In [1]:

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, r2_score
from scipy import stats
import statsmodels.api as sm
from statsmodels.stats.stattools import durbin_watson
```

JAPAN

In [2]:

```
dfjap = pd.read_csv('Japan_Dataset.csv')
```

In [3]:

```
dfjap
```

Out[3]:

| | Date | Unemployment | Inflation | GDP |
|---|---|---|---|---|
| **0** | 1961-01-01 | 1.43 | 0.43 | 10.88 |
| **1** | 1961-04-01 | 1.47 | 0.43 | 13.03 |
| **2** | 1961-07-01 | 1.43 | 0.93 | 11.38 |
| **3** | 1961-10-01 | 1.43 | 1.07 | 12.18 |
| **4** | 1962-01-01 | 1.30 | 0.13 | 10.33 |
| **...** | ... | ... | ... | ... |
| **231** | 2018-10-01 | 2.47 | 0.07 | -0.25 |
| **232** | 2019-01-01 | 2.47 | -0.04 | -0.12 |
| **233** | 2019-04-01 | 2.33 | 0.02 | -0.10 |
| **234** | 2019-07-01 | 2.33 | 0.06 | 0.64 |
| **235** | 2019-10-01 | 2.30 | 0.19 | -2.00 |

236 rows × 4 columns

In [4]:

```
dfjap.shape
```

Out[4]:

```
(236, 4)
```

In [5]:

```
dfjap.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 236 entries, 0 to 235
Data columns (total 4 columns):
 #   Column        Non-Null Count  Dtype  
---  ------        --------------  -----  
 0   Date          236 non-null    object 
 1   Unemployment  236 non-null    float64
 2   Inflation     236 non-null    float64
 3   GDP           236 non-null    float64
dtypes: float64(3), object(1)
memory usage: 7.5+ KB
```

In [6]:

```
dfjap['Date']=pd.to_datetime(dfjap['Date'], errors='coerce')
dfjap['Unemployment']=pd.to_numeric(dfjap['Unemployment'], errors='coerce')
dfjap['Inflation']=pd.to_numeric(dfjap['Inflation'], errors='coerce')
```

In [7]:

```
dfjap.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 236 entries, 0 to 235
Data columns (total 4 columns):
 #   Column        Non-Null Count  Dtype         
---  ------        --------------  -----         
 0   Date          236 non-null    datetime64[ns]
 1   Unemployment  236 non-null    float64       
 2   Inflation     236 non-null    float64       
 3   GDP           236 non-null    float64       
dtypes: datetime64[ns](1), float64(3)
memory usage: 7.5 KB
```

In [8]:

```
dfjap.isnull().sum()
```

Out[8]:

```
Date            0
Unemployment    0
Inflation       0
GDP             0
dtype: int64
```

In [9]:

```
dfjap.describe()
```

Out[9]:

|  | Date | Unemployment | Inflation | GDP |
|---|---|---|---|---|
| **count** | 236 | 236.000000 | 236.000000 | 236.000000 |
| **mean** | 1990-05-17 01:25:25.423728768 | 2.805466 | 0.245720 | 3.735169 |
| **min** | 1961-01-01 00:00:00 | 1.070000 | -0.270000 | -8.850000 |
| **25%** | 1975-09-08 00:00:00 | 1.900000 | -0.010000 | 0.980000 |
| **50%** | 1990-05-16 12:00:00 | 2.550000 | 0.120000 | 2.900000 |
| **75%** | 2005-01-23 12:00:00 | 3.877500 | 0.422500 | 5.755000 |
| **max** | 2019-10-01 00:00:00 | 5.430000 | 2.320000 | 14.450000 |
| **std** | NaN | 1.278722 | 0.378014 | 4.009092 |

In [10]:

```
dfjap[['Unemployment', 'Inflation', 'GDP']].corr()
```

Out[10]:

|  | Unemployment | Inflation | GDP |
|---|---|---|---|
| **Unemployment** | 1.000000 | -0.618499 | -0.672763 |
| **Inflation** | -0.618499 | 1.000000 | 0.368458 |
| **GDP** | -0.672763 | 0.368458 | 1.000000 |

In [11]:

```
dfjap[['Unemployment', 'Inflation', 'GDP']].var()
```

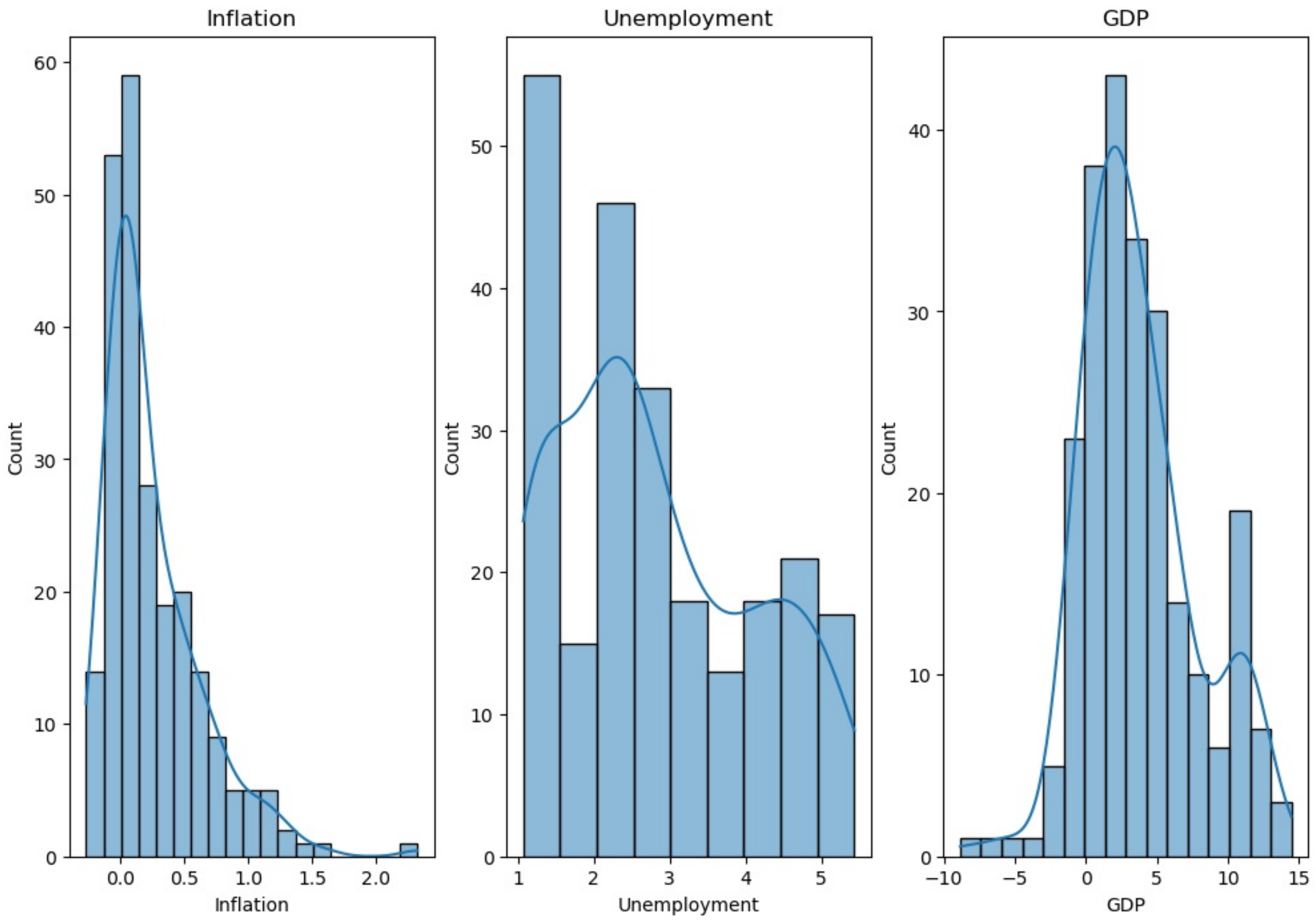
Out[11]:

```
Unemployment     1.635130
Inflation        0.142894
GDP             16.072815
dtype: float64
```

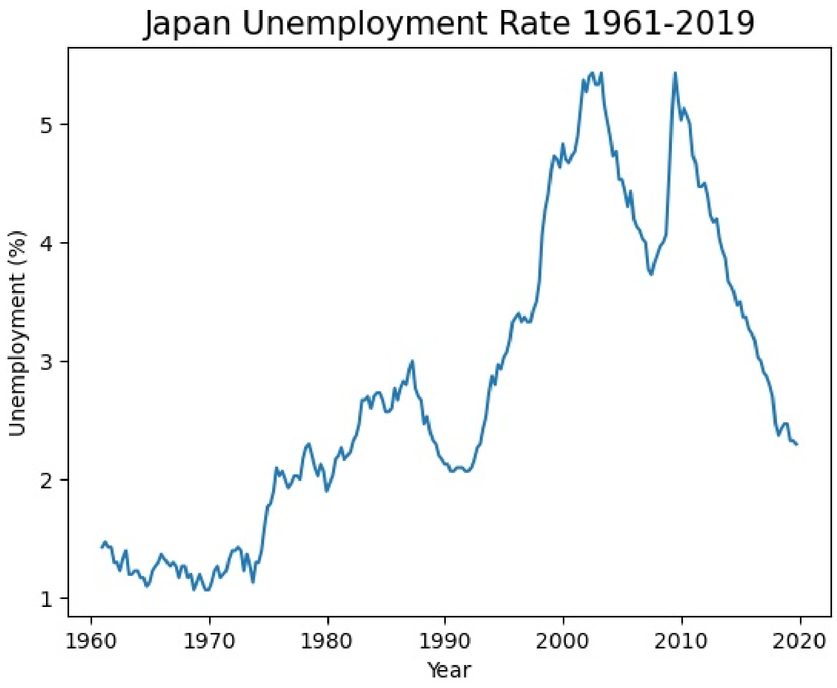
-> Exploratory Analysis

In [12]:

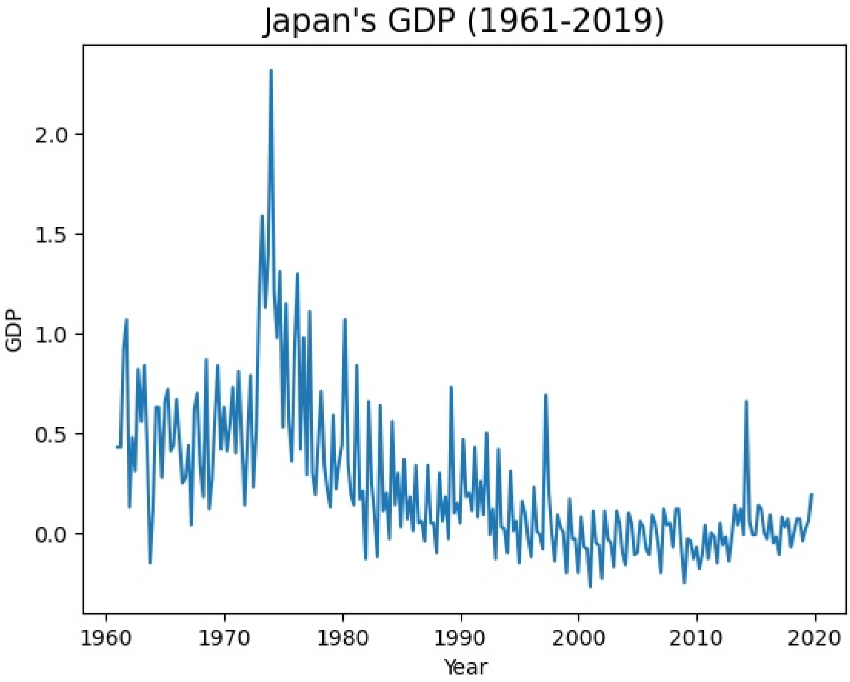
```
plt.figure(figsize=(12, 8))
plt.subplot(1,3,1)
sns.histplot(dfjap["Inflation"], kde=True)
plt.title("Inflation")
plt.subplot(1,3,2)
sns.histplot(dfjap["Unemployment"], kde=True)
plt.title("Unemployment")
plt.subplot(1,3,3)
sns.histplot(dfjap['GDP'], kde=True)
plt.title("GDP")
plt.show()
```



In [13]:
```
plt.plot( dfjap['Date'], dfjap['Unemployment'])
plt.xlabel("Year", fontsize =10)
plt.ylabel("Unemployment (%)", fontsize =10)
plt.title("Japan Unemployment Rate 1961-2019", fontsize=15);
```

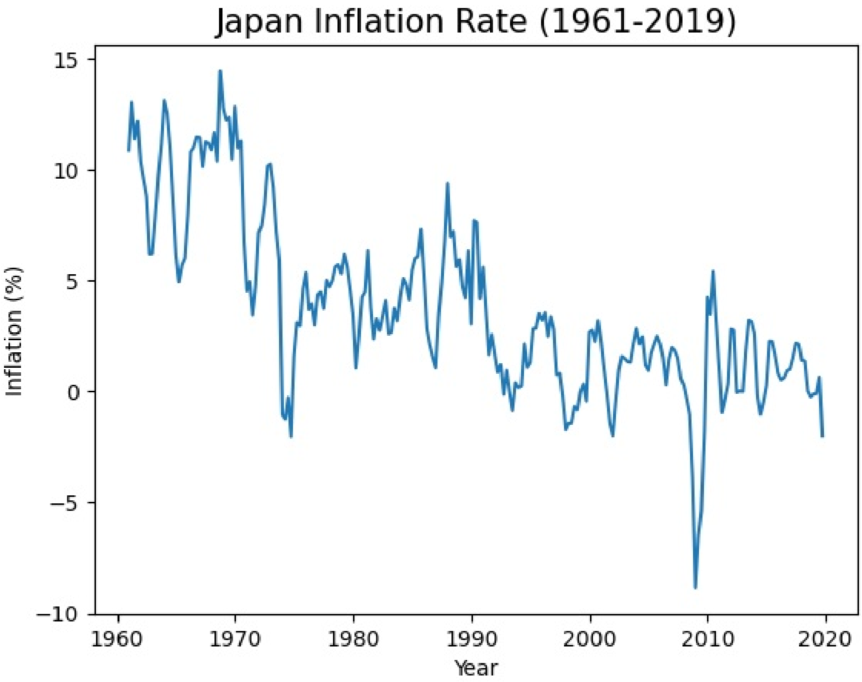


In [14]:
```
plt.plot(dfjap['Date'], dfjap['Inflation'])
plt.xlabel("Year", fontsize =10)
plt.ylabel("GDP", fontsize =10)
plt.title("Japan's GDP (1961-2019)", fontsize=15);
```



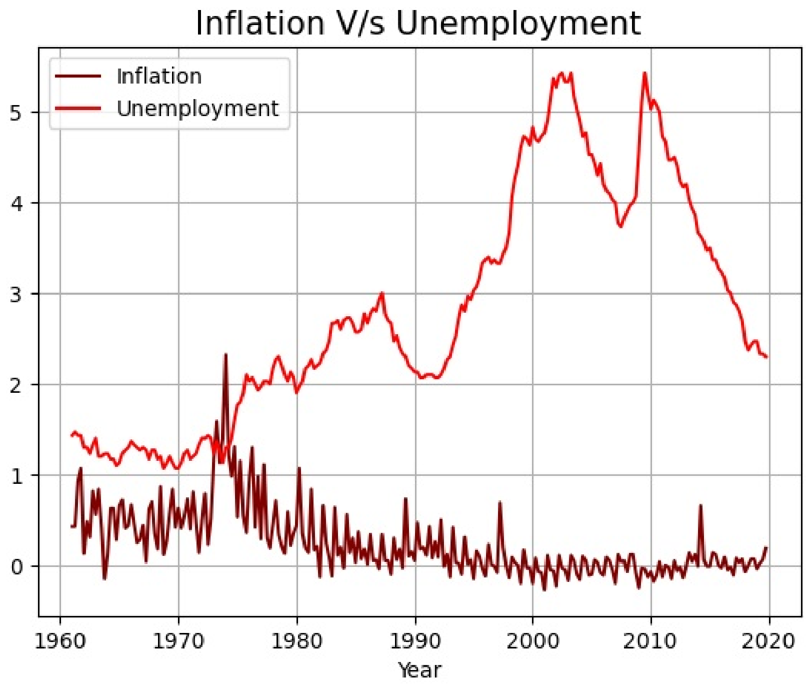
In [15]:

```python
plt.plot(dfjap['Date'], dfjap['GDP'])
plt.xlabel("Year", fontsize =10)
plt.ylabel("Inflation (%)", fontsize =10)
plt.title("Japan Inflation Rate (1961-2019)", fontsize=15);
```

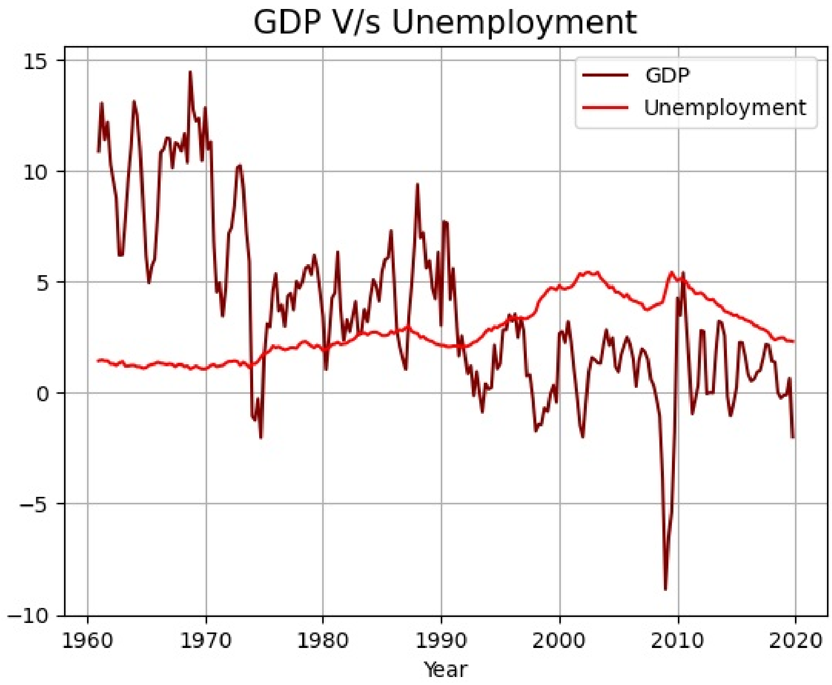


In [16]:

```python
plt.plot(dfjap['Date'], dfjap['Inflation'], color='maroon', label='Inflation')
plt.plot(dfjap['Date'], dfjap['Unemployment'], color='r', label='Unemployment')
plt.xlabel('Year', fontsize=10)
plt.title('Inflation V/s Unemployment', fontsize=15)
plt.grid()
plt.legend();
```



```
In [17]: plt.plot(dfjap['Date'], dfjap['GDP'], color='maroon', label='GDP')
         plt.plot(dfjap['Date'], dfjap['Unemployment'], color='r', label='Unemployment')
         plt.xlabel('Year', fontsize=10)
         plt.title('GDP V/s Unemployment', fontsize=15)
         plt.grid()
         plt.legend();
```



Linear Regression Model

```
In [18]: a = dfjap['Unemployment']
         a = a.values.reshape(-1,1)
```

```
In [19]: b = dfjap['Inflation']
         b = b.values.reshape(-1,1)
```

```
In [20]: a_train, a_test, b_train, b_test = train_test_split(a, b, test_size=0.25, random_state=42)
```

```
In [21]: model = LinearRegression()
```

```
In [22]: model1 = model.fit(a_train,b_train)
         model1
```

```
Out[22]: ▾ LinearRegression
         LinearRegression()
```

```
In [23]: model1.coef_
```

```
Out[23]: array([[-0.18606511]])
```

```
In [24]: model1.intercept_
```

```
Out[24]: array([0.77490772])
```

```
In [25]: b_pred = model.predict(a_test)
```

```
In [26]: b_pred
```

```
Out[26]: array([[ 0.35253992],
                [ 0.03064727],
                [ 0.00645881],
                [ 0.55162959],
                [ 0.35253992],
                [ 0.3153269 ],
                [ 0.33393341],
                [ 0.26694997],
                [-0.01959031],
                [ 0.5702361 ],
                [ 0.25392541],
                [ 0.47720354],
                [-0.07540984],
                [ 0.55162959],
                [ 0.21113043],
                [ 0.09204876],
                [ 0.53302308],
                [ 0.34137601],
                [ 0.27253192],
                [ 0.53302308],
                [ 0.24090085],
                [ 0.55162959],
                [ 0.27811387],
                [-0.12378677],
                [-0.06238528],
                [ 0.53860503],
                [ 0.33393341],
                [ 0.03064727],
                [ 0.35998252],
                [ 0.53860503],
                [ 0.12367983],
                [ 0.39719555],
                [ 0.51441657],
                [ 0.14786829],
                [ 0.38417099],
                [-0.16099979],
                [-0.22984389],
                [ 0.56465415],
                [ 0.05483574],
                [-0.02517226],
                [-0.17960631],
                [-0.13681133],
                [ 0.38975294],
                [ 0.39719555],
                [-0.10518026],
                [ 0.3153269 ],
                [ 0.37114643],
                [ 0.37858903],
                [ 0.13670439],
                [ 0.12926178],
                [-0.10518026],
                [ 0.34137601],
                [ 0.01204076],
                [ 0.17391741],
                [ 0.27811387],
                [ 0.29672038],
                [-0.1684424 ],
                [ 0.39719555],
                [ 0.54604764]])
```

```
In [27]: r2score1 = r2_score(b_test, b_pred)
         r2score1
```

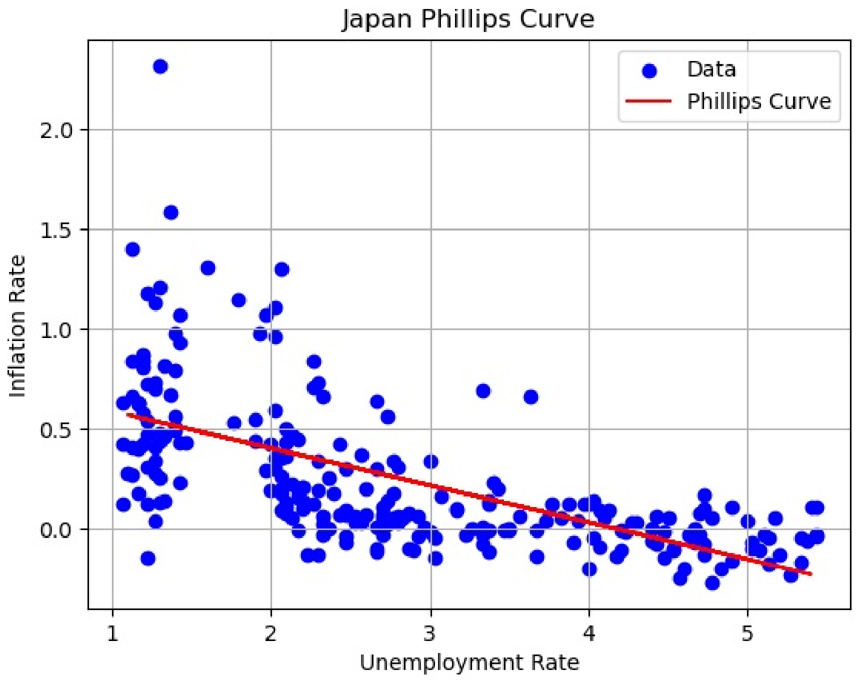
```
Out[27]: 0.4178141242316723
```

```
In [28]: mse1 = mean_squared_error(b_test,b_pred)
         mse1
```

```
Out[28]: 0.062040810951619965
```

Phillips Curve

```
In [29]: plt.scatter(a, b, color='blue', label='Data')
         plt.plot(a_test, b_pred, color='red', label='Phillips Curve')
         plt.xlabel('Unemployment Rate')
         plt.ylabel('Inflation Rate')
         plt.title('Japan Phillips Curve')
         plt.legend()
         plt.grid()
         plt.show()
```



-> USA

```
In [30]: dfusa = pd.read_csv('USA_Dataset.csv')
```

```
In [31]: dfusa
```

Out[31]:

| | Date | Unemployment | Inflation | GDP |
|---|---|---|---|---|
| **0** | 1961-01-01 | 6.80 | 1.50 | 0.44 |
| **1** | 1961-04-01 | 7.00 | 0.90 | 2.67 |
| **2** | 1961-07-01 | 6.77 | 1.27 | 4.04 |
| **3** | 1961-10-01 | 6.20 | 0.70 | 7.48 |
| **4** | 1962-01-01 | 5.63 | 0.90 | 8.99 |
| **...** | ... | ... | ... | ... |
| **231** | 2018-10-01 | 3.83 | 2.20 | 4.91 |
| **232** | 2019-01-01 | 3.87 | 1.67 | 4.64 |
| **233** | 2019-04-01 | 3.63 | 1.80 | 4.05 |
| **234** | 2019-07-01 | 3.60 | 1.73 | 3.82 |
| **235** | 2019-10-01 | 3.60 | 2.07 | 3.98 |

236 rows × 4 columns

```
In [32]: dfusa.shape
```

```
Out[32]: (236, 4)
```

```
In [33]: dfusa.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 236 entries, 0 to 235
Data columns (total 4 columns):
 #   Column        Non-Null Count  Dtype  
---  ------        --------------  -----  
 0   Date          236 non-null    object 
 1   Unemployment  236 non-null    float64
 2   Inflation     236 non-null    float64
 3   GDP           236 non-null    float64
dtypes: float64(3), object(1)
memory usage: 7.5+ KB
```

In [34]:

```
dfusa['Date']=pd.to_datetime(dfusa['Date'], errors='coerce')
dfusa['Unemployment']=pd.to_numeric(dfusa['Unemployment'], errors='coerce')
dfusa['Inflation']=pd.to_numeric(dfusa['Inflation'], errors='coerce')
```

In [35]:

```
dfusa.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 236 entries, 0 to 235
Data columns (total 4 columns):
 #   Column        Non-Null Count  Dtype         
---  ------        --------------  -----         
 0   Date          236 non-null    datetime64[ns]
 1   Unemployment  236 non-null    float64       
 2   Inflation     236 non-null    float64       
 3   GDP           236 non-null    float64       
dtypes: datetime64[ns](1), float64(3)
memory usage: 7.5 KB
```

In [36]:

```
dfusa.isnull().sum()
```

Out[36]:

```
Date            0
Unemployment    0
Inflation       0
GDP             0
dtype: int64
```

In [37]:

```
dfusa.describe()
```

Out[37]:

| | Date | Unemployment | Inflation | GDP |
|---|---|---|---|---|
| **count** | 236 | 236.000000 | 236.000000 | 236.000000 |
| **mean** | 1990-05-17 01:25:25.423728768 | 5.972627 | 3.762373 | 6.471144 |
| **min** | 1961-01-01 00:00:00 | 3.400000 | -1.630000 | -3.060000 |
| **25%** | 1975-09-08 00:00:00 | 4.822500 | 1.852500 | 4.585000 |
| **50%** | 1990-05-16 12:00:00 | 5.700000 | 3.030000 | 6.110000 |
| **75%** | 2005-01-23 12:00:00 | 7.040000 | 4.547500 | 8.167500 |
| **max** | 2019-10-01 00:00:00 | 10.670000 | 14.500000 | 14.700000 |
| **std** | NaN | 1.618211 | 2.833593 | 2.992605 |

In [38]:

```
dfusa[['Unemployment', 'Inflation', 'GDP']].corr()
```

Out[38]:

| | Unemployment | Inflation | GDP |
|---|---|---|---|
| **Unemployment** | 1.000000 | 0.140027 | -0.102878 |
| **Inflation** | 0.140027 | 1.000000 | 0.587212 |
| **GDP** | -0.102878 | 0.587212 | 1.000000 |

In [39]:

```
dfusa[['Unemployment', 'Inflation', 'GDP']].var()
```

Out[39]:

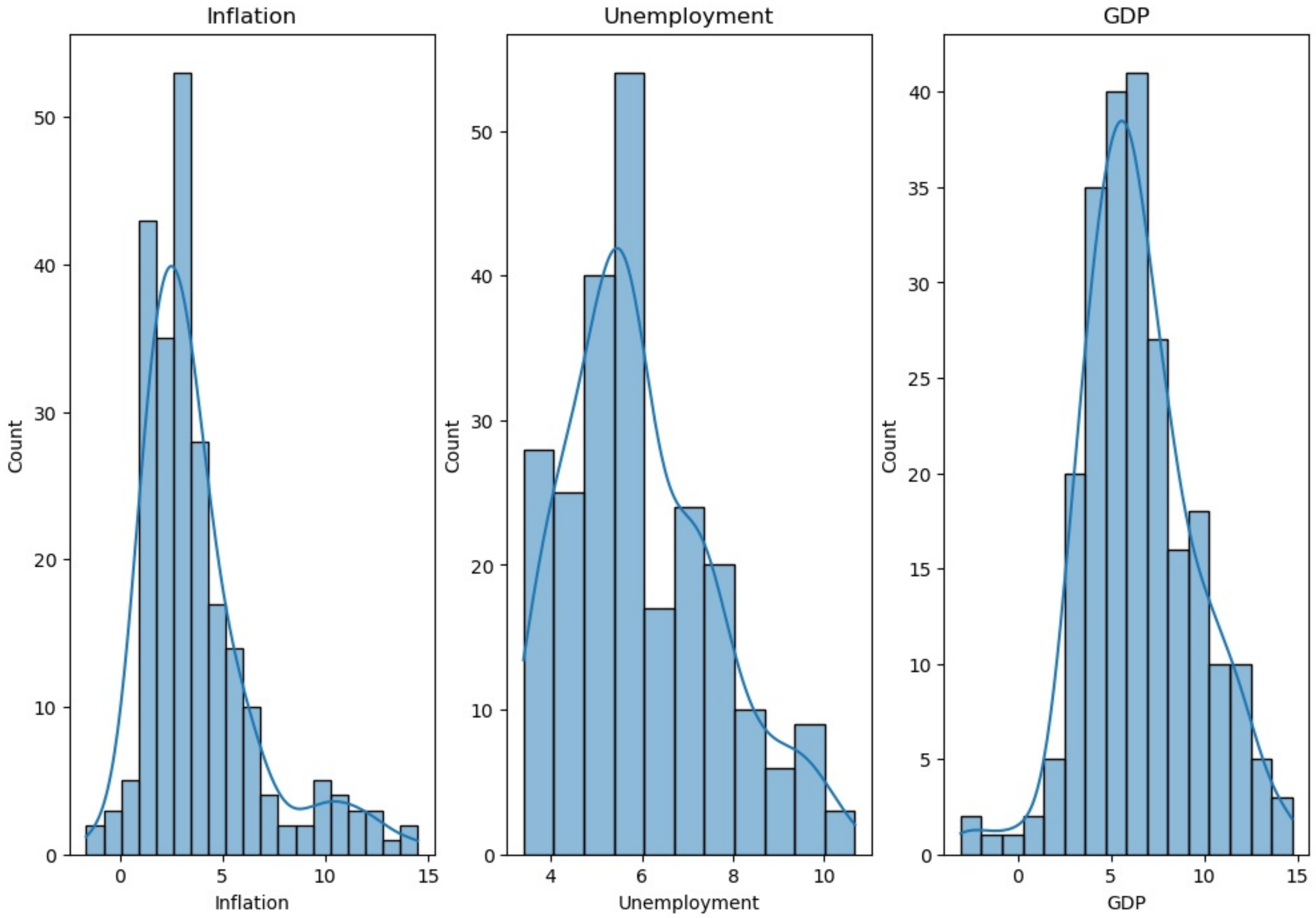
```
Unemployment    2.618608
Inflation       8.029249
GDP             8.955683
dtype: float64
```

-> Exploratory Analysis

In [40]:

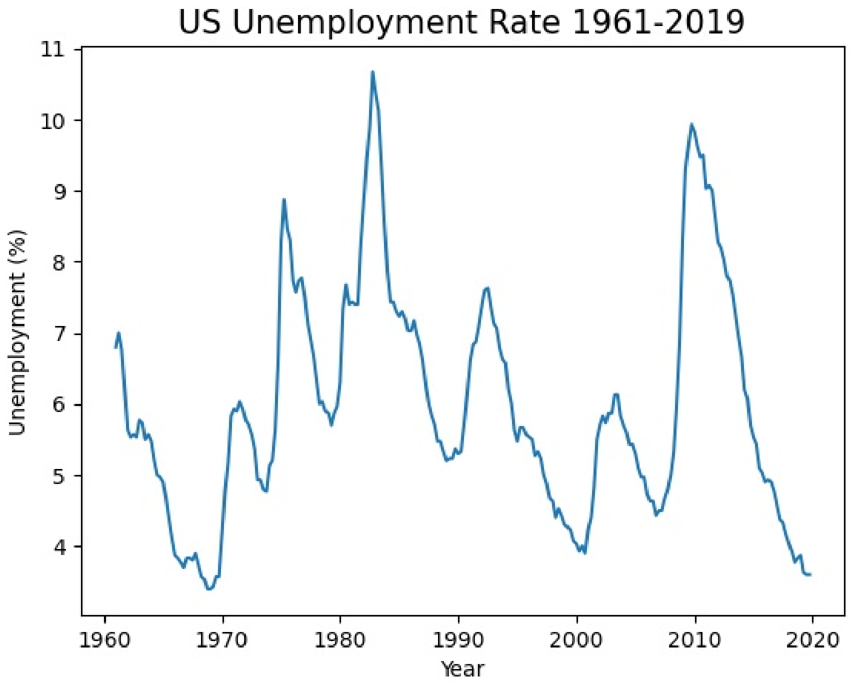
```
plt.figure(figsize=(12, 8))
plt.subplot(1,3,1)
sns.histplot(dfusa["Inflation"], kde=True)
plt.title("Inflation")
plt.subplot(1,3,2)
sns.histplot(dfusa["Unemployment"], kde=True)
plt.title("Unemployment")
plt.subplot(1,3,3)
```

```
sns.histplot(dfusa['GDP'], kde=True)
plt.title("GDP")
plt.show()
```



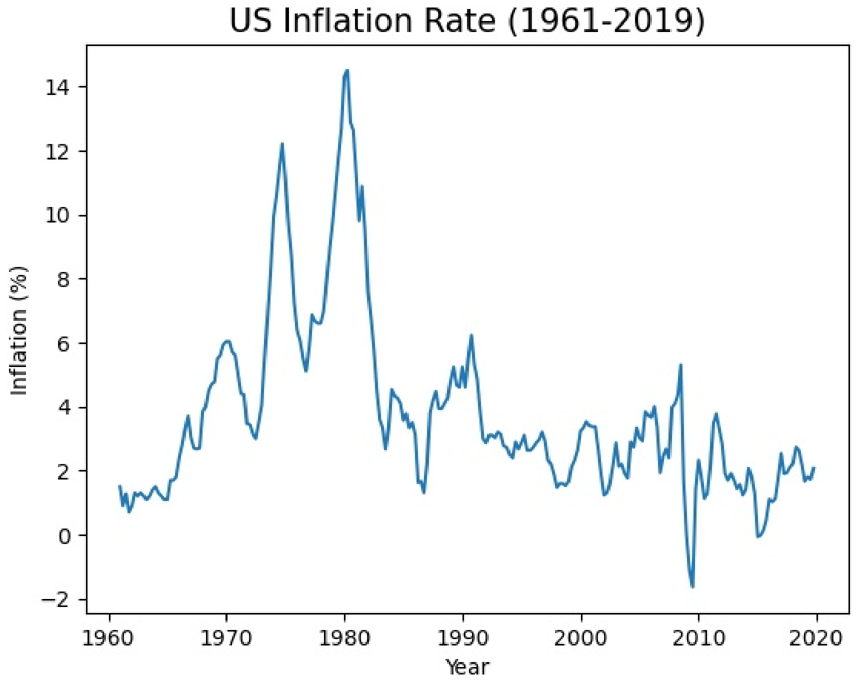
In [41]:

```
plt.plot( dfusa['Date'], dfusa['Unemployment'])
plt.xlabel("Year", fontsize =10)
plt.ylabel("Unemployment (%)", fontsize =10)
plt.title("US Unemployment Rate 1961-2019", fontsize=15);
```



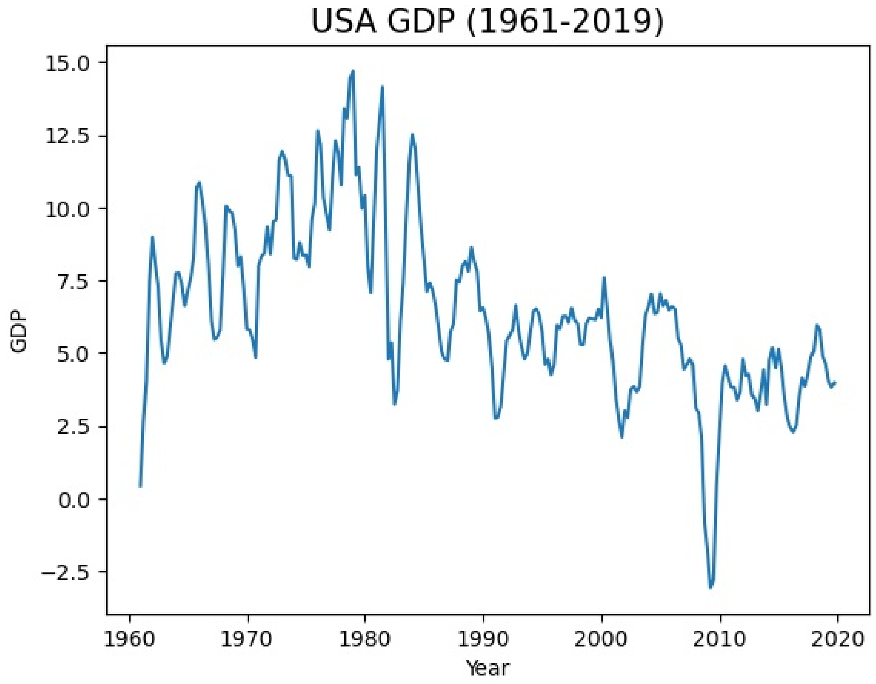
In [42]:

```
plt.plot(dfusa['Date'], dfusa['Inflation'])
plt.xlabel("Year", fontsize =10)
plt.ylabel("Inflation (%)", fontsize =10)
plt.title("US Inflation Rate (1961-2019)", fontsize=15);
```



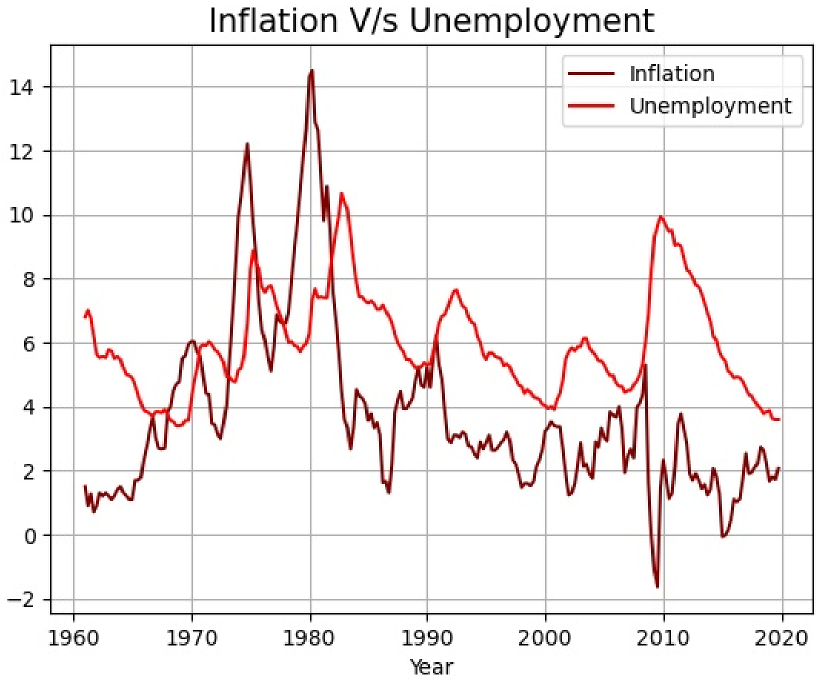
In [43]:

```
plt.plot(dfusa['Date'], dfusa['GDP'])
plt.xlabel("Year", fontsize =10)
plt.ylabel("GDP", fontsize =10)
plt.title("USA GDP (1961-2019)", fontsize=15);
```

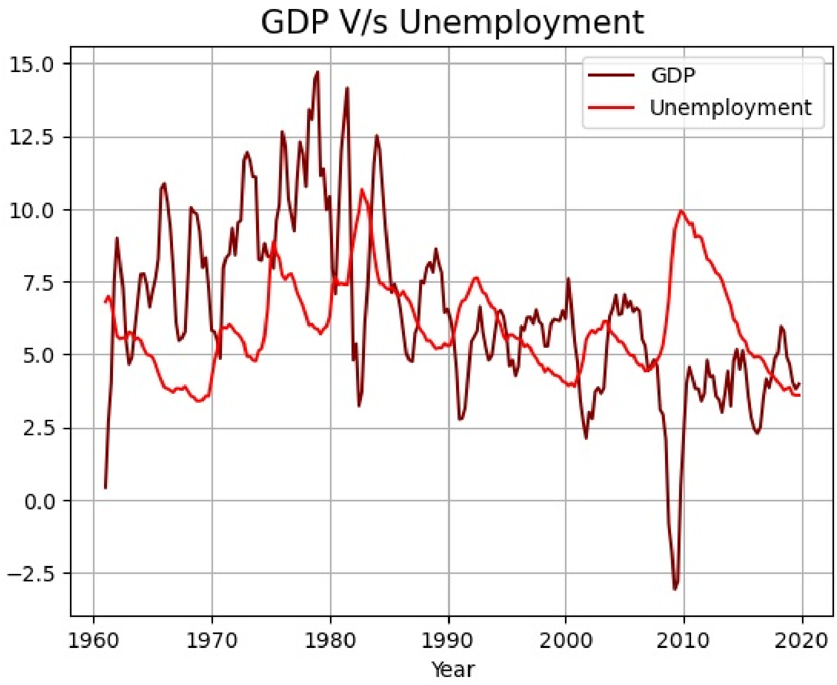


In [44]:

```
plt.plot(dfusa['Date'], dfusa['Inflation'], color='maroon', label='Inflation')
plt.plot(dfusa['Date'], dfusa['Unemployment'], color='r', label='Unemployment')
plt.xlabel('Year', fontsize=10)
plt.title('Inflation V/s Unemployment', fontsize=15)
plt.grid()
plt.legend();
```



In [45]:
```
plt.plot(dfusa['Date'], dfusa['GDP'], color='maroon', label='GDP')
plt.plot(dfusa['Date'], dfusa['Unemployment'], color='r', label='Unemployment')
plt.xlabel('Year', fontsize=10)
plt.title('GDP V/s Unemployment', fontsize=15)
plt.grid()
plt.legend();
```



Linear Regression Model

In [46]:
```
x = dfusa['Unemployment']
x = x.values.reshape(-1,1)
```

In [47]:
```
y = dfusa['Inflation']
y = y.values.reshape(-1,1)
```

In [48]:
```
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.25, random_state=42)
```

In [49]:
```
model2=model.fit(x_train,y_train)
model2
```

Out[49]:
```
▾ LinearRegression
LinearRegression()
```

In [50]:
```
model2.coef_
```

