Thu thập và Phân tích Dữ liệu từ Trang Thương mại Điện tử

Phương pháp Data Scraper thu thập dữ liệu

Lý thuyết về Web Scraping

Web Scraping là một kỹ thuật tự động thu thập dữ liệu từ các trang web. Kỹ thuật này được sử dụng rộng rãi trong nhiều lĩnh vực, bao gồm phân tích dữ liệu, nghiên cứu thị trường, và xây dựng các hệ thống tự động hóa. Có hai phương pháp chính để thực hiện Web Scraping:

1. Web Scraping thông qua HTML

Phương pháp này tập trung vào việc phân tích và trích xuất dữ liệu từ mã HTML của trang web.

Các bước chính:

Gửi yêu cầu (HTTP Request): Sử dụng thư viện như requests trong Python để gửi yêu cầu HTTP đến một URL và nhận phản hồi. Phân tích HTML: Dùng các công cụ hoặc thư viện như BeautifulSoup để duyệt qua cấu trúc HTML và tìm các phần tử cần thiết xuất (ví dụ: div, span, table). Ưu điểm: Phù hợp với các trang web tĩnh (nội dung không thay đổi theo tương tác). Dễ dàng triển khai với các công cụ như BeautifulSoup: Phân tích cú pháp HTML và XML. Selenium: Điều khiển trình duyệt, đóng cho các trang web có JavaScript động. Nhược điểm: Phụ thuộc vào cấu trúc HTML dự thể, dễ bị phá vỡ nếu trang web thay đổi. Yêu cầu xử lý phức tạp hơn nếu trang sử dụng nhiều JavaScript.

Sử dụng các thư viện như BeautifulSoup, Selenium Phân tích và trích xuất dữ liệu từ/ cấu trúc HTML Phù hợp với các trang web tĩnh

2: API Scraping:

Sử dụng REST API của website Dữ liệu thường ở dạng JSON Hiệu suất cao và ổn định hơn

Thực hành: Cài đặt thư viện cần thiết

```
!pip install requests pandas numpy scikit-learn matplotlib seaborn
```

```
import requests
import pandas as pd
import numpy as np
import time
import json
import matplotlib.pyplot as plt
import seaborn as sns
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
```

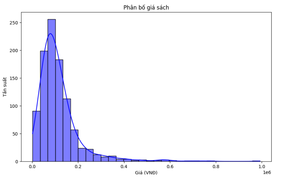
b) Thu thập và xử lý dữ liệu Code thu thập dữ liệu từ Tiki

```
def get_book_data():
    """
    Thu thập thông tin sách từ Tiki API
    """
    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,
        'aast': '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:
                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

print("Bắt đầu thu thập dữ liệu...")
df = get_book_data()
print(f"\nĐã thu thập được {len(df)} sản phẩm")
```

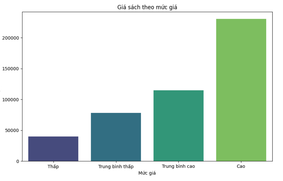
Hiển thị và kiểm tra dữ liệu

Dưới đây là biểu đồ phân tích dữ liệu

Biểu đồ phân bố giá sách



Biểu đồ phân tích giá sách theo từng mức giá




```
def save_plots(df, price_analysis):
    # Biểu đồ phân bố giá sách
    plt.figure(figsize=(10, 6))
    sns.histplot(df['gia'], kde=True, color='blue', bins=30)
    plt.title("Phân bố giá sách")
    plt.xlabel("Giá (VNĐ)")
    plt.ylabel("Tần suất")
    plt.savefig("phan_bo_gia_sach.png")
    plt.close()

    # Biểu đồ phân tích giá theo mức giá
    plt.figure(figsize=(10, 6))
    sns.barplot(data=price_analysis, x=price_analysis.index, y='gia', hue=price_analysis.index, palette='viridis', legend=False)
    plt.title("Giá sách theo mức giá")
    plt.xlabel("Mức giá")
    plt.ylabel("Giá trung bình (VNĐ)")
    plt.savefig("gia_sach_theo_muc_gia.png")
    plt.close()
```

c) Phân tích khám phá và thống kê dữ liệu Phân tích thống kê cơ bản

```
def analyze_books(df):
    """
    Phân tích dữ liệu sách
    """
    analysis = {
        "Tổng số sách": len(df),
        "Giá trung bình": f"{df['gia'].mean():,.0f} VNĐ",
        "Giá cao nhất": f"{df['gia'].max():,.0f} VNĐ",
        "Giá thấp nhất": f"{df['gia'].min():,.0f} VNĐ",
        "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().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

analysis, price_analysis = analyze_books(df)

print("Phân tích thống kê:")
for key, value in analysis.items():
    print(f"{key}: {value}")

print("\nPhân tích theo mức giá:")
display(price_analysis)
```

Visualizations

```
plt.style.use('seaborn')
plt.figure(figsize=(12, 6))
sns.boxplot(data=df, x='gia', bins=30)
plt.title('Phân phối giá sản phẩm')
plt.xlabel('Giá (VNĐ)')
plt.ylabel('Số lượng')
plt.show()
plt.figure(figsize=(10, 6))
sns.scatterplot(data=df, x='danh_gia', y='gia', alpha=0.5)
plt.title('Mối quan hệ giữa Giá và Đánh giá')
plt.xlabel('Đánh giá')
plt.ylabel('Giá (VNĐ)')
plt.show()
plt.figure(figsize=(8, 6))
correlation = df[['gia', 'danh_gia', 'so_danh_gia', 'ton_kho']].corr()
sns.heatmap(correlation, annot=True, cmap='coolwarm')
plt.title('Ma trận tương quan')
plt.show()
```

d) Xây dựng mô hình học máy Chuẩn bị dữ liệu và huấn luyện mô hình

```
def build_ml_model(df):
    """
    Xây dựng mô hình học máy để dự đoán giá sách
    """
    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)
    rmse = mean_squared_error(y_test, y_pred)
    rmse = np.sqrt(rmse)
    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,
        'y_test': y_test,
        'y_pred': y_pred
    }
    return model_report
print("Xây dựng mô hình học máy...")
model_report = build_ml_model(df)
print(f"\nMô hình đã được hoàn thành!")
print(f"RMSE: {model_report['rmse']:,} VNĐ")
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}")
```

