BỘ GIÁO DỤC VÀ ĐÀO TẠO

TRƯỜNG ĐẠI HỌC SƯ PHẠM KỸ THUẬT TP.HCM

KHOA CÔNG NGHỆ THÔNG TIN

-----□□□□□-----

BÁO CÁO CUỐI KỲ

Môn học: Ứng dụng dữ liệu lớn : Học máy ở quy mô lớn

Mã môn học: BDML434077

Giáo viên hướng dẫn: Quách Đình Hoàng

Thành viên nhóm 2:

Nguyễn Thành An : 22133002

Dương Minh Hiếu : 22133018

Trương Trọng Đại Long : 22133033

Nguyễn Đức Cao Thăng : 22133053

TP.Hồ Chí Minh, 26 tháng 10, năm 2025

Mục lục

1. Tóm tắt (abstract)

Ngành hàng không Mỹ từ năm 1993 đến 2024 đã trải qua nhiều biến động do ảnh hưởng của kinh tế, công nghệ và cạnh tranh thị trường. Việc phân tích xu hướng giá vé, nhu cầu hành khách và hiệu suất tuyến bay là cần thiết để các hãng hàng không và nhà hoạch định chính sách đưa ra quyết định chiến lược. Bài thực hiện của nhóm xây dựng mô hình dự đoán giá vé và số lượng hành khách trung bình trên một chuyến bay, phân tích tác động của khoảng cách bay đến giá cả và nhu cầu, ước lượng thị phần và phân loại tuyến bay theo tiềm năng tăng trưởng.Nhóm nghiên cứu sử dụng các thuật toán học máy như Logistic Regression, Random Forest, Naive Bayes, Gradient Boosted Trees (GBT), và SVM. Kết quả cho thấy:

- Bài phân tích cho thấy khoảng cách bay ảnh hưởng rõ rệt đến giá vé và nhu cầu, trong khi việc phân loại tuyến bay giúp nhận diện tiềm năng thị trường.
- Mô hình dự đoán giá vé trung bình đạt $R^2 > 0.95$, mô hình dự đoán số hành khách trung bình đạt $R^2 > 0.97$.
- Mô hình phát hiện bất thường đạt AUC: 0.75, Accuracy: 0.76, Precision: 0.827, Recall: 0.7692, F1-Score: 0.71
- Nghiên cứu khẳng định rằng học máy có thể nắm bắt hiệu quả các động lực phức tạp trong ngành hàng không, hỗ trợ tối ưu hóa hoạt động và lập kế hoạch chiến lược.

2. Giới thiệu (introduction)

Trong hơn ba thập kỷ qua, ngành hàng không Hoa Kỳ đã phát triển mạnh mẽ, trở thành một trong những hệ thống vận tải lớn nhất, hiện đại nhất và năng động nhất thế giới. Với hàng nghìn chuyến bay mỗi ngày kết nối hàng trăm thành phố trên khắp lãnh thổ, hàng không không chỉ đóng vai trò trung tâm trong hệ thống giao thông quốc gia mà còn là yếu tố then chốt thúc đẩy thương mại, du lịch và hội nhập kinh tế. Kể từ đầu thập niên 1990, cùng với sự tự do hóa thị trường và sự tham gia của nhiều hãng hàng không mới, mức độ cạnh tranh trong ngành đã tăng lên đáng kể. Sự cạnh tranh này kéo theo những biến động phức tạp trong giá vé, lượng hành khách, và thị phần của các hãng, phản ánh mối quan hệ đa chiều giữa cung – cầu, chi phí vận hành và hành vi tiêu dùng của hành khách.

Song song đó, sự phát triển của công nghệ và dữ liệu lớn (Big Data) trong hai thập niên gần đây đã mở ra cơ hội mới cho việc phân tích, dự báo và giám sát hoạt động hàng không trên quy mô lớn. Các tổ chức hàng không, cơ quan quản lý và doanh nghiệp khai thác hiện nay không chỉ quan tâm đến việc dự báo xu hướng giá vé và hành khách, mà còn cần phát hiện kịp thời các dấu hiệu bất thường trong dữ liệu vận hành — chẳng hạn như các tuyến bay có giá vé tăng đột ngột, thị phần giảm mạnh,

hoặc lượng khách sụt giảm bất thường. Việc nhận diện sớm các bất thường này đóng vai trò quan trọng trong việc phòng ngừa rủi ro, tối ưu chiến lược giá và duy trì sự ổn định của thị trường.

