

**ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH**

**TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN**

**KHOA KHOA HỌC MÁY TÍNH**

**A blue logo with a black background

Description automatically generated with low confidence**

**BÀI TẬP CHƯƠNG 7**

Giảng Viên: Nguyễn Thị Anh Thư

Mã Lớp: CS313.O21

Nhóm 04

| *Trương Văn Khải* | *21520274* |
| --- | --- |
| *Đoàn Nhật Sang* | *21522542* |
| *Lê Ngô Minh Đức* | *21520195* |
| *Lê Minh Quang* | *21522510* |
| *Lê Yến Nhi* | *21522427* |
| *Hoàng Thị Mỹ Hạnh* | *21522044* |
| *Hoàng Tiến Đạt* | *21520696* |
| *Phạm Minh Quốc* | *22540017* |

**TP. HỒ CHÍ MINH, 3/2024**

**NHẬN XÉT CỦA GIẢNG VIÊN HƯỚNG DẪN**

........................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................ ............................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................................ .................................................................................................................................... .................................................................................................................................... .................................................................................................................................... .................................................................................................................................... ...................................................................................................................................

**BẢNG PHÂN CÔNG CÔNG VIỆC**

| **Chi tiết công việc** | | **Trương Văn Khải (NT)** | **Đoàn Nhật Sang** | **Lê Ngô Minh Đức** | **Lê Minh Quang** | **Lê Yến Nhi** | **Hoàng Thị Mỹ Hạnh** | **Hoàng Tiến Đạt** | **Phạm Minh Quốc** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **Phân công** | |  |  |  |  |  |  |  |  |
| **Xác định mục tiêu và yêu cầu** | |  |  |  |  |  |  |  |  |
| **Chuẩn bị dữ liệu** | **3.1**  **3.2** |  |  |  |  |  |  |  |  |
| **3.3**  **3.4** |  |  |  |  |  |  |  |  |
| **3.5**  **3.6** |  |  |  |  |  |  |  |  |
| **Thuật toán phân loại và đánh giá mô hình** | **4.1**  **4.2** |  |  |  |  |  |  |  |  |
| **4.3**  **4.4** |  |  |  |  |  |  |  |  |
| **4.5**  **4.6** |  |  |  |  |  |  |  |  |
| **4.7**  **4.8** |  |  |  |  |  |  |  |  |
| **Kết luận** | |  |  |  |  |  |  |  |  |
| **Quay video** | |  |  |  |  |  |  |  |  |
| **Format docs** | |  |  |  |  |  |  |  |  |
| **Kiểm tra** | |  |  |  |  |  |  |  |  |
| **Mức độ hoàn thành** | | **100%** | **100%** | **100%** | **100%** | **100%** | **100%** | **100%** | **100%** |

**MỤC LỤC**

[**1. Xác định mục tiêu và yêu cầu 1**](#_tnkkyqiynv4h)

[1.1. Mục tiêu chính 1](#_3uz7rgcxflyz)

[1.2. Yêu cầu 1](#_rfto33wif7ye)

[**2. Bộ dữ liệu sử dụng 1**](#_qmezuwmzyeud)

[**3. Chuẩn bị dữ liệu 2**](#_ltaq93byrcsz)

[3.1. Load dữ liệu và EDA 2](#_vh906nsx90ux)

[3.2. Phát hiện ngoại lai (Outlier) 8](#_nsh701oxou3q)

[3.3. Rút trích đặc trưng (Feature engineering) 10](#_q92ggdghttte)

[3.4. Chọn lựa đặc trưng (Feature selection) 11](#_coo4j6d7xwl4)

[3.5. Train test split 12](#_mph3pprrqkaw)

[3.6. Chuyển đổi dữ liệu (Data transformation) 12](#_9g6yik8093hr)

[**4. Thuật toán phân loại và đánh giá mô hình 14**](#_hkktwv6glma7)

[4.1. LogisticRegression 14](#_hww6dv1h7oez)

[4.2. KNeighborsClassifier 15](#_w721yxpxbbqw)

[4.3. SVM 16](#_3jedxqyf0ilr)

[4.4. DecisionTreeClassifier 16](#_ywkzqvi2wcro)

[4.5. RandomForestClassifier 17](#_pw36w9b4lt12)

[4.6. LGBM 18](#_n4on958j4rrw)

[4.7. CatBoost 19](#_xlqlxuo2vlfc)

[4.8. XGBoost 20](#_2xiy0op0nakj)

[**5. Kết luận 21**](#_hjjwqd4mitw8)

**Link video thuyết trình**:

# 

# Xác định mục tiêu và yêu cầu

## 1.1. Mục tiêu chính

* Biết đến các loại mô hình phân lớp. Hiểu được nguyên lý hoạt động và sử dụng được các mô hình phân lớp cụ thể.
* Hiểu và áp dụng các kỹ thuật xây dựng đặc trưng, chuyển đổi dữ liệu, huấn luyện mô hình, và đánh giá hiệu suất của các mô hình phân lớp trên tập dữ liệu huấn luyện và kiểm tra.

## 1.2. Yêu cầu

* Sử dụng được các loại mô hình phân lớp kinh điển.
* Biết cách xử lý và làm sạch dữ liệu. Triển khai một số cách rút trích đặc trưng để tạo dữ liệu mới.

# Bộ dữ liệu sử dụng

Bộ dữ liệu mà nhóm lựa chọn sử dụng là bộ dữ liệu Titanic ([link](https://www.kaggle.com/competitions/titanic)):

**Giới thiệu:**

Bộ dữ liệu Titanic trên Kaggle là một tập dữ liệu lịch sử nổi tiếng được sử dụng rộng rãi trong các bài toán học máy. Dữ liệu này mô tả thông tin về hành khách trên tàu Titanic, bao gồm tuổi tác, giới tính, giai cấp xã hội, v.v., cùng với số phận của họ (sống sót hay thiệt mạng) trong vụ đắm tàu.

