In [ ]:
```
#1. Importe la base de datos a una base en Jupyter Notebook con pandas.
```

In [2]:
```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import numpy as np
```

In [8]:
```
base= "C:/Users/i7/Downloads/Walmart.csv"
data = pd.read_csv(base, sep=',')
```

In [9]:
```
data.head()
```

Out[9]:

	Store	Date	Weekly_Sales	Holiday_Flag	Temperature	Fuel_Price	CPI	Unemployment
0	1	05-02-2010	1643690.90	0	42.31	2.572	211.096358	8.106
1	1	12-02-2010	1641957.44	1	38.51	2.548	211.242170	8.106
2	1	19-02-2010	1611968.17	0	39.93	2.514	211.289143	8.106
3	1	26-02-2010	1409727.59	0	46.63	2.561	211.319643	8.106
4	1	05-03-2010	1554806.68	0	46.50	2.625	211.350143	8.106

In [11]:
```
data
```

Out[11]:

	Store	Date	Weekly_Sales	Holiday_Flag	Temperature	Fuel_Price	CPI	Unemployment
0	1	05-02-2010	1643690.90	0	42.31	2.572	211.096358	8.106
1	1	12-02-2010	1641957.44	1	38.51	2.548	211.242170	8.106
2	1	19-02-2010	1611968.17	0	39.93	2.514	211.289143	8.106
3	1	26-02-2010	1409727.59	0	46.63	2.561	211.319643	8.106
4	1	05-03-2010	1554806.68	0	46.50	2.625	211.350143	8.106
...	...	...	...	...	...	...	...	...
6430	45	28-09-2012	713173.95	0	64.88	3.997	192.013558	8.684
6431	45	05-10-2012	733455.07	0	64.89	3.985	192.170412	8.667
6432	45	12-10-2012	734464.36	0	54.47	4.000	192.327265	8.667
6433	45	19-10-2012	718125.53	0	56.47	3.969	192.330854	8.667
6434	45	26-10-2012	760281.43	0	58.85	3.882	192.308899	8.667

6435 rows × 8 columns

In [ ]:

```
#2. Obtenga los descriptivos resumen de la base de datos e identifique a las variables numéricas y categóricas.
#¿Hay algo que le llame la atención?
```

In [10]:

```
print(data.dtypes)
```

```
Store             int64
Date             object
Weekly_Sales    float64
Holiday_Flag      int64
Temperature     float64
Fuel_Price      float64
CPI             float64
Unemployment    float64
dtype: object
```

In [ ]:

```
#la variable date se encuentra como object se puede ajustar como fecha
```

In [15]:

```
data['Date'] = pd.to_datetime(data['Date'], format='%m/%d/%Y')
```

```
# Ahora la columna "Date" debería estar en formato de fecha
print(data.dtypes)
```

```
Store                    int64
Date            datetime64[ns]
Weekly_Sales           float64
Holiday_Flag             int64
Temperature            float64
Fuel_Price             float64
CPI                    float64
Unemployment           float64
dtype: object
```

In [ ]:
```
#3. Evalúe si la base contiene datos perdidos
```

In [17]:
```
data_perdidos = data.isnull().sum()
```

In [18]:
```
print(data_perdidos)
```

```
Store           0
Date            0
Weekly_Sales    0
Holiday_Flag    0
Temperature     0
Fuel_Price      0
CPI             0
Unemployment    0
dtype: int64
```

In [19]:
```
total_cells = data.size
total_missing = data.isnull().sum().sum()
missing_percentage = (total_missing / total_cells) * 100
print(f"Porcentaje de valores perdidos: {missing_percentage:.2f}%")
```

```
Porcentaje de valores perdidos: 0.00%
```

In [ ]:
```
#No se registran datos perdidos
```

In [ ]:
```
#4. Evalúe si alguna de las variables contiene datos atípicos (outliers)
```

