Hướng dẫn xây dựng ứng dụng phân tích và dự đoán dữ liệu bán hàng bằng Python và Tkinter

Tác giả: Đặng Kim Thi

Mục lục

1. Giới thiệu dự án
2. Mục tiêu học tập
3. Công nghệ sử dụng
    - 3.1. Python
    - 3.2. Tkinter
    - 3.3. Pandas
    - 3.4. Matplotlib
    - 3.5. Scikit-learn
    - 3.6. NumPy
4. Cấu trúc dự án
5. Hướng dẫn thực hiện
    - 5.1. Thiết lập môi trường ảo và cài đặt thư viện
    - 5.2. Tạo dữ liệu mẫu
    - 5.3. Phát triển ứng dụng chính
    - 5.4. Kiểm tra và chạy ứng dụng
6. Kết luận

---

1. Giới thiệu dự án

Dự án này phát triển một ứng dụng desktop sử dụng Python và Tkinter để phân tích và dự đoán dữ liệu bán hàng từ file CSV. Ứng dụng cung cấp giao diện người dùng trực quan với các chức năng chính:

- Tải file CSV chứa dữ liệu bán hàng (bao gồm ngày và doanh thu).
- Hiển thị thống kê cơ bản và biểu đồ xu hướng doanh thu theo thời gian.
- Dự đoán doanh thu cho 30 ngày tiếp theo bằng mô hình học máy.

Dự án được thiết kế cho người mới bắt đầu với Python cơ bản, nhằm cung cấp trải nghiệm thực hành về lập trình giao diện, xử lý dữ liệu, và ứng dụng học máy đơn giản.

---

2. Mục tiêu học tập

Hoàn thành dự án này giúp người học đạt được các kỹ năng sau:

- Phát triển giao diện người dùng với Tkinter.
- Xử lý dữ liệu định dạng CSV bằng Python.
- Trực quan hóa dữ liệu thông qua biểu đồ.
- Ứng dụng mô hình học máy cơ bản (hồi quy tuyến tính) để dự đoán.
- Kết hợp các thư viện Python để xây dựng một ứng dụng hoàn chỉnh.

Dự án cung cấp nền tảng để người học mở rộng kiến thức về lập trình và phân tích dữ liệu.

---

3. Công nghệ sử dụng

3.1. Python

- **Mô tả**: Python là ngôn ngữ lập trình cấp cao, dễ học, được sử dụng rộng rãi trong phát triển phần mềm, phân tích dữ liệu, và học máy.
- **Ứng dụng**: Là nền tảng chính để triển khai toàn bộ dự án.
- **Tài liệu tham khảo**: Trang chính thức Python

3.2. Tkinter

- **Mô tả**: Tkinter là thư viện giao diện người dùng mặc định của Python, hỗ trợ tạo cửa sổ, nút bấm, và các thành phần giao diện khác.
- **Ứng dụng**: Xây dựng giao diện đồ họa cho ứng dụng.
- **Tài liệu tham khảo**: Tài liệu Tkinter

3.3. Pandas

- **Mô tả**: Pandas là thư viện xử lý dữ liệu mạnh mẽ, cung cấp các công cụ để đọc, phân tích, và thao tác dữ liệu dạng bảng.
- **Ứng dụng**: Đọc và xử lý dữ liệu từ file CSV.
- **Tài liệu tham khảo**: Trang chính thức Pandas

3.4. Matplotlib

- **Mô tả**: Matplotlib là thư viện trực quan hóa dữ liệu, hỗ trợ vẽ các loại biểu đồ như đường, cột, và phân tán.
- **Ứng dụng**: Tạo biểu đồ hiển thị xu hướng doanh thu.
- **Tài liệu tham khảo**: Trang chính thức Matplotlib

3.5. Scikit-learn

- **Mô tả**: Scikit-learn là thư viện học máy phổ biến, cung cấp các mô hình và công cụ để huấn luyện và dự đoán.
- **Ứng dụng**: Triển khai mô hình hồi quy tuyến tính cho dự đoán doanh thu.
- **Tài liệu tham khảo**: Trang chính thức Scikit-learn

