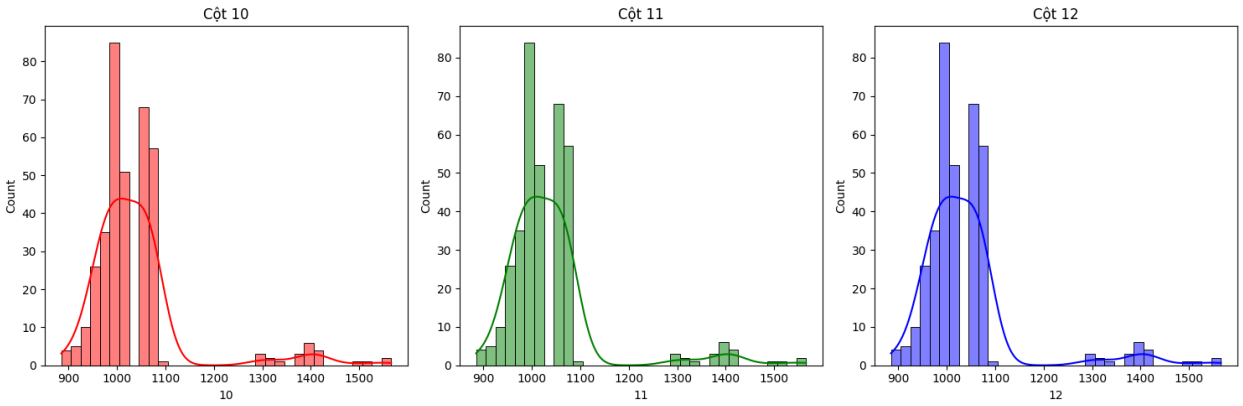
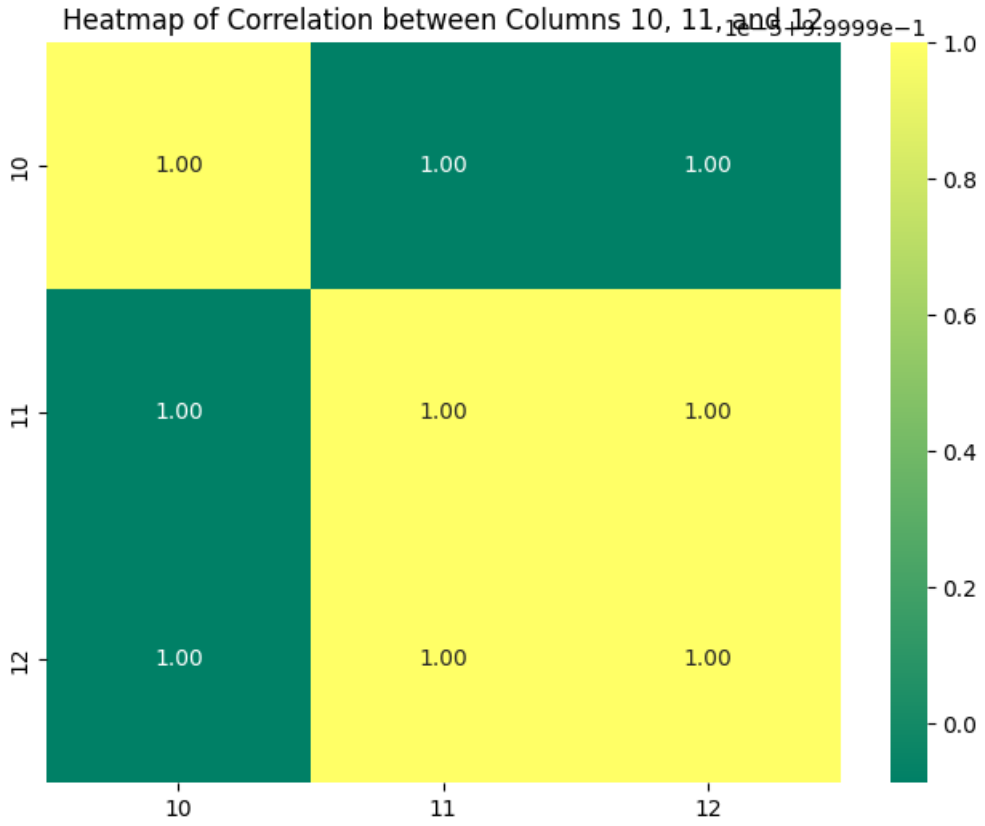
EDA

