Phân loại Tidal Disruption Events (TDE) từ Dữ liệu Đường cong Ánh sáng Thiên văn

Lời nói đầu

Báo cáo này trình bày phương pháp tiếp cận của chúng tôi cho thử thách MALLORN Astronomical Classification Challenge, với mục tiêu phân loại các sự kiện Tidal Disruption Events (TDE) từ dữ liệu đường cong ánh sáng thiên văn. TDE là hiện tượng hiếm gặp xảy ra khi một ngôi sao bị xé nát bởi lực thủy triều của một lỗ đen siêu khối lượng. Bộ dữ liệu bao gồm 3.043 đối tượng thiên văn, trong đó chỉ có 148 TDE (4,86%), tạo nên một bài toán phân loại với độ mất cân bằng lớp đáng kể. Chúng tôi đã áp dụng các kỹ thuật tiền xử lý dữ liệu để xử lý các vấn đề về Redshift (Z) và Extinction by Dust (EBV), kết hợp với các phương pháp kỹ thuật đặc trưng từ dữ liệu đường cong ánh sáng. Các mô hình học máy bao gồm XGBoost, LightGBM, CatBoost và mạng nơ-ron ResNet được huấn luyện với cross-validation và tối ưu hóa siêu tham số bằng Optuna. Kết quả cuối cùng được kết hợp thông qua phương pháp ensemble để tối đa hóa hiệu suất phân loại.

1 Giới thiệu

Tidal Disruption Events (TDE) là hiện tượng thiên văn hiếm gặp xảy ra khi một ngôi sao di chuyển đủ gần một lỗ đen siêu khối lượng và bị xé nát bởi lực thủy triều (Rees, 1988; Gezari, 2021). Việc phát hiện và phân loại TDE đóng vai trò quan trọng trong nghiên cứu vật lý lỗ đen và sự tiến hóa của thiên hà. Thử thách MALLORN Astronomical Classification Challenge đặt ra bài toán phân loại nhị phân: xác định các đối tượng thiên văn là TDE hay không-TDE dựa trên dữ liệu đường cong ánh sáng (lightcurve). Đây là một bài toán có độ khó cao do:

- **Mất cân bằng lớp nghiêm trọng**: Chỉ 4,86% đối tượng là TDE
- **Nhiễu và độ không chắc chắn**: Dữ liệu quan sát chứa nhiều nguồn nhiễu
- **Sự đa dạng của đối tượng**: Nhiều loại đối tượng thiên văn có thể bị nhầm lẫn với TDE

Trong báo cáo này, chúng tôi trình bày:

- Phân tích khám phá dữ liệu (EDA) để hiểu đặc điểm của bộ dữ liệu
- Quy trình tiền xử lý và kỹ thuật đặc trưng
- Các mô hình học máy được sử dụng: XGBoost, LightGBM, CatBoost, ResNet
- Phương pháp ensemble và kết quả đánh giá

Mã nguồn và các implementation chi tiết được công khai tại GitHub[1].

2 Phân tích Dữ liệu Khám phá (EDA)

2.1 Mô tả Bộ dữ liệu

Bộ dữ liệu MALLORN bao gồm hai thành phần chính:

- **Bảng log (metadata)**: Chứa thông tin về các đối tượng thiên văn, bao gồm định danh, phân loại quang phổ (SpecType), Redshift (Z), và hệ số tổn thất ánh sáng do bụi vũ trụ (EBV).
- **Bảng lightcurves**: Chứa dữ liệu đường cong ánh sáng theo thời gian với các phép đo thông lượng (flux) ở nhiều dải bước sóng khác nhau.

Thống kê	Giá trị
Tổng số đối tượng	3.043
Số lượng TDE	148
Tỷ lệ TDE	4,86%
Số lượng non-TDE	2.895
Tỷ lệ non-TDE	95,14%

Table 1: Thống kê tổng quan bộ dữ liệu MALLORN.

[1] https://github.com/EnderMagician/ML-Final

Với tỷ lệ TDE chỉ chiếm khoảng 5% tổng số dữ liệu, đây là một bài toán phân loại với **độ mất cân bằng lớp (class imbalance) nghiêm trọng** (He and Garcia, 2009). Điều này đòi hỏi các kỹ thuật xử lý đặc biệt như điều chỉnh trọng số lớp (class weighting), sử dụng các hàm mất mát phù hợp, hoặc kỹ thuật oversampling như SMOTE (Chawla et al., 2002).

