từ nội dung cô giao, các bạn làm step đầu tiên là **phân tích vấn đề**, xác định **input**, **output**, **các bước thực hiện**, các **kỹ thuật**, **mô hình có thể sử dụng**, **vẽ sơ đồ tổng thể**

từ bộ dataset + đề tài, e phân tích các thuộc tính, xử lý dữ liệu, trực quan

sau đó phân tích lựa chọn mô hình phù hợp để dự đoán sinh viên bỏ học/ kết quả học tập kém/ đánh giá kết quả giảng dạy của gv và đề xuất cải tiến

chủ yếu sẽ cần em có khả năng phân tích, sử dụng các kỹ thuật chuẩn hóa, tiền xử lý dữ liệu

lựa chọn huấn luyện mô hình

# 1. Vấn đề bài toán

* Trực quang hóa dữ liệu đánh giá hiệu xuất học tập của sinh viên
* Lựa chọn mô hình phù hợp dể dự đoán sinh viên có khả năng bỏ học/ kết quả học tập kém/ đánh giá kết quả giảng dại của giảng viên – từ đó đề xuất hướng cải tiến (ví dụ: gợi ý giờ học hợp lý, tư vấn đăng ký tính chỉ phù hợp, có thể tích hợp chat bot hỗ trợ học tập)

# 2. intput

<https://www.kaggle.com/datasets/mahmoudelhemaly/students-grading-dataset?select=Students_Grading_Dataset.csv>

* Bộ dữ liệu này là dữ liệu thực của 5.000 hồ sơ được thu thập từ một nhà cung cấp dịch vụ học tập tư nhân.  
  Bộ dữ liệu bao gồm các thuộc tính chính cần thiết để khám phá các mô hình, mối tương quan và thông tin chi tiết liên quan đến hiệu suất học tập.

**Các cột như sau** :

1. Student\_ID: Mã định danh duy nhất cho mỗi sinh viên.
2. First\_Name: Tên của học sinh.
3. Last\_Name: Họ của học sinh.
4. Email: Email liên hệ (có thể ẩn danh).
5. Giới tính: Nam, Nữ, Khác.
6. Tuổi: Độ tuổi của học sinh.
7. Khoa: Khoa của sinh viên (ví dụ: Khoa học máy tính, Kỹ thuật, Kinh doanh).
8. Tỷ lệ tham dự (%): Tỷ lệ tham dự (0-100%).
9. Midterm\_Score: Điểm thi giữa kỳ (trên thang điểm 100).
10. Final\_Score: Điểm thi cuối kỳ (trên thang điểm 100).
11. Assignments\_Avg: Điểm trung bình của tất cả bài tập (trên thang điểm 100).
12. Quizzes\_Avg: Điểm bài kiểm tra trung bình (trên 100).
13. Participation\_Score: Điểm dựa trên sự tham gia của lớp (0-10).
14. Projects\_Score: Điểm đánh giá dự án (trên thang điểm 100).
15. Total\_Score: Tổng điểm có trọng số của tất cả các điểm.
16. Điểm: Điểm chữ (A, B, C, D, F).
17. Study\_Hours\_per\_Week: Số giờ học trung bình mỗi tuần.
18. Hoạt động ngoại khóa: Học sinh có tham gia hoạt động ngoại khóa hay không (Có/Không).
19. Internet\_Access\_at\_Home: Học sinh có thể truy cập Internet tại nhà không? (Có/Không).
20. Parent\_Education\_Level: Trình độ học vấn cao nhất của cha mẹ (Không có, Trung học phổ thông, Cử nhân, Thạc sĩ, Tiến sĩ).
21. Mức thu nhập gia đình: Thấp, Trung bình, Cao.
22. Mức độ căng thẳng (1-10): Mức độ căng thẳng tự báo cáo (1: Thấp, 10: Cao).
23. Sleep\_Hours\_per\_Night: Số giờ ngủ trung bình mỗi đêm.

