

NHÓM 8

PHÂN TÍCH VÀ DỰ BÁO DOANH THU

NGÀNH



GVHD: TS. Đỗ Thanh Thái

SVTH1: Nguyễn Quốc Thịnh - 2213296

SVTH2: Trần Đình Tưởng - 2213892

SVTH3: Trần Quốc Khánh - 2311538

1. GIỚI THIỆU DỮ LIỆU VÀ QUY TRÌNH

1.1. Giới thiệu bộ dữ liệu

Bộ dữ liệu được sử dụng là **TMDB 5000 Movies Dataset**, bao gồm hơn 5000 bộ phim với nhiều thuộc tính quan trọng như:

- Kinh phí sản xuất (budget)
- Độ phổ biến (popularity)
- Điểm đánh giá, số lượng đánh giá
- Thể loại (genres)
- Ngôn ngữ gốc (original_language)
- Doanh thu (revenue)

Dự đoán doanh thu phim dựa trên các features còn lại



	id	genre_ids	popularity	vote_average	vote_count	budget	\
0	19995	[28, 12, 14, 878]	150.437577	7.2	11800	237000000	
1	285	[12, 14, 28]	139.082615	6.9	4500	300000000	
2	206647	[28, 12, 80]	107.376788	6.3	4466	245000000	
3	49026	[28, 80, 18, 53]	112.312950	7.6	9106	250000000	
4	49529	[28, 12, 878]	43.926995	6.1	2124	260000000	
5	559	[14, 28, 12]	115.699814	5.9	3576	258000000	

	original_language	revenue
0	en	2.787965e+09
1	en	9.610000e+08
2	en	8.806746e+08
3	en	1.084939e+09
4	en	2.841391e+08
5	en	8.908716e+08

Hình 1: Bộ dữ liệu sử dụng

1. GIỚI THIỆU DỮ LIỆU VÀ QUY TRÌNH

1.2. Quy trình tổng quan

1. Tiền xử lý dữ liệu
2. Biến đổi và mã hóa đặc trưng
3. Chia train/ test
4. Huấn luyện mô hình Random Forest
5. Đánh giá bằng MSE, RMSE, R²
6. Trực quan hóa và phân tích lỗi



2. TIỀN XỬ LÝ DỮ LIỆU

```
def load_tmdb_movies_df(input_csv_path: str) -> pd.DataFrame:  
    """  
    Đọc file TMDB, chuẩn hóa các cột số, giữ nguyên cột 'genres' và 'original_language'  
    để xử lý sau bằng MultilabelBinarizer và OneHotEncoder.  
    """  
    df = pd.read_csv(input_csv_path, encoding="utf-8")  
    df=encode_genres(df)  
    df[\"original_language\"] = df[\"original_language\"].apply(  
        lambda x:x if x == "en" else "other"  
    )  
    #print(df[\"original_language\"].value_counts())  
  
    # Chuẩn hóa kiểu dữ liệu số  
    numeric_cols = ['popularity', 'vote_average', 'vote_count', 'budget', 'revenue']  
    for c in numeric_cols:  
        if c in df.columns:  
            df[c] = pd.to_numeric(df[c], errors='coerce')  
  
    # Loại bỏ revenue bị 0 hoặc Null  
    if 'revenue' in df.columns:  
        df['revenue'] = df['revenue'].replace(0, np.nan)  
        df = df.dropna(subset=[ 'revenue' ])  
  
    # Giữ lại các cột cần thiết (không dùng tới genres và original_language)  
    cols = ['id', 'genre_ids', 'popularity', 'vote_average', 'vote_count',  
           'budget', 'original_language', 'revenue']  
    df = df[[c for c in cols if c in df.columns]].copy()  
  
    return df
```

```
def prepare_dataset_for_sklearn(df: pd.DataFrame, target: str = "revenue"):  
    data = df.copy()  
    #data=data.explode("genre_ids")  
    # chỉ impute 4 cột này (0/NaN -> mean)  
    cols_impute = ["popularity", "vote_average", "vote_count", "budget"]  
    impute_zero_nan_with_mean(data, cols_impute)  
  
    numeric_features = ["popularity", "vote_average", "vote_count", "budget"]  
    lang_feature = ["original_language"]  
    gen_feature=[\"genre_ids\"]  
    mlb=MultilabelBinarizer()  
    genre_ohe=mlb.fit_transform(data[\"genre_ids\"])  
    genre_cols = [f\"genre_{g}\" for g in mlb.classes_]  
    preprocessor=ColumnTransformer(  
        [  
            ("num", "passthrough", numeric_features),  
            ("lang", OneHotEncoder(sparse_output=False), lang_feature),  
        ],  
        remainder="drop"  
    )  
  
    X_transform = preprocessor.fit_transform(data)  
    X=np.hstack([X_transform,genre_ohe])  
    num_cols = numeric_features  
    lang_cols = preprocessor.named_transformers_[\"lang\"].get_feature_names_out(lang_feature)  
    feature_cols = list(num_cols) + list(lang_cols) + genre_cols  
    y = data[target].astype(float).values  
    X=pd.DataFrame(X,columns=feature_cols)  
    y=pd.DataFrame(y,columns=[target])  
    return X, y, feature_cols
```

2. TIỀN XỬ LÝ DỮ LIỆU

2.1. Làm sạch dữ liệu

- Chuyển tất cả cột số về numeric.
- Giá trị revenue = 0 hoặc NaN được loại bỏ.
- Các cột số (budget, popularity, vote_count,...) được thay 0 bằng mean.

