

Họ và tên: Nguyễn Đăng Khoa

MSSV : 3123410169

Báo cáo Lab_3: Bài toán Titanic

1. Giới thiệu (Introduction)

1.1. Bối cảnh và cơ sở dữ liệu Titanic

Vụ đắm tàu Titanic năm 1912 đã trở thành một trong những thảm họa hàng hải lớn nhất vào đầu thế kỉ XX cũng như trong lịch sử, gây ra cái chết của hơn 1500 hành khách và thủy thủ trên tàu. Đáng chú ý, khả năng sống sót của các hành khách không phải ngẫu nhiên mà chịu ảnh hưởng từ nhiều yếu tố như độ tuổi, giới tính, hạng vé, ... Bộ dữ liệu về Titanic được công bố trên Kaggle đã trở thành một trong những bộ dữ liệu kinh điển, thường được sử dụng thực hành các kỹ thuật học máy và phân tích dữ liệu.

1.2. Mục tiêu nghiên cứu và ý nghĩa thực tiễn

Mục tiêu của nghiên cứu này là có thể xây dựng được một mô hình máy học có khả năng dự đoán xác suất sống sót của một hành khách dựa vào các đặc trưng được cung cấp. Ý nghĩa bài toán này không chỉ dừng lại ở việc minh họa khả năng máy học trong phân loại nhị phân, mà còn giúp người học nắm vững quy trình chuẩn: từ tiền xử lý dữ liệu, lựa chọn đặc trưng, huấn luyện mô hình, đánh giá hiệu năng, cho tới triển khai dự đoán trên dữ liệu mới.

1.3. Phát biểu bài toán học máy

Bài toán Titanic có thể được phát biểu như bài toán phân loại nhị phân trong máy học. Với đầu vào là 1 vector $X = \{x_1, x_2, x_3, ..., x_n\}$ mô tả các thông tin cơ bản của một hành khách (tuổi, giới tính, hạng vé, ...), mục tiêu cần mô hình thực hiện là dự đoán đầu ra $y \in \{0; 1\}$, trong đó:

- $y = 0$: hành khách không sống sót
- $y = 1$: hành khách sống sót.

Nhiệm vụ của bài toán máy học là có thể tìm ra một hàm $f: X \rightarrow y$ có khả năng dự đoán chính xác y cho những hành khách chưa từng được quan sát.

2. Tập dữ liệu và mô tả (Dataset Description)

2.1. Nguồn dữ liệu

Bộ dữ liệu sử dụng cho bài toán này được cung cấp bởi Kaggle trong cuộc thi “Titanic: Machine Learning from Disaster”. Dữ liệu gồm hai tập chính:

- **train.csv** : dữ liệu dùng cho huấn luyện, cung cấp các thông tin của khách hàng và cột đánh giá Survived (sống sót = 1, chết = 0).
- **test.csv** : dữ liệu dùng để kiểm tra, tập dữ liệu này chỉ bao gồm thông tin của khách hàng mà không có nhãn Survived. Nhiệm vụ của chúng ta là dự đoán nhãn này.

Ngoài ra, còn có tệp **gender_submission.csv**, đây là một ví dụ về format kết quả cần nộp.

2.2. Các thuộc tính trong bộ dữ liệu

Các cột trong dữ liệu huấn luyện gồm:

- *PassengerId*: số thứ tự hành khách (ID).
- *Survived*: biến mục tiêu (0 = chết, 1 = sống sót).
- *Pclass*: hạng vé (1, 2 hoặc 3).
- *Name*: tên hành khách.
- *Sex*: giới tính.
- *Age*: tuổi.
- *SibSp*: số lượng anh/chị/em hoặc vợ/chồng đi cùng.
- *Parch*: số lượng cha/mẹ hoặc con đi cùng.
- *Ticket*: số vé.
- *Fare*: giá vé.
- *Cabin*: số phòng.
- *Embarked*: cảng lên tàu (C = Cherbourg, Q = Queenstown, S = Southampton).

