



ĐẠI HỌC QUỐC GIA TP. HỒ CHÍ MINH
TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN

XÂY DỰNG MÔ HÌNH MÁY HỌC



NỘI DUNG

I. Model Selection và Evaluation

- A. Holdout Validation
- B. Cross Validation
- C. Adversarial Validation
- D. Hyper-parameters Optimization
- E. Evaluation Metric

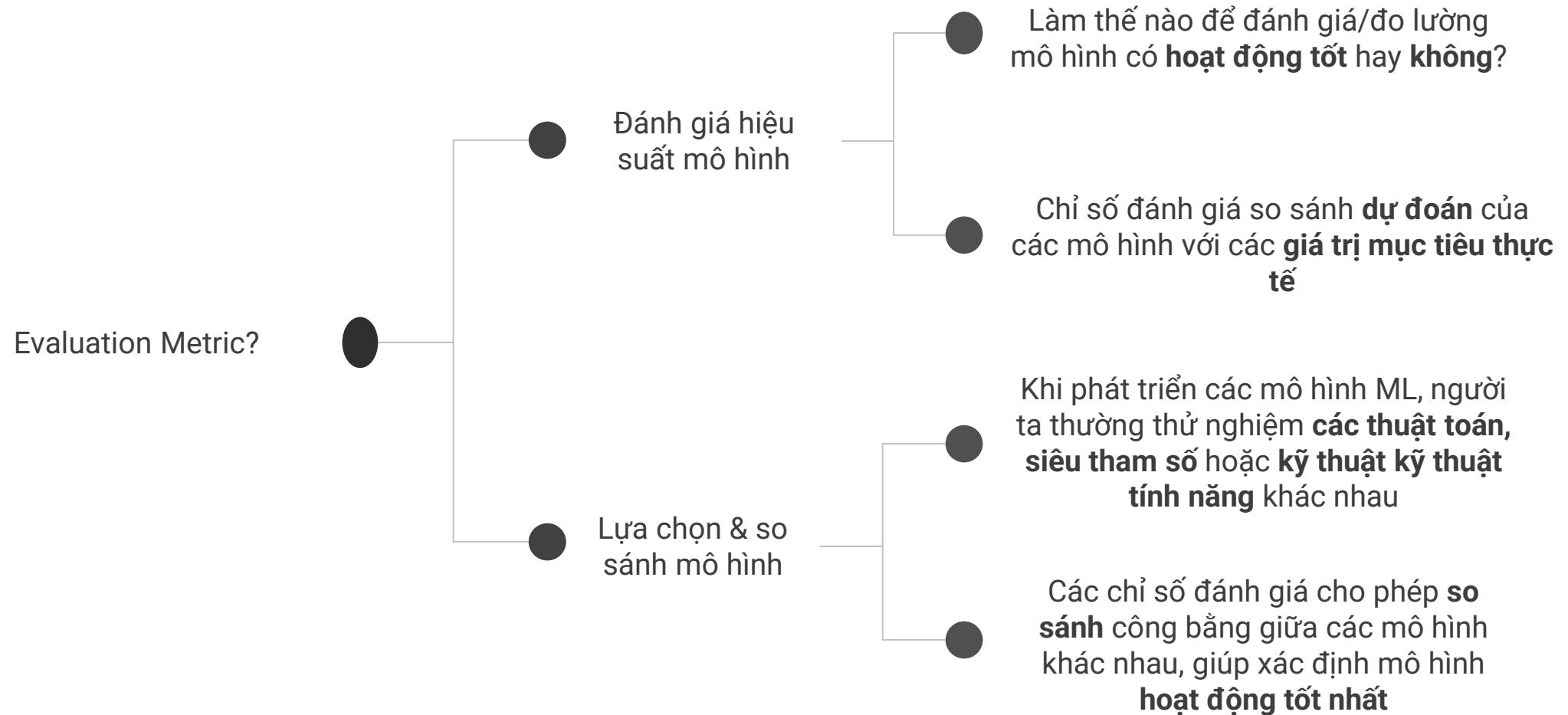
II. Handle Imbalanced Data Method

- A. Resampling
- B. Cost sensitive learning
- C. Tools

III. Error Analysis Model

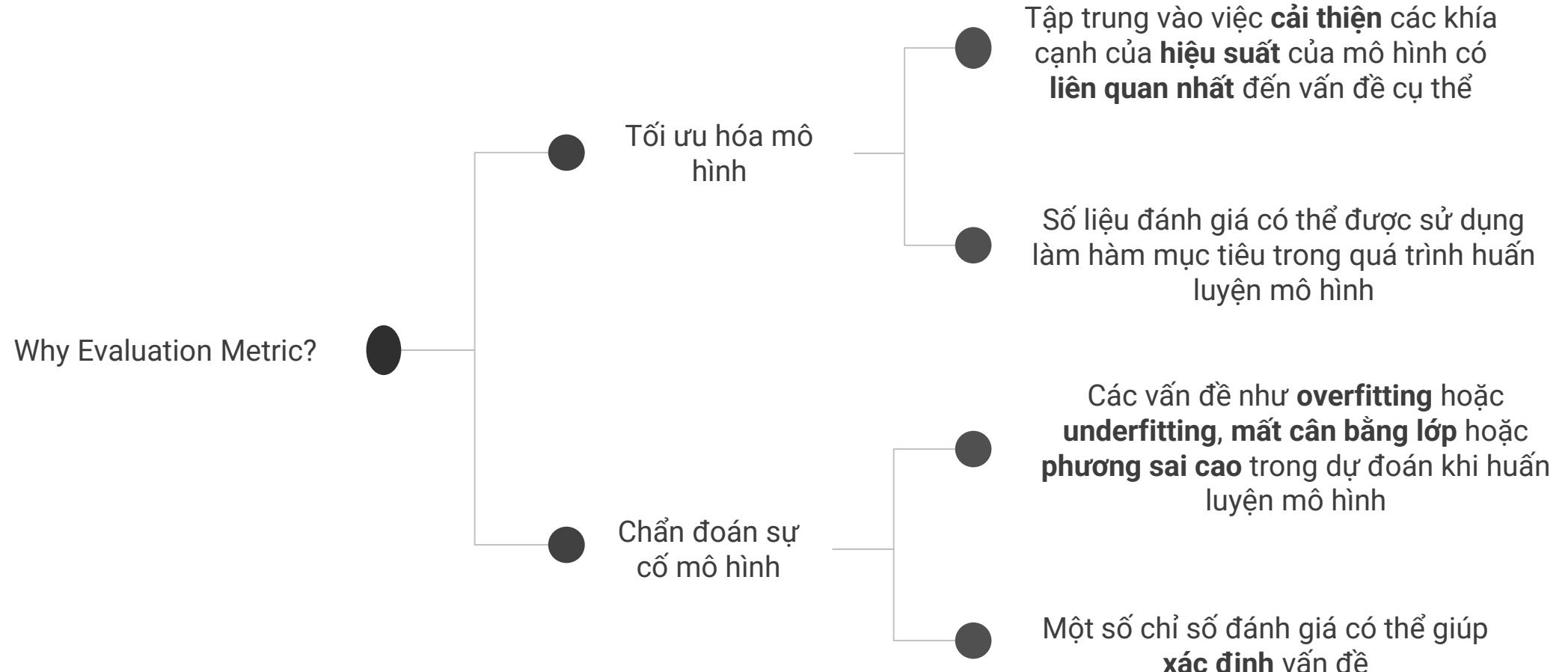


Tại sao lại dùng đánh giá số liệu?



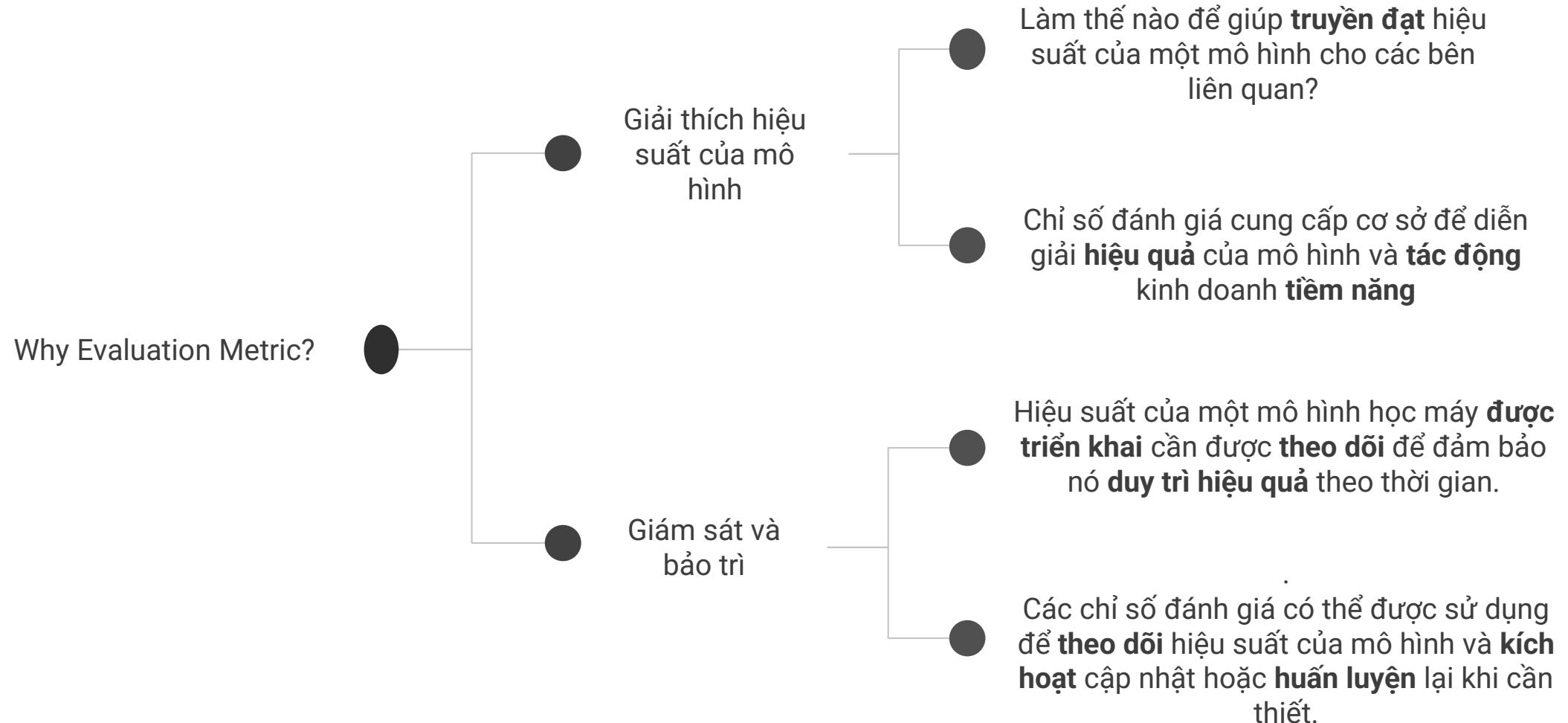


Tại sao Evaluation Metric?





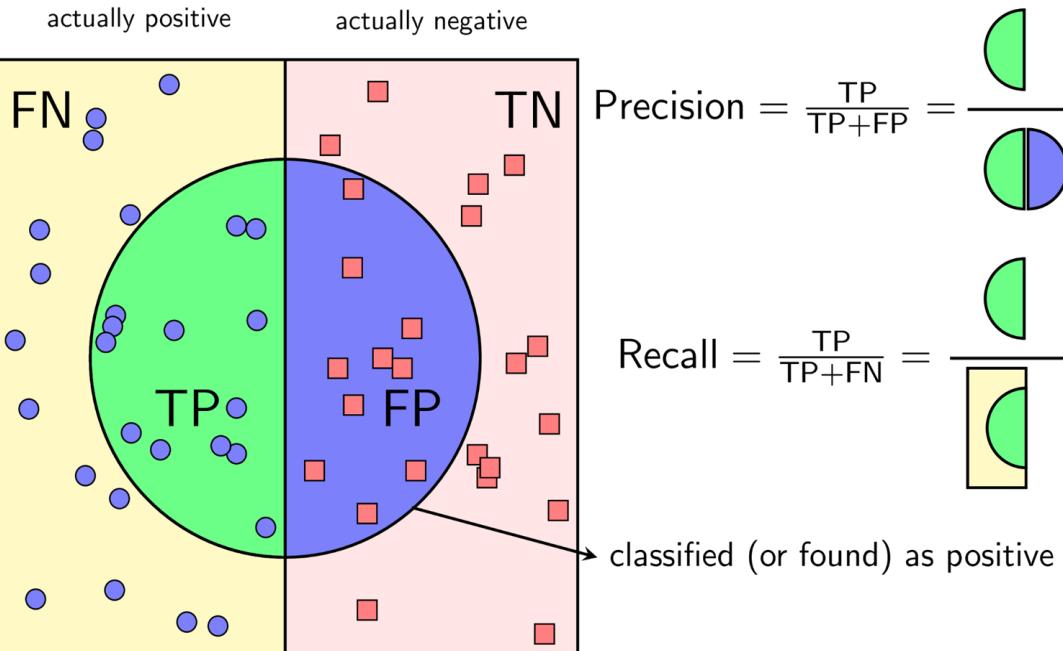
Tại sao Evaluation Metric?





Chỉ số đánh giá phân loại

☐ Một số chỉ số đánh giá phân loại



Metric	Formula	Evaluation Focus
Accuracy	$ACC = \frac{TP+TN}{TP+TN+FP+FN}$	Overall effectiveness of a classifier
Error rate	$ERR = \frac{FP+FN}{TP+TN+FP+FN}$	Classification error
Precision	$PRC = \frac{TP}{TP+FP}$	Class agreement of the data labels with the positive labels given by the classifier
Sensitivity	$SNS = \frac{TP}{TP+FN}$	Effectiveness of a classifier to identify positive labels
Specificity	$SPC = \frac{TN}{TN+FP}$	How effectively a classifier identifies negative labels
ROC	$ROC = \frac{\sqrt{SNS^2+SPC^2}}{\sqrt{2}}$	Combined metric based on the Receiver Operating Characteristic (ROC) space [53]
F_1 score	$F_1 = 2 \frac{PRC \cdot SNS}{PRC + SNS}$	Combination of precision (PRC) and sensitivity (SNS) in a single metric
Geometric Mean	$GM = \sqrt{SNS \cdot SPC}$	Combination of sensitivity (SNS) and specificity (SPC) in a single metric

<https://machinelearningcoban.com/2017/08/31/evaluation/>



Khi nào Precision > Recall?

- Sự lựa chọn giữa precision và recall thường phụ thuộc **vào vấn đề cụ thể** và chi phí của False positives so với false negative.
- Khi FP tồn kém hơn:
 - Phát hiện email spam: FP (một email quan trọng bị đánh dấu không chính xác là spam)
 - Lọc nội dung an toàn (dành cho trẻ em): tốt hơn hết là lọc sai một số nội dung an toàn (Lower recall) thay vì mạo hiểm để nội dung không phù hợp đi qua (lower precision)

Precision Score

F-Beta score (beta < 1)
F0.5 score

PR curve & AP



Khi nào Recall > Precision?