2.2. Xử lý thể loại phim

- Trường genres là chuỗi JSON.
- Parse thành danh sách ID thể loại.
- Mã hóa bằng MultiLabelBinarizer (tạo nhiều cột genre_x).

2.3. Xử lý ngôn ngữ phim

- Vì đa số là ngôn ngữ en, nên ở cột original language, các hàng có giá trị là en sẽ giữ nguyên, không sẽ là "other"
- Mã hóa bằng OneHotEncoder.

2.4. Tạo ma trận đặc trưng

- Tổng số đặc trưng sau xử lý:
- 4 đặc trưng số
 - 2 đặc trưng ngôn ngữ
 - >20 đặc trưng thể loại

```
print(x.head(5))
```

...	popularity	vote_average	vote_count	budget	original_language_en																				
0	150.437577	7.2	11800.0	2370000000.0																				1.0	
1	139.082615	6.9	4500.0	3000000000.0																				1.0	
2	107.376788	6.3	4466.0	2450000000.0																				1.0	
3	112.312950	7.6	9106.0	2500000000.0																				1.0	
4	43.926995	6.1	2124.0	2600000000.0																				1.0	
	original_language_other	genre_12	genre_14	genre_16	genre_18	...	\																		
0	0.0	1.0	1.0	0.0	0.0	0.0	...																		
1	0.0	1.0	1.0	0.0	0.0	0.0	...																		
2	0.0	1.0	0.0	0.0	0.0	0.0	...																		
3	0.0	0.0	0.0	0.0	0.0	1.0	...																		
4	0.0	1.0	0.0	0.0	0.0	0.0	...																		
	genre_53	genre_80	genre_99	genre_878	genre_9648	genre_10402	\																		
0	0.0	0.0	0.0	1.0	0.0	0.0	...																		
1	0.0	0.0	0.0	0.0	0.0	0.0	...																		
2	0.0	1.0	0.0	0.0	0.0	0.0	...																		
3	1.0	1.0	0.0	0.0	0.0	0.0	...																		
4	0.0	0.0	0.0	1.0	0.0	0.0	...																		
	genre_10749	genre_10751	genre_10752	genre_10769																					
0	0.0	0.0	0.0	0.0																					
1	0.0	0.0	0.0	0.0																					
2	0.0	0.0	0.0	0.0																					
3	0.0	0.0	0.0	0.0																					
4	0.0	0.0	0.0	0.0																					

[5 rows x 25 columns]

3. XÂY DỰNG MÔ HÌNH

3.1. Lý do chọn Random Forest

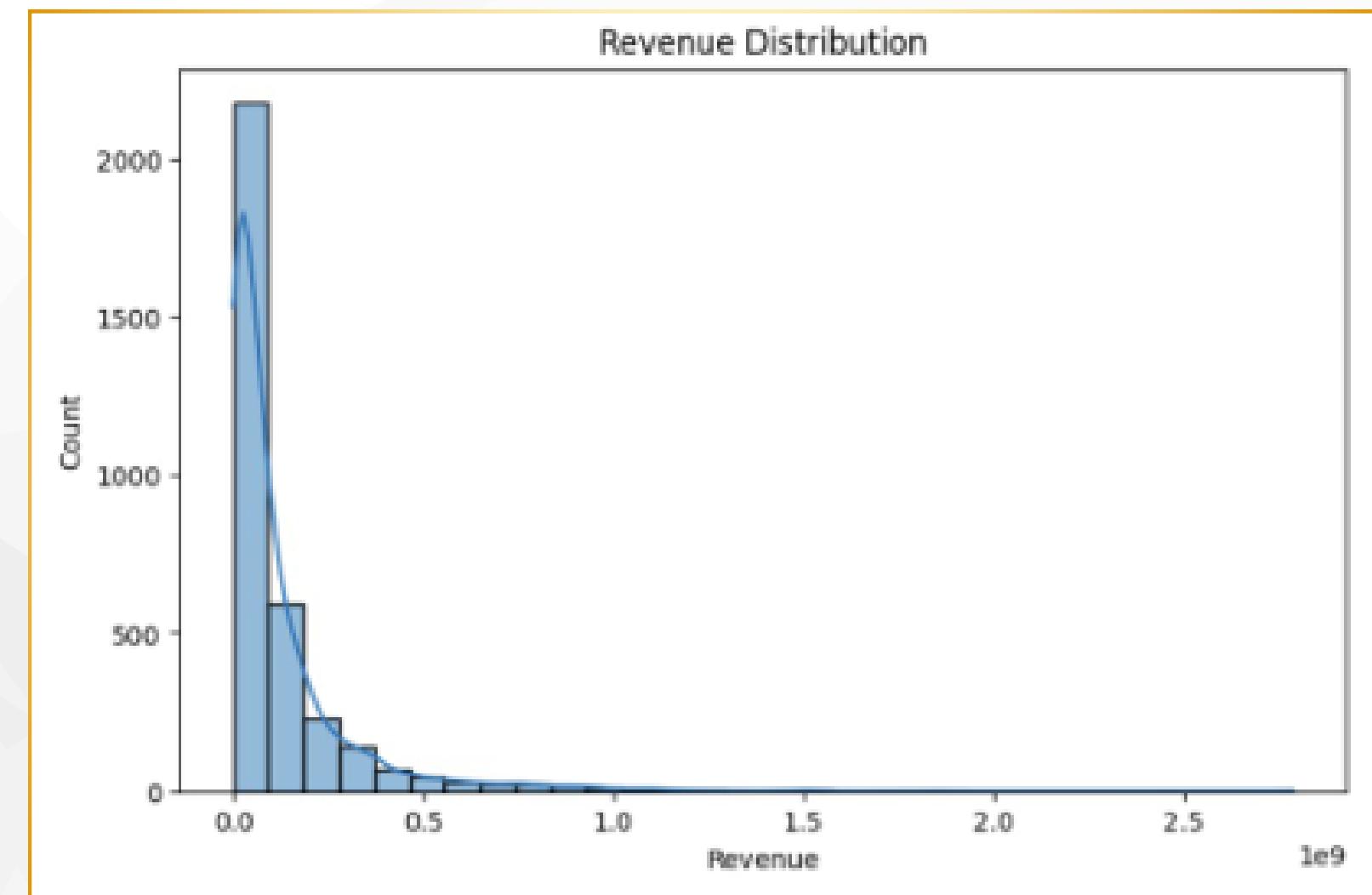
- Mạnh trên dữ liệu dạng tabular
- Hiệu quả với quan hệ phi tuyến
- Chống overfitting nhờ cơ chế bagging
- Không cần chuẩn hóa dữ liệu đầu vào