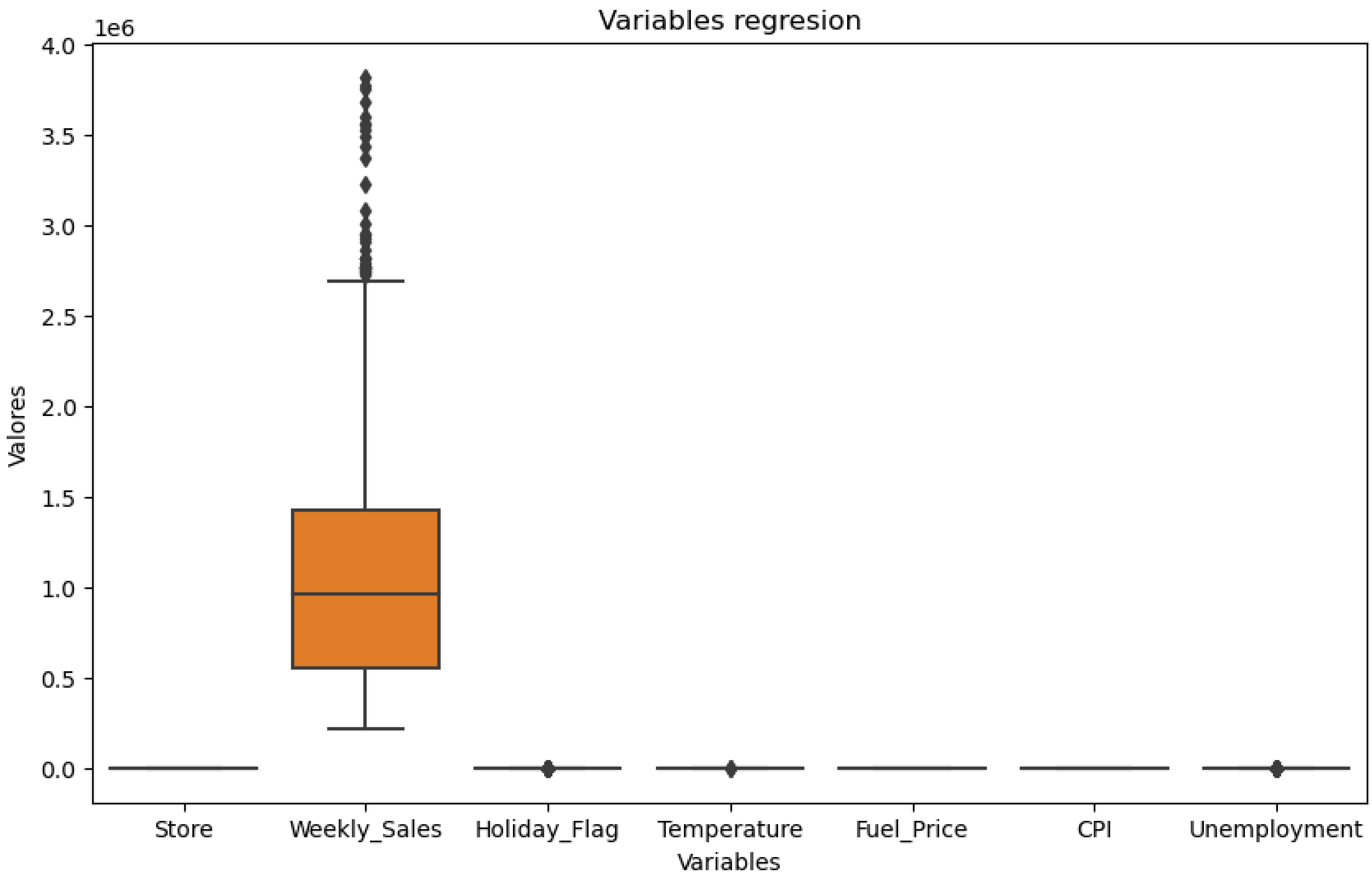
In [20]:
```
data.describe()
```

Out[20]:

	Store	Weekly_Sales	Holiday_Flag	Temperature	Fuel_Price	CPI	Unemployment
count	6435.000000	6.435000e+03	6435.000000	6435.000000	6435.000000	6435.000000	6435.000000
mean	23.000000	1.046965e+06	0.069930	60.663782	3.358607	171.578394	7.999151
std	12.988182	5.643666e+05	0.255049	18.444933	0.459020	39.356712	1.875885
min	1.000000	2.099862e+05	0.000000	-2.060000	2.472000	126.064000	3.879000
25%	12.000000	5.533501e+05	0.000000	47.460000	2.933000	131.735000	6.891000
50%	23.000000	9.607460e+05	0.000000	62.670000	3.445000	182.616521	7.874000
75%	34.000000	1.420159e+06	0.000000	74.940000	3.735000	212.743293	8.622000
max	45.000000	3.818686e+06	1.000000	100.140000	4.468000	227.232807	14.313000

In [21]:

```
plt.figure(figsize=(10, 6))
sns.boxplot(data=data)
plt.title('Variables regresion')
plt.xlabel('Variables')
plt.ylabel('Valores')
plt.show()
```

