

MỤC LỤC

[**1. Giới thiệu 3**](#_gwja2gewjajo)

[**2. Các kiến thức liên quan và những phương pháp chính 4**](#_ovuo34rue1v)

[2.1. EDA 4](#_uzz63ap2iior)

[2.2. Tiền xử lý dữ liệu 5](#_p21xqouj6d9d)

[2.3. Model 7](#_d0zw1muj4q27)

[2.4. Các phương pháp đánh giá mô hình: 10](#_iav93birn3uj)

[**3. Nghiên cứu vấn đề 11**](#_you6jh8el49r)

[3.1. Xác định vấn đề 11](#_7hw5r5sld436)

[3.2. Khám phá dữ liệu (EDA) 12](#_nmi9vnedygjp)

[3.3. Tiền xử lý dữ liệu 15](#_r0sijilfj8rx)

[**4. Thử nghiệm và kết quả 18**](#_e2iigcurj50i)

[4.1. Môi trường thiết lập 18](#_rfl9d8ir48ws)

[4.2. Đề xuất chiến lược 19](#_sohun1if423c)

[4.3. Kết quả dữ liệu huấn luyện (tập train) 20](#_am1fbroa3yga)

[4.4. Kết quả dữ liệu dự báo (tập test) 23](#_t5qvi34wqqh9)

[**5. Kết luận 25**](#_815bd789fvzc)

[**TÀI LIỆU THAM KHẢO 27**](#_nx0h7psohjy4)

# 

DANH SÁCH THÀNH VIÊN

| **Họ và tên** | **MSSV** | **Phần trăm đóng góp** |
| --- | --- | --- |
| Nguyễn Khánh Toàn | 31221024692 | 100% |
| Hứa Tuấn Vĩ | 31221020248 | 100% |
| Trần Phương Nghi | 31221020434 | 100% |
| Phạm Mai Chi | 31221023956 | 100% |
| Trần Gia Huy | 31221023695 | 100% |

# 

# 

# 1. Giới thiệu

“Colic” là một thuật ngữ rộng được sử dụng để mô tả rối loạn tiêu hóa ở ngựa. Nó có thể do rất nhiều nguyên nhân khác nhau gây ra và thường bắt đầu đột ngột. Vì không dễ dàng nhận ra cho nên tính nghiêm trọng của rối loạn tiêu hóa ở ngựa thường tăng nhanh chóng, phải can thiệp phẫu thuật thậm chí khiến ngựa tử vong.

Theo Hội đồng Ngựa Hoa Kỳ, “colic” là nguyên nhân dẫn đến tử vong của khoảng 64.000 con ngựa tại Hoa Kỳ mỗi năm.



***Hình 1.Horse Colic***

Trong bài nghiên cứu này, nhóm chúng tôi xây dựng một mô hình phân loại để hỗ trợ nhân viên chăm sóc ngựa và các bác sĩ thú y trong việc xác định các trường hợp ngựa mắc bệnh có nguy cơ tử vong hay không, nhằm giúp các bác sĩ đưa ra quyết định điều trị một cách đúng đắn, từ đó giúp doanh nghiệp tiết kiệm chi phí cũng như có các kế hoạch giảm thiểu tổn thất tốt nhất.

Mô hình dự đoán sử dụng bộ dữ liệu Horse Colic Dataset với các phương pháp phân loại sau đây: (1) kNN, (2) naive Bayes, (3) SVM, (4) decision tree, (5) random forest, (6) AdaBoost, (7) gradient Boosting, (8) linear discriminant analysis, (9) multi-layer perceptron và (10) logistic regression. Sử dụng các phương pháp đánh giá như accuracy, precision, recall, F1-Score và confusion matrix để so sánh hiệu suất của các phương pháp phân loại. Bài báo cáo được chia thành 5 phần, bao gồm:

**Phần 1:** Giới thiệu

**Phần 2:** Các kiến thức liên quan và những phương pháp chính

**Phần 3:** Nghiên cứu vấn đề

**Phần 4:** Thử nghiệm và kết quả

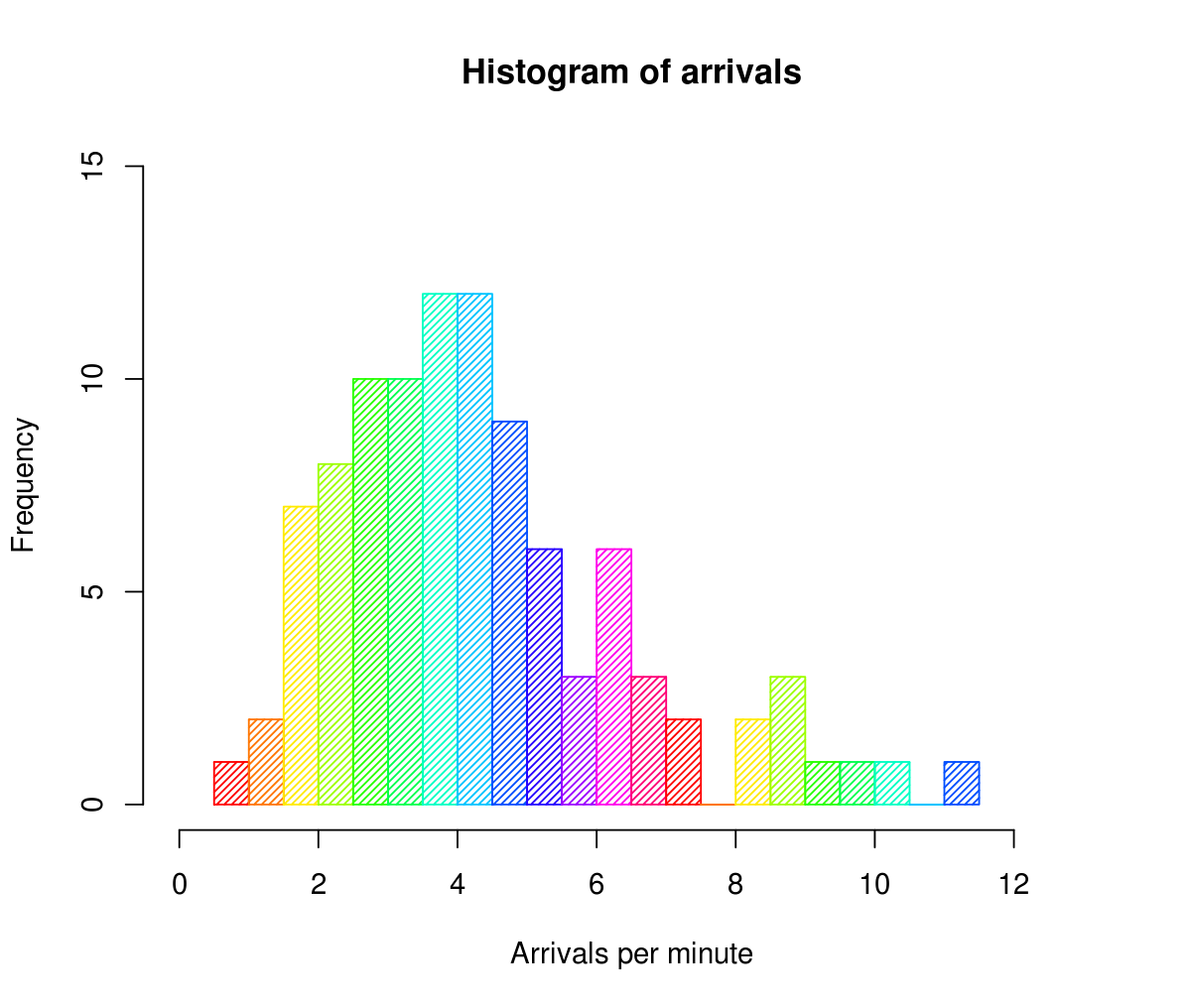
**Phần 5:** Kết luận

# 2. Các kiến thức liên quan và những phương pháp chính

## 2.1. EDA

**Countplot:** Count Plot là một loại biểu đồ thống kê dùng để đếm số lượng các điểm dữ liệu thuộc vào từng nhóm và hiển thị chúng dưới dạng các cột dọc. Đây thường là biểu đồ phổ biến để so sánh tần suất xuất hiện của các nhóm trong dữ liệu.

**Histogram:** Histogram là một biểu đồ thống kê biểu diễn phân phối tần suất của dữ liệu số bằng cách chia dữ liệu thành các khoảng và biểu diễn số lượng quan sát trong mỗi khoảng bằng cách sử dụng các cột.



***Hình 2.1. Histogram***

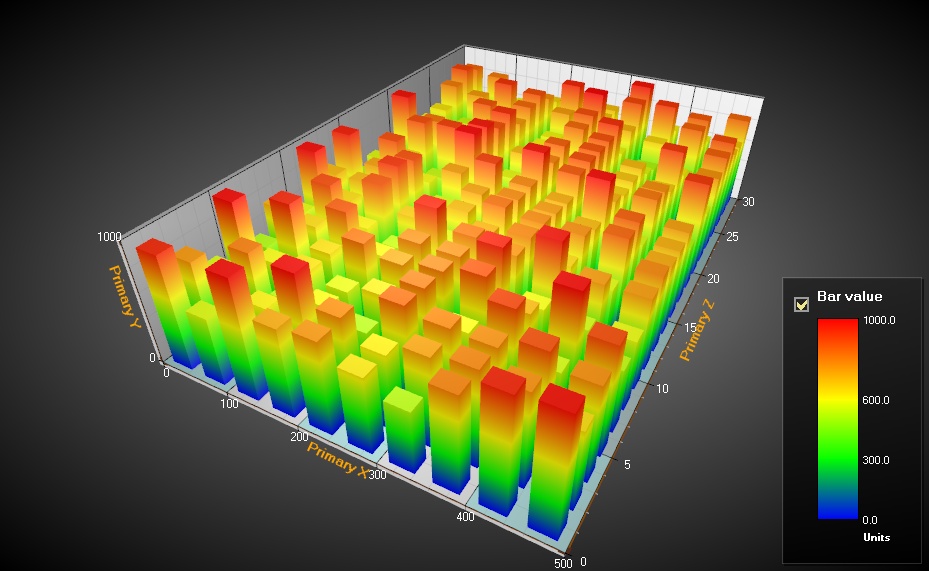
**Density Plots:** Density Plots là biểu đồ biểu diễn dạng mật độ của phân phối xác suất của một biến số dữ liệu. Nó thường được sử dụng để xác định hình dạng của phân phối dữ liệu và so sánh phân phối của nhiều nhóm.

**Box và whisker plots:** Box và whisker plots là một biểu đồ thống kê biểu diễn phân phối của dữ liệu số thông qua việc minh họa các giá trị trung bình, phần tư và giá trị ngoại lệ của dữ liệu dưới dạng hộp và các dãy đường (whisker).

**Categorical plot:** categorical plot là một dạng biểu đồ thống kê được sử dụng để thể hiện mối quan hệ giữa các biến phân loại

**Correlogram:** Correlogram là biểu đồ thống kê biểu diễn mối quan hệ tuyến tính giữa các biến số dữ liệu thông qua biểu diễn ma trận các hệ số tương quan.