Trong bối cảnh đó, mô hình phát hiện bất thường (Anomaly Detection) trở thành một hướng nghiên cứu có giá trị thực tiễn cao. Dữ liệu hàng không thường rất lớn, đa chiều và biến động mạnh theo thời gian, khiến việc phát hiện thủ công gần như không khả thi. Do đó, các thuật toán học máy và học sâu như *Isolation Forest*, *One-Class SVM* hay *Autoencoder Neural Network* được ứng dụng nhằm tự động phát hiện những mẫu dữ liệu có hành vi khác biệt so với xu hướng chung. Mô hình này không chỉ giúp phát hiện bất thường trong giá vé, hành khách hay thị phần mà còn hỗ trợ xây dựng hệ thống cảnh báo sớm cho các hãng hàng không và cơ quan điều hành.

Bộ dữ liệu nhóm sử dụng bao gồm thông tin chi tiết về các tuyến bay nội địa Hoa Kỳ giai đoạn 1993–2024, với các thuộc tính như: năm và quý của chuyến bay (Year, quarter), thành phố khởi hành và điểm đến (city1, city2), mã sân bay (airport_1, airport_2), khoảng cách bay (nsmiles), số lượng hành khách (passengers), giá vé trung bình (fare), thị phần của hãng lớn nhất và hãng có giá thấp nhất (large_ms, lf_ms), cùng các chỉ số liên quan đến hãng hàng không và giá vé (fare_lg, fare_low). Dữ liệu này cung cấp nền tảng phong phú cho việc phân tích xu hướng, phát hiện bất thường, đánh giá cạnh tranh và dự báo hành vi thị trường trong ngành hàng không.

Dựa trên bộ dữ liệu này, nhóm tiến hành một hệ thống phân tích và dự đoán đa hướng gồm sáu bài toán chính:

1. Phân tích xu hướng & dự báo giá vé, hành khách: Xác định xu hướng biến động và dự báo giá vé, lượng hành khách theo thời gian.
2. Ảnh hưởng của khoảng cách đến giá vé & hành khách: Phân tích mối quan hệ giữa độ dài tuyến bay (nsmiles) và các chỉ số như giá vé, lượng hành khách.
3. Phát hiện bất thường (Anomaly Detection): Xác định các tuyến bay hoặc thời kỳ có hành vi khác thường về giá vé, hành khách hoặc thị phần.
4. Phân tích thị phần & cạnh tranh giữa các hãng: So sánh thị phần, giá vé và tốc độ tăng trưởng của các hãng hàng không lớn qua các năm.
5. Phân loại tuyến bay theo tiềm năng phát triển: Phân nhóm các tuyến bay dựa trên các đặc trưng hành khách, giá vé và thị phần để xác định tiềm năng mở rộng.

Input của hệ thống là tập dữ liệu định lượng và định danh mô tả đặc trưng tuyến bay, thời gian, khoảng cách, giá vé, hành khách và thị phần.
Output của mô hình thay đổi tùy theo bài toán con, bao gồm:

- Giá vé dự báo (fare_pred) và số lượng hành khách dự báo (passengers_pred) trong tương lai;
- Nhãn bất thường (0 hoặc 1) cho các tuyến bay có dấu hiệu khác thường;
- Kết quả phân loại tuyến bay theo tiềm năng phát triển hoặc theo mức độ cạnh tranh giữa các hãng.

Việc triển khai đồng thời các bài toán trên giúp nhóm có cái nhìn toàn diện về hoạt động hàng không trong giai đoạn 1993–2024, từ đó đưa ra những phân tích sâu sắc về xu hướng thị trường, chiến lược giá và khả năng phát triển của từng tuyến bay. Kết quả nghiên cứu không chỉ mang ý nghĩa học thuật mà còn có giá trị ứng dụng thực tiễn trong quản lý, hoạch định và tối ưu hóa vận hành cho ngành hàng không Hoa Kỳ nói riêng và lĩnh vực vận tải nói chung.