# 3. output

* **Biểu đồ** thể hiện hiệu suất học tập (điểm số, tỷ lệ tham dự, căng thẳng) theo khoa, tuổi, hoặc các yếu tố khác.(**trực quan hóa dữ liệu**)
* **Mô hình dự đoán**:
  + Xác suất sinh viên bỏ học (binary: Có/Không).
  + Xác suất kết quả học tập kém (Total\_Score < 60).
  + Đánh giá hiệu quả giảng dạy của giảng viên (dựa trên trung bình điểm lớp, tỷ lệ tham dự).
* **Đề xuất cải tiến**: Gợi ý giờ học hợp lý, số tín chỉ phù hợp, hoặc tích hợp **chatbot** hỗ trợ.

# 4. các bước thực hiện

* **Thu thập** data kaggle ta được file csv (Extract)
* **Làm sạch & chuẩn hóa dữ liệu (T**ransform**)**
* Lưu dữ liệu đã xử lý ra một cơ sở dữ liệu mới (**Load**)
* **Phân tích dữ liệu**: Xem xét các thuộc tính, kiểm tra giá trị thiếu, bất thường
* **Trực quan hóa**: Vẽ biểu đồ (scatter, bar) để phân tích mối quan hệ (ví dụ: căng thẳng vs điểm số, Tương quan giữa điểm và giờ ngủ, Tỷ lệ học sinh yếu).