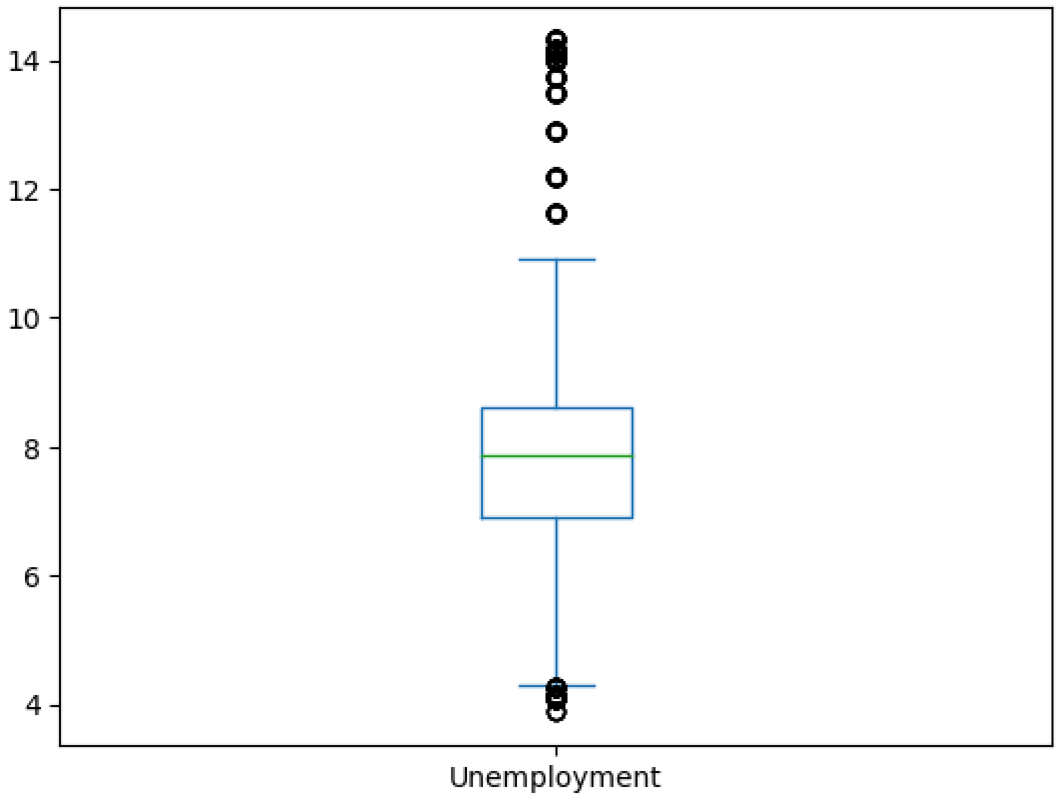


In [30]:

```
data['Unemployment'].plot.box()
```

Out[30]:

```
<Axes: >
```

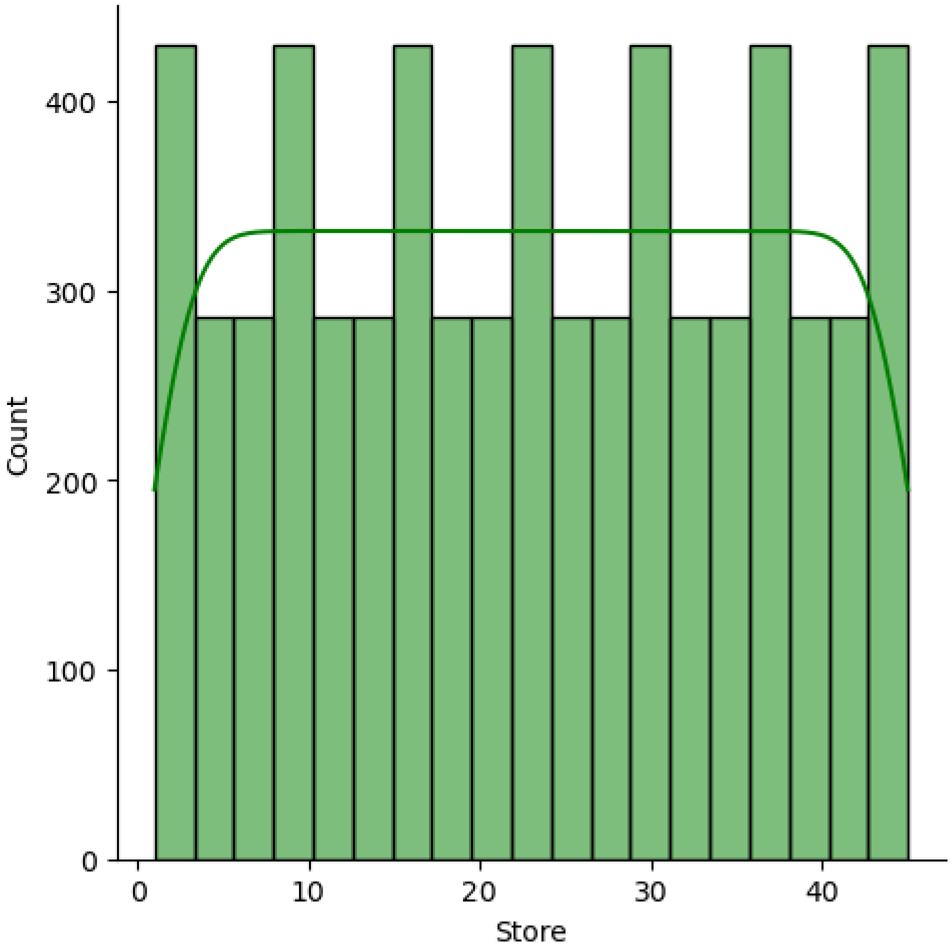


In [ ]:

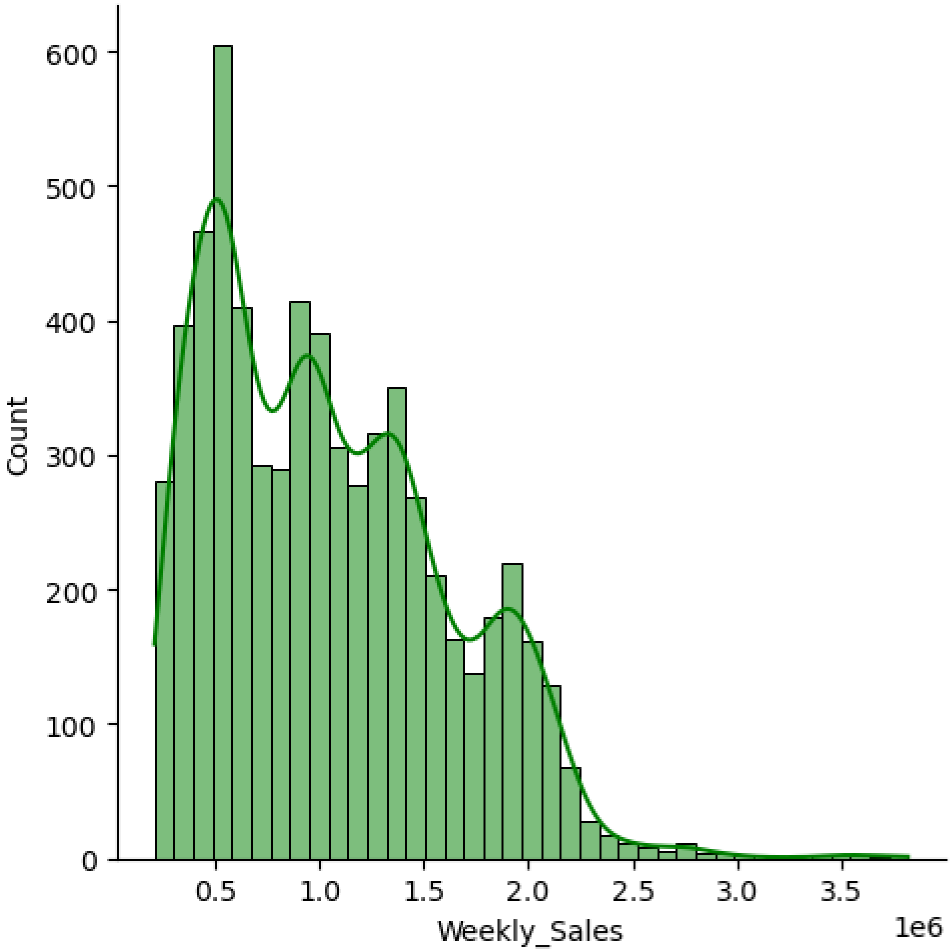
```
#5.Grafique las distribuciones de las variables y a priori comente sobre ellas.
```

In [33]:

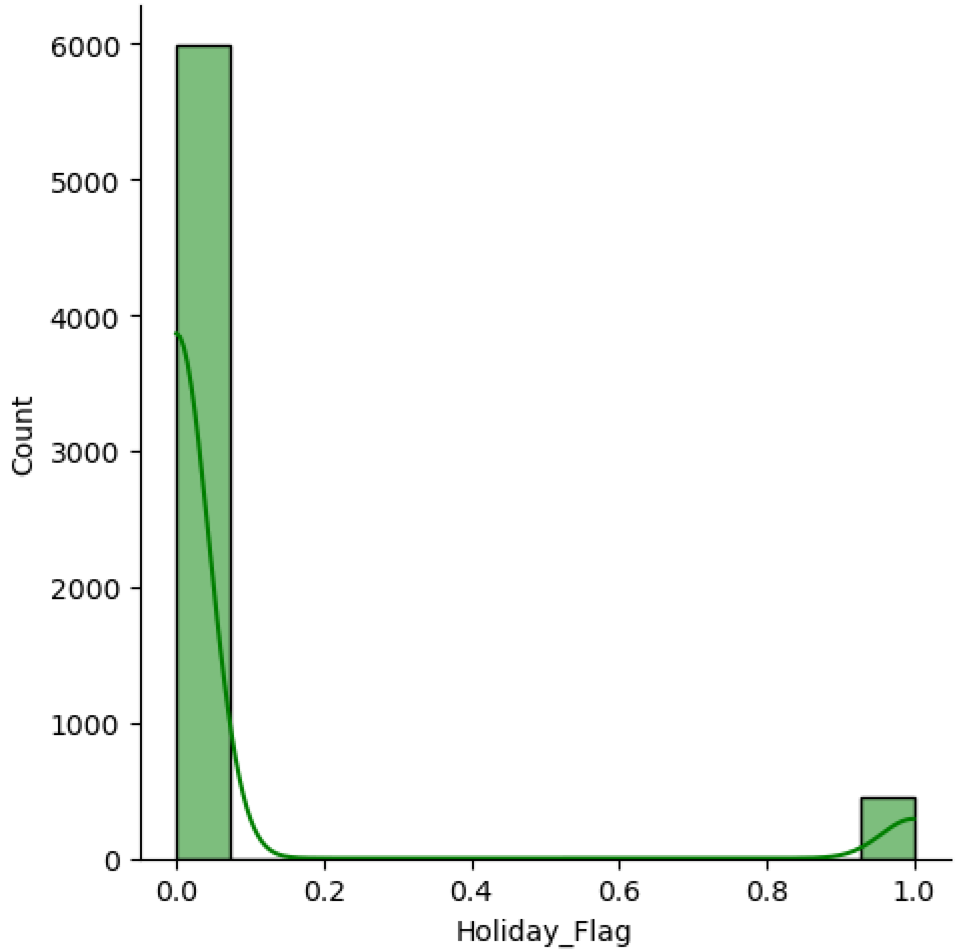
```
import seaborn as sns
import matplotlib.pyplot as plt

sns.displot(data['Store'], color="green", kde=True)

plt.show()
```



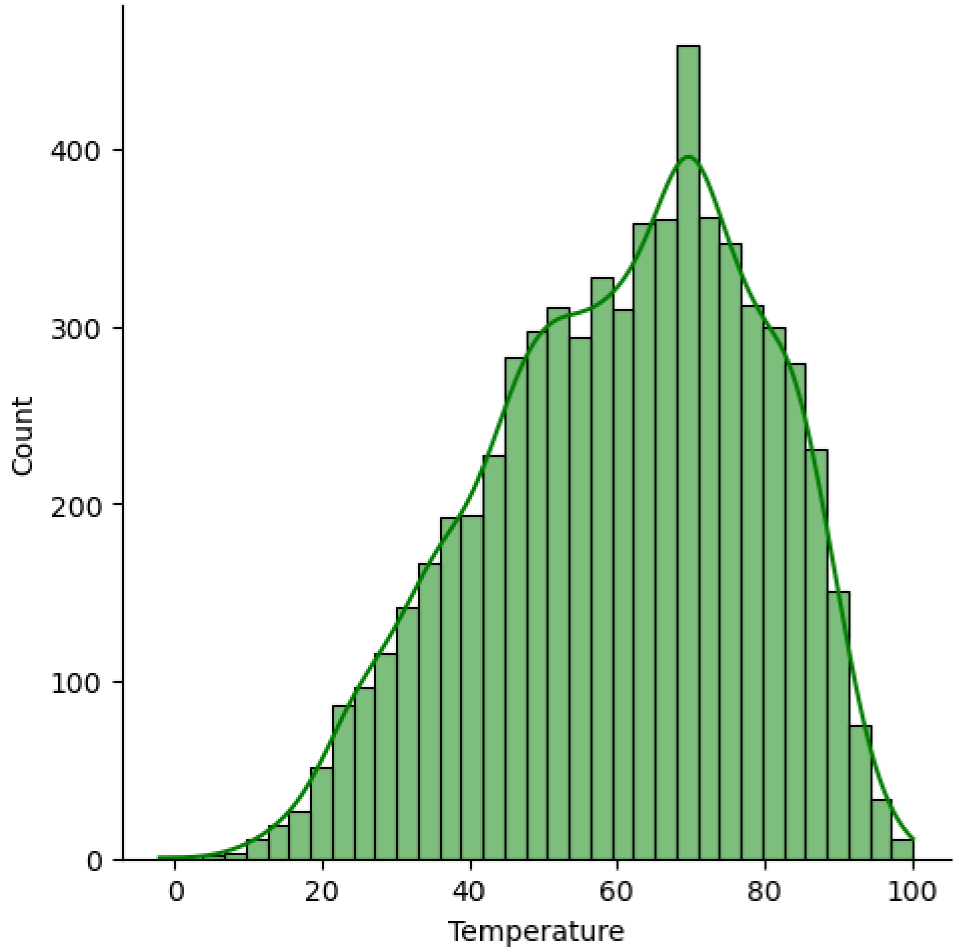
In [35]:
```
sns.displot(data['Weekly_Sales'], color="green", kde=True)

plt.show()
```