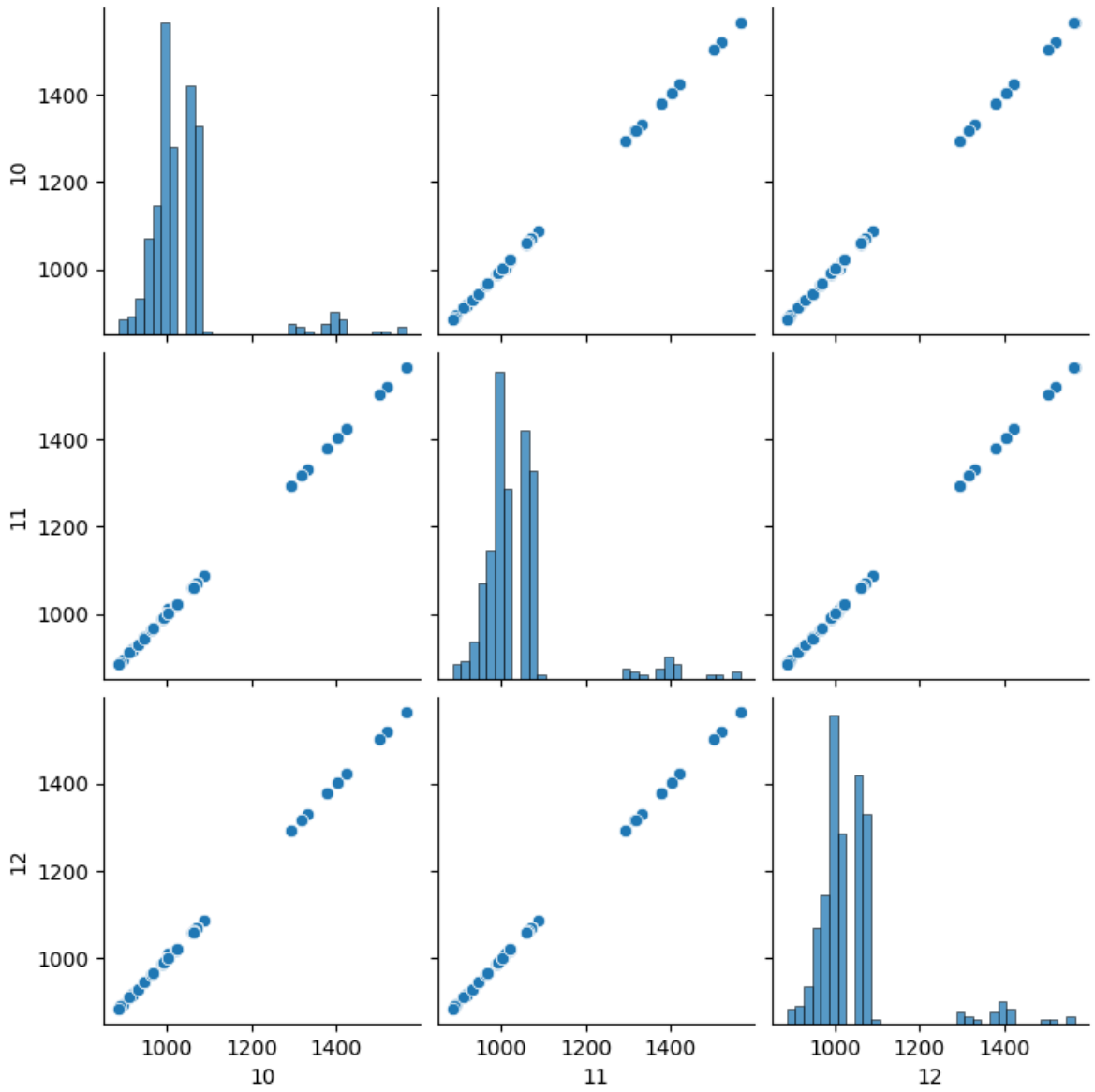
```python
# Trực quan hóa phân phối của cột 10,11,12
plt.figure(figsize=(15, 5))

plt.subplot(1, 3, 1)
sns.histplot(data=df, x=df.columns[10], kde=True, color='red')
plt.title('Cột ' + df.columns[10])

plt.subplot(1, 3, 2)
sns.histplot(data=df, x=df.columns[11], kde=True, color='green')
plt.title('Cột ' + df.columns[11])

plt.subplot(1, 3, 3)
sns.histplot(data=df, x=df.columns[12], kde=True, color='blue')
plt.title('Cột ' + df.columns[12])

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




