# **Titanic - Machine Learning from Disaster**

**Họ tên: Võ Gia Kiệt**

**Mã Sinh Viên: 3123410182**

1. **Introduce**

**Mục tiêu:**

Dự đoán khả năng sống sót của hành khách trên tàu Titanic dựa vào các đặc trưng trong bộ dữ liệu.

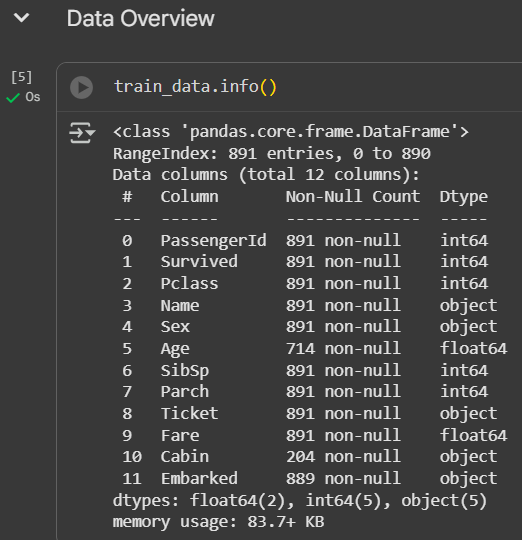
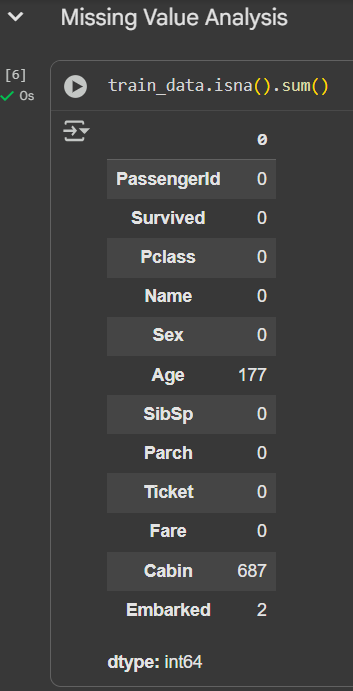
Xây dựng mô hình học máy để phân loại nhị phân (0 = không sống sót, 1 = sống sót).

Đánh giá hiệu quả mô hình thông qua các chỉ số như Accuracy, Precision, Recall, F1-score, AUC.

Bộ dữ liệu: Titanic từ Kaggle.

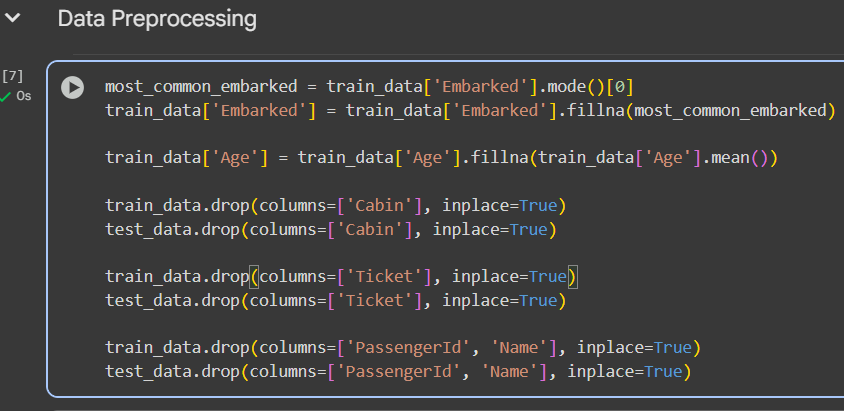
1. **Exploratory** **Data Analysis (EDA)**
2. **Missing Value Analysis**

Sau khi loading data ta có thể nhận thấy các dữ liệu bị thiếu như các cột Age, Cabin, Embarked.

