### ****Tóm tắt quy trình lựa chọn mô hình****

1. **Hiểu bài toán**: Xác định phân loại/hồi quy, metric, yêu cầu.
2. **Phân tích dữ liệu**: Kích thước, loại đặc trưng, chất lượng.
3. **Chọn mô hình tiềm năng**: Dựa trên dữ liệu và bài toán.
4. **Đánh giá**: Cross-validation, so sánh hiệu suất.
5. **Finetuning**: Tinh chỉnh siêu tham số để tối ưu.
6. **Kết luận**: Chọn mô hình tốt nhất, giải thích lý do.

#### ****Bước 1: Hiểu bài toán****

* **Mục tiêu**: Dự đoán Grade (A, B, C, D) → Bài toán phân loại đa lớp (multi-class classification).
* **Đầu vào**: 20 đặc trưng (7 phân loại, 13 số).
* **Đầu ra**: Nhãn phân loại (Grade).
* **Yêu cầu**:
  + Hiệu suất: Tối ưu F1-score (do dữ liệu có thể mất cân bằng giữa các lớp A, B, C, D).
  + Học kỹ năng finetuning (theo mentor).

#### ****Bước 2: Phân tích dữ liệu****

* **Kích thước**: 5.000 hàng, 23 cột → Dữ liệu trung bình.
* **Đặc trưng**:
  + **Phân loại (rời rạc)**: Gender, Department, Extracurricular\_Activities, Internet\_Access\_at\_Home, Parent\_Education\_Level, Family\_Income\_Level, Grade.
  + **Số (liên tục)**: Age, Attendance (%), Midterm\_Score, Final\_Score, Assignments\_Avg, Quizzes\_Avg, Participation\_Score, Projects\_Score, Total\_Score, Study\_Hours\_per\_Week, Stress\_Level (1-10), Sleep\_Hours\_per\_Night.

|  |  |  |
| --- | --- | --- |
| **Data Type** | **Description** | **Column Name** |
| String | Unique identifier for each student | **Student\_ID** |
| String | Student’s first name | **First\_Name** |
| String | Student’s last name | **Last\_Name** |
| String | Contact email (can be anonymized) | **Email** |
| Categorical | Male, Female, Other | **Gender** |
| Integer | Age of the student | **Age** |
| String | Student's department (e.g., CS, Engineering, Business) | **Department** |
| Float | Attendance percentage (0-100%) | **Attendance (%)** |
| Float | Midterm exam score (out of 100) | **Midterm\_Score** |
| Float | Final exam score (out of 100) | **Final\_Score** |
| Float | Average score of all assignments (out of 100) | **Assignments\_Avg** |
| Float | Average quiz scores (out of 100) | **Quizzes\_Avg** |
| Float | Score based on class participation (0-10) | **Participation\_Score** |
| Float | Project evaluation score (out of 100) | **Projects\_Score** |
| Float | Weighted sum of all grades | **Total\_Score** |
| Categorical | Letter grade (A, B, C, D, F) | **Grade** |
| Float | Average study hours per week | **Study\_Hours\_per\_Week** |
| Boolean | Whether the student participates in extracurriculars (Yes/No) | **Extracurricular\_Activities** |
| Boolean | Does the student have access to the internet at home? (Yes/No) | **Internet\_Access\_at\_Home** |
| Categorical | Highest education level of parents (None, High School, Bachelor's, Master's, PhD) | **Parent\_Education\_Level** |
| Categorical | Low, Medium, High | **Family\_Income\_Level** |
| Integer | Self-reported stress level (1: Low, 10: High) | **Stress\_Level (1-10)** |
| Float | Average hours of sleep per night | **Sleep\_Hours\_per\_Night** |

