

Đọc data, chọn 3 cột: 3, 11, 24

• Click to add a breakpoint `h1.csv')`

✓ 0.0s

- Sử dụng 4 cột 1, 2, 3

```
columns = ['3', '11', '24']
```

✓ 0.0s

```
data = df[columns]
```

✓ 0.0s

Sử dụng HMM để phân tích chuỗi với 2 trạng thái ẩn

• `from hmmlearn.hmm import GaussianHMM`

✓ 0.0s

- Lấy ra các quan sát từ dữ liệu: `observations`

```
observations = data.values
```

✓ 0.0s

- Định nghĩa mô hình HMM với 2 trạng thái ẩn

```
model = GaussianHMM(n_components=2, covariance_type="diag", n_iter=1000)
```

✓ 0.0s

```
model.fit(observations)
```

✓ 0.0s

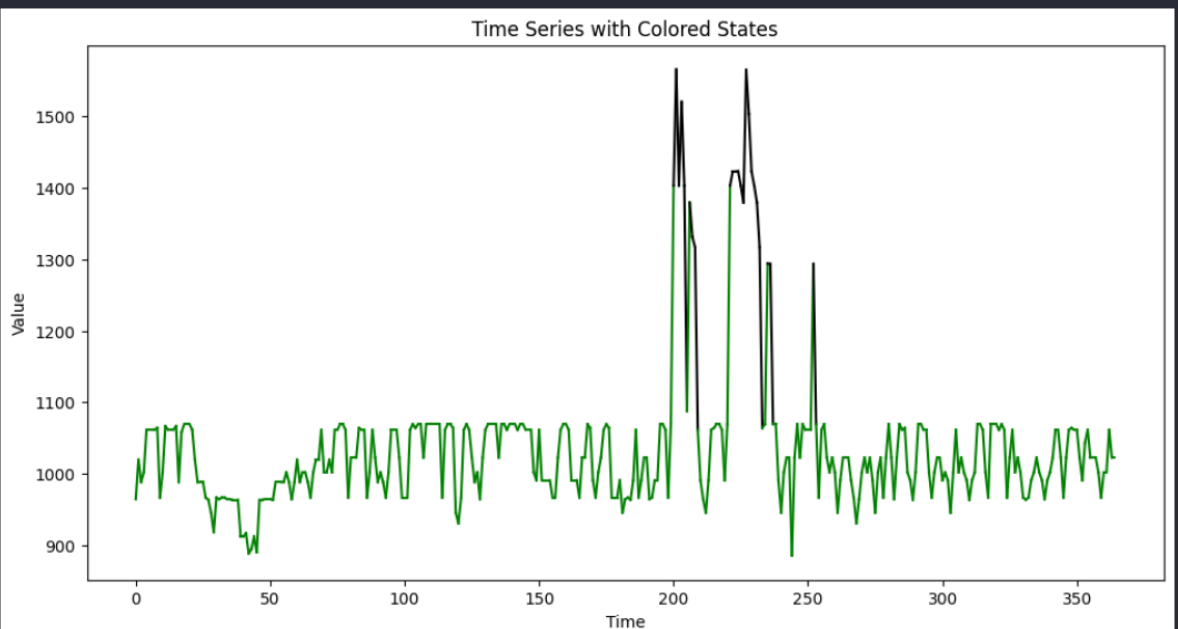
```
▼ GaussianHMM  
GaussianHMM(n_components=2, n_iter=1000)
```

```
hidden_states = model.predict(observations)
```

