In [ ]:
```python
import requests
from bs4 import BeautifulSoup
import pandas as pd
import numpy as np
import time
import json
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.ensemble import RandomForestRegressor
from sklearn.metrics import mean_squared_error, r2_score


def get_book_data():
    books_data = []

    url = "https://tiki.vn/api/v2/products"

    headers = {
        'User-Agent': 'Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/91.0.4472.124 Safari/537.36',
        'Accept': 'application/json'
    }

    params = {
        'limit': 40,
        'category': 8322,
        'page': 1,
        'sort': 'top_seller'
    }
    total_items = 0

    try:
        while total_items < 1000:
            print(f"Đang thu thập trang {params['page']}...")

            response = requests.get(url, headers=headers, params=params)

            if response.status_code != 200:
                print(f"Lỗi khi tải trang {params['page']}: {response.status_code}")
                break

            data = response.json()

            if not data.get('data'):
                break

            for product in data['data']:
                try:
                    book_info = {
                        'id': product.get('id', ''),
                        'ten': product.get('name', ''),
                        'gia': product.get('price', 0),
                        'gia_goc': product.get('original_price', 0),
                        'danh_gia': product.get('rating_average', 0),
                        'so_danh_gia': product.get('review_count', 0),
                        'nha_cung_cap': product.get('seller_name', ''),
                        'brand': product.get('brand_name', ''),
                        'ton_kho': product.get('stock_item', {}).get('qty', 0)
                    }
                    books_data.append(book_info)
                    total_items += 1

                except Exception as e:
                    print(f"Lỗi khi xử lý sản phẩm: {e}")
                    continue

            params['page'] += 1
            time.sleep(1)

    except Exception as e:
        print(f"Lỗi khi thu thập dữ liệu: {e}")

    df = pd.DataFrame(books_data)
    df['ton_kho'] = pd.to_numeric(df['ton_kho'], errors='coerce').fillna(0)
    df['danh_gia'] = pd.to_numeric(df['danh_gia'], errors='coerce').fillna(0)
    df['so_danh_gia'] = pd.to_numeric(df['so_danh_gia'], errors='coerce').fillna(0)
    return df


def analyze_books(df):
    analysis = {
        'Tổng số sách': len(df),
        'Giá trung bình': f"{df['gia'].mean():,.0f} VND",
        'Giá cao nhất': f"{df['gia'].max():,.0f} VND",
        'Giá thấp nhất': f"{df['gia'].min():,.0f} VND",
        'Đánh giá trung bình': round(df['danh_gia'].mean(), 2),
        'Số lượng đánh giá trung bình': round(df['so_danh_gia'].mean(), 2),
        'Top 5 nhà cung cấp': df['nha_cung_cap'].value_counts().head(5).to_dict()
    }

    df['muc_gia'] = pd.qcut(df['gia'], q=4, labels=['Thấp', 'Trung bình thấp', 'Trung bình cao', 'Cao'])
    price_analysis = df.groupby('muc_gia', observed=True).agg({
        'gia': 'mean',
        'danh_gia': 'mean',
        'so_danh_gia': 'sum'
    }).round(2)

    return analysis, price_analysis


def build_ml_model(df):
    features = ['danh_gia', 'so_danh_gia', 'ton_kho']
    X = df[features]
    y = df['gia']

    X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

    scaler = StandardScaler()
    X_train_scaled = scaler.fit_transform(X_train)
    X_test_scaled = scaler.transform(X_test)

    model = RandomForestRegressor(n_estimators=100, random_state=42)
    model.fit(X_train_scaled, y_train)

    y_pred = model.predict(X_test_scaled)

    mse = mean_squared_error(y_test, y_pred)
    rmse = np.sqrt(mse)
    r2 = r2_score(y_test, y_pred)

    feature_importance = dict(zip(features, model.feature_importances_))

    model_report = {
        'rmse': int(rmse),
        'r2': round(r2, 3),
        'feature_importance': feature_importance,
        'model': model,
        'scaler': scaler
    }

    return model_report


def save_results(df, analysis, price_analysis):
    df.to_excel("Cau1.xls", index=False, engine='openpyxl')

    with open('Cau1.txt', 'w', encoding='utf-8') as f:
        f.write("PHÂN TÍCH DỮ LIỆU SÁCH TIKI\n")
        f.write("=" * 50 + "\n\n")

        for key, value in analysis.items():
            f.write(f"{key}: {value}\n")

        f.write("\n\nPHÂN TÍCH THEO MỨC GIÁ\n")
        f.write("=" * 50 + "\n")
        f.write(price_analysis.to_string())

def main():
    print("Bắt đầu thu thập dữ liệu...")
    df = get_book_data()
    print("\nPhân tích dữ liệu...")
    analysis, price_analysis = analyze_books(df)

    print("\nXây dựng mô hình học máy...")
    model_report = build_ml_model(df)

    print("\nKết quả mô hình học máy:")
    print(f"RMSE: {model_report['rmse']:,} VND")
    print(f"R2 Score: {model_report['r2']}")
    print("\nTầm quan trọng của các features:")
    for feature, importance in model_report['feature_importance'].items():
        print(f"{feature}: {importance:.3f}")

    print("\nLưu kết quả...")
    save_results(df, analysis, price_analysis)

    return df, analysis, price_analysis, model_report

if __name__ == "__main__":
    df, analysis, price_analysis, model_report = main()
```