2.3. Phân tích sơ bộ dữ liệu

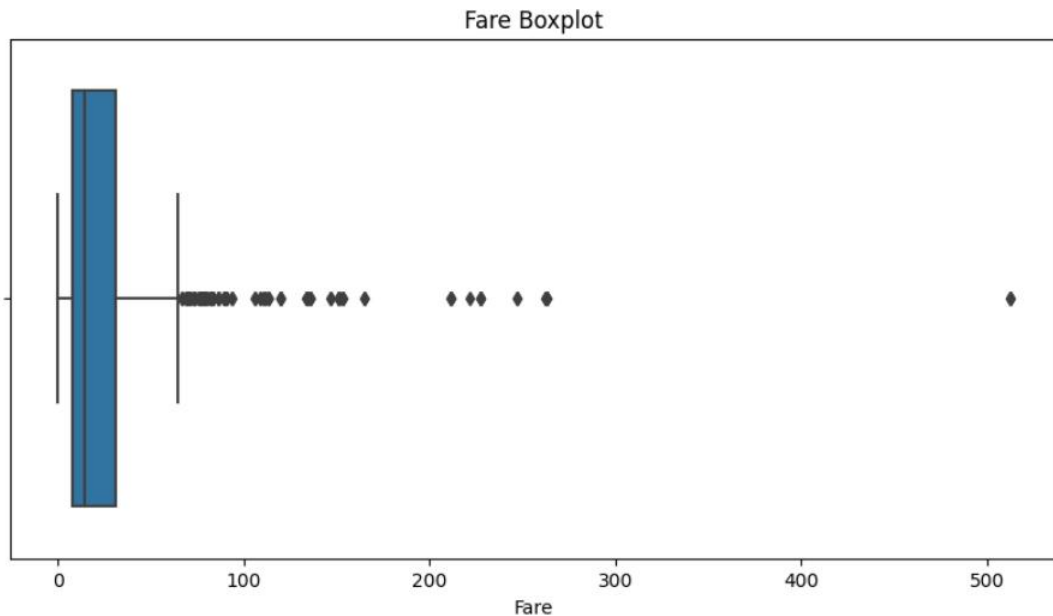
Khi xem qua dữ liệu sơ bộ ban đầu bằng ***train_data.info()*** và ***test_data.info()***, ta có thể nhận thấy ở một số cột vẫn còn thiếu thông tin, cụ thể là cột *Age*, *Cabin* và *Embarked*.

In [4]:

```
train_data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 891 entries, 0 to 890
Data columns (total 12 columns):
#   Column      Non-Null Count  Dtype  
---  -
0   PassengerId  891 non-null    int64  
1   Survived     891 non-null    int64  
2   Pclass       891 non-null    int64  
3   Name         891 non-null    object  
4   Sex          891 non-null    object  
5   Age          714 non-null    float64 
6   SibSp        891 non-null    int64  
7   Parch        891 non-null    int64  
8   Ticket       891 non-null    object  
9   Fare         891 non-null    float64 
10  Cabin        204 non-null    object  
11  Embarked     889 non-null    object  
dtypes: float64(2), int64(5), object(5)
```

Ngoài ra, nếu vẽ biểu đồ *boxplot* cho cột *Fare*, ta thấy dữ liệu có nhiều điểm ngoại lai (outliers). Điều này cho thấy cần biến đổi (ví dụ lấy log) để dữ liệu “cân đối” hơn cho mô hình.