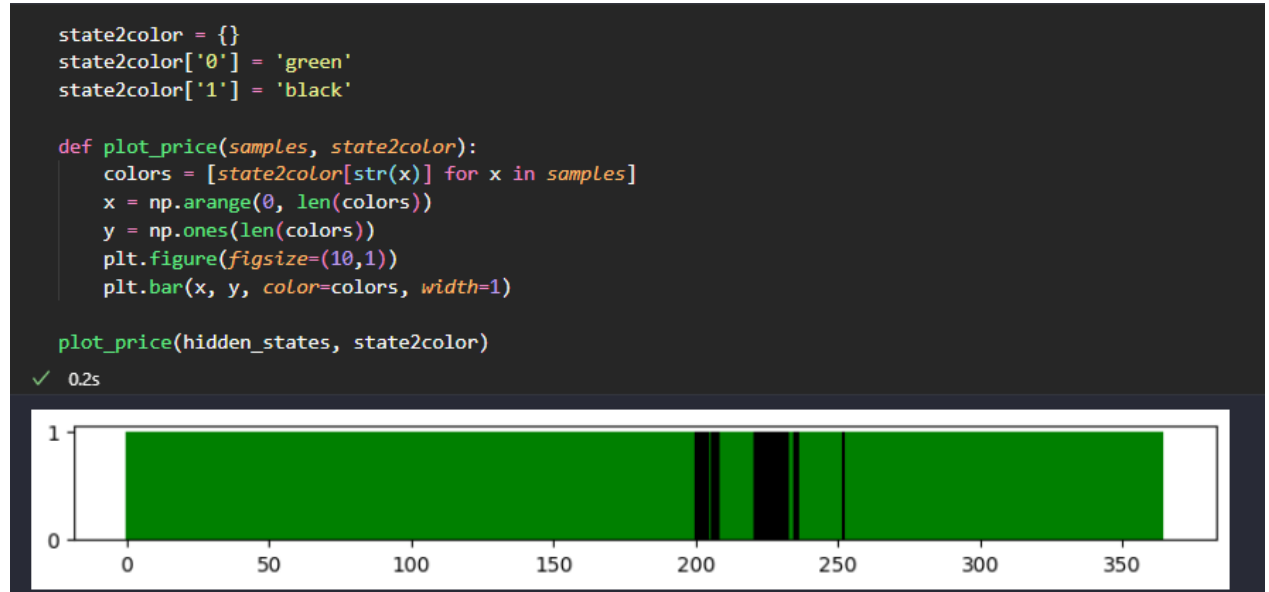
✓ 0.0s

Vẽ biểu đồ đường

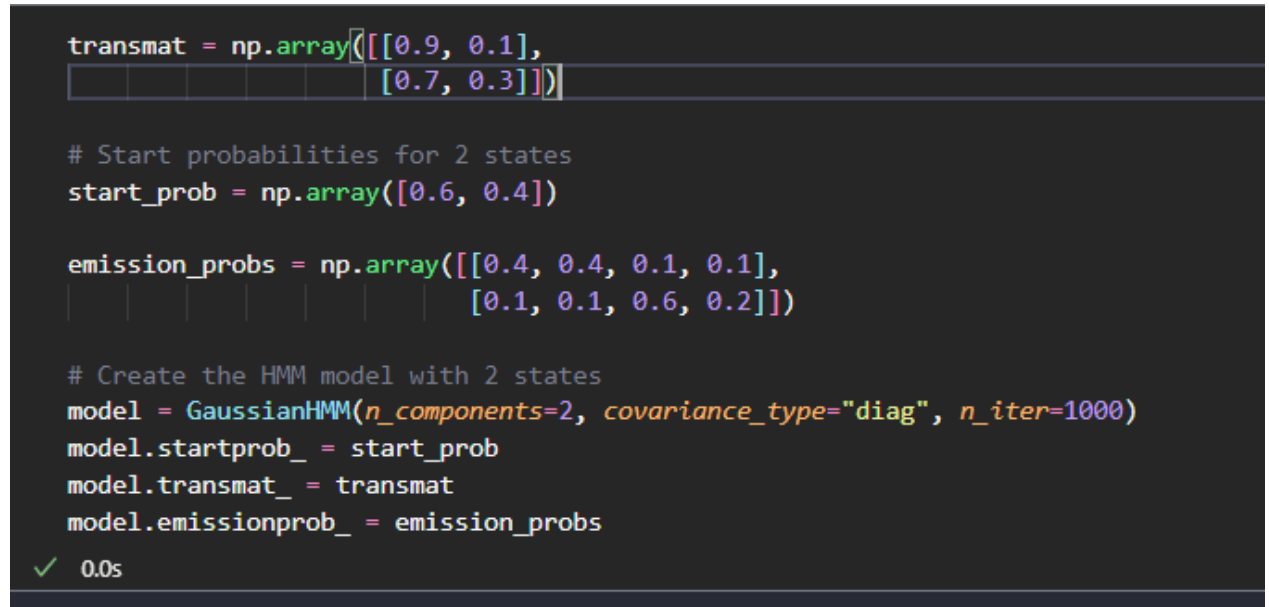
```
# Tạo một biểu đồ đường cho cột giá trị số thực  
plt.figure(figsize=(12, 6))  
  
# Vẽ biểu đồ đường với màu sắc dựa trên giá trị số nguyên  
for i in range(len(data) - 1):  
    plt.plot(data.index[i:i+2], data['3'][i:i+2], color='green' if hidden_states[i] == 0 else 'black')  
  
# Thiết lập nhãn và tiêu đề  
plt.xlabel('Time')  
plt.ylabel('Value')  
plt.title('Time Series with Colored States')  
plt.show()  
✓ 0.2s
```