3.6. NumPy

- **Mô tả**: NumPy là thư viện hỗ trợ tính toán số học trên mảng và ma trận, tối ưu hóa hiệu suất xử lý dữ liệu.
- **Ứng dụng**: Hỗ trợ các phép tính trong Scikit-learn và Pandas.
- **Tài liệu tham khảo**: Trang chính thức NumPy

---

4. Cấu trúc dự án

Dự án được tổ chức trong một thư mục như sau:

```
SalesAnalysisApp/
│
├── generate_sales.py    # Script tạo dữ liệu mẫu
├── sales_app.py         # Ứng dụng chính
├── requirements.txt     # Danh sách thư viện cần cài đặt
└── sales_data.csv       # File dữ liệu mẫu (được tạo tự động)
```

- `generate_sales.py`: Tạo file CSV chứa dữ liệu bán hàng giả lập.
- `sales_app.py`: Chứa mã nguồn ứng dụng Tkinter.
- `requirements.txt`: Liệt kê các thư viện cần thiết để chạy dự án.
- `sales_data.csv`: File dữ liệu đầu vào với các cột `Date` và `Sales`.

---

5. Hướng dẫn thực hiện

5.1. Thiết lập môi trường ảo và cài đặt thư viện

Lý do sử dụng môi trường ảo

Môi trường ảo (virtual environment) được sử dụng để:

- **Tránh xung đột phiên bản**: Các thư viện trong dự án có thể không tương thích với các gói đã cài đặt toàn cục trên máy (ví dụ: `numpy 1.26.4` xung đột với yêu cầu `numpy<1.24` của một số gói khác).
- **Cô lập dự án**: Đảm bảo mỗi dự án có môi trường riêng, không ảnh hưởng lẫn nhau.
- **Dễ quản lý**: Chỉ cài các thư viện cần thiết cho dự án, giảm nguy cơ lỗi phụ thuộc.

Các bước thực hiện

1. **Kiểm tra Python**: Mở terminal (Windows: Command Prompt; Mac/Linux: Terminal) và chạy:

   ```
   python --version
   ```

   Nếu Python chưa được cài đặt, tải từ python.org (khuyến nghị phiên bản 3.8-3.10).
2. **Tạo thư mục dự án**: Tạo thư mục `SalesAnalysisApp` và di chuyển vào thư mục:

   ```
   mkdir SalesAnalysisApp
   cd SalesAnalysisApp
   ```

3. **Tạo môi trường ảo**: Chạy lệnh:

   ```
   python -m venv venv
   ```

- `venv` là tên thư mục chứa môi trường ảo.

4. **Kích hoạt môi trường ảo**:
   - Windows:

```
venv\Scripts\activate
```

   - Mac/Linux:

```
source venv/bin/activate
```

   Sau khi kích hoạt, dấu nhắc lệnh sẽ hiển thị `(venv)` phía trước.
5. **Tạo file `requirements.txt`**: Tạo file văn bản `requirements.txt` với nội dung:

```
pandas==1.5.3
matplotlib==3.6.0
scikit-learn==1.2.0
numpy==1.23.5
```

   - Các phiên bản được chọn để đảm bảo tính tương thích và ổn định với Python 3.8-3.10.
   - Tkinter không cần liệt kê vì đã có sẵn trong Python.
6. **Cài đặt thư viện**: Trong môi trường ảo, chạy:

```
pip install -r requirements.txt
```

   - Nếu lệnh không hoạt động, thử `pip3 install -r requirements.txt`.

5.2. Tạo dữ liệu mẫu

1. Tạo file `generate_sales.py` trong thư mục `SalesAnalysisApp` và sao chép mã sau:

```
import pandas as pd
import numpy as np
from datetime import datetime, timedelta
import random

def generate_sales_data(num_records=1000):
    start_date = datetime(2022, 1, 1)
    dates = [start_date + timedelta(days=i) for i in
range(num_records)]
    base_sales = 1000
    sales = [base_sales + i * 5 + random.uniform(-200, 200) for i in
range(num_records)]
```

```
    df = pd.DataFrame({'Date': dates, 'Sales': sales})
    df['Date'] = df['Date'].dt.strftime('%Y-%m-%d')
    df.to_csv('sales_data.csv', index=False)
    print(f"Đã tạo file 'sales_data.csv' với {num_records} bản ghi.")

generate_sales_data(1000)
```