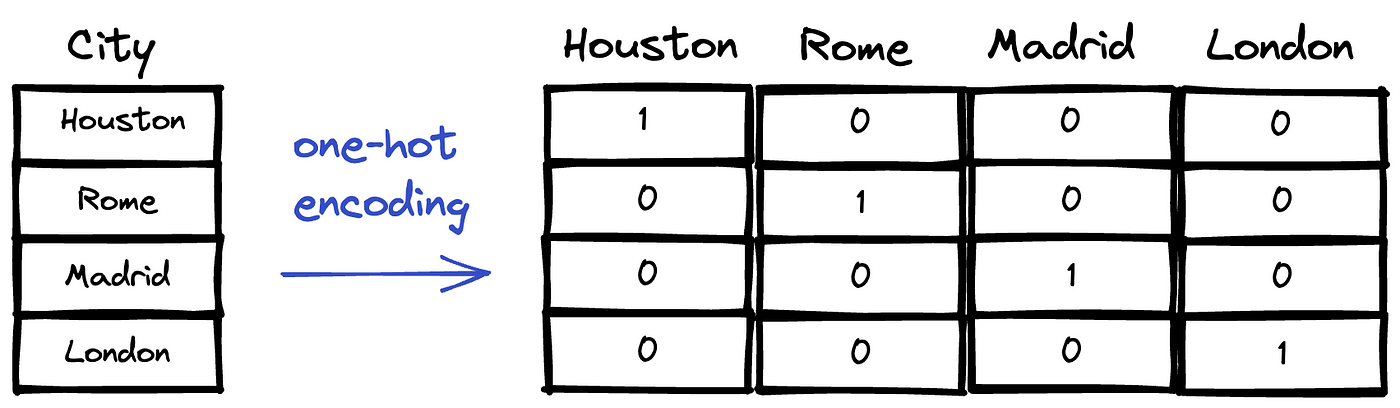
**Heatmap:** Heatmap là biểu đồ thống kê biểu diễn dữ liệu dưới dạng một bảng màu, trong đó giá trị của mỗi ô sẽ được biểu diễn bằng một màu sắc phụ thuộc vào giá trị của nó. Heatmap thường được sử dụng để minh họa mức độ tương quan giữa các biến số trong một tập dữ liệu lớn.

****

***Hình 2.2. Heat map***

## 2.2. Tiền xử lý dữ liệu

**One Hot Encoder:** One Hot Encoder là một phương pháp biến đổi dữ liệu phân loại thành dạng số liệu, cụ thể là các biến dummies để có thể sử dụng trong các mô hình máy học.

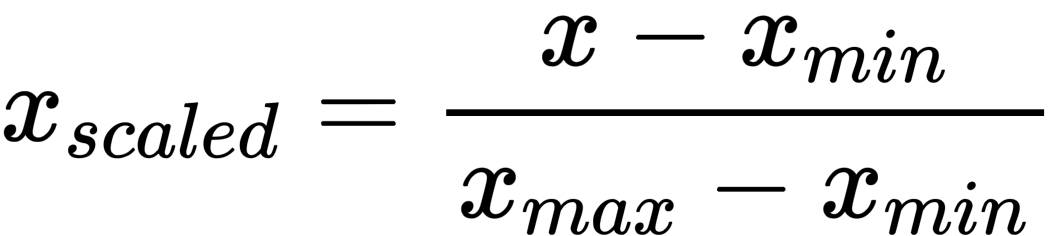


***Hình 2.3. One Hot Encoder***

**Label Encoder:** Label Encoder là phương pháp chuyển đổi các biến phân loại thành các số nguyên tăng dần, thường được sử dụng cho các biến mục tiêu trong quá trình huấn luyện mô hình.

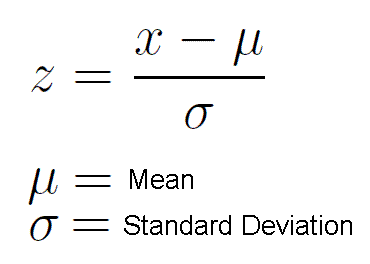
**Ordinal Encoder:** Ordinal Encoder cũng chuyển đổi các biến phân loại thành dạng số nguyên, nhưng có thể xác định thứ tự giữa các giá trị của biến phân loại.

**Min Max Scaler:** Min Max Scaler là phương pháp thao tác để chuẩn hóa dữ liệu về một phạm vi đặc biệt, thường là từ 0 đến 1, để đồng nhất dữ liệu đầu vào cho mô hình máy học.



***Hình 2.4. Min Max Scaler***

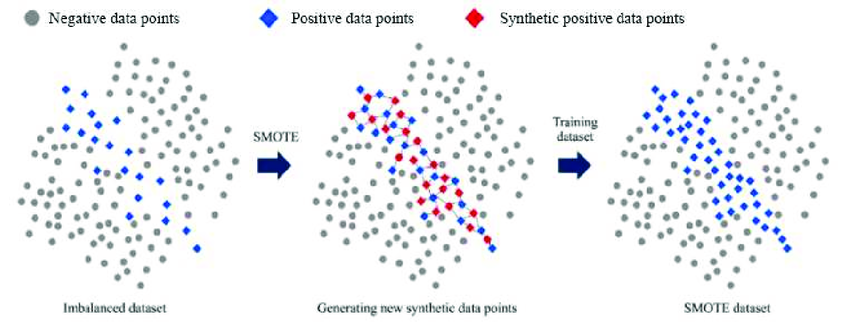
**Standard Scaler:** Standard Scaler cũng là một phương pháp chuẩn hóa dữ liệu, nhưng sử dụng phương sai và độ lệch chuẩn để đưa dữ liệu về một trung tâm và độ lệch chuẩn nhất định.



***Hình 2.5. Standard Scaler***

**Pipeline:** Pipeline là một công cụ quản lý chuỗi các bước xử lý dữ liệu và huấn luyện mô hình trong máy học, giúp tự động hóa quá trình xử lý dữ liệu và đào tạo mô hình.

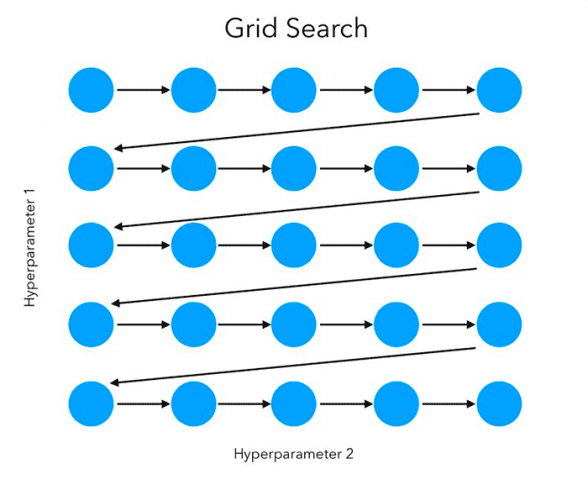
**SMOTE:** SMOTE (Synthetic Minority Over-sampling Technique) là một kỹ thuật tăng cường dữ liệu được sử dụng trong bài toán xử lý mẫu không cân bằng. Kỹ thuật này tạo ra các mẫu dữ liệu nhân tạo cho các lớp thiểu số bằng cách kết hợp các mẫu hiện có để tạo ra các mẫu mới có đặc tính tương tự nhưng khác nhau về một số chi tiết để cân bằng dữ liệu. SMOTE giúp cải thiện tính cân bằng của dữ liệu và từ đó cải thiện khả năng dự đoán cho các lớp thiểu số trong các bài toán phân loại.



***Hình 2.6. Synthetic Minority Over-sampling Technique***

## 2.3. Model

**GridSearchCV:** GridSearchCV là một kỹ thuật tìm kiếm siêu tham số (hyperparameter) tốt nhất cho mô hình bằng cách duyệt qua tất cả các tổ hợp có thể của siêu tham số và đánh giá chúng thông qua phương pháp cross-validation.

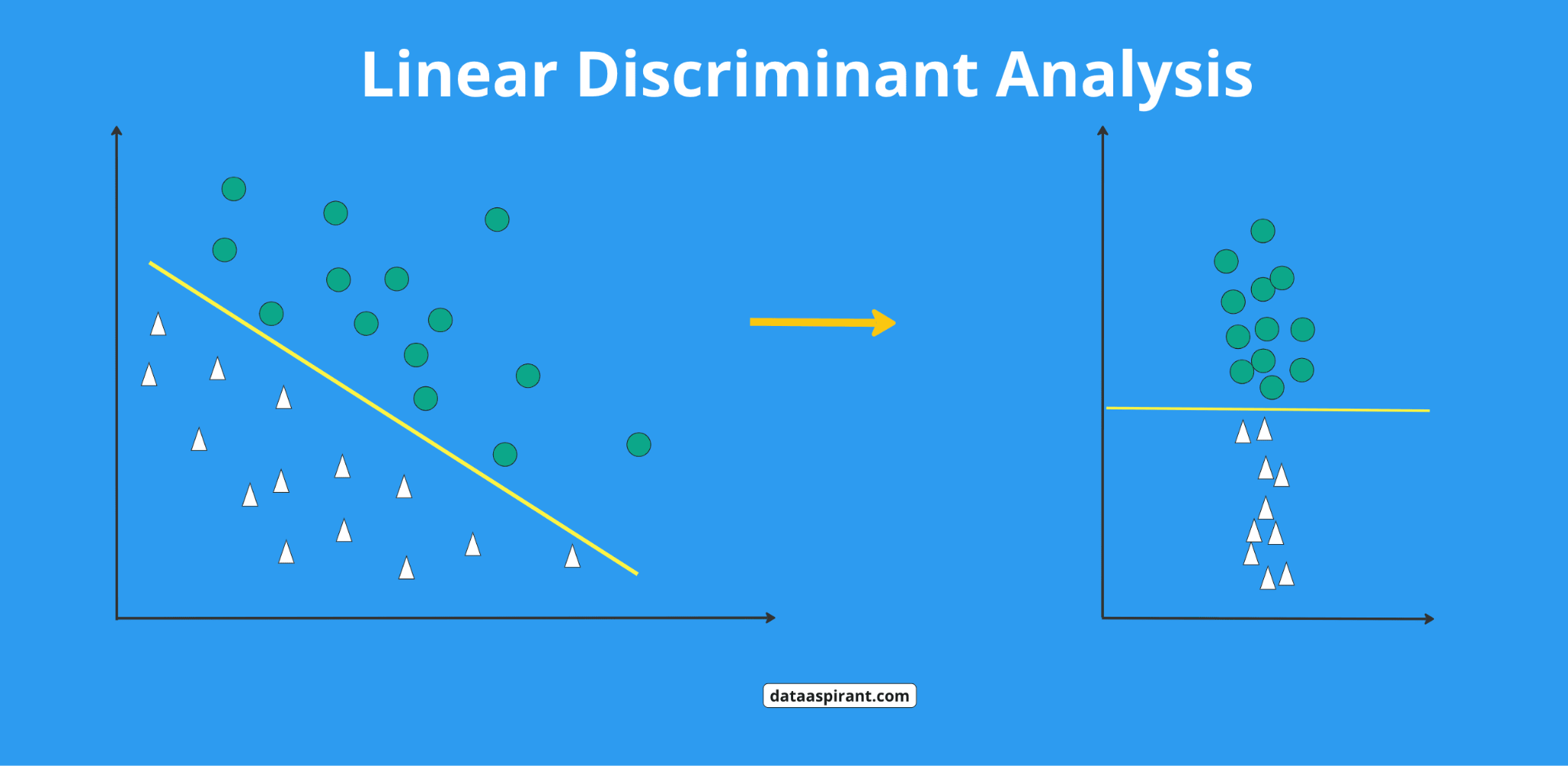


***Hình 2.7. GridSearchCV***

**K-Fold:** K-Fold là một phương pháp chia dữ liệu thành các tập con (folds) để sử dụng trong quá trình cross-validation, giúp đánh giá hiệu suất của mô hình một cách chính xác hơn.

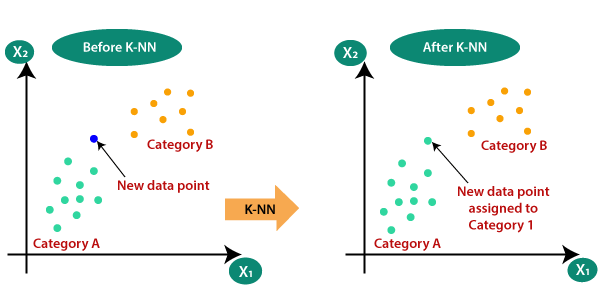
**DecisionTreeClassifier:** Decision Tree là một mô hình học có giám sát dựa trên cây quyết định, nó đưa ra dự đoán bằng cách tạo ra một chuỗi các quyết định dựa trên các đặc trưng của dữ liệu đầu vào.

