```
df=pd.read_csv("price_smp.csv")
data2 = np.loadtxt("price_smp.csv",encoding='latin-1', delimiter=',', skiprows=1, usecols=(8, 9, 10), dtype=float)
df.head()
```

	day	1	2	3	4	5	6	7	8	9	...	39	40	41	42	43	44	45	46	47	48
0	1/01/2021	964.4	964.4	964.4	964.4	964.4	964.4	964.4	964.4	964.4	...	964.4	964.4	964.4	964.4	964.4	964.4	964.4	964.4	964.4	964.4
1	1/02/2021	1019.7	1019.7	1019.7	1019.7	1019.7	1019.7	1019.7	1019.7	1019.7	...	1019.7	1019.7	1019.7	1019.7	1019.7	1019.7	1019.7	1019.7	1019.7	1019.7
2	1/03/2021	988.4	988.4	988.4	988.4	988.4	988.4	988.4	988.4	988.4	...	988.4	988.4	988.4	988.4	988.4	988.4	988.4	988.4	988.4	988.4
3	1/04/2021	1002.0	1002.0	1002.0	1002.0	1002.0	1002.0	1002.0	1002.0	1002.1	...	1010.8	1010.8	1010.8	1010.8	1010.8	1010.8	1010.8	1010.8	1010.8	1010.8
4	1/05/2021	1061.5	1061.5	1061.5	1061.5	1061.5	1061.5	1061.5	1061.5	1061.5	...	1061.5	1061.5	1061.5	1061.5	1061.5	1061.5	1061.5	1061.5	1061.5	1061.5

5 rows × 49 columns

```
feats1= ['8','9','10']
df[feats1].head()
```

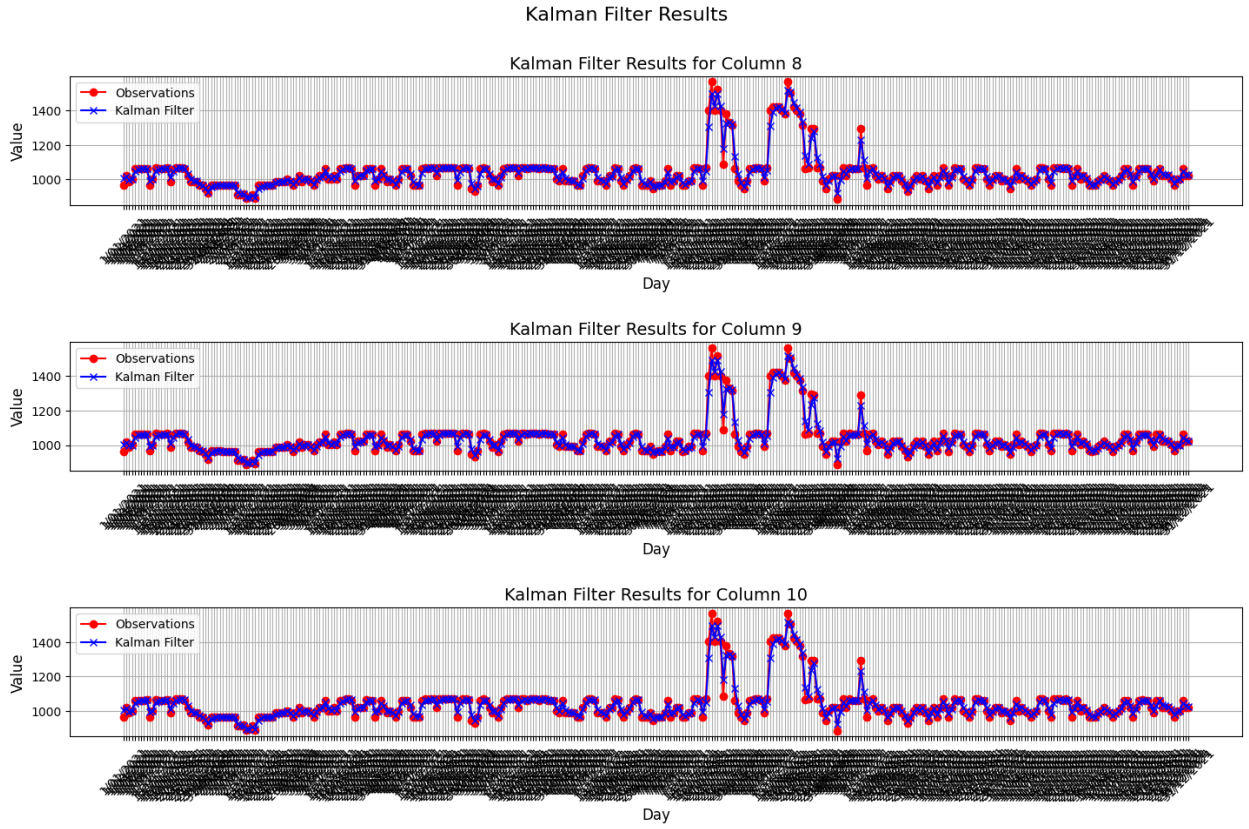
	8	9	10
0	964.4	964.4	964.4
1	1019.7	1019.7	1019.7
2	988.4	988.4	988.4
3	1002.0	1002.1	1002.1
4	1061.5	1061.5	1061.5