2.2 Phân phối các Đặc trưng

Loại quang phổ (SpecType): Bộ dữ liệu bao gồm nhiều loại đối tượng thiên văn:

- **TDE**: Sự kiện xé nát sao bởi lỗ đen
- **AGN**: Nhân thiên hà hoạt động (Active Galactic Nuclei)
- **Supernovae**: SN Ia, SN Ib, SN Ic, SN II, SN IIn, SN IIL, SN IIP

Redshift (Z): Đo lường sự dịch chuyển về phía đỏ của ánh sáng do sự giãn nở của vũ trụ. Các TDE thường có phân bố Redshift khác biệt, khiến Z trở thành một **đặc trưng phân biệt mạnh**.

Extinction by Dust (EBV): Hệ số đo lường mức độ hấp thụ và tán xạ ánh sáng bởi bụi vũ trụ, ảnh hưởng đến màu sắc quan sát được của đối tượng.

2.3 Phân tích Chất lượng Dữ liệu

Dữ liệu thô có một số vấn đề cần xử lý:

- **Dữ liệu gây hiểu lầm**: Các giá trị thông lượng (flux) chưa được hiệu chỉnh theo Redshift và EBV
- **Giá trị khuyết**: Một số trường như Z_err có giá trị NaN
- **Tính đa dạng nguồn**: Các đối tượng đến từ nhiều nguồn quan sát khác nhau

Chúng tôi sử dụng hàm `preprocess.clean_data()` để tính toán giá trị thông lượng thực có xét đến ảnh hưởng của Redshift và Extinction.

3 Data Engineering

Phần này trình bày quy trình xử lý dữ liệu và kỹ thuật đặc trưng (feature engineering) được sử dụng để chuyển đổi dữ liệu đường cong ánh sáng thô thành các đặc trưng có ý nghĩa cho bài toán phân loại. Tổng cộng, chúng tôi trích xuất **290 đặc trưng**, trong đó có **221 đặc trưng từ Gaussian Process**.

3.1 Tiền xử lý Dữ liệu: Hiệu chỉnh Extinction

Ánh sáng từ các đối tượng thiên văn bị hấp thụ và tán xạ bởi bụi vũ trụ (dust extinction). Để có được giá trị thông lượng (flux) thực, chúng tôi thực hiện hiệu chỉnh extinction sử dụng hệ số $E(B-V)$ từ metadata.

Công thức hiệu chỉnh: Độ tổn thất ánh sáng tại bước sóng λ được tính bằng:

$$A_\lambda = R_\lambda \times E(B-V)$$

trong đó R_λ là hệ số extinction cho từng bộ lọc LSST:

Filter	u	g	r	i	z	y
R_λ	4.81	3.64	2.70	2.06	1.58	1.31
λ (nm)	368	480	622	754	868	973

Table 2: Hệ số extinction R_λ và bước sóng trung tâm cho các bộ lọc LSST.

Thông lượng được hiệu chỉnh theo công thức:

$$\text{Flux}_{\text{corrected}} = \text{Flux}_{\text{observed}} \times 10^{0.4 \times A_\lambda}$$

3.2 Feature Engineering với Gaussian Process 2D

Điểm đặc biệt trong phương pháp của chúng tôi là sử dụng **Gaussian Process (GP) 2 chiều** để mô hình hóa đường cong ánh sáng trên cả hai chiều: **Thời gian (Time)** và **Bước sóng (Wavelength)**. Điều này cho phép "mượn" thông tin từ các bộ lọc có nhiều dữ liệu để bù đắp cho các bộ lọc thưa thớt.

Kernel Configuration:

$$k(x, x') = k_{\text{Matérn}}(x, x'; \nu = 1.5) + k_{\text{White}}(x, x')$$

với length scales khởi tạo: $l_{\text{time}} = 50$ ngày, $l_{\text{wavelength}} = 2000$ nm.