3. Dữ liệu (data)

Tập dữ liệu được nhóm sử dụng cung cấp thông tin chi tiết về các tuyến bay nội địa tại Hoa Kỳ trong giai đoạn 1993–2024, bao gồm dữ liệu về giá vé, số lượng hành khách, khoảng cách bay và thông tin hãng hàng không.

Tập dữ liệu chứa thông tin cho hàng triệu tuyến bay theo từng quý trong suốt hơn ba thập kỷ, phản ánh sự thay đổi của thị trường hàng không theo thời gian. Mỗi bản ghi tương ứng với một tuyến bay giữa hai thành phố cụ thể trong một quý của một năm nhất định, bao gồm các nhóm đặc trưng chính sau:

- Thông tin định danh tuyến bay: city1, city2 (thành phố đi/đến), airport_1, airport_2 (mã sân bay), nsmiles (khoảng cách giữa hai sân bay, tính bằng dặm).
- Thông tin thời gian: Year (năm), quarter (quý).
- Chỉ số vận hành: passengers (số hành khách), fare (giá vé trung bình), fare_low, fare_lg (giá vé thấp nhất và giá vé của hãng lớn nhất), large_ms, lf_ms (thị phần của hãng lớn nhất và hãng có giá thấp nhất).
- Thông tin về hãng hàng không: carrier_lg (hãng có lượng khách lớn nhất), carrier_low (hãng có giá thấp nhất).
- Thông tin hỗ trợ: Geocoded_City1, Geocoded_City2 (tọa độ địa lý của thành phố đi/đến), tbl1apk (mã định danh duy nhất cho tuyến bay).

Sau khi thu thập, dữ liệu được nhóm làm sạch và tiền xử lý để đảm bảo chất lượng đầu vào cho các mô hình học máy. Cụ thể:

- Xử lý giá trị thiếu (missing values): Các bản ghi thiếu thông tin trọng yếu như fare, passengers hoặc nsmiles bị loại bỏ. Đối với một số trường hợp thiếu hãng

bay (carrier_lg, carrier_low), nhóm thực hiện điền giá trị "Unknown" để duy trì cấu trúc dữ liệu.

- Chuẩn hóa (normalization): Dữ liệu được chuẩn hóa bằng Min-Max Scaling (đưa về [0, 1]) và StandardScaler (chuẩn hóa z-score) để tăng tính ổn định và tốc độ hội tụ của mô hình.
- Xử lý ngoại lệ (outlier handling): Các giá trị bất thường về fare và passengers được kiểm tra, lọc bỏ hoặc đánh nhãn phục vụ cho bài toán phát hiện bất thường (anomaly detection).

4. PHƯƠNG PHÁP (METHODS)

4.1. Random Forest

Random Forest là thuật toán ensemble learning dựa trên bagging, kết hợp nhiều cây quyết định huấn luyện độc lập bằng bootstrap sampling. Ở mỗi nút, mô hình chỉ chọn một tập con đặc trưng ngẫu nhiên, giúp tăng độ chính xác và giảm overfitting

Dự đoán cuối cùng được tính bằng trung bình của tất cả các cây:

$$\widehat{y} = \frac{1}{n}\sum_{i=1}^{N} fi(x)$$

Hàm mục tiêu khi xây dựng mỗi cây là tối thiểu hóa phương sai (variance) tại mỗi nút:

$$\text{MSE} : \frac{1}{n}\sum_{i=1}^{n}\left(y_i - \overline{y}\right)^2$$

Phân loại cuối cùng :

$$\widehat{y} = mode\{h_1(x), h_2(x),..., h_T(x)\}$$

- với T: số lượng cây trong rừn
- $h_T(x)$: dự đoán của cây thứ t cho mẫu đầu vào x

4.2. Gradient Boosted Trees (GBT)

Gradient Boosting là thuật toán ensemble learning dựa trên kỹ thuật boosting, xây dựng các cây quyết định tuần tự, mỗi cây mới học từ sai số của các cây trước đó.GBT xây dựng mô hình theo phương pháp additive, bắt đầu với một dự đoán ban đầu đơn giản và liên tục thêm các cây mới để sửa lỗi (residuals):

$$f_m(x) = f_{m-1}(x) + \alpha . h_m(x)$$

Trong đó:

- $f_m(x)$ là mô hình sau mm m lần lặp
- α là learning rate (step size)
- $h_m(x)$ là cây quyết định thứ mm m dự đoán residual

Hàm mất mát cho bài toán hồi quy (Mean Squared Error):

$$L(y, F(x)) = \frac{1}{2}(y - F(x))^2$$

Gradient descent được áp dụng trong không gian hàm số:

$$f_m(x) = f_{m-1}(x) - \alpha \frac{\partial L}{\partial f_{m-1}(x)}$$

Với MSE loss, gradient chính là residual:

$$-\frac{\partial L}{\partial f_{m-1}(x)} = y - f_{m-1}(x)$$

4.3. XGBoost (Extreme Gradient Boosting)

XGBoost là phiên bản tối ưu và mở rộng của Gradient Boosting, được thiết kế để đạt hiệu suất cao và tốc độ nhanh. Đây là thuật toán phổ biến nhất trong các cuộc thi Machine Learning.

Nguyên lý hoạt động:

XGBoost sử dụng regularized objective function để tránh overfitting:

$$L(\phi) = \sum_{i=1}^{n} l(y_i, \hat{y}_i) + \sum_{k=1}^{K} \Omega(f_k)$$

Trong đó:

$l(y_i, \hat{y}_i)$ là loss function (MSE cho regression)

$\Omega(f_k)$ là regularization term cho cây thứ k

Regularization term được định nghĩa:

$$\Omega(f) = \gamma T + \frac{1}{2}\lambda \sum_{j=1}^{T} w_j^2$$

Trong đó:

- T là số lượng lá (leaves) của cây
- w_j là weight của lá thứ j
- γ là complexity penalty (L1 regularization)
- λ là regularization coefficient (L2 regularization)

Objective function tại iteration t:

$$L^{(t)} = \sum_{i=1}^{n} l(y_i, \hat{y}_i^{(t-1)}) + f_t(x_i) + \Omega(f_t)$$

XGBoost sử dụng Taylor expansion bậc hai để xấp xỉ loss function:

$$L^{(t)} \approx \sum_{i=1}^{n} [l(y_i, \hat{y}_i^{(t-1)}) + g_i f_t(x_i) + \frac{1}{2} h_i f_t^2(x_i)] + \Omega(f_t)$$

Trong đó :

$g_i = \frac{\partial(y_i, \hat{y}_i^{(t-1)})}{\partial \hat{y}_i^{(t-1)}}$ (gradient bậc nhất)

$h_i = \frac{\partial^2 l(y_i, \hat{y}_i^{(t-1)})}{\partial (\hat{y}_i^{(t-1)})^2}$ (gradient bậc hai - Hessian)

4.4. Multi-Layer Perceptron (MLP) - Deep Neural Network

MLP là mạng neural network feedforward với nhiều hidden layers.
Kiến trúc:

- Input layer: n features
- Hidden layer 1: 256 neurons + ReLU + BatchNorm + Dropout(0.3)
- Hidden layer 2: 128 neurons + ReLU + BatchNorm + Dropout(0.3)
- Hidden layer 3: 64 neurons + ReLU + BatchNorm + Dropout(0.2)
- Hidden layer 4: 32 neurons + ReLU + Dropout(0.2)
- Output layer: 1 neuron + Sigmoid

Forward propagation:

$$h^{(l)} = \sigma(W^{(l)}h^{(l-1)} + b^{(l)})$$

Trong đó:

- $h^{(l)}$ là activation của layer l
- $W^{(l)}, b^{(l)}$ là weights và bias
- σ là activation function (ReLU cho hidden layers, Sigmoid cho output)

Loss Function

$$L = \frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 + \lambda\sum_{l}\left\|W^{(l)}\right\|_2^2$$

Regularization techniques:

- L2 regularization (0.001) trên weights
- Dropout để giảm co-adaptation
- Batch Normalization để ổn định training

4.5. SVM

LinearSVC (Linear Support Vector Classifier) - một biến thể của SVM được tối ưu hóa cho dữ liệu lớn trong PySpark ML. Đây là thuật toán SVM tuyến tính với các tham số cụ thể được áp dụng.