3.2. Log-transform biến mục tiêu

Do doanh thu phân phối lệch phải:

$$y' = \log(1 + y)$$

Mô hình sử dụng TransformedTargetRegressor sẽ train với $\log(1+revenue)$ nhưng sẽ tự động đảo ngược log khi dự đoán.



Hình 1: Phân phối doanh thu phim trong dataset

3. XÂY DỰNG MÔ HÌNH

3.3. Tuning tham số

Sử dụng RandomizedSearchCV với:

- n_estimators: 100, 300, 500, 800
- max_depth: None, 10, 20, 30, 40, 50
- min_samples_split: 2, 5, 10, 20
- min_samples_leaf: 1, 2, 5, 10
- max_features: sqrt, log2, 0.3, 0.5, None

3.4. Model cuối cùng

```
log_transformer = FunctionTransformer(np.log1p, inverse_func=np.expm1, validate=True)

model=TransformedTargetRegressor(
    regressor=RandomForestRegressor(random_state=42),
    #random_state= seed ngẫu nhiên,seed=42=> khi mà ông gọi random=>7,10,30,1,4
    transformer=log_transformer
)
from sklearn.model_selection import RandomizedSearchCV
#gridsearch
param_distrib={
    "regressor__n_estimators": [100,300,500,800],
    "regressor__max_depth": [None,10,20,30,40,50],
    "regressor__min_samples_split": [2,5,10,20],
    "regressor__min_samples_leaf": [1,2,5,10],
    "regressor__max_features": ["sqrt","log2",0.3,0.5,None],
    "regressor__bootstrap": [True,False]
}
# 1 mô hình machine learning, sẽ ko biết chọn các tham số trên như nào cho hợp lý
#dùng randomized_search
rnd=RandomizedSearchCV(
    model,# cái mô hình
    param_distributions=param_distrib,# 
    n_iter=30,# chạy 30 lần, chọn mô hình tốt nhất trong 30 lần
    cv=5,
    scoring="neg_root_mean_squared_error",
    #sqrt((y_train_du doan-y_train_that)^2)/ (samples)
    n_jobs=-1,#tận dụng gpu chạy song song
    random_state=42
)
```