Đọc dữ liệu và chọn 3 cột 8 9 10 .

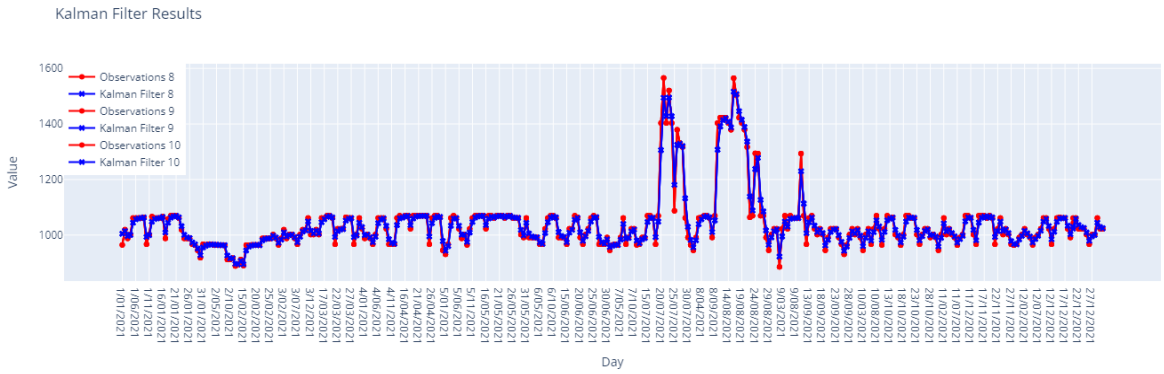
```
# from scipy.signal import KalmanFilter
from filterpy.kalman import KalmanFilter

# Check the head of the DataFrame for the specified columns
# print(df[feats1].head())

# Get data for columns 8, 9, 10
observations = df[feats1].values

# Define the Kalman Filter model
kf = KalmanFilter(initial_state_mean=np.mean(observations, axis=0),
                  n_dim_obs=observations.shape[1])

# Estimate the states
state_means, state_covariances = kf.em(observations).filter(observations)

# Debugging information
print("Shape of observations:", observations.shape)
print("Shape of state_means:", state_means.shape)
```

Khởi tạo mô hình Kalman và in ra giá trị observations và state_means

```python
# Plot the results for each column
plt.figure(figsize=(15, 10))

for i, col in enumerate(feats1):
    plt.subplot(len(feats1), 1, i+1)
    plt.plot(df['day'], observations[:, i], 'r', label='Observations', linestyle='-', marker='o')
    plt.plot(df['day'], state_means[:, i], 'b', label='Kalman Filter', linestyle='-', marker='x')
    plt.title(f'Kalman Filter Results for Column {col}', fontsize=14)
    plt.xlabel('Day', fontsize=12)
    plt.ylabel('Value', fontsize=12)
    plt.legend(loc='upper left', fontsize=10)
    plt.grid(True)
    plt.xticks(rotation=45)
    plt.tight_layout(pad=3.0)

# Show the plot
plt.suptitle('Kalman Filter Results', fontsize=16)
plt.subplots_adjust(top=0.9)
plt.show()
```



Vẽ Biểu đồ kết quả của mô hình Kalman với 3 cột 8 , 9 , 10



Biểu đồ kết quả Kalman giá trị kalman fillter so với giá trị Observation

```
from pykalman import KalmanFilter
# Lấy dữ liệu cột 8, 9, 10
observations = df[feats1].values

# Định nghĩa mô hình Kalman Filter
kf = KalmanFilter(initial_state_mean=np.mean(observations, axis=0),
                  n_dim_obs=3)

# Ước lượng trạng thái
state_means, state_covariances = kf.em(observations).filter(observations)
```

```
print("Shape of state_means:", state_means.shape)
```

```
Shape of state_means: (365, 3)
```