Bộ dữ liệu bao gồm hơn 890 bản ghi, mỗi bản ghi chứa thông tin về một hành khách trên tàu Titanic. Các thuộc tính chính của mỗi bản ghi bao gồm:

* PassengerId: Mã định danh duy nhất cho mỗi hành khách
* Survived: Biến mục tiêu cho biết hành khách có sống sót hay không (1: sống sót, 0: thiệt mạng)
* Pclass: Giai cấp xã hội của hành khách (1: thượng lưu, 2: trung lưu, 3: hạ lưu)
* Name: Tên của hành khách
* Sex: Giới tính của hành khách (male: nam, female: nữ)
* Age: Tuổi của hành khách
* Sibsp: Số lượng anh chị em ruột hoặc vợ/chồng đi cùng
* Parch: Số lượng con đi cùng
* Ticket: Số vé tàu
* Fare: Giá vé tàu
* Cabin: Số hiệu cabin (nếu có)
* Embarked: Cảng mà hành khách lên tàu (C: Cherbourg, Q: Queenstown, S: Southampton)

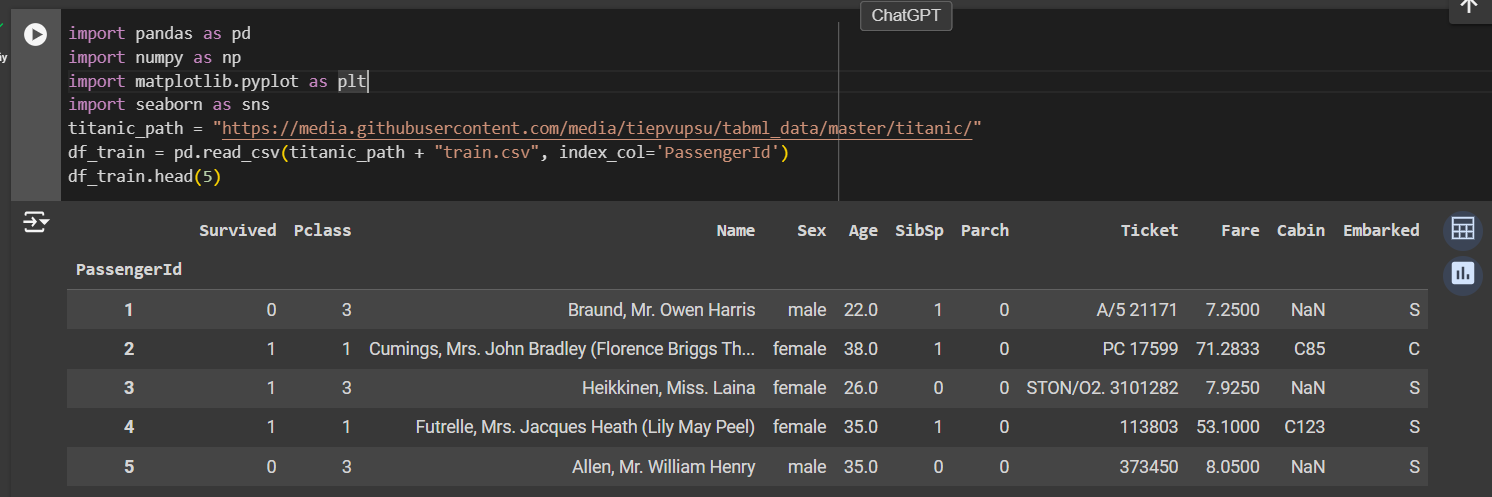
**Sử dụng:**

Bộ dữ liệu Titanic thường được sử dụng trong các bài toán học máy để: Dự đoán khả năng sống sót của hành khách

# Chuẩn bị dữ liệu

## 3.1. Load dữ liệu và EDA

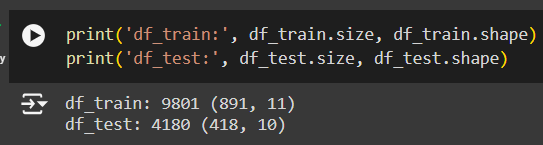
Đầu tiên, cùng xem nhanh dữ liệu trong các file này bằng cách hiển thị các dòng đầu tiên của mỗi file bằng phương thức head() trong pandas.



*Hình 3.1: 5 mẫu đầu của df\_train*



*Hình 3.2: 5 mẫu đầu của df\_test*



*Hình 3.3: Kích thước, số lượng hàng, số lượng cột của df\_train, df\_test*

Nhận xét:

* File train.csv và test.csv có tập hợp các cột với tên gần như nhau, ngoài trừ việc cột "Survived" không xuất hiện ở file test.csv. Bài toán đặt ra là dùng các cột còn lại của file train.csv để huấn luyện một mô hình sao cho nó có thể dự đoán được cột "Survived" này dựa trên những cột của file test.csv.
* Tập train có 891 mẫu dữ liệu và 11 cột thuộc tính.
* Tập test có 418 mẫu dữ liệu và 10 cột thuộc tính.

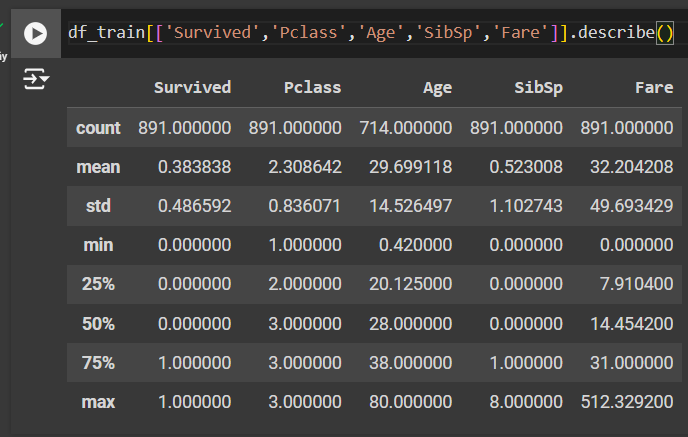
**Thống kê mô tả:**

|  |  |
| --- | --- |

*Hình 3.4: info và nunique của df\_train*

Nhận xét:

* Trong hình bên phải, khi dùng hàm .info(), cho ta biết tập train có 11 trường dữ liệu (trong đó có 2 cột dữ liệu có kiểu là float64, 4 cột dữ liệu có kiểu là là int64, 5 cột dữ liệu có kiểu là object). Ngoài ra, hầu như các cột dữ liệu đều không chứa các mẫu null, ngoại trừ các cột Age, Cabin, Embarked lần lượt chứa: 177, 687 và 2 mẫu dữ liệu bị null.
* Trong hình bên trái, các cột Pclass, Sex, Embarked có số lượng giá trị duy nhất ít, nên chúng là dữ liệu dạng phân loại (Categorical). Các cột như Age, Fare có số lượng giá trị duy nhất khá cao, cho thấy đây là dữ liệu dạng liên tục (Continuous data). Cột Name không có giá trị trùng lặp, điều này hợp lý vì mỗi hành khách có một tên riêng biệt.