3.3 Các Nhóm Đặc trưng

3.3.1 Nhóm 1: Hyperparameters của Kernel

Các tham số học được từ GP phản ánh đặc tính biến thiên của đường cong ánh sáng:

- `gp_time_scale`: Thang thời gian biến thiên (TDE: dài, SN: ngắn)
- `gp_wavelength_scale`: Mức độ tương quan giữa các bộ lọc
- `gp_noise_level`: Mức độ nhiễu trong dữ liệu
- `gp_log_likelihood`: Chất lượng fit của mô hình GP

3.3.2 Nhóm 2: Thống kê Residual

Phân tích sai số giữa dữ liệu thực và dự đoán GP:

- Mean, Std, Median, Skewness, Kurtosis của residual
- MAE, MSE: Đo lường chất lượng xấp xỉ của GP

3.3.3 Nhóm 3: Đặc trưng theo Bộ lọc

Cho mỗi bộ lọc (u, g, r, i, z, y):

- `gp_smoothed_mean/max/min/std`: Thống kê đường cong được làm mượt
- `gp_deriv_mean/std/max/min`: Đạo hàm bậc nhất (tốc độ biến thiên)
- `gp_peak_time/value`: Thời điểm và giá trị cực đại

3.3.4 Nhóm 4: Đặc trưng Màu sắc (Color Features)

Màu sắc thiên văn được định nghĩa là hiệu thông lượng giữa các bộ lọc:

$$\text{Color}_{X-Y} = \text{Flux}_X - \text{Flux}_Y$$

Chúng tôi tính màu sắc tại 3 thời điểm: đầu (early), giữa (mid), cuối (late):

- `gp_color_ug`, `gp_color_gr`, `gp_color_ri`, `gp_color_iz`
- `gp_color_evolution`: Sự thay đổi màu sắc theo thời gian

Lý do quan trọng cho TDE: TDE thường có màu xanh (blue) và tiến hóa từ xanh sang đỏ theo thời gian, trong khi AGN có màu sắc ổn định.

3.3.5 Nhóm 5: Tương quan Xuyên Bộ lọc

$$\rho_{XY} = \text{corr}(\text{Flux}_X(t), \text{Flux}_Y(t))$$

Lý do: TDE và SN có tương quan cao giữa các bộ lọc (biến thiên đồng bộ), trong khi AGN có tương quan thấp hơn (biến thiên ngẫu nhiên).

3.3.6 Nhóm 6: Đặc trưng Rise/Decline (Quan trọng cho TDE)

TDE có đặc điểm: **tăng nhanh, giảm chậm** (rise time $\ll$ decline time).

- `gp_rise_duration/rate`: Thời gian và tốc độ tăng sáng
- `gp_decline_duration/rate`: Thời gian và tốc độ giảm sáng
- `gp_asymmetry_ratio`: Tỷ lệ bất đối xứng

$$\text{Asymmetry Ratio} = \frac{\text{Decline Duration}}{\text{Rise Duration}}$$

TDE có asymmetry ratio cao (có thể > 5), trong khi SN Ia có tỷ lệ gần $1 - 2$.

3.3.7 Nhóm 7: Peak Timing Across Bands

TDE có đặc điểm: **bộ lọc xanh đạt cực đại trước bộ lọc đỏ**.

- `gp_peak_timing_diff_ur`: $t^u_{\text{peak}} - t^r_{\text{peak}}$
- `gp_peak_timing_diff_gi`: $t^g_{\text{peak}} - t^i_{\text{peak}}$
- `gp_peak_timing_spread`: Độ phân tán thời điểm peak

Lý do: AGN có các peak đồng bộ (spread thấp), TDE có peak lệch pha (spread cao).

3.3.8 Nhóm 8: Đặc trưng Spectral

Phân tích sự biến đổi thông lượng theo bước sóng:

$$\text{Spectral Slope} = \frac{d(\text{Flux})}{d\lambda}$$

Lý do: TDE có spectral slope âm (xanh hơn), trong khi các đối tượng khác có thể có slope khác nhau.

3.3.9 Nhóm 9: Excess Variance (Phân biệt AGN)

AGN có biến thiên ngẫu nhiên (stochastic), trong khi TDE/SN có biến thiên trơn (smooth):

$$\sigma^2_{\text{excess}} = \frac{\text{Var}(\text{Flux}) - \langle \sigma^2_{\text{err}} \rangle}{\langle \text{Flux} \rangle^2}$$

AGN có excess variance cao, TDE có excess variance thấp.

3.3.10 Nhóm 10: Structure Function

Đo lường biến thiên ở các thang thời gian khác nhau:

$$SF(\tau) = \sqrt{\langle [\text{Flux}(t+\tau) - \text{Flux}(t)]^2 \rangle}$$

Tính cho $\tau = 1, 5, 10$ điểm. **Lý do:** AGN có structure function tăng theo τ (red noise), TDE có pattern khác biệt.

3.3.11 Nhóm 11: Curvature (Đạo hàm bậc 2)

$$\text{Curvature} = \frac{d^2(\text{Flux})}{dt^2}$$

Lý do: TDE có curvature trơn, SN có curvature mạnh (exponential decay), AGN có curvature bất thường.