```
Out[50]: array([[0.1787626]])
```

```
In [51]: model2.intercept_
```

```
Out[51]: array([2.79801958])
```

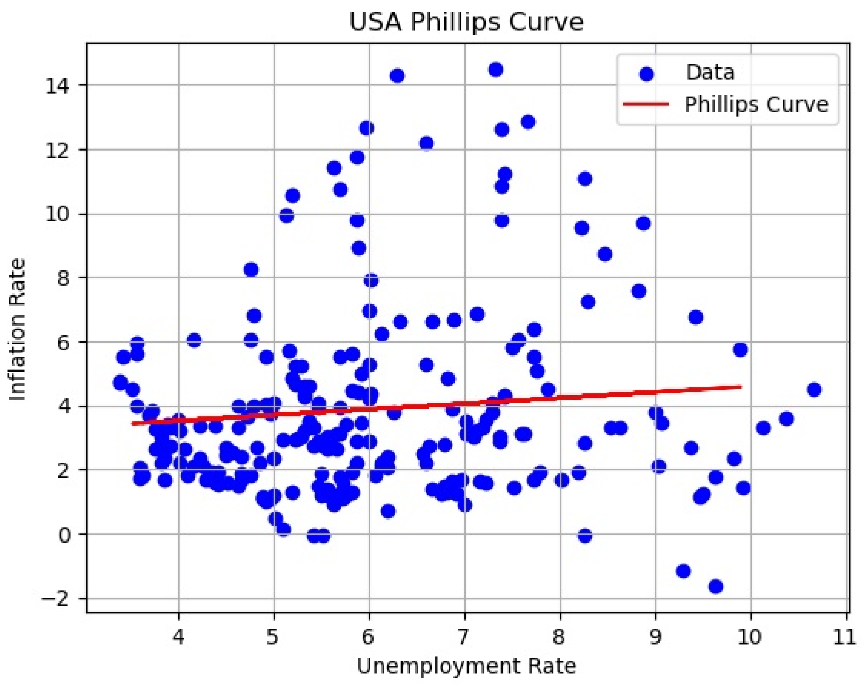
```
In [52]: y_pred = model.predict(x_test)
```

```
In [53]: y_pred
```

```
Out[53]: array([[3.87059517],
                [3.87059517],
                [3.62569041],
                [3.82232927],
                [4.11549993],
                [3.775851  ],
                [3.50055659],
                [4.12622569],
                [3.60781415],
                [3.6864697 ],
                [3.57206163],
                [3.97785273],
                [4.27638627],
                [3.42905155],
                [3.67395631],
                [3.62569041],
                [3.53094624],
                [3.72758509],
                [3.54345962],
                [3.48268033],
                [3.57921214],
                [3.78121387],
                [4.10298655],
                [3.51843285],
                [4.27638627],
                [3.48268033],
                [4.56776931],
                [3.60245128],
                [4.37649333],
                [3.57921214],
                [3.63284092],
                [4.03148151],
                [3.81696639],
                [3.78657675],
                [4.15661533],
                [3.84020553],
                [3.84020553],
                [3.67395631],
                [4.03684439],
                [3.6864697 ],
                [3.66144293],
                [3.58457502],
                [4.11549993],
                [4.17985447],
                [4.41224584],
                [3.48268033],
                [4.12086281],
                [3.75082423],
                [3.66859344],
                [3.81696639],
                [3.49519372],
                [3.44692781],
                [3.62569041],
                [3.67395631],
                [3.81696639],
                [4.10298655],
                [4.49090139],
                [3.81696639],
                [3.72222222]])
```

Phillips Curve

```
In [54]: plt.scatter(x, y, color='blue', label='Data')
         plt.plot(x_test, y_pred, color='red', label='Phillips Curve')
         plt.xlabel('Unemployment Rate')
         plt.ylabel('Inflation Rate')
         plt.title("USA Phillips Curve")
         plt.legend()
         plt.grid()
         plt.show()
```



Cross-Sectional Analysis

->1961-1983

In [55]:

```
start_date='1961-01-01'
end_date='1983-10-01'
m = (dfusa['Date']>start_date)&(dfusa['Date']<= end_date)
olddf = dfusa.loc[m]
olddf
```

Out[55]:

| | Date | Unemployment | Inflation | GDP |
|---|---|---|---|---|
| **1** | 1961-04-01 | 7.00 | 0.90 | 2.67 |
| **2** | 1961-07-01 | 6.77 | 1.27 | 4.04 |
| **3** | 1961-10-01 | 6.20 | 0.70 | 7.48 |
| **4** | 1962-01-01 | 5.63 | 0.90 | 8.99 |
| **5** | 1962-04-01 | 5.53 | 1.30 | 8.07 |
| **...** | ... | ... | ... | ... |
| **87** | 1982-10-01 | 10.67 | 4.50 | 3.71 |
| **88** | 1983-01-01 | 10.37 | 3.60 | 6.08 |
| **89** | 1983-04-01 | 10.13 | 3.33 | 7.41 |
| **90** | 1983-07-01 | 9.37 | 2.67 | 9.59 |
| **91** | 1983-10-01 | 8.53 | 3.33 | 11.53 |

91 rows × 4 columns

In [56]:

```
M = olddf['Unemployment']
M = M.values.reshape(-1,1)
```

In [57]:

```
N = olddf['Inflation']
N = N.values.reshape(-1,1)
```

In [58]:

```
M_train, M_test, N_train, N_test = train_test_split(M, N, test_size=0.25, random_state=42)
```

In [59]:

```
oldmodel=model.fit(M_train,N_train)
oldmodel
```

Out[59]:

```
▾ LinearRegression
LinearRegression()
```

In [60]:

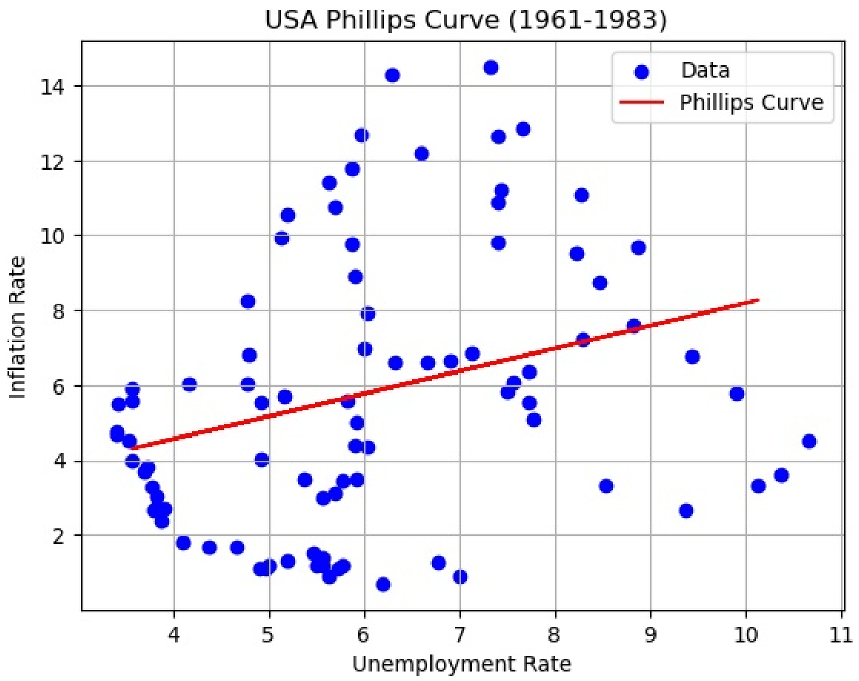
```
N_pred = model.predict(M_test)
```

In [61]:

```
N_pred
```

```
Out[61]: array([[5.70968529],
                [4.37953144],
                [7.14262376],
                [8.26720838],
                [6.37476222],
                [4.50045452],
                [5.72782376],
                [6.17523914],
                [5.51016222],
                [5.58876222],
                [8.12814684],
                [4.66370067],
                [5.70968529],
                [6.84031607],
                [5.28645452],
                [5.4859776 ],
                [4.6213776 ],
                [4.30093144],
                [5.04460837],
                [6.45336222],
                [5.10506991],
                [5.96966991],
                [6.77985453]])
```

```
In [62]: plt.scatter(M, N, color='blue', label='Data')
         plt.plot(M_test, N_pred, color='red', label='Phillips Curve')
         plt.xlabel('Unemployment Rate')
         plt.ylabel('Inflation Rate')
         plt.title(' USA Phillips Curve (1961-1983)')
         plt.legend()
         plt.grid()
         plt.show()
```



-> 1984-2019