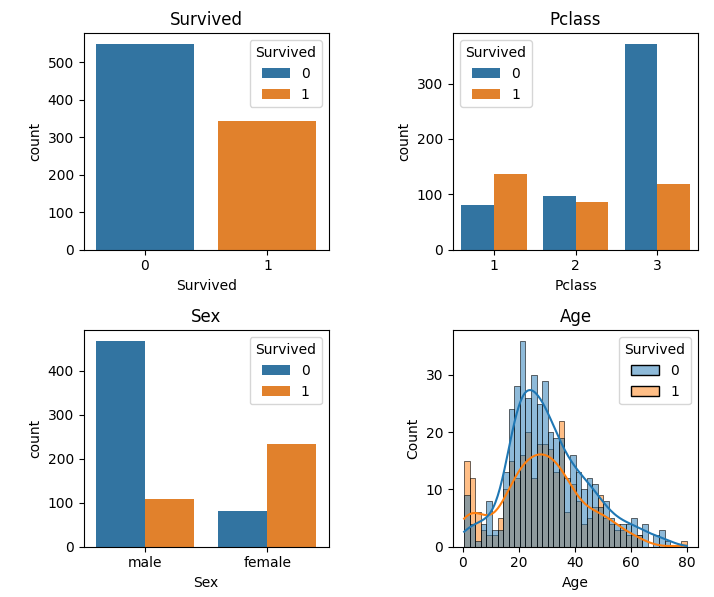


*Hình 3.5: Thống kê mô tả của df\_train*

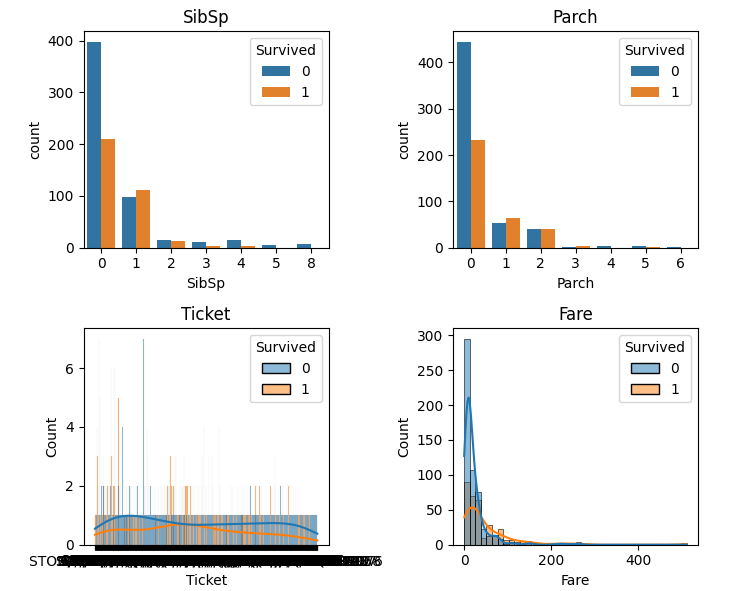
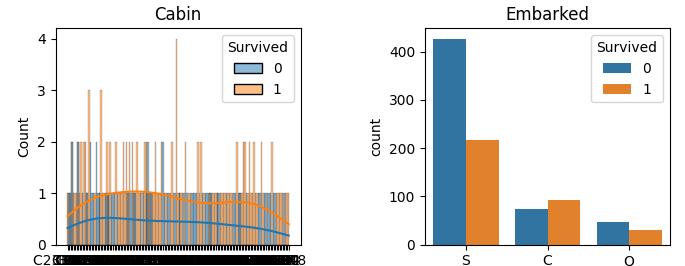
Nhận xét:

* Với cột Survived, giá trị trung bình trong cột là 0.384. Đây là cột nhãn mà mô hình cần dự đoán. Cột này chỉ mang các giá trị 0 và 1 nên ta có thể nói rằng 38.4% giá trị trong cột bằng 1. Có thể thấy, dữ liệu này cũng không quá mất cân bằng.
* Với cột Age, ta thấy rằng count = 714 và nhỏ hơn số lượng phần từ ở các cột còn lại (891). Việc này chứng tỏ có tới 891 - 714 = 177 mẫu dữ liệu có Age bị khuyết. Người nhỏ nhất trên tàu mới chỉ 0.42 tuổi, trong khi người nhiều tuổi nhất đã 80.
* Với cột Sibsp, số lượng anh chị em hoặc vợ/chồng nhiều nhất với một hành khách là 8, nhưng có tới 75% số hành khách có nhiều nhất 1 anh chị em hoặc vợ/chồng đi cùng. Việc này chứng tỏ phân bố của dữ liệu này khá lệch.
* Cột Fare cũng khá lệch khi trung binh là 32 trong khi trung vị chỉ là 14 và giá trị lớn nhất lên tới 512. Những hành khách với giá vé bằng 0 khả năng nằm trong thủy thủ đoàn.