3.3.12 Nhóm 12: Đặc trưng tỉ lệ với Redshift (Metadata)

Để chuẩn hóa độ sáng theo hiệu ứng khoảng cách vũ trụ, một số đặc trưng được hiệu chỉnh theo redshift (độ dịch chuyển đỏ) theo dạng tỉ lệ z^2. Ví dụ:

$$\text{Flux\_Z\_Scaled}_{\text{band}} = \text{Flux\_mean}_{\text{band}} \times z^2$$
$$\text{Flux\_Max\_Z\_Scaled}_{\text{band}} = \text{Flux\_max}_{\text{band}} \times z^2$$

Cơ sở vật lý của bước này dựa trên định luật *luminosity distance* trong vũ trụ học, nhằm giảm ảnh hưởng của khoảng cách khi so sánh độ sáng tuyệt đối giữa các đối tượng thiên văn.

3.3.13 Nhóm 13: Von Neumann Ratio

Để đánh giá mức độ "trơn" của chuỗi thời gian, nghiên cứu sử dụng chỉ số Von Neumann Ratio (η), được xác định bởi:

$$VN = \frac{\text{Mean}(\Delta x^2)}{\text{Variance}(x)} = \frac{\frac{1}{n-1}\sum_{i=1}^{n-1}(x_{i+1} - x_i)^2}{\frac{1}{n}\sum_{i=1}^{n}(x_i - \bar{x})^2}$$

Chỉ số này có ý nghĩa thống kê quan trọng trong việc phân loại:

- $VN \approx 2$: Tương ứng với nhiễu trắng (white noise).
- $VN < 2$: Thể hiện chuỗi có tương quan dương và biến thiên trơn (smooth).
- $VN > 2$: Thể hiện tính dao động mạnh và tương quan âm.

Trong ứng dụng thực tế, VN giúp phân biệt hiệu quả giữa **AGN** (thường có biến thiên ngẫu nhiên, chỉ số VN cao) với **SNe/TDE** (thường có quá trình tiến hóa trơn, chỉ số VN thấp hơn).

3.4 Tổng kết Feature Engineering

Loại đặc trưng	Số lượng
Thống kê cơ bản (mean, max, min, std, skew)	30
Von Neumann Ratio	6
Redshift scaling	12
GP Kernel Hyperparameters	6
GP Residual Statistics	10
GP Per-band Statistics	84
GP Color Features	16
GP Cross-band Correlations	15
GP Rise/Decline Features	30
GP Spectral Features	2
GP Variability Metrics	20
GP Curvature Features	18
Metadata (Z, target)	2
Tổng cộng	**290**

Table 3: Tổng hợp các nhóm đặc trưng được trích xuất.

4 Mô hình Học máy

Chiến lược mô hình hóa của chúng tôi dựa trên việc **kết hợp (ensemble) 4 mô hình**: 3 mô hình cây quyết định (XGBoost, LightGBM, CatBoost) và 1 mạng nơ-ron (ResNet). Mỗi mô hình được huấn luyện với 5-fold Stratified Cross-Validation để đảm bảo tính nhất quán và đáng tin cậy. **Lưu ý quan trọng:** Các đặc trưng được trích xuất ở phần Data Engineering được thiết kế tối ưu cho các mô hình tree-based (có thể xử lý NaN, không yêu cầu scaling). Do đó, mạng nơ-ron cần quy trình tiền xử lý riêng biệt.

4.1 Tiền xử lý cho Neural Network

Các mô hình tree-based có thể xử lý trực tiếp dữ liệu thô với giá trị NaN, nhưng Neural Network yêu cầu:

- Không có giá trị NaN trong đầu vào
- Các đặc trưng được chuẩn hóa về cùng một thang đo

Quy trình tiền xử lý:

1. **Feature Engineering - NaN Count:** Tạo đặc trưng mới đếm số lượng giá trị NaN cho mỗi mẫu:

$$\texttt{nan\_count}_i = \sum_{j=1}^{d} \mathbb{1}[\text{feature}_{i,j} = \text{NaN}]$$

   Đặc trưng này bảo toàn thông tin về "độ hoàn chỉnh" của dữ liệu quan sát.