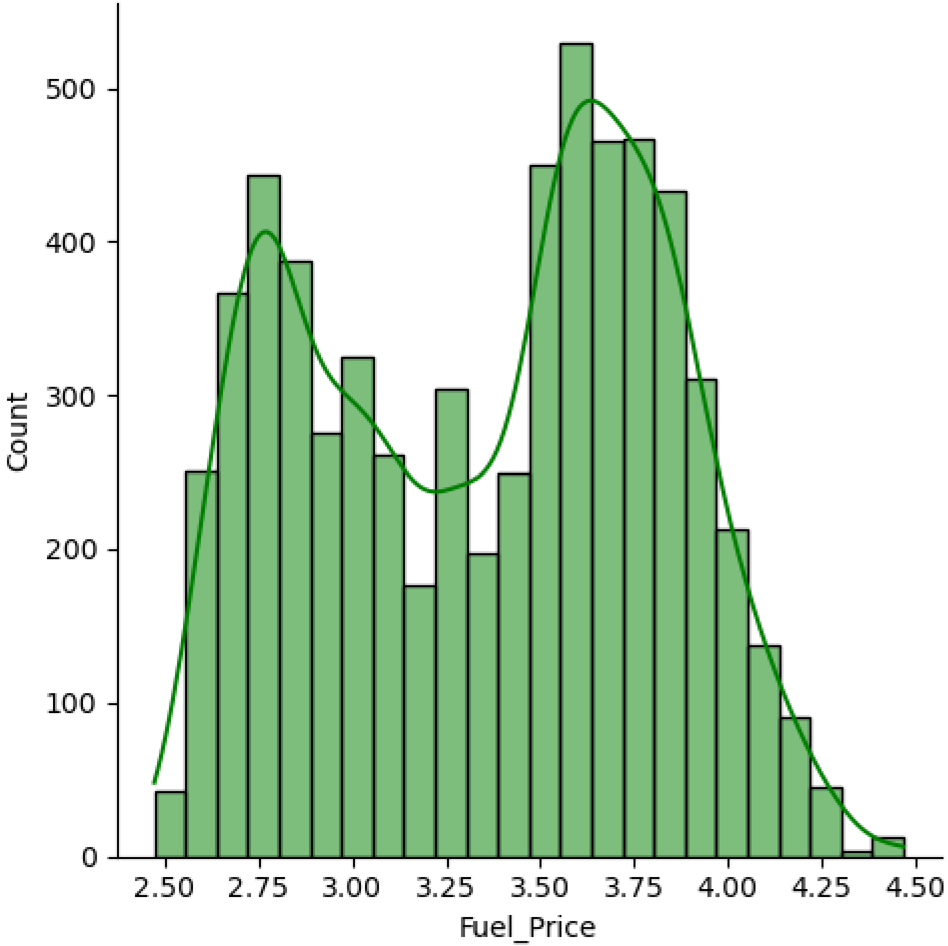
In [36]:
```
sns.displot(data['Holiday_Flag'], color="green", kde=True)

plt.show()
```



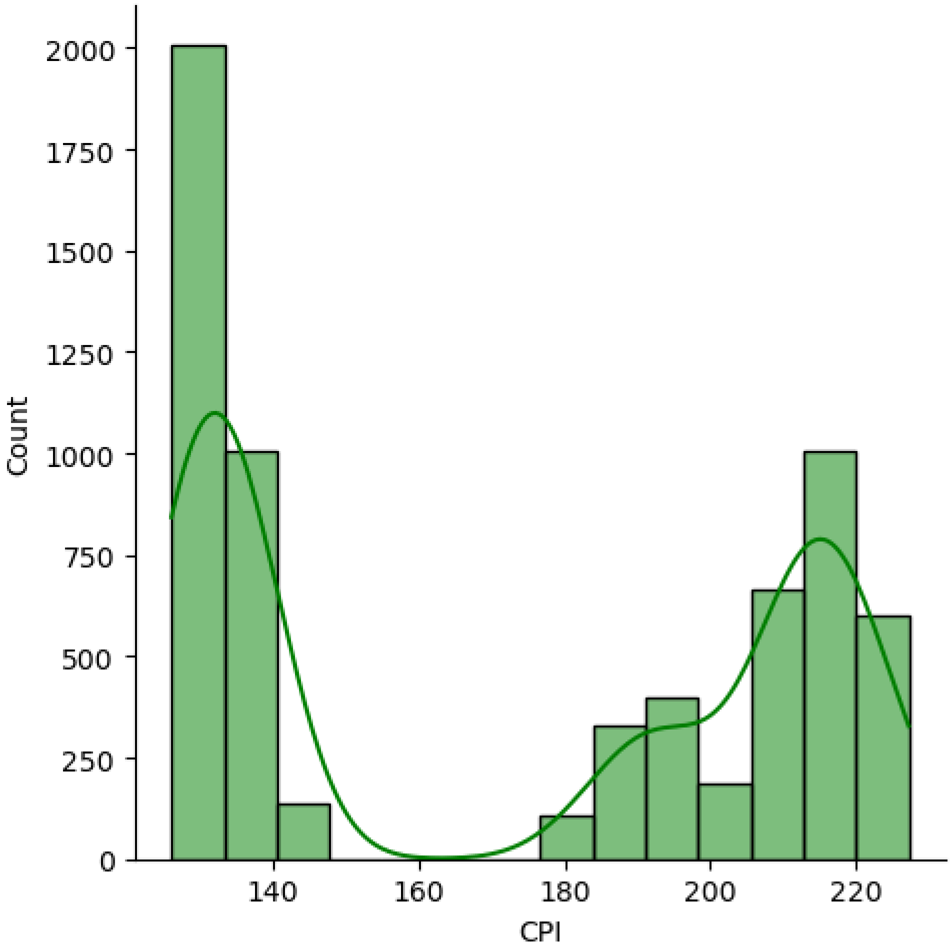
In [38]:
```
sns.displot(data['Temperature'], color="green", kde=True)

plt.show()
```