Tiếp theo, sẽ trực quan hóa dữ liệu ở từng cột theo trường Survived xem thử các cột dữ liệu này có ảnh hưởng đến khả năng sống sót của hành khách hay không:



*Hình 3.6: Histplot của Pclass, Age với hue là Survived; Countplot của Survived; Countplot của Sex với hue là Survived*



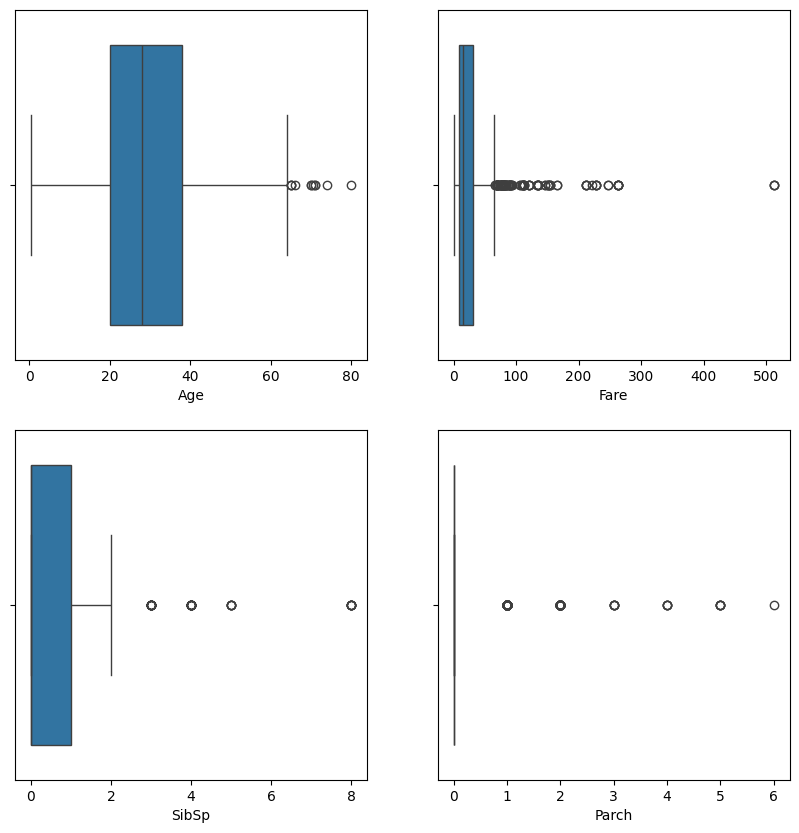
*Hình 3.7: Countplot của Embarked, SibSp, Parch với hue là Survived; Histplot của Fare với hue là Survived*

Nhận xét: ta có thể đưa ra một số nhận xét nhanh như sau

* Đồ thị của PClass cho thấy hành khách có ticket class thứ 3 có tỷ lệ sống sót thấp đáng kể so với 2 ticket class còn lại
* Đồ thị của Sex cho thấy tỷ lệ sống sót trong nữ giới cao hơn nam giới.
* Đồ thị của Age cho thấy tỷ lệ sống sót trong những người có độ tuổi từ 10 tuổi trở xuống sẽ cao hơn so với những người có độ tuổi > 10.
* Đồ thị Embarked cho thấy tỷ lệ sống sót trong cảng Cherbourg cao hơn so với các cảng còn lại
* Đồ thị SibSp cho thấy tỷ lệ sống sót và tỷ lệ tử gần bằng nhau trong trường hợp giá trị bằng 1 hoặc 2, và tỷ lệ sống sót thấp đáng kể so với tỷ lệ tử trong trường hợp giá trị khác.
* Đồ thị Parch cho thấy tỷ lệ sống sót và tỷ lệ tử gần bằng nhau trong trường hợp giá trị bằng 1, 2, 3, 5; tỷ lệ sống thấp đáng kể so với tỷ lệ tử đối trong các trường hợp còn lại.
* Đồ thị của Fare cho thấy giá vé trong khoảng 50 - 200 sẽ có tỷ lệ sống cao hơn.

## 3.2. Phát hiện ngoại lai (Outlier)

Đầu tiên, nhóm sẽ trực quan hóa outlier của các đặc trưng số bằng boxplot:



*Hình 3.8: Boxplot của Age, Fare, SibSP, Parch*

Có thể thấy cả 4 đặc trưng này đều tồn tại outliers. Tiếp theo nhóm sẽ sử dụng Local Outlier Factor với 20 hàng xóm để tự động phát hiện outlier, sau đó bỏ đi những hàng outlier này với đoạn code sau:

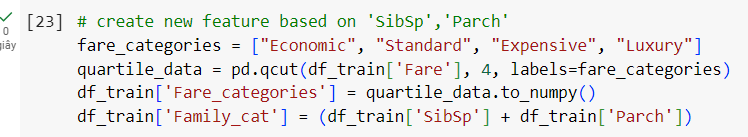


*Hình 3.9: Code loại bỏ outlier*

Sau khi loại bỏ các outliers, df\_train chỉ còn 813 hàng.

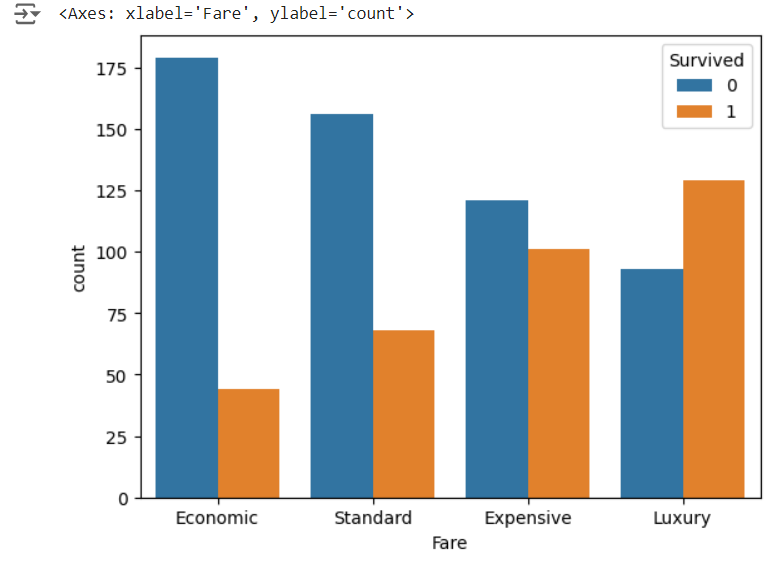
## 3.3. Rút trích đặc trưng (Feature engineering)

Tạo đặc trưng mới từ các đặc trưng trước đó. Ở đây, chúng ta tạo rời rạc hóa đặc trưng Fare\_categories bằng cách phân giá vé ra 4 loại; đặc trưng Famili\_cat bằng cách tính tổng của 2 đặc trưng SibSp, Parch. Đây là đoạn code thực hiện:



*Hình 3.10: Code thêm đặc trưng Fare\_categories và Famili\_cat*

Sau đó, ta sẽ visualize histogram của Fare với hue là Survived:

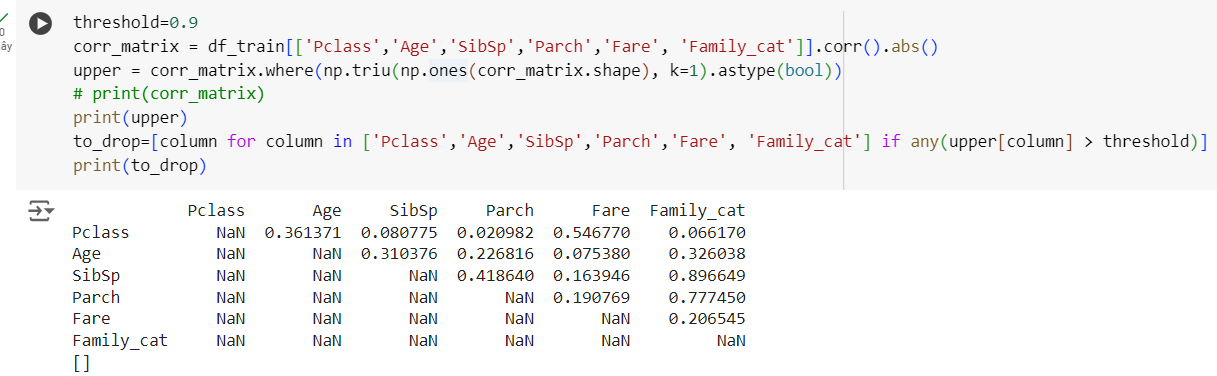


*Hình 3.11: Countplot của Fare\_categories với hue là Survived*

Biểu đồ trên cho biết tỷ lệ sống sót của loại vé Luxury cao hơn so với các loại vé khác. Đây có thể là một đặc trưng có ảnh hưởng tốt đến dự đoán của mô hình.

## 3.4. Chọn lựa đặc trưng (Feature selection)

Trong phần này, nhóm sẽ lọc bỏ các đặc trưng có độ tương quan cao (corr > 0.9). Trước hết chúng ta cần tính corr\_matrix và xác định các cột có độ tương tương quan vượt quá ngưỡng như đoạn code sau:

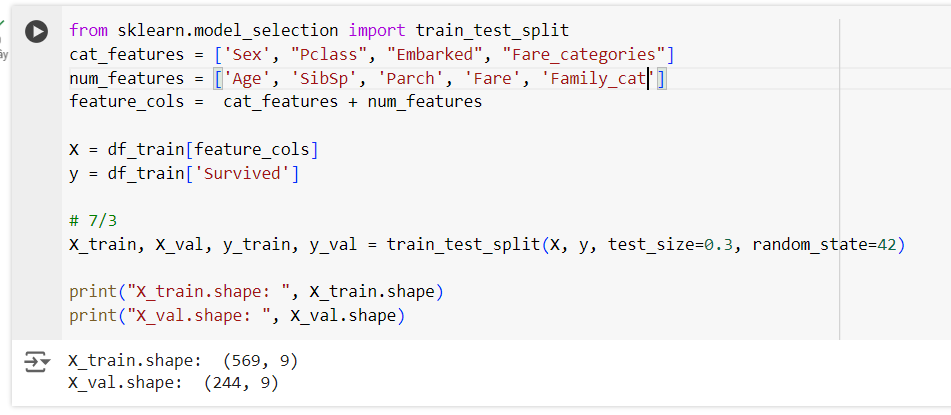


*Hình 3.12: Code xác định các cột có độ tương quan cao và kết quả thực hiện*

Không có cặp đặc trưng nào có độ trương quan cao hơn 0.9, vì vậy ta sẽ không loại bỏ cột nào cả.

## 3.5. Train test split

Ở đây, nhóm xác định biến phụ thuộc là biến ‘Survived’, các biến còn lại là độc lập. Sau đó chia tập train thành tập hai tập là train và val theo tỷ lệ 70/30.



*Hình 3.13: Code train test split*

Kết quả thu được tập hai tập:

* X\_train có kích thước 569 mẫu và 9 cột (không tính cột target)
* X\_val có kích thước 244 mẫu và 9 cột.

## 3.6. Chuyển đổi dữ liệu (Data transformation)

Trước đó, nhóm có định nghĩa:

* Các cột dữ liệu có kiểu là Categorical là: 'Sex', "Pclass", "Embarked", "Fare\_categories"
* Các cột dữ liệu có kiểu là Numerical là: 'Age', 'SibSp', 'Parch', 'Fare', 'Family\_cat'

Sau đó, tương ứng với các cột dữ liệu có kiểu Categorical và Numerical thì thực hiện các bước chuyển hóa dữ liệu như sau:

* Đối với dữ liệu có kiểu là Categorical: dùng SimpleImputer(strategy='most\_frequent'): Điền các giá trị thiếu bằng giá trị xuất hiện nhiều nhất trong các cột. Sau đó dùng OrdinalEncoder(): Mã hóa các đặc trưng phân loại thành các giá trị số theo thứ tự. (Các dòng bị comment có thể là các tùy chọn thay thế cho việc mã hóa đặc trưng phân loại).
* Đối với dữ liệu có kiểu là Numerical: dùng SimpleImputer(strategy='mean'): Điền các giá trị thiếu bằng giá trị trung bình của các cột. Sau đó dùng StandardScaler(): Chuẩn hóa các đặc trưng số bằng cách trừ đi trung bình và chia cho độ lệch chuẩn để có phân phối với trung bình 0 và độ lệch chuẩn 1.