```python
# Trực quan hóa mối quan hệ giữa các cột

columns_of_interest = ['10', '11', '12']
X = df[columns_of_interest].values


plt.figure(figsize=(15, 5))

#Biểu đồ heatmap
correlation_matrix = df[columns_of_interest].corr()
plt.figure(figsize=(8, 6))
sns.heatmap(correlation_matrix, annot=True, cmap='summer', fmt='.2f')
plt.title('Heatmap of Correlation between Columns 10, 11, and 12')
plt.show()
#Biểu đồ scatter
sns.pairplot(df.iloc[:, [10, 11, 12]])
plt.show()
```

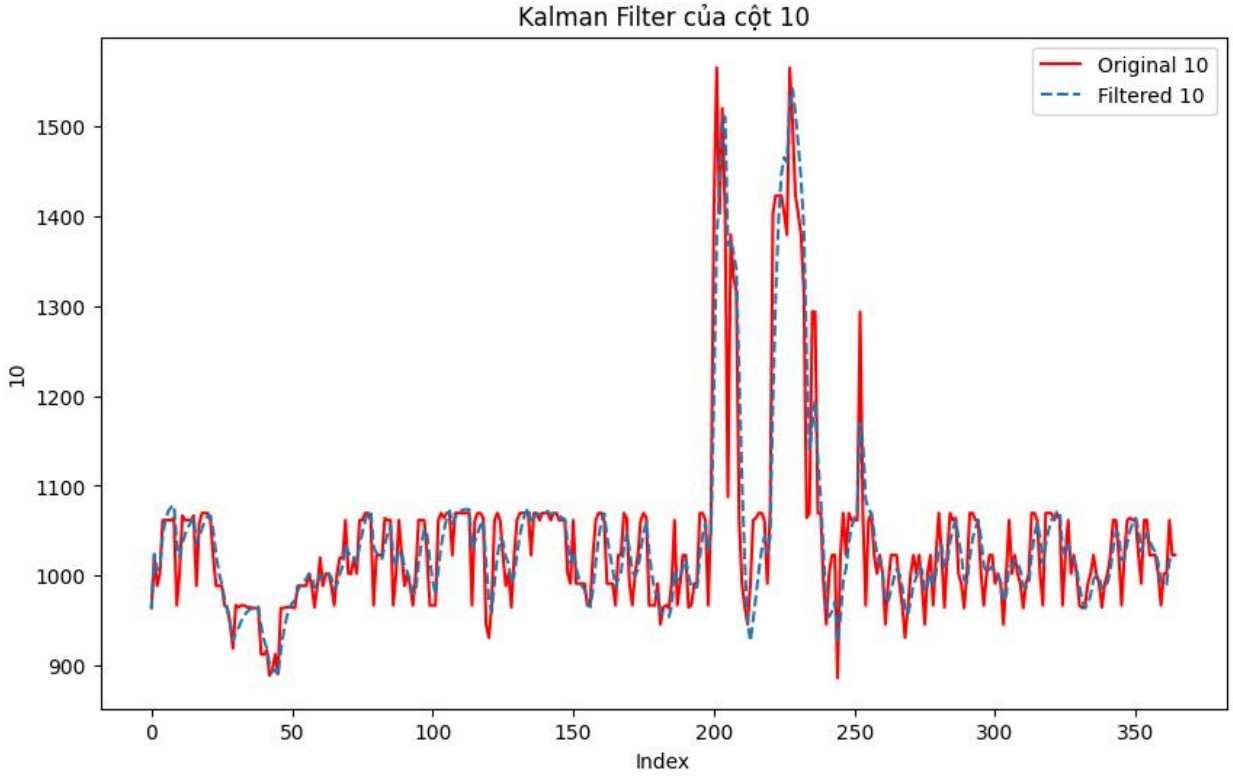
✓ 2.6s Python


Heatmap of Correlation between Columns 10, 11, and 12
1e−5+9.9999e−1
1.0
0.8
0.6
0.4
0.2
0.0
10
11
12
1.00
1.00
1.00
1.00
1.00
1.00
1.00
1.00
1.00
10
11
12



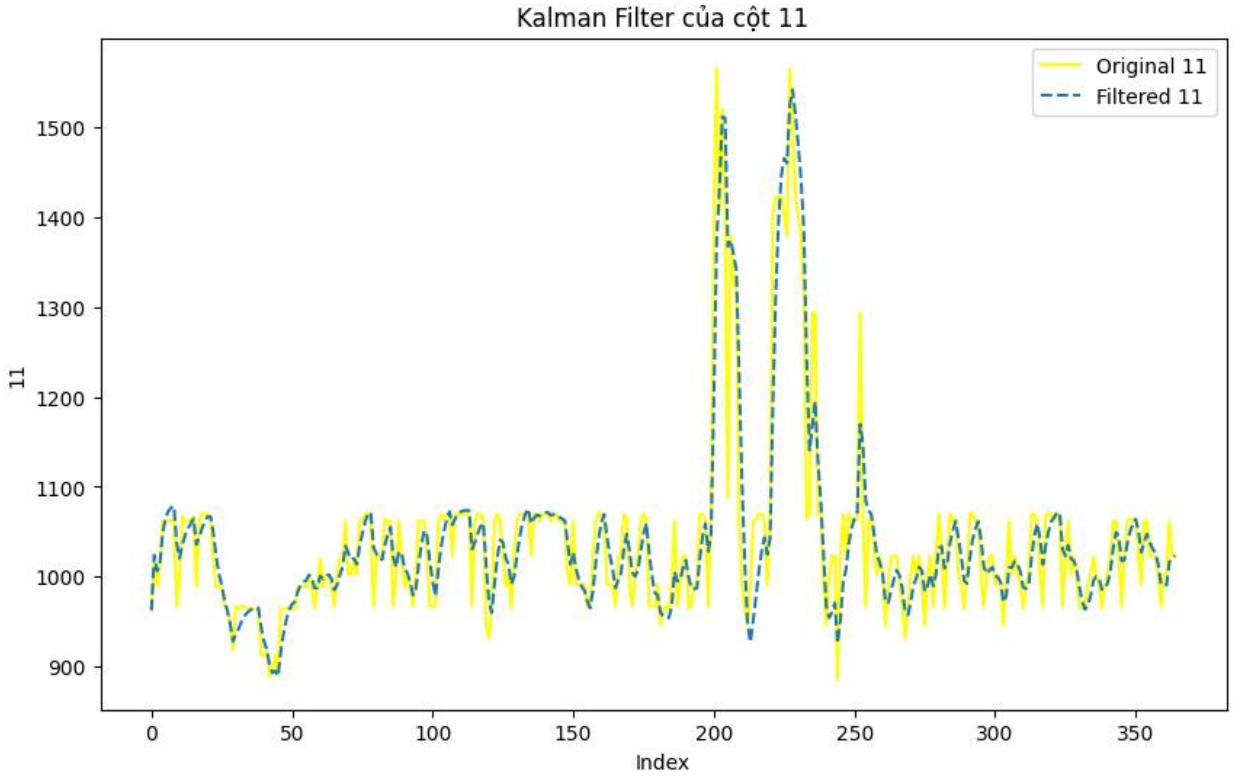
KALMA