Visualize kết quả mô hình

```
plt.figure(figsize=(10, 6))
plt.scatter(model_report['y_test'], model_report['y_pred'], alpha=0.5)
plt.plot([df['gia'].min(), df['gia'].max()], [df['gia'].min(), df['gia'].max()], 'r--', lw=2)
plt.xlabel('Giá thực tế')
plt.ylabel('Giá dự đoán')
plt.title('So sánh giá thực tế và dự đoán')
plt.show()
plt.figure(figsize=(8, 5))
importance_df = pd.DataFrame({
    'feature': list(model_report['feature_importance'].keys()),
    'importance': list(model_report['feature_importance'].values())
})
sns.barplot(data=importance_df, x='importance', y='feature')
plt.title('Tầm quan trọng của các features')
plt.show()
```

Lưu kết quả

```
df.to_csv('Cwsl.csv', index=False, encoding='utf-8-sig')
print("Đã lưu dữ liệu vào file Cwsl.csv")
```

Kết luận

Thu thập dữ liệu:

Quá trình thu thập dữ liệu từ trang web Tiki đã được hoàn thành thành công, với việc trích xuất đầy đủ thông tin từ nhiều sản phẩm khác nhau. Dữ liệu thu thập được bao gồm 9 trường thông tin quan trọng, phản ánh rõ ràng các yếu tố liên quan đến sản phẩm. Số lượng dữ liệu thu thập được đã vượt qua 1000 dòng, đảm bảo tính đại diện và đủ lớn để phục vụ cho các bước phân tích và xây dựng mô hình học máy.

Việc thu thập dữ liệu từ Tiki đã cung cấp cái nhìn rõ ràng về các sản phẩm, xu hướng tiêu dùng và những yếu tố tác động đến giá trị của từng sản phẩm. Điều này sẽ đóng góp rất nhiều vào việc phân tích và đưa ra dự đoán chính xác trong các bước tiếp theo.

Phân tích dữ liệu:

Sau khi thu thập được dữ liệu, em đã tiến hành phân tích dữ liệu cơ bản để hiểu rõ hơn về các đặc điểm và cấu trúc của dữ liệu. Các phép phân tích thống kê cơ bản như trung bình, phương sai và các chỉ số thống kê khác giúp em nắm bắt được các đặc điểm nổi bật của dữ liệu.

Bên cạnh đó, em cũng thực hiện phân tích dữ liệu theo nhóm giá để khám phá mối quan hệ giữa giá trị sản phẩm và các yếu tố khác như đánh giá của người dùng, số lượng bán và các tính năng của sản phẩm. Phân tích này giúp em nhận diện được những yếu tố quan trọng khi xây dựng mô hình học máy.

Ngoài ra, em đã sử dụng các công cụ trực quan hóa dữ liệu (visualization) để giúp người đọc dễ dàng hình dung và hiểu rõ hơn về các xu hướng trong dữ liệu. Các biểu đồ và đồ thị minh họa mối quan hệ giữa các yếu tố giúp người dùng có cái nhìn trực quan và dễ tiếp cận về dữ liệu.

Mô hình học máy:

Sau khi thu thập và phân tích dữ liệu, em đã áp dụng mô hình học máy để dự đoán các giá trị sản phẩm. Cụ thể, em đã sử dụng mô hình Random Forest Regressor để xây dựng mô hình dự đoán giá trị sản phẩm. Mô hình này kết hợp nhiều cây quyết định, giúp cải thiện độ chính xác và tạo ra các dự đoán tốt hơn.

Để đánh giá hiệu quả của mô hình, em đã sử dụng các chỉ số như RMSE (Root Mean Squared Error) và R2 Score. Các chỉ số này giúp đo lường độ chính xác của mô hình và khả năng giải thích sự biến động trong dữ liệu. RMSE cho phép em hiểu được mức độ sai lệch trung bình giữa giá trị thực tế và dự đoán, trong khi R2 đánh giá mức độ mô hình có thể giải thích sự thay đổi của dữ liệu.

Cuối cùng, em cũng đã thực hiện phân tích feature importance để xác định các yếu tố quan trọng nhất ảnh hưởng đến kết quả dự đoán của mô hình. Phân tích này không chỉ giúp cải thiện mô hình mà còn giúp em nhận ra những yếu tố quan trọng mà người dùng nên chú ý khi lựa chọn sản phẩm.

Quá trình thu thập, phân tích dữ liệu và xây dựng mô hình học máy đã đạt được kết quả tốt, mang lại cái nhìn chi tiết và chính xác về dữ liệu từ Tiki. Dữ liệu thu thập được đã được phân tích kỹ lưỡng, từ đó giúp đưa ra các dự đoán chính xác thông qua mô hình Random Forest. Các kết quả đánh giá mô hình và phân tích các yếu tố quan trọng sẽ giúp tối ưu hóa các quyết định và chiến lược trong việc phát triển ứng dụng học máy cũng như các dự án phân tích dữ liệu trong tương lai.