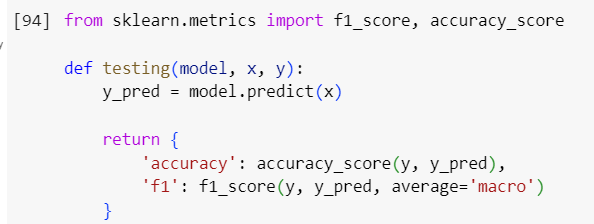


*Hình 3.14: Code chuyển đổi dữ liệu*

# Thuật toán phân loại và đánh giá mô hình

Để thực nghiệm một cách nhanh chóng và đơn giản, nhóm định nghĩa hàm testing(model, x, y) thực hiện việc kiểm tra mô hình. Thực hiện bằng cách:

* model.predict(x) dùng mô hình đã huấn luyện để dự đoán nhãn cho tập dữ liệu x.
* Tính độ chính xác của dự đoán bằng cách so sánh y\_pred với y (nhãn thực tế) thông qua hàm accuracy, f1\_score.
* Trả về kết quả dưới dạng dictionary với khóa 'accuracy' và ‘f1’

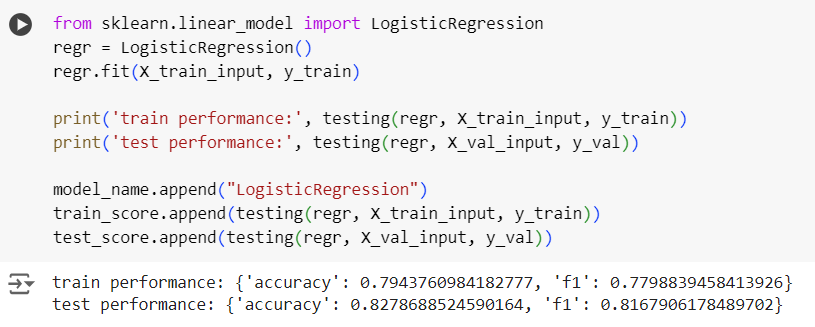


*Hình 3.15: Code testing*

## 4.1. LogisticRegression

**Giới thiệu ngắn gọn:** Mô hình Logistic Regression là một mô hình học máy được sử dụng để giải quyết các bài toán phân loại nhị phân (binary classification). Thay vì dự đoán một giá trị liên tục, Logistic Regression dự đoán xác suất của một mẫu thuộc về một trong hai lớp và sử dụng ngưỡng xác suất (thường là 0.5) để phân loại mẫu đó.

**Code và kết quả:**

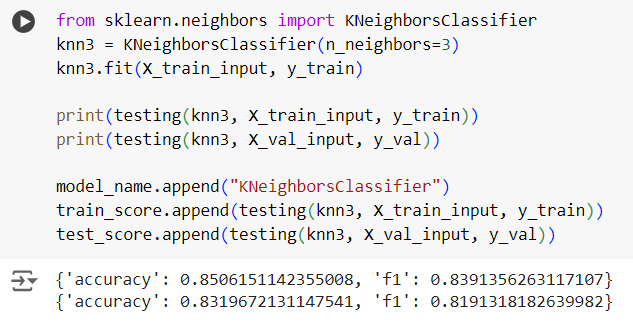
****

*Hình 3.16: Performance của logistic regression*

## 4.2. KNeighborsClassifier

**Giới thiệu ngắn gọn:** KNeighborsClassifier là một thuật toán học máy được sử dụng cho các bài toán phân loại. Đây là một trong những mô hình đơn giản và phổ biến nhất thuộc nhóm k-Nearest Neighbors (k-NN). KNeighborsClassifier là một mô hình phân loại dựa trên nguyên tắc "đa số láng giềng gần nhất".

**Code và kết quả:**

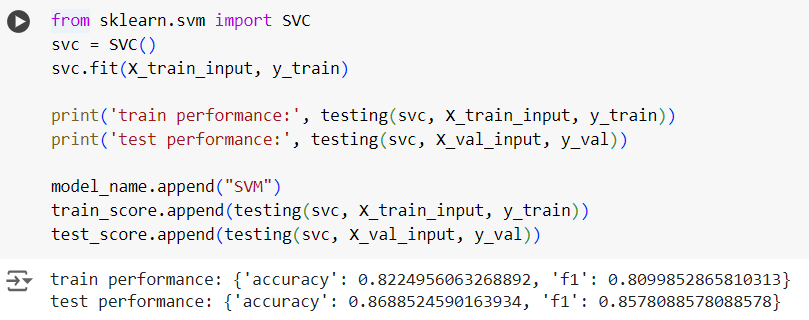
****

*Hình 3.17: Performance của KNN*

## 4.3. SVM

**Giới thiệu ngắn gọn:** SVM (viết tắt của Support Vector Machine) là một thuật toán học máy thuộc nhóm Supervised Learning (học có giám sát) được sử dụng trong các bài toán phân lớp dữ liệu (classification) hay hồi quy (regression).

**Code và kết quả:**

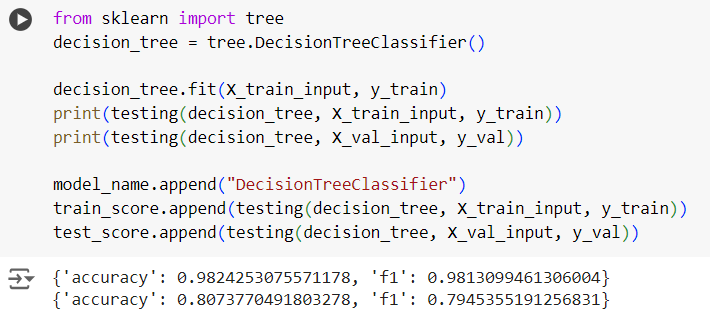
****

*Hình 3.18: Performance của SVM với kernel ‘rbf’*

## 4.4. DecisionTreeClassifier

**Giới thiệu ngắn gọn:** DecisionTreeClassifier là một thuật toán học máy dùng cho các bài toán phân loại, hoạt động bằng cách tạo ra một cây quyết định từ dữ liệu huấn luyện. Mô hình này chia dữ liệu thành các nhánh dựa trên các thuộc tính, tại mỗi nút, nó chọn thuộc tính phân chia dữ liệu tốt nhất dựa trên các tiêu chí như Gini impurity hoặc entropy. Cây tiếp tục phân chia đến khi đạt đến các lá, mỗi lá đại diện cho một nhãn phân loại. DecisionTreeClassifier dễ hiểu và trực quan, nhưng có thể dễ bị overfitting, đặc biệt khi cây trở nên quá sâu và phức tạp.