Công thức cốt lõi của LinearSVC:

Hàm mục tiêu (Objective Function): $\min(1/2\ ||w||^2 + C \sum_{i=1}^{n} \xi_i)$

Trong đó: w là vector trọng số (weight vector); C là regularization parameter (trong project: regParam = 0.01); ξ_i là slack variables cho soft margin

Ràng buộc (Constraints): $y_i(w^T x_i + b) \geq 1 - \xi_i$, $\xi_i \geq 0$

Trong đó: y_i là label (-1 hoặc +1); x_i là feature vector đã được scaled; b là bias term

Decision function: $f(x) = w^T x + b$

Prediction rule: prediction = $\text{sign}(w^T x + b)$

Quá trình xử lý trong project:

- Feature Scaling: Dữ liệu được chuẩn hóa bằng StandardScaler trước khi đưa vào SVM
- Class Weighting: Sử dụng `class_weight` để xử lý imbalanced data
- Linear Kernel: Sử dụng kernel tuyến tính (không có kernel trick)
- Soft Margin: Cho phép một số điểm được phân loại sai để tăng generalization

4.6. Logistic Regression

Logistic Regression là một thuật toán phân loại tuyến tính, dùng để ước lượng xác suất một mẫu thuộc về lớp dương. Mô hình tính tổng có trọng số của các đặc trưng: $z = w_1 x_1 + w_2 x_2 + \cdots + w_n x_n + b$

Và chuyển đổi giá trị này thông qua hàm sigmoid: $\sigma(z) = \frac{1}{1 + e^{-z}}$

Thu được $P(y = 1 \mid \mathbf{x}) = \frac{1}{1 + e^{-(\mathbf{w}^T \mathbf{x} + b)}}$

Mô hình tìm trọng số 'w, b' sao cho hàm mất mát log-loss nhỏ nhất.

Hàm Softmax là phần mở rộng của hàm sigmoid dùng cho bài toán phân loại nhiều lớp (multiclass classification). Thay vì chỉ tính xác suất cho một lớp (như sigmoid), softmax chuyển đổi toàn bộ đầu ra tuyến tính của mô hình thành phân phối xác suất trên tất cả các lớp — sao cho tổng xác suất bằng 1.

5. Thực nghiệm, kết quả, và thảo luận (experiments, results, and discussions)

5.1.1. Mô hình dự đoán giá vé trung bình của một chuyến bay

Thiết lập thực nghiệm và lựa chọn siêu tham số

- Nhóm tiến hành thử nghiệm hai thuật toán hồi quy: Random Forest Regressor và Gradient Boosted Trees (GBT) Regressor. Dữ liệu được chia thành 80% cho huấn luyện và 20% cho kiểm tra. Trong giai đoạn huấn luyện, nhóm sử dụng TrainValidationSplit với tỷ lệ 0.8–0.2 thay cho Cross-Validation (nhằm giảm chi phí tính toán trên PySpark MLlib).
- Thuật toán Random Forest Regressor
  - Các siêu tham số được tinh chỉnh bao gồm:
    - numTrees: [50, 100, 200]
    - maxDepth: [10, 15]

      Mục tiêu là chọn tổ hợp tham số tối ưu sao cho R^2 trên tập validation cao nhất.
  - Kết quả chọn được:
    - numTrees = 200, maxDepth = 15

- Thuật toán Gradient Boosted Trees (GBT)
  - Các siêu tham số được thử nghiệm:
    - maxDepth: [4, 8]
    - maxIter : [20, 40]
    - stepSize : [0.05, 0.1]
  - Kết quả chọn được:
    - maxDepth = 8, maxIter = 40, stepSize = 0.1
- Cả hai mô hình đều được thực hiện trên PySpark MLlib với parallelism = 4 để tăng tốc xử lý.

Kết quả thực nghiệm

Mô hình	R^2 Train	R^2 Test	MAE Train	MAE Test	MSE Train	MSE Test
Random Forest	0.9621	0.9482	9.5988	10.4521	256.98	353.32
GBT Regressor	0.9736	0.9656	7.6887	7.9885	178.73	234.39

Nhận xét: GBT Regression cho kết quả tốt hơn Random Forest với R^2 cao hơn (0.9656 so với 0.9482) và sai số thấp hơn (MAE ≈ 8).Cả hai mô hình đều không bị overfit, nhưng GBT tổng quát và ổn định hơn, phù hợp cho dự đoán giá vé thực tế.