\*\* phần mô hình em sẽ tìm hiểu kỉ sau do rộng quá\*\*

**dự đoán Xác suất kết quả học tập kém , Xác suất sinh viên bỏ học**

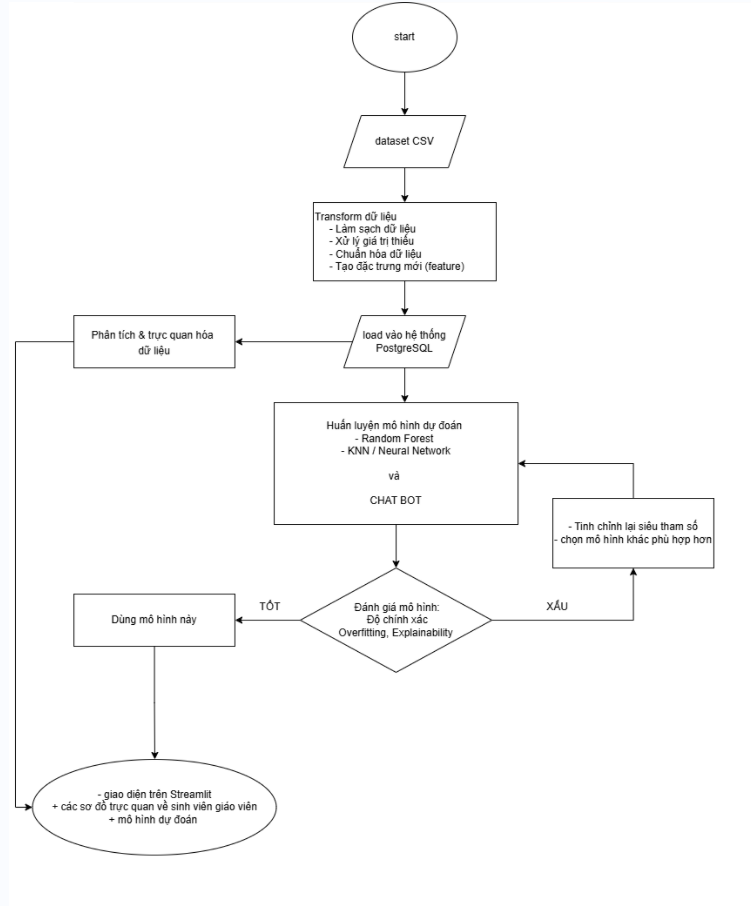
| **Tiêu chí** | **Random Forest** | **Logistic Regression** | **K-Nearest Neighbors (KNN)** | **Neural Networks (MLP)** |
| --- | --- | --- | --- | --- |
| **Loại mô hình** | Ensemble learning (tập hợp nhiều cây quyết định) | Mô hình tuyến tính | Mô hình dựa trên khoảng cách (distance-based) | Mô hình phi tuyến gồm nhiều lớp và nơ-ron nhân tạo |
| **Nguyên lý hoạt động** | Tạo nhiều cây quyết định trên các tập con dữ liệu, lấy kết quả theo đa số | Ước lượng xác suất qua hàm sigmoid và tìm ngưỡng phân loại | Tính khoảng cách (thường là Euclidean) từ điểm cần dự đoán đến các điểm gần nhất | Lan truyền dữ liệu qua nhiều lớp ẩn để học đặc trưng phi tuyến của dữ liệu |
| **Xử lý dữ liệu mất cân bằng** | Tốt (có thể dùng class\_weight hoặc sampling) | Trung bình (có thể điều chỉnh trọng số) | Kém (nhạy với dữ liệu lệch lớp) | Tốt nếu được huấn luyện đúng cách (dùng trọng số, kỹ thuật data augmentation, v.v.) |
| **Khả năng giải thích** | Tốt (có thể xem feature importance) | Rất tốt (hệ số hồi quy thể hiện rõ vai trò biến đầu vào) | Kém (khó giải thích tại sao dự đoán như vậy) | Kém (hộp đen, khó hiểu với người dùng không chuyên) |
| **Hiệu suất trên dữ liệu lớn** | Tốt, song song tốt | Rất tốt | Kém (tính khoảng cách với toàn bộ tập dữ liệu) | Trung bình đến tốt (tùy kiến trúc mạng và số lớp) |
| **Yêu cầu xử lý đặc trưng (feature)** | Không cần chuẩn hóa, xử lý rất linh hoạt | Cần chuẩn hóa nếu có nhiều tỷ lệ khác nhau | Cần chuẩn hóa dữ liệu (min-max, z-score) | Cần chuẩn hóa dữ liệu (rất quan trọng để huấn luyện hiệu quả) |
| **Tốc độ huấn luyện** | Trung bình (nhanh hơn neural net, chậm hơn logistic) | Rất nhanh | Không cần huấn luyện, nhưng dự đoán chậm khi dữ liệu lớn | Chậm nếu nhiều lớp và nhiều dữ liệu |
| **Tốc độ dự đoán** | Nhanh | Rất nhanh | Chậm nếu dữ liệu lớn (phải tính khoảng cách với tất cả điểm) | Trung bình đến chậm (tùy số lớp) |
| **Độ chính xác (Accuracy)** | Cao, ổn định trên nhiều bài toán | Trung bình đến cao (nếu dữ liệu tuyến tính) | Trung bình (bị ảnh hưởng bởi noise) | Rất cao nếu đủ dữ liệu và huấn luyện đúng cách |
| **Ứng dụng phù hợp** | Hầu hết các bài toán phân loại (đa lớp, nhị phân), đặc biệt với dữ liệu tabular | Phân loại nhị phân, bài toán tuyến tính | Phân loại đơn giản, bài toán nhỏ | Phân loại ảnh, tiếng nói, văn bản, dữ liệu phức tạp, nhận diện mẫu |
| **Khả năng overfitting** | Thấp (do sử dụng nhiều cây) | Trung bình (có thể dùng regularization) | Cao nếu K quá nhỏ | Cao nếu không regularize hoặc dữ liệu không đủ |

**xây dựng chat bot**

* Chatbot AI từ Hugging Face (trả lời tự do giống ChatGPT)
* Mô hình nội bộ (dự đoán sinh viên có nguy cơ bỏ học. tư vấn sâu…)
* Giao diện với Streamlit
* Cuối cùng Kết nối Gắn mô hình với giao diện, xử lý đầu vào & xuất kết quả

# vẽ sơ đồ tổng thể (flowchart)

<https://drive.google.com/file/d/1kC5IbvXnCHZsVtNdjv4ZG1PvBaTjzpVv/view?usp=sharing>



**1️⃣ Mở đầu – Giới thiệu tổng quan**

Lý do chọn đề tài: tầm quan trọng của việc dự đoán kết quả học tập, giúp giảm bỏ học. nhiều sinh viên vẫn đang gặp khó khăn trong quá trình học tập mà nhà trường chưa thể phát hiện và hỗ trợ kịp thời.