Random Forest đã tuning

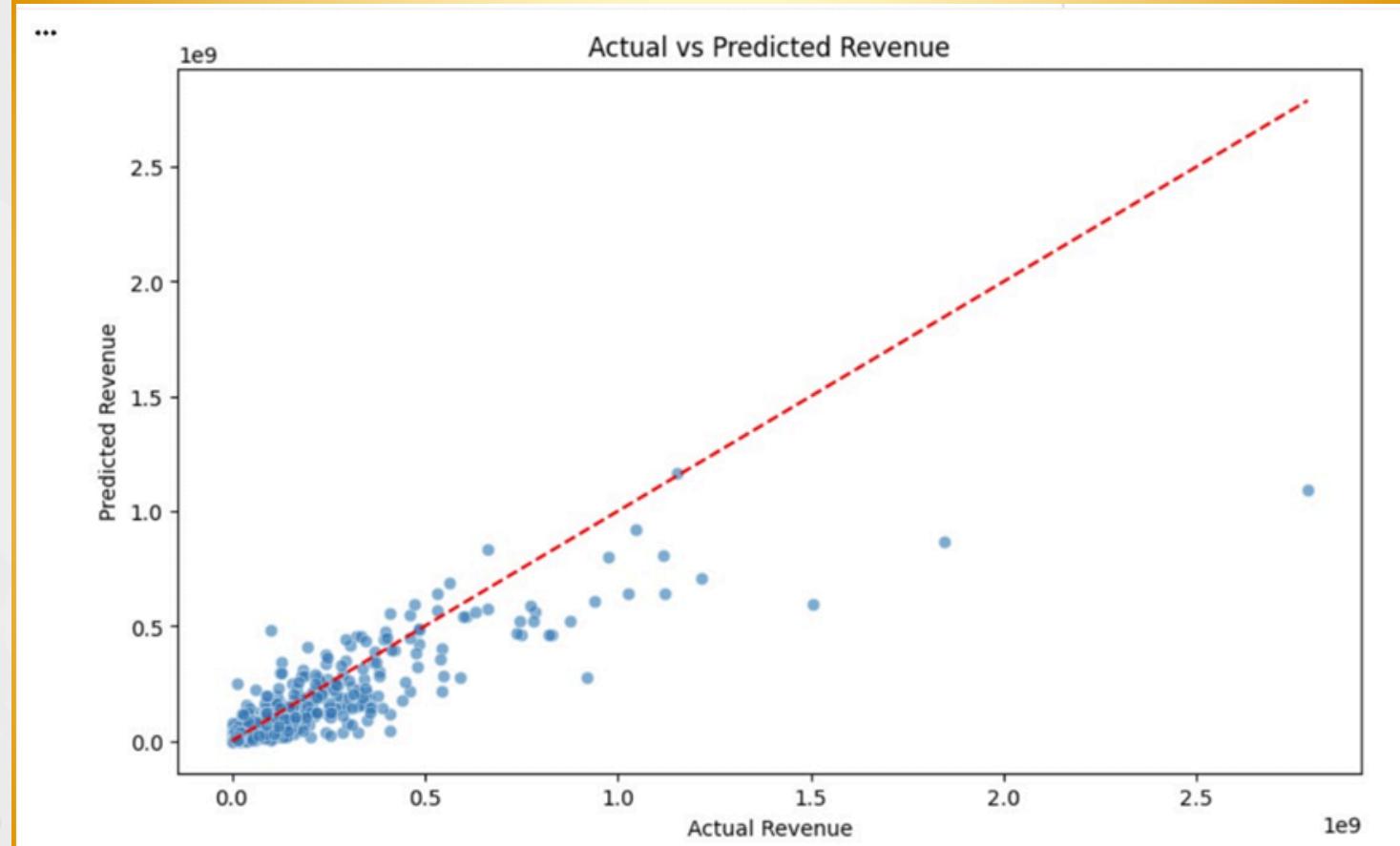
4. PHÂN TÍCH KẾT QUẢ

4.1. Đánh giá tổng quan

*** MSE train: 1103265448334052.50, RMSE train: 33,215,439.91, R2 train: 0.962
MSE test : 15100617485585192.00, RMSE test : 122,884,569.76, R2 test : 0.707

- RMSE train khoảng 33 triệu
- RMSE test cao hơn nhiều khoảng 122 triệu → có hiện tượng overfitting
- R^2 test khoảng là 0.707

4.2. Actual vs Predicted

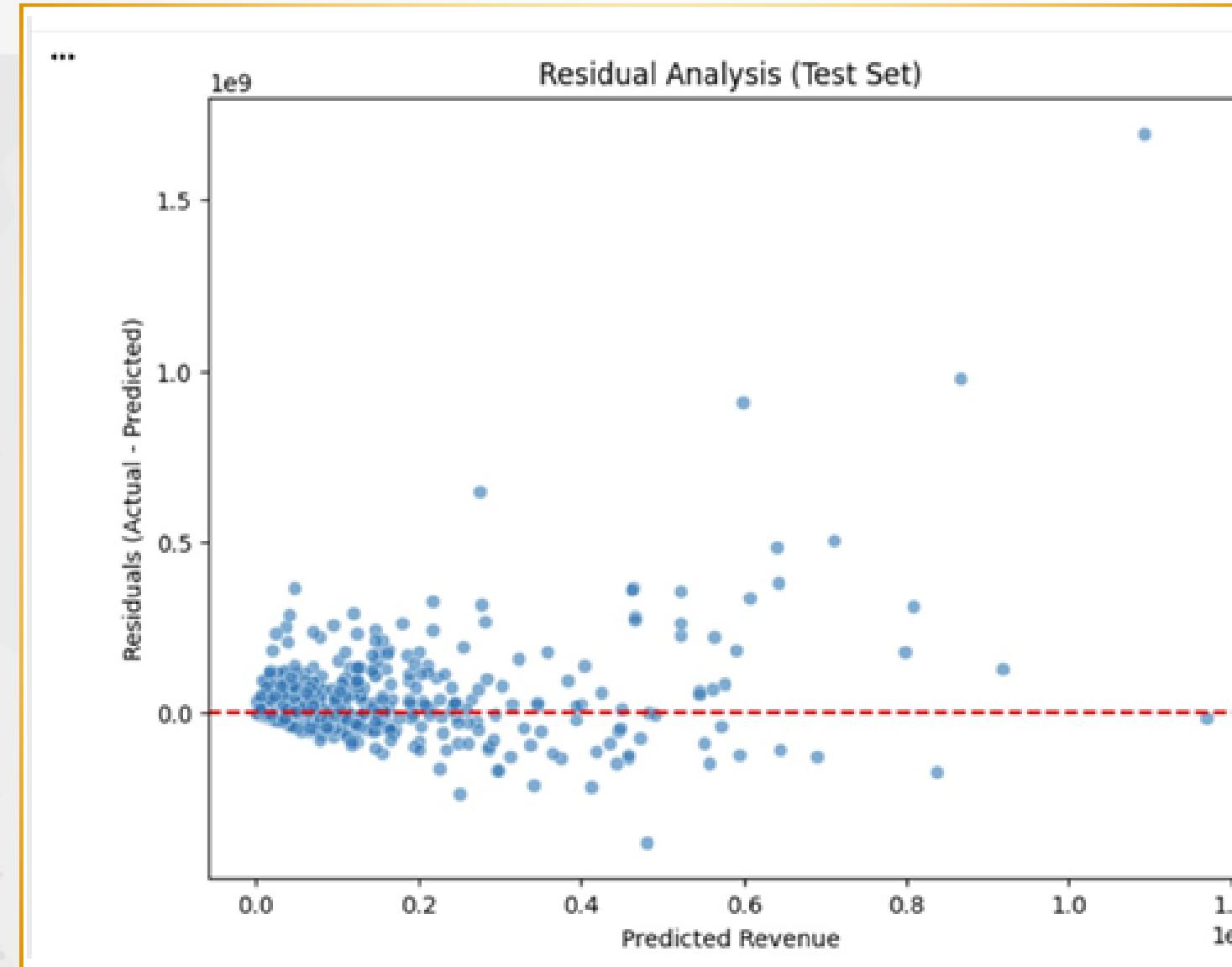


Mô hình dự đoán tốt các phim doanh thu thấp - trung bình nhưng kém chính xác ở phim doanh thu rất cao (outliers).