**Linear Discriminant Analysis:** Linear Discriminant Analysis là một phương pháp giảm chiều dữ liệu và phân loại dữ liệu mục tiêu thành các lớp khác nhau, với mục tiêu là tối ưu hóa sự phân biệt giữa các lớp đó.



***Hình 2.8. Linear Discriminant Analysis***

**K-Neighbors Classifier:** K-Neighbors là một mô hình phân loại dựa trên nguyên lý gần nhất hạn chế, sử dụng khoảng cách từ điểm dữ liệu đến các điểm láng giềng để dự đoán lớp của mẫu dữ liệu đó.

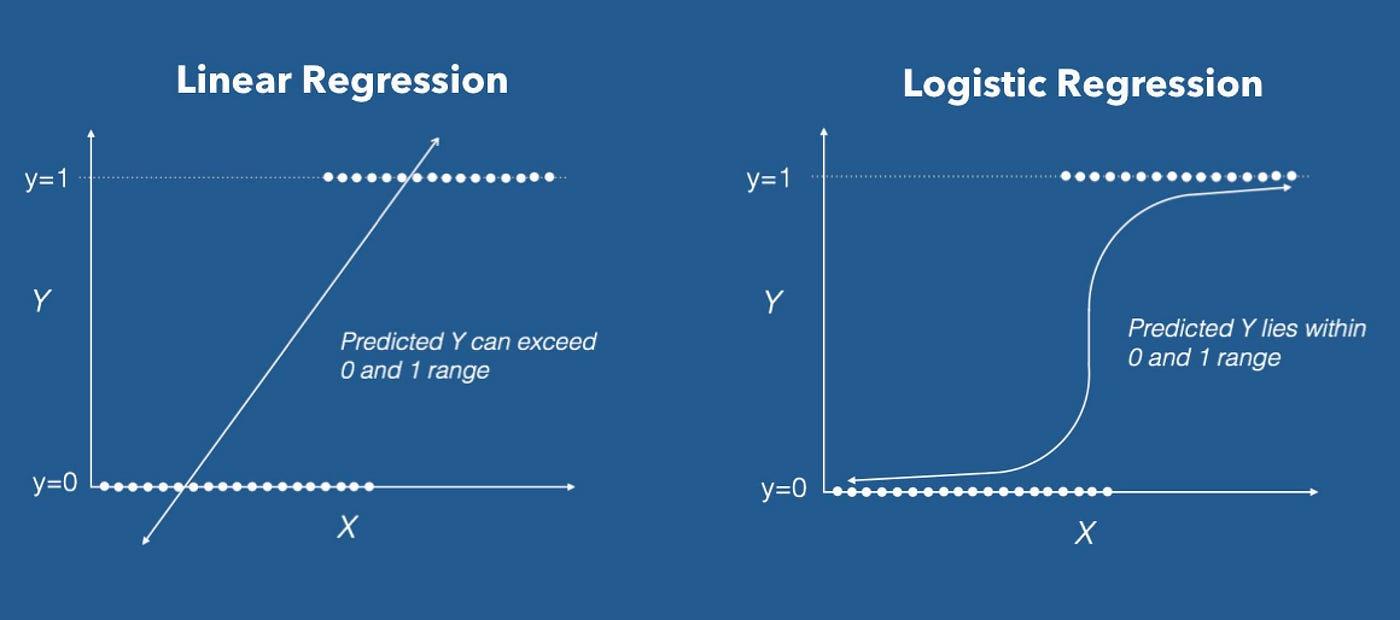


***Hình 2.9. K-Neighbors Classifier***

**SVC:** SVC (Support Vector Classifier) là một mô hình phân loại dựa trên máy vector hỗ trợ, cố gắng tạo ra ranh giới phân chia tối ưu giữa các lớp trong không gian đa chiều.

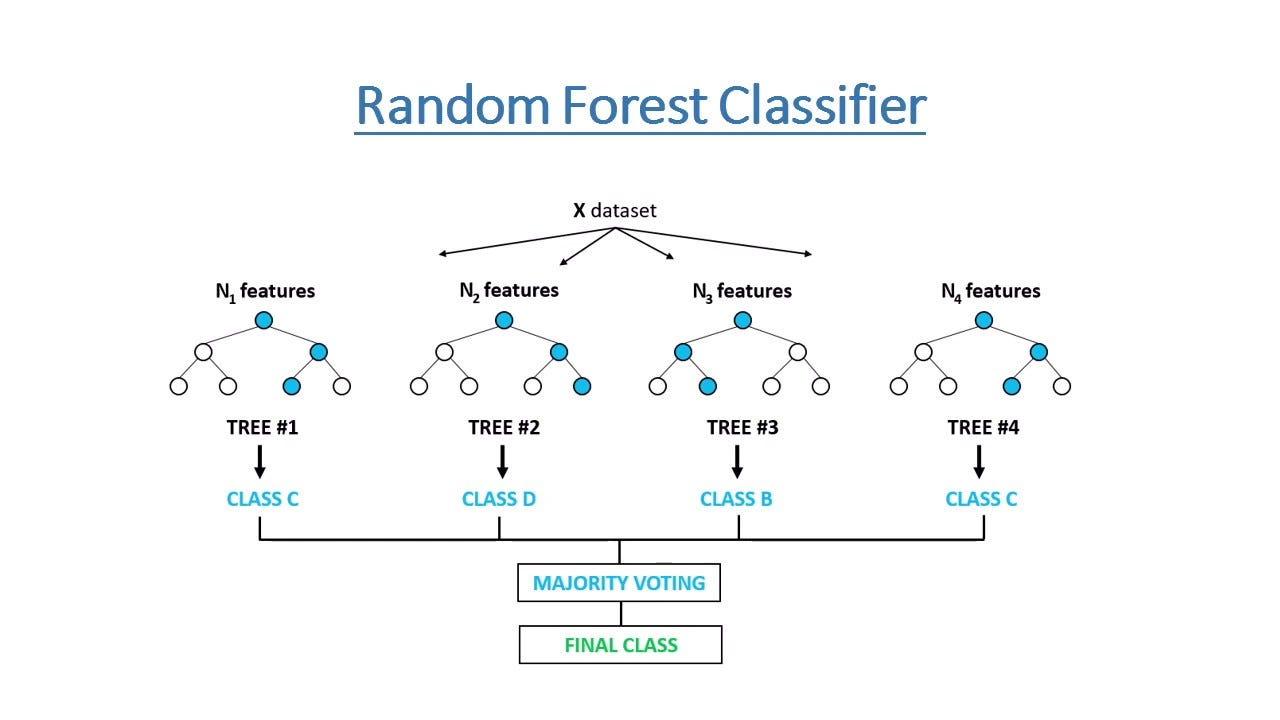
**Gaussian NB:** Gaussian NB là một mô hình phân loại sử dụng giả sử giảm đơn giản rằng phân phối của các đặc trưng đầu vào là phân phối Gaussian, thường được sử dụng trong việc phân loại văn bản và dữ liệu có đặc trưng liên tục.

**Logistic Regression:** Logistic Regression là một mô hình phân loại dựa trên hàm sigmoid, thường được sử dụng để dự đoán xác suất của một sự kiện xảy ra dựa trên các biến đầu vào.



***Hình 2.10. Logistic Regression***

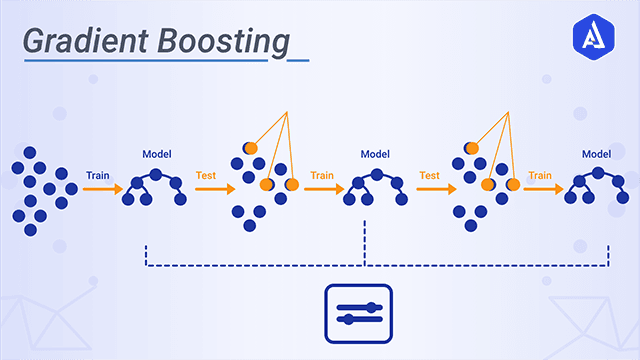
**RandomForestClassifier:** Random Forest là một mô hình phân loại dựa trên nguyên lý rừng ngẫu nhiên, kết hợp nhiều cây quyết định để tạo ra dự đoán cuối cùng.



***Hình 2.11. Random Forest***

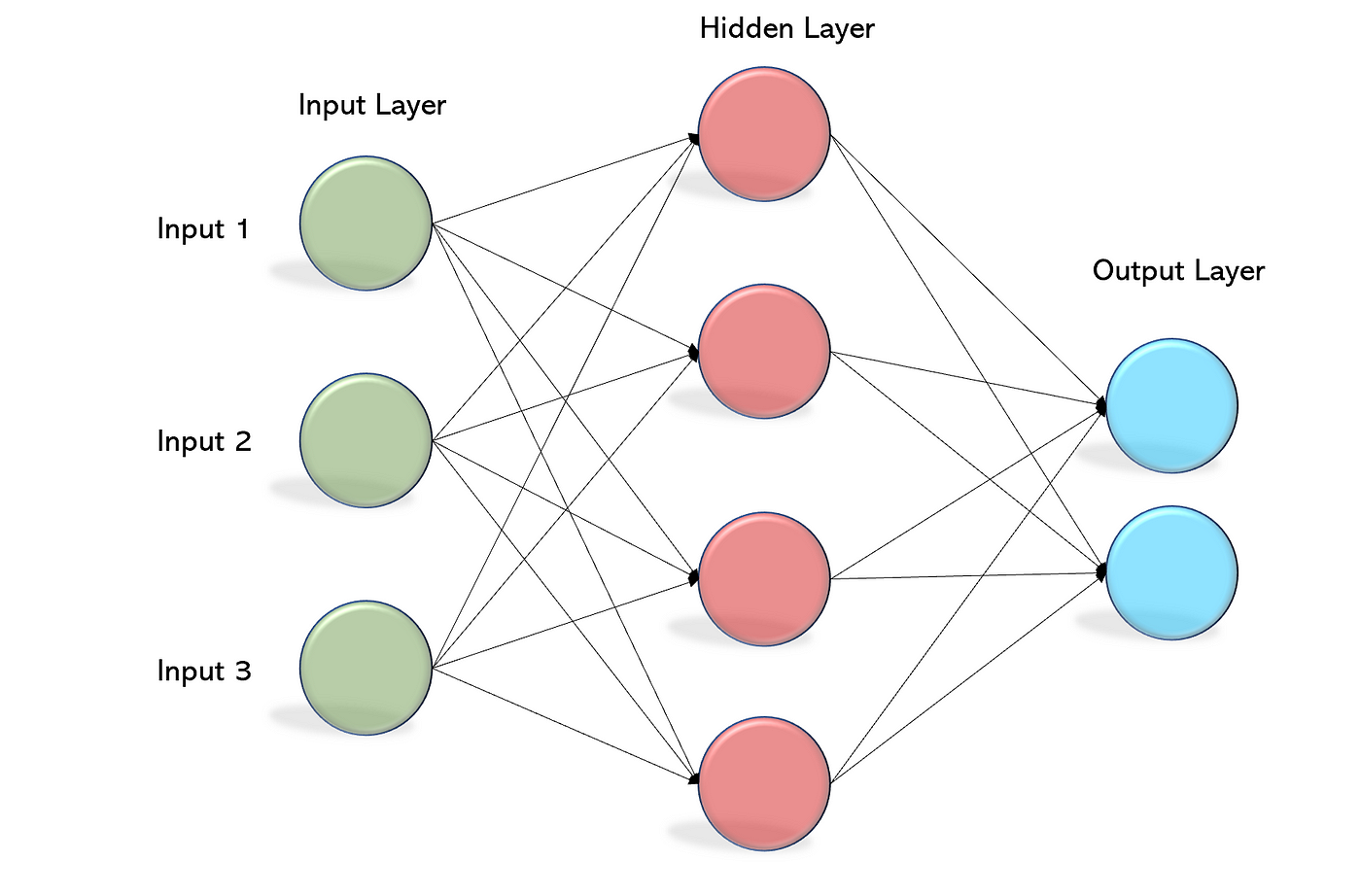
**AdaBoostClassifier:** Ada Boost là một mô hình phân loại thuộc loại "boosting" trong đó các mô hình yếu tuần tự được tạo ra và kết hợp để tạo ra một mô hình mạnh hơn.