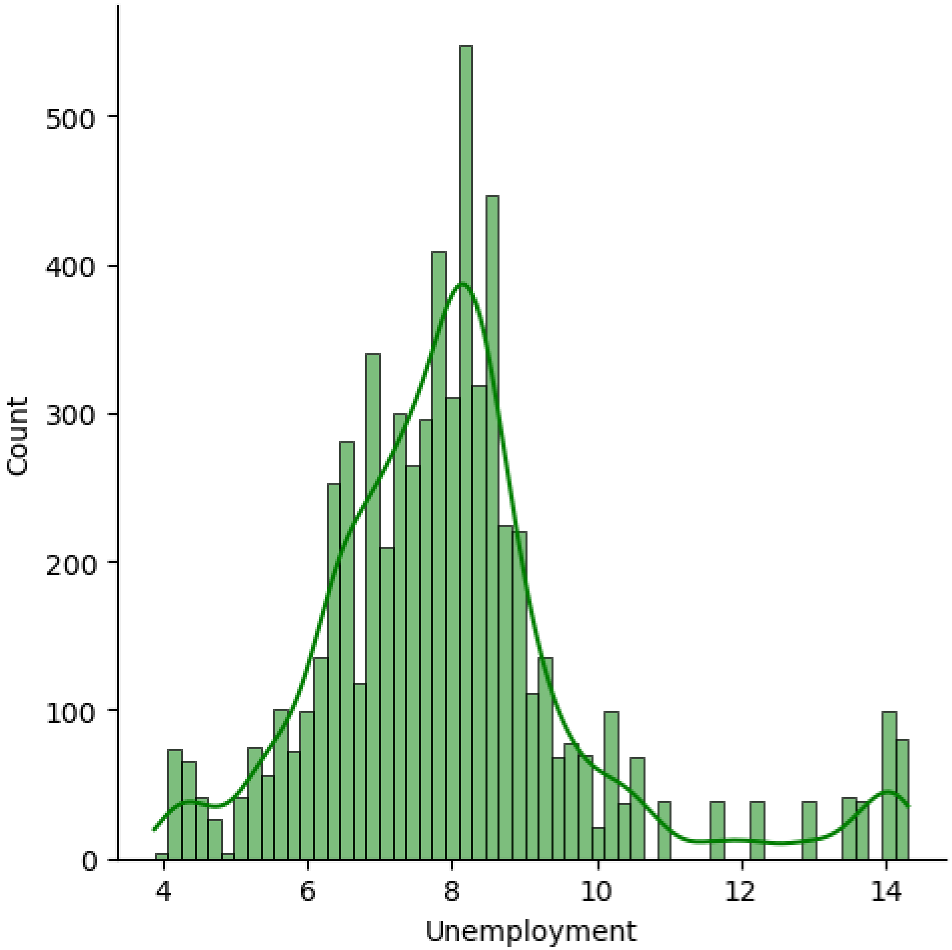
In [39]:
```
sns.displot(data['Fuel_Price'], color="green", kde=True)

plt.show()
```



In [40]:
```
sns.displot(data['CPI'], color="green", kde=True)

plt.show()
```

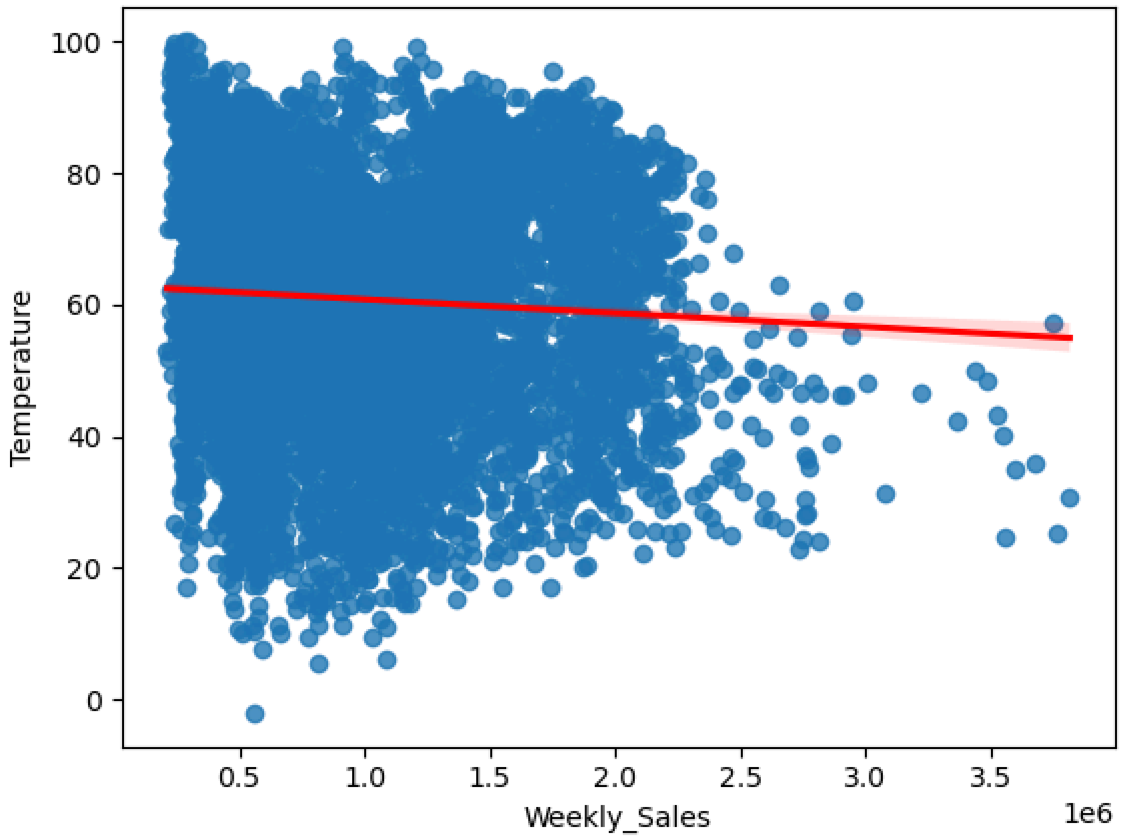


In [41]:
```
sns.displot(data['Unemployment'], color="green", kde=True)

plt.show()
```