Định nghĩa mô hình KalmanFilter và in ra số giá trị và các cột

```python
# Khởi tạo HMM
model = hmm.GaussianHMM(n_components=5, covariance_type="full", n_iter=1000)

# Huấn luyện HMM với dữ liệu quan sát
model.fit(data2)

# In ra các tham số của mô hình sau khi huấn luyện
print("Start probabilities: ", model.startprob_)
print("Transition matrix: ", model.transmat_)
print("Means: ", model.means_)
print("Covariances: ", model.covars_)
```

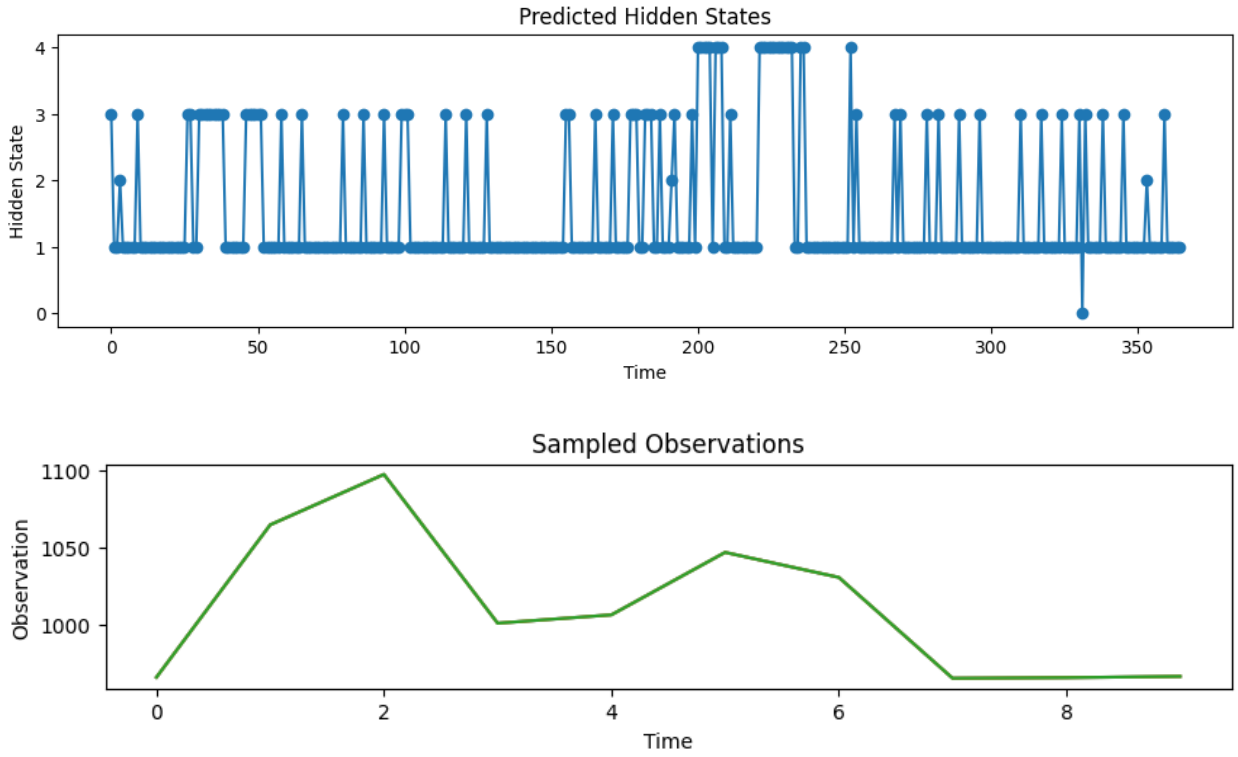
```
Start probabilities:  [0.00000000e+000 9.33326627e-031 1.02195364e-099 1.00000000e+000
 6.45746626e-132]
Transition matrix:  [[0.00000000e+000 1.84785127e-027 3.70098404e-102 1.00000000e+000
  8.97202522e-119]
 [1.04867199e-014 8.64317896e-001 1.05871675e-002 1.07895280e-001
  1.71996569e-002]
 [2.46630885e-076 6.83675248e-001 9.61839595e-075 3.16324752e-001
  5.05201795e-071]
 [1.86914755e-002 6.07913198e-001 2.93162582e-072 3.73395326e-001
  6.80330807e-058]
 [4.72755062e-106 2.10743795e-001 2.48222681e-092 2.26754057e-046
  7.89256205e-001]]
Means:  [[ 963.7         963.7         963.6       ]
 [1025.63258475 1025.63258475 1025.63258475]
 [1003.26618104 1009.33262326 1009.33262326]
 [ 965.63189667  965.63189667  965.63189667]
 [1400.36484472 1400.36484472 1400.36484472]]
Covariances:  [[[1.00000000e-02 1.00000000e-02 1.00000000e-02]
  [1.00000000e-02 1.00000000e-02 1.00000000e-02]
  [1.00000000e-02 1.00000000e-02 1.00000000e-02]]

 [[1.77665266e+03 1.77665266e+03 1.77665266e+03]
  [1.77665266e+03 1.77665266e+03 1.77665266e+03]
  [1.77665266e+03 1.77665266e+03 1.77665266e+03]]

 [[1.03568505e+03 1.27979785e+03 1.27979785e+03]
  [1.27979785e+03 1.59752485e+03 1.59752485e+03]
  [1.27979785e+03 1.59752485e+03 1.59752485e+03]]

 [[1.75297416e+00 1.75297416e+00 1.75297416e+00]
```