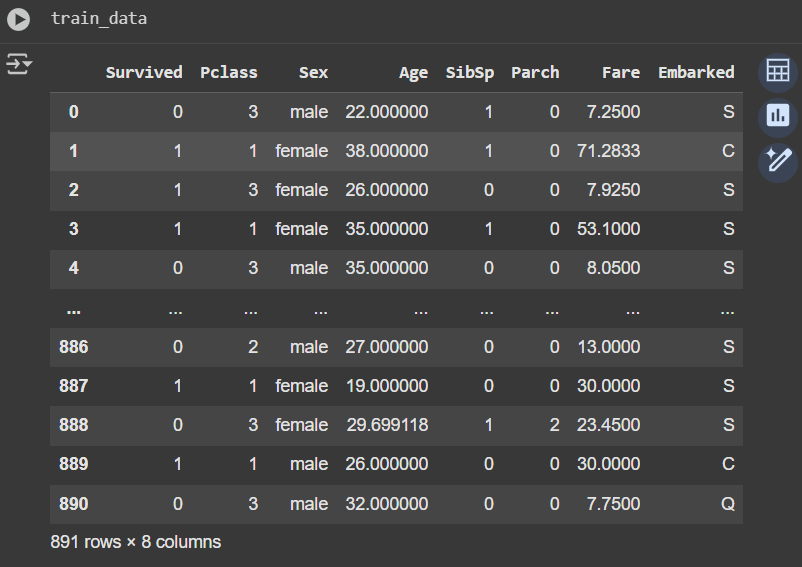
 

1. **Data Preprocessing**

Sau khi phân tích các dữ liệu bị thiếu tiến hành xử lí các dữ liệu bị thiếu để model hoạt động chính xác.



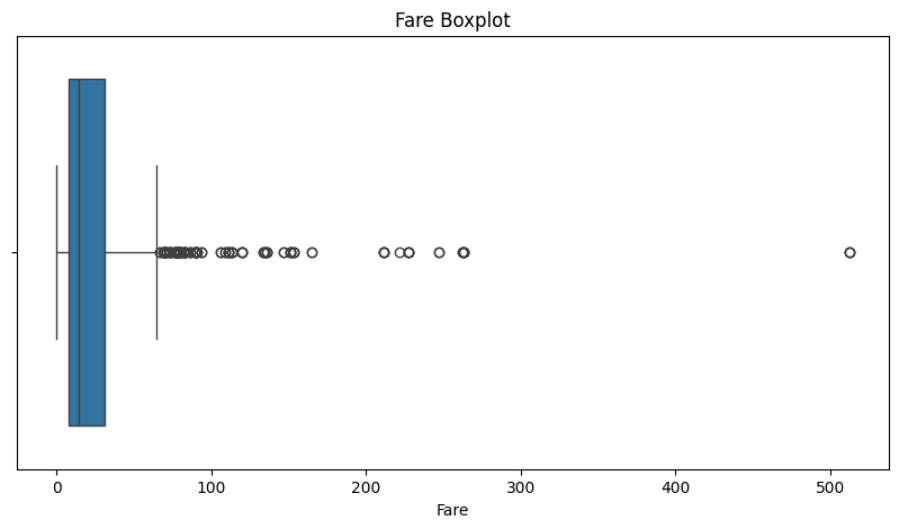
* mode()[0]: tìm giá trị xuất hiện nhiều nhất trong cột Embarked và thay thế các giá trị bị thiếu thành các giá trị phổ biến nhất.
* Dùng **giá trị trung bình (mean)** của Age để điền vào các dòng bị thiếu và loại bỏ các cột Cabin, Ticket, PassengerID, Name.



1. **Validating Data Cleaning**

* Kiểm tra xem tập dữ liệu còn dữ liệu chưa được xử lí hay không và sau khi xử lí dữ liệu bị thiếu tiến hành làm sạch dữ liệu các cột Sex và Embarked đều có dạng object.
* Sau đó, ta thường phải **chuyển đổi (encode)** các cột này thành số (ví dụ dùng One-Hot Encoding hoặc Label Encoding), vì mô hình ML không hiểu trực tiếp dữ liệu dạng text.

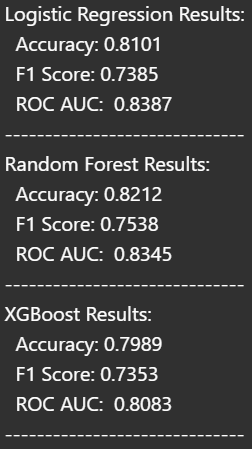




* Biểu đồ Boxplot cho biến Fare được vẽ ra để kiểm tra sự phân bố và phát hiện các giá trị bất thường trong dữ liệu. Kết quả cho thấy phần lớn hành khách có mức giá vé tương đối thấp, nhưng tồn tại một số trường hợp có giá vé rất cao so với đa số. Điều này cho thấy dữ liệu bị lệch phải (right-skewed) và có nhiều outliers.
* Xử lý các outliers bằng cách lấy log để dữ liệu cân đối cho mô hình.