2. **Imputation - Median Fill:** Điền các giá trị NaN bằng median của từng cột:

$$x_{i,j}^{\text{imputed}} = \begin{cases} x_{i,j} & \text{nếu } x_{i,j} \neq \text{NaN} \\ \text{median}(x_{\cdot,j}) & \text{nếu } x_{i,j} = \text{NaN} \end{cases}$$

3. **Scaling - StandardScaler:** Chuẩn hóa đặc trưng về phân phối chuẩn:

$$x_{i,j}^{\text{scaled}} = \frac{x_{i,j}^{\text{imputed}} - \mu_j}{\sigma_j}$$

Quan trọng: Scaler được fit trên tập huấn luyện và áp dụng cho cả tập huấn luyện và kiểm tra để tránh data leakage.

Thống kê	Giá trị
Số đặc trưng ban đầu	288
Đặc trưng mới (nan_count)	1
Tổng đặc trưng sau tiền xử lý	289
NaN trước imputation	13,945
NaN sau imputation	0

Table 4: Thống kê tiền xử lý cho Neural Network.

4.2 Optuna: Framework Tối ưu Siêu tham số

Optuna (Akiba et al., 2019) là một framework tối ưu siêu tham số tự động (Automated Hyperparameter Optimization - HPO) với các đặc điểm nổi bật:

Thuật toán Tree-structured Parzen Estimator (TPE): Optuna sử dụng TPE (Bergstra et al., 2011) - một phương pháp Bayesian Optimization hiệu quả:

1. Chia không gian siêu tham số thành hai phân phối: $l(x)$ cho các trial tốt và $g(x)$ cho các trial xấu
2. Tối đa hóa Expected Improvement:

$$\text{EI}(x) = \frac{l(x)}{g(x)}$$

3. Lựa chọn siêu tham số tiếp theo dựa trên maximize EI

Cơ chế Pruning: Optuna hỗ trợ **early stopping** cho các trial không triển vọng, tiết kiệm thời gian tính toán đáng kể.

Cấu hình trong dự án:

- **Số trials**: 100 trials cho mỗi mô hình
- **Objective**: Maximize F1 Score trên validation set
- **Direction**: Maximize
- **Cross-validation**: 5-fold Stratified CV trong mỗi trial

4.3 XGBoost

XGBoost (Chen and Guestrin, 2016) (eXtreme Gradient Boosting) là một thuật toán gradient boosting được tối ưu cho hiệu suất và tốc độ.

Tham số	Phạm vi	Mô tả
learning_rate	[0.01, 0.15]	Tốc độ học
max_depth	[3, 9]	Độ sâu tối đa của cây
min_child_weight	[1, 10]	Tổng trọng số tối thiểu của lá
subsample	[0.6, 0.95]	Tỷ lệ mẫu cho mỗi cây
colsample_bytree	[0.5, 0.95]	Tỷ lệ đặc trưng cho mỗi cây
reg_lambda	$[10^{-3}, 10]$	L2 regularization
reg_alpha	$[10^{-3}, 10]$	L1 regularization
scale_pos_weight	[10, 40]	Trọng số lớp dương (xử lý imbalance)

Table 5: Không gian tìm kiếm siêu tham số cho XGBoost.

Không gian siêu tham số được tối ưu:

Xử lý Class Imbalance: Tham số `scale_pos_weight` được tính toán và tối ưu để cân bằng giữa lớp TDE (thiểu số) và non-TDE (đa số):

$$\texttt{scale\_pos\_weight} = \frac{\text{số mẫu negative}}{\text{số mẫu positive}} = \frac{2895}{148} \approx 19.56$$

Kết quả tối ưu:

- Best F1 Score (Optuna): **0.6323**
- Số estimators huấn luyện cuối: 3000 (với early stopping)
- OOF F1 Score: **0.5967** tại threshold 0.2859

4.4 LightGBM

LightGBM (Ke et al., 2017) (Light Gradient Boosting Machine) của Microsoft sử dụng kỹ thuật Gradient-based One-Side Sampling (GOSS) và Exclusive Feature Bundling (EFB) để tăng tốc huấn luyện.

Ưu điểm so với XGBoost:

- Huấn luyện nhanh hơn với bộ nhớ thấp hơn
- Hỗ trợ categorical features native
- Sử dụng leaf-wise growth thay vì level-wise

Không gian siêu tham số: Tương tự XGBoost với các tham số đặc trưng của LightGBM:

- `num_leaves`: Số lá tối đa trong cây
- `feature_fraction`: Tương đương colsample_bytree
- `bagging_fraction`: Tương đương subsample
- `scale_pos_weight`: Xử lý class imbalance

4.5 CatBoost

CatBoost (Prokhorenkova et al., 2018) (Categorical Boosting) của Yandex được thiết kế đặc biệt cho dữ liệu có nhiều categorical features và xử lý tốt overfitting.