Mục tiêu: xây dựng hệ thống dự đoán và trực quan hóa kết quả.

Đối tượng áp dụng: sinh viên, giảng viên, cố vấn học tập có dự đoán hiệu xuất học tập và cải thiện những điểm yếu

**2️⃣Quy trình xử lý dữ liệu**

Thu thập dữ liệu sinh viên. <https://www.kaggle.com/datasets/mahmoudelhemaly/students-grading-dataset>

Tiền xử lý: chuẩn hóa, xử lý giá trị thiếu.

Trích xuất đặc trưng và sắp xếp dữ liệu huấn luyện.

**3️⃣mô hình áp dụng và lý do chọn mô hình**

Giới thiệu mô hình CatBoost

CatBoost là một bộ công cụ mã nguồn mở được phát triển bởi Yandex, rất phổ biến cho kỹ thuật **gradient boosting trên cây quyết định**

|  |  |  |  |
| --- | --- | --- | --- |
|  | **CatBoost** | **LightGBM** | **XGboost** |
| **Xử lý đặc trưng phân loại (Categorical Features)** | Xử lý đặc trưng phân loại **tự động**, không cần tiền xử lý | Hỗ trợ one-hot encoding và xử lý trực tiếp đặc trưng phân loại | **Bắt buộc** tiền xử lý đặc trưng phân loại |
| **Chiến lược chia nhánh cây (Tree Splitting Strategy)** | Chia **đối xứng** (Symmetric) | Chia **theo lá** (Leaf-wise) | Chia **theo độ sâu** (Depth-wise) |
| **Khả năng diễn giải mô hình (Interpretability)** | Quan trọng đặc trưng (Feature importances), **SHAP values** | Quan trọng đặc trưng, biểu đồ tần suất giá trị chia (split value histograms) | Quan trọng đặc trưng, sơ đồ cây (tree plots) |
| **Tốc độ và hiệu suất (Speed and Efficiency)** | Tối ưu cho tốc độ và bộ nhớ | Hiệu quả cho tập dữ liệu lớn | Khả năng mở rộng và nhanh |

Giải thích lý do chọn catboost thay vì mô hình khác ngắn gọn

Công cụ: Python, Pandas, Matplotlib, Streamlit.

**4️⃣ Xây dựng & huấn luyện mô hình**

Mô tả cách chia dữ liệu train/test.

So sánh các mô hình, chọn mô hình tốt nhất.

Đánh giá bằng các chỉ số (accuracy, precision, recall…).

So sánh độ chính xác 2 mô hình   
của catbost

Fitting 3 folds for each of 10 candidates, totalling 30 fits

precision recall f1-score support

0 0.68 0.79 0.73 592

1 0.60 0.45 0.51 408

accuracy 0.65 1000

macro avg 0.64 0.62 0.62 1000

weighted avg 0.65 0.65 0.64 1000

Accuracy: 0.65, F1: 0.51  
  
của **XGboost**

✅ Best Hyperparameters: {'subsample': 0.6, 'n\_estimators': 100, 'max\_depth': 5, 'learning\_rate': 0.01, 'colsample\_bytree': 0.6}

🎯 Accuracy: 0.5970, F1-score: 0.5524

Ít hơn nên chọn của catbost

**Kết luận & Hướng phát triển**

* Dự án sử dụng mô hình CatBoost dự đoán kết quả học tập dựa trên 23 đặc trưng liên quan đến học tập và điều kiện sống, đạt độ chính xác 65%
* Hạn chế (đặc trưng tương quan thấp (<0.3) với biến mục tiêu, dữ liệu nhiễu và mất cân bằng nhãn, số mẫu chưa đủ lớn, và tham số mô hình chưa tối ưu sâu).
* Đề xuất mở rộng: tích hợp dashboard thời gian thực, thêm yếu tố tâm lý, hành vi.