Hình 2: Biểu đồ so sánh giữa doanh thu dự đoán và doanh thu thật dựa trên test set

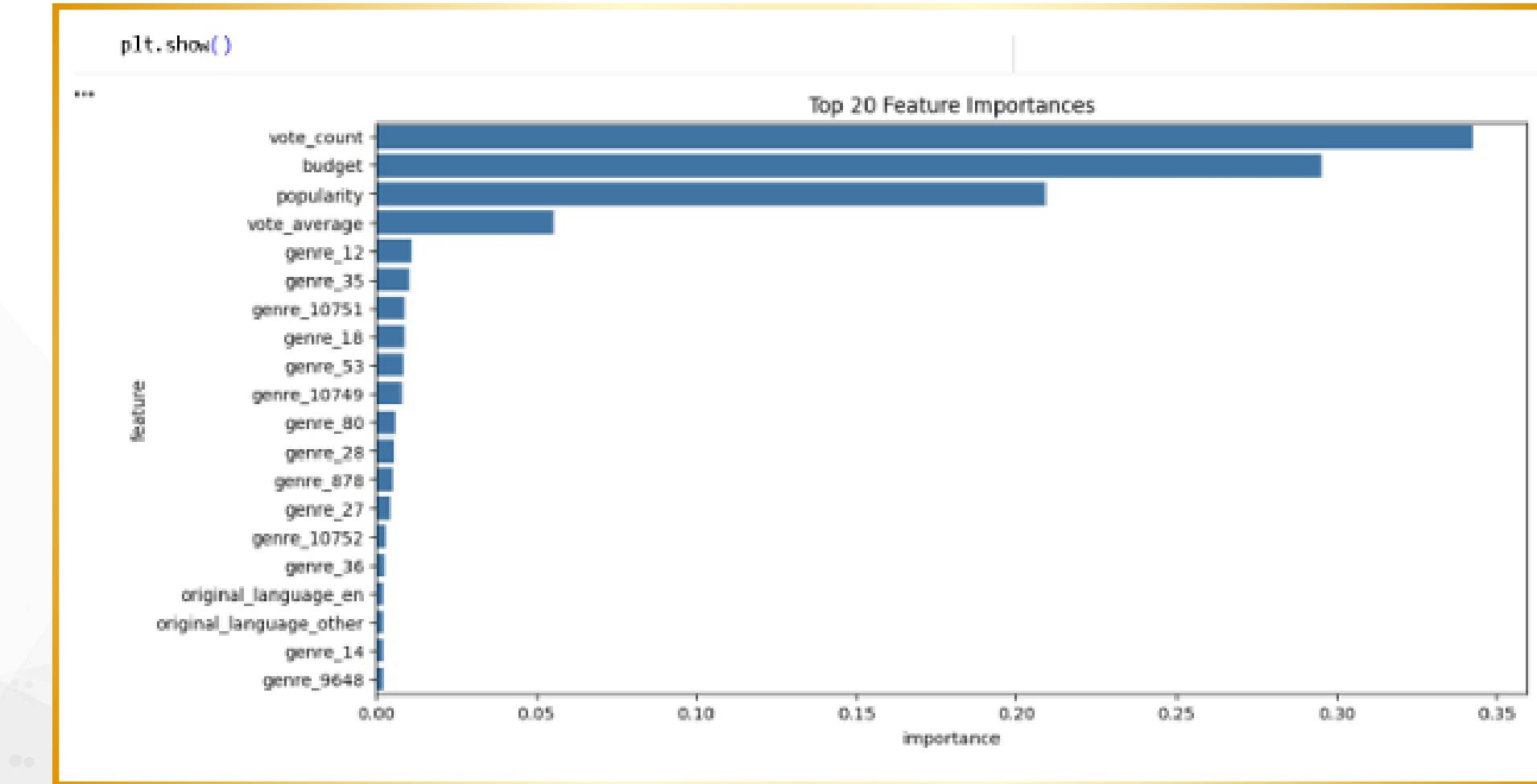
4. PHÂN TÍCH KẾT QUẢ

4.3. Residual Analysis



Hình 3: Biểu đồ residual analysis

4.4. Feature Importance



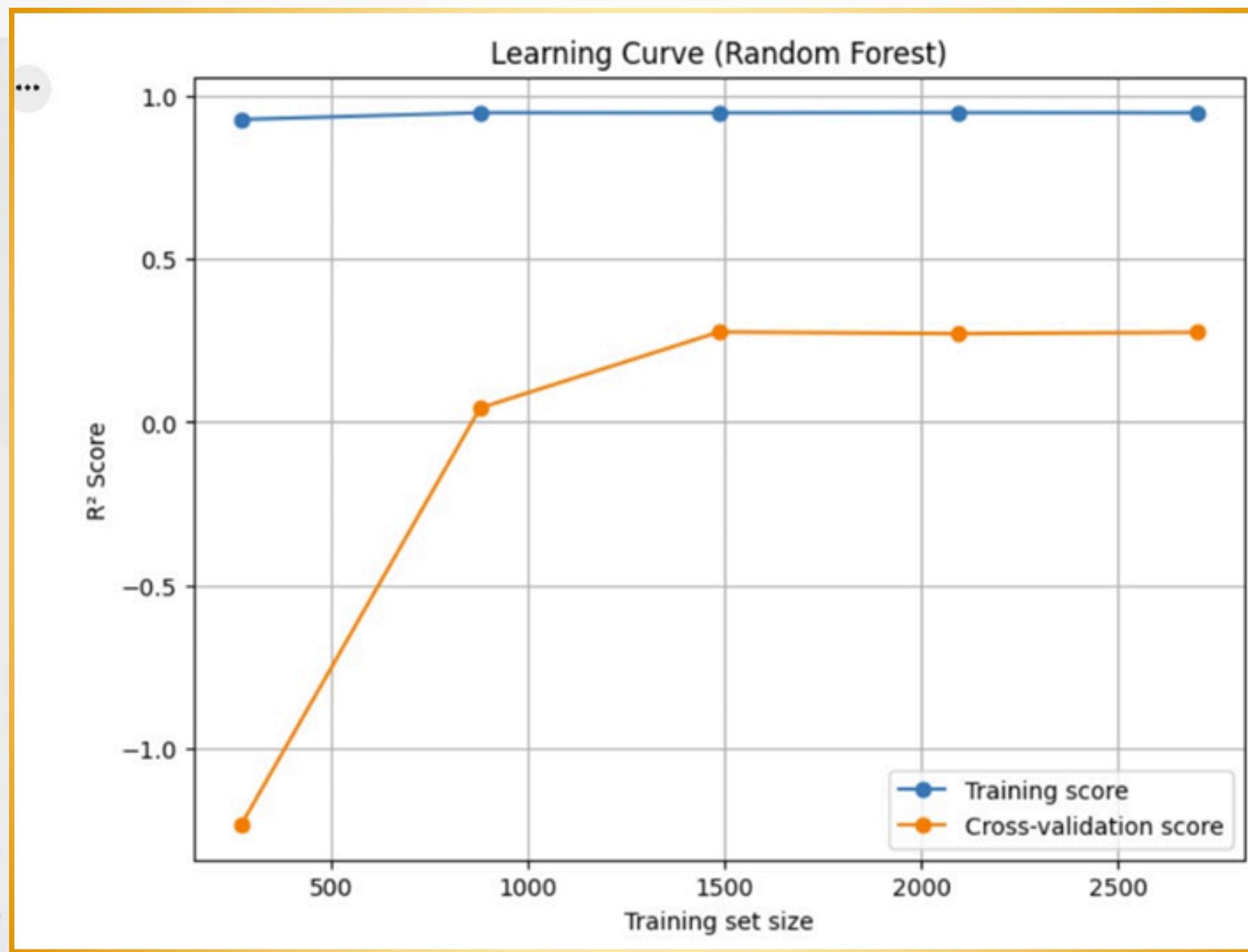
Hình 4: Các feature quan trọng của mô hình

- budget
- popularity
- vote_count
- vote_average

Sai số tăng mạnh ở các phim doanh thu cao
→ cho thấy mô hình khó học được các phim
có doanh thu cao, phim bom tấn