2. Chạy script trong môi trường ảo:

```
python generate_sales.py
```

3. Kết quả: File `sales_data.csv` sẽ được tạo với 1000 bản ghi, ví dụ:

```
Date,Sales
2022-01-01,987.34
2022-01-02,992.15
...
```

5.3. Phát triển ứng dụng chính

1. Tạo file `sales_app.py` trong thư mục `SalesAnalysisApp` và sao chép mã sau:

```
import tkinter as tk
from tkinter import ttk, filedialog, messagebox
import pandas as pd
import matplotlib.pyplot as plt
from matplotlib.backends.backend_tkagg import FigureCanvasTkAgg
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
import numpy as np

def load_csv():
    file_path = filedialog.askopenfilename(filetypes=[("CSV files",
"*.csv")])
    if file_path:
        try:
            global df
            df = pd.read_csv(file_path)
            lbl_status.config(text=f"Đã tải: {file_path.split('/')
[-1]} | {len(df)} bản ghi")
            btn_analyze.config(state="normal")
            btn_predict.config(state="normal")
        except Exception as e:
            messagebox.showerror("Lỗi", f"Không thể tải file: {e}")

def analyze_data():
```

```
    if 'df' not in globals():
        messagebox.showwarning("Cảnh báo", "Vui lòng tải file CSV
trước!")
        return
    analysis_window = tk.Toplevel(root)
    analysis_window.title("Phân tích dữ liệu bán hàng")
    analysis_window.geometry("800x600")
    analysis_window.configure(bg="#f0f0f0")
    stats = df.describe().to_string()
    stats_label = tk.Label(analysis_window, text="Thống kê cơ bản:",
font=("Arial", 12, "bold"), bg="#f0f0f0")
    stats_label.pack(pady=5)
    stats_text = tk.Text(analysis_window, height=10, width=80)
    stats_text.insert(tk.END, stats)
    stats_text.pack(pady=5)
    if 'Sales' in df.columns and 'Date' in df.columns:
        fig, ax = plt.subplots(figsize=(8, 4))
        df.plot(x='Date', y='Sales', ax=ax, title="Doanh thu theo thời
gian", color="blue")
        ax.set_xlabel("Ngày")
        ax.set_ylabel("Doanh thu")
        plt.xticks(rotation=45)
        canvas = FigureCanvasTkAgg(fig, master=analysis_window)
        canvas.draw()
        canvas.get_tk_widget().pack(pady=10)
    else:
        messagebox.showwarning("Cảnh báo", "File CSV cần có cột 'Date'
và 'Sales'!")

def predict_sales():
    if 'df' not in globals():
        messagebox.showwarning("Cảnh báo", "Vui lòng tải file CSV
trước!")
        return
    if 'Sales' not in df.columns or 'Date' not in df.columns:
        messagebox.showwarning("Cảnh báo", "File CSV cần có cột 'Date'
và 'Sales'!")
        return
    df['Date'] = pd.to_datetime(df['Date'])
    df['Days'] = (df['Date'] - df['Date'].min()).dt.days
    X = df[['Days']]
    y = df['Sales']
    X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.2, random_state=42)
    model = LinearRegression()
    model.fit(X_train, y_train)
    score = model.score(X_test, y_test)
    last_day = df['Days'].max()
    future_days = np.array(range(last_day + 1, last_day +
31)).reshape(-1, 1)
    predictions = model.predict(future_days)
    predict_window = tk.Toplevel(root)
    predict_window.title("Dự đoán doanh thu")
    predict_window.geometry("800x600")
```

```
    predict_window.configure(bg="#f0f0f0")
    result_label = tk.Label(predict_window, text=f"Độ chính xác mô
hình: {score:.2%}", font=("Arial", 12, "bold"), bg="#f0f0f0")
    result_label.pack(pady=5)
    fig, ax = plt.subplots(figsize=(8, 4))
    ax.plot(df['Days'], df['Sales'], label="Doanh thu thực tế",
color="blue")
    ax.plot(future_days, predictions, label="Dự đoán 30 ngày",
color="red", linestyle="--")
    ax.set_title("Dự đoán doanh thu")
    ax.set_xlabel("Số ngày")
    ax.set_ylabel("Doanh thu")
    ax.legend()
    canvas = FigureCanvasTkAgg(fig, master=predict_window)
    canvas.draw()
    canvas.get_tk_widget().pack(pady=10)

root = tk.Tk()
root.title("Phân tích & Dự đoán dữ liệu bán hàng")
root.geometry("600x400")
root.configure(bg="#e0e0e0")
title_label = tk.Label(root, text="Ứng dụng phân tích dữ liệu bán
hàng", font=("Arial", 16, "bold"), bg="#e0e0e0", fg="#333333")
title_label.pack(pady=20)
btn_load = ttk.Button(root, text="Tải file CSV", command=load_csv)
btn_load.pack(pady=10)
lbl_status = tk.Label(root, text="Chưa tải file nào", font=("Arial",
10), bg="#e0e0e0", fg="#666666")
lbl_status.pack(pady=5)
btn_analyze = ttk.Button(root, text="Phân tích dữ liệu",
command=analyze_data, state="disabled")
btn_analyze.pack(pady=10)
btn_predict = ttk.Button(root, text="Dự đoán doanh thu",
command=predict_sales, state="disabled")
btn_predict.pack(pady=10)
root.mainloop()
```