**1️⃣ Mở đầu – Giới thiệu tổng quan**

* **Lý do chọn đề tài**: Kết quả học tập phản ánh chất lượng đào tạo; nhiều sinh viên gặp khó khăn nhưng chưa được hỗ trợ kịp thời; dự đoán giúp giảm bỏ học.
* **Mục tiêu**: Xây dựng hệ thống dự đoán & trực quan hóa kết quả học tập.
* **Đối tượng áp dụng**: Sinh viên, giảng viên, cố vấn học tập nhằm dự đoán hiệu suất và cải thiện điểm yếu.

**2️⃣ dữ liệu**

* **Nguồn dữ liệu**: Kaggle (students grading dataset).
* **Tiền xử lý**: Chuẩn hóa dữ liệu, xử lý giá trị thiếu.
* **Trích xuất đặc trưng**: 23 đặc trưng về học tập

**3️⃣ Lựa chọn mô hình – Vì sao CatBoost**

* **So sánh nhanh**:

| **Thuộc tính** | **CatBoost** | **LightGBM** | **XGBoost** |
| --- | --- | --- | --- |
| Xử lý đặc trưng phân loại | Tự động | Hỗ trợ cơ bản | Cần tiền xử lý |
| Chiến lược chia cây | Đối xứng | Theo lá | Theo độ sâu |
| Tốc độ & hiệu suất | Tối ưu | Lớn | Nhanh & mở rộng |

🎯 **Vì sao chọn CatBoost?**

🔍 **Xử lý đặc trưng phân loại tốt** *(categorical features)* mà **không cần One-Hot Encoding** ➡️ tiết kiệm thời gian tiền xử lý.

⚡ **Hiệu suất ổn định** ngay cả khi dữ liệu không quá lớn hoặc không cân bằng.

🛡 **Chống overfitting tốt** nhờ kỹ thuật Ordered Boosting độc quyền của CatBoost.

🧠 **Dễ diễn giải mô hình** ➡️ dễ giải thích cho giáo viên/nhà quản lý vì có thể xem tầm quan trọng của từng đặc trưng.

📈 **So sánh nhanh:**

* **CatBoost**: Accuracy **0.65**, F1 **0.51** ✅
* **XGBoost**: Accuracy **0.597**, F1 **0.5524** 🔻
* **LightGBM**: Tốc độ huấn luyện nhanh nhưng cần xử lý dữ liệu kỹ hơn.

**5️⃣ Kết luận & Hướng phát triển**

* **Kết quả**: CatBoost dự đoán trên 23 đặc trưng, Accuracy 65%.
* **Hạn chế**: Tương quan đặc trưng thấp, dữ liệu nhiễu, mất cân bằng nhãn, tập dữ liệu nhỏ.
* **Hướng phát triển**:
  + Mở rộng dữ liệu từ nhiều nguồn.
  + Tích hợp dashboard thời gian thực.
  + Thêm yếu tố tâm lý, hành vi.

# ## \*\*Quy trình hoạt động của CatBoost\*\*

**1. \*\*Nhập dữ liệu (Input Data)\*\***

\* Bao gồm cả \*\*dữ liệu số\*\* (numerical) và \*\*dữ liệu phân loại\*\* (categorical).

\* Không cần mã hoá thủ công (\*one-hot\* hoặc \*label encoding\*) — CatBoost sẽ xử lý tự động.

**2. \*\*Mã hoá dữ liệu phân loại – Ordered Target Statistics\*\***

\* CatBoost dùng phương pháp \*\*Ordered Boosting\*\*:

\* Sắp xếp dữ liệu theo một thứ tự ngẫu nhiên.

\* Với mỗi mẫu, chỉ dùng thông tin từ các mẫu \*\*trước đó\*\* để tính thống kê (ví dụ: trung bình giá trị mục tiêu).

\* Cách này giúp \*\*tránh rò rỉ dữ liệu\*\* (\*data leakage\*) và giảm \*overfitting\*.

\* Với các cột phân loại có số giá trị ít (low cardinality), có thể dùng \*one-hot encoding\*.

**3. \*\*Xây dựng cây quyết định đối xứng (Symmetric Tree)\*\***

\* Tất cả các nút ở cùng một mức trong cây dùng \*\*cùng một điều kiện chia nhánh\*\*.

\* Điều này giúp huấn luyện và dự đoán nhanh hơn, đồng thời dễ tối ưu trên GPU.