- Khi chi phí bỏ lỡ một mẫu positive (false negative) là cao
 - Chẩn đoán y tế: khi chẩn đoán một căn bệnh nguy hiểm như ung thư, chúng ta muốn phát hiện càng nhiều trường hợp thực sự mắc bệnh (true positive) càng tốt (cần xử lý một số FP → xét nghiệm thêm)
 - Phát hiện gian lận: bắt càng nhiều giao dịch gian lận càng tốt (high recall, sau đó xử lý FP theo yêu cầu thêm thông tin)

Recall Score

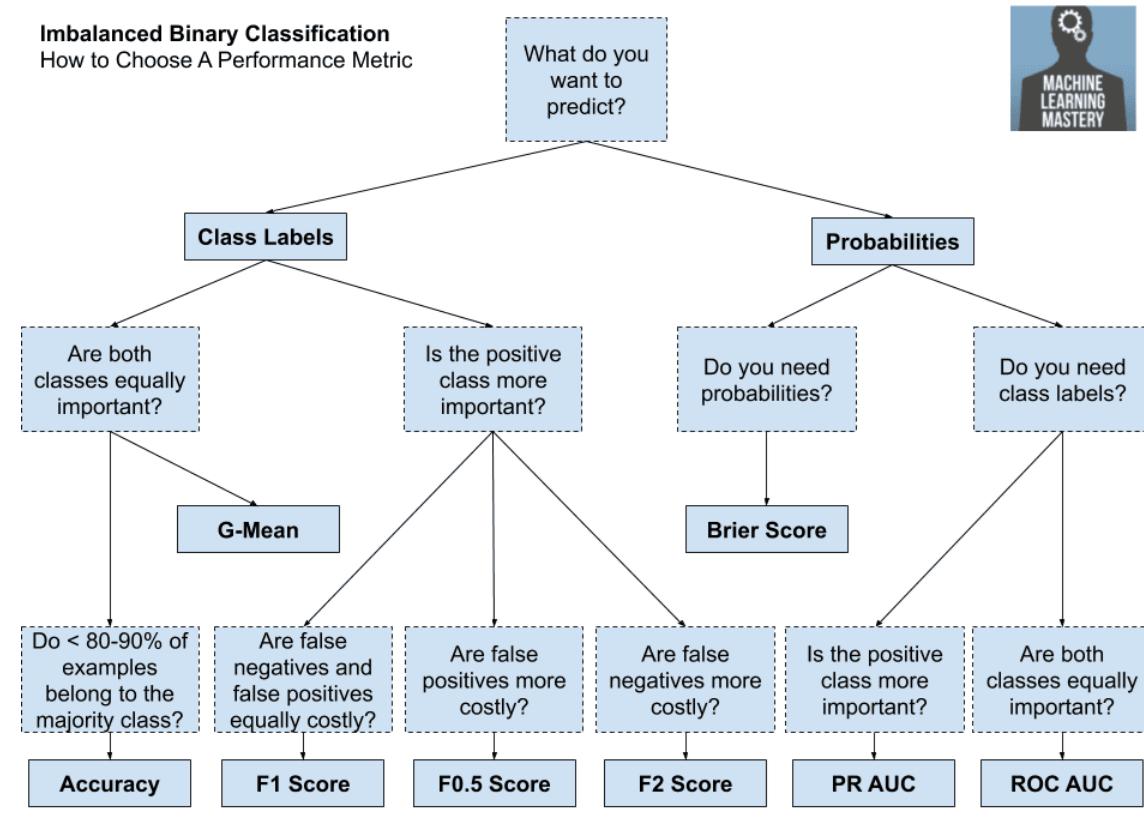
F-Beta score (beta > 1)
F2 score

ROC - AUC



Số liệu đánh giá cho Imbalanced Classification

- Imbalanced Classification: Chọn chỉ số phù hợp với trường hợp



© 2019 MachineLearningMastery.com All Rights Reserved.

<https://machinelearningmastery.com/tour-of-evaluation-metrics-for-imbalanced-classification/>



Chỉ số cho bài toán hồi quy

- ☐ Một số chỉ số hồi quy:

Example Use Case	Error Metric	Formula	Notes
Which model best captures the rapid changes in the volatile stock market?	1.1 Mean squared error (MSE)	$\frac{1}{n} \sum_{i=1}^n (y_i - f(x_i))^2$	Weights big differences more
	1.2 Root Mean squared error (RMSE)	$\sqrt{\frac{1}{n} \sum_{i=1}^n (y_i - f(x_i))^2}$	
Which model best estimates the energy consumption in the long term?	2. Mean absolute error (MAE)	$\frac{1}{n} \sum_{i=1}^n y_i - f(x_i) $	Equal weights to all distances
Are the sales forecasting models for different products equally accurate?	3. Mean absolute percentage error (MAPE)	$\frac{1}{n} \sum_{i=1}^n \frac{ y_i - f(x_i) }{ y_i }$	Requires non-zero target column values
Does a running app provide unrealistic expectations?	4. Mean signed difference	$\frac{1}{n} \sum_{i=1}^n (y_i - f(x_i))$	Only informative about the direction of the error
How much of our years of education can be explained through access to literature?	5. R-squared	$1 - \frac{\sum_{i=1}^n (y_i - f(x_i))^2}{\sum_{i=1}^n (y_i - \bar{y})^2}$	Universal range: the closer to 1 the better



Sklearn: Metrics

❑ sklearn.metrics

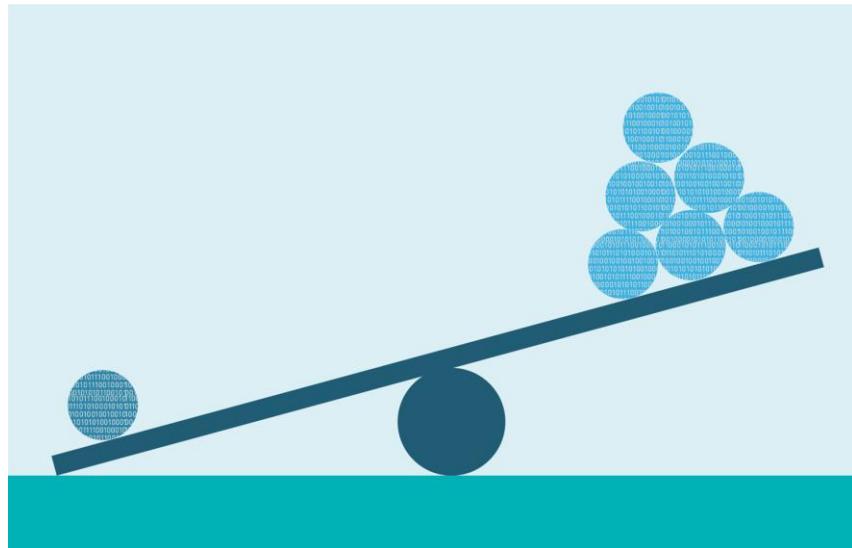
Scoring	Function
Classification	
'accuracy'	<code>metrics.accuracy_score</code>
'balanced_accuracy'	<code>metrics.balanced_accuracy_score</code>
'top_k_accuracy'	<code>metrics.top_k_accuracy_score</code>
'average_precision'	<code>metrics.average_precision_score</code>
'neg_brier_score'	<code>metrics.brier_score_loss</code>
'f1'	<code>metrics.f1_score</code>
'f1_micro'	<code>metrics.f1_score</code>
'f1_macro'	<code>metrics.f1_score</code>
'f1_weighted'	<code>metrics.f1_score</code>
'f1_samples'	<code>metrics.f1_score</code>
'neg_log_loss'	<code>metrics.log_loss</code>
'precision' etc.	<code>metrics.precision_score</code>
'recall' etc.	<code>metrics.recall_score</code>
'jaccard' etc.	<code>metrics.jaccard_score</code>
'roc_auc'	<code>metrics.roc_auc_score</code>
'roc_auc_ovr'	<code>metrics.roc_auc_score</code>
'roc_auc_ovo'	<code>metrics.roc_auc_score</code>
'roc_auc_ovr_weighted'	<code>metrics.roc_auc_score</code>
'roc_auc_ovo_weighted'	<code>metrics.roc_auc_score</code>

Regression	
'explained_variance'	<code>metrics.explained_variance_score</code>
'max_error'	<code>metrics.max_error</code>
'neg_mean_absolute_error'	<code>metrics.mean_absolute_error</code>
'neg_mean_squared_error'	<code>metrics.mean_squared_error</code>
'neg_root_mean_squared_error'	<code>metrics.mean_squared_error</code>
'neg_mean_squared_log_error'	<code>metrics.mean_squared_log_error</code>
'neg_median_absolute_error'	<code>metrics.median_absolute_error</code>
'r2'	<code>metrics.r2_score</code>
'neg_mean_poisson_deviance'	<code>metrics.mean_poisson_deviance</code>
'neg_mean_gamma_deviance'	<code>metrics.mean_gamma_deviance</code>
'neg_mean_absolute_percentage_error'	<code>metrics.mean_absolute_percentage_error</code>
'd2_absolute_error_score'	<code>metrics.d2_absolute_error_score</code>
'd2_pinball_score'	<code>metrics.d2_pinball_score</code>
'd2_tweedie_score'	<code>metrics.d2_tweedie_score</code>



Imbalanced Data

- Dữ liệu mất cân bằng (imbalanced data) đề cập đến một tình huống trong đó sự phân bố của lớp đầu ra **không bằng nhau** hoặc **sai lệch đáng kể**
- Mô hình có xu hướng thiên về **lớp đa số** → hiệu suất kém trong lớp thiểu số





Phương pháp xử lý dữ liệu mất cân bằng

Resampling

Cost Sensitive Learning
(Class Weighted)

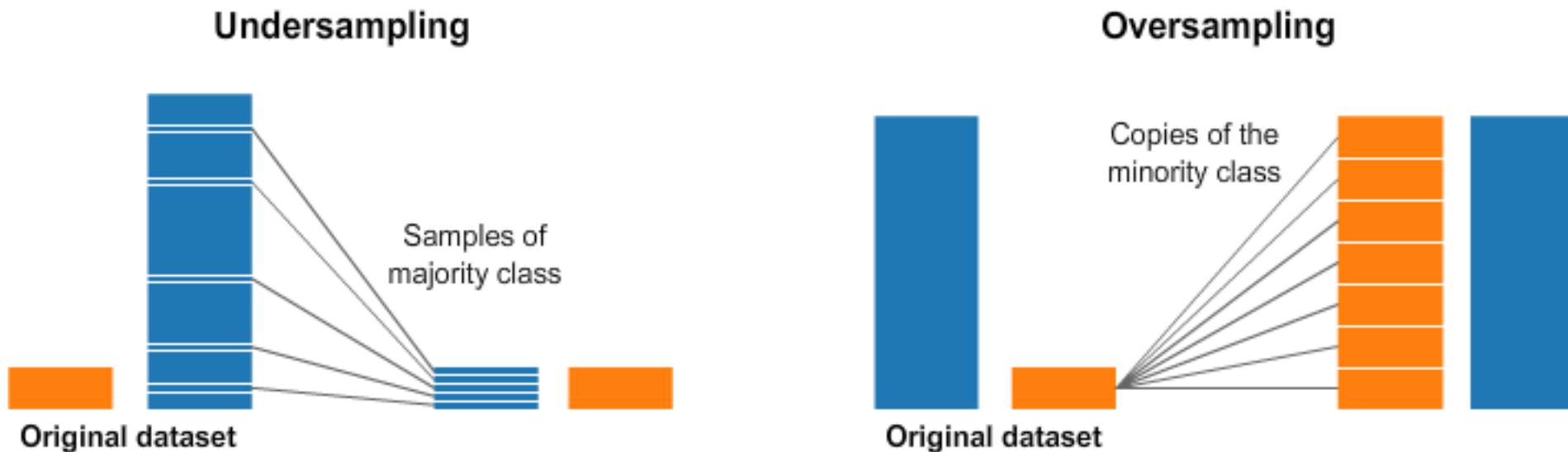
One-class Learning

Ensemble Method



Resampling

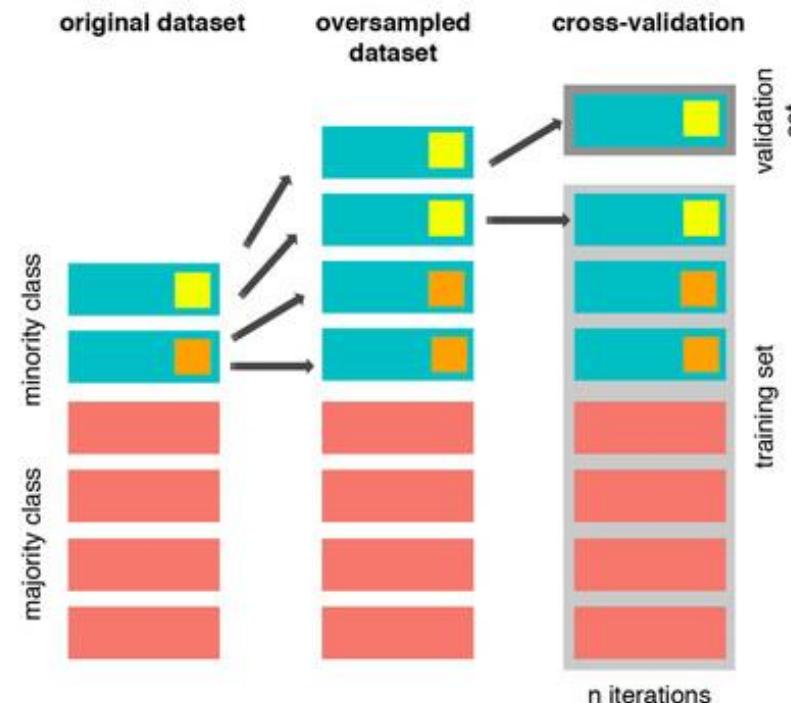
- ❑ Cách phổ biến nhất để xử lý dữ liệu mất cân bằng là Resampling (lấy mẫu lại)
- ❑ Có hai phương pháp chính: undersampling và oversampling



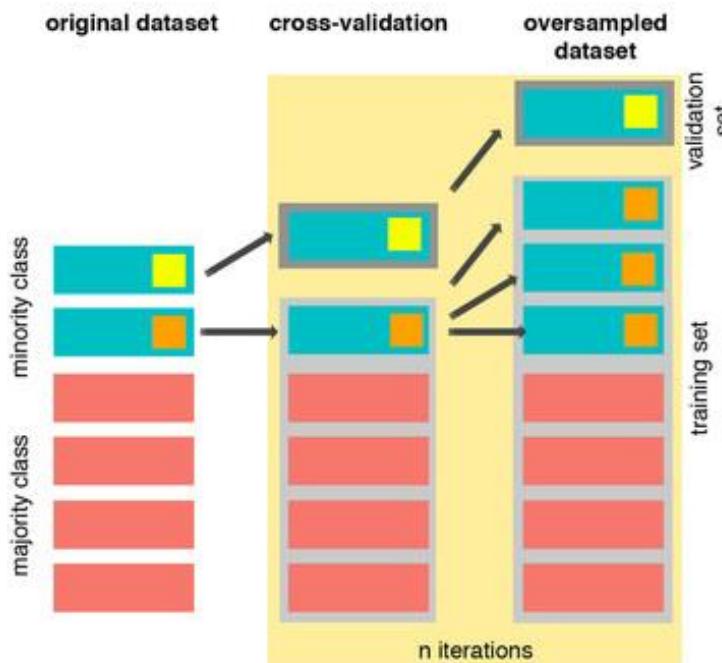


Overfitting trong quá trình Cross Validation

- Hãy cẩn thận với vấn đề "Rò rỉ dữ liệu" (data leakage) → Overfitting



The Wrong Way



The Right Way



Imbalanced-learn

- github: [imbalanced learn scikit learn](#)
- Example: [example code](#)
- Documentation: [introduction](#)

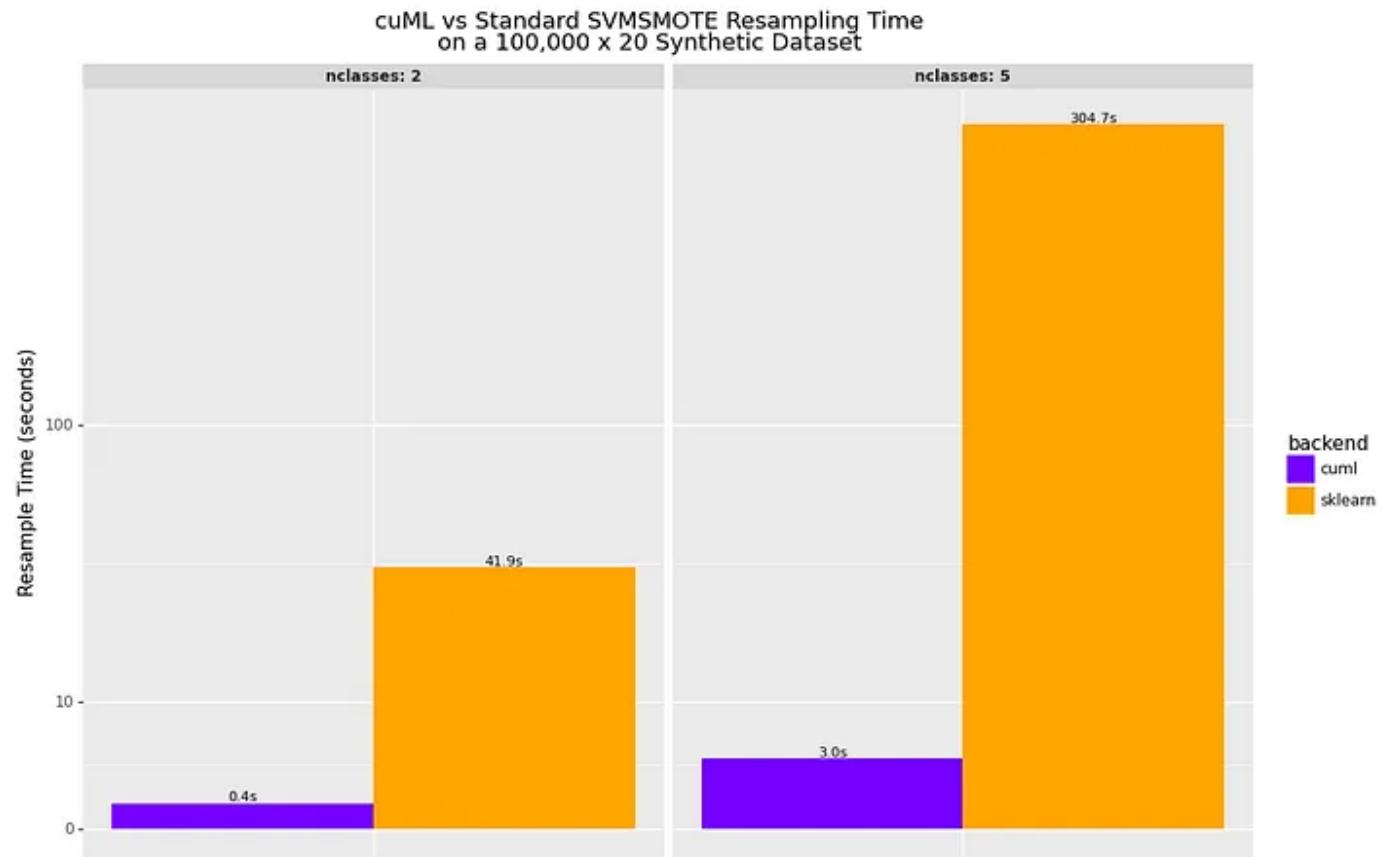
```
from imblearn.over_sampling import SVMSMOTE
from sklearn.datasets import make_classification

X, y = make_classification(
    n_samples=100000,
    n_features=100,
    n_redundant=0,
    n_informative=100,
    n_classes=5,
    n_clusters_per_class=1,
    weights=[0.8, 0.05, 0.05, 0.05, 0.05]
)

X_resampled, y_resampled = SVMSMOTE().fit_resample(X, y)
```