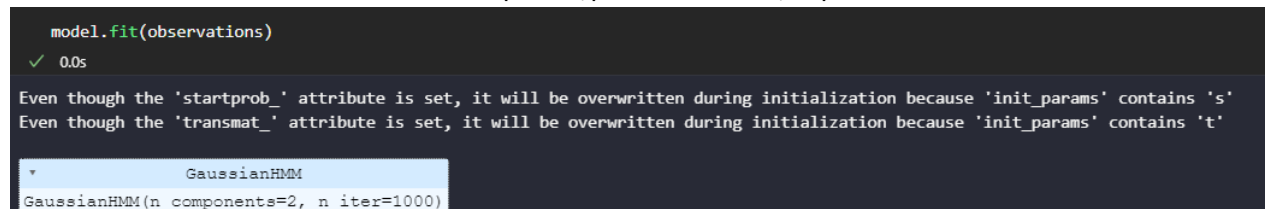
- Vẽ biểu đồ hiển thị trạng thái của dữ liệu trong chuỗi thời gian, trong đó mỗi trạng thái được mã hóa bằng một màu sắc cụ thể.



Sử dụng HMM với 1 số điều chỉnh về transmat, start prob, emission prob



Tối ưu hóa các tham số của nó sao cho phù hợp nhất với dữ liệu quan sát



Dự đoán trạng thái ẩn cho các quan sát được cung cấp bằng cách sử dụng mô hình HMM đã được huấn luyện

✓ 0.0s

✓ 0.0s

### Biểu đồ hiển thị trạng thái ẩn của dữ liệu trong mô hình HMM

✓ 0.4s



tao ra một bộ lọc Kalman với các tham số được chỉ định

```
from pykalman import KalmanFilter
✓ 0.0s

kf = KalmanFilter(initial_state_mean=np.mean(observations, axis=0), n_dim_obs=3)

state_means, state_covariances = kf.em(observations, n_iter=5).filter(observations)
✓ 0.4s
```

Thực hiện ước lượng và lọc dữ liệu bằng cách sử dụng thuật toán EM trong bộ lọc Kalman. Với: observations là dữ liệu quan sát được sử dụng cho việc ước lượng và lọc.

n\_iter=5 là số lần lặp của thuật toán EM.

Sau khi hoàn thành, state\_means chứa ước lượng trung bình của trạng thái tại mỗi thời điểm, và state\_covariances chứa ma trận hiệp phương sai của trạng thái tại mỗi thời điểm.