1. **Model Training & Evaluation**

* Sau khi làm sạch dữ liệu tiến hành huấn luyện mô hình với 3 mô hình sau logistic regression, random forest, xgboost nhằm so sánh hiệu suất và độ chính xác của từng mô hình.



* Độ chính xác được đánh giá qua 3 thông số:

**Accuracy**: tỷ lệ dự đoán đúng trên toàn bộ dữ liệu.

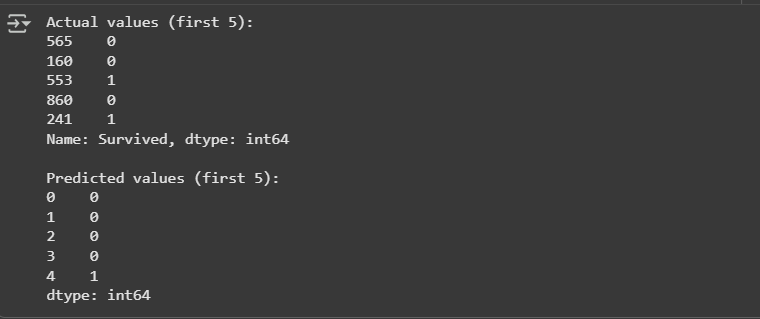
**F1 Score**: cân bằng giữa Precision và Recall, phù hợp trong bài toán mất cân bằng nhãn.

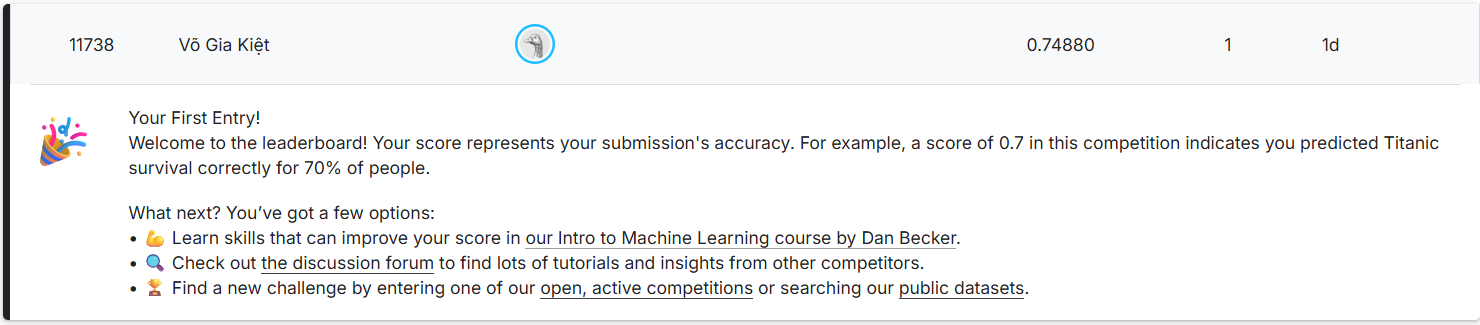
**ROC AUC**: đo lường khả năng phân biệt giữa hai lớp (sống sót và không sống sót).

* **Random Forest** đạt hiệu suất cao nhất với Accuracy và F1 Score vượt trội so với Logistic Regression và XGBoost.
* **Logistic Regression** cũng cho kết quả khá tốt, chứng tỏ quan hệ tuyến tính có thể mô tả một phần dữ liệu.
* **Xgboost** cho kết quả thấp nhất.

Từ đó kết luận mô hình **Random Forest đưa ra kết quả chính xác nhất.**

1. **Result**





**Nhận xét:**

* Do xử lí các dữ liệu bị thiếu chưa được tối ưu nên dẫn đến việc dự đoán kết quả bài toán chưa thật sự đúng.
* Một số đặc trưng quan trọng như Cabin đã bị loại bỏ do dữ liệu thiếu, dẫn đến mất thông tin hữu ích. Nếu thay vì loại bỏ, ta xử lý bằng cách tạo biến HasCabin (0/1), có thể cải thiện hiệu quả mô hình.