```python
from filterpy.kalman import KalmanFilter
import numpy as np

def apply_kalman_filter(data):
    kf = KalmanFilter(dim_x=2, dim_z=1)
    kf.x = np.array([0., 0.])
    kf.F = np.array([[1., 1.], [0., 1.]])
    kf.H = np.array([[1., 0.]])
    kf.P *= 1000.
    kf.R = 5
    kf.Q = np.array([[0.1, 0.1], [0.1, 0.1]])

    filtered_data = []
    for z in data:
        kf.predict()
        kf.update(z)
        filtered_data.append(kf.x[0])
    return filtered_data

# Áp dụng Kalman Filter cho cột thứ 10 (Feature_10)
filtered_feature_10 = apply_kalman_filter(X[:, 0])

# So sánh dữ liệu gốc và dữ liệu đã lọc
plt.figure(figsize=(10, 6))
plt.plot(X[:, 0], label='Original ' + df.columns[10],color = 'red')
plt.plot(filtered_feature_10, label='Filtered ' + df.columns[10], linestyle='dashed')
plt.title('Kalman Filter của cột ' + df.columns[10])
plt.xlabel('Index')
plt.ylabel(df.columns[10])
plt.legend()
plt.show()
```



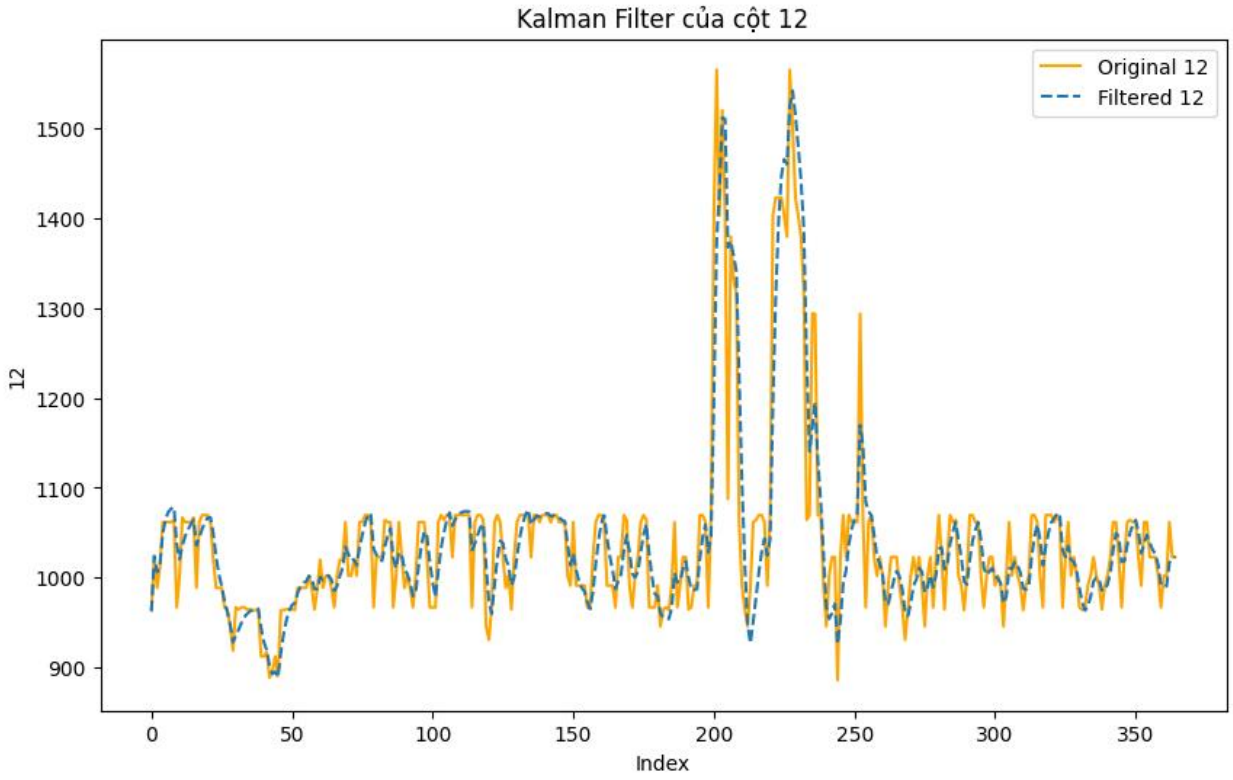
```python
def apply_kalman_filter(data):
    kf = KalmanFilter(dim_x=2, dim_z=1)
    kf.x = np.array([0., 0.])
    kf.F = np.array([[1., 1.], [0., 1.]])
    kf.H = np.array([[1., 0.]])
    kf.P *= 1000.
    kf.R = 5
    kf.Q = np.array([[0.1, 0.1], [0.1, 0.1]])

    filtered_data = []
    for z in data:
        kf.predict()
        kf.update(z)
        filtered_data.append(kf.x[0])
    return filtered_data
filtered_feature_11 = apply_kalman_filter(X[:, 2])
plt.figure(figsize=(10, 6))
plt.plot(X[:, 1], label='Original ' + df.columns[11],color ='yellow')
plt.plot(filtered_feature_11, label='Filtered ' + df.columns[11], linestyle='dashed')
plt.title('Kalman Filter của cột ' + df.columns[11])
plt.xlabel('Index')
plt.ylabel(df.columns[11])
plt.legend()
plt.show()
```

✓ 0.3s Python



```python
def apply_kalman_filter(data):
    kf = KalmanFilter(dim_x=2, dim_z=1)
    kf.x = np.array([0., 0.])
    kf.F = np.array([[1., 1.], [0., 1.]])
    kf.H = np.array([[1., 0.]])
    kf.P *= 1000.
    kf.R = 5
    kf.Q = np.array([[0.1, 0.1], [0.1, 0.1]])

    filtered_data = []
    for z in data:
        kf.predict()
        kf.update(z)
        filtered_data.append(kf.x[0])
    return filtered_data

filtered_feature_12 = apply_kalman_filter(X[:, 2])