Faster Resampling with Imbalance-learn + CuML





Rapids Imbalanced Learn

- cuML thường cung cấp tốc độ gấp 5-15 lần cho các kỹ thuật như SMOTE, ADASYN, và EditedNearestNeighbours và tăng tốc 100–300x cho các kỹ thuật như SVMSMOTE và CondensedNearestNeighbor.
- Cài đặt: [link](#)

```
# conda/mamba
mamba create -n rapids-imblearn -c rapidsai -c conda-forge -c nvidia rapids=23.0
mamba activate rapids-imblearn

# pip
python3.8 -m venv rapids-imblearn
source rapids-imblearn/bin/activate
pip install --upgrade pip
pip install cuml-cu11 --extra-index-url=https://pypi.nvidia.com
pip install imbalanced-learn
```



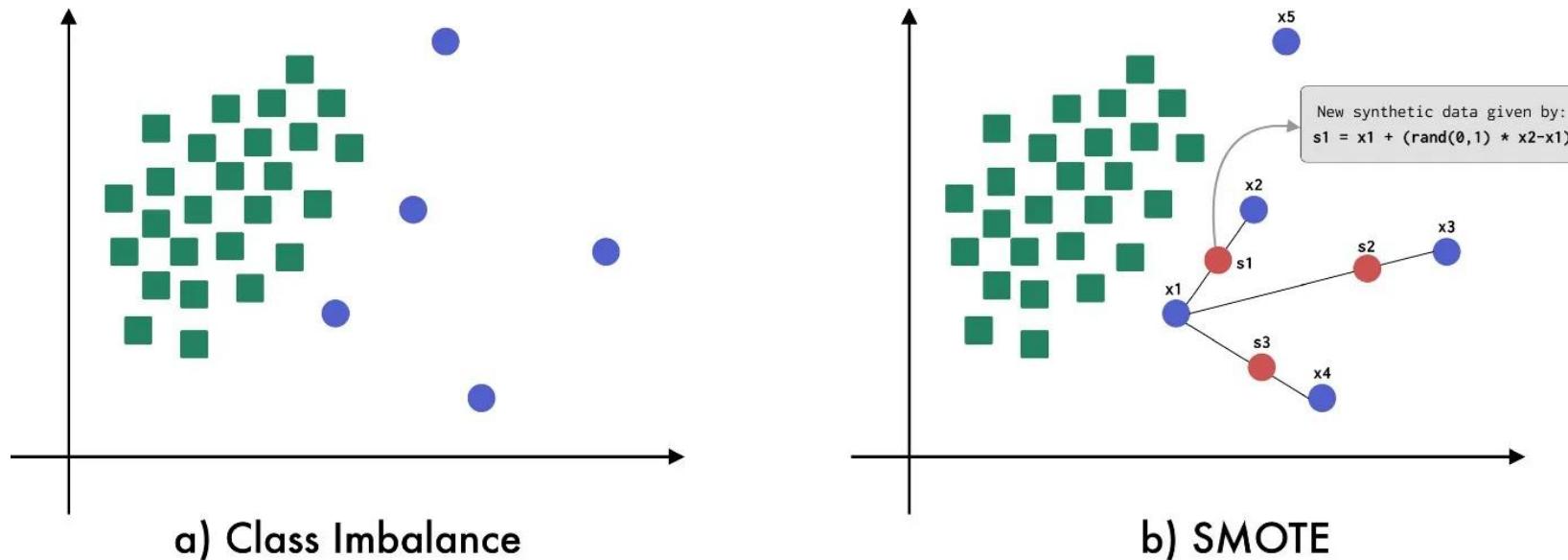
Oversampling method

- Có một số kỹ thuật phổ biến:
 - SMOTE
 - SVMSMOTE
 - ADASYN



SMOTE

- SMOTE (Synthetic Minority Oversampling Technique) là một kỹ thuật oversampling được sử dụng để tạo dữ liệu tổng hợp cho các lớp ít dữ liệu.



<https://varshasaini.in/glossary/smote/>



Hạn chế của SMOTE

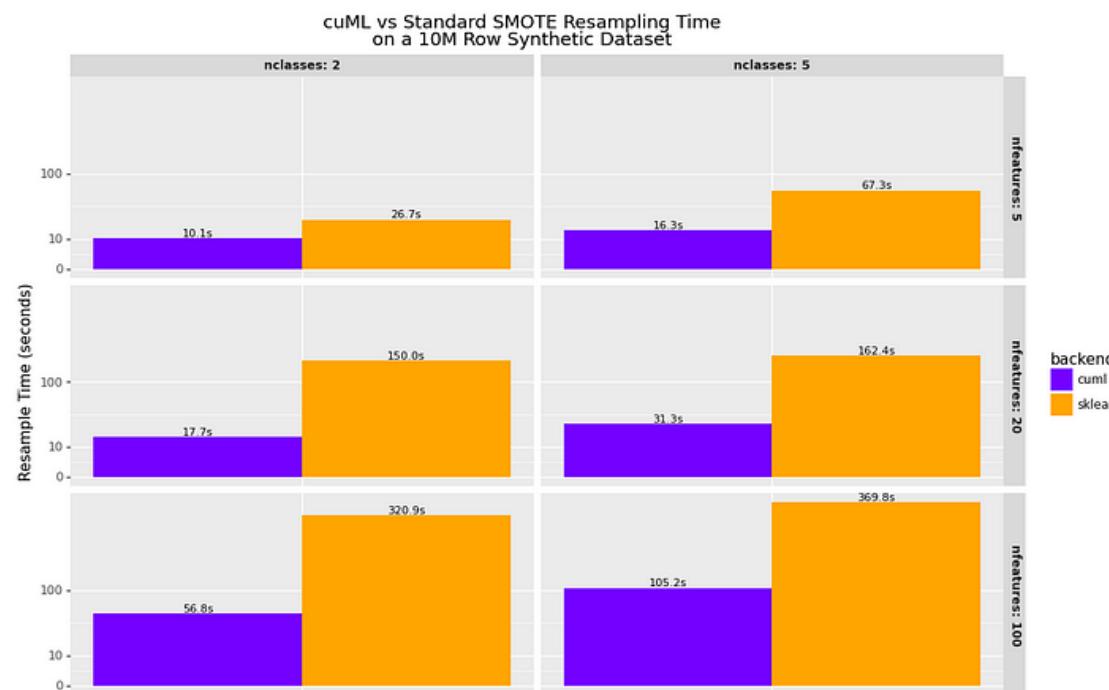
Nó chỉ hoạt động với dữ liệu liên tục, nó không được thiết kế để tạo dữ liệu tổng hợp phân loại

Dữ liệu được tạo ra phụ thuộc tuyến tính, có thể gây ra sự sai lệch trong dữ liệu được tạo và tạo ra overfitted model.



SMOTE

□ SMOTE (sklearn & cuML)



```
from imblearn.over_sampling import SMOTE
from sklearn.datasets import make_classification
from cuml.neighbors import NearestNeighbors

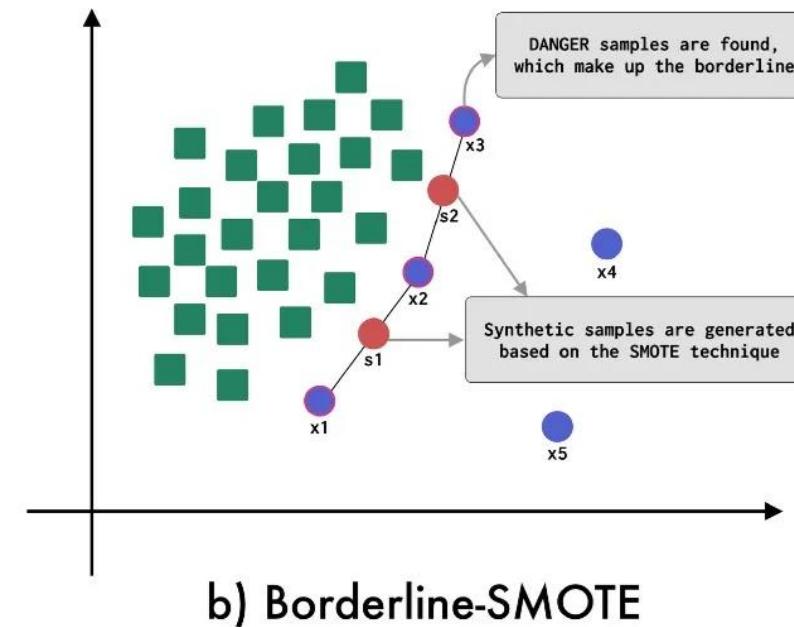
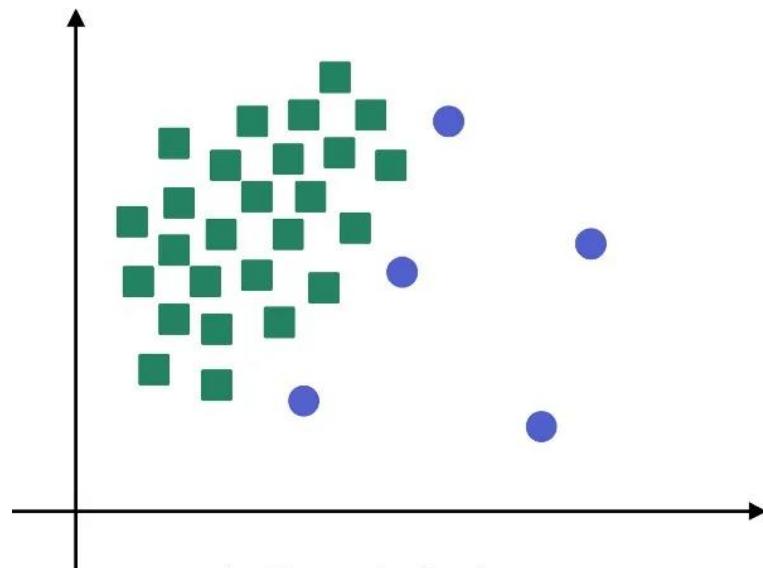
X, y = make_classification(
    n_samples=10000000,
    n_features=20,
    n_redundant=0,
    n_informative=100,
    n_classes=2,
    n_clusters_per_class=1,
    weights=[0.9, 0.1]
)

nn = NearestNeighbors(n_neighbors=6)
X_resampled, y_resampled = SMOTE(n_neighbors=nn).fit_resample(X, y)
```



SVMSMOTE

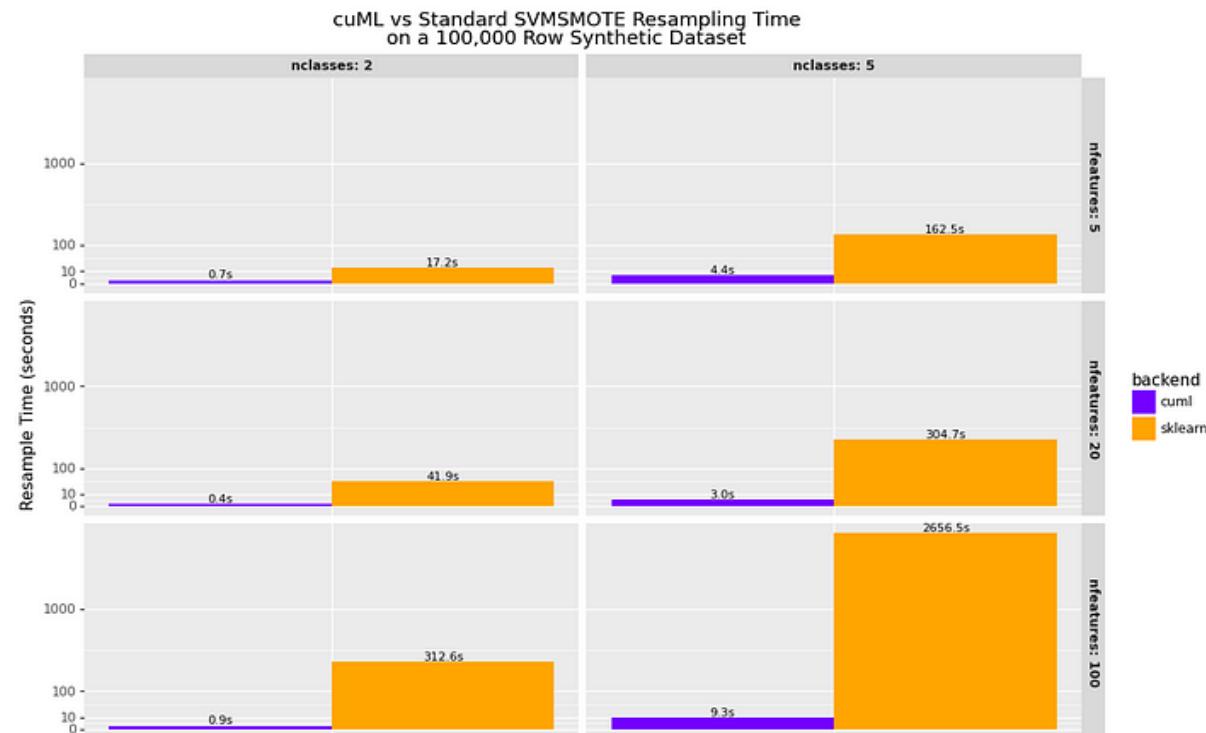
- ❑ Borderline-SMOTE là thuật toán dùng SVM dụng thay vì KNN để xác định các mẫu được phân loại sai trên ranh giới quyết định





SVMSMOTE

- Tăng hiệu suất đáng kể trên một số lớp và đặc trưng khác nhau, từ 50x đến 340x nếu sử dụng cuML.



```
from imblearn.over_sampling import SVMSMOTE
from sklearn.datasets import make_classification
from cuml.neighbors import NearestNeighbors
from cuml.svm import SVC

X, y = make_classification(
    n_samples=100000,
    n_features=100,
    n_redundant=0,
    n_informative=100,
    n_classes=5,
    n_clusters_per_class=1,
    weights=[0.8, 0.05, 0.05, 0.05, 0.05]
)

nn = NearestNeighbors(n_neighbors=6)
svm = SVC()
X_resampled, y_resampled = SVMSMOTE(
    k_neighbors=nn,
    m_neighbors=nn,
    svm_estimator=svm
).fit_resample(X, y)
```