**GradientBoostingClassifier:** Gradient Boosting cũng thuộc loại "boosting" nhưng là phương pháp tối ưu hóa từng bước, tập trung vào việc cải thiện bước dự đoán tiếp theo dựa trên sai số của bước trước đó.



***Hình 2.12. Gradient Boosting***

**MLPClassifier:** MLP là một mô hình mạng nơ-ron nhiều lớp, thường được sử dụng trong bài toán phân loại, kỹ thuật này cho phép mô hình học các quy luật phức tạp từ dữ liệu.



***Hình 2.13. MLP***

## 2.4. Các phương pháp đánh giá mô hình:

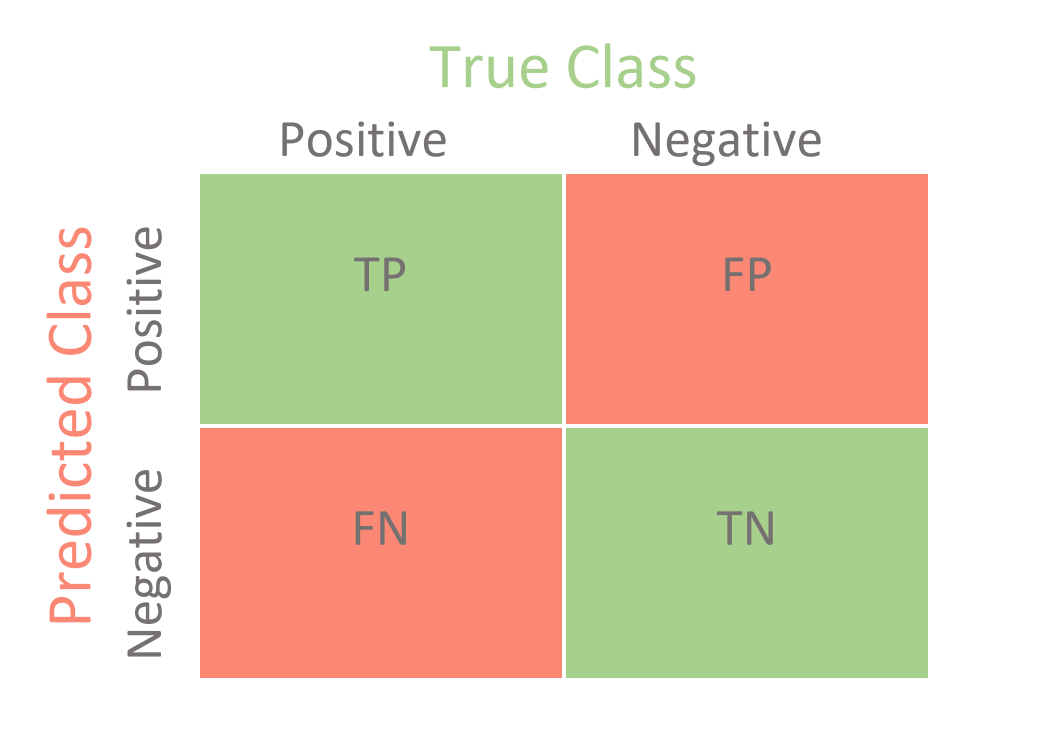
**Accuracy (Độ chính xác):** Tỷ lệ giữa số lần dự đoán đúng và tổng số dự đoán. Đây là một phép đo cơ bản của khả năng dự đoán chính xác của mô hình.

**Precision (Độ chính xác):** Tỷ lệ giữa số lần dự đoán đúng positive và tổng số positive dự đoán. Đây là mức độ chính xác của dự đoán positive.

**Recall (Tỷ lệ nhớ):** Tỷ lệ giữa số lần dự đoán đúng positive và tổng số true positive trên thực tế. Nó đánh giá khả năng của mô hình trong việc phát hiện tất cả các trường hợp positive.

**F1-score:** Trung bình điều hòa của Precision và Recall, cung cấp một cách tổng quan về hiệu suất của mô hình.

**Confusion matrix (Ma trận nhầm lẫn):** là một bảng dùng để mô tả hiệu suất của một mô hình phân loại trên một tập dữ liệu. Ma trận này bao gồm các thông tin về số lượng các điểm dữ liệu được phân loại đúng và sai cho từng lớp.

****

***Hình 2.14. Confusion Matrix***

# 3. Nghiên cứu vấn đề

Trong phần này, chúng tôi sẽ xác định vấn đề mà chúng tôi muốn giải quyết và đưa ra tổng quan ngắn gọn về kết quả phân tích khám phá dữ liệu (EDA). Dựa trên kết quả EDA, chúng tôi tiến hành đưa ra chiến lược tiền xử lý dữ liệu và các thuật toán để đánh giá. Các chi tiết của mô hình sẽ được thảo luận trong phần tiếp theo.

## 3.1. Xác định vấn đề

Tập dữ liệu “Horse Colic Dataset” được sử dụng trong bài nghiên cứu này được đặt tại kho lưu trữ máy học của UCI và được tạo bởi M. McLeish và M. Cecile (Khoa Khoa học Máy tính, Đại học Guelph, Ontario, Canada). Bộ dữ liệu này bao gồm 300 bản ghi ghi chép hồ sơ lâm sàng của từng con ngựa bị đau bụng và đã được các bác sĩ thú y đánh giá. Mặc dù kích thước của tập dữ liệu không lớn, nhưng mỗi bản ghi có tổng cộng 28 thuộc tính giúp tập dữ liệu đủ mạnh cho quá trình xây dựng mô hình.

Dữ liệu đầu ra trong quá trình tạo mô hình của chúng tôi gồm 2 thuộc tính như sau: cái chết là kết quả có thể xảy ra; và can thiệp phẫu thuật là giải pháp cần thiết để khắc phục vấn đề, cho biết liệu mạng sống của con ngựa có thể được cứu bằng phẫu thuật hay không, thông qua kết quả tổn thương do phẫu thuật - một phát hiện hồi cứu, được thực hiện nhờ phẫu thuật hoặc qua khám nghiệm tử thi.

Mục tiêu chính của bài nghiên cứu sẽ là tạo ra một mô hình có thể dự đoán kết quả với độ chính xác cao thông qua các công cụ học máy để điều tra hai yếu tố quan trọng tạo nên sự thành công của các ứng dụng học máy (scikit-learn), đó là chỉ số hiệu suất và tối ưu hóa. Thông qua bộ dữ liệu Horse Colic Dataset sẽ hỗ trợ được nhân viên cơ sở nuôi ngựa cũng như bác sĩ thú y có thể xác định được những trường hợp ngựa có nguy cơ tử vong cao cùng với các trường hợp can thiệp phẫu thuật có thể là giải pháp tối ưu. Từ đó, giúp dễ dàng xác thực được những phán đoán của họ để điều trị chính xác, tích cực và nhanh chóng hơn tình trạng ngựa bị đau bụng.

## 3.2. Khám phá dữ liệu (EDA)

**Giới thiệu về bộ dữ liệu**

Dạng dữ liệu: Dữ liệu này được thu thập từ 300 trường hợp bệnh lý của ngựa. Bao gồm 3 file chính:

* horse-colic.names
* horse-colic.data
* horse-colic.csv

Mục tiêu chính: Dự đoán kết quả cuối cùng của các trường hợp viêm ruột của ngựa

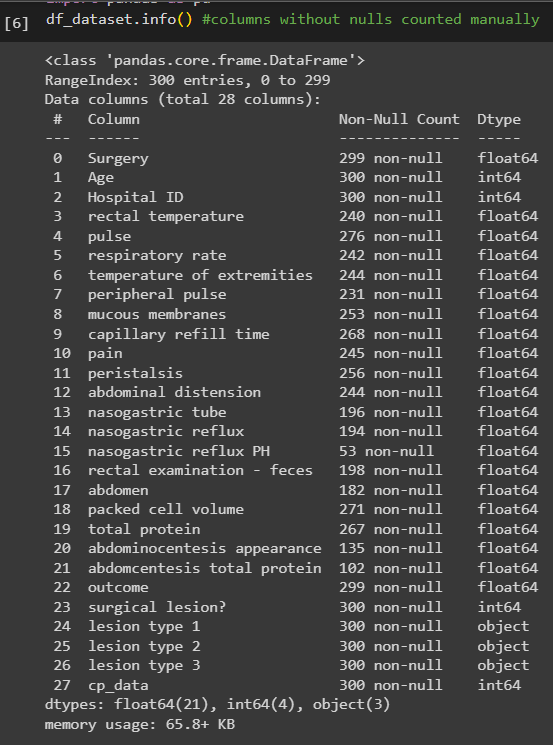
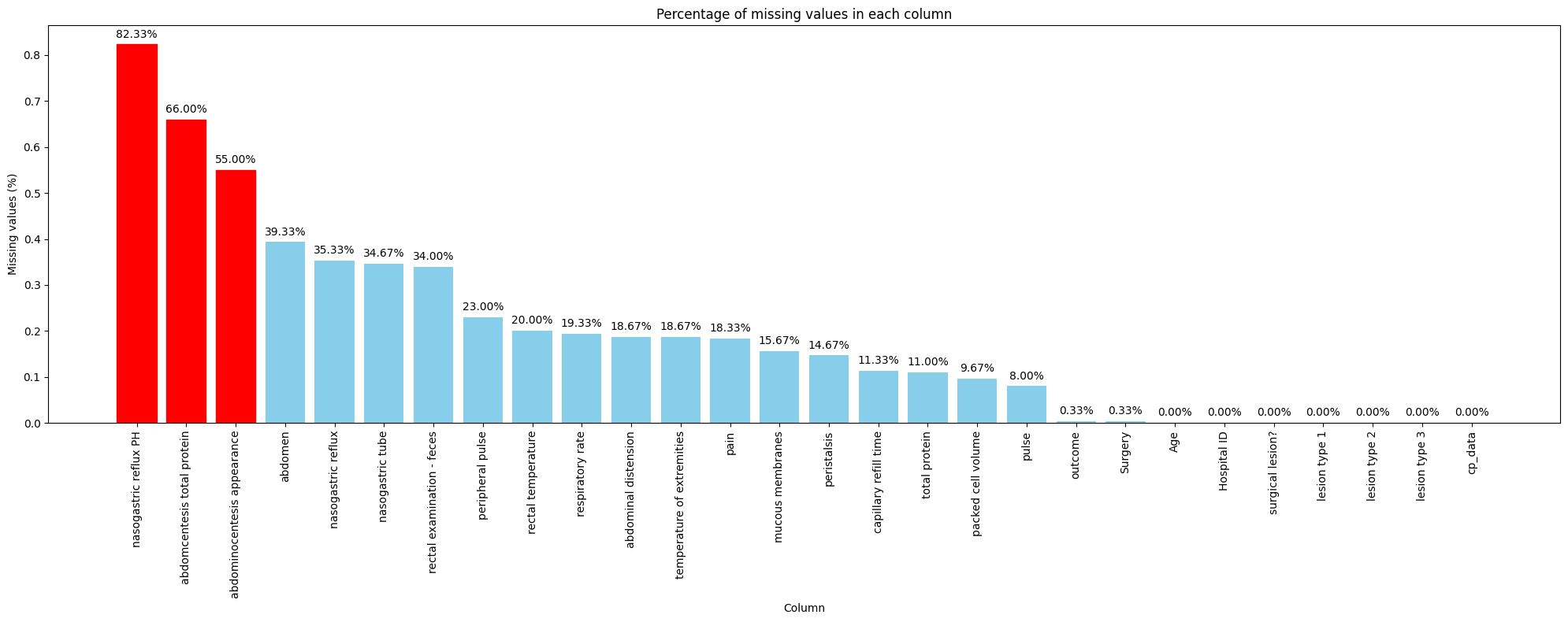
là sống, chết hay trợ tử.