Đặc điểm nổi bật:

- **Ordered Boosting**: Giảm prediction shift và overfitting
- **Symmetric Trees**: Cây đối xứng giúp inference nhanh hơn
- **Native Categorical Support**: Không cần one-hot encoding

Không gian siêu tham số:

- `depth`: Độ sâu cây [4, 10]
- `learning_rate`: Tốc độ học [0.01, 0.15]
- `l2_leaf_reg`: L2 regularization
- `scale_pos_weight`: Xử lý class imbalance

4.6 ResNet (Neural Network)

TabularResNet là kiến trúc mạng nơ-ron theo phong cách ResNet (He et al., 2016) được thiết kế cho dữ liệu dạng bảng (tabular data), được triển khai bằng PyTorch (Paszke et al., 2019). Khác với các mô hình tree-based, mạng nơ-ron yêu cầu dữ liệu đã được tiền xử lý (không có NaN, đã được chuẩn hóa).

Kiến trúc mô hình:

- **Input Layer**: Linear($d_{input} \rightarrow d_{hidden}$) + BatchNorm + PReLU + Dropout
- **ResNet Blocks**: 1-3 blocks với skip connections
- **Output Layer**: Linear($d_{hidden} \rightarrow 1$)

ResNet Block: Mỗi block bao gồm:

$$\text{Output} = x + f(x)$$

trong đó $f(x)$ là một chuỗi gồm: Linear $\rightarrow$ BatchNorm $\rightarrow$ PReLU $\rightarrow$ Dropout (Srivastava et al., 2014) $\rightarrow$ Linear $\rightarrow$ BatchNorm $\rightarrow$ PReLU $\rightarrow$ Dropout.

Skip connection giúp gradient lan truyền dễ dàng hơn và tránh vấn đề vanishing gradient.

Xử lý Class Imbalance: Sử dụng `BCEWithLogitsLoss` với `pos_weight`:

$$\text{pos\_weight} = \frac{\text{số mẫu negative}}{\text{số mẫu positive}} \approx 19.56$$

Bagging: Để tăng tính ổn định và giảm variance (Breiman, 1996), chúng tôi sử dụng kỹ thuật bagging với 5 models mỗi fold, mỗi model được huấn luyện với random seed khác nhau và tối ưu bằng Adam (Kingma and Ba, 2014). Kết quả cuối cùng là trung bình của tất cả các predictions.

Không gian siêu tham số tối ưu bởi Optuna:

- `hidden_dim`: [64, 128, 256, 512]
- `n_blocks`: [1, 3]
- `dropout`: [0.1, 0.5]
- `learning_rate`: $[10^{-4}, 10^{-2}]$
- `weight_decay`: $[10^{-6}, 10^{-3}]$

4.7 Tạo Cross-Validation Folds

Để đảm bảo tính nhất quán và công bằng trong việc so sánh giữa các mô hình (Kohavi, 1995), chúng tôi tạo một file cross-validation folds tĩnh sử dụng `StratifiedKFold` từ Scikit-learn (Pedregosa et al., 2011):

Cấu hình:

- `n_splits = 5`: Đảm bảo khoảng 30 TDE mỗi fold từ 148 TDE tổng
- `shuffle = True`: Xáo trộn dữ liệu trước khi chia
- `random_state = 15`: Fixed seed để đảm bảo tái tạo được kết quả

Kết quả phân chia: **Lợi ích:** Tất cả các mô hình (XGBoost, LightGBM, CatBoost, ResNet) được huấn luyện trên **cùng các splits**, cho phép so sánh công bằng và kết hợp predictions một cách nhất quán.

Fold	TDE	Tổng mẫu	Tỷ lệ TDE
Fold 0	30	609	4,93%
Fold 1	30	609	4,93%
Fold 2	30	609	4,93%
Fold 3	29	608	4,77%
Fold 4	29	608	4,77%

Table 6: Phân phối TDE trong các folds.

4.8 Ensemble Methods

Phương pháp ensemble của chúng tôi kết hợp predictions từ 4 mô hình (XGBoost, LightGBM, CatBoost, ResNet) thông qua hai bước tối ưu.