Kết quả chương trình chi tiết

Bắt đầu thu thập dữ liệu...

- Đang thu thập trang 1...
- Đang thu thập trang 2...
- Đang thu thập trang 3...
- Đang thu thập trang 4...
- Đang thu thập trang 5...
- Đang thu thập trang 6...
- Đang thu thập trang 7...
- Đang thu thập trang 8...
- Đang thu thập trang 9...
- Đang thu thập trang 10...
- Đang thu thập trang 11...
- Đang thu thập trang 12...
- Đang thu thập trang 13...
- Đang thu thập trang 14...
- Đang thu thập trang 15...
- Đang thu thập trang 16...
- Đang thu thập trang 17...
- Đang thu thập trang 18...
- Đang thu thập trang 19...
- Đang thu thập trang 20...
- Đang thu thập trang 21...
- Đang thu thập trang 22...
- Đang thu thập trang 23...
- Đang thu thập trang 24...
- Đang thu thập trang 25...

Phân tích dữ liệu...

Xây dựng mô hình học máy...

Kết quả mô hình học máy:

Metric	Value
RMSE	118.421 VND
R2 Score	-0.478

Tầm quan trọng của các features:

Feature	Importance
danh_gia	0.139
so_danh_gia	0.861
ton_kho	0.000

Lưu kết quả...

In [ ]:
```
Đây là file tổng kết cau1.txt
```

PHÂN TÍCH DỮ LIỆU SÁCH TIKI

==

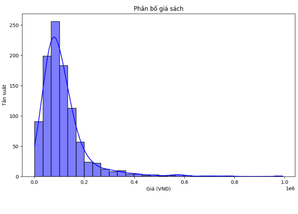
- **Tổng số sách:** 1000
- **Giá trung bình:** 115,217 VND
- **Giá cao nhất:** 992,000 VND
- **Giá thấp nhất:** 1,900 VND
- **Đánh giá trung bình:** 4.83
- **Số lượng đánh giá trung bình:** 411.7
- **Top 5 nhà cung cấp:**
  - Tiki Trading: 729
  - Nhà sách Fahasa: 34
  - Deli Official Store: 30
  - Phúc Minh Books: 28
  - Dtpbooks: 24

PHÂN TÍCH THEO MỨC GIÁ

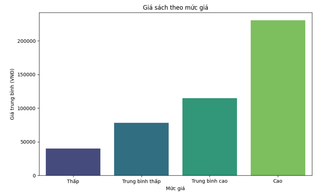
==

Mức giá	Giá (VND)	Đánh giá	Số lượng đánh giá
Thấp	39,803.09	4.82	99,078
Trung bình thấp	77,777.97	4.87	129,098
Trung bình cao	114,553.45	4.80	98,307
Cao	230,410.31	4.84	85,213

Biểu đồ phân bố giá sách



Biểu đồ phân tích giá sách theo từng mức giá


Giá sách theo mức giá
Giá trung bình (VND)
0
50000
100000
150000
200000
Thấp
Trung bình thấp
Trung bình cao
Cao
Mức giá