**4. \*\*Quá trình Boosting\*\***

\* CatBoost là \*\*Gradient Boosting\*\*:

\* Xây nhiều cây tuần tự.

\* Mỗi cây mới học từ \*\*lỗi còn lại\*\* của mô hình trước.

\* Dùng \*\*Gradient Descent\*\* để tìm cách giảm hàm mất mát (\*loss function\*).

**5. \*\*Đánh giá mô hình\*\***

\* Sau khi huấn luyện, CatBoost kiểm tra mô hình bằng các chỉ số như:

\* RMSE (Root Mean Squared Error)

\* Log loss

\* AUC (Area Under Curve)

\* Có thể theo dõi \*\*feature importance\*\* để biết đặc trưng nào quan trọng nhất.

**6. \*\*Dự đoán (Prediction)\*\***

\* Khi có dữ liệu mới, CatBoost dùng chuỗi cây đã huấn luyện để tạo ra dự đoán cuối cùng.

📌 \*\*Tóm tắt dễ hiểu:\*\*

Bạn có thể hình dung CatBoost như một \*\*dàn cây quyết định thông minh\*\*, mỗi cây học từ sai sót của cây trước, nhưng khác ở chỗ nó \*\*xử lý dữ liệu phân loại trực tiếp\*\* bằng cách tính thống kê theo thứ tự (Ordered Boosting) để tránh sai lệch.

siêu tham số (hyperparameters)

quan trọng của **CatBoost** và cách dùng chúng sao cho hiệu quả.  
chia thành 4 nhóm để dễ nhớ:

## 1. Nhóm tham số chung cho mô hình

| **Tham số** | **Ý nghĩa** | **Gợi ý giá trị** |
| --- | --- | --- |
| **iterations** | Số lượng cây được xây trong quá trình boosting | Thường 500–2000, tuỳ dữ liệu |
| **learning\_rate** | Bước học (mức độ điều chỉnh sau mỗi cây) | 0.01–0.3, nhỏ thì chính xác hơn nhưng cần nhiều iterations |
| **depth** | Độ sâu của mỗi cây quyết định | 4–10 (nhỏ giúp tránh overfitting) |
| **loss\_function** | Hàm mất mát | Logloss (phân loại), RMSE (hồi quy), MAE... |
| **l2\_leaf\_reg** | Hệ số regularization L2 để tránh overfitting | 1–10 (mặc định 3) |
| **random\_seed** | Hạt giống random cho tái lập kết quả | Ví dụ: 42 |

## 2. Nhóm tham số xử lý dữ liệu phân loại

| **Tham số** | **Ý nghĩa** | **Gợi ý** |
| --- | --- | --- |
| **cat\_features** | Danh sách cột phân loại | Truyền index hoặc tên cột |
| **one\_hot\_max\_size** | Nếu số giá trị phân loại ≤ giá trị này → dùng one-hot encoding | 2–20 (mặc định 2) |
| **max\_ctr\_complexity** | Độ phức tạp của kết hợp các đặc trưng phân loại khi mã hoá | 1–3 |

## 3. Nhóm tham số tăng tốc & tối ưu

| **Tham số** | **Ý nghĩa** | **Gợi ý** |
| --- | --- | --- |
| **task\_type** | Chạy trên CPU hay GPU | "GPU" nếu có card |
| **thread\_count** | Số luồng CPU dùng để huấn luyện | Số lõi CPU |
| **bootstrap\_type** | Cách lấy mẫu bootstrap | Bayesian, Bernoulli, Poisson... |
| **subsample** | Tỷ lệ mẫu dùng cho mỗi cây | 0.5–1.0 (giúp tránh overfitting) |

## 4. Nhóm tham số dừng sớm & theo dõi

| **Tham số** | **Ý nghĩa** | **Gợi ý** |
| --- | --- | --- |
| **eval\_metric** | Chỉ số đánh giá khi huấn luyện | AUC, Accuracy, F1... |
| **use\_best\_model** | Chỉ lấy mô hình tốt nhất trên tập validation | True |
| **od\_type** | Kiểu dừng sớm (*overfitting detector*) | "Iter" hoặc "IncToDec" |
| **od\_wait** | Số vòng không cải thiện trước khi dừng | 20–50 |

.