5.1.2. Mô hình dự đoán số lượng hành khách trung bình của một chuyến bay

Thiết lập thực nghiệm và lựa chọn siêu tham số

- Nhóm tiến hành thử nghiệm hai thuật toán hồi quy: Random Forest Regressor và Gradient Boosted Trees (GBT) Regressor.Dữ liệu được chia thành 80% cho huấn luyện và 20% cho kiểm tra. Trong giai đoạn huấn luyện, nhóm sử dụng TrainValidationSplit với tỷ lệ 0.8–0.2 thay cho Cross-Validation (nhằm giảm chi phí tính toán trên PySpark MLlib).
- Thuật toán Random Forest Regressor
  - Các siêu tham số được tinh chỉnh bao gồm:
    - numTrees: [50, 70, 100]
    - maxDepth: [5, 10]
      Mục tiêu là chọn tổ hợp tham số tối ưu sao cho R^2 trên tập validation cao nhất.
  - Kết quả chọn được:
    - numTrees = 50, maxDepth = 10
- Thuật toán Gradient Boosted Trees (GBT)
  - Các siêu tham số được thử nghiệm:
    - maxDepth: [5, 10]
    - maxIter: [10, 30]

        - stepSize: [0.1]
    - Kết quả chọn được:
        - maxDepth = 10, maxIter = 30, stepSize = 0.1
    - Cả hai mô hình đều được thực hiện trên PySpark MLlib với parallelism = 4 để tăng tốc xử lý.
- Kết quả thực nghiệm

Mô hình	R^2 Train	R^2 Test	MAE Train	MAE Test	MSE Train	MSE Test
Random Forest	0.9745	0.9719	0.1946	0.1996	0.0972	0.1023
GBT Regressor	0.9859	0.9762	0.1583	0.1847	0.0511	0.0864

Nhận xét: GBT Regression cho kết quả tốt hơn Random Forest với R^2 cao hơn (0.9762 so với 0.9719) và sai số thấp hơn (MAE ≈ 0.18).Cả hai mô hình đều không bị overfit, nhưng GBT tổng quát và ổn định hơn, phù hợp cho dự đoán số lượng hành khách thực tế.

5.2 Mô hình phát hiện bất thường

- Từ dữ liệu gốc nhóm đã thực hiện tổng hợp và tạo ra 20 đặc trưng mới.
- Cân bằng Lớp (Class Balancing): Để giải quyết vấn đề này, nhóm áp dụng Class Weighting với công thức:

> weight_crisis = total_samples / (2 × crisis_samples)
>
> weight_normal = total_samples / (2 × normal_samples)

Phương pháp Chia Dữ liệu: Temporal Split, vì: Mô hình được huấn luyện trên dữ liệu quá khứ để dự đoán tương lai và tránh data leakage.

Phản ánh đúng cách sử dụng thực tế trong ngành hàng không

Naive Bayes kết quả:

- AUC: 0.75
- Accuracy: 0.77
- Precision: 0.83

- Recall: 0.76
- F1-Scrore: 0.72

Giải thích:

- Naive Bayes đạt AUC = 0.75, cho thấy khả năng phân biệt tốt giữa hai lớp
- Precision cao (0.8269) cho thấy ít dự đoán sai dương
- Recall = 0.76 cho thấy mô hình dự đoán được 76% các trường hợp khủng hoảng thực tế
- F1-Score = 0.72 cho thấy sự cân bằng tốt giữa Precision và Recall

5.3 Mô hình dự đoán thị phần các hãng hàng không

- Cấu hình thực nghiệm

Dữ liệu được chia 80% huấn luyện và 20% kiểm tra. Nhóm tạo 14 features mới từ dữ liệu gốc (revenue, fare_per_mile, route_density, v.v.), tổng cộng 21 features.