Cấu trúc của dữ liệu: Bộ dữ liệu chứa các thông tin về tuổi, nhiệt độ, nhịp tim, hô hấp, các triệu chứng cơ thể, thông tin về điều trị,...

**Khám phá dữ liệu (EDA)**

Bộ dữ liệu bao gồm 28 biến bao gồm 27 cột và 300 dòng. Các biến được chia làm 2 loại

* Biến phân loại : 'Surgery', 'Age', 'mucous membranes', 'pain', 'peristalsis', 'abdominal distension', 'nasogastric tube', 'nasogastric reflux', 'rectal examination - feces', 'abdomen', 'abdominocentesis appearance', 'outcome', 'surgical lesion?', 'cp\_data'
* Biến số : 'Hospital ID', 'rectal temperature', 'pulse', 'respiratory rate', 'temperature of extremities', 'peripheral pulse', 'capillary refill time', 'nasogastric reflux PH', 'packed cell volume', 'total protein', 'abdomcentesis total protein', 'lesion type 1', 'lesion type 2', 'lesion type 3'

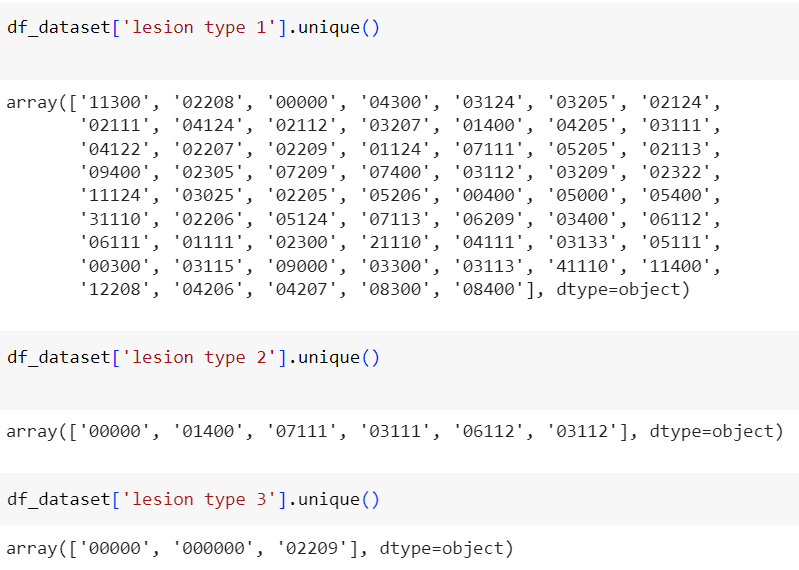
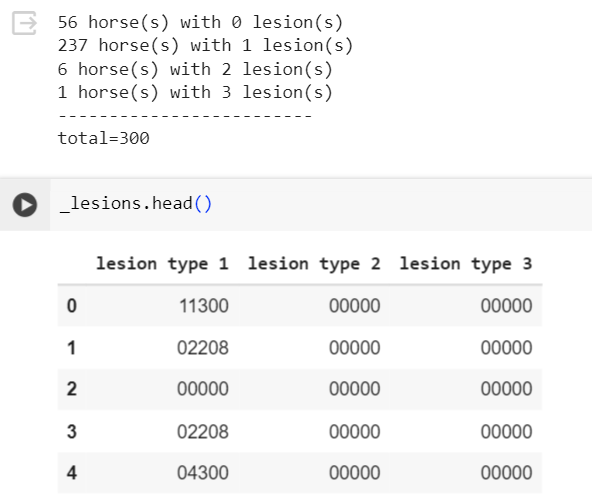


***Hình 3.1. Thông tin về dữ liệu Hình 3.2. Biểu đồ missing value***

Nhiều trường dữ liệu có giá trị null - theo mô tả, khoảng 30% giá trị bị thiếu. Chỉ có 7 cột trong tổng số 28 cột mà có giá trị trong mỗi hàng. Hơn nữa, chỉ có 6 hàng trong tổng số 300 hàng có đầy đủ giá trị trong mỗi cột.

Một số cột chứa rất nhiều dữ liệu bị thiếu, gây ảnh hưởng đáng kể đến tính toàn vẹn của thông tin. Các cột như nasogastric reflux PH, abdominocentesis appearance, và abdomcentesis total protein đang mất rất nhiều dữ liệu vì thế nên được loại bỏ ở bước tiền xử lý dữ liệu.

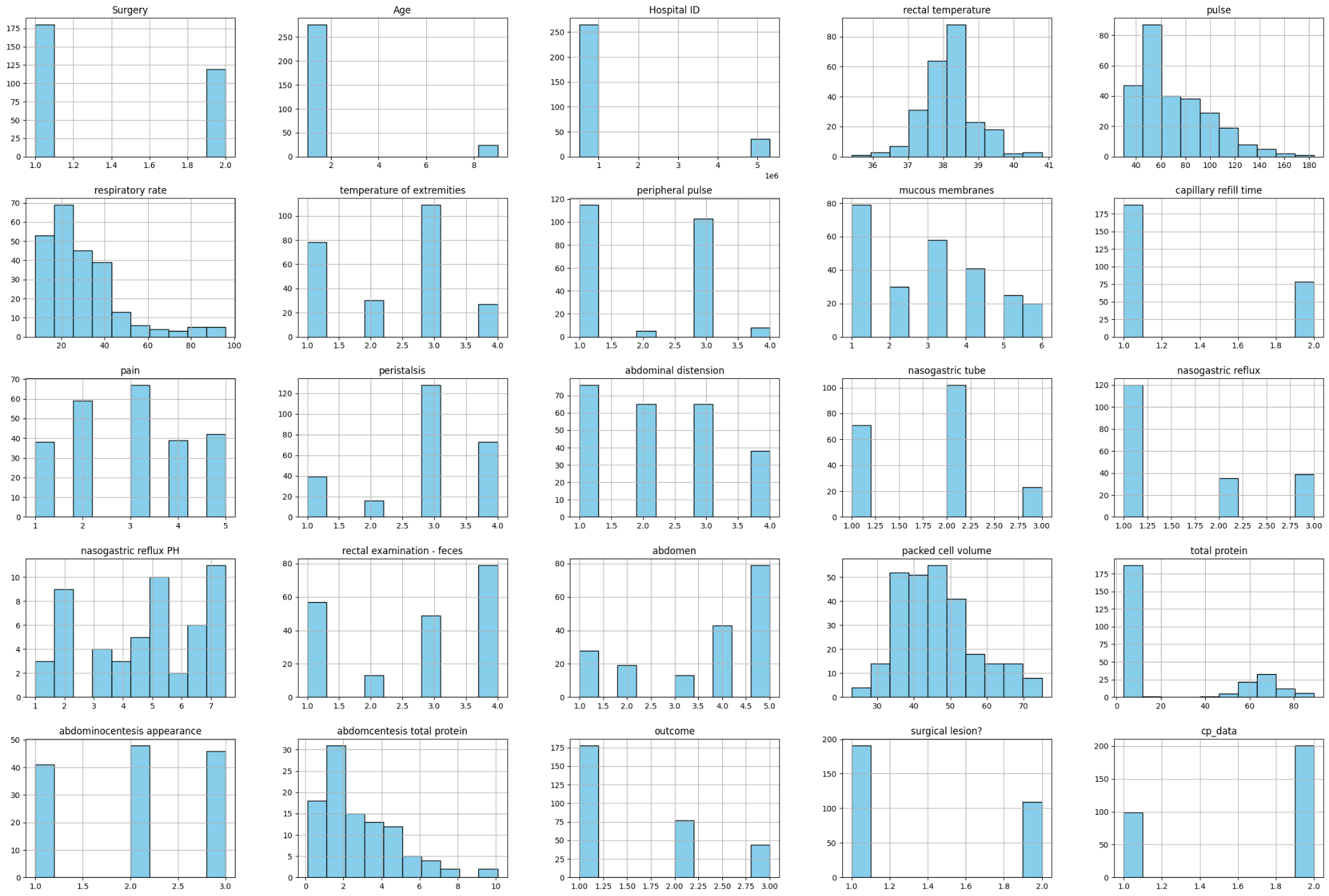
Những cột như “Hospital ID” và “cp\_data” không có ảnh hưởng lớn tới kết quả của bộ dữ liệu. Chúng sẽ được đề xuất loại bỏ để tập trung vào những yếu tố quan trọng hơn trong việc phân tích. Đối với cột “Surgical lesion”, mặc dù nó có ý nghĩa cho các nhiệm vụ phân loại khác, nhưng trong phạm vi phân tích hiện tại, chúng ta cũng sẽ loại bỏ để tập trung vào những khía cạnh chủ yếu liên quan đến sức khỏe và kết quả của ngựa.



***Hình 3.3. Quan sát 3 biến “lesion type 1”, “lesion type 2”, “lesion type 3”***

Dựa vào quan sát, ta nhận thấy dữ liệu của phần này tập trung vào tổn thương type of lession 1, trong khi đó dữ liệu trong cột type of lesion 2, và type of lesion 3 không có nhiều. Chính vì vậy, chúng tôi quyết định loại bỏ cột dữ liệu type of lesion 2, type of lesion 3 ở phần tiền xử lý dữ liệu. Đối với cột dữ liệu type of lesion 1 chúng tôi sẽ chuyển nó thành 5 cột mới “site of lesion”, “type”, “subtype”, “specific code”, “none” rồi loại bỏ cột type of lesion 1. Việc này sẽ giúp tối ưu hóa và chuẩn hóa thông tin tổn thương, dễ dàng phân tích hơn về các loại tổn thương của ngựa.

Như vậy, chúng ta sẽ đề xuất loại bỏ 9 cột và thêm vào 5 cột mới, tuy nhiên vẫn còn một số cột có số lượng ít giá trị rỗng, chúng tôi sẽ đề xuất xử lý giá trị rỗng theo các phương pháp khác nhau phù hợp tính chất của từng loại biến.

***Hình 3.4. Biểu đồ histogram***

Dựa vào quan sát biểu đồ ta có thể thấy :

Các biến liên tục như ‘rectal temperature’, ‘pulse’, ‘respiratory rate’, ‘nasogastric reflux PH’, ‘packed cell volume’, ‘total protein’, và ‘abdomcentesis total protein’ thường phản ánh thông tin liên quan đến tình trạng sức khỏe của ngựa. Phân bố lệch phải của chúng cho thấy có nhiều trường hợp có giá trị nhỏ hơn trung bình, điều này có thể là dấu hiệu cho việc có những trường hợp đặc biệt hoặc ngoại lệ, ví dụ như nồng độ protein cao hoặc nhiệt độ hậu môn không bình thường.

Các biến liên tục này có thể quan trọng trong việc xây dựng mô hình phân lớp, nhưng cũng cần xử lý các giá trị thiếu, ngoại lệ và nhiễu trước khi sử dụng chúng. Phương pháp thay thế giá trị thiếu hoặc giảm thiểu ảnh hưởng của ngoại lệ có thể được áp dụng, bao gồm sử dụng giá trị trung bình, trung vị hoặc chuẩn hóa dữ liệu. Kỹ thuật như KNN trong quá trình tiền xử lý cũng có thể giúp giải quyết các vấn đề này.