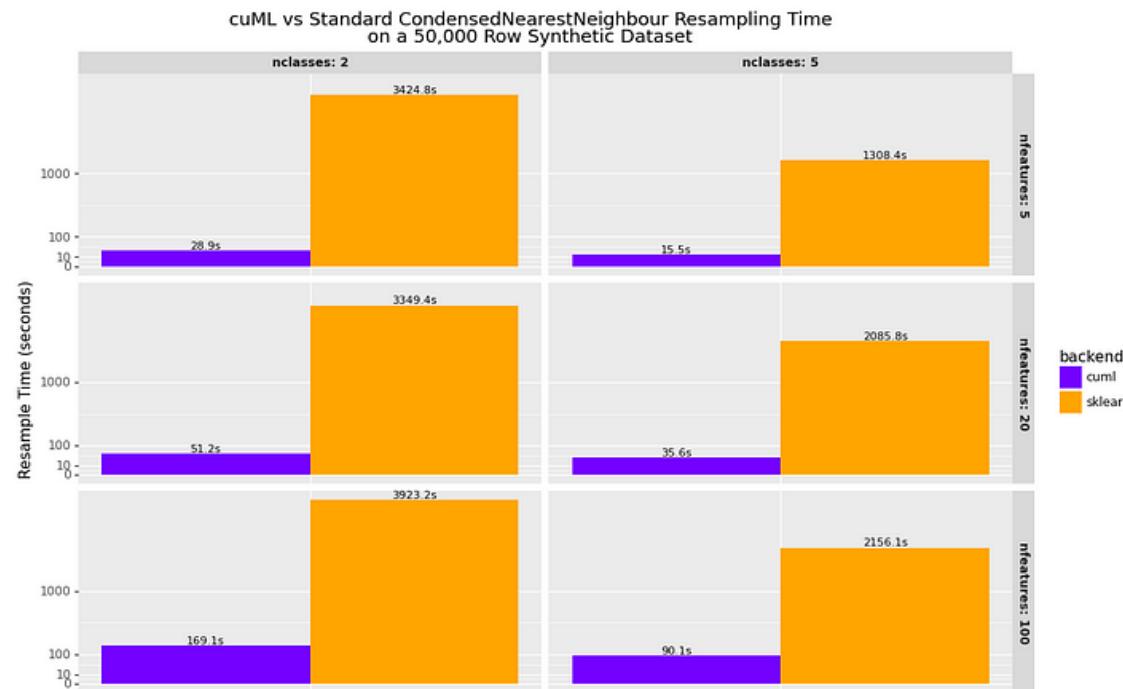
Phương pháp Undersampling

- Có một số kỹ thuật phổ biến:
 - Condensed Nearest Neighbour
 - Edited Nearest Neighbour



Condensed Nearest Neighbour

- Chỉ trên 50.000 hàng, kết quả điểm chuẩn cho thấy cuML có thể cung cấp tốc độ gấp 50-100 lần



```
from imblearn.under_sampling import CondensedNearestNeighbour
from sklearn.datasets import make_classification
from cuml.neighbors import KNeighborsClassifier

X, y = make_classification(
    n_samples=50000,
    n_features=20,
    n_redundant=0,
    n_informative=100,
    n_classes=2,
    n_clusters_per_class=1,
    weights=[0.9, 0.1]
)

knn = KNeighborsClassifier(n_neighbors=1)
cnn = CondensedNearestNeighbour(n_neighbors=knn)
cnn.estimator_ = knn # extra step for this resampler
X_res, y_res = cnn.fit_resample(X, y)
```



Cost Sensitive Learning

- Cost-sensitive learning là một lĩnh vực con của học máy giải quyết các vấn đề phân loại trong đó **misclassification costs** không bằng nhau
- Python sklearn cung cấp hỗ trợ học tập nhạy cảm với chi phí cho hầu hết các trình phân loại cơ sở nhờ tham số `class_weight`

None

The misclassification costs are set to 1
(default)

balanced

Các chi phí được đặt theo tỷ lệ mất cân bằng IR
($n_{\text{minority}}/n_{\text{majority}}$)

`class_weight = {0: IR; 1: 1}`

{0: c10; 1: c01}

Chi phí phân loại sai
được thiết lập rõ ràng
cho hai lớp bằng
dictionary

`class_weight = {0: 1; 1: 5}`



Tree-based: scale pos weight parameter

Notebook example

XGBoost

- `scale_pos_weight` [default=1]

○ Control the balance of positive and negative weights, useful for unbalanced classes. A typical value to consider: `sum(negative instances) / sum(positive instances)`. See [Parameters Tuning](#) for more discussion. Also, see Higgs Kaggle competition demo for examples: [R](#), [py1](#), [py2](#), [py3](#).

[auto_class_weights](#)

Command-line: `--auto-class-weights`

Automatically calculate class weights based either on the total weight or the total number of objects in each class. The values are used as multipliers for the object weights.

Supported values:

- None — All class weights are set to 1
- Balanced:

$$CW_k = \frac{\max_{c=1}^K (\sum_{t_i=c} w_i)}{\sum_{t_i=k} w_i}$$

- SqrtBalanced:

$$CW_k = \sqrt{\frac{\max_{c=1}^K (\sum_{t_i=c} w_i)}{\sum_{t_i=k} w_i}}$$

CatBoost

[scale_pos_weight](#)

The weight for class 1 in binary classification. The value is used as a multiplier for the weights of objects from class 1.



Imbalance XGBoost

- Tân dụng Weighted và Focal Losses cho Binary Label-Imbalanced Classification với XGBoost
- [Paper](#)
- [github](#)

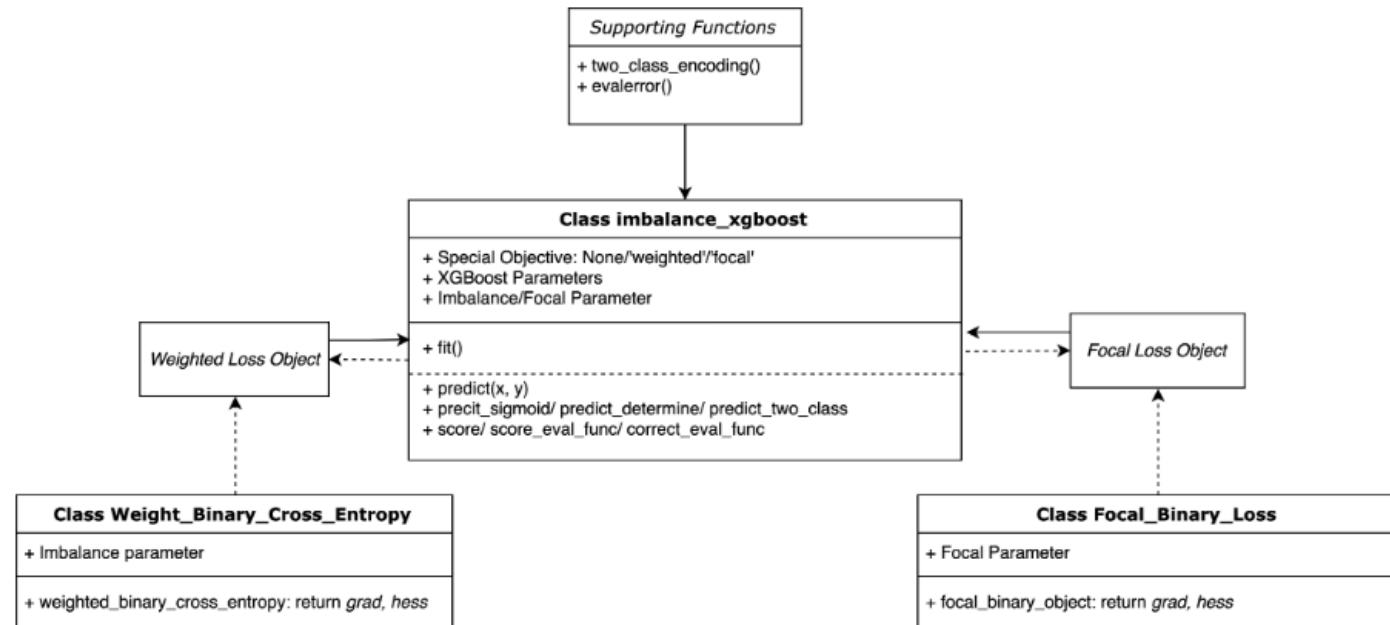
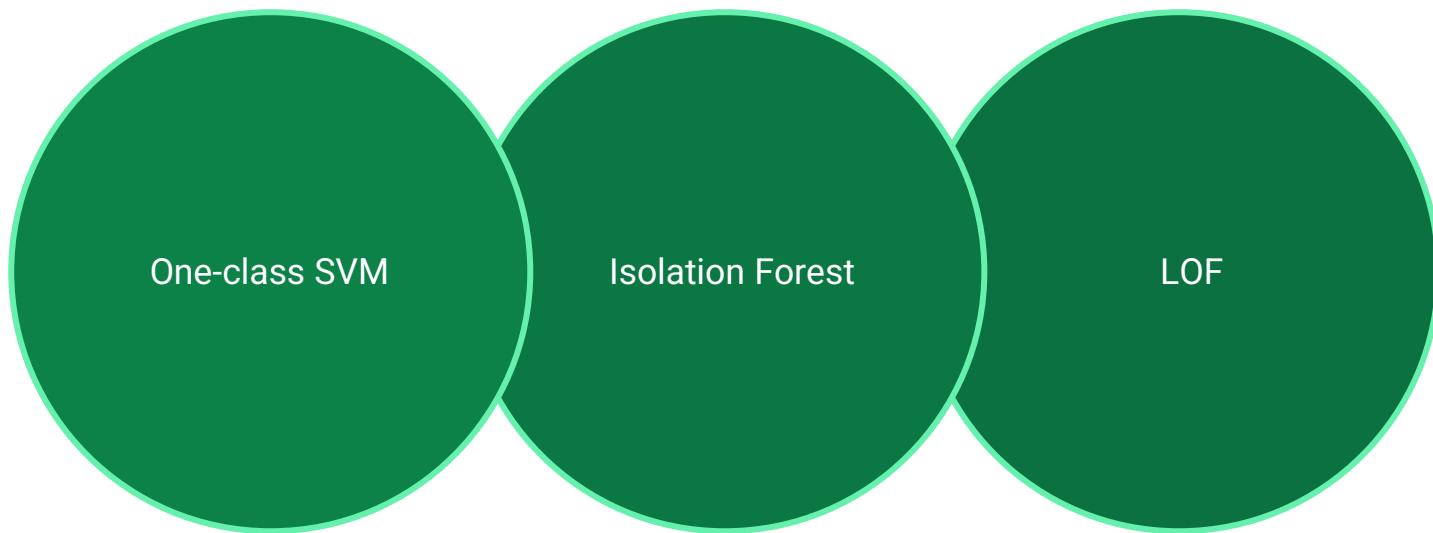


Figure 1: The Overall Structure of the Program



One-class Classification

- Có thể được sử dụng cho các bài toán phân loại **mất cân bằng** nhị phân (hai lớp) trong đó trường hợp âm tính (lớp 0) được coi là "bình thường" và trường hợp dương tính (lớp 1) được coi là **ngoại lệ hoặc bất thường**
- Trường hợp số trường hợp dương tính trong bộ đào tạo quá ít đến mức chúng không có giá trị đưa vào mô hình, chẳng hạn như **vài chục ví dụ** hoặc ít hơn





Error Analysis Model

- Phân tích lỗi thúc đẩy sâu hơn để cung cấp sự hiểu biết tốt hơn về các hành vi của mô hình học máy của bạn.
 - Xem xét các mẫu mà thuật toán của bạn **phân loại sai** thành nguyên nhân cơ bản của **lỗi**
 - Xác định các nhóm có đặc trưng giống nhau và **có tỷ lệ lỗi cao hơn** để chẩn đoán **nguyên nhân gốc rễ** đằng sau những lỗi này



Error Analysis: Approach

Learning Curves

Adversarial Test

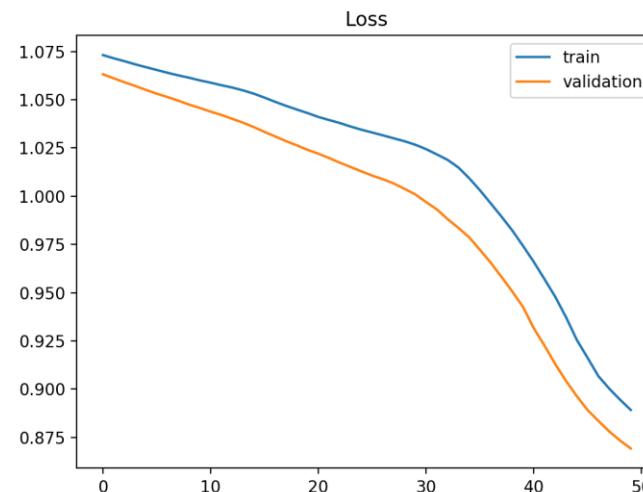
Confusion Matrix

Error Analysis Tools
Knowledge Domain

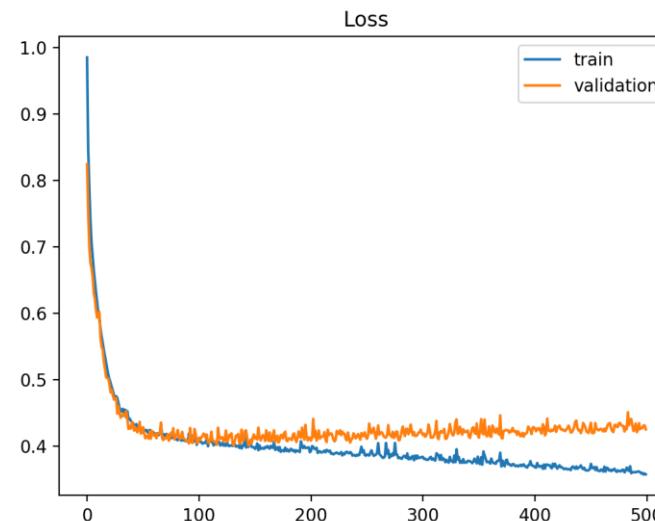


Learning Curves: Phát hiện hành vi mô hình

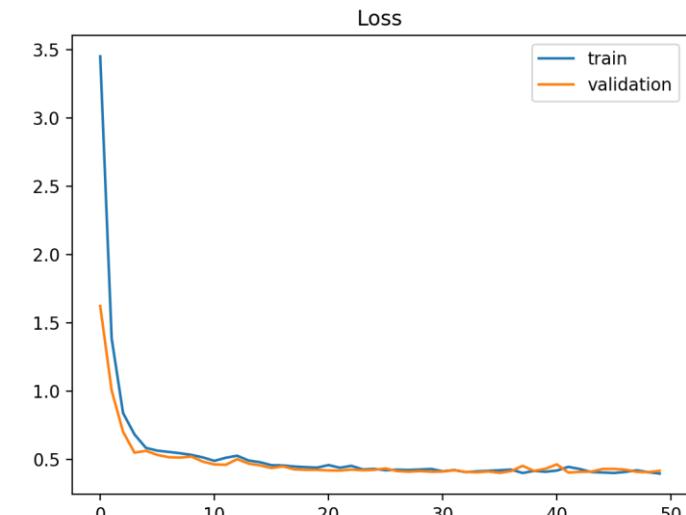
- Optimization Learning Curves (ex: MSE) & Performance Learning Curves (ex: F1 score)
- Vẽ biểu đồ hiệu suất của mô hình trên cả bộ dữ liệu huấn luyện và xác thực trong quá trình huấn luyện



Underfitting



Overfitting



Good fit

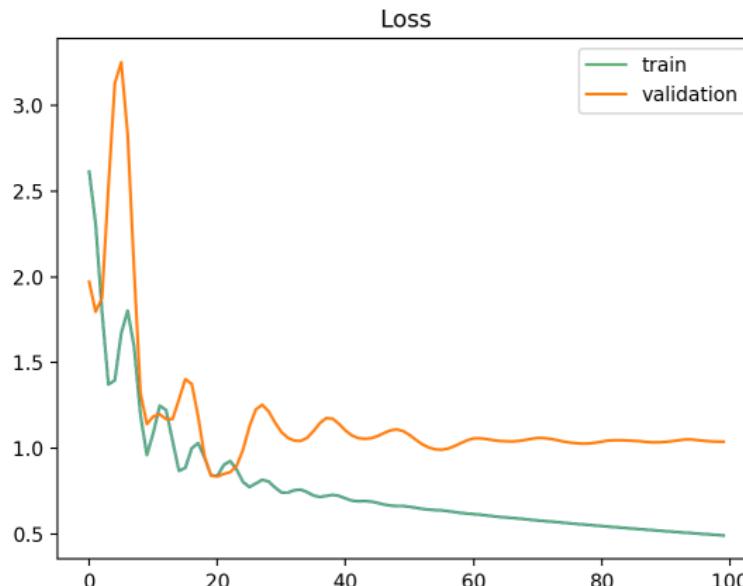
<https://machinelearningmastery.com/learning-curves-for-diagnosing-machine-learning-model-performance/>



Learning Curves: Detect Representativeness

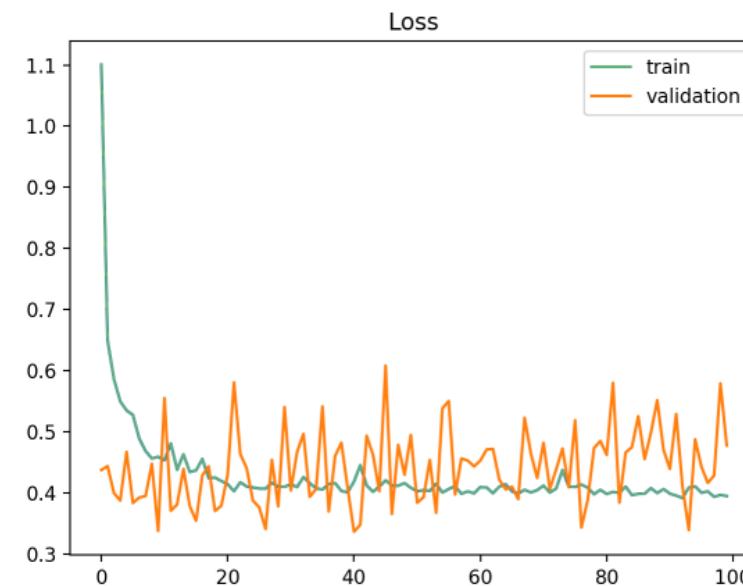
- Một **tập dữ liệu đại diện** phản ánh các **đặc điểm thống kê** tương ứng trong một tập dữ liệu khác từ cùng một miền

Unrepresentative Training Dataset

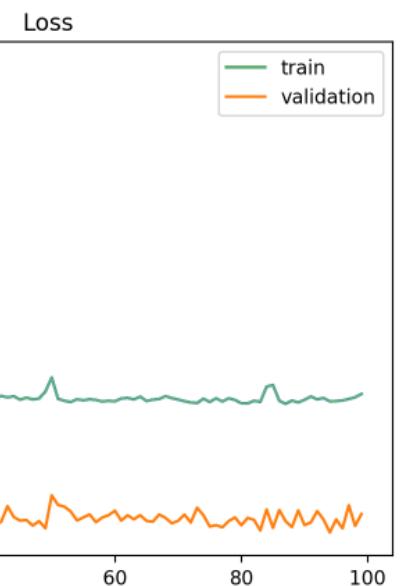


Khi dữ liệu có sẵn trong quá trình huấn luyện **không đủ để nắm bắt** mô hình, liên quan đến tập dữ liệu xác thực. **Khoảng cách lớn** giữa đường cong huấn luyện và đường cong xác thực.