4. PHÂN TÍCH KẾT QUẢ

4.5. Learning Curve

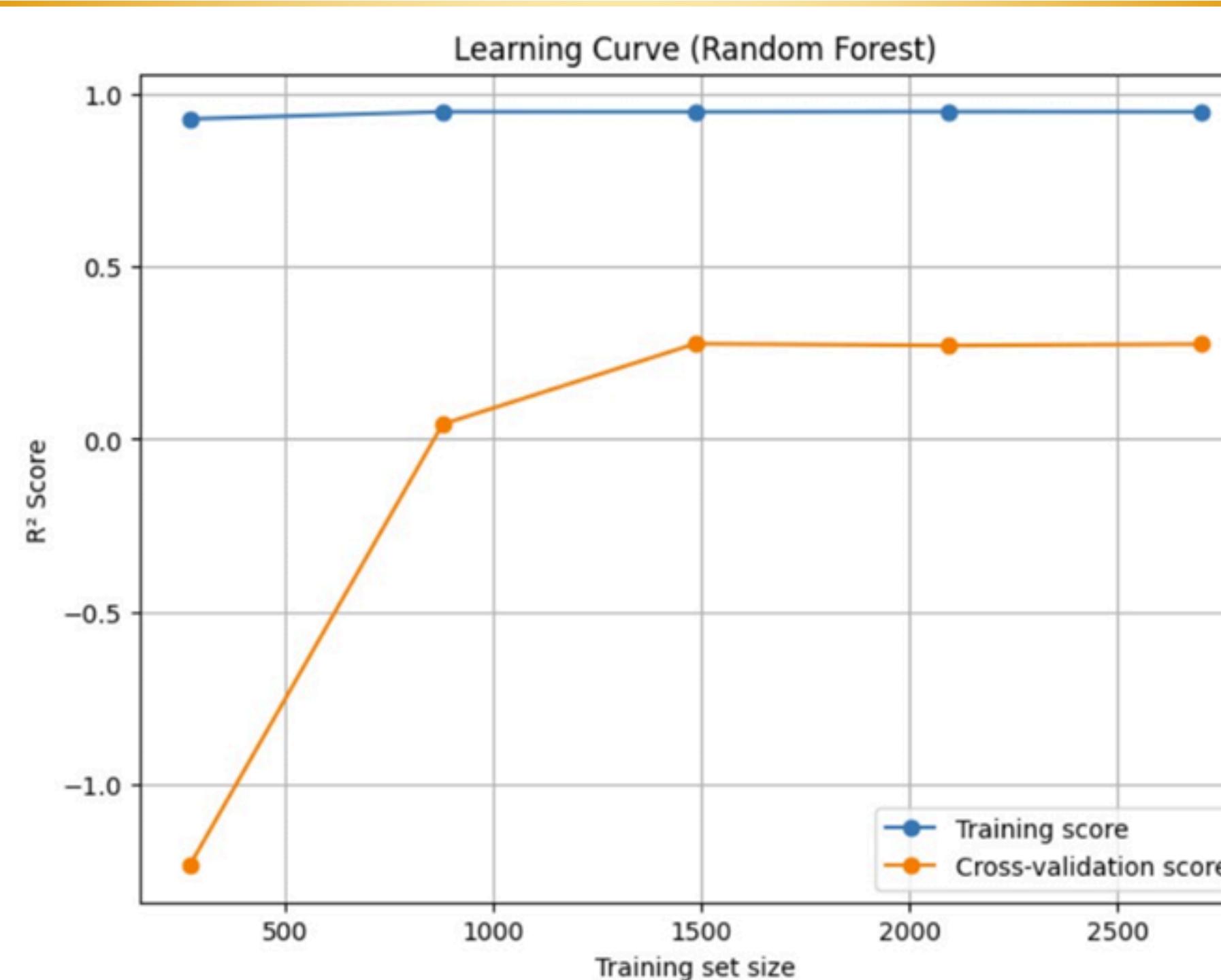


Hình 5: Learning curve

- Training score luôn duy trì ở mức rất cao (xấp xỉ 0.95–1.0), cho thấy mô hình học rất tốt trên tập huấn luyện. Đây là dấu hiệu đặc trưng của hiện tượng overfitting.
- Cross-validation score ban đầu rất thấp, thậm chí âm, cho thấy mô hình dự đoán kém khi lượng dữ liệu huấn luyện nhỏ. Khi kích thước tập huấn luyện tăng lên, điểm số cải thiện dần và ổn định quanh mức $R^2 \approx 0.15–0.20$.