Các biến rời rạc, bao gồm thông tin như 'Surgery', 'Age', 'pain' có thể cung cấp thông tin về các đặc điểm và triệu chứng của viêm ruột ở ngựa. Tuy nhiên, chúng cũng có thể đối mặt với vấn đề giá trị rỗng, không có thứ tự tự nhiên hoặc có quá nhiều hạng mục. Để giải quyết, có thể áp dụng các phương pháp như mã hóa one-hot, giảm số lượng hạng mục hoặc mã hóa nhãn để chuẩn hóa và tối ưu hóa dữ liệu trước khi sử dụng cho mô hình.

Quan sát biểu đồ ta có thể thấy vấn đề mất cân bằng dữ liệu trên lớp dự báo “outcome”. Điều này có thể tác động lớn đến hiệu suất của mô hình và đưa ra dự đoán chính xác. Chúng tôi dự định sử dụng một phương pháp gọi là SMOTE để xử lý việc mất cân bằng dữ liệu của biến “outcome” ở bước tiền xử lý dữ liệu. Phương pháp này tạo ra các mẫu nhân tạo từ lớp thiểu số trong dữ liệu, giúp cân bằng lại tỷ lệ giữa các lớp và cải thiện khả năng dự đoán của mô hình. Sau khi tiến hành khám phá dữ liệu cơ bản chúng ta sẽ tiến hành tiền xử lý dữ liệu dựa trên các đề xuất đưa ra.

## 3.3. Tiền xử lý dữ liệu

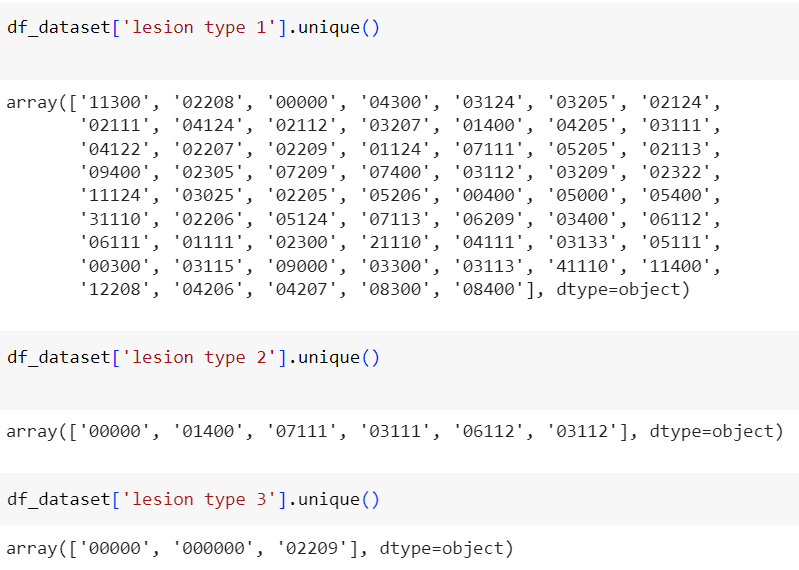
Như đã mô tả trong phần khai phá dữ liệu, chúng tôi sẽ bắt đầu tiến hành các bước chuẩn bị dữ liệu cho các mô hình học máy.

Đầu tiên, chúng ta sẽ chia bộ dữ liệu thành hai phần. Với việc chia dữ liệu thành hai phần sẽ giúp chúng ta có thể huấn luyện dữ liệu trên một phần gọi tập train và đánh giá kết quả mô hình ở trên phần dữ liệu còn lại gọi là tập test. Cụ thể, bộ dữ liệu sẽ được chia theo tỉ lệ 70-30, trong đó 70% dữ liệu được sử dụng cho tập train, 20% còn lại là tập test.

Sau khi việc chia dữ liệu hoàn tất, chúng tôi sẽ tiến hành tiền xử lý dữ liệu và huấn luyện mô hình trên tập train sau đó sẽ áp dụng những bước tương tự đối với tập test.

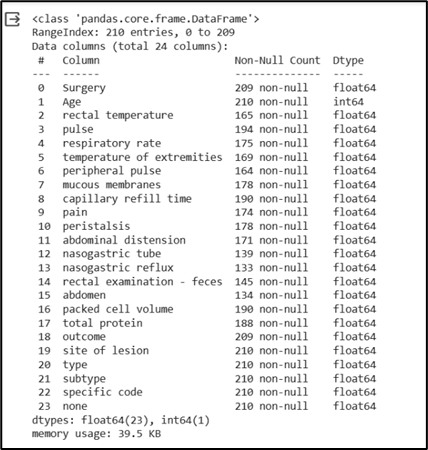
Trước hết, nhóm tác giả sẽ tiến hành xử lí những dữ liệu không chính xác trong tập dữ liệu ở biến “Age” để chuyển giá trị từ “9” sang “2” trong dataset để tương đồng với bản mô tả dữ liệu.

Việc dữ liệu bị thiếu quá nhiều ở các trường dữ liệu có thể gây ảnh hưởng đến kết quả dự báo phân lớp của các mô hình học máy, chính vì vậy chúng tôi tiến hành loại bỏ các cột: *“nasogastric reflux PH”, “abdominocentesis appearance”, “abdominocentesis total protein”.* Những biến không có tác dụng lớn tới kết quả của bộ dữ liệu cũng sẽ được loại bỏ như *“Hospital ID”* và *“cp\_data”.* Đối với biến *“Surgical lesion”* là biến được sử dụng cho nhiệm vụ phân loại khác như đã đề cập ở trên chính vì vậy chúng tôi cũng sẽ loại bỏ.



***Hình 3.5. Quan sát biến lesion type 1,2,3***

Như đã phân tích ở phần khám phá dữ liệu, chúng tôi cũng sẽ tiếp tục loại bỏ trường dữ liệu “lesion type 2”, “lesion type 3”. Sau đó chia giá trị của cột “lesion type 1” thành 5 cột mới “site of lesion”, “type”, “subtype”, “specific code”, “none”. Cuối cùng, chúng ta sẽ loại bỏ đi cột “lesion type 1”. Sau khi thực hiện quá trình này, tập dữ liệu huấn luyện sẽ bao gồm 24 biến như hình.



***Hình 3.6 .Thông tin của tập huấn luyện sau xử lí***

Từ quan sát, chúng tôi nhận thấy có 1 hàng duy nhất có giá trị bị thiếu trong hai cột “surgery” và “outcome”, chính vì vậy chúng tôi sẽ loại bỏ hàng này. Bằng việc này, chúng ta chỉ mất đi 1 quan sát có thể sửa được giá trị rỗng trong cả 2 cột.

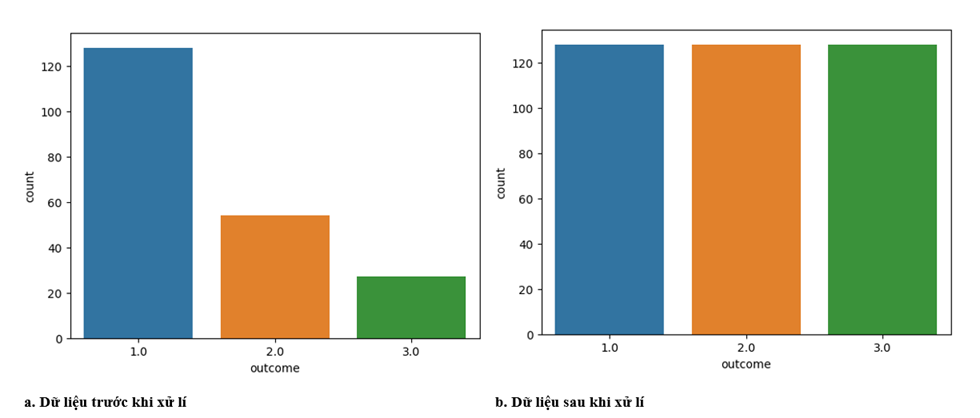
Tiếp đến, chúng tôi sẽ điền dữ liệu bị khuyết vào trong những ô bị khuyết còn lại trong bộ dữ liệu thông qua 3 cách khác nhau:

- Điền vào các cột liên tục bằng giá trị trung bình (mean)

- Điền vào các cột rời rạc bằng giá trị mode.

- Và cuối cùng, chúng ta sẽ điền vào những cột quan trọng như “abdominal distension” bằng mô hình kNN.

Sau khi đã điền giá trị khuyết theo các phương pháp trên, chúng tôi tiếp tục đi đến bước tiếp theo là xử lý vấn đề mất cân bằng trên lớp dự báo “outcome”. Có nhiều phương pháp khác nhau để có thể giải quyết vấn đề này, và một trong những thuật toán phổ biến mà chúng tôi sử dụng trong phạm vi bài này là phương pháp SMOTE dùng để sinh thêm các phần tử thuộc lớp thiểu số, giúp mô hình dự báo chính xác hơn.



***Hình 3.7. Dữ liệu trước và sau khi xử lý bằng SMOTE***

Sau khi xử lí vấn đề mất cân bằng giữa các lớp dự báo, tập dữ liệu đã tăng từ 209 quan sát lên 384 quan sát. Điều này có thể giúp tập dữ liệu cải thiện hiệu suất của mô hình dự báo, tạo sự cân bằng từ đó đảm bảo rằng kết quả dự đoán sẽ chính xác và tin cậy hơn.

Đối với những mô hình học máy như SVm hay KNN, việc chuẩn hóa dữ liệu là một việc hết sức quan trọng, vì vậy trước khi đưa dữ liệu vào các mô hình học máy chúng tôi sẽ sử dụng phương pháp chuẩn hóa Min Max Normalization để chuyển đổi dữ liệu về cùng một miền giá trị.

Từ những kết quả khám phá dữ liệu và sau những quy trình tiền xử lý, bộ dữ liệu đã sẵn sàng để đưa vào huấn luyện dự báo và sẽ được trình bày trong phần tiếp theo của báo cáo.

# 4. Thử nghiệm và kết quả

## 4.1. Môi trường thiết lập

Chúng tôi xây dựng chương trình trên máy ảo Linux được cung cấp bởi Colaboratory của Google, với Python 3.10. Trong chương trình, chúng tôi sử dụng sklearn như là thư viện chính để phát triển các mô hình học máy.

**Thiết lập huấn luyện**

Trong quá trình huấn luyện, chúng tôi thử nghiệm trên 10 mô hình: (1) kNN, (2) naive Bayes, (3) SVM, (4) decision tree, (5) random forest, (6) AdaBoost, (7) gradient Boosting, (8) linear discriminant analysis, (9) multi-layer perceptron và (10) logistic regression và sử dụng các chỉ số accuracy và confusion matrix để so sánh hiệu suất của các phương pháp phân loại. Chúng tôi chuẩn bị dữ liệu kiểm nghiệm theo phương pháp hold-out: Tập dữ liệu được chia thành 2 phần train/test với tỉ lệ 7/3.

Tập train sẽ được dùng để huấn luyện, điều chỉnh tham số với chiến lược:

* Hold-out (tiếp tục chia 7/3 với train/valid).
* k-fold : k=5 (chia thành k phần đều nhau với k-1 phần cho train/1 phần cho valid).
* Random\_state = 42: siêu tham số ngẫu nhiên trong học máy, để thiết lập seed ngẫu nhiên, giúp đảm bảo tính nhất quán và tái tạo kết quả giữa các lần chạy mô hình khác nhau.

Trong đó, train là dùng huấn luyện và valid dùng điều chỉnh tham số.

Tập test dùng để kiểm nghiệm lại độ hiệu quả của thuật toán sau khi chọn mô hình tối ưu.