**Code và kết quả:**

****

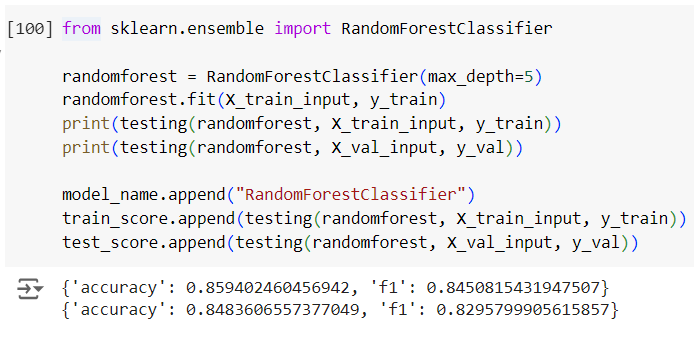
*Hình 3.19: Performance của Decision Tree*

Nhận xét: Mô hình có đang bị Overfitting → thay đổi độ sâu tối đa của cây

## 4.5. RandomForestClassifier

**Giới thiệu ngắn gọn:** RandomForestClassifier là một mô hình máy học phân lớp, hoạt động bằng cách xây dựng nhiều cây quyết định độc lập từ các mẫu ngẫu nhiên của tập dữ liệu huấn luyện và dự đoán bằng cách lấy kết quả dự đoán của từng cây và chọn nhãn dự đoán theo số đông (majority vote). Việc sử dụng nhiều cây giúp giảm thiểu hiện tượng overfitting, cải thiện độ chính xác và khả năng tổng quát hóa của mô hình. Random Forest đặc biệt hiệu quả trong việc xử lý các tập dữ liệu lớn và phức tạp, đồng thời có thể cung cấp các chỉ số quan trọng (feature importance) để hiểu được mức độ ảnh hưởng của các đặc trưng đầu vào.

**Code và kết quả:**

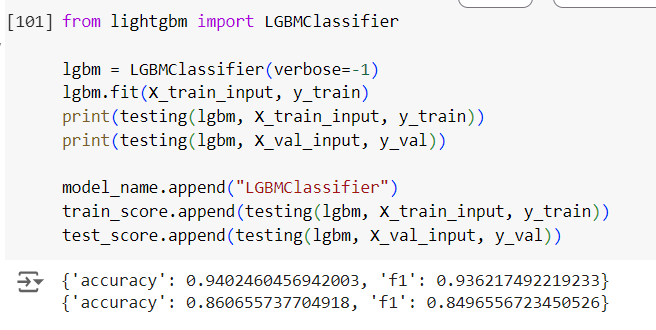
****

*Hình 3.20: Performance của Random Forest*

## 4.6. LGBM

**Giới thiệu ngắn gọn:** LightGBM (Light Gradient Boosting Machine) là một khung công tác học máy được thiết kế để cung cấp tốc độ và hiệu suất cao cho các bài toán phân loại và hồi quy. LightGBM là một biến thể của Gradient Boosting Decision Trees (GBDT), tối ưu hóa việc xây dựng cây quyết định để tăng tốc độ huấn luyện và giảm mức tiêu thụ bộ nhớ, đặc biệt là với các tập dữ liệu lớn. LightGBM sử dụng các kỹ thuật như histogram-based decision tree learning, leaf-wise tree growth và các thuật toán tối ưu để tăng tốc độ huấn luyện và dự đoán. Nó đặc biệt hiệu quả trong các bài toán yêu cầu huấn luyện mô hình trên tập dữ liệu lớn với nhiều đặc trưng và có khả năng xử lý cả dữ liệu phân loại và dữ liệu hồi quy.

**Code và kết quả:**



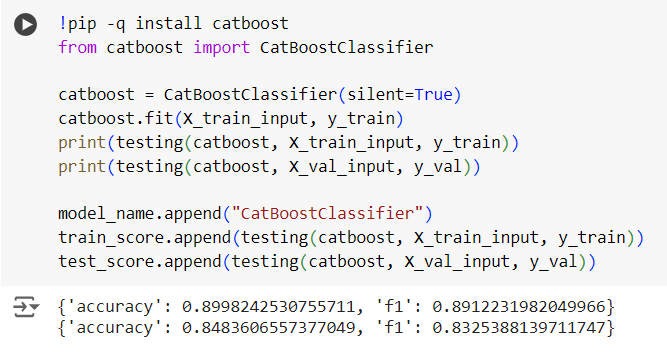
*Hình 3.21: Performance của Light GBM*

Nhận xét: Mô hình đang bi Overfitting → Chỉnh max\_depth

## 4.7. CatBoost

**Giới thiệu ngắn gọn:** CatBoost (Categorical Boosting) là một thuật toán học máy mạnh mẽ được phát triển bởi Yandex, được thiết kế đặc biệt để xử lý các tập dữ liệu có nhiều đặc trưng phân loại (categorical features). CatBoost là một mô hình máy học nhanh và mạnh mẽ, chuyên xử lý tốt các đặc trưng phân loại và cung cấp độ chính xác cao cho cả bài toán phân loại và hồi quy. CatBoost sử dụng các kỹ thuật như xử lý đặc trưng phân loại hiệu quả, tránh overfitting, và hỗ trợ song song hóa để tối ưu hóa quá trình huấn luyện.

**Code và kết quả:**

****

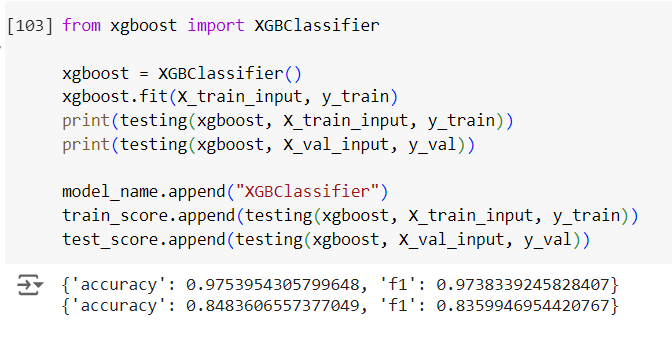
*Hình 3.22: Performance của CatBoost*

Nhận xét: Mô hình đang bi Overfitting → chỉnh max\_depth của cây

## 4.8. XGBoost

**Giới thiệu ngắn gọn:** XGBoost (Extreme Gradient Boosting) là một thư viện học máy mạnh mẽ và hiệu quả, được thiết kế để cải thiện hiệu suất và tốc độ của các thuật toán Gradient Boosting Decision Trees (GBDT). XGBoost nổi bật với khả năng xử lý các tập dữ liệu lớn, hỗ trợ phân tán và song song hóa, và thường đạt được kết quả xuất sắc trong các cuộc thi học máy và các ứng dụng thực tế.