**Bước 3: Lựa chọn mô hình tiềm năng**

| **Tiêu chí** | **CatBoost** | **XGBoost** | **LightGBM** |
| --- | --- | --- | --- |
| **Kỹ thuật chính** | **Ordered Boosting** + **Oblivious Tree** (cây cân bằng) | **Gradient Boosting** + Taylor expansion (bậc 2) + Regularization | **Histogram-based Gradient Boosting** + **Leaf-wise Tree Growth** |
| **Cách học** | Học theo thứ tự để tránh overfitting khi xử lý biến phân loại | Dùng gradient & hessian để tìm nhánh tốt nhất cho cây | Chia nhỏ giá trị feature thành bins → tăng tốc huấn luyện |
| **Xử lý biến phân loại** | **Tự động encode**, rất hiệu quả | Phải **label encode hoặc one-hot** trước | Phải **label encode hoặc one-hot** trước |
| **Tốc độ huấn luyện** | Trung bình (nhanh hơn scikit-learn, chậm hơn LightGBM) | Nhanh (nhưng RAM ngốn hơn LightGBM) | **Rất nhanh**, đặc biệt với dataset lớn |
| **Độ chính xác** | Rất cao, ổn định, ít overfit | Rất cao, đặc biệt khi tuning kỹ | Cao, nhưng đôi khi **nhạy cảm với preprocessing** |
| **Khả năng tránh overfit** | **Tốt nhờ Ordered Boosting** | Tốt nếu tuning đủ | Có thể overfit do **leaf-wise splitting** |
| **Độ phức tạp tuning** | Thấp – thường hiệu quả ngay cả khi dùng mặc định | Trung bình – cần tuning eta, max\_depth, gamma... | Cao – cần kiểm soát overfit & bins |
| **Khả năng cập nhật online** | Không hỗ trợ tốt | Có thể dùng theo từng batch | Có thể cập nhật online (tốt với dữ liệu lớn) |
| **Hỗ trợ dữ liệu mất (NaN)** | Tự xử lý | Có xử lý mặc định | Tự động xử lý |
| **Tùy chỉnh mô hình** | Trung bình – ít tham số hơn | **Cao** – rất nhiều tham số & kỹ thuật tinh chỉnh | **Cao**, nhưng dễ gây sai nếu chọn sai tham số |
| Ưu điểm | - Xử lý tốt biến phân loại tự động - Tránh overfit tốt - Dễ dùng | - Cân bằng giữa tốc độ và chính xác - Hỗ trợ rất mạnh và phổ biến | - Rất nhanh, tiết kiệm RAM - Rất hiệu quả với dữ liệu lớn & nhiều chiều |
| Nhược điểm | - Huấn luyện chậm hơn LightGBM - Ít lựa chọn tuning hơn | - Cần encode dữ liệu thủ công - RAM tiêu thụ lớn với dữ liệu lớn | - Dễ bị overfit - Phân nhánh leaf-wise đôi khi không ổn định với dữ liệu nhỏ |

**### ✅ 1. CatBoost**

**| Kỹ thuật chính | Ordered Boosting + Oblivious Tree |**

**🔍 Giải thích:**

**\* Ordered Boosting: CatBoost xử lý các biến rời rạc (categorical) rất tốt mà không cần one-hot hay label encode bằng tay. Nó dùng dữ liệu huấn luyện theo một cách có thứ tự để tránh leak (data leakage).**

**\* Oblivious Tree: Mỗi node trong cây quyết định trên cùng một đặc trưng và ngưỡng, tạo ra cây đối xứng → nhanh và hiệu quả.**

**Ví dụ:**

**Với cột `"Parent\_Education\_Level"` (có giá trị: None, High School, Bachelor's, ...), CatBoost tự động mã hóa thành các giá trị số có ý nghĩa thống kê → giúp mô hình học được mối liên hệ giữa trình độ cha mẹ và điểm số học sinh mà không cần bạn phải one-hot hoặc encode thủ công.**

**--**

**### ✅ 2. XGBoost**

**| Kỹ thuật chính | Gradient Boosting + Taylor Expansion (bậc 2) + Regularization |**

**🔍 Giải thích:**

**\* Gradient Boosting: Học sai số (residuals) của mô hình trước đó để dần cải thiện kết quả.**

**\* Taylor expansion bậc 2: Sử dụng cả đạo hàm bậc 1 (gradient) và bậc 2 (hessian) để cập nhật cây, giúp mô hình ổn định và hiệu quả hơn.**

**\* Regularization (`reg\_lambda`, `reg\_alpha`): Giúp kiểm soát overfitting đặc biệt hữu ích khi có nhiều biến phân loại như `Gender`, `Stress\_Level`, `Family\_Income\_Level`.**

**Ví dụ:**

**\* Khi bạn sử dụng `reg\_lambda=1.0`, XGBoost sẽ phạt các cây có độ phức tạp cao, ví dụ như khi cây học quá kỹ mối quan hệ giữa `"Stress\_Level"` và `"Grade"`.**

**\* Các feature có nhiều giá trị như `"Department"` hoặc `"Parent\_Education\_Level"` khi được one-hot encode sẽ sinh ra rất nhiều cột → dễ gây overfit nếu không có regularization.**

**---**

**### ✅ 3. LightGBM**

**| Kỹ thuật chính | Histogram-based Gradient Boosting + Leaf-wise Tree Growth |**

**🔍 Giải thích:**

**\* Histogram-based: Chuyển dữ liệu liên tục thành các bin rời rạc → tăng tốc độ huấn luyện rất nhiều.**

**\* Leaf-wise Tree Growth: Thay vì chia cây theo độ sâu (`level-wise`), LightGBM chọn nhánh có độ lợi cao nhất để phát triển tiếp → tăng độ chính xác, nhưng dễ bị overfit trên dữ liệu nhỏ.**

**📘 Ví dụ:**

**\* Nếu bạn có cột `"Sleep\_Hours\_per\_Night"` với giá trị liên tục từ 0–12h, LightGBM sẽ chia thành 255 bin (mặc định), như 0–2h, 2–4h,... → tăng tốc độ tính toán.**