```
In [63]: start_date='1984-01-01'
         end_date='2019-10-01'
         mask = (dfusa['Date']>start_date)&(dfusa['Date']<= end_date)
         newdf = dfusa.loc[mask]
         newdf
```

Out[63]:

|  | Date | Unemployment | Inflation | GDP |
|---|---|---|---|---|
| **93** | 1984-04-01 | 7.43 | 4.33 | 12.04 |
| **94** | 1984-07-01 | 7.43 | 4.27 | 10.71 |
| **95** | 1984-10-01 | 7.30 | 4.10 | 9.32 |
| **96** | 1985-01-01 | 7.23 | 3.57 | 8.24 |
| **97** | 1985-04-01 | 7.30 | 3.77 | 7.12 |
| **...** | ... | ... | ... | ... |
| **231** | 2018-10-01 | 3.83 | 2.20 | 4.91 |
| **232** | 2019-01-01 | 3.87 | 1.67 | 4.64 |
| **233** | 2019-04-01 | 3.63 | 1.80 | 4.05 |
| **234** | 2019-07-01 | 3.60 | 1.73 | 3.82 |
| **235** | 2019-10-01 | 3.60 | 2.07 | 3.98 |

143 rows × 4 columns

In [64]:
```
newdf['Inflation'].corr(newdf['Unemployment'])
```

Out[64]:
```
-0.13103919494763133
```

In [65]:
```
X = newdf['Unemployment']
X = X.values.reshape(-1,1)
```

In [66]:
```
Y = newdf['Inflation']
Y = Y.values.reshape(-1,1)
```

In [67]:
```
X_train, X_test, Y_train, Y_test = train_test_split(X, Y, test_size=0.25, random_state=42)
```

In [68]:
```
model3=model.fit(X_train,Y_train)
model3
```

Out[68]:
```
▾ LinearRegression ℹ ?
LinearRegression()
```

In [69]:
```
model3.coef_
```

Out[69]:
```
array([[-0.08237851]])
```

In [70]:
```
model3.intercept_
```

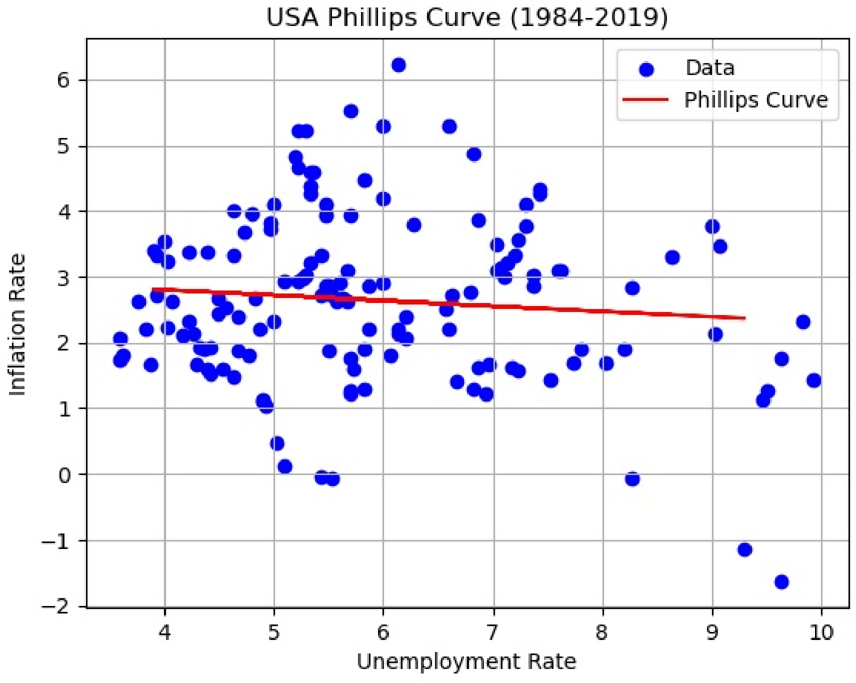
Out[70]:
```
array([3.13069492])
```

In [71]:
```
Y_pred = model.predict(X_test)
```

In [72]:
```
Y_pred
```

```
Out[72]: array([[2.53509831],
                [2.70232668],
                [2.68337963],
                [2.63642388],
                [2.76822949],
                [2.61418168],
                [2.75422514],
                [2.80118089],
                [2.80941874],
                [2.69161748],
                [2.69985533],
                [2.65042822],
                [2.73527809],
                [2.77070084],
                [2.3645748 ],
                [2.80694739],
                [2.58699677],
                [2.73280673],
                [2.71056453],
                [2.65866607],
                [2.58699677],
                [2.58123028],
                [2.4197684 ],
                [2.46919551],
                [2.74928243],
                [2.66360878],
                [2.55651672],
                [2.52933182],
                [2.62571467],
                [2.68337963],
                [2.52356532],
                [2.56475458],
                [2.54827887],
                [2.68008449],
                [2.56804972],
                [2.74598729]])
```

```
In [73]: plt.scatter(X, Y, color='blue', label='Data')
         plt.plot(X_test, Y_pred, color='red', label='Phillips Curve')
         plt.xlabel('Unemployment Rate')
         plt.ylabel('Inflation Rate')
         plt.title('USA Phillips Curve (1984-2019)')
         plt.legend()
         plt.grid()
         plt.show()
```



```
In [74]: r2score2 = r2_score(Y_test, Y_pred)
         r2score2
```

```
Out[74]: 0.023331148994326867
```

```
In [75]: mse2 = mean_squared_error(Y_test, Y_pred)
         mse2
```

```
Out[75]: 2.2650632695620514
```

OKUN's LAW

-> JAPAN

```
In [76]: c = dfjap['Unemployment']
         c = c.values.reshape(-1,1)
```

```
In [77]: d = dfjap['GDP']
         d = d.values.reshape(-1,1)
```

```
In [78]: c_train, c_test, d_train, d_test = train_test_split(c, d, test_size=0.25, random_state=42)
```

```
In [79]: c_train
```

```
Out[79]: array([[2.93],
                [4.73],
                [4.9 ],
                [2.4 ],
                [3.  ],
                [2.  ],
                [3.27],
                [2.3 ],
                [2.17],
                [2.07],
                [1.17],
                [2.2 ],
                [2.17],
                [2.17],
                [1.3 ],
                [1.77],
                [2.1 ],
                [3.33],
                [4.73],
                [2.03],
                [3.03],
                [5.  ],
                [3.33],
                [1.07],
                [1.23],
                [1.07],
                [1.27],
                [1.23],
                [2.57],
                [2.07],
                [4.67],
                [1.13],
                [3.77],
                [3.73],
                [1.9 ],
                [1.2 ],
                [2.77],
                [3.37],
                [2.03],
                [3.17],
                [1.17],
                [4.7 ],
                [4.53],
                [5.13],
                [1.43],
                [1.43],
                [1.97],
                [1.43],
                [2.67],
                [2.17],
                [3.4 ],
                [5.33],
                [2.7 ],
                [2.33],
                [5.27],
                [4.63],
                [2.6 ],
                [1.07],
                [3.07],
                [2.07],
                [1.3 ],
                [2.07],
                [4.6 ],
                [2.87],
                [1.2 ],
                [1.23],
                [2.43],
       [4.03],
       [1.23],
       [1.27],
       [3.33],
       [1.3 ],
       [2.1 ],
       [1.13],
       [5.33],
       [2.  ],
       [2.93],
       [2.3 ],
       [3.67],
       [4.53],
       [2.3 ],
       [2.33],
       [1.97],
       [1.4 ],
       [3.37],
       [1.17],
       [2.07],
       [1.27],
       [5.1 ],
       [5.17],
       [2.27],
       [1.27],
       [5.03],
       [1.4 ],
       [2.73],
       [4.43],
       [1.33],
       [3.33],
       [4.4 ],
       [1.43],
       [3.  ],
       [1.3 ],
       [2.8 ],
       [3.63],
       [4.2 ],
       [3.37],
       [1.37],
       [5.37],
       [2.2 ],
       [1.13],
       [1.33],
       [2.53],
       [2.6 ],
       [2.2 ],
       [4.03],
       [2.9 ],
       [2.67],
       [1.4 ],
       [1.17],
       [2.1 ],
       [4.67],
       [2.73],
       [1.23],
       [5.43],
       [2.1 ],
       [4.23],
       [2.97],
       [4.2 ],
       [3.5 ],
       [1.93],
       [1.4 ],
       [2.7 ],
       [1.27],
       [5.43],
       [4.77],
       [5.2 ],
       [3.97],
       [2.47],
       [4.17],
       [5.43],
       [1.9 ],
       [1.23],
       [2.67],
       [1.33],
       [1.8 ],
       [4.47],
       [4.77],
       [2.47],
       [3.83],
       [4.07],
       [2.43],
       [1.13],
       [4.7 ],
       [3.17],
       [1.47],
       [1.3 ],
       [4.07],
       [2.53],
       [4.4 ],
       [2.8 ],
       [2.77],
       [2.13],
       [2.47],
       [4.47],
       [2.13],
       [3.57],
       [3.93],
       [2.1 ],
       [2.3 ],
       [1.37],
       [3.9 ],
       [2.2 ],
       [2.77],
       [1.17],
       [2.7 ],
       [4.43],
       [2.83]])
```

In [80]:
```
model4 = model.fit(c_train, d_train)
model4
```

Out[80]:
```
▾ LinearRegression ⓘ ?
LinearRegression()
```

In [81]:
```
model4.coef_
```

Out[81]:
```
array([[-2.18108901]])
```

In [82]:
```
model4.intercept_
```

Out[82]:
```
array([9.92076638])
```

In [83]:
```
d_pred = model.predict(c_test)
```

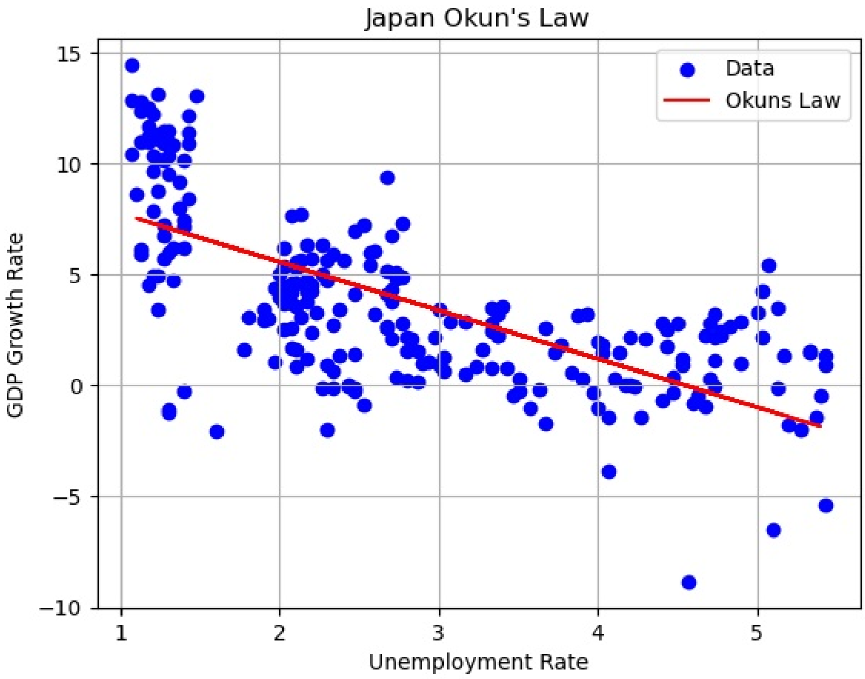
In [84]:
```
d_pred
```

Out[84]:

```
array([[ 4.96969433],
       [ 1.19641034],
       [ 0.91286877],
       [ 7.30345957],
       [ 4.96969433],
       [ 4.53347653],
       [ 4.75158543],
       [ 3.96639338],
       [ 0.60751631],
       [ 7.52156847],
       [ 3.81371715],
       [ 6.43102397],
       [-0.0468104 ],
       [ 7.30345957],
       [ 3.31206668],
       [ 1.91616971],
       [ 7.08535067],
       [ 4.83882899],
       [ 4.03182605],
       [ 7.08535067],
       [ 3.66104092],
       [ 7.30345957],
       [ 4.09725872],
       [-0.61389354],
       [ 0.10586583],
       [ 7.15078334],
       [ 4.75158543],
       [ 1.19641034],
       [ 5.05693789],
       [ 7.15078334],
       [ 2.28695485],
       [ 5.49315569],
       [ 6.86724177],
       [ 2.57049642],
       [ 5.34047946],
       [-1.05011134],
       [-1.85711428],
       [ 7.4561358 ],
       [ 1.47995191],
       [ 0.54208364],
       [-1.26822024],
       [-0.76656977],
       [ 5.40591213],
       [ 5.49315569],
       [-0.39578464],
       [ 4.53347653],
       [ 5.18780323],
       [ 5.27504679],
       [ 2.43963108],
       [ 2.35238752],
       [-0.39578464],
       [ 4.83882899],
       [ 0.97830144],
       [ 2.87584888],
       [ 4.09725872],
       [ 4.31536763],
       [-1.1373549 ],
       [ 5.49315569],
       [ 7.2380269 ]])
```

In [85]:

```
plt.scatter(c, d, color='blue', label='Data')
plt.plot(c_test, d_pred, color='red', label='Okuns Law')
plt.xlabel('Unemployment Rate')
plt.ylabel('GDP Growth Rate')
plt.title("Japan Okun's Law")
plt.legend()
plt.grid()
plt.show()
```



In [86]:
```
r2score3=r2_score(d_test,d_pred)
r2score3
```

Out[86]:
```
0.35869812125440825
```

In [87]:
```
mse3=mean_squared_error(d_test,d_pred)
mse3
```

Out[87]:
```
9.33926912756552
```

-> USA

In [88]:
```
e = dfusa['Unemployment']
e = e.values.reshape(-1,1)
```

In [89]:
```
f = dfusa['GDP']
f = f.values.reshape(-1,1)
```

In [90]:
```
e_train, e_test, f_train, f_test = train_test_split(e, f, test_size=0.25, random_state=42)
```

In [91]:
```
model5 = model.fit(e_train, f_train)
model5
```

Out[91]:
```
▾ LinearRegression
LinearRegression()
```

In [92]:
```
model5.coef_
```

Out[92]:
```
array([[-0.27949369]])
```

In [93]:
```
model5.intercept_
```

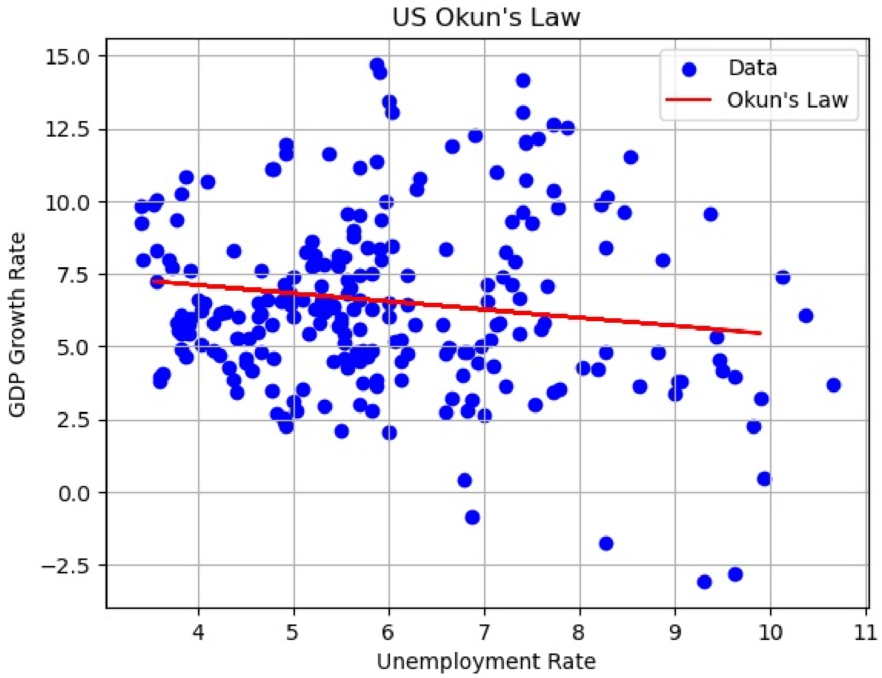
Out[93]:
```
array([8.23035137])
```

In [94]:
```
f_pred = model.predict(e_test)
```

In [95]:
```
f_pred
```

```
Out[95]: array([[6.55338923],
                [6.55338923],
                [6.93629559],
                [6.62885253],
                [6.17048288],
                [6.70152089],
                [7.13194117],
                [6.15371325],
                [6.96424496],
                [6.84126773],
                [7.02014369],
                [6.38569302],
                [5.91893855],
                [7.24373865],
                [6.86083229],
                [6.93629559],
                [7.08442724],
                [6.77698418],
                [7.06486268],
                [7.15989054],
                [7.00896395],
                [6.69313608],
                [6.19004743],
                [7.1039918 ],
                [5.91893855],
                [7.15989054],
                [5.46336384],
                [6.97262977],
                [5.76242209],
                [7.00896395],
                [6.92511584],
                [6.30184491],
                [6.63723734],
                [6.68475127],
                [6.10619933],
                [6.60090316],
                [6.60090316],
                [6.86083229],
                [6.2934601 ],
                [6.84126773],
                [6.88039685],
                [7.00057914],
                [6.17048288],
                [6.06986515],
                [5.70652335],
                [7.15989054],
                [6.16209806],
                [6.74065   ],
                [6.8692171 ],
                [6.63723734],
                [7.14032598],
                [7.21578928],
                [6.93629559],
                [6.86083229],
                [6.63723734],
                [6.19004743],
                [5.58354613],
                [6.63723734],
                [6.78536899]])
```

```
In [96]: plt.scatter(e, f, color='blue', label='Data')
         plt.plot(e_test, f_pred, color='red', label="Okun's Law")
         plt.xlabel('Unemployment Rate')
         plt.ylabel('GDP Growth Rate')
         plt.title("US Okun's Law")
         plt.legend()
         plt.grid()
         plt.show()
```



In [97]:

```
r2score4=r2_score(f_test,f_pred)
r2score4
```

Out[97]:

```
-0.04496610388241695
```

In [98]:

```
mse4=mean_squared_error(f_test,f_pred)
mse4
```

Out[98]:

```
9.284116713480742
```

Loading [MathJax]/jax/output/CommonHTML/fonts/TeX/fontdata.js