**Code và kết quả:**

****

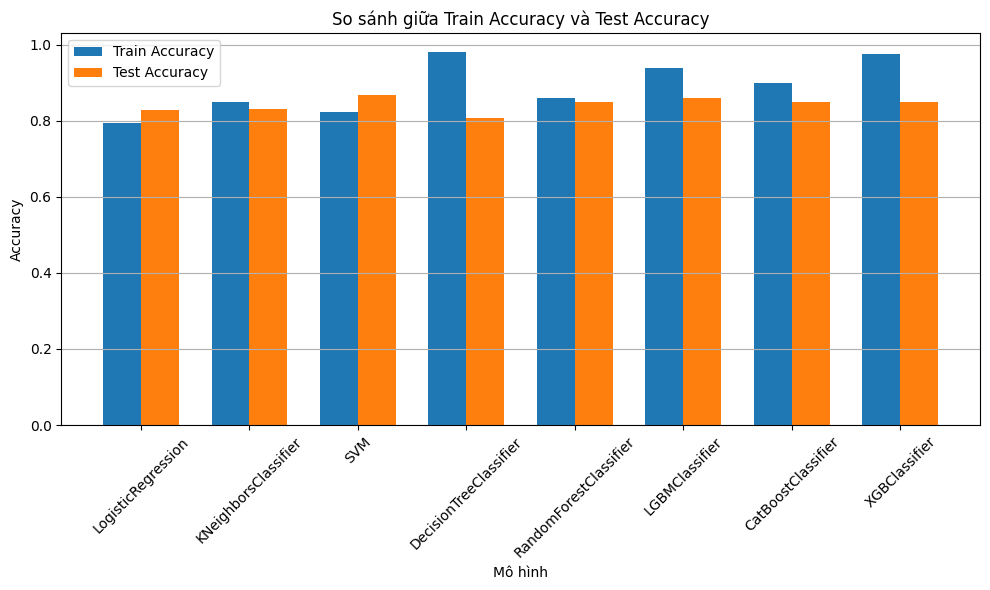
*Hình 3.23: Performance của XGBoost*

Nhận xét: Mô hình đang bị Overfitting nặng → Chỉnh max\_depth của cây

# Kết luận

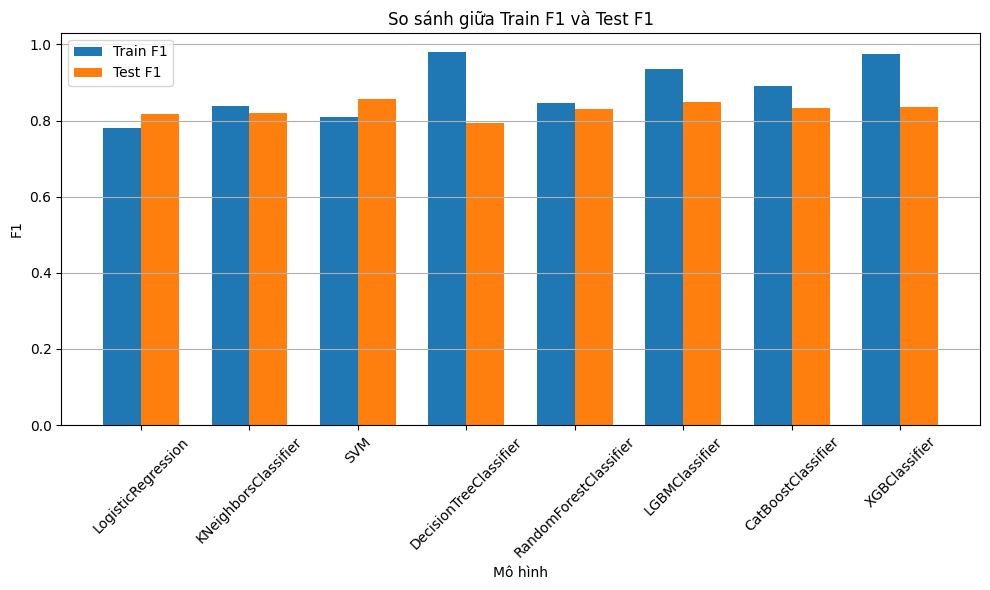
Từ kết quả của từng mô hình trên, nhóm giữ lại để vẽ biểu đồ trực quan hiệu suất của từng mô hình trên tập train và tập test:

Biểu đồ thể hiện hiệu suất tính theo Accuracy giữa các mô hình:



*Hình 3.24: So sánh giữa train accuracy và test accuracy*

Biểu đồ thể hiện hiệu suất tính theo F1-score giữa các mô hình:



*Hình 3.25: So sánh giữa train f1 và test f1*

Nhận xét:

* Mô hình DecisionTreeClassifier và XGBClassifier cùng đạt hiệu suất tốt nhất trên tập train cho cả Accuracy và F1-score.
* Mô hình SVM đạt hiệu suất tốt nhất trên test cho cả Accuracy và F1-score.
* Trên tập train, mô hình LogisticRegression đạt hiệu suất thấp nhất cho Accuracy và F1-score.
* Trên tập test, mô hình DecisionTreeClassifier đạt hiệu suất thấp nhất cho Accuracy và F1-score.
* Các mô hình bị Overfitting thường là các mô hình có sử dụng cấu trúc cây vì nhóm chưa hiệu chỉnh max\_depth phù hợp nhất cho các mô hình này, có thể kể đến như: DecisionTreeClassifier, RandomForestClassifier, LGBMClassifier, CatBoostClassifier, XGBClassifier.

Trên đây là toàn bộ quá trình tụi em thực nghiệm và học cách sử dụng 8 loại mô hình phân lớp thường được sử dụng. Qua bài báo cáo này giúp chúng em hiểu sâu hơn về cách xử lý dữ liệu cũng như tiếp cận và học hỏi cách sử dụng các mô hình phân lớp thường được sử dụng trong Học Máy.