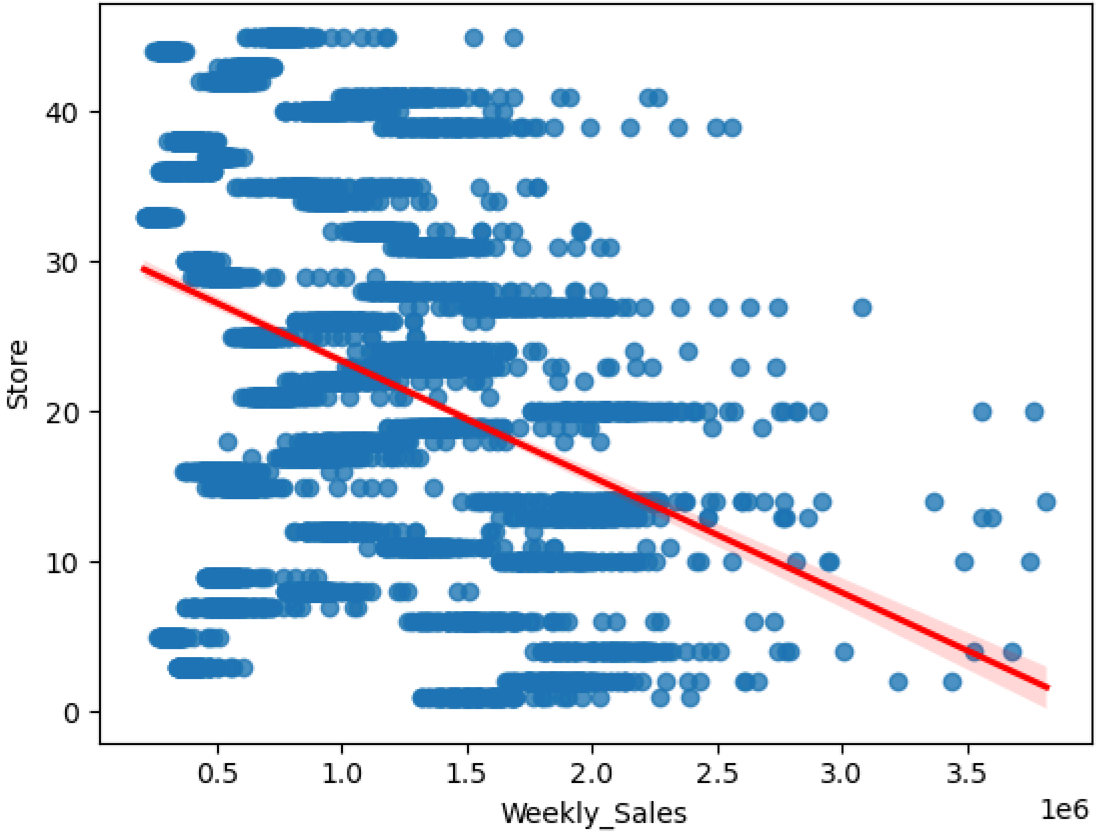
In [42]:
```
sns.regplot(y='Temperature',x='Weekly_Sales',data=data,line_kws={'color':'red'})
```

Out[42]:
```
<Axes: xlabel='Weekly_Sales', ylabel='Temperature'>
```



In [48]:
```
sns.regplot(y='Store',x='Weekly_Sales',data=data,line_kws={'color':'red'})
```

Out[48]:
```
<Axes: xlabel='Weekly_Sales', ylabel='Store'>
```



In [50]:
```
data.corr(numeric_only=True).style.background_gradient(cmap='coolwarm')
```

Out[50]:

	Store	Weekly_Sales	Holiday_Flag	Temperature	Fuel_Price	CPI	Unemployment
Store	1.000000	-0.335332	-0.000000	-0.022659	0.060023	-0.209492	0.223531
Weekly_Sales	-0.335332	1.000000	0.036891	-0.063810	0.009464	-0.072634	-0.106176
Holiday_Flag	-0.000000	0.036891	1.000000	-0.155091	-0.078347	-0.002162	0.010960
Temperature	-0.022659	-0.063810	-0.155091	1.000000	0.144982	0.176888	0.101158
Fuel_Price	0.060023	0.009464	-0.078347	0.144982	1.000000	-0.170642	-0.034684
CPI	-0.209492	-0.072634	-0.002162	0.176888	-0.170642	1.000000	-0.302020
Unemployment	0.223531	-0.106176	0.010960	0.101158	-0.034684	-0.302020	1.000000