Khởi tạo mô hình HMM và in ra các giá trị

```python
# Dự đoán trạng thái ẩn cho các quan sát hiện có
hidden_states = model.predict(data2)
print("Hidden states: ", hidden_states)

# Lấy mẫu mới từ mô hình đã huấn luyện
X, Z = model.sample(10)  # Lấy mẫu 5 quan sát mới
print("Sampled observations: ", X)
print("Sampled hidden states: ", Z)
```

```
Hidden states:  [3 1 1 2 1 1 1 1 1 3 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 3 3 1 1 3 3 3 3 3 3 3
 3 3 1 1 1 1 1 1 1 3 3 3 3 3 3 1 1 1 1 1 1 3 1 1 1 1 1 1 3 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 3 1 1 1 1 1 1 3 1 1 1 1 1 1 3 1 1 1 1 1 3 3 3 1 1 1 1 1 1 1 1 1 1
 1 1 1 3 1 1 1 1 1 1 3 1 1 1 1 1 1 3 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1 1
 1 1 1 1 1 1 1 3 3 1 1 1 1 1 1 1 3 1 1 1 1 1 3 1 1 1 1 1 3 3 3 1 1 3 3 3
 1 1 3 1 1 1 2 3 1 1 1 1 1 3 1 4 4 4 4 4 4 1 4 4 4 1 1 3 1 1 1 1 1 1 1 1 4
 4 4 4 4 4 4 4 4 4 4 4 1 1 4 4 1 1 1 1 1 1 1 1 1 1 1 1 1 1 4 1 3 1 1 1 1
 1 1 1 1 1 1 1 1 3 1 3 1 1 1 1 1 1 1 1 3 1 1 1 3 1 1 1 1 1 1 3 1 1 1 1 1 1
 3 1 1 1 1 1 1 1 1 1 1 1 1 1 3 1 1 1 1 1 1 3 1 1 1 1 1 1 3 1 1 1 1 1 3 0 3
 1 1 1 1 1 3 1 1 1 1 1 1 3 1 1 1 1 1 1 1 2 1 1 1 1 1 3 1 1 1 1 1]
Sampled observations:  [[ 965.89294685  965.89294684  965.89294684]
 [1064.63019489 1064.63019443 1064.63019443]
 [1097.30453106 1097.30453077 1097.30453077]
 [1000.91056082 1000.91056053 1000.91056053]
 [1006.31468588 1006.31468575 1006.31468575]
 [1046.79155095 1046.79155123 1046.79155123]
 [1030.58617716 1030.58617729 1030.58617729]
 [ 965.26577106  965.26577106  965.26577106]
 [ 965.65175984  965.65175986  965.65175986]
 [ 966.45520481  966.45520481  966.45520481]]
Sampled hidden states:  [3 1 1 1 1 1 1 3 3 3]
```

Dự đoán trạng thái ẩn cho quan sát và lấy mẫu mới từ mô hình



Biểu đồ thể hiện các trạng thái ẩn dự đoán và các quan sát mẫu

```
# Khởi tạo HMM
# model = hmm.MultinomialHMM(n_components=5, covariance_type="full", n_iter=1000)
model = hmm.MultinomialHMM(n_components=2, n_trials=1)
# Huấn luyện HMM với dữ liệu quan sát
model.fit(df[feats1])

# In ra các tham số của mô hình sau khi huấn luyện
print("Start probabilities: ", model.startprob_)
print("Transition matrix: ", model.transmat_)
print("Means: ", model.means_)
print("Covariances: ", model.covars_)
```

Khởi tạo một model HMM với biến thể khác Multinomial