- Hyperparameter Tuning

Random Forest (TrainValidationSplit):

- Siêu tham số: numTrees [50,100,150], maxDepth [10,15,20], minInstancesPerNode [1,5], maxBins [32,64], subsamplingRate [0.8, 1.0]
- Kết quả tốt nhất: numTrees=150, maxDepth=20, minInstancesPerNode = 1, maxBins=64, subsamplingRate = 1.0

Gradient Boosted Trees (TrainValidationSplit):

- Siêu tham số: maxIter [100,150,200], maxDepth [5,7,10], stepSize [0.05,0.1,0.15]
- Kết quả tốt nhất: maxIter=200, maxDepth=10, stepSize=0.05

XGBoost (3-Fold Cross-Validation):

- Siêu tham số: max_depth [4,6,8], learning_rate [0.05,0.1,0.2], n_estimators [100,200], subsample [0,7]
- Kết quả tốt nhất: max_depth=8, learning_rate=0.2, n_estimators=200, subsample = 7
- Best CV RMSE: 0.0677

MLP:

- Optimizer: Adam (lr=0.001), Batch size: 512, Epochs: 100
- Callbacks: EarlyStopping (patience=15), ReduceLROnPlateau

- Độ đo đánh giá

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2} \qquad R^2 = 1 - \frac{\Sigma(y_i - \hat{y}_i)^2}{\Sigma(y_i - \bar{y}_i)^2}$$

- Kết quả thực nghiệm

Mô hình	R^2 Train	R^2 Test	MAE Train	MAE Test	RMSE Train	RMSE Test
Random Forest	0.9653	0.9084	0.0239	0.0401	0.0419	0.0684
Gradient boosting	0.9670	0.9115	0.0252	0.0401	0.0408	0.0672
Xgboost	0.9547	0.9152	0.0275	0.0373	0.0479	0.0658
Neural Network	0.9253	0.8931	0.0309	0.0449	0.0531	0.0772

Nhận xét :

- XGBoost đạt hiệu suất tốt nhất trên tập test với R^2=0.9152 và RMSE=0.0658, vượt trội các models còn lại về khả năng dự đoán và generalization.
- Về overfitting: Random Forest và GBT có chênh lệch R^2 train-test cao (>6%), trong khi XGBoost chỉ ~4% nhờ regularization hiệu quả. Neural Network có performance thấp nhất (R^2=0.8931) do dữ liệu phù hợp hơn với tree-based models.
- Kết luận: XGBoost là lựa chọn tối ưu với sai số trung bình chỉ 3.73% market share (MAE=0.0373) và cân bằng tốt giữa accuracy và generalization.

5.4. Mô hình phân loại tuyến bay theo tiềm năng phát triển

Thiết lập thực nghiệm và lựa chọn siêu tham số

- Nhóm tiến hành thử nghiệm hai thuật toán phân loại : Random Forest Classifier và Logistic Regression. Dữ liệu được chia thành 80% cho huấn

luyện và 20% cho kiểm tra. Trong giai đoạn huấn luyện, nhóm sử dụng Cross-Validation (3 Fold) để tìm siêu tham số

- Thuật toán Random Forest Classifier
  - Các siêu tham số được tinh chỉnh bao gồm:
    - numTrees: [35, 50, 75, 100]
    - maxDepth: [5, 10, 15]
    - maxBins: [32, 64]

      Mục tiêu là chọn tổ hợp tham số tối ưu sao cho Accuracy cao nhất.
  - Kết quả chọn được:
    - numTrees = 100, maxDepth = 15, maxBins = 64
- Thuật toán Logistic Regression
  - Các siêu tham số được thử nghiệm:
    - regParam, [0.001, 0.1, 0.3, 0.5, 1.0]
    - elasticNetParam, [0.0, 0.25, 0.5, 0.75, 1.0]
  - Kết quả chọn được:
    - regParam= 0.001, elasticNetParam = 1.0

Kết quả thực nghiệm

Mô hình	Accuracy	Precision	Recall	F1-Score
Random Forest Classifier	0.9794	0.9794	0.9794	0.9794
Logistic Regression	0.9961	0.9961	0.9961	0.9961

Nhận xét:

Kết quả cho thấy Logistic Regression đạt độ chính xác và các chỉ số Precision, Recall, F1-Score cao hơn (0.9961) so với Random Forest Classifier (0.9794). Điều này chứng tỏ dữ liệu có xu hướng phân tách tuyến tính rõ ràng, nên Logistic Regression — dù đơn giản hơn — vẫn hiệu quả và tổng quát hóa tốt hơn.