## 1. Nhóm tham số huấn luyện cơ bản *(thường dùng nhất)*

| **Tham số** | **Ý nghĩa** | **Mẹo dùng** |
| --- | --- | --- |
| iterations, n\_estimators, num\_boost\_round, num\_trees | Số lượng cây trong mô hình | 500–2000 với learning\_rate nhỏ |
| learning\_rate, eta | Tốc độ học (bước gradient) | 0.01–0.3, nhỏ → chính xác hơn nhưng chậm hơn |
| depth, max\_depth | Độ sâu của cây | 4–10 |
| loss\_function, objective | Hàm mất mát | Logloss, RMSE, CrossEntropy... |
| l2\_leaf\_reg, reg\_lambda | Giảm độ phức tạp (phạt (penalize) các giá trị dự đoán quá lớn ở lá.) | 1–10 để tránh overfitting  (tang khi overfitting, giảm khi underfitting) |
| random\_seed, random\_state | Hạt giống random | Giúp tái lập kết quả |

## 2. Nhóm xử lý dữ liệu phân loại

| **Tham số** | **Ý nghĩa** | **Mẹo dùng** |
| --- | --- | --- |
| cat\_features | Danh sách cột phân loại | Truyền tên hoặc index cột |
| one\_hot\_max\_size | Ngưỡng để one-hot encoding | 2–20 |
| max\_ctr\_complexity | Độ phức tạp kết hợp đặc trưng | Giữ 1–3 để tránh quá phức tạp |
| ctr\_description, simple\_ctr, combinations\_ctr, per\_feature\_ctr | Cấu hình CTR encoding | Chỉ chỉnh khi hiểu sâu |
| border\_count, max\_bin | Số lượng bins cho dữ liệu số | 128–255 (tăng → chính xác hơn nhưng chậm hơn) |

## 3. Nhóm tăng tốc & tài nguyên

| **Tham số** | **Ý nghĩa** | **Mẹo dùng** |
| --- | --- | --- |
| task\_type | "CPU" hoặc "GPU" | GPU nhanh hơn nhiều với dữ liệu lớn |
| thread\_count | Số luồng CPU | = số core CPU |
| bootstrap\_type | Kiểu lấy mẫu | Bayesian, Bernoulli, Poisson... |
| subsample | Tỷ lệ mẫu mỗi cây | 0.5–1.0 |
| used\_ram\_limit, gpu\_ram\_part | Giới hạn RAM/GPU | Dùng khi bị thiếu bộ nhớ |

## 4. Nhóm dừng sớm & đánh giá

| **Tham số** | **Ý nghĩa** | **Mẹo dùng** |
| --- | --- | --- |
| eval\_metric | Metric để theo dõi | AUC, Accuracy, F1... |
| use\_best\_model | Lưu mô hình tốt nhất | True khi có tập validation |
| od\_type | Kiểu phát hiện overfitting | "Iter", "IncToDec" |
| od\_wait, early\_stopping\_rounds | Số vòng không cải thiện trước khi dừng | 20–50 |

## 5. Nhóm nâng cao & đặc biệt

* **Kiểm soát cấu trúc cây**:  
  grow\_policy, min\_data\_in\_leaf, max\_leaves, score\_function, leaf\_estimation\_method, leaf\_estimation\_iterations  
  → Tuning sâu hơn để tối ưu tốc độ & overfitting.
* **Xử lý dữ liệu văn bản**:  
  text\_features, tokenizers, dictionaries, feature\_calcers, text\_processing  
  → Dùng khi input có cột dạng văn bản.
* **Các cơ chế regularization bổ sung**:  
  model\_shrink\_rate, model\_shrink\_mode, langevin, diffusion\_temperature  
  → Thường dùng khi muốn chống overfitting mạnh.
* **Khác**:  
  class\_weights, scale\_pos\_weight, auto\_class\_weights → Xử lý dữ liệu mất cân bằng.