- Khoảng cách lớn giữa đường training score và cross-validation score thể hiện rằng mô hình không tổng quát hoá tốt và đang bị overfit mạnh.

4. PHÂN TÍCH KẾT QUẢ

4.5. Learning Curve

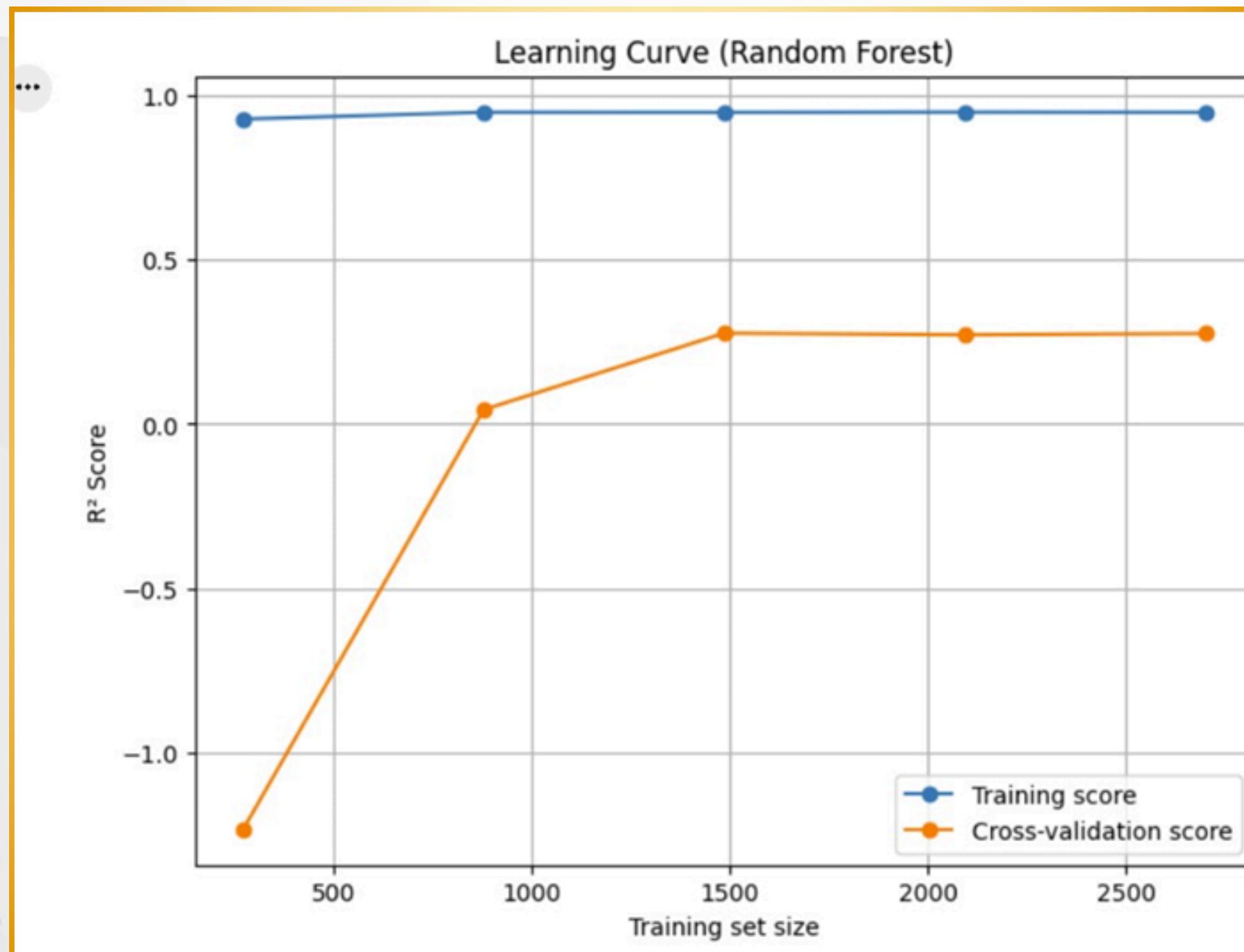


- Đường cross-validation gần như không tăng thêm khi số lượng mẫu vượt quá 1500, cho thấy việc bổ sung thêm dữ liệu huấn luyện không giúp cải thiện đáng kể hiệu suất mô hình.
- Nguyên nhân có thể đến từ đặc tính phân phối lệch phải mạnh của biến doanh thu (revenue) và sự xuất hiện của nhiều outlier, khiến mô hình Random Forest khó học được các quy luật tổng quát.