6. Kết luận (conclusion)

Nghiên cứu này đã triển khai thành công hệ thống phân tích và dự đoán toàn diện cho ngành hàng không Hoa Kỳ giai đoạn 1993-2024, bao gồm năm bài toán chính: dự đoán giá vé, dự đoán số lượng hành khách, phát hiện bất thường, dự đoán thị phần và phân loại tuyến bay.

- Ở cả hai bài toán dự đoán giá vé và số lượng hành khách, mô hình GBT Regression đạt $R^2 > 0.95$, cho độ chính xác cao, sai số thấp, và tổng quát tốt hơn nhờ cơ chế học tuần tự và hiệu chỉnh sai số giữa các cây.

- Ở bài toán phát hiện bất thường, mô hình Naive Bayes: Hoạt động tốt với dữ liệu ít và có thể xử lý class imbalance thông qua class weighting.
- Ở bài toán dự đoán thị phần các hãng hàng không, XGBoost đạt hiệu suất cao nhất với $R^2 = 0.9152$ và RMSE = 0.0658 trên tập test, vượt trội hơn Random Forest ($R^2 = 0.9084$), GBT ($R^2 = 0.9115$) và Neural Network ($R^2 = 0.8931$). XGBoost có khả năng tổng quát hóa tốt nhất với chênh lệch R^2 train-test chỉ ~4%, trong khi Random Forest và GBT có chênh lệch >6%. Regularization (L1, L2) trong XGBoost giúp kiểm soát overfitting hiệu quả, phù hợp với dữ liệu có nhiều features và pattern phức tạp.
- Bài toán phân loại tuyến bay theo tiềm năng : Kết quả thực nghiệm cho thấy Logistic Regression đạt hiệu suất cao nhất với các chỉ số Accuracy, Precision, Recall và F1-Score đều đạt 0.9961, vượt trội so với Random Forest Classifier (0.9794 ở tất cả các chỉ số). Điều này cho thấy dữ liệu của bài toán có ranh giới phân tách tuyến tính rõ ràng, nên mô hình Logistic Regression - vốn là mô hình tuyến tính - phù hợp hơn và tổng quát hóa tốt hơn. Ngược lại, Random Forest Classifier có cấu trúc phức tạp hơn, thường mạnh với dữ liệu phi tuyến, nên trong trường hợp này mô hình có thể chưa phát huy được ưu thế của mình.

Hướng cải tiến tương lai:

Nếu có thêm thời gian và tài nguyên, nhóm sẽ: (1) Thu thập dữ liệu ngoài như GDP, giá nhiên liệu, thời tiết để nâng cao độ chính xác, (2) Xây dựng features temporal phức tạp hơn với seasonal decomposition và lag features, (3) Thử nghiệm mô hình nâng cao như LSTM/GRU cho time series, LightGBM và ensemble methods, (4) Chuyển sang dữ liệu monthly để tăng số lượng samples

→Nghiên cứu khẳng định machine learning có khả năng nắm bắt hiệu quả các động lực phức tạp trong ngành hàng không, hỗ trợ tối ưu hóa hoạt động và lập kế hoạch chiến lược.

7. Đóng góp (contributions)

Họ và tên	Nhiệm vụ	Hoàn thành
Nguyễn Thành An	Tiền xử lý dữ liệu, Xây dựng mô hình dự đoán thị phần	100%
Dương Minh Hiếu	Phân tích ảnh hưởng của khoảng cách đến giá vé. Xây dựng mô hình dự đoán bất thường	100%
Trương Trọng Đại Long	Xây dựng mô hình phân loại tuyến bay theo tiềm năng	100%
Nguyễn Đức Cao Thăng	Xây dựng mô hình dự đoán giá vé, số lượng hành khách trên một chuyến bay	100%

8. Tham khảo (references)

- Spark MLlib Documentation (2024). *Regression and Classification Algorithms*. Truy cập tại:https://spark.apache.org/docs/latest/ml-guide.html
- Brownlee, J. (2020). *Machine Learning Algorithms From Scratch*. Machine Learning Mastery.
- Akadir0223. (n.d.). *Flights Fare Prediction* [Notebook]. Kaggle. https://www.kaggle.com/code/akadir0223/flights-fare-prediction
- OpenAI. (2025). *ChatGPT (GPT-5)* [Large language model]. Retrieved October 26, 2025,