Unrepresentative Validation Dataset



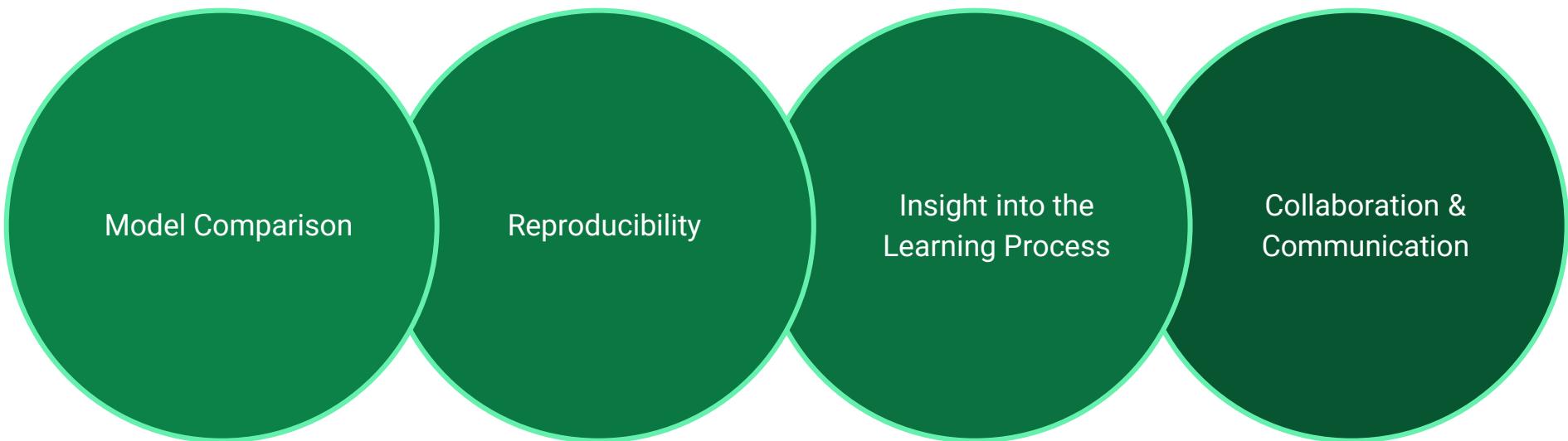
Dữ liệu xác thực **khan hiếm** và không **đại diện** cho dữ liệu huấn luyện, vì vậy mô hình gặp **khó khăn** để mô hình hóa các mẫu này (đường cong xác thực di chuyển **giao động mạnh**)



Các đường cong xác thực **tốt hơn nhiều** so với đường cong huấn luyện, điều này phản ánh tập dữ liệu xác thực **để dự đoán** hơn tập dữ liệu huấn luyện



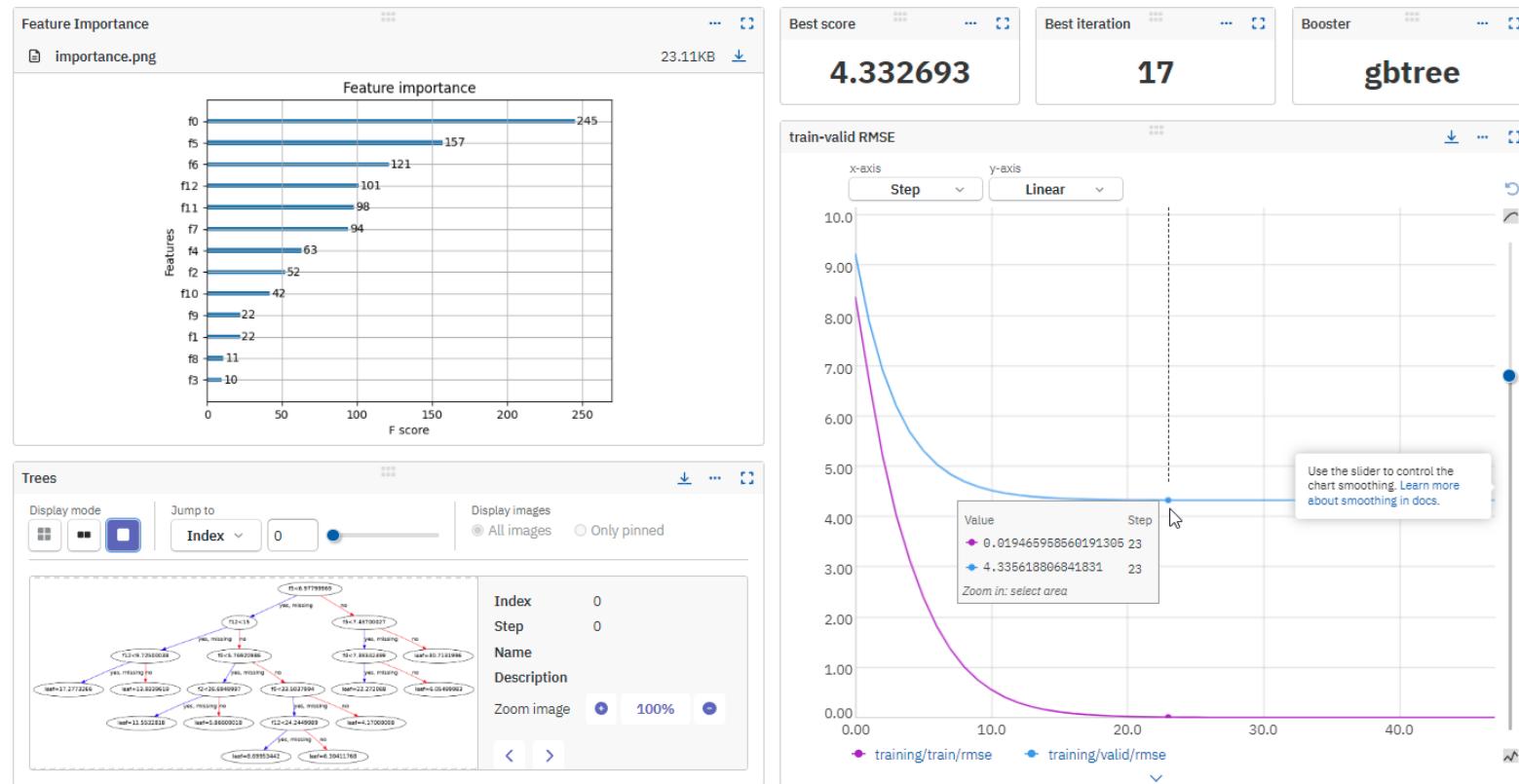
Theo dõi thực nghiệm





Neptune.ai

- ☐ Neptune là một kho siêu dữ liệu cung cấp tính năng **theo dõi thử nghiệm và đăng ký mô hình** cho các nhà nghiên cứu và kỹ sư học máy
- ☐ Với Neptune, bạn có thể ghi log, truy vấn, quản lý, hiển thị và so sánh tất cả siêu dữ liệu mô hình
- ☐ [Documentations](#)





Neptune.ai: Ví dụ

- [Details logs](#)
- [Colab Example](#)

```
import neptune
from neptune.integrations.xgboost import NeptuneCallback

# Create run
run = neptune.init_run()

# Create neptune callback
neptune_callback = NeptuneCallback(run=run, log_tree=[0, 1, 2, 3])

# Prepare data, params, etc.
...

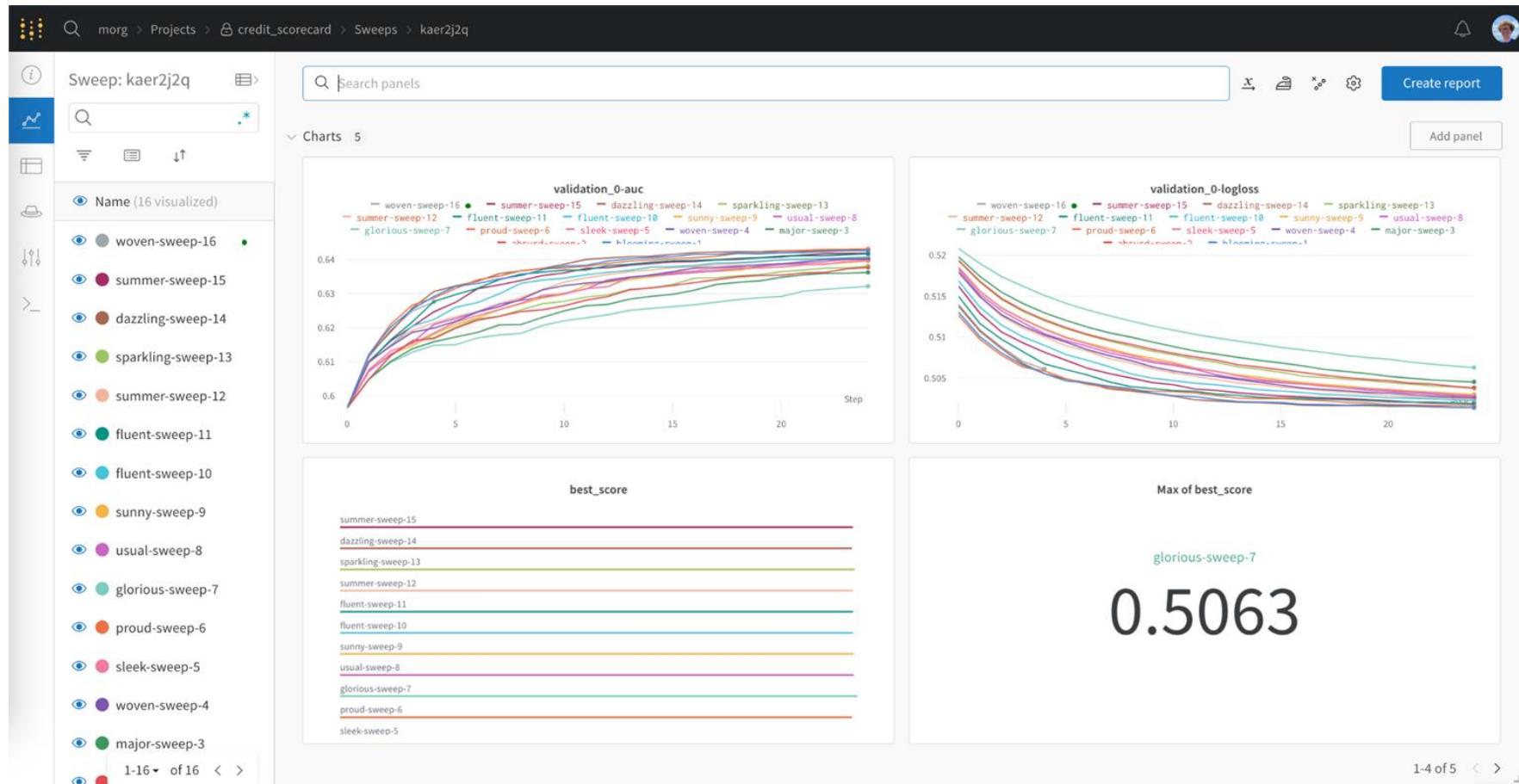
# Run cross validation and log metadata to the run in Neptune
xgb.cv(
    params=model_params,
    dtrain=dtrain,
    callbacks=[neptune_callback],
)

# Stop run
run.stop()
```



Weights & Biases

□ Documentations





Weights & Biases: Ví dụ

- [Colab example](#)
- [live dashboard example](#)

```
from wandb.xgboost import WandbCallback
import xgboost as XGBClassifier

...
# Start a wandb run
run = wandb.init()

# Pass WandbCallback to the model
bst = XGBClassifier()
bst.fit(X_train, y_train,
       callbacks=[WandbCallback(log_model=True)])

# Close your wandb run
run.finish()
```

```
from wandb.lightgbm import wandb_callback, log_summary
import lightgbm as lgb

# Log metrics to W&B
gbm = lgb.train(..., callbacks=[wandb_callback()])

# Log feature importance plot and upload model checkpoint to W&B
log_summary(gbm, save_model_checkpoint=True)
```



Confusion Matrix

- Một bảng cho phép **trực quan hóa** hiệu suất của thuật toán
- Cung cấp **thông tin chi tiết** về mô hình phân loại đang hoạt động tốt như thế nào và chính xác nơi đang gây ra lỗi
 - Overall Accuracy, Misclassification Rate ($FP+FN$), Precision, Recall,..
 - Ví dụ: high specificity nhưng a low sensitivity (overly conservative) → điều chỉnh **threshold**

	Predicted Negative	Predicted Positive
Actual Negative	True Negative (TN)	False Positive (FP)
Actual Positive	False Negative (FN)	True Positive (TP)

Example confusion matrix for binary classification



Case study: Phát hiện lạm dụng quảng cáo

Đối với người lạm dụng(label 2):

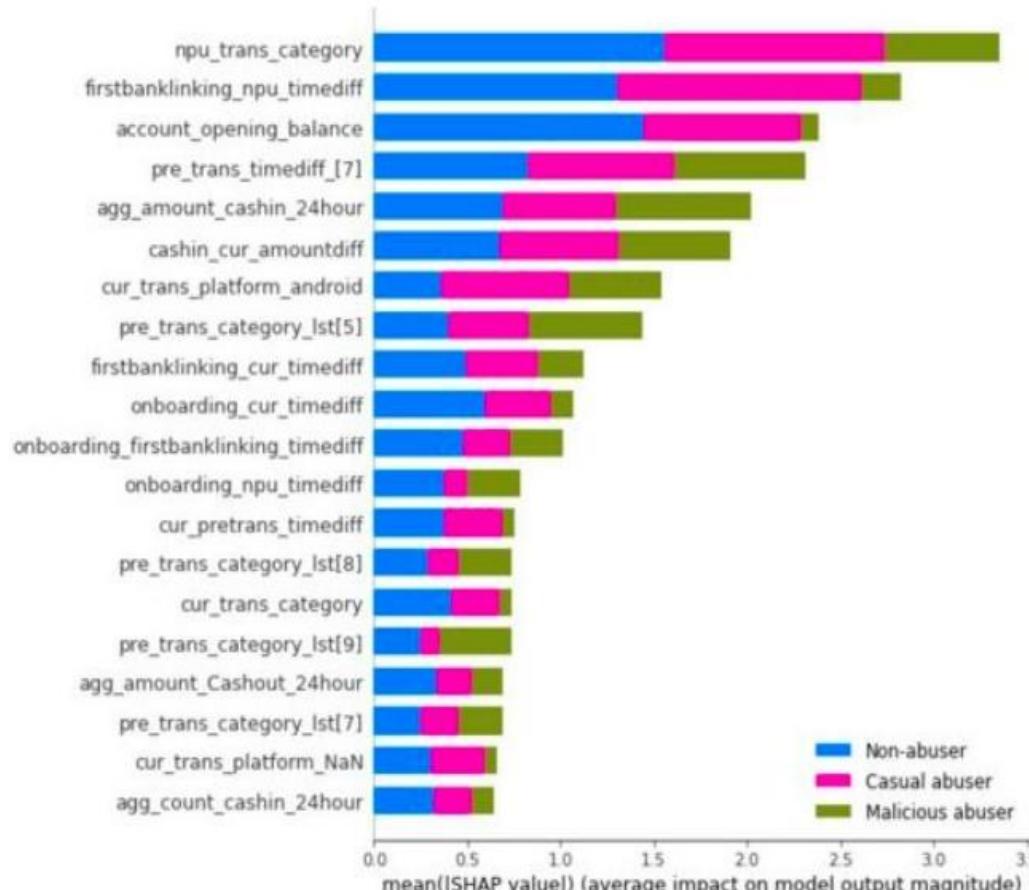
- Very low false positive (high precision)
- High false negative (low recall)