In [54]:
```python
import statsmodels.api as sm
from statsmodels.formula.api import ols
regression = ols("Weekly_Sales ~ Store + Holiday_Flag + Temperature + Fuel_Price + CPI + Unemployment", data=data)
# Ajustar el modelo a los datos
results = regression.fit()
# Imprimir el resumen del modelo
print(results.summary())
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:           Weekly_Sales   R-squared:                       0.142
Model:                            OLS   Adj. R-squared:                  0.141
Method:                 Least Squares   F-statistic:                     176.7
Date:                Tue, 19 Mar 2024   Prob (F-statistic):          9.33e-209
Time:                        23:33:11   Log-Likelihood:                -93861.
No. Observations:                6435   AIC:                         1.877e+05
Df Residuals:                    6428   BIC:                         1.878e+05
Df Model:                           6
Covariance Type:            nonrobust
================================================================================
                   coef    std err          t      P>|t|      [0.025      0.975]
--------------------------------------------------------------------------------
Intercept     1.996e+06   7.54e+04     26.461      0.000    1.85e+06    2.14e+06
Store        -1.539e+04    521.895    -29.486      0.000   -1.64e+04   -1.44e+04
Holiday_Flag  7.303e+04   2.59e+04      2.815      0.005    2.22e+04    1.24e+05
Temperature   -975.4019    375.974     -2.594      0.009   -1712.436    -238.367
Fuel_Price    9596.0739   1.48e+04      0.648      0.517   -1.94e+04    3.86e+04
CPI          -2319.4558    184.772    -12.553      0.000   -2681.670   -1957.241
Unemployment -2.188e+04   3788.000     -5.776      0.000   -2.93e+04   -1.45e+04
==============================================================================
Omnibus:                      188.961   Durbin-Watson:                   0.130
Prob(Omnibus):                  0.000   Jarque-Bera (JB):              205.250
Skew:                           0.435   Prob(JB):                     2.69e-45
Kurtosis:                       3.100   Cond. No.                     2.19e+03
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The condition number is large, 2.19e+03. This might indicate that there are
strong multicollinearity or other numerical problems.
```

In [55]:
```python
import statsmodels.api as sm
from statsmodels.formula.api import ols
```

```
regression = ols("Weekly_Sales ~ Store + Holiday_Flag + Temperature + CPI + Unemployment", data=data)
# Ajustar el modelo a los datos
results = regression.fit()
# Imprimir el resumen del modelo
print(results.summary())
```

```
                            OLS Regression Results
==============================================================================
Dep. Variable:           Weekly_Sales   R-squared:                       0.141
Model:                            OLS   Adj. R-squared:                  0.141
Method:                 Least Squares   F-statistic:                     211.9
Date:                Tue, 19 Mar 2024   Prob (F-statistic):          7.51e-210
Time:                        23:34:24   Log-Likelihood:                -93861.
No. Observations:                6435   AIC:                         1.877e+05
Df Residuals:                    6429   BIC:                         1.878e+05
Df Model:                           5
Covariance Type:            nonrobust
================================================================================
                   coef    std err          t      P>|t|      [0.025      0.975]
--------------------------------------------------------------------------------
Intercept     2.032e+06   5.07e+04     40.114      0.000    1.93e+06    2.13e+06
Store        -1.537e+04    521.337    -29.488      0.000   -1.64e+04   -1.44e+04
Holiday_Flag  7.222e+04   2.59e+04      2.787      0.005    2.14e+04    1.23e+05
Temperature   -929.0252    369.081     -2.517      0.012   -1652.547    -205.503
CPI          -2345.9264    180.191    -13.019      0.000   -2699.160   -1992.693
Unemployment  -2.22e+04   3755.948     -5.910      0.000   -2.96e+04   -1.48e+04
==============================================================================
Omnibus:                      188.685   Durbin-Watson:                   0.130
Prob(Omnibus):                  0.000   Jarque-Bera (JB):              204.924
Skew:                           0.434   Prob(JB):                     3.17e-45
Kurtosis:                       3.100   Cond. No.                     1.46e+03
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The condition number is large, 1.46e+03. This might indicate that there are
strong multicollinearity or other numerical problems.
```

In [56]:

```
import statsmodels.api as sm
from statsmodels.formula.api import ols
regression = ols("Weekly_Sales ~ Store + Holiday_Flag  + CPI + Unemployment", data=data)
# Ajustar el modelo a los datos
results = regression.fit()
```

```python
# Imprimir el resumen del modelo
print(results.summary())
```

```
                            OLS Regression Results                            
==============================================================================
Dep. Variable:           Weekly_Sales   R-squared:                       0.141
Model:                            OLS   Adj. R-squared:                  0.140
Method:                 Least Squares   F-statistic:                     263.1
Date:                Tue, 19 Mar 2024   Prob (F-statistic):         1.02e-209
Time:                        23:35:12   Log-Likelihood:                -93864.
No. Observations:                6435   AIC:                         1.877e+05
Df Residuals:                    6430   BIC:                         1.878e+05
Df Model:                           4                                         
Covariance Type:            nonrobust                                         
================================================================================
                   coef    std err          t      P>|t|      [0.025      0.975]
--------------------------------------------------------------------------------
Intercept     2.004e+06   4.94e+04     40.528      0.000    1.91e+06     2.1e+06
Store        -1.535e+04    521.499    -29.443      0.000   -1.64e+04   -1.43e+04
Holiday_Flag  8.273e+04   2.56e+04      3.234      0.001    3.26e+04    1.33e+05
CPI          -2444.4274    175.963    -13.892      0.000   -2789.374   -2099.481
Unemployment -2.379e+04   3703.784     -6.424      0.000   -3.11e+04   -1.65e+04
==============================================================================
Omnibus:                      198.096   Durbin-Watson:                   0.130
Prob(Omnibus):                  0.000   Jarque-Bera (JB):              216.005
Skew:                           0.442   Prob(JB):                     1.24e-47
Kurtosis:                       3.150   Cond. No.                     1.35e+03
==============================================================================

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The condition number is large, 1.35e+03. This might indicate that there are
strong multicollinearity or other numerical problems.
```

In [58]:
```python
from statsmodels.stats.outliers_influence import variance_inflation_factor
```

In [59]:
```python
results.predict()
```

Out[59]:
```
array([1279838.7887136 , 1362216.26679386, 1279367.54050364, ...,
        636770.34491274,  636761.57235155,  636815.24048924])
```

In [ ]:
```python
#Se toma como referencia el enfoque econométrico, tomando los elementos de p Value < 0.05 a fin de validar la significancia
#de las variables.
```