3. Tiền xử lý dữ liệu (Data Preprocessing)

Trước khi bắt đầu huấn luyện mô hình, ta cần phải làm sạch và biến đổi dữ liệu để dễ dàng cho việc huấn luyện:

- Xử lý dữ liệu đang bị thiếu:

- Điền giá trị phổ biến nhất cho các vị trí trống trong cột *Embarked*.
- Điền giá trị trung bình của cột *Age* vào các ô còn thiếu.
- Cột *Cabin* còn thiếu quá nhiều dữ liệu nên bị loại bỏ.

```
train_data.isna().sum()
```

PassengerId	0
Survived	0
Pclass	0
Name	0
Sex	0
Age	177
SibSp	0
Parch	0
Ticket	0
Fare	0
Cabin	687
Embarked	2

dtype: int64

➔

```
train_data.isna().sum()
```

Survived	0
Pclass	0
Sex	0
Age	0
SibSp	0
Parch	0
Fare	0
Embarked_Q	0
Embarked_S	0

dtype: int64

- Nhận thấy cột *Fare* trong *test.csv* còn thiếu dữ liệu nên điền giá trị trung bình vào các ô này.

- Loại bỏ những cột không cần thiết:

PassengerId, *Name*, *Ticket* không mang lại quá nhiều ý nghĩa cho mô hình nên bị loại bỏ.

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	S
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	C
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	S
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C123	S
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	NaN	S

	Survived	Pclass	Sex	Age	SibSp	Parch	Fare	Embarked
0	0	3	male	22.0	1	0	7.2500	S
1	1	1	female	38.0	1	0	71.2833	C
2	1	3	female	26.0	0	0	7.9250	S
3	1	1	female	35.0	1	0	53.1000	S
4	0	3	male	35.0	0	0	8.0500	S

- Mã hóa các biến phân loại:

- Thay đổi giá trị trong cột *Sex* thành: 0 là *male* và 1 là *female*.
- Cột *Emberked* sẽ dùng *one-hot encoding*.

	Survived	Pclass	Sex	Age	SibSp	Parch	Fare	Embarked
0	0	3	male	22.000000	1	0	7.2500	S
1	1	1	female	38.000000	1	0	71.2833	C
2	1	3	female	26.000000	0	0	7.9250	S
3	1	1	female	35.000000	1	0	53.1000	S
4	0	3	male	35.000000	0	0	8.0500	S

	Survived	Pclass	Sex	Age	SibSp	Parch	Fare	Embarked_Q	Embarked_S
0	0	3	0	-0.592481	1	0	-0.879741	False	True
1	1	1	1	0.638789	1	0	1.361220	False	False
2	1	3	1	-0.284663	0	0	-0.798540	False	True
3	1	1	1	0.407926	1	0	1.062038	False	True
4	0	3	0	0.407926	0	0	-0.784179	False	True

- Chuẩn hóa dữ liệu số:
 - Các giá trị của *Age* và *Fare* có sự chênh lệch về khoảng khá lớn sẽ làm cho việc huấn luyện bị chi phối không còn chính xác nên cần dùng *StandardScaler* để chuẩn hóa đưa về cùng thang đo.

```
train_data.head()
test_data.head()
```

	Pclass	Sex	Age	SibSp	Parch	Fare	Embarked_Q	Embarked_S
0	3	0	0.369449	0	0	-0.809683	True	False
1	3	1	1.331378	1	0	-0.911513	False	True
2	2	0	2.485693	0	0	-0.612461	True	False
3	3	0	-0.207709	0	0	-0.716562	False	True
4	3	1	-0.592481	1	1	-0.387631	False	True

4. Phương pháp và mô hình hóa (Methodology and Models)

Trong bài toán này, ta sẽ thử với ba mô hình phổ biến để so sánh hiệu quả huấn luyện:

- **Logistic Regression:**

Mô hình cơ bản cho phân loại nhị phân. Dễ triển khai, chạy nhanh, thường được dùng làm baseline.

- **Random Forest:**

Mô hình tập hợp nhiều cây quyết định (decision trees). Khả năng dự đoán khá tốt, ít bị overfitting, phù hợp với dữ liệu có nhiều loại đặc trưng.

- **XGBoost:**

Mô hình boosting mạnh mẽ, thường đạt kết quả cao trong nhiều bài toán Kaggle. Tuy nhiên cần tinh chỉnh nhiều siêu tham số hơn.