=> Cần đi sâu vào false
negative sample để cải thiện
recall

label	cb_rmshifted	cb_rmshifted_onehot	count	Error type	percent
0	0	0	152,979	TRUE	91.82%
0	0	1	373		0.22%
0	0	2	6	False positive	0.00%
0	1	0	432		0.26%
0	1	1	2,848		1.71%
0	1	2	1	False positive	0.00%
0	2	0	2	False positive	0.00%
0	2	1	1	False positive	0.00%
0	2	2	8	False positive	0.00%
1	0	0	3,383		2.03%
1	0	1	197		0.12%
1	1	0	199		0.12%
1	1	1	2,624	TRUE	1.57%
1	2	2	2		0.00%
2	0	0	2,933	False negative	1.76%
2	0	1	64		0.04%
2	0	2	6	False negative	0.00%
2	1	0	88		0.05%
2	1	1	317		0.19%
2	1	2	15		0.01%
2	2	0	22	False negative	0.01%
2	2	1	3		0.00%
2	2	2	108	TRUE	0.06%
			166,611		



Case study: Phát hiện lạm dụng quảng cáo



Feature importance over FN samples

Adversarial test to know which features can distinguish between True-positive (TP) and FN
roc_auc_score: 0.97

Feature Id	Importances
0	npu_trans_category 20.758266
1	cur_trans_category 18.601582
2	onboarding_npu_timediff 10.036132
3	pre_trans_category_lst[1] 9.929088
4	pre_trans_category_lst[2] 7.524854
5	pre_trans_category_lst[0] 7.164027
6	onboarding_cur_timediff 5.214986
7	pre_trans_category_lst[4] 4.349288
8	pre_trans_category_lst[3] 2.958303
9	account_opening_balance 2.786141

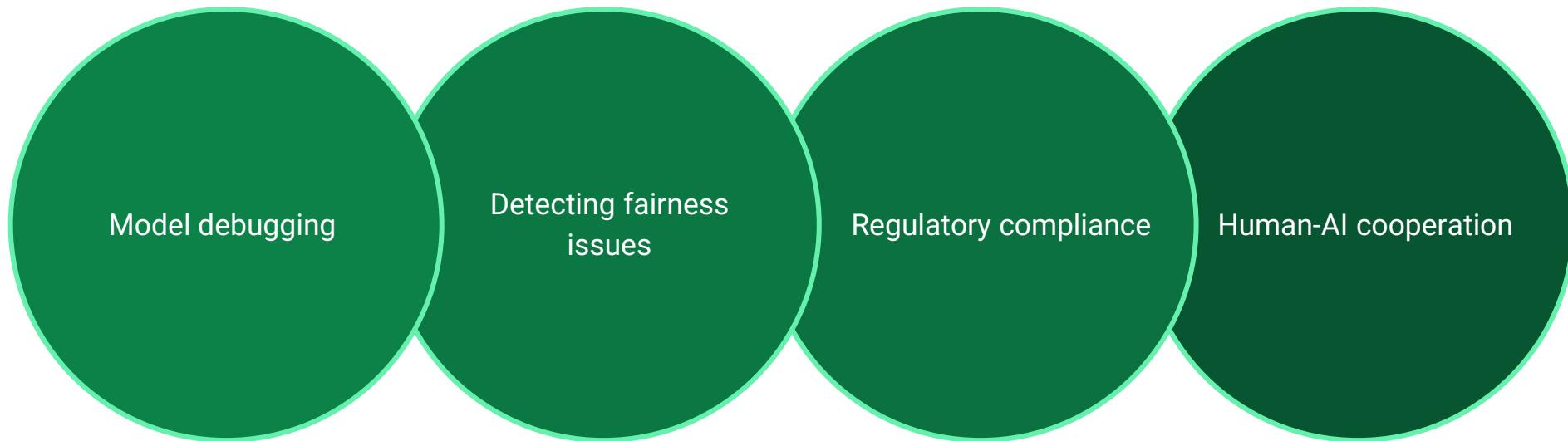
Feature importance on adversarial model

=> Cần thêm **abusers data** từ chiến dịch khác!



Công cụ: InterpretML

- ❑ InterpretML giúp bạn hiểu hành vi toàn cầu của mô hình hoặc hiểu lý do đằng sau các dự đoán riêng lẻ
- ❑ [Github](#)
- ❑ [Documentations](#)





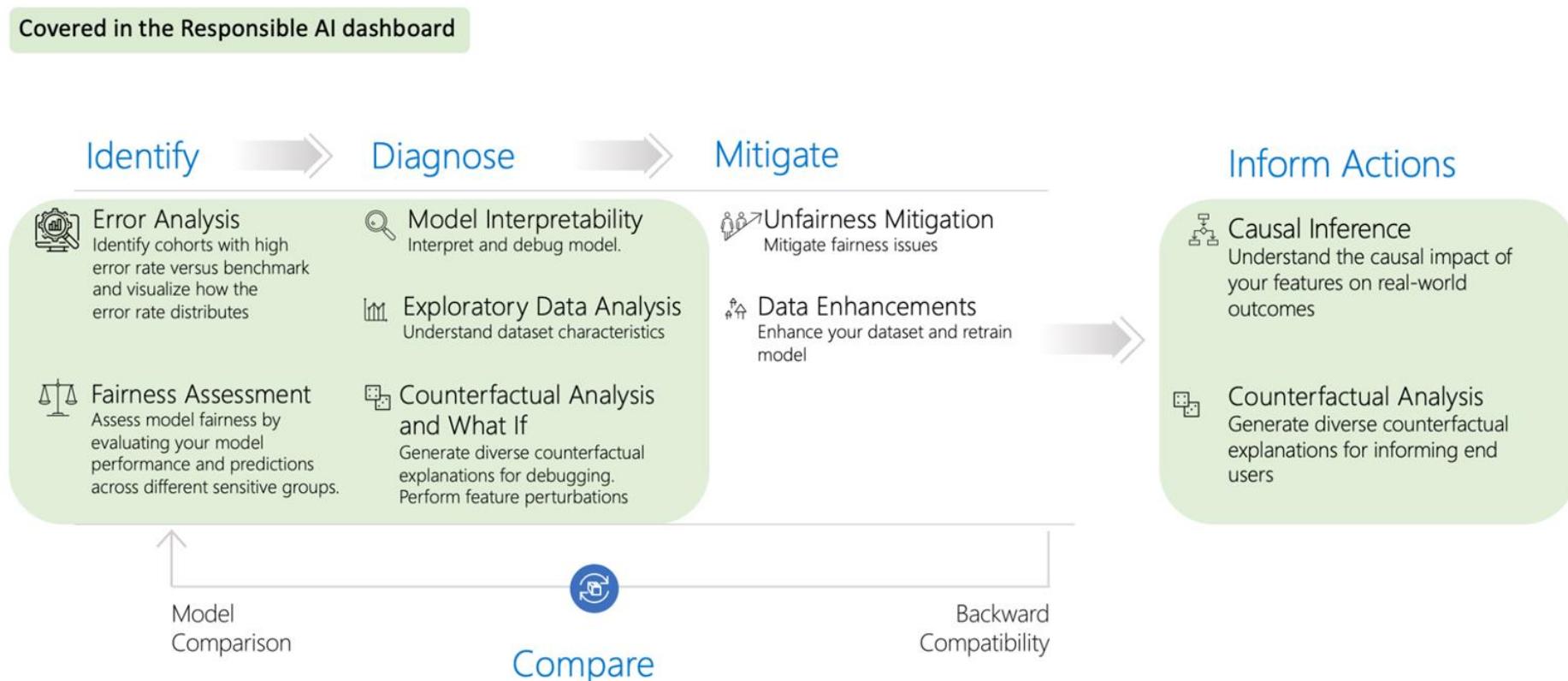
InterpretML: Các kỹ thuật hỗ trợ

Interpretability Technique	Type
Explainable Boosting	glassbox model
Decision Tree	glassbox model
Decision Rule List	glassbox model
Linear/Logistic Regression	glassbox model
SHAP Kernel Explainer	blackbox explainer
LIME	blackbox explainer
Morris Sensitivity Analysis	blackbox explainer
Partial Dependence	blackbox explainer



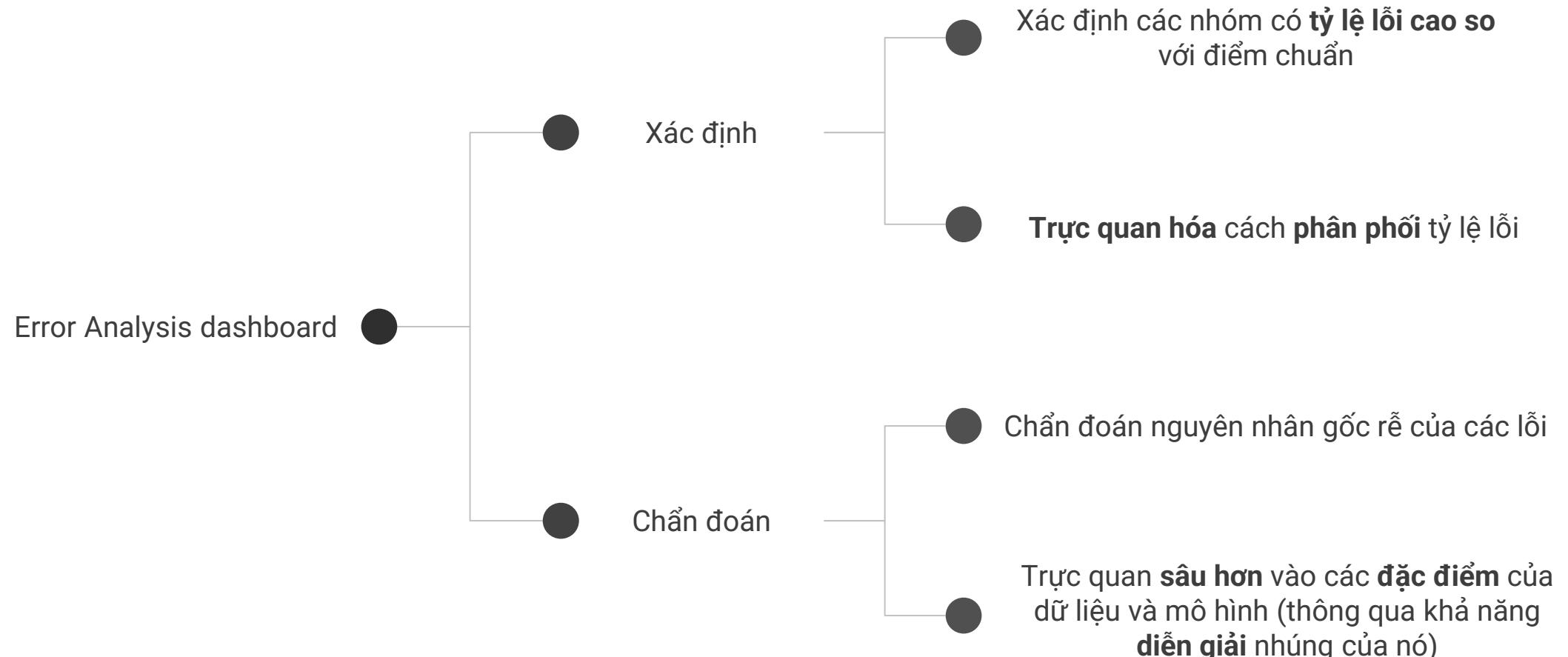
Tools: Responsible AI Toolbox

□ [github](#)





Error Analysis





Error Analysis Dashboard

Đánh giá các nhóm có đặc điểm chung

Tìm hiểu cách lỗi **phân bố** giữa các nhóm khác nhau ở **các mức độ chi tiết** khác nhau

Khám phá các dự đoán

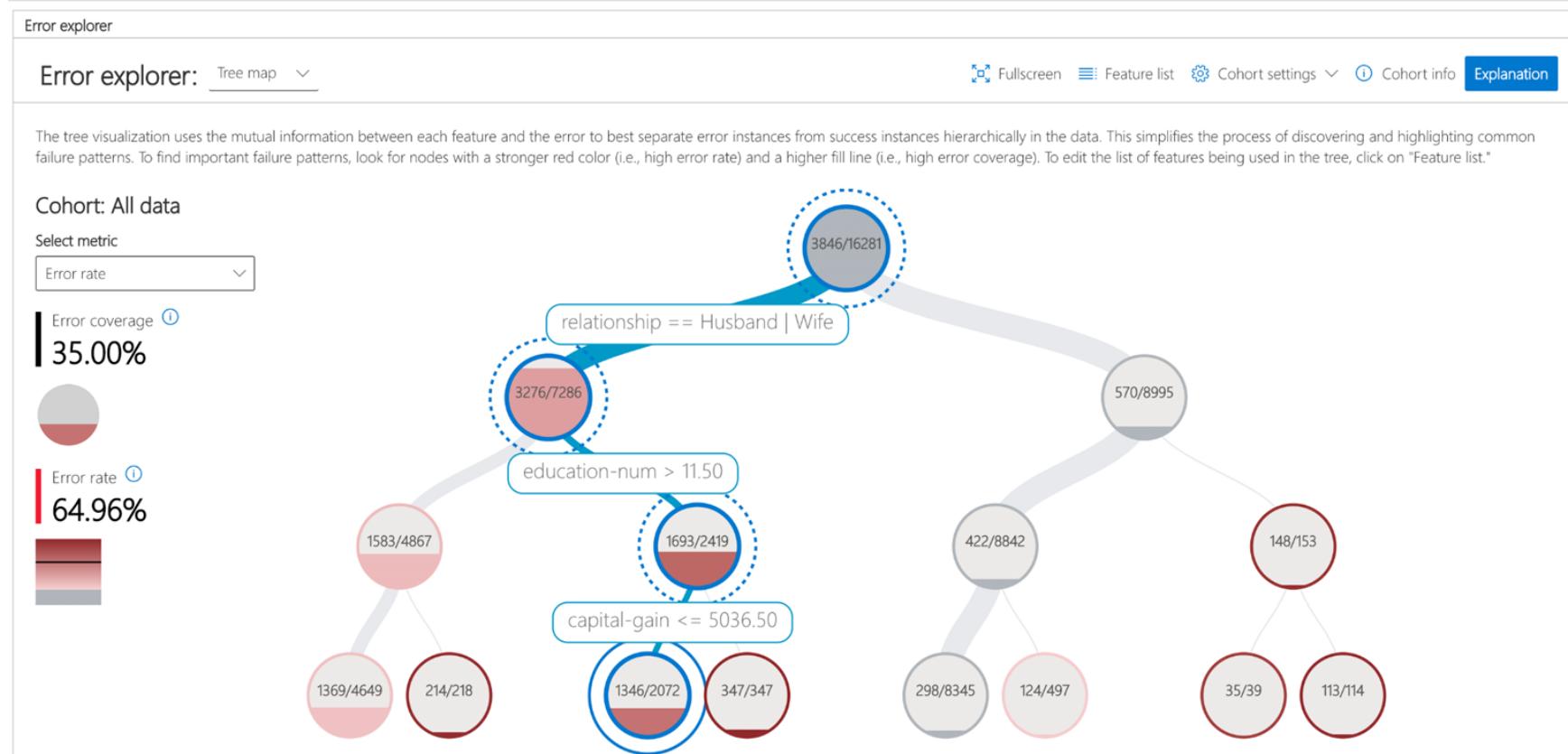
Sử dụng các **tính năng diễn giải** tích hợp sẵn hoặc kết hợp với **InterpretML** để tăng khả năng debugging