**\* Nhưng với dữ liệu nhỏ (ví dụ chỉ có 200 học sinh), mô hình có thể quá khớp (overfit) nếu không dùng `min\_data\_in\_leaf`, `max\_depth`, `lambda\_l2`.**

| **Tiêu chí** | **TabPFN** |
| --- | --- |
| **Kỹ thuật chính** | **Pretrained Transformer** + **Bayesian Posterior Approximation** |
| **Cách học** | Học **meta-learning** trên hàng triệu bài toán tabular tổng hợp, sau đó dự đoán **phân phối posterior** của hàm mục tiêu (không cần huấn luyện truyền thống). |
| **Xử lý biến phân loại** | Hỗ trợ tốt, có thể xử lý cả categorical và numerical mà không cần encode thủ công. |
| **Tốc độ huấn luyện** | **Gần như không cần huấn luyện** (vì là mô hình pretrained), chỉ cần dự đoán posterior trong vài giây. |
| **Độ chính xác** | **Cao**, đặc biệt trong bài toán nhỏ – vừa với số lượng feature < 100 và rows < 10K. |
| **Khả năng tránh overfit** | **Rất tốt** – nhờ học từ rất nhiều phân phối khả dĩ (posterior). |
| **Độ phức tạp tuning** | **Không cần tuning** – chạy một lần là ra kết quả. |
| **Khả năng cập nhật online** | Không hỗ trợ learning theo batch hoặc cập nhật liên tục. |
| **Hỗ trợ dữ liệu mất (NaN)** | Không xử lý NaN trực tiếp – cần xử lý trước khi đưa vào. |
| **Tùy chỉnh mô hình** | Không tùy chỉnh được cấu trúc mô hình hoặc loss – tất cả đã được học sẵn. |
| ✅ **Ưu điểm** | - Không cần huấn luyện- Không cần tuning- Rất nhanh trên dataset nhỏ- Học được mối quan hệ phức tạp, phi tuyến tính |
| ❌ **Nhược điểm** | - Không phù hợp cho dataset lớn (>10k hàng)- Không hỗ trợ update online- Không có khả năng tùy chỉnh sâu- Cần xử lý NaN và scale thủ công |

| **Tiêu chí** | **✅ XGBoost** | **❌ TabPFN** |
| --- | --- | --- |
| Kiểm soát mô hình | Rất cao | Không có |
| Dễ thực hiện FE thủ công | Có thể đánh giá bằng feature importance, SHAP | Không hỗ trợ SHAP hoặc kiểm soát đầu vào cụ thể |
| Học cách tuning mô hình | Có nhiều hyperparameter quan trọng | Không cần tuning, mất cơ hội học |
| Học cách chọn biến | Dễ thực nghiệm, loại biến, đo độ ảnh hưởng | Không kiểm soát được điều này |
| Phân tích sâu vào từng đặc trưng | Có thể đo ảnh hưởng từng feature | Không biết mô hình học như thế nào |
| Kiểm soát mô hình | Rất cao | Không có |
| Dễ thực hiện FE thủ công | Có thể đánh giá bằng feature importance, SHAP | Không hỗ trợ SHAP hoặc kiểm soát đầu vào cụ thể |
| Học cách tuning mô hình | Có nhiều hyperparameter quan trọng | Không cần tuning, mất cơ hội học |
| Học cách chọn biến | Dễ thực nghiệm, loại biến, đo độ ảnh hưởng | Không kiểm soát được điều này |
| Phân tích sâu vào từng đặc trưng | Có thể đo ảnh hưởng từng feature | Không biết mô hình học như thế nào |

## ****XGBoost xử lý linh hoạt****

| **Loại đặc trưng** | **Dữ liệu của** | **Lợi thế khi dùng XGBoost** |
| --- | --- | --- |
| **Phân loại (rời rạc)** | Gender, Department, Grade,... | có thể **tự encode** hoặc dùng OneHot, sau đó kiểm tra **feature importance** từng biến. |
| **Liên tục (số)** | Total\_Score, Midterm\_Score,... | XGBoost **không yêu cầu scale** và học tốt quan hệ phi tuyến giữa các biến. |

TabPFN sẽ **chỉ trả về kết quả**, không thể giải thích được điều gì đang xảy ra.

****X** = Extreme:**

Thể hiện sự **tối ưu hóa cực đại** so với các phương pháp Boosting thông thường.

Có nghĩa là XGBoost **cải tiến mạnh mẽ về tốc độ, độ chính xác và khả năng tùy chỉnh**.

****GBoost** = Gradient Boosting:**

Là kỹ thuật học máy mà các cây quyết định (decision trees) được xây dựng tuần tự.

Mỗi cây mới học dựa trên **sai số (gradient)** của những cây trước đó.

Ý tưởng chính là **học từ lỗi** để cải thiện dần mô hình.

