng xử lý dữ liệu mất cân bằng tốt, nhờ vào khả năng của nó học từ các lỗi của mô hình trước đó và tập trung vào việc cải thiện các điểm yếu đó.
3. XGBoost (Extreme Gradient Boosting): XGBoost là một biến thể của thuật toán Gradient Boosting, được phát triển để tối ưu hóa hiệu suất và tốc độ của GBM. Nó sử dụng các kỹ thuật như pruning để tránh overfitting và có thể xử lý dữ liệu mất cân bằng hiệu quả hơn nhiều so với các thuật toán khác.

* Ở cả 3 loại mô hình kết hợp em kể ở trên và so với kết quả tuyệt vời của random forest thì em quyết định sẽ chọn random forest làm thuật toán thứ 2 để em đào tạo mô hình.

Em có đoạn mã đào tạo kết hợp cả 2 thuật toán knn và random forest sử dụng VotingClassifier như sau:

A screen shot of a computer

Description automatically generated

Và em có kết quả như sau :

A screenshot of a computer

Description automatically generated

Nhìn vào các chỉ số rất tuyệt vời với tỉ lệ đúng là 99.74% dường như đây là mô hình rất tốt có thể sử dụng được trong phân loại.

Em có đoạn mã đào tạo kết hợp cả 2 thuật toán knn và random forest sử dụng StackingClassifier như sau:

A screen shot of a computer

Description automatically generated

Kết quả đánh giá mô hình :

A screenshot of a computer

Description automatically generated

Đối với đánh giá như này em không có gì để nhận xét vì tỉ lệ mô hình rất tốt. độ chính xác là 100% với các chỉ số giữa 2 lớp là đúng chính xác 100%.

Tiếp theo em sẽ đưa ra đánh giá xem thử 2 mô hình mà em kết hợp knn và random forest lại với nhau có bị overfiting hay không?

A graph with a red line and green line

Description automatically generated

Thì dựa vào hình ảnh em đánh giá : Cả hai đường đều tăng và hội tụ, cho thấy mô hình có thể đã học tốt từ dữ liệu đào tạo và có khả năng tổng quát hóa tốt trên dữ liệu chưa biết. Do đó, có thể nói rằng mô hình này không bị quá khớp. Khi đó 2 mô hình trên em có thể sử dụng.

Tiếp theo em sẽ kiểm thử mô hình thực tế :

A computer screen shot of a program code

Description automatically generated

Kết quả dự đoán :

A black background with white text

Description automatically generated

Cho thấy mô hình tốt và dự đoán đúng

Báo cáo tuần 5:

Phân tích nghiệp vụ khảo sát bài toán, UML vẽ usecase, actor, database, lên thiết kế UI/UX ứng dụng của ai làm (cái này quan trọng nên góp ý 3 người vào đánh giá. Vì sai từ phân tích là sai toàn bộ bài làm)…………….

Báo cáo tuần 6:

Viết chương trình lựa chọn công nghệ (java, python, C#) của ai làm ……………

Báo cáo tuần 7:

Đưa mô hình thuật toán tương tứng với các thuộc tính vào chương trình (có thể viết api python trả về request modal)…………………

Báo cáo tuần 8:

Kiểm thử lỗi và chạy thử nghiệm……………………..

------------------------------------Hết-------------------------------------

A diagram of positive and negative

Description automatically generated