Chế độ xem bảng điều khiển tương tác

Hình ảnh được xây dựng sẵn có thể tùy chỉnh để nhanh chóng xác định lỗi và chẩn đoán nguyên nhân gốc rễ

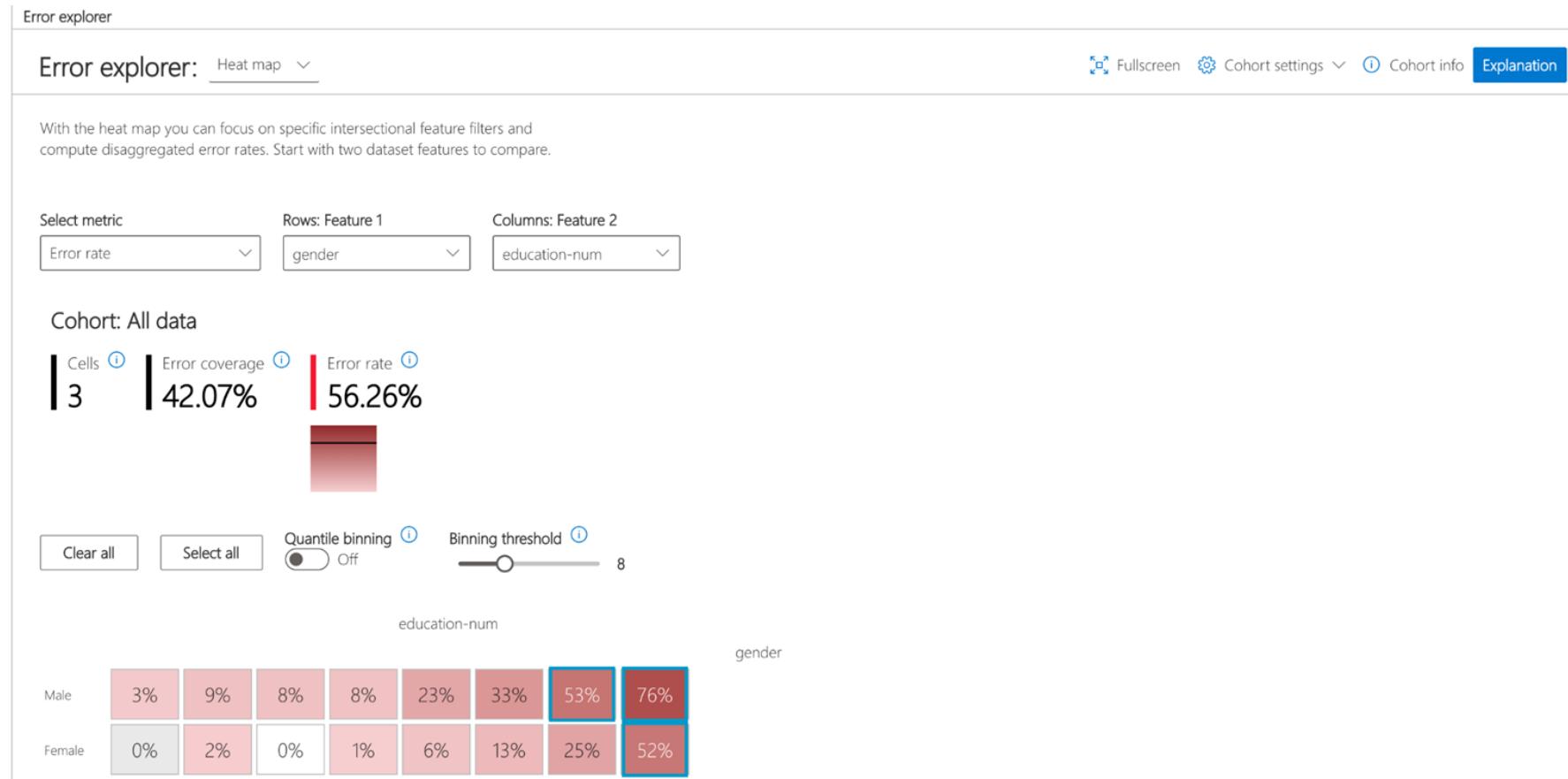


Identification of Errors: Decision Tree





Identification of Errors: Heat Map





Diagnosis of Errors

Khám phá dữ liệu

**Khám phá số liệu
thống kê tập dữ liệu
và phân phối đặc
trưng**

Giải thích toàn cục

Khám phá K tính năng
quan trọng hàng đầu
ảnh hưởng đến mô hình
tổng thể. Giải thích
toàn cục cho một nhóm
dữ liệu đã chọn

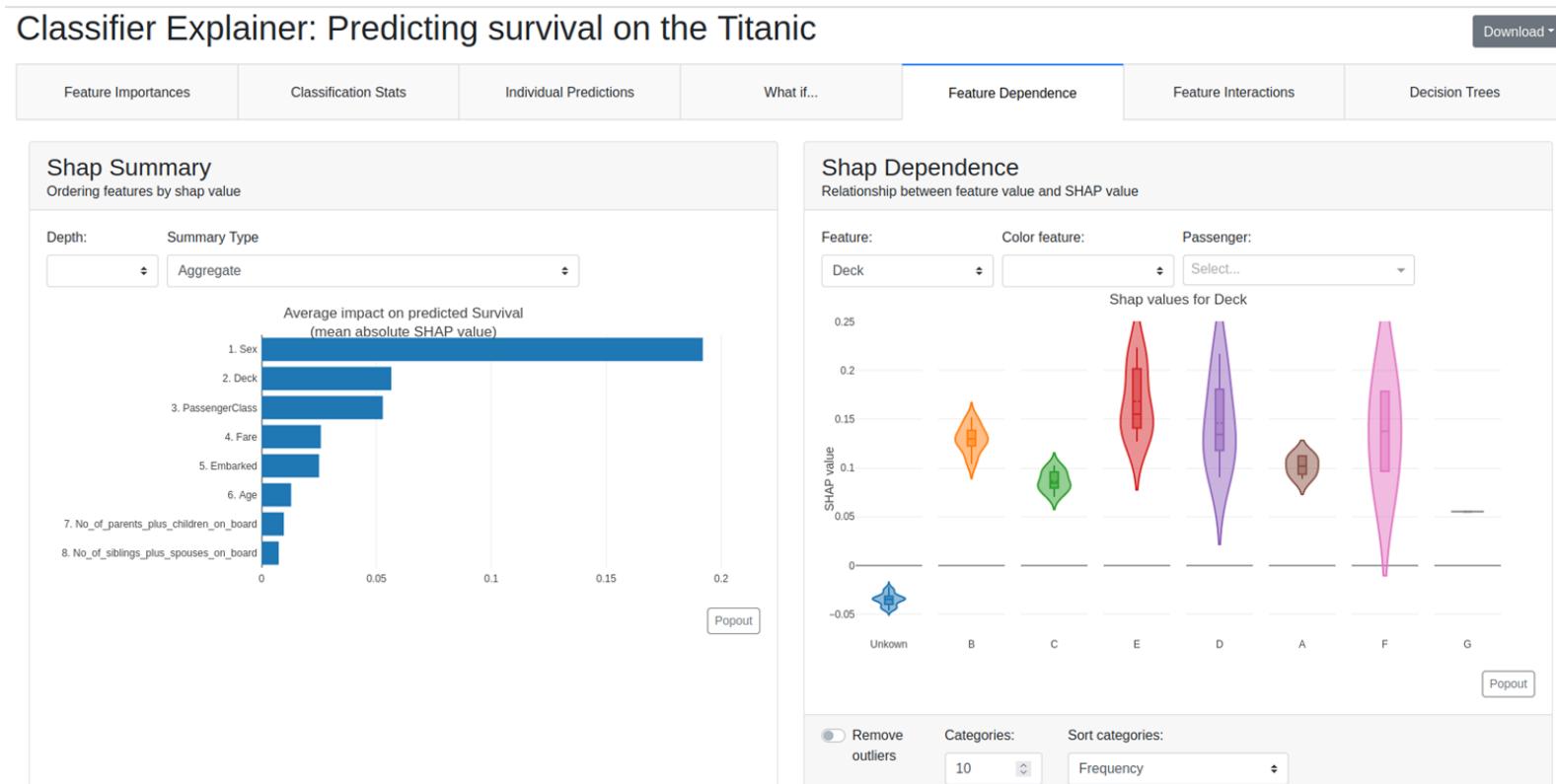
Giải thích cục bộ

Cho phép quan sát dữ
liệu thô trong Chế độ
xem từng đối tượng. Hiểu
cách mỗi điểm dữ liệu có
dự đoán chính xác hoặc
không chính xác.



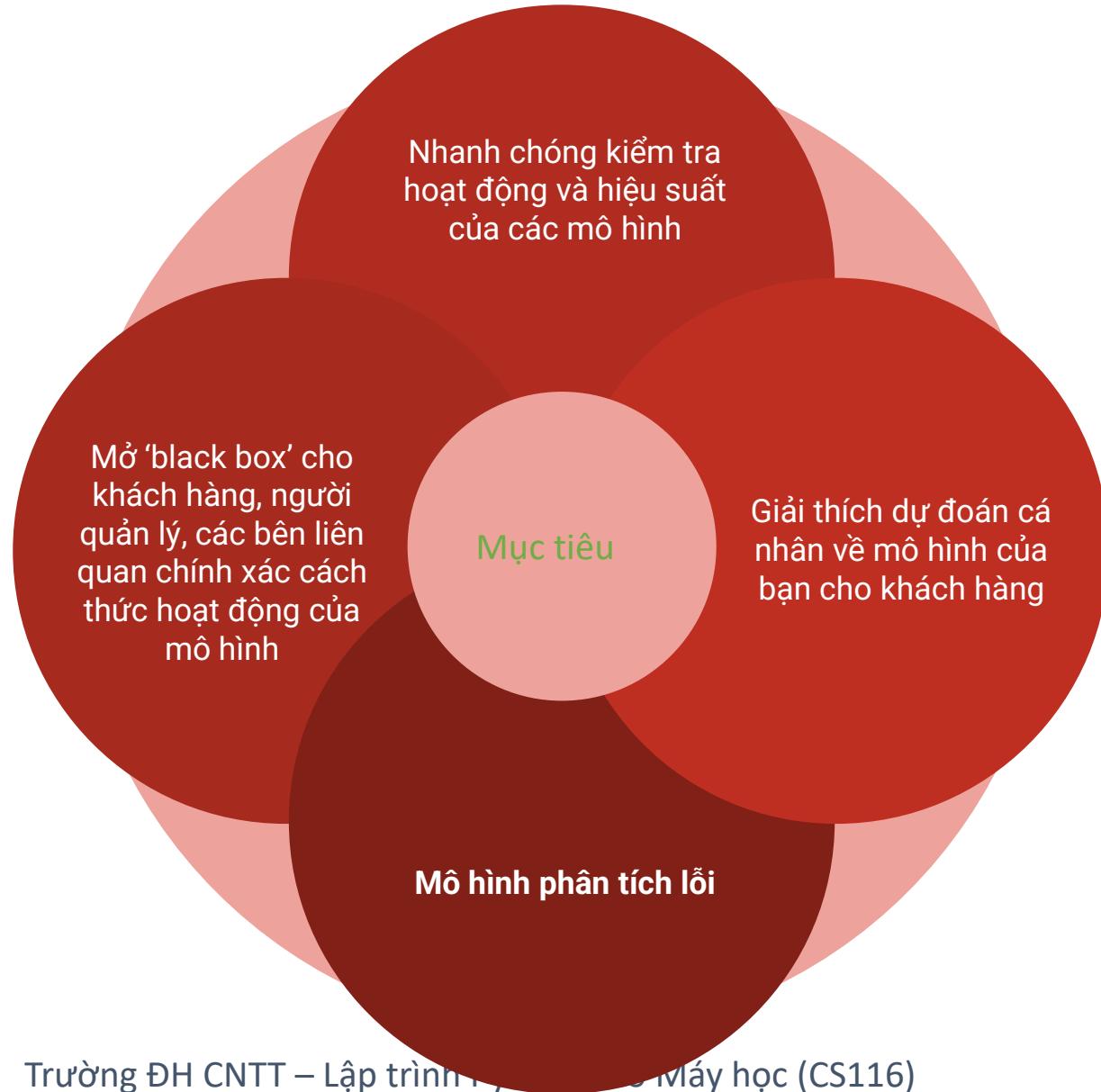
Tools: explainerdashboard

- Thư viện python giúp dễ dàng nhanh chóng xây dựng một bảng điều khiển tương tác giải thích hoạt động bên trong của một mô hình ML được trang bị
- github: [link](#)
- example: [link](#)