plt.figure(figsize=(10, 6))
plt.plot(X[:, 1], label='Original ' + df.columns[12],color='orange')
plt.plot(filtered_feature_12, label='Filtered ' + df.columns[12], linestyle='dashed')
plt.title('Kalman Filter của cột ' + df.columns[12])
plt.xlabel('Index')
plt.ylabel(df.columns[12])
plt.legend()
plt.show()
```

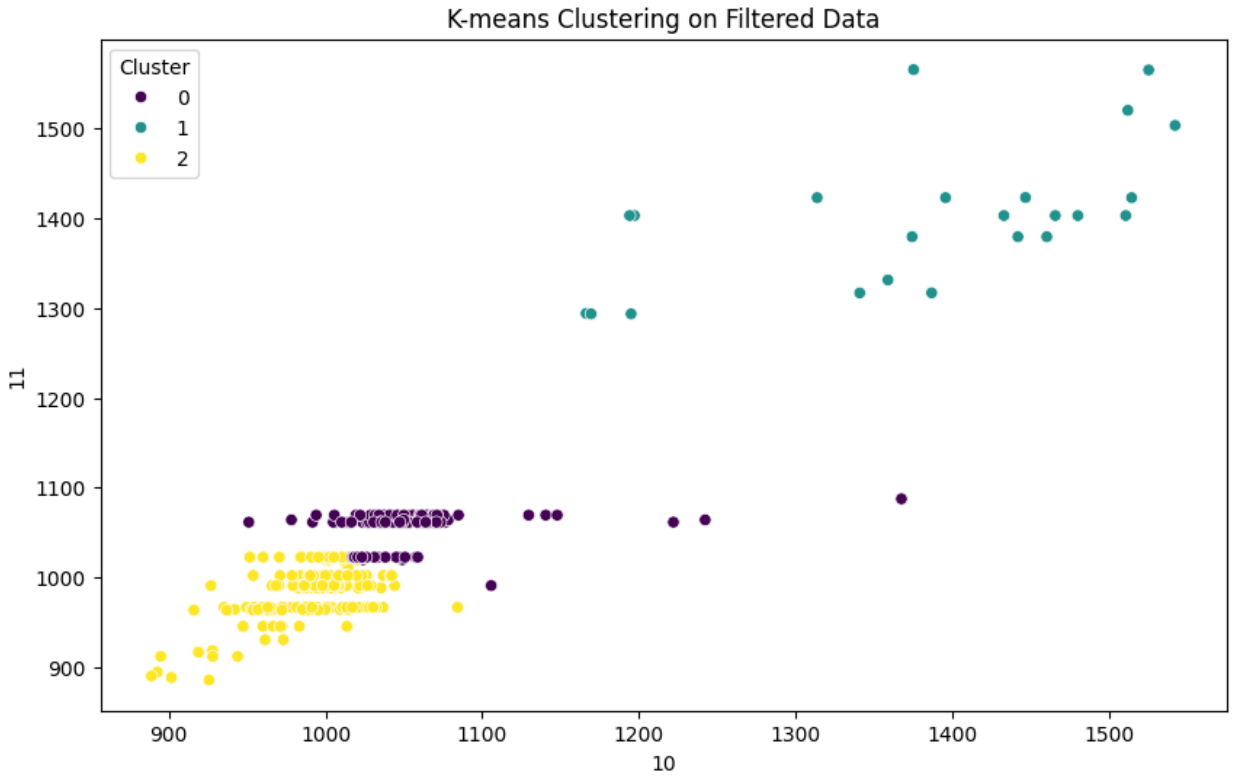
✓ 0.3s Python



KMEANS

```python
from sklearn.cluster import KMeans

# Tạo một DataFrame mới với cột 10, cột 11 đã được lọc, và cột 12
df_filtered = pd.DataFrame({
    df.columns[10]: X[:, 0],
    df.columns[11]: filtered_feature_11,
    df.columns[12]: X[:, 2]
})

# Sử dụng k-means clustering với 3 cụm
kmeans = KMeans(n_clusters=3, random_state=42)
df_filtered['Cluster'] = kmeans.fit_predict(df_filtered)

# Trực quan hóa kết quả cụm
plt.figure(figsize=(10, 6))
sns.scatterplot(x=df_filtered[df.columns[11]], y=df_filtered[df.columns[12]], hue=df_filtered['Cluster'], palette='viridis')
plt.title('K-means Clustering on Filtered Data')
plt.xlabel(df.columns[10])
plt.ylabel(df.columns[11])
plt.show()
```

0.3s Python



Link gihub: [trinhdat24/ThucHanh3_TimeSeries (github.com)]