5.4. Kiểm tra và chạy ứng dụng

1. Đảm bảo môi trường ảo đã được kích hoạt (dấu `(venv)` xuất hiện trong terminal).
2. Chạy ứng dụng:

```
python sales_app.py
```

3. Thực hiện các thao tác:
   - Nhấn nút "Tải file CSV" và chọn file `sales_data.csv`.
   - Nhấn "Phân tích dữ liệu" để xem thống kê và biểu đồ.
   - Nhấn "Dự đoán doanh thu" để xem kết quả dự đoán 30 ngày tiếp theo.

4. Xử lý sự cố:
    - Đảm bảo file CSV có các cột `Date` và `Sales`.
    - Kiểm tra việc cài đặt thư viện từ `requirements.txt` đã hoàn tất.
5. Thoát môi trường ảo (nếu cần):

```
deactivate
```

---

6. Kết luận

Dự án này cung cấp một cách tiếp cận thực tế để học lập trình Python, từ phát triển giao diện, xử lý dữ liệu, đến ứng dụng học máy cơ bản. Việc sử dụng môi trường ảo đảm bảo tính cô lập và tránh xung đột phiên bản, tạo điều kiện thuận lợi cho việc triển khai và bảo trì. File `requirements.txt` giúp đơn giản hóa quá trình cài đặt thư viện, đảm bảo tính nhất quán trong môi trường làm việc. Người học có thể mở rộng dự án bằng cách thêm tính năng phân tích theo các yếu tố khác, cải thiện giao diện, hoặc thử nghiệm các mô hình dự đoán nâng cao hơn. Tài liệu tham khảo đã cung cấp là nguồn hỗ trợ hữu ích để nghiên cứu thêm.

Lý do cụ thể cho việc sử dụng môi trường ảo

- **Xung đột phiên bản**: Các thư viện như `numpy`, `pandas` có thể yêu cầu phiên bản khác nhau giữa các dự án. Ví dụ, dự án này dùng `numpy==1.23.5`, nhưng các gói khác trên máy (như `qiskit-ibmq-provider`) có thể yêu cầu phiên bản khác, dẫn đến lỗi.
- **Bảo vệ hệ thống toàn cục**: Cài đặt trực tiếp vào môi trường toàn cục có thể làm hỏng các gói hiện có, ảnh hưởng đến các dự án hoặc công cụ khác.
- **Tính di động**: Môi trường ảo cho phép tái tạo dự án trên máy khác mà không cần lo lắng về cấu hình hệ thống.

Phiên bản này đã tích hợp môi trường ảo một cách liền mạch và giải thích rõ lý do sử dụng. Nếu bạn cần thêm thông tin hoặc điều chỉnh, hãy cho biết!