Hình 5: Learning curve

4. PHÂN TÍCH KẾT QUẢ

4.5. Learning Curve



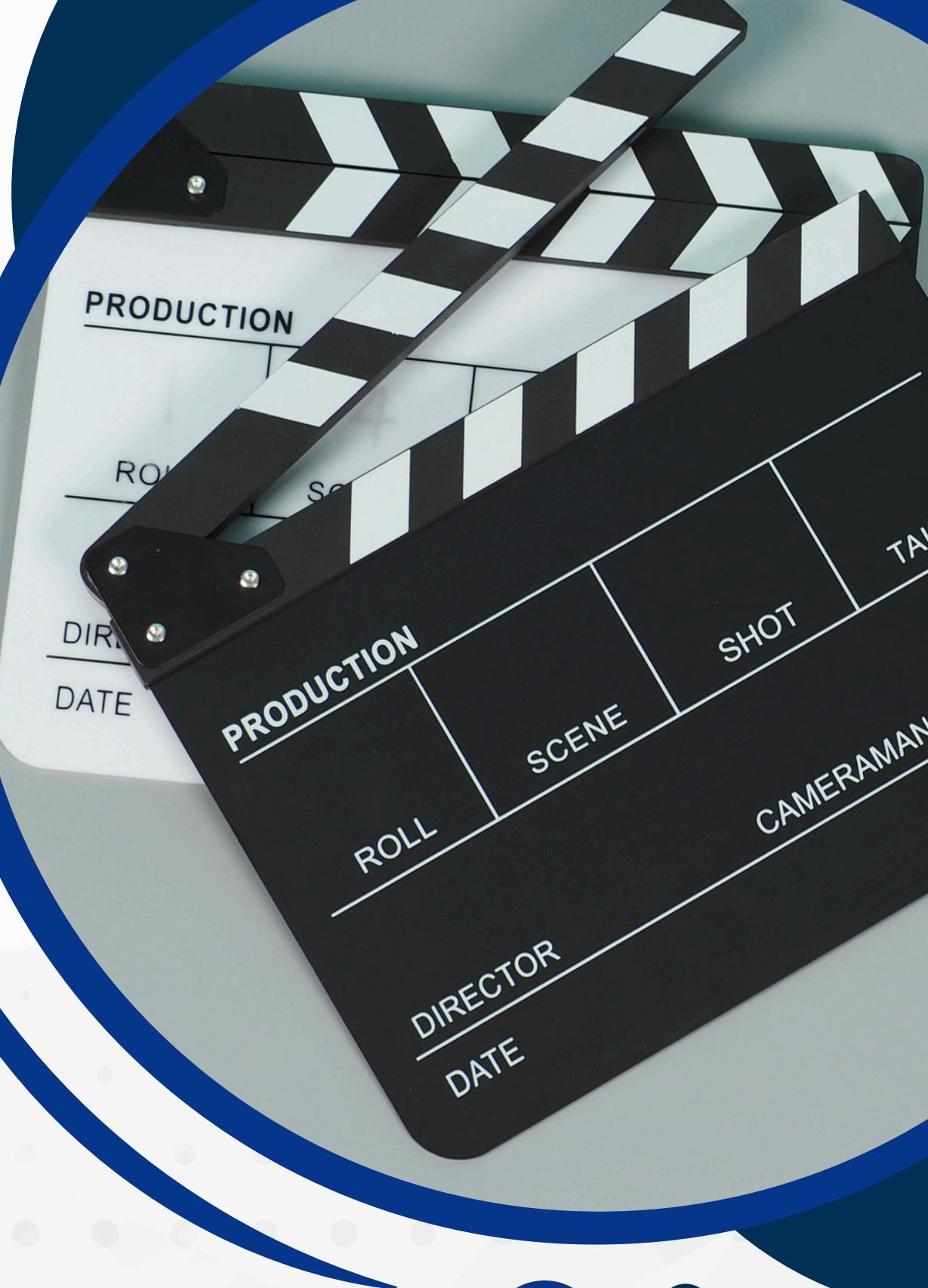
Từ biểu đồ *Learning Curve*,

có thể kết luận rằng mô hình Random Forest đang gặp vấn đề overfitting và cần các kỹ thuật nâng cao như giảm độ sâu cây, tăng min_samples_leaf, hoặc chuyển sang các mô hình mạnh hơn như Gradient Boosting, XGBoost hoặc LightGBM để cải thiện khả năng tổng quát hóa.

Hình 5: Learning curve

5. KẾT LUẬN

- Mô hình Random Forest cho kết quả tốt ở nhóm phim doanh thu thấp–trung bình.
- Hiện tượng overfitting xuất hiện do dữ liệu có nhiều outliers.
- Có thể cải thiện bằng các mô hình mạnh hơn như XGBoost, LightGBM.
- Có thể tune các siêu tham số của mô hình random forest để chống overfitting



NHÓM 8

THANK YOU
for
YOUR ATTENTION