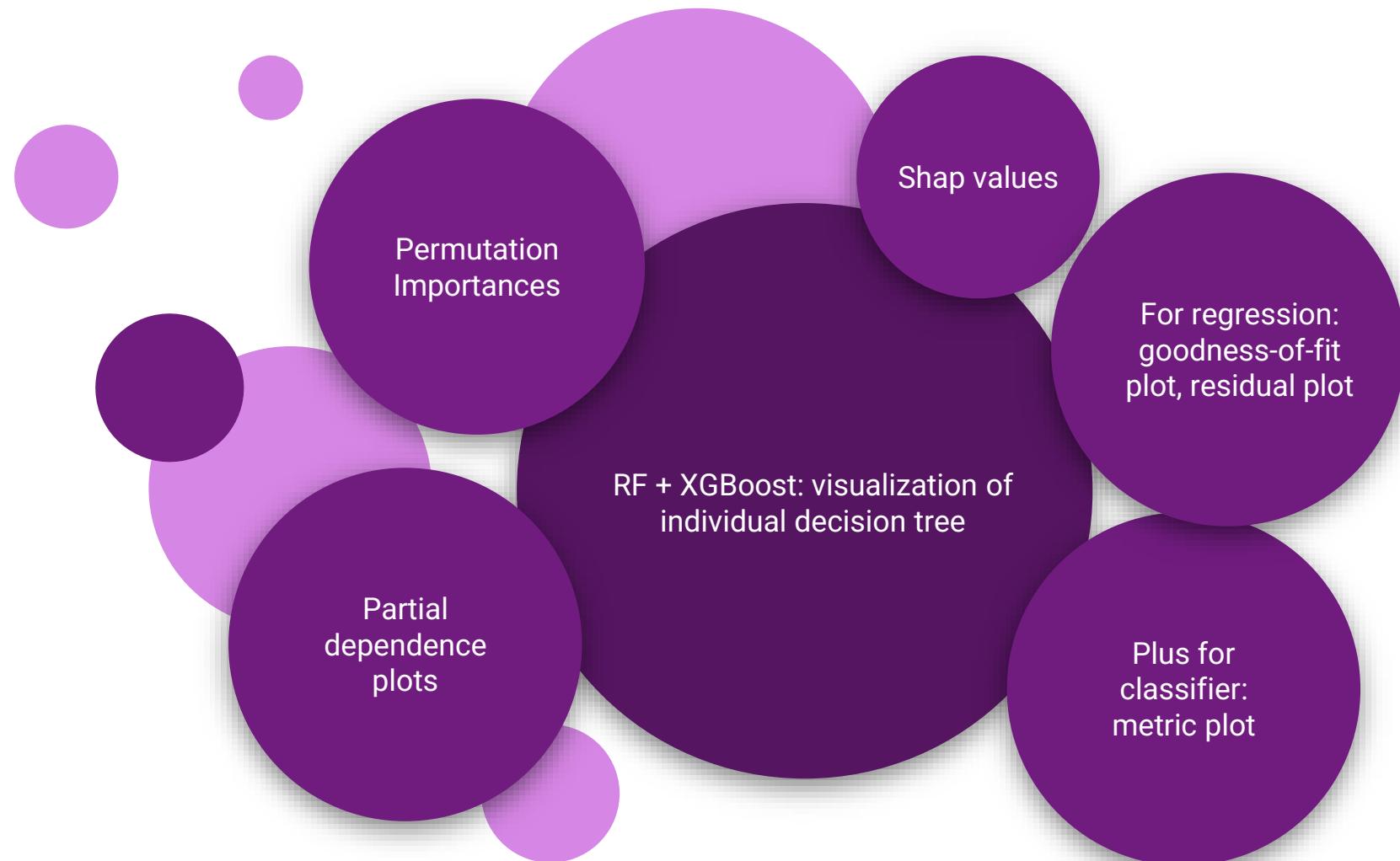


Background: explainerdashboard





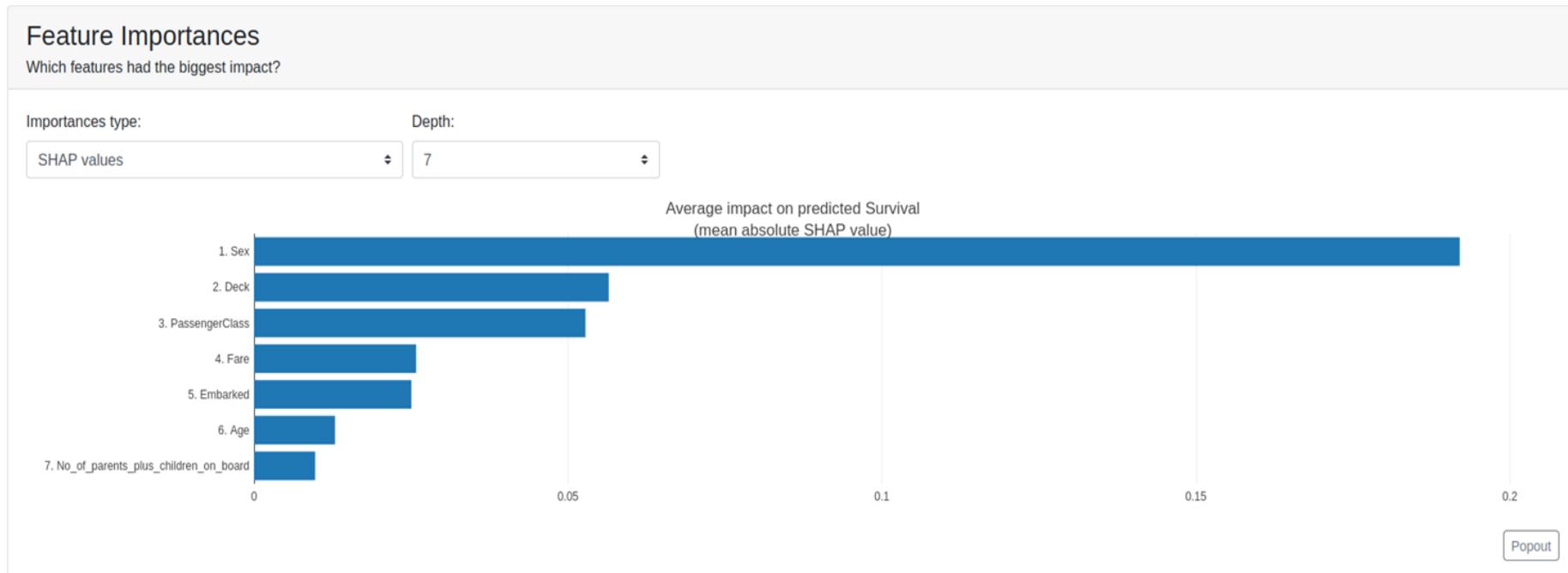
Features: explainerdashboard





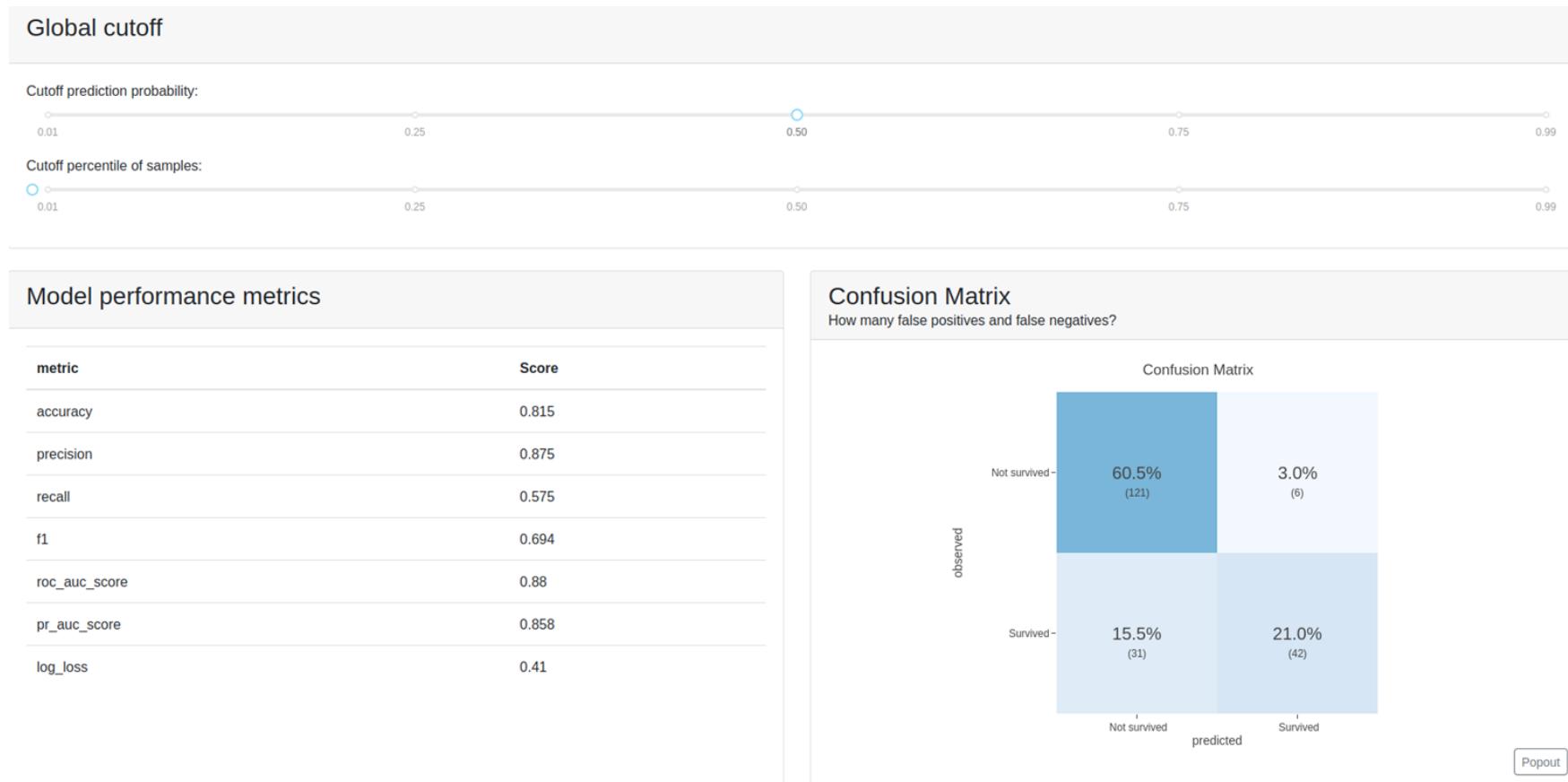
Feature Importances

Feature Importances



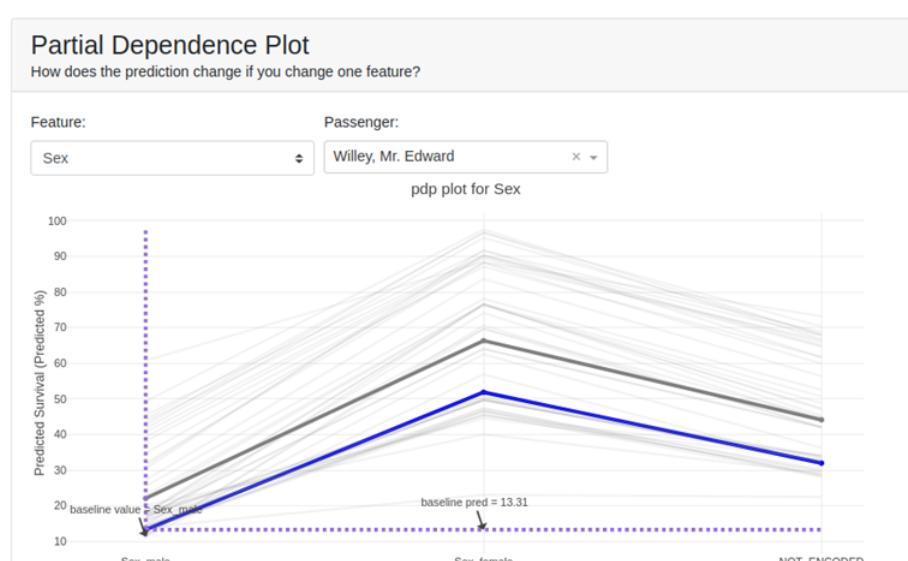
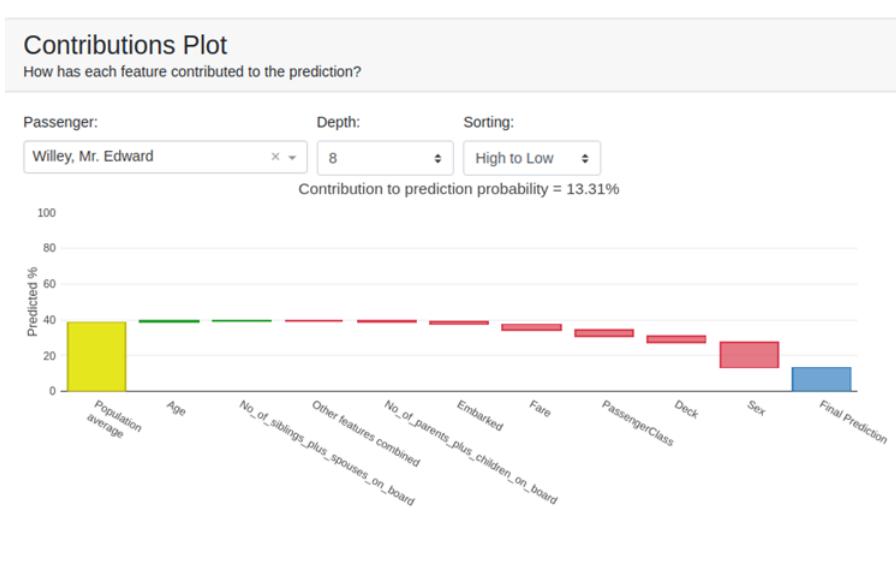
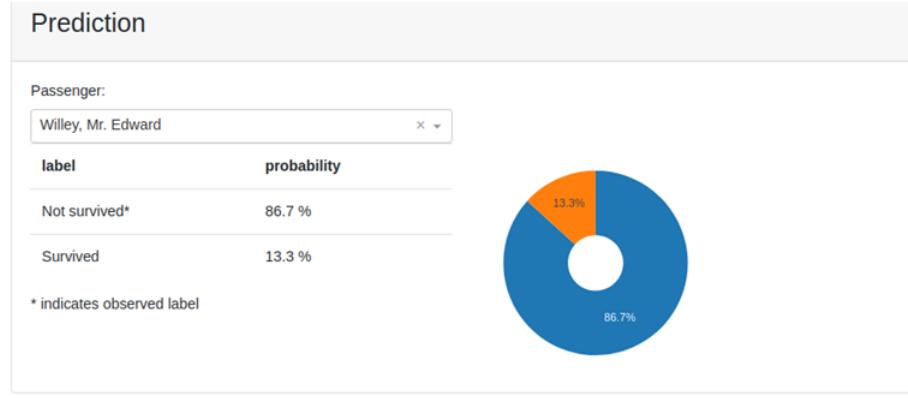
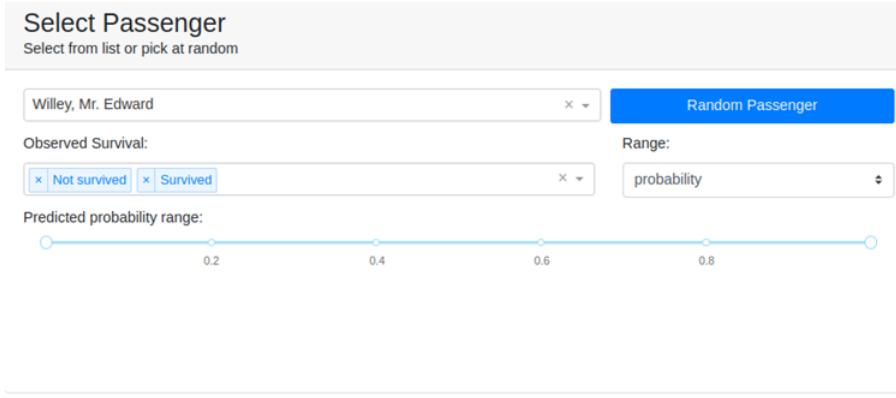


Thống kê cho bài toán phân loại



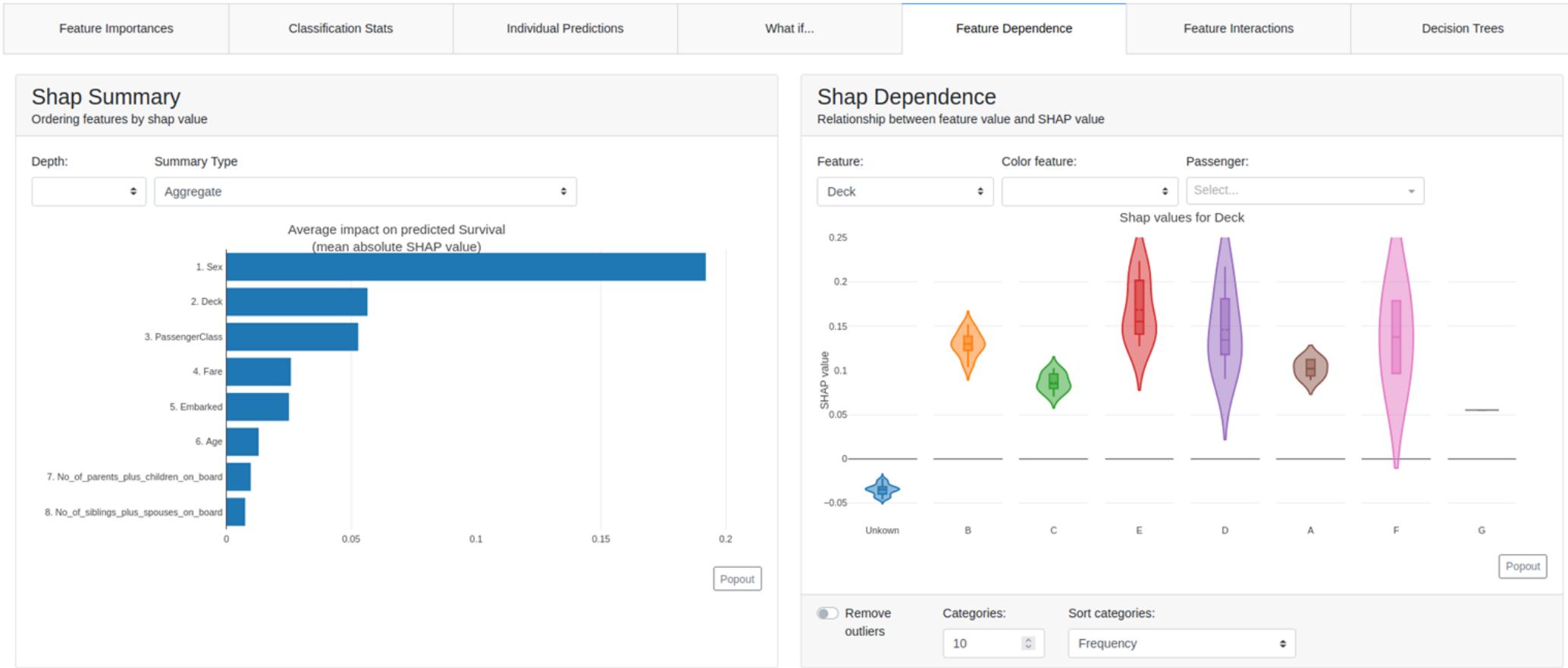


Individual Predictions





Feature Dependence





Decision Tree

Feature	Condition	Adjustment	New Prediction
		Starting average	39.07%
Sex_male	0.0 < 0.5	+33.37%	72.44%
No_of_siblings_plus_spouses_on_board	1.0 < 2.5	+4.48%	76.92%
Fare	16.1 >= 10.879199981689453	+8.13%	85.05%
Age	26.0 < 51.0	-1.53%	83.52%
Fare	16.1 < 40.6335037231445	-8.29%	75.23%
		Final Prediction	75.23%

Decision path graph
Visualizing entire decision tree

Passenger: Lobb, Mrs. William Arthur (Cordelia K Stanlick)

Show tree: 3

Generate Tree Graph

The decision path graph visualizes the flow of data through a decision tree. It starts with a passenger record: Lobb, Mrs. William Arthur (Cordelia K Stanlick). The tree has three levels of splits. The first split is based on Sex_male (0.50). The second level includes Age (18.00), No_of_siblings_plus_spouses_on_board (2.50), and Fare (10.88). The third level includes Fare (14.50), Embarked_Cherbourg (0.50), and Age (51.00). The final prediction is Survived, indicated by a green circle.



Tài liệu tham khảo

- [Pattern Recognition and Machine Learning](#)
- <https://www.cs.cornell.edu/courses/cs4780/2018fa/lectures/>



HỎI ĐÁP