Bước 1: Tối ưu Trọng số (Weighted Average) Sử dụng thuật toán `SLSQP` để tìm trọng số tối ưu cho mỗi mô hình, với mục tiêu **tối thiểu hóa Log Loss**:

$$\min_{w} \mathcal{L}_{log}\left(y, \sum_{i=1}^{n} w_i \cdot p_i\right) \quad \text{s.t.} \quad \sum_{i=1}^{n} w_i = 1,\ w_i \geq 0$$

trong đó p_i là predictions từ mô hình thứ i.

Mô hình	Trọng số tối ưu
XGBoost	0,4736
LightGBM	0,2264
CatBoost	0.1500
ResNet (NN)	0.1500

Table 7: Trọng số tối ưu của các mô hình trong ensemble.

Bước 2: Tối ưu Threshold Sau khi có weighted predictions, sử dụng `Precision-Recall Curve` để tìm threshold tối ưu cho **F1 Score cao nhất**:

$$\text{F1} = \frac{2 \cdot \text{Precision} \cdot \text{Recall}}{\text{Precision} + \text{Recall}}$$

Threshold tối ưu: 0,1940 (thấp hơn 0,5 mặc định do class imbalance).

5 Đánh giá Kết quả và Kết luận

5.1 Phân tích Tương quan Mô hình

Hình 1 minh họa sự tương quan giữa các mô hình và mức độ overlap trong việc phát hiện TDE. Việc đánh giá dựa trên nhiều chỉ số bao gồm AUC-ROC (Ling et al., 2003; Fawcett, 2006) và F1 Score.

Nhận xét:

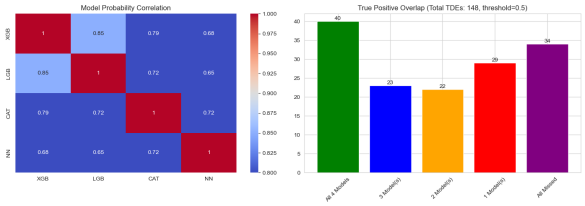

Figure 1: Phân tích tương quan giữa các mô hình: (trái) Ma trận tương quan probability, (phải) Overlap True Positive giữa các mô hình.

Mô hình	OOF F1	Threshold	Log Loss
XGBoost	0,5967	0,2859	–
LightGBM	0,58+	–	–
CatBoost	0,58+	–	–
ResNet (NN)	0,5417	0,6424	–
Ensemble	**0,6992**	**0,1940**	**0,1026**

Table 8: So sánh kết quả các mô hình. **Competition Public Score: 0,6992**.

- Các mô hình tree-based (XGBoost, LightGBM, CatBoost) có tương quan cao với nhau (>0,95)
- ResNet (NN) có tương quan thấp hơn với các mô hình tree-based, cho thấy nó "nhìn" dữ liệu theo cách khác
- Điều này giải thích tại sao ensemble giúp cải thiện hiệu suất

5.2 Kết quả Overlap với Threshold Tối ưu

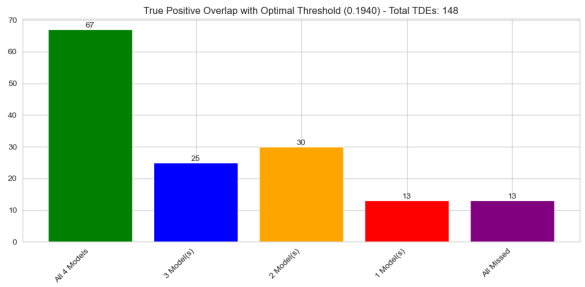

Figure 2: True Positive Overlap với threshold tối ưu (0,1940). Trong 148 TDE: 67 được tất cả 4 mô hình phát hiện, 25 được 3 mô hình, 30 được 2 mô hình, 13 được 1 mô hình, và 13 bị miss bởi tất cả.

5.3 Kết quả Cuộc thi

Nhận xét:

- Ensemble đạt **F1 Score 0,6992**, cao hơn mọi mô hình đơn lẻ

- XGBoost đóng góp lớn nhất (trọng số 47,36%)
- Threshold tối ưu thấp (0,1940) phản ánh tính chất mất cân bằng lớp

5.4 Kết quả Dự đoán trên Tập Test

Thống kê	Giá trị
Tổng số mẫu test	7.135
Dự đoán positive (TDE)	557
Dự đoán negative (non-TDE)	6.578
Tỷ lệ positive	7,81%

Table 9: Phân phối dự đoán trên tập test.