Các mô hình được huấn luyện trên tập train (80%) và kiểm tra trên tập validation (20%). Các thước đo để so sánh gồm *Accuracy*, *F1 Score*, *ROC-AUC*.

5. Thực nghiệm và kết quả (Experiments and Results)

Cách thực hiện:

- Dữ liệu được chia thành 80% huấn luyện và 20% kiểm tra (*validation*).
- Mỗi mô hình được huấn luyện trên tập train và dự đoán trên tập validation.

- Kết quả được đánh giá bằng 3 chỉ số: *Accuracy*, *F1 Score*, *ROC-AUC*.

Kết quả:

Logistic Regression Results:

Accuracy: 0.8101

F1 Score: 0.7385

ROC AUC: 0.8387

Random Forest Results:

Accuracy: 0.8212

F1 Score: 0.7538

ROC AUC: 0.8345

XGBoost Results:

Accuracy: 0.7989

F1 Score: 0.7353

ROC AUC: 0.8080

Đánh giá:

- *Logictis*: *Accuracy* khá cao, *ROC AUC* có thể phân biệt sống/ chết khá tốt, *F1_Score* cân bằng tốt giữa *precision* và *recall*.
- *Random Forest*: *Accuracy* và *F1_Score* cao nhất trong 3 mô hình, *ROC AUC* thấp hơn *Logictis* một chút nhưng vẫn khá cao.
- *XGBoost*: *Accuracy* và *ROC AUC* thấp nhất trong 3 mô hình, tức là khả năng phân biệt chưa bằng 2 mô hình trên, *F1_Score* cao hơn *Logictis* 1 chút nhưng vẫn kém *Random Forest*.

⇒ ***Random Forest*** là mô hình tốt nhất trong 3 mô hình trên.

6. Dự đoán và nộp kết quả (Prediction and Submission)

Sau khi so sánh ta chọn mô hình *Random Forest* làm mô hình cuối cùng vì đạt được kết quả cao nhất.

Sử dụng mô hình *Random Forest* để thực hiện dự đoán dữ liệu trong file *test.csv*.

Kết quả được lưu thành file *submission.csv* gồm:

- *PassengerId*.
- *Survived* (giá trị dự đoán).


submission.csv (2.84 kB)

PassengerId	Survived
892	0
893	0
894	0
895	1
896	0
897	0
898	1
899	0
900	1
901	0

Sau khi nộp bài lên *Kaggle* đã được hệ thống chấm điểm trực tiếp với điểm số và thứ hạng tương ứng.

11735


Nguyễn Đăng Khoa



0.74880

1

2d



Your First Entry!
Welcome to the leaderboard! Your score represents your submission's accuracy. For example, a score of 0.7 in this competition indicates you predicted Titanic survival correctly for 70% of people.

What next? You've got a few options:

- 📖 Learn skills that can improve your score in our [Intro to Machine Learning course](#) by Dan Becker.
- 🗣️ Check out the [discussion forum](#) to find lots of tutorials and insights from other competitors.
- 🏆 Find a new challenge by entering one of our [open, active competitions](#) or searching our [public datasets](#).

7. Kết luận và hướng phát triển (Conclusion and Future Work).

Trong bài toán Titanic, em đã áp dụng các bước cơ bản của học máy: tiền xử lý dữ liệu, chọn đặc trưng, huấn luyện và so sánh mô hình. Kết quả cho thấy *Random Forest* cho độ chính xác cao nhất, được chọn làm mô hình cuối cùng để dự đoán.

Tuy nhiên, bài toán vẫn còn có thể cải thiện:

- Tạo thêm đặc trưng mới (ví dụ: nhóm gia đình, độ dài tên).
- Tối ưu siêu tham số (*Hyperparameter Tuning*).
- Kết hợp nhiều mô hình (*Ensemble/Stacking*).