## 4.2. Đề xuất chiến lược

Trong quá trình xây dựng mô hình, chúng tôi sử dụng những metric như accuracy, F1 score, precision và recall để đánh giá chất lượng của mô hình:

**Accuracy**: tỷ lệ giữa số điểm dữ liệu được phân loại đúng và tổng số điểm dữ liệu. Accuracy giúp đánh giá khả năng dự đoán chung của mô hình, nhưng không phản ánh được sự cân bằng giữa các lớp dữ liệu.

**F1-scores**: trung bình điều hòa của Precision và Recall, giúp đánh giá sự cân bằng giữa hai chỉ số này. F1 càng cao tương ứng Precision và Recall càng cao, mô hình phân loại càng tốt.

**Precision**: tỷ lệ giữa số điểm dữ liệu thuộc lớp Positive được phân loại đúng và tổng số điểm dữ liệu được phân loại là Positive. Precision giúp đánh giá khả năng tránh nhầm lẫn các điểm Negative là Positive.

**Recall**: tỷ lệ giữa số điểm dữ liệu thuộc lớp Positive được phân loại đúng và tổng số điểm dữ liệu thuộc lớp Positive. Recall giúp đánh giá khả năng phát hiện được các điểm Positive.

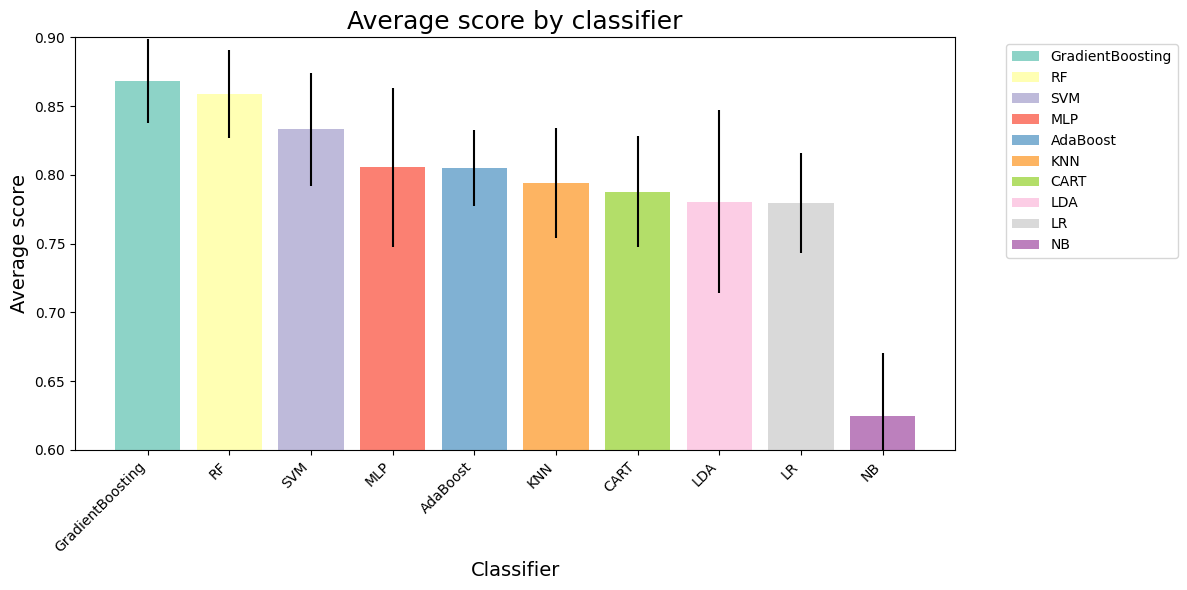
Từ những nội dung ở trên, nhóm tác đưa ra mô hình đề xuất để dự báo như bảng dưới đây:

| **STT** | **Mô hình** | **Accuracy** | **F1** | **Precision** | **Recall** |
| --- | --- | --- | --- | --- | --- |
| 1 | Gradient Boosting |  |  |  |  |
| 2 | Random Forest |  |  |  |  |
| 3 | SVM |  |  |  |  |
| 4 | Multi-layer perceptron |  |  |  |  |
| 5 | AdaBoost |  |  |  |  |
| 6 | KNN |  |  |  |  |
| 7 | Decision tree |  |  |  |  |
| 8 | Linear discriminant analysis |  |  |  |  |
| 9 | Logistic regression |  |  |  |  |
| 10 | Naive Bayes |  |  |  |  |

## 4.3. Kết quả dữ liệu huấn luyện (tập train)

## 

***Hình 3.8. Kết quả huấn luyện mô hình***



Điểm trung bình của các mô hình theo thứ tự

- Mô hình **Gradient Boosting** có hiệu suất cao nhất trên tất cả các chỉ số, cho thấy nó là mô hình phù hợp nhất cho bài toán phân loại này. Mô hình này có *Accuracy, F1, Precision, Recall* và *Average score* đều trên **86%**, và *Average std* thấp nhất, chỉ **0.03**, cho thấy kết quả của nó ổn định và ít biến động.

- Mô hình **RF** cũng có hiệu suất tốt, xếp thứ hai trên tất cả các chỉ số, chỉ thua Gradient Boosting một chút. Mô hình này có *Accuracy, F1, Precision, Recall* và *Average score* đều trên **85%**, và *Average std* cũng thấp, chỉ **0.03**. Điều này cho thấy mô hình này cũng có khả năng phân loại tốt và ổn định.

- Mô hình **NB** có hiệu suất thấp nhất trên tất cả các chỉ số, cho thấy nó là mô hình không phù hợp cho bài toán phân loại này. Mô hình này có *Accuracy, F1, Recall và Average score* đều dưới **62%**, và *Average std* cao nhất, **0.04**. Điều này cho thấy mô hình này có khả năng phân loại kém và không nhất quán.

- Mô hình **LDA** có *Average std* cao nhất trong các mô hình còn lại: **0.06**, cho thấy kết quả của nó có độ biến động lớn và không ổn định. Mô hình này có *Accuracy, F1, Precision, Recall và Average score đều dưới* ***78%*,** cho thấy hiệu suất của nó không cao.

=> Lựa chọn chiến lược tinh chỉnh: 2 mô hình **Gradient Boosting** và **RF:** tìm kiếm siêu tham số (**hyperparameter tuning**) bằng cách sử dụng **'GridSearchCV'**

* **Lý do đề xuất tinh chỉnh:**

- Mô hình máy học có nhiều siêu tham số quyết định hiệu suất của nó. Việc lựa chọn tổ hợp tốt nhất giữa các giá trị này là quan trọng để tránh overfitting hoặc underfitting.

- Tùy thuộc vào dữ liệu cụ thể, các giá trị tối ưu của siêu tham số có thể thay đổi. Do đó, tìm kiếm này giúp định rõ các giá trị phù hợp nhất cho tập dữ liệu đang xử lý.

- Việc tinh chỉnh siêu tham số giúp đạt được mô hình chính xác hóa trên dữ liệu mới và giảm thiểu sai số dự đoán.

**Ưu việt của GridSearchCV so với các phương pháp khác và độ phù hợp với mô hình:**

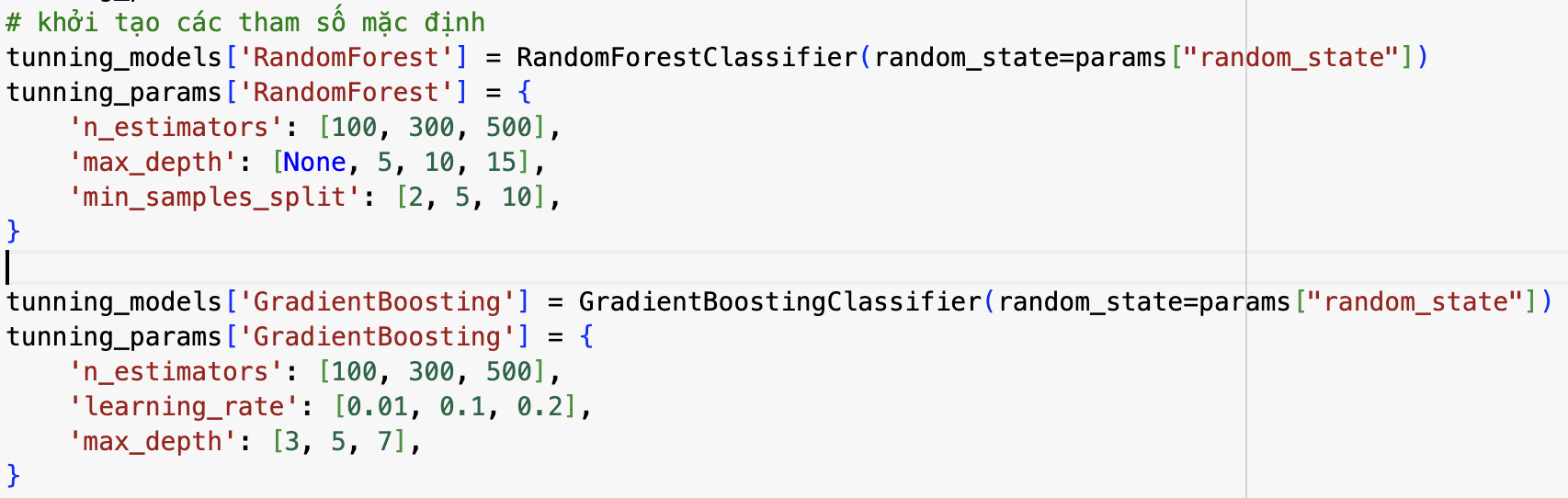
1. Toàn diện: Kiểm tra mọi tổ hợp có thể của các giá trị siêu tham số, đảm bảo không bỏ sót giải pháp tốt nhất.

2. Tự động hóa: GridSearchCV tự động thực hiện quá trình lặp qua lưới thử nghiệm, giúp giảm bớt công việc lặp đi lặp lại.

3. Đánh giá chéo: Sử dụng phương pháp kiểm tra chéo (cross-validation) để đánh giá hiệu suất mô hình trên nhiều tập dữ liệu khác nhau, giúp đảm bảo tính khái quát của kết quả.

*=>* Phương pháp này không phải lúc nào cũng là lựa chọn tốt nhất, vì có những ưu điểm và nhược điểm. Một số ưu điểm bao gồm sự toàn diện, tự động hóa và đánh giá chéo, nhưng có nhược điểm về chi phí tính toán, đặc biệt là khi không gian thử nghiệm dữ liệu lớn. Ở đây trong trường hợp bộ dữ liệu chỉ có 300 quan sát và chỉ có 2 mô hình cần tinh chỉnh, đặc biệt là GB và RF thì việc sử dụng GridSearchCV là khá tối ưu.