5.5 Kết luận

Báo cáo này trình bày phương pháp tiếp cận toàn diện cho bài toán phân loại TDE:

1. **Data Engineering**: Sử dụng Gaussian Process 2D để trích xuất 290 đặc trưng từ đường cong ánh sáng, bao gồm các đặc trưng về màu sắc, rise/decline, và variability.
2. **Modeling**: Kết hợp 4 mô hình (XGBoost, LightGBM, CatBoost, ResNet) với 5-fold Stratified Cross-Validation và Optuna hyperparameter tuning.
3. **Ensemble**: Tối ưu trọng số bằng Log Loss minimization và threshold bằng F1 maximization.
4. **Kết quả**: Đạt **OOF F1 Score: 0,6709** và **Competition Public Score: 0,6992**.

5.6 Hướng Phát triển

- Khám phá các kiến trúc neural network phức tạp hơn (Transformer, TabNet)
- Áp dụng các kỹ thuật augmentation cho dữ liệu time-series
- Kết hợp thêm các đặc trưng domain-specific từ kiến thức thiên văn học
- Sử dụng stacking (Wolpert, 1992) thay vì weighted average cho ensemble

Lời cảm ơn

Nghiên cứu này được thực hiện trong khuôn khổ môn học Học máy tại Trường Đại học Công nghệ (UET). Chúng tôi xin gửi lời cảm ơn chân thành đến Ban tổ chức cuộc thi MALLORN Astronomical Classification Challenge đã cung cấp bộ dữ liệu và khung đánh giá cho nghiên cứu này.

References

Takuya Akiba, Shotaro Sano, Toshihiko Yanase, Takeru Ohta, and Masanori Koyama. 2019. Optuna: A next-generation hyperparameter optimization framework. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 2623–2631.

James Bergstra, Rémi Bardenet, Yoshua Bengio, and Balázs Kégl. 2011. Algorithms for hyper-parameter optimization. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 24.

Leo Breiman. 1996. Bagging predictors. *Machine Learning*, 24(2):123–140.

Nitesh V. Chawla, Kevin W. Bowyer, Lawrence O. Hall, and W. Philip Kegelmeyer. 2002. SMOTE: Synthetic minority over-sampling technique. *Journal of Artificial Intelligence Research*, 16:321–357.

Tianqi Chen and Carlos Guestrin. 2016. XGBoost: A scalable tree boosting system. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 785–794.

Tom Fawcett. 2006. An introduction to ROC analysis. *Pattern Recognition Letters*, 27(8):861–874.

Suvi Gezari. 2021. Tidal disruption events. *Annual Review of Astronomy and Astrophysics*, 59:21–58.

Haibo He and Edwardo A. Garcia. 2009. Learning from imbalanced data. *IEEE Transactions on Knowledge and Data Engineering*, 21(9):1263–1284.

Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. 2016. Deep residual learning for image recognition. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 770–778.

Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. 2017. LightGBM: A highly efficient gradient boosting decision tree. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 30, pages 3146–3154.

Diederik P. Kingma and Jimmy Ba. 2014. Adam: A method for stochastic optimization. *arXiv preprint arXiv:1412.6980*.

Ron Kohavi. 1995. A study of cross-validation and bootstrap for accuracy estimation and model selection. 2:1137–1145.

Charles X. Ling, Jin Huang, and Harry Zhang. 2003. AUC: A better measure than accuracy in comparing learning algorithms. *Advances in Artificial Intelligence*, pages 329–341.

Adam Paszke, Sam Gross, Francisco Massa, Adam Lerer, James Bradbury, Gregory Chanan, Trevor Killeen, Zeming Lin, Natalia Gimelshein, Luca Antiga, and 1 others. 2019. PyTorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems (NeurIPS)*, volume 32, pages 8026–8037.

Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, and 1 others. 2011. Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Liudmila Prokhorenkova, Gleb Gusev, Aleksandr Vorobev, Anna Veronika Dorogush, and Andrey Gulin. 2018. CatBoost: unbiased boosting with categorical features. *Advances in Neural Information Processing Systems (NeurIPS)*, 31.

Martin J. Rees. 1988. Tidal disruption of stars by black holes of 10^6–10^8 solar masses in nearby galaxies. *Nature*, 333(6173):523–528.

Nitish Srivastava, Geoffrey Hinton, Alex Krizhevsky, Ilya Sutskever, and Ruslan Salakhutdinov. 2014. Dropout: A simple way to prevent neural networks from overfitting. volume 15, pages 1929–1958.

David H. Wolpert. 1992. Stacked generalization. *Neural Networks*, 5(2):241–259.