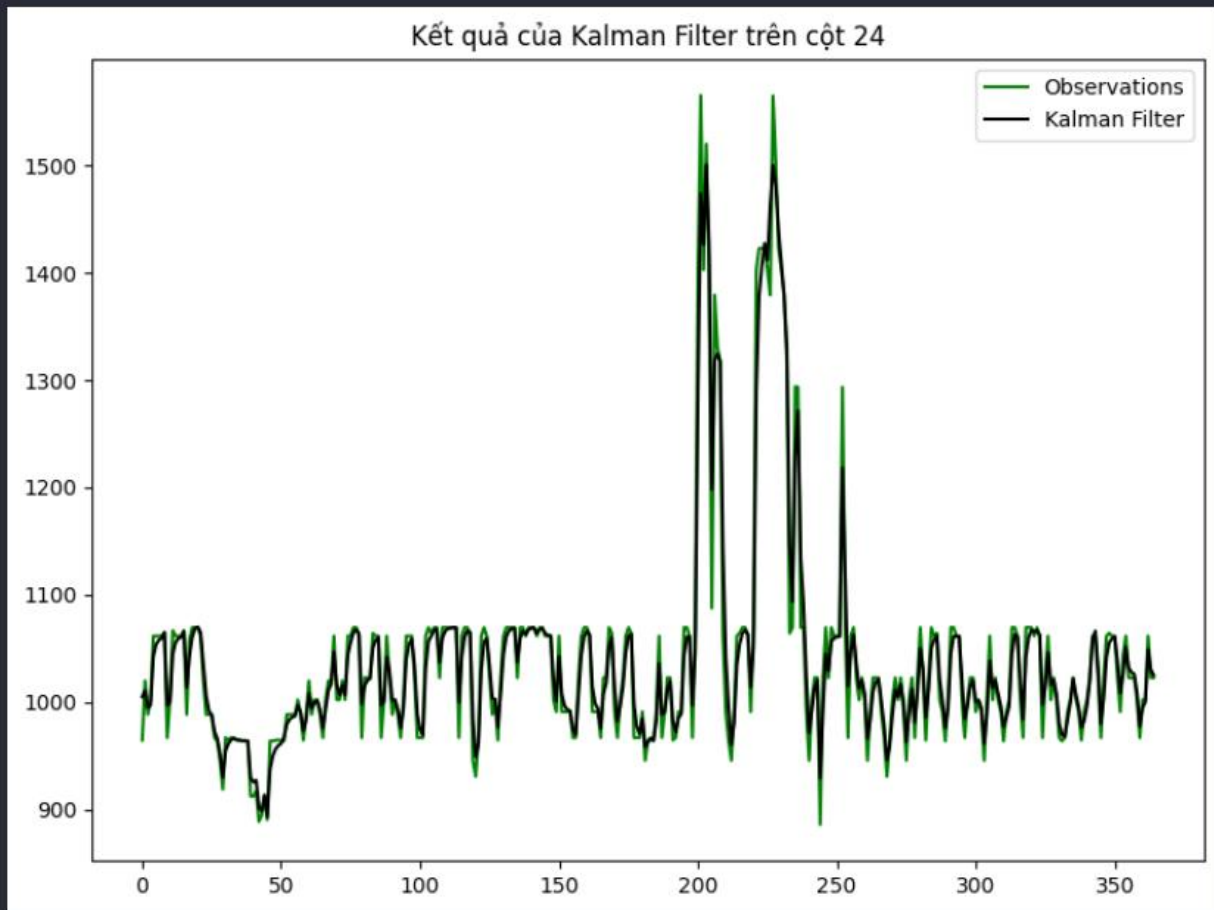
Biểu đồ so sánh giữa dữ liệu quan sát và ước lượng của bộ lọc Kalman trên cột đầu tiên của dữ liệu

```
plt.figure(figsize=(8, 6))

plt.plot(range(observations.shape[0]), observations[:, 0], 'green', label='Observations')
plt.plot(range(observations.shape[0]), state_means[:, 0], 'black', label='Kalman Filter')
plt.title(f'Kết quả của Kalman Filter trên cột 24')
plt.legend()

plt.tight_layout()
plt.show()
```

✓ 0.1s



Với:     màu xanh: dữ liệu quan sát  
           màu đen: dữ liệu sau khi được lọc