***Thông số tinh chỉnh***

**

***Kết quả thông số tinh chỉnh:***

1. **RandomForest:**

Kết quả tốt nhất: 0.8621326042378673

Số cây quyết định (n\_estimators): 500

Độ sâu tối đa của cây (max\_depth): Không giới hạn (None)

Số mẫu tối thiểu để chia một nút (min\_samples\_split): 2

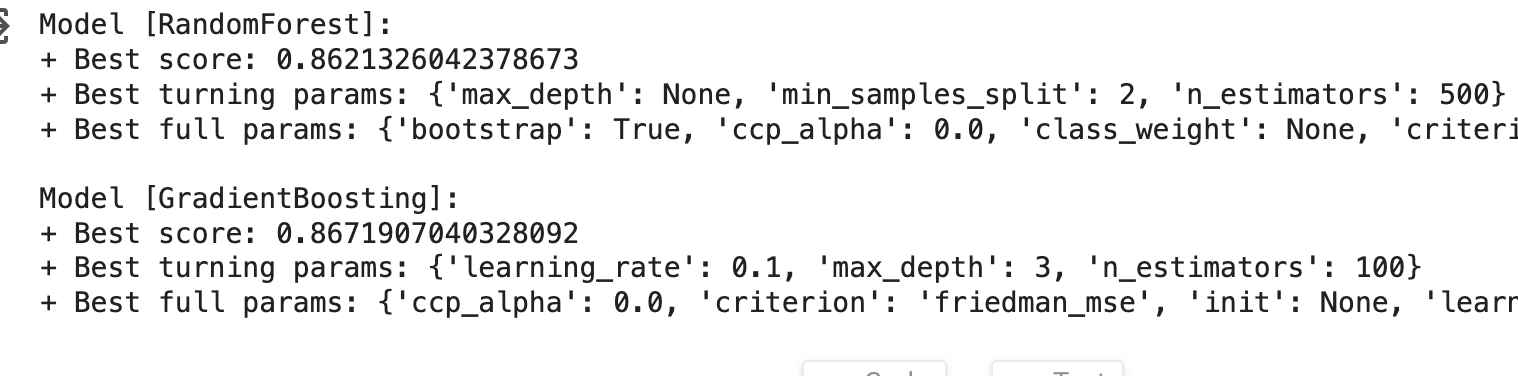
1. **Gradient Boosting:**

Kết quả tốt nhất: 0.8671907040328092

Số cây quyết định (n\_estimators): 100

Tốc độ học (learning\_rate): 0.1

Độ sâu tối đa của cây (max\_depth): 3



## 

## 4.4. Kết quả dữ liệu dự báo (tập test)

*1. Vì bộ dữ liệu đã chọn có quá nhiều missing value và nhiều feature nên ta cần tiền xử lý dữ liệu của tập test, để tránh cho một số mô hình không thể chạy được trên tập test có missing value. Thực hiện fill và scale cũng như chuẩn hóa lại lại giống bước processing (không làm thay đổi kết quả của biến* ***outcome*** *cần dự đoán).*

*2. Thực hiện đánh giá baseline các chỉ số, ta thu được kết quả 2 mô hình trong tập train cao nhất cũng chính là 2 mô hình của tập test có giá trị cao nhất:*

***Tunning [RandomForest]***

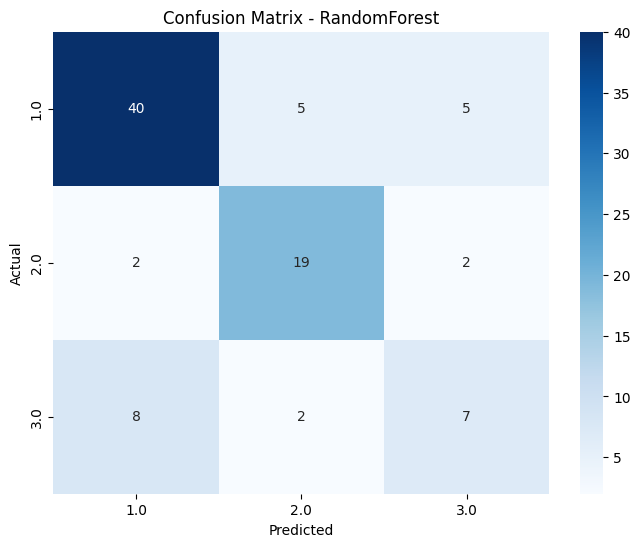
*Baseline [RF]*

*+ acc = 0.700 => Sau khi tinh chỉnh 0.733*

*+ precision = 0.699*

*+ recall = 0.700*

*+ F1-score = 0.697*

**

***Tunning [GradientBoosting]***

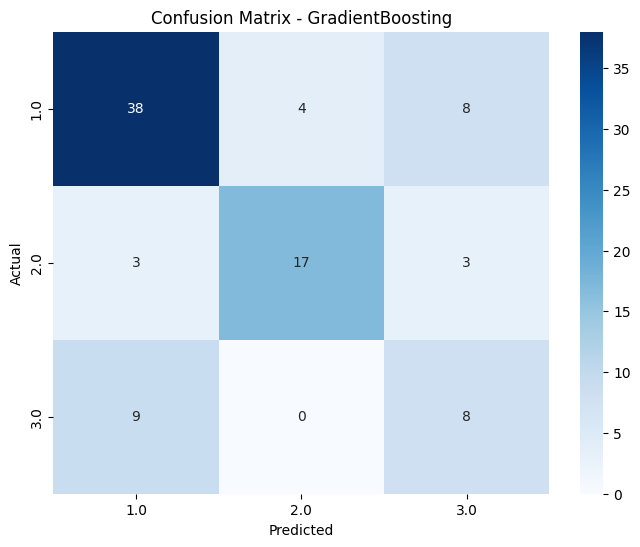
*Baseline [GradientBoosting]*

*+ acc = 0.700*

*+ precision = 0.709*

*+ recall = 0.700*

*+ F1-score = 0.704*

**

**Chất lượng Dữ liệu:** Có thể bộ dữ liệu Horse Colic có chất lượng tốt, đủ đa dạng và có tính biểu diễn cho bài toán dự đoán về sức khỏe của ngựa sau mổ phẫu thuật.

**Hiệu Suất Tổng Thể:** Cả RF và GB đều đạt được độ chính xác (accuracy) ở mức cao (0.733 và 0.700) sau quá trình tinh chỉnh, cho thấy khả năng dự đoán tổng thể là khá tốt. Trong 2 ma trận nhầm lẫn, thì các trường hợp dự đoán Sống: 1.0, dự đoán tốt hơn các trường hợp chết và trợ tử (2.0 và 3.0). Lớp thứ 3.0 trong CFM đánh giá không chính xác nhất.

**Chính Xác và Đầy Đủ:** Cả RF và GB đều có precision, recall và F1-score ở mức cao, có nghĩa là cả hai mô hình đều có khả năng dự đoán chính xác và đầy đủ trên cả hai lớp (ngựa bình thường và có vấn đề sau mổ).

**Khả năng Tinh Chỉnh (Tuning):** Qua quá trình tinh chỉnh, không có sự cải thiện đáng kể về các chỉ số đánh giá. Cả RF và GB đều giữ nguyên độ chính xác và hiệu suất so với baseline.

**Sự Lựa Chọn giữa RF và GB:** RF và GB đều cho kết quả tốt và không có sự chênh lệch đáng kể giữa chúng. Lựa chọn giữa RF và GB có thể phụ thuộc vào yêu cầu cụ thể của bài toán và đặc điểm của dữ liệu.

**Ổn Định của RF và GB:** Cả hai mô hình đều có khả năng đồng đều và ổn định trong việc dự đoán trên bộ dữ liệu này, không có dấu hiệu của overfitting hay underfitting.

# 

# 5. Kết luận

*Bài toán dự đoán sức khỏe của ngựa sau mổ phẫu thuật trên bộ dữ liệu horse colic là một vấn đề quan trọng trong lĩnh vực y học thú y. Dựa trên thông tin y tế và các chỉ số liên quan, mục tiêu là xác định liệu ngựa có phục hồi thành công sau phẫu thuật hay không.*

1. **Random Forest (RF):**

RF là một mô hình ensemble dựa trên ý tưởng của rừng cây quyết định. RF có khả năng làm việc tốt trên dữ liệu có nhiều đặc trưng và tự động xử lý việc chọn lọc các đặc trưng quan trọng. Mô hình RF giữ được độ chính xác và hiệu suất ổn định trên bộ dữ liệu, phù hợp khi có sự phức tạp trong mối quan hệ giữa các biến đầu vào và kết quả.

**Ưu điểm:**

* Ổn định và Hiệu suất Cao: RF thường cho kết quả ổn định và hiệu suất cao trên nhiều loại dữ liệu khác nhau mà không yêu cầu nhiều điều chỉnh.
* Xử lý Đặc trưng: RF tự động xử lý các đặc trưng quan trọng, giảm công đoạn lựa chọn đặc trưng riêng biệt.

**Nhược điểm:**

* Tốn Thời Gian Huấn Luyện: Đối với số cây lớn, quá trình huấn luyện có thể tốn thời gian.
* Không Phải Là Mô Hình Nhỏ Gọn: RF có thể tạo ra một số lượng lớn cây, làm tăng kích thước của mô hình.

1. **Gradient Boosting (GB):**

GB là một mô hình ensemble khác, tập trung vào việc xây dựng các cây quyết định theo cách tuần tự và cố gắng khắc phục các sai số của các cây trước đó. GB có khả năng học được từ những lỗi trong quá trình dự đoán, điều này có thể giúp mô hình cải thiện hiệu suất. Trong trường hợp này, GB không thể vượt qua RF về hiệu suất.

**Ưu điểm:**

* Hiệu Suất Cao: GB thường có hiệu suất cao và có thể học được từ lỗi của các cây trước đó.
* Khả năng Học Tốt với Dữ liệu Khó: GB có khả năng xử lý dữ liệu có độ phức tạp cao hơn và có thể tạo ra các dự đoán chính xác hơn trên tập dữ liệu phức tạp.

**Nhược điểm:**

* Cần Điều Chỉnh Tham số: GB yêu cầu sự điều chỉnh tham số kỹ lưỡng để đạt được hiệu suất tốt nhất.
* Nhạy cảm với Nhiễu: Có thể bị ảnh hưởng bởi nhiễu và outliers.

**So Sánh và Lựa Chọn Mô Hình:**

Cả hai mô hình đều đạt được kết quả tốt, nhưng không có sự chênh lệch đáng kể giữa RF và GB. Sự chọn lựa giữa RF và GB có thể phụ thuộc vào các yếu tố khác nhau như tốc độ huấn luyện, khả năng giải thích, và sự phức tạp của mô hình. Trong trường hợp này, sự ổn định và hiệu suất của RF có thể làm cho nó trở thành một lựa chọn hợp lý.

Để có cái nhìn toàn diện hơn, cần kiểm tra kỹ hơn về từng mô hình, kiểm tra các đặc trưng quan trọng, và thử nghiệm các phương pháp tinh chỉnh khác nhau để xem liệu có thể cải thiện hiệu suất hay không.

**Phương hướng sau khi thực hiện:**

* **Tinh chỉnh Tham số:** Tiếp tục tinh chỉnh tham số cho cả hai mô hình để xem liệu có thể cải thiện hiệu suất hay không.
* **Tổ Hợp Mô Hình (Ensemble):** Xem xét việc sử dụng ensemble của RF và GB để kết hợp ưu điểm của cả hai mô hình.
* **Kiểm Soát Kích Thước Mô Hình:** Nếu kích thước mô hình là một vấn đề, có thể xem xét giảm số cây hoặc thử nghiệm các biện pháp kiểm soát kích thước.

# TÀI LIỆU THAM KHẢO

[1] **Horse Colic - UCI Machine Learning Repository,** “Mary McLeish”

(https://archive.ics.uci.edu/dataset/47/horse+colic)

[2]Jason Brownlee, “Machine Learning Mastery with Python: Understand Your Data Create Accurate Models and Work Projects End-to-End

[3] **Identifying-Serious-Cases-of-Colic-in-Equines,** “yeagercmbpd”

(https://github.com/yeagercmbpd/Identifying-Serious-Cases-of-Colic-in-Equines)

[4] **Similarity networks for classification: a case study in the Horse Colic problem**, “Lluís Belanche, Jerónimo Hernández”

[5] **DP4ML - Missing Data - Phần 3 - kNN Imputation**

(https://tiensu.github.io/blog/80\_data\_prepeation\_for\_ml\_data\_cleaning\_missing\_data\_3/)

[6] **Các phương pháp đánh giá mô hình học máy, học sâu (Machine learning & Deep learning)**

(https://rabiloo.com/vi/blog/cac-phuong-phap-danh-gia-mo-hinh-machine-learning-va-deep-learning)

[7] **Data visualization trong Machine Learning - Phần 1,** “Cao Minh Ngoc”

(https://tiensu.github.io/blog/79\_data\_prepeation\_for\_ml\_data\_cleaning\_missing\_data\_2/)

[8] **Chuẩn bị dữ liệu cho học máy,** “Jason Brownlee”:

( https://machinelearningmastery.com